SBBD 2014

Simpósio Brasileiro de Banco de Dados

06 a 09 de outubro de 2014

Curitiba - Centro de Convenções do Hotel Pestana



Proceedings of the 29th Brazilian Symposium on Databases.
Mirella Moro, Renata Galante, Carmem S. Hara, Vidal Martins,
José Antônio Macedo, Flávio R. C. Souza, João Eduardo Ferreira,
Fabio Porto, and Vanessa Braganholo (Eds.).
Publisher: Brazilian Computer Society.

Realização

PUCPR GRUPO MARISTA

UFPR 100 Anos 1912 - 2012

UTFPR Universidade Tecnológica Federal do Paraná



29th BRAZILIAN SYMPOSIUM ON DATABASES

October 6th to 9th, 2014
Curitiba – PR – Brazil

PROCEEDINGS

Promotion
Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization
Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

Steering Committee Chair
Cristina Ciferri, ICMC-USP

Local Organization Chairs
Carmem Hara, UFPR and Vidal Martins, PUC-PR

Program Committee Chairs
Mirella M. Moro, UFMG and Renata Galante, UFRGS

ISSN: 2316-5170

S612 Simpósio Brasileiro de Banco de Dados (29. : 2014 out. 6-9 : Curitiba, PR).
[Anais do] 29º Simpósio Brasileiro de Banco de Dados [recurso eletrônico] / [Organização: Departamento de Informática da UFPR e Escola Politécnica da PUC-PR ; editado por Mirella Moro... et al.]. – Curitiba: UFPR; PUC-PR, 2014.
1 pendrive.

Em português e inglês.
Disponível também no site do SBBD 2014: http://www.inf.ufpr.br/sbbd-sbsc2014/sbbd.
ISSN 2316-5170.

1. Bancos de dados - Encontros. 2. Gerência de Informações. 3. SBBD. I. Universidade Federal do Paraná. Departamento de Informática (Org.). II. Pontifícia Universidade Católica do Paraná. Escola Politécnica (Org.). III. Moro, Mirella (Ed.). IV. Título.

CDD: 005.74

Editorial

The Brazilian Symposium on Databases is the official database event of the Brazilian Computer Society (SBC) and the largest venue in Latin America for presentation and discussion of research results in the database domain. The 29th edition of the symposium (SBBD 2014) has been held in Curitiba, the capital of Paraná State, from the 6th to 9th of October.

The SBBD 2014 program offers a variety of activities, suited for an audience ranging from undergraduate to Ph.D. students, database professionals, and researchers. The program includes: technical sessions, short courses, tutorials, invited talks, demo sessions and a thesis and dissertation workshop. Papers presented at technical sessions have been submitted either to the Journal of Information and Data Management (JIDM) track or to the SBBD Proceedings track. JIDM articles present interesting results or novel thought-provoking ideas, and they are published in the October 2014 issue of the journal. SBBD Proceedings papers, on the other hand, present initial results or on-going work. After a thorough review process, a total of 10 high quality JIDM articles, and 18 SBBD Proceeding papers have been selected.

Given the great buzz around big data nowadays, it is no surprise that the three short courses in SBBD 2014 were related to the management of large volumes of data: (i) "Big Social Data: Princípios sobre Coleta, Tratamento e Análise de Dados Sociais", by Tiago Cruz França, Fabrício Firmino de Faria, Fabio Medeiros Rangel, Claudio Miceli de Farias, and Jonice Oliveira; (ii) "Estratégias para Proteção da Privacidade de Dados Armazenados na Nuvem", by Eliseu C. Branco Jr., Javam C. Machado, and José Maria da Silva Monteiro Filho; and (iii) "Otimização de Desempenho em Processamento de Consultas MapReduce", by Ivan Luiz Picoli, Leandro Batista de Almeida, and Eduardo Cunha de Almeida.

SBBD 2014 also offers two tutorials, both presented by experts on each topic. The first, with Brazilian authors, is "Métodos Automáticos para Desambiguação de Nomes de Autores em Repositórios de Dados Bibliográficos", by Anderson Almeida Ferreira, Alberto H. F. Laender, and Marcos André Gonçalves. The international tutorial, entitled "Human-Powered Database Operations", is presented by Dongwon Lee.

We feel honored to have three distinguished invited speakers in the 2014 edition of SBBD: (i) Gerhard Weikum, from Max Planck Institute for Informatics, who is giving a talk entitled "Big Text: from Language to Knowledge"; (ii) Juliana Freire, from New York University, talking about "Exploring Big Urban Data"; and (iii) Sandra de Amo, from Universidade Federal de Uberlâdia, with a talk entitled "Preference Mining for Personalization and Recommendation".

Finally, SBBD 2014 includes the presentation of 8 papers in the Demo session, and 15 papers in the Thesis and Dissertation Workshop.

SBBD 2014 is the result of the collective effort of a large community, which we gratefully acknowledge. We thank the local organization committee and the symposium chairs who worked hard to guarantee an outstanding symposium. We are also grateful to the Steering Committee members for their help, advice and support. Further, we thank the program committee members and external reviewers who provided high quality reviews for the submitted papers, and the authors who submitted their papers to SBBD 2014. Finally, we are grateful to our sponsors. Without their support we would not be able to organize this annual event that brings together our community.

We hope you all enjoy SBBD 2014 in Curitiba.

Carmem Satie Hara, UFPR
SBBD 2014 Local Organization Committee Chair

Vidal Martins, PUC-PR
SBBD 2014 Local Organization Committee Chair

Table of Contents

Full Papers 7

JIDM Articles 13

SBBD Proceedings 15

Demos and Applications Session 197

Workshop on Thesis and Dissertations in Databases 251

Tutorials 367

Invited Talks 377

29th Brazilian Symposium on Databases

full

October 6th to 9th, 2014
Curitiba – PR – Brazil

FULL PAPERS

PROCEEDINGS

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

SBBD Program Committee Chairs

Mirella M. Moro, UFMG
Renata Galante, UFRGS

Editorial

It is our pleasure to introduce the *Proceedings of SBBD* with the papers accepted for presentation at the 29th Brazilian Symposium on Databases (SBBD 2014), held in the city of Curitiba, in the state of Paraná, Brazil, from October 6th to 9th, 2014. SBBD 2014 was organized by two academic institutions: Informatics Department of the Federal University of Paraná (UFPR) and the Politechnic School of the Catholic University of Paraná (PUC-PR). SBBD is the official database event of the Brazilian Computer Society (SBC) and the largest venue in Latin America for presentation and discussion of research results in the databases domain. Along with technical sessions, SBBD includes invited talks and tutorials given by distinguished speakers from the national and international research communities. SBBD also promotes a Demos and Applications Session, and a Workshop on Thesis and Dissertations as co-located events. All papers submitted to SBBD should present interesting results or novel thought-provoking ideas in any subject on the databases and related areas. For the 2014 edition, SBBD has accepted two kinds of submissions.

The first one is the *JIDM Articles category*. The review process was performed in two rounds. The first round included the initial evaluation of the articles, which could be directly accepted for publication, selected for the second round, or rejected. Selected submissions were updated based on the reviewers' comments and submitted again for the second round of evaluation, in which the papers faced a fast-track revision. All papers were evaluated by at least three members of the SBBD 2014 Program Committee. All accepted papers (in the first and second rounds) are published in the third issue of JIDM volume 5 and were invited for oral presentation at SBBD 2014.

The second is the *SBBD Proceedings category*. The review process of SBBD Proceedings was performed in one round with a rebuttal phase. All authors were initially notified with the reviews and then had one week for answering the reviewers' comments during the rebuttal phase. After evaluating the rebuttal comments, a final decision was achieved. All accepted papers are published in this proceedings and were invited for oral presentation at SBBD 2014. By the end of SBBD, a subset of the Program Committee will once again evaluate the presented papers for selecting those that shall be invited for extended publication at JIDM.

The statistics of SBBD 2014 are as follows. The JIDM category received 36 submissions, out of which 10 were accepted: 4 directly from the first round and 6 from the second round. The SBBD Proceedings category received 55 submissions, out of which 18 were accepted: 12 directly from the first round and 6 after the rebuttal phase. Altogether, SBBD 2014 received 91 submissions, out of which 28 were accepted, which represents an overall acceptance rate of 32%. The topics with more submissions (according to the author's selection from the Topics of Interest) were: Applications of Data Mining (with 11 submissions), Query Languages and Proces-

sing (with 10) and Information Retrieval Models and Techniques (with 8). Likewise, the topics with more accepted submissions were Applications of Data Annotation and Provenance, Data Mining, and Web Data (each with 4 papers accepted).

The Proceedings of SBBD and its respective JIDM issue are the result of the collective effort of a large community, which we gratefully acknowledge. We thank SBBD 2014 local organization committee and its symposium chairs who worked hard to guarantee an outstanding Symposium. We are also grateful to the Steering Committee members for their help, advice and support. We also thank the Program Committee members and External Reviewers who provided high quality reviews for the submissions, working very hard in all rounds of review. Finally, we are grateful to the authors who submitted their work to JIDM volume 5 and SBBD 2014.

Mirella M. Moro, UFMG
SBBD 2014 Program Committee Chair

Renata Galante, UFRGS
SBBD 2014 Program Committee Chair

29th Brazilian Symposium on Databases

October 6th to 9th, 2014
Curitiba – PR – Brazil

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

SBBD Steering Commitee

Ana Carolina Salgado, UFPE
Cristina Ciferri, ICMC-USP (chair)
José Palazzo Moreira de Oliveira, UFRGS
Marco A. Casanova, PUC-Rio
Mirella M. Moro, UFMG
Vanessa Braganholo, UFF

SBBD 2014 Commitee

Steering Committee Chair
Cristina Ciferri, ICMC-USP

Local Organization Chairs
Carmem Hara, UFPR and Vidal Martins, PUC-PR

Program Committee Chairs
Mirella M. Moro, UFMG and Renata Galante, UFRGS

Demos and Applications Session Chairs
José Antônio Macedo, UFC and Flávio Sousa, UFC

Workshop on Thesis and Dissertations in Databases Chairs
João Eduardo Ferreira, IME-USP and Fabio Porto, LNCC

Lectures Chair
Bernadette Farias Lóscio, UFPE

Tutorials Chair
Vanessa Braganholo, UFF

Local Organization Committee

Carmem Satie Hara, UFPR (chair)
Cristina Verçosa Pérez Barrios de Souza, PUC-PR
Daniel S. Kaster, UEL (proceedings)
Marcos A. Carrero, UFPR
Nadia Puchalski Kozievitch, UTFPR
Vidal Martins, PUC-PR (chair)

SBBD Program Committee

Alan Pirotte (Un. Catholique de Louvain, Belgium)
Alberto Laender (UFMG, Brazil)
Alexandre Plastino (UFF, Brazil)
Altigran Silva (UFAM, Brazil)
Alvaro Fernandes (Un. of Manchester, UK)
Amr El Abbadi (UCSB, USA)
Anastasios Kementsietsidis (IBM Research, USA)
André C. P. L. F. Carvalho (ICMC-USP, Brazil)
Angelo Brayner (UNIFOR, Brazil)
Bernadette Lóscio (UFPE, Brazil)
Caetano Traina Jr (ICMC-USP, Brazil)
Carina Dorneles (UFSC, Brazil)
Carmen Hara (UFPR, Brazil)
Celso Massaki Hirata (ITA, Brazil)
Cláudio de Souza Baptista (UFCG, Brazil)
Clodoveu Davis (UFMG, Brazil)
Cristina Ciferri (ICMC-USP, Brazil)
Dhananjay Kulkarni (Boston University, USA)
Elaine P. M. de Sousa (ICMC-USP, Brazil)
Eli Cortez (Microsoft Research, USA)
Estevam Hrushka (UFSCar, Brazil)
Fabio Porto (LNCC, Brazil)
Javam Machado (UFC, Brazil)
João Eduardo Ferreira (IME-USP, Brazil)
José Palazzo Moreira de Oliveira (UFRGS, Brazil)
José Viterbo Filho (UFF, Brazil)
Lipyeow Lim (University of Hawaii, USA)

Luciano Barbosa (IBM Research, Brazil)
Marcelo Arenas (PUC, Chile)
Marco Antonio Casanova (PUC-Rio, Brazil)
Marco Cristo (UFAM, Brazil)
Marcos Vaz Salles (University of Copenhagen, Denmark)
Maria Luiza Machado Campos (UFRJ, Brazil)
Mario Nascimento (University of Alberta, Canada)
Marta Mattoso (UFRJ, Brazil)
Oscar Pastor Lopez (Un Poli de València, Spain)
Ricardo Torres (UNICAMP, Brazil)
Ronaldo dos Santos Mello (UFSC, Brazil)
Sandra de Amo (UFU, Brazil)
Theo Haerder (Un. of Kaiserslautern, Germany)
Valéria Times (UFPE, Brazil)
Vanessa Braganholo (UFF, Brazil)
Viviane P. Moreira (UFRGS, Brazil)
Wagner Meira Jr. (UFMG, Brazil)

External Reviewers

Adriana Prado (UFF, Brazil)
Eduardo Ogasawara (CEFET-RJ, Brazil)
Michele A. Brandão (UFMG, Brazil)
Raul Weber (UFRGS, Brazil)

Table of Contents (JIDM Articles)

jidm

A 'Wider' Concept for Similarity Joins
Luiz Olmes Carvalho, Willian Dener de Oliveira, Ives R. V. Pola, Agma Juci Machado Traina, Caetano Traina Jr.
https://seer.lcc.ufmg.br/index.php/jidm/article/view/715

Analyzing Missing Data in Metric Spaces
Safia Brinis, Agma Juci Machado Traina, Caetano Traina Jr.
https://seer.lcc.ufmg.br/index.php/jidm/article/view/714

Combining Ontology Modules for Scientific Text Annotation
Marcus Albert Alves da Silva, Maria Claudia Cavalcanti
https://seer.lcc.ufmg.br/index.php/jidm/article/view/724

Embedding k-Nearest Neighbor Queries into Relational Database Management Systems
Gabriel V. De Pierro, Mônica Ribeiro Porto Ferreira, Lúcio Fernandes Dutra Santos, Willian Dener de Oliveira, Agma Juci Machado Traina, Caetano Traina Jr.
https://seer.lcc.ufmg.br/index.php/jidm/article/view/717

Examining Multiple Features for Author Profiling
Edson R. D. Weren, Anderson U. Kauer, Lucas Mizusaki, Viviane P. Moreira, José Palazzo Moreira de Oliveira, Leandro K. Wives
https://seer.lcc.ufmg.br/index.php/jidm/article/view/716

Improving Index Maintenance Using Decision Analysis
Pedro Holanda, João Paulo Pordeus, José Maria da Silva Monteiro Filho, Angelo Brayner, Sérgio Lifschitz
https://seer.lcc.ufmg.br/index.php/jidm/article/view/720

Reducing Fragmentation in Incremental Author Name Disambiguation
Luciano Vilas Boas Esperidião, Anderson A. Ferreira, Alberto H. F. Laender, Marcos André Gonçalves, David Menotti Gomes, Andrea Iabrudi Tavares, Guilherme Tavares de Assis
https://seer.lcc.ufmg.br/index.php/jidm/article/view/721

Seamless Integration of Distance Functions and Feature Vectors for Similarity-Queries Processing
Marcos Vinicius Naves Bedo, Agma Juci Machado Traina, Caetano Traina Jr.
https://seer.lcc.ufmg.br/index.php/jidm/article/view/689

The HeightBL Algorithm for Bulk-loading F-Onion-trees
Arthur Emanuel de Oliveira Carosia, Ricardo Rodrigues Ciferri, Cristina Dutra de Aguiar Ciferri
https://seer.lcc.ufmg.br/index.php/jidm/article/view/729

Towards Cloud Data Warehouses of Multivalued Encrypted Values
Claudivan Cruz Lopes, Valéria Cesário Times, Stan Matwin
https://seer.lcc.ufmg.br/index.php/jidm/article/view/726

Table of Contents (SBBD Proceedings)

sbbd

A Similarity-Based Approach to Match Elements across Versions of XML Documents 17
Fernando Campello, Bruno Pinto, Gabriel Tessarolli, Alessandreia Oliveira, Carlos Roberto Carvalho Oliveira, Márcio Tadeu Oliveira Júnior, Leonardo Murta, Vanessa Braganholo

Análise de Estratégias de Acesso a Grandes Volumes de Dados 27
Douglas Oliveira, Cristina Boeres, Fábio Porto

Captura e Consulta a Dados de Proveniência Retrospectiva Implícita Intra-Atividade 37
Wellington Moreira de Oliveira, Vitor Neves, Kary Ocaña, Leonardo Murta, Vanessa Braganholo

CARLO: Modelo Ontológico de Contexto para Recomendação de Objetos de Aprendizagem em Ambientes Pervasivos 47
Guilherme Medeiros Machado, José Palazzo Moreira de Oliveira

Data Extraction from Structured Databases using Keyword-based Queries 57
Mariana Soller Ramada, João Carlos da Silva, Plínio de Sá Leitão Júnior

Debugging Scientific Workflows with Provenance: Achievements and Lessons Learned 67
Daniel de Oliveira, Flavio Costa, Vítor Silva, Kary Ocaña, Marta Mattoso

Definição e Avaliação de uma Abordagem para Extração e Rotulação de Conteúdo Obtido da DeepWeb 77
Augusto Ferreira de Souza, Ronaldo dos Santos Mello

Gerência de Proveniência Multigranular em Linked Data com a Abordagem ETL4LinkedProv 87
Rogers Reiche de Mendonça, Sérgio Manuel Serra da Cruz, Maria Luiza M. Campos

Haliteds: Agrupamento de Dados em Subespaços de Séries Temporais Multidimensionais 97
Afonso Expedito da Silva, Robson L. F. Cordeiro

Improving Pairwise Preference Mining Algorithms using Preference Degrees ... 107
Juliete Ramos da Costa, Sandra de Amo

Mineração da Opinião sobre Aspectos de Candidatos a Eleições em Comentarios de Noticias 117
Leonardo Augusto Sápiras, Karin Becker

Mineração de Emoções em Textos Multilíngues usando um Corpus Paralelo ... 127
Aline Graciela Lermen dos Santos, Karin Becker, Viviane P. Moreira

oLinDa: Uma Abordagem para Decomposição de Consultas em Federacções de Dados Interligados 137
Danusa Cunha , Bernadette Farias Lóscio

Principais Componentes na Ordenação de Anúncios: Um Experimento em Ambiente Real de Publicidade Computacional 147
André Casimiro, Marcos Broinizi, João Eduardo Ferreira

Ranking de Pesquisadores em Cenários Ambíguos: uma Abordagem Probabilística 157
Luciana B. Maroun, Mirella M. Moro

Towards Supporting Systematic Mappings Studies: An Automatic Snowballing Approach 167
Fábio Bezerra, Carlos Favacho, Rafael Souza, Cleidson de Souza

Uma Abordagem para Recomendação no apoio à Modelagem de Processos de Negócio 177
Mateus Barcellos Costa, Thanner Soares Silva

Uso de SGBDs NoSQL na Gerência da Proveniência Distribuída em Workflows Científicos 187
Guilherme R. Ferreira, Carlos Filipe Jr., Daniel de Oliveira

paper:70

A similarity-based approach to match elements across versions of XML documents

Fernando Campello[1], Bruno Pinto[1], Gabriel Tessarolli[1], Alessandreia Oliveira[1,2], Carlos Oliveira[2], Marcio Oliveira Junior[2], Leonardo Murta[1], Vanessa Braganholo[1]

[1]Universidade Federal Fluminense
[2]Universidade Federal de Juiz de Fora

{fernandocampello, brunoferreirapinto}@id.uff.br,
{carlosroberto, mtojunior}@ice.ufjf.br,
{alessandreia, gtessarolli, leomurta, vanessa}@ic.uff.br

Abstract. *XML documents are often used to provide inter-system interoperability. A related problem is that XML documents evolve over time, so identifying and understanding the changes they undergo become crucial. Some diff approaches based on syntactic and semantic analysis of the documents have been developed to address this problem. The strategy is to find data fragments that are identical in both versions of an XML document and match the corresponding elements through the use of context keys. However, depending on how XML documents are managed, there is no guarantee that the values of these keys remain the same across versions. Thus, differently from existing approaches, this paper proposes the use of similarity to match corresponding elements across XML versions, rather than key equality. It also shows how this can be applied to support both syntactic and semantic XML diff applications.*

1. Introduction

XML is being adopted as a standard language by many industries and research communities to provide inter-system interoperability, including healthcare [Argüello *et al.* 2009; Thuy *et al.* 2013], legislative houses [Hallo Carrasco *et al.* 2013], and Web Science [Getov 2008]. In this context, users are often not only interested in the current value of data but also in their changes. This explains a renewed interest for managing document evolution. It is not a trivial task, though. The challenge resides in the fact that these documents contain a hierarchical structure and user-defined tags [Bray *et al.* 2008]. While allowing flexibility in data representation, this complicates the monitoring of its evolution, especially in large data repositories. To deal with this problem, some *diff* approaches specific to XML documents have been developed [Cobena *et al.* 2002; Wang *et al.* 2003; Santos e Hara 2004]. Based on syntactic analysis, the main strategy on these approaches is to find data fragments that are identical in both versions of the XML document and match the corresponding elements through the use of context keys [Buneman *et al.* 2003].

Although effective, the use of context keys may not be suitable in all scenarios. Depending on how XML documents are managed, there is no guarantee that the values of the attributes and elements that represent these keys remain the same across versions. For example, there may be a typing error on the key value for an element in version *v1* of an XML document that is corrected in version *v2* of the same document. Even if all the remaining data in that element is kept unchanged, the approaches that rely on keys cannot uniquely identify this element across the versions.

In this paper we propose the use of a similarity-based approach to match elements across versions. It overcomes the problem illustrated before, since it does not rely on key equality to identify elements. Instead, it uses a set of characteristics of the elements, employing different strategies according to the element type to determine the similarity degree. According to this strategy, elements are matched if their similarity degree is greater than a given threshold. Additionally, our matching approach aims at obtaining a global optimum matching instead of a local optimum matching.

Both syntactic and semantic XML diff applications can benefit from this similarity-based matching approach. In this paper, we present Phoenix and XChange [Oliveira *et al.* 2014], two applications that use our similarity-based matching approach for both syntactic and semantic XML diff problems, respectively. Phoenix is a novel syntactic XML diff tool that compares two XML documents, which may be two revisions of the same document or two variants with little shared content. XChange is a semantic XML diff tool that helps understanding the evolution of XML documents. It was originally developed to work with a key-based matching approach, but was adapted to work with our similarity-based approach.

The remaining of this paper is organized as follows. Section 2 describes the similarity matching in XML documents. Section 3 presents examples of applications that use this similarity-based approach. Section 4 discusses related work. Finally, Section 5 presents conclusions and suggestions for future work.

2. Similarity-based Matching in XML Documents

The basic task for every diff algorithm is to identify corresponding elements from two documents. For instance, text-based diff basically adopt LCS [Cormen *et al.* 2009] algorithms for this purpose, detecting the longest common subsequence of lines of text between two files. However, depending on the type of document, such task becomes trickier. XML document is one of the types that brings extra complexity to the matching task, due to its tree-based semistructured nature and flexible compositions rules.

There are different ways to identify corresponding elements in XML documents. Among them, we can cite context key approaches and similarity-based approaches. A context key guarantees the uniqueness of an element throughout versions, using key values. It assumes the key does not change, thus, the matching across documents is fairly simple and relies on the equality of such key. However, this precondition may not hold or, even worse, the elements may not have keys. A related problem is that depending on how XML documents are managed, there may be no guarantee that the value chosen as key remains the same across different versions. For such situations, a similarity-based approach fits better, since it does not rely on key attributes. Similarity-based approaches investigate characteristics of the elements under comparison in order to quantify how much alike they are. Thus, the comparison is no longer binary (equal or not equal), but proportional, returning a similarity degree.

The similarity-based approach for XML elements matching proposed by this paper recursively evaluates the similarity between the information contained within the structures of the XML elements and expresses it through a real value. This value ranges from zero to one, zero representing total inequality and one representing the total equality of the elements.

It is important to note that our similarity technique considers XML documents as unordered trees. That is, only the ancestor-descendant relationship is significant, as opposed to ordered trees in which child ordering is also significant. Although ordering amongst elements in the same level is considered in XML specification, there are several scenarios where it should not impact the use of the document. In fact, several approaches in the literature consider unordered XML trees [Chawathe e Garcia-Molina 1997; Wang *et al.* 2003].

Figure 1 represents our similarity-based approach. Starting from the root elements of the documents under comparison, the similarity-based approach analyzes their similarity based on four features: (1) element name; (2) textual content; (3) element attributes; and (4) subelements. For every feature, the approach evaluates its values on both elements under comparison and compares them to produce a similarity component related to that feature. At the end, the similarity components are combined, resulting in a real value that indicates the similarity degree between the elements under comparison.

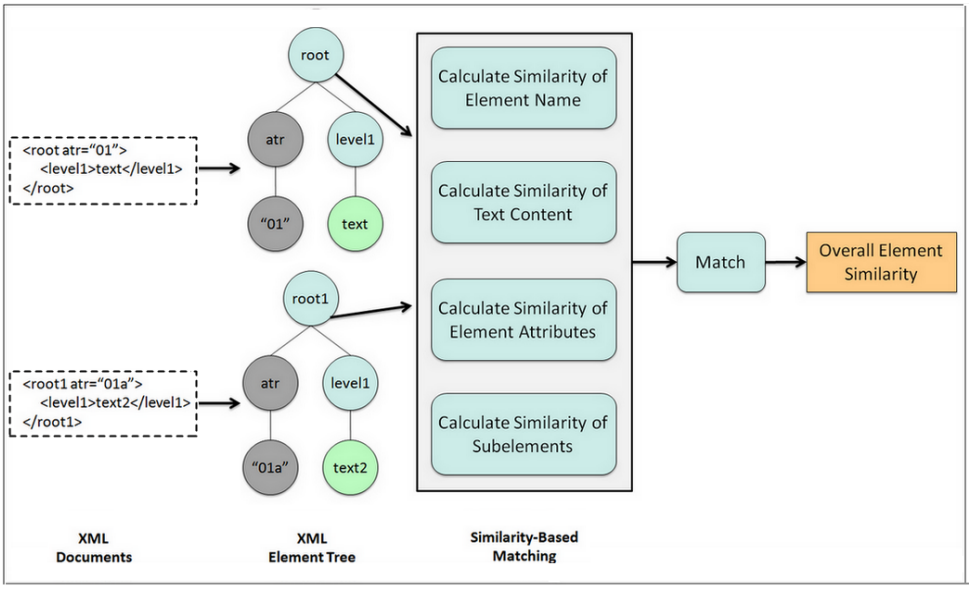


Figure 1. Similarity-based approach to XML matching

In order to calculate each similarity component, our approach uses different algorithms, chosen specifically to deal with the characteristics of each component type. These algorithms and the rationale on their use are briefly described in the following:

- *Name similarity component*. Since element names are strings, the Longest Common Subsequence (LCS) algorithm [Cormen *et al.* 2009] is applied over the names of the elements under comparison. The length of the resulting LCS sequence is then divided by the average length of the two input strings. The result indicates the similarity degree between the element names.
- *Textual content similarity component*. Since the textual content is also a string, the LCS algorithm is applied once more to calculate this similarity component. First, it identifies the LCS sequence and then divides its length by the average length of the textual content from both elements.
- *Attributes similarity component*. Attributes are compared in the similarity-based approach by the following steps:

1. Extract the complete set of attributes names used in both elements under comparison;
2. For each identified attribute, compare its value in the first element with its value in the second element using the LCS algorithm, as already explained, and keep the resulting similarity; if the attribute is not present in one of the elements, this similarity is set to zero. It is worth mentioning that we only compare values when there is a match in the attribute name, i.e., if the name changes, we consider it a new attribute;
3. Sum all the similarities calculated in the previous step and divide by the number of attributes in the complete set;
4. The resulting value is the attributes similarity component.

- *Subelements similarity component*. To calculate this component, another multi-step procedure is used:
  1. Each subelement from the first element is compared to each subelement of the second element using the same similarity calculation algorithm (thus, characterizing recursion), and the results are registered in a matrix;
  2. The all-to-all similarity matrix calculated in the previous step is then provided to the Hungarian algorithm [Kuhn 1955], which outputs the global optimum match amongst the subelements;
  3. From this best match, the subelements similarity component is calculated by dividing the sum of all the similarities in the match by the number of subelements.

All the similarity components are real numbers ranging from zero to one, where zero means total dissimilarity in the given feature, and one means total similarity (equality). With all the similarity components calculated, the overall similarity is calculated using a weighted average.

The weight of each similarity component in the overall similarity calculation is configurable, and should be carefully chosen, depending on the application using the approach. For instance, it may be suitable in some applications to give more importance to text content. That is the case when it is known that the documents being compared use the same schema (thus, the same element names). There might be cases where the application does not use one of the components (by giving it a weight of zero).

3. Similarity-based Matching Applications

Similarity algorithms for comparing XML documents are important in various applications that manipulate semistructured data. The first application that comes to mind in this case is querying XML documents using similarity instead of exact matches [Cohen *et al.* 2003; Dorneles *et al.* 2004; Lima *et al.* 2004]. In this section, however, we focus on other types of applications, such as syntactic and semantic diff. The use of similarity functions to evaluate comparisons in these applications has the potential of making them more generic, as no key is necessary to be set beforehand. The remaining of this section introduces two diff tools that used our similarity-based matching algorithm.

3.1 Phoenix

Phoenix is a diff tool that uses our similarity-based approach to identify and present the differences between two or more XML documents. When used to compare two XML documents, Phoenix calculates the similarities between the elements in these documents and composes a diff document containing the resulting information. This document is presented to the user using a tree view, where each node represents one element from the resulting diff document. Figure 2 presents such visualization. A color scheme is applied to each node, depending on the calculated similarity for the represented element. If the element is only found in the first document, the red color is used for its node, representing that this element was "removed" in the second document. If it is only found in the second document, the green color is applied, meaning that the element was "added". When the resulting similarity of an element is greater than a given threshold, this element is represented in the tree using a gray scale color scheme. The lighter the color, the greater the similarity found for that element.

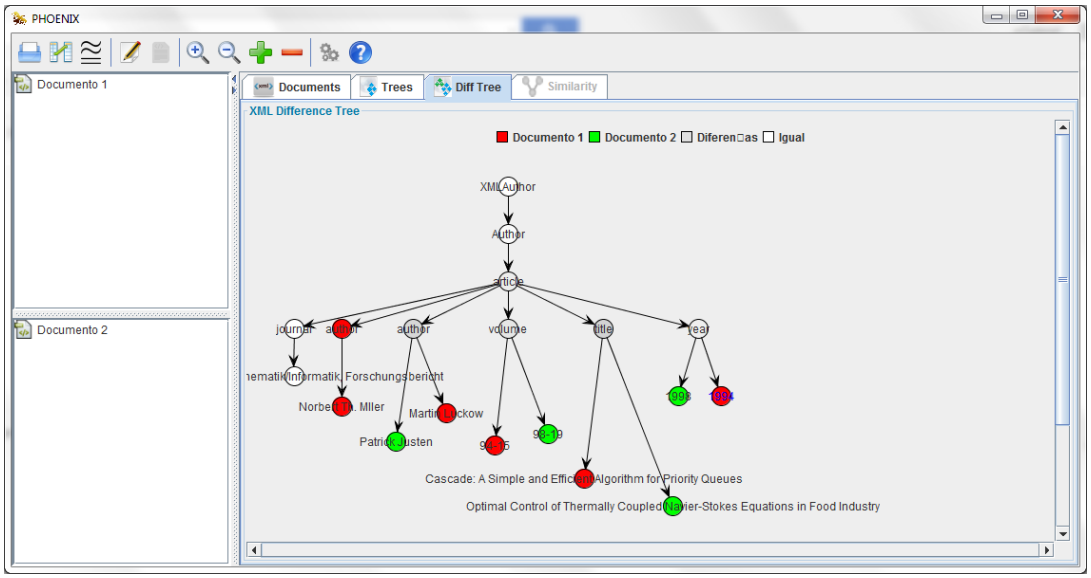


Figure 2. Phoenix diff tree visualization

In addition to the already presented configurable weights of the similarity-based approach, Phoenix tool allows the user to set extra parameters that adapt the similarity calculation to specific application scenarios. A situation that deviates the overall similarity calculation is the presence of trivial similarities, that is, total similarities (100%) pointed by similarity component calculation where the considered feature (attributes, for instance) is non-existent in both elements being compared. To illustrate this problem, consider, for instance, two elements that have no attributes. The attribute similarity component in this case would be 100%. This may not be suitable in certain situations, especially if we would like to focus on the differences instead of on the equalities. For that, a parameter *ignore trivial similarities* was created. If set, the similarity calculation algorithm will simply discard the similarity components identified as trivial during evaluation. It is worth mentioning that the weight of discarded similarity components will also be left out of the similarity formula.

When comparing XML documents within the same XML schema, it is already expected that the element names will be equal. If they are not equal, we should consider

the documents dissimilar. For these scenarios, a parameter called *name similarity required* was created. If set, the similarity calculation will stop if the name similarity component is not 1.0, that is, if the names are not equal. It is also worth mentioning that if this parameter is set, the name similarity component is excluded from the overall similarity formula. With that, the weights from the remaining similarity components shall sum up to 1.0.

Another parameter is the *automatic similarity weight*, which equally distributes the similarity weights, considering the other parameters. That is, if *name similarity required* is set, it will set the other three similarity component weights to 0.33. Also, if a trivial similarity is found and the *ignore trivial similarity* parameter is set, the approach will again equally distribute the weights to the remaining similarity components.

Finally, the last parameter allows the configuration of the *similarity threshold*, that is, the lower limit for the resulting overall similarity in which the approach considers the elements to match (have some similarity involved). If the found similarity is lower than the threshold, the elements are not matched.

3.2 XChange

XChange is an XML semantic diff tool. Its main contribution is to support the understanding of XML documents evolution [Martins *et al.* 2013; Oliveira *et al.* 2014]. Initially, XChange used context keys to support the match of the corresponding elements in different versions of the same XML document. The current tool, however, uses our similarity-based approach.

The input data consists of two versions (revisions) of an XML document and a set of rules. These rules are divided into two categories: match rules and semantic enrichment rules. The match rules are used to identify the corresponding elements in different versions. The rules of semantic enrichment try to understand the meaning of these changes, i.e., to produce a semantic diff [Mens 2002].

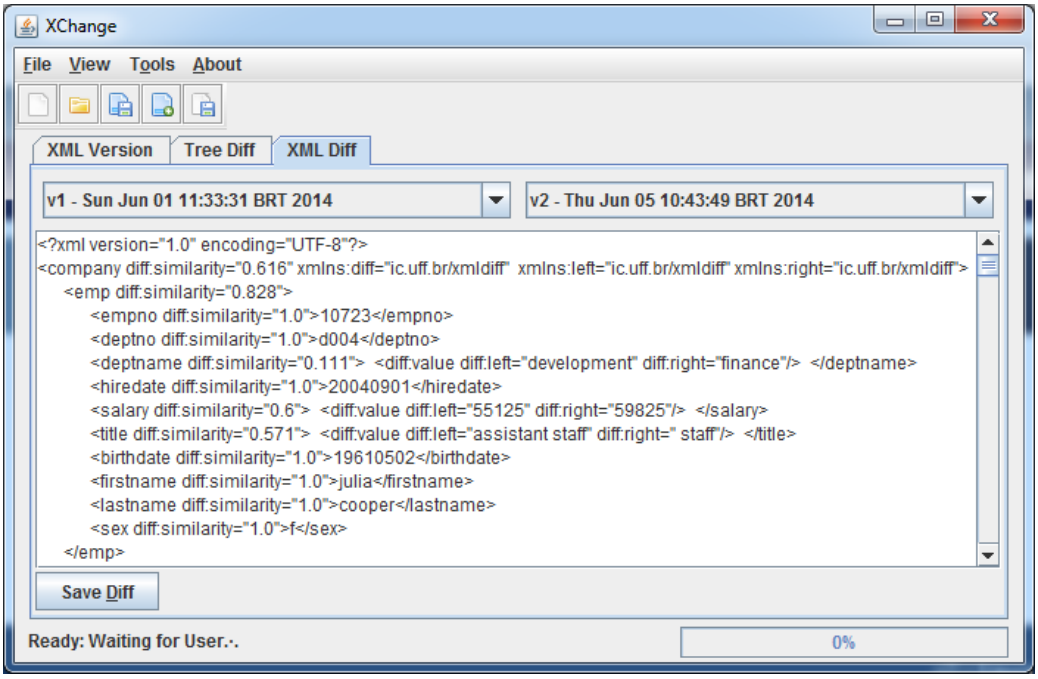


Figure 3. Similarity results

XML document versions are pre-processed and transformed into a set of Prolog facts [Bratko 2001]. However, before generating the Prolog facts, it is necessary to calculate the similarity between all the elements that compose the XML document versions to support the matching of the elements. This task is performed using the described similarity-based approach. A domain expert specifies the similarity weight of each component, the options to use with such weights and sets the minimum similarity threshold. The result is an XML document that lists the elements of the two versions and their respective similarities, as shown in Figure 3. It is worth mentioning that, in this application, we only compare the elements values when there is a match in their names. This is performed by configuring the similarity-based approach underneath. Our reasoning for this is that we respect the schema and assume that elements or attributes with different names should not be matched, regardless of the content.

This output is used to modify the original versions of the XML document, adding an *<id>* element to the corresponding pairs of elements, matched by the similarity-based approach rate. For elements with similarity below the minimum similarity threshold, and also for those that do not have a corresponding match element, we add different *<id>* tags. When all the similarity calculation is finished, the translation process generates several Prolog facts, transforming elements into predicates and their contents into constants (shown in Figure 4). Finally, the Prolog inference engine applies the match rules and the semantic rules on the generated Prolog facts and returns the high level meaning of the changes. In other words, it is able to identify the reason of the evolution of the XML document from an earlier version to a later one.

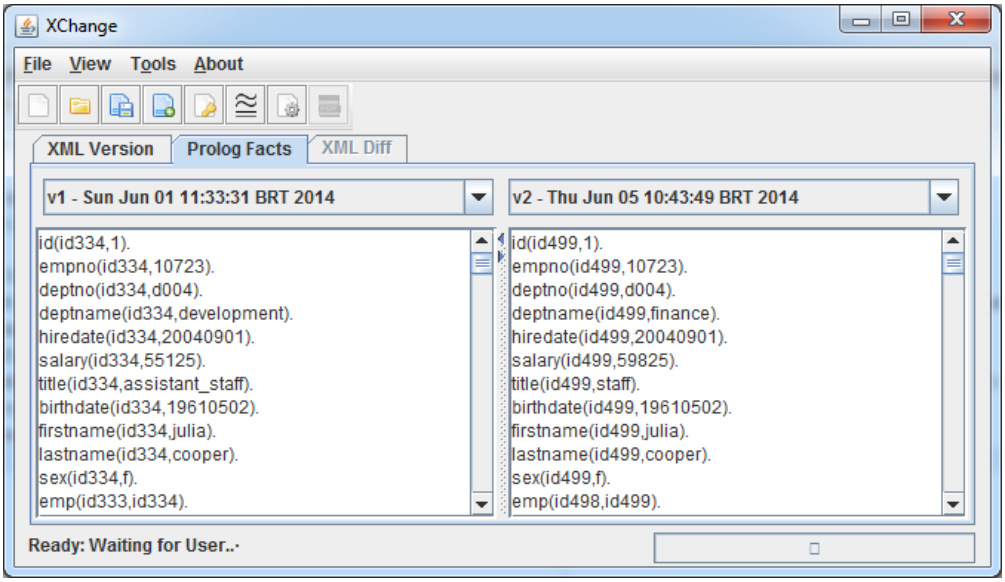


Figure 4. Prolog facts

4. Related Work

During XML documents evolution, it is important to highlight which were the changes that took place to transform from an old version into a new version. To deal with this problem, some diff approaches were developed. There are a number of techniques that identify corresponding elements in XML documents based on context keys. There are also algorithms that support change detection and subtree matching based on similarity techniques. Following, we discuss some approaches that use these strategies.

XyDiff [Cobena *et al.* 2002] detects the differences between versions of XML documents using a key-based approach. It uses ordered trees. Its strategy is to identify large subtrees unchanged between two versions of the XML document, thus reducing the amount of data to compare. Its running time is linear in the size of the document, but it cannot guarantee an optimal result. In certain cases, the change set is not minimal. It is worth mentioning that it considers the operations of insert, delete, and update of elements. It also considers the move operation on subtrees that transfers a node or a subtree from one position to another. X-Diff [Wang *et al.* 2003] is also a key-based approach to detect changes between versions of XML documents. It focuses on ensuring the optimal delta, i.e., the minimum operation sequence that can transform the XML tree from one version to the other. Moreover, it considers only the standard operation of insert, delete, and update of elements in diffs. XKeyDiff [Santos e Hara 2004] is a diff algorithm that uses XML keys to match elements that refer to the same entity in two versions of the document. With that approach, it is able to find matches that may not be possible using solely the structural analysis of XML documents. It uses the XyDiff algorithm [Cobena *et al.* 2002] to find additional matches, and generates the edit script corresponding to the correct order of operations that transforms a previous XML version into a new one. MH-DIFF [Chawathe e Garcia-Molina 1997] detects changes between two structured data snapshots, or trees. It shows the changes as an edit script and it considers the standard operation, plus move and copy operations on subtrees. According to the authors, MH-Diff transforms the change detection into a problem of computing a minimum-cost edge cover of a bipartite graph. Although key-based techniques are effective, they do not work on cases where the elements don't present a key or if the elected key changes across versions. Our similarity-based approach handles such situations.

There are also algorithms that already apply similarity across XML documents. Tekli *et al.* [2009] discusses and classifies several algorithms according to the technique they use: (1) Edit Distance (ED), for algorithms that use edit distance between the trees representing the documents to derive similarity; (2) Information Retrieval, for those used in query/document matching, generally applying approximations to reduce the response time of queries, at the expense of decreasing the exactness and quality of the matching; (3) Other application-specific techniques, such as structure matching and path similarity. Our work provides an alternative to ED-based techniques, which are focused on document/document structure and content matching. XML-SIM-CHANGE [Viyanon e Madria 2010] detects XML similarity using change detection mechanism to join XML document versions. Keys in subtrees play an important role in order to avoid unnecessary comparisons of subtrees within different XML versions of the same document. It uses a relational database to store XML versions and apply SQL for detecting similarities. Differently from these approaches, our similarity-based approach is able to provide a similarity degree amongst attributes, elements, and documents as a whole, allowing a precise comparison of versions. We believe that this characteristic, combined with the fine tuning of the weight in each similarity component, can increase the applicability of our approach. Although not entirely related to our approach, it is worth mentioning that there are initiatives to XML document/grammar matching, such as [Bertino *et al.* 2004; Xing 2006; Tekli *et al.* 2007]. These approaches try to identify how similar to the grammar a document is. Thuy *et al.* [2013] also proposes metrics and an approach to calculate XML schema similarity, useful in data integration.

5. Conclusion

This work introduced an approach for matching XML documents through an analysis of similarity. Traditional approaches compare XML documents versions using, for example, context keys to match elements. In some situations this is not feasible. There are cases where it is not possible to define a context key. In other situations, when the elements are edited, there is no guarantee that the key values will remain the same across versions. Our approach provides an alternative to element matching during XML document comparison that is applicable even in those situations. Furthermore, with the flexibility achieved through the configurable weights for each similarity component and the parameters, our approach allows applications to obtain different similarity degrees from the document comparison. Future work may contribute to enhance the user application interaction, so it becomes more intuitive in the context of setting the parameters. We will perform independent sensibility tests to find out the best values for each parameter in each context. Also, we plan to conduct an experimental evaluation to measure the runtime and the quality of the match when compared to existing approaches.

Acknowledgement

We would like to thank CNPq and FAPERJ for the financial support.

References

ARGÜELLO, M., DES, J., FERNANDEZ-PRIETO, M. J., PEREZ, R., PANIAGUA, H. Executing Medical Guidelines on the Web: Towards Next Generation Healthcare. In *Applications and Innovations in Intelligent Systems XVI*, Springer London, p. 197–210, 2009.

BERTINO, E., GUERRINI, G., MESITI, M. A matching algorithm for measuring the structural similarity between an XML document and a DTD and its applications. *Inf. Syst.* 29(1): 23–46, 2004.

BRATKO, I. *Prolog programming for artificial intelligence*. Addison Wesley, Harlow, England; New York, 2001.

BRAY, T., PAOLI, J., SPERBERG-MCQUEEN, C. M., MALER, E., YERGEAU, F. Extensible Markup Language (XML) 1.0 (Fifth Edition)., 2008.

BUNEMAN, P., DAVIDSON, S., FAN, W., HARA, C., TAN, W.-C. Reasoning about keys for XML. *Inf. Syst.* 28(8): 1037–1063, 2003.

CHAWATHE, S. S., GARCIA-MOLINA, H. Meaningful Change Detection in Structured Data. In *ACM SIGMOD International Conference on Management of Data*, ACM, New York, USA, p. 26–37, 1997.

COBENA, G., ABITEBOUL, S., MARIAN, A. Detecting changes in XML documents. In *International Conference on Data Engineering (ICDE)*, IEEE Computer Society, San Jose, California, USA, p. 41–52, 2002.

COHEN, W. W., RAVIKUMAR, P., FIENBERG, S. E. A Comparison of String Metrics for Matching Names and Records. In *KDD Workshop on Data Cleaning and Object Consolidation*, p. 73–78, 2003.

CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L., STEIN, C. *Introduction to Algorithms*. The MIT Press, 2009.

DORNELES, C. F., HEUSER, C. A., LIMA, A. E. N., SILVA, A. S. DA, MOURA, E. S. DE. Measuring Similarity Between Collection of Values. In *Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management* WIDM '04., ACM, New York, NY, USA, p. 56–63, 2004.

GETOV, V. e-Science: The Added Value for Modern Discovery. *Computer* 41(11): 30–31, 2008.

HALLO CARRASCO, M., MARTÍNEZ-GONZÁLEZ, M. M., LA FUENTE REDONDO, P. DE. Data Models for Version Management of Legislative Documents. *J. Inf. Sci.* 39(4): 557–572, 2013.

KUHN, H. W. The Hungarian Method for the Assignment Problem. *Nav. Res. Logist.* 2(1-2): 83–97, 1955.

LIMA, A. E. N., DORNELES, C., HEUSER, C. Eris: Um protótipo para consulta por similaridade a bases de dados XML. In *Sessão de Demos do Simpósio Brasileiro de Banco de Dados*, SBC, Brasília, DF, p. 13–18, 2004.

MARTINS, G., LARCHER, J., OLIVEIRA, A., MURTA, L., BRAGANHOLO, V. XChange: Compreensão de Mudanças em Documentos XML. In *Sessão de Demos do Simpósio Brasileiro de Banco de Dados*, SBC, Recife, PE, p. 31–36, 2013.

MENS, T. A State-of-the-Art Survey on Software Merging. *IEEE Trans. Softw. Eng.* 28(5): 449–462, 2002.

OLIVEIRA, A., MURTA, L., BRAGANHOLO, V. Towards Semantic Diff of XML Documents. In *Symposium on Applied Computing (SAC)*, ACM, Gyeongju, Korea, p. 833–838, 2014.

SANTOS, R. C., HARA, C. S. XKeyDiff - Um algoritmo semântico para detecção de mudanças entre documentos XML. *Rev. Eletrônica Iniciaç. Científica REIC* 4(3), 2004.

TEKLI, J., CHBEIR, R., YETONGNON, K. Structural Similarity Evaluation Between XML Documents and DTDs. In *Web Information Systems Engineering – WISE 2007* Lecture Notes in Computer Science., Springer Berlin Heidelberg, p. 196–211, 2007.

TEKLI, J., CHBEIR, R., YETONGNON, K. An overview on XML similarity: Background, current trends and future directions. *Comput. Sci. Rev.* 3(3): 151–173, 2009.

THUY, P. T. T., LEE, Y.-K., LEE, S. Semantic and structural similarities between XML Schemas for integration of ubiquitous healthcare data. *Pers. Ubiquitous Comput.* 17(7): 1331–1339, 2013.

VIYANON, W., MADRIA, S. K. XML-SIM-CHANGE: Structure and Content Semantic Similarity Detection among XML Document Versions. In *On the Move to Meaningful Internet Systems, OTM 2010* Lecture Notes in Computer Science., Springer Berlin Heidelberg, p. 1061–1078, 2010.

WANG, Y., DEWITT, D. J., CAI, J.-Y. X-Diff: an effective change detection algorithm for XML documents. In *International Conference on Data Engineering (ICDE)*, IEEE Computer Society, Bangalore, India, p. 519 – 530, 2003.

XING, G. Fast Approximate Matching Between XML Documents and Schemata. In *Frontiers of WWW Research and Development - APWeb 2006* Lecture Notes in Computer Science., Springer Berlin Heidelberg, p. 425–436, 2006.

paper:69

Análise de Estratégias de Acesso a Grandes Volumes de Dados

Douglas Ericson M. de Oliveira[1], Cristina Boeres[1], Fábio Porto[2]

[1]Instituto de Computação – Universidade Federal Fluminense (UFF)

[2]Data Extrem Lab (DEXL) – Laboratório Nacional de Computação Científica (LNCC)

doliveira@ic.uff.br, boeres@ic.uff.br, fporto@lncc.br

Abstract. *The efficient processing of big data has become an issue for several areas. In science, researchers have used dataflows to express computational analysis and experiments on data. An important feature of scientific dataflows is that the analysis must scan a large set of data. In this sense, this work investigates alternatives for storing large volumes of data favoring the execution of dataflows and evaluates two parallel execution models: process and data, according to the following metrics: response time, degree of concurrency and fault tolerance.*

Resumo. *Realizar o processamento eficiente de grandes volumes de dados tem se tornado uma questão a ser solucionada por diversas áreas. Na área da ciência, os pesquisadores têm usado dataflows para expressar análises e experimentos computacionais sobre os dados. Uma característica dos dataflows científicos é que as análises precisam processar um conjunto muito grande dos dados de entrada. Neste sentido, este trabalho investiga alternativas de armazenamento de grandes volumes de dados que favoreçam o processamento de dataflows e avalia dois tipos de paralelismo na execução: processo e dados, de acordo com: tempo de resposta, escalonamento, grau de concorrência e tolerância à falhas.*

1. Introdução

Realizar o processamento eficiente de grandes volumes de dados tem se tornado uma questão a ser solucionada por empresas, ciência, governo, entre outros. Na área científica aplicações da astronomia (imagens de telescópios), biologia (sequência de DNA) e física (aceleradores de partículas) estão gerando um grande volume de dados a serem analisados pelos cientistas.

Tipicamente, cientistas têm usado *dataflows* para exprimir computacionalmente análises e experimentos científicos sobre dados *in-silico*. *Dataflows* científicos definem um processo computacional composto por um conjunto de atividades estruturadas segundo uma ordem parcial e obedecendo uma dependência de dados entre as mesmas. Desta forma, uma atividade recupera o resultado em dados da atividade da qual depende.

Adicionalmente, uma característica relevante aparente em *dataflows* científicos é a necessidade de analisar um conjunto muito grande dos dados de entrada, devido, por exemplo, a busca de registro de um fenômeno sobre o qual não se sabe sobre em que região dos dados ele pode se encontrar. Tal característica leva à um modelo de leitura em que grande volume de dados é recuperado da fonte de dados pela primeira atividade de leitura do *dataflow*.

É interessante observar ainda que o conjunto de dados recuperado da fonte de dados atravessa o *dataflow* sendo submetido à uma sequência de gravação e leitura. Contextualizado no grande volume de dados de leitura para análise, temos um novo cenário de consumo e processamento de dados, com relação à aplicações tradicionais de banco de dados. Neste novo cenário procura-se identificar estratégias de armazenamento de grandes volumes de dados que favoreçam seu processamento e, não seu consumo pelo usuário ou aplicação externa. Espera-se que uma estratégia integrada possa levar a recuperação de dados de forma a favorecer seu processamento pelas atividades de um *dataflow* científico.

Neste sentido, este trabalho experimental investiga o efeito de alternativas de armazenamento e particionamento de grandes volumes de dados no desempenho de *dataflows*. Avaliamos extensivamente em um *cluster* de até 80 nós as seguintes estratégias: BD centralizado, BD particionado, BD distribuído e sistema de arquivo distribuído (HDFS). Observamos que a partir da estratégia de armazenamento dos dados abrem-se oportunidades de definição de arquiteturas de processamento de *dataflows*. Avaliamos duas estratégias básicas: paralelismo de processo e paralelismo de dados. No primeiro, o paralelismo do *dataflow* é independente da localização dos dados, enquanto o segundo explora a execução que privilegia a localização de dados. A combinação da estratégia de armazenamento com a arquitetura de processamento impacta no tempo de resposta, escalonamento, grau de concorrência e tolerância à falhas.

Em particular, observa-se que, o processamento de *dataflows* em grandes volumes de dados, o modelo de paralelismo de dados, adotado pela estratégia *MapReduce*, pode não produzir a melhor estratégia de execução. Características das atividades do *dataflow* podem prejudicar o *speed-up* quando aumenta-se o número de partições de dados. Estas características podem estar associadas com a gravação de resultados intermediários em sistemas de arquivos distribuído, cuja sobrecarga aumenta com o número de partições, e mesmo com a relação entre o volume de dados em cada partição e o tempo de processamento da atividade. Estas conclusões apontam para modelos de custo não lineares com relação à redução de dados a serem processados.

O restante do artigo está estruturado da seguinte forma: Na seção 2 apresentamos a conceituação de base a ser utilizada e os trabalhos relacionados. A seção 3 apresenta as características, funcionamento e vantagens das estratégias de armazenamento investigadas. Estas por sua vez são avaliadas na seção 4. Finalmente, a seção 5 apresenta as conclusões e direções futuras.

2. Preliminares

Nesta seção são descritos alguns conceitos importantes, além de trabalhos relacionados que estão inseridos no contexto deste trabalho.

2.1. Trabalhos Relacionados

Em [Fadika et al. 2012] defende-se que a produção de dados de várias aplicações (como simulações) está se tornando maior do que a capacidade de gerenciar, analisar e compartilhar estes dados. Por isso, a computação *data-intensive* está revolucionando a atual geração de *softwares*. Neste contexto, o paradigma *MapReduce* e sua implementação Hadoop têm sido adotados recentemente pela comunidade científica de forma gradual, dada a ausência de conhecimento dos cientistas nos detalhes da implementação e configuração do Hadoop.

[Fadika et al. 2012] avalia o Hadoop especificamente para operações *data-intensive* (filtrar, reordenar e juntar) para entender suas várias características e compensações de desempenho no contexto de ambientes HPC (*High Performance Computing*), conhecendo assim o impacto do sistema de arquivos, rede e modos de programação no desempenho.

[Loebman et al. 2009] avalia de forma comparativa a execução de aplicações sob o *framework* Pig/Hadoop com relação à SGBDs. Esta análise é realizada em dados de simulações astrofísicas. Foram desenvolvidos casos de uso que são constituídos basicamente de cinco consultas. Estas consultas têm como característica de base realizar a filtragem e junção dos dados. Os testes foram realizados sobre três volumes de dados diferentes: um com 4.2 milhões de tuplas (169 MB), outro com 33.6 milhões de tuplas (1.4 GB) e outro com 916.8 milhões de tuplas (36 GB). Nestes experimentos, as execuções das consultas nos SGBD relacionais tiveram melhor desempenho do que no Pig/Hadoop. Além disso, é consideravelmente mais simples realizar a implementação de uma consulta em SQL do que em Pig Latin.

Em [Stonebraker et al. 2010] também é realizada uma análise similar a acima descrita. Porém, neste artigo o foco é demonstrar que os SGBD paralelos apresentam um melhor desempenho no acesso aos dados, uma vez que a tecnologia vem sendo desenvolvida desde a década de 80 de forma a proporcionar execução eficiente sobre grandes volumes de dados. Por outro lado, o processamento de *workflows*, principalmente científicos, obtiveram um melhor desempenho quando implementados sob o paradigma *MapReduce*. O autor conclui que ambos os modelos são melhores em áreas diferentes e que as aplicações devem implementar o melhor das duas tecnologias.

O Qserv [Wang et al. 2011] é um sistema que tem como objetivo realizar uma leitura eficiente (consultas) de dados que são da ordem de *petabytes*. Os dados neste caso são do catálogo do LSST (*Large Synoptic Survey Telescope*). A cada ano, as imagens coletadas são processadas e suas informações armazenadas no catálogo do LSST. Este fica disponível para os astrônomos e para o público em geral através de um sistema de consultas. O Qserv é um protótipo deste sistema e portanto deve ser capaz de realizar um acesso aos dados de forma eficiente para os usuários.

Observam-se nestes trabalhos que a execução de *dataflows* intensivos de dados é um desafio. A escolha entre SGBDs e sistemas do tipo Hadoop não é simples envolvendo aspectos de particionamento de dados, operações do *dataflow*, tipo de paralelismo, etc. O trabalho discutido neste artigo pretende contribuir com o entendimento das variáveis envolvidas no problema e lançar a luz sobre as alternativas disponíveis.

2.2. Banco de Dados Distribuído - PgPool-II

PgPool-II [PgPool 2012] é um *middleware* entre um conjunto de servidores de BD PostgreSQL e seus clientes, ou seja, é um sistema gerenciador de banco de dados distribuído [Özsu and Valduriez 2011]. O sistema distribui os dados entre um conjunto de nós do *cluster* segundo uma chave de particionamento. Uma consulta enviada ao sistema é redirecionada pelo *middleware* aos nós do *cluster* que contenham um sub-conjunto da solução da consulta, desde que seja expressa segundo a chave de particionamento.

No cenário de *dataflow* desenvolvido neste trabalho, considera-se um *dataflow* interessado em ler um grande percentual da base de dados, para efetuar análises científicas.

Desta forma, a consulta enviada ao *middleware* é re-escrita e enviada a cada nó participante da distribuição. Tipicamente, a adoção da estratégia de BD distribuídos pretende reduzir o tempo de processamento de consultas, alocando parte do processamento para o nó de armazenamento. No contexto de execução de *dataflows*, no entanto, conforme observaremos, a parte significativa do processamento ocorre após a recuperação dos dados, reduzindo o benefício do processamento de consultas local.

2.3. *MapReduce* e Hadoop

MapReduce [Dean and Ghemawat 2008] é um modelo de programação proposto pelo Google para para facilitar o processamento de grandes volumes de dados. A partir deste modelo foram criados *frameworks* que permitem a manipulação de grandes volumes de dados de forma paralela e distribuída, além de prover tolerância a falha, escalonamento de I/O e monitoramento. Um exemplo de *framework* baseado em *MapReduce* é o Hadoop [White 2009]. Uma vez que a operação *Map* realiza uma transformação do tipo de mapeamento nos dados, a operação *Reduce* agrega valores.

HDFS [HDFS 2012] é um sistema de arquivos distribuído nativo do Hadoop. O sistema gerencia dados distribuídos pelos nós e armazenados em unidades chamadas de *chunks*. Cada *chunk* representa uma unidade física de dados, por exemplo de 64k de tamanho. Aplicações acessam múltiplos de *chunks* chamados *splits*. Arquivos armazenados em HDFS são particionados em unidade de *chunks* e distribuídos pelos nós cobertos pelo sistema e replicados segundo a estratégia adotada em número de nós.

Em [Thusoo et al. 2010] é apresentado o Hive, que é um sistema de acesso a dados implementado sobre o Hadoop [White 2009]. Basicamente, a criação do Hive se deve ao fato de aplicações que fazem o acesso aos dados, armazenados de forma não estruturada no HDFS, utilizando o *MapReduce* possuírem códigos muito difíceis de manter e reutilizar. O Hive suporta consultas expressadas em uma linguagem declarativa chamada HiveQL compiladas em tarefas *MapReduce* e executadas no Hadoop. Os dados são armazenados de forma estruturada pelo Hive no HDFS que gerencia a leitura e escrita dos dados inseridos ou selecionados pelo Hive.

2.4. Aplicação - Caso de Uso

Neste trabalho avaliam-se diferentes estratégias tendo como aplicação exemplo, um *dataflow* real desenvolvido por parceiros da área de astronomia. De fato, a questão colocada sobre a eficiência da arquitetura de armazenamento de dados frente à de processamento de *dataflows* foi motivada por esta cooperação.

O *Science Portal*[1] é uma aplicação do Laboratório LIneA (Laboratório Interinstitucional de e–Astronomia). O grande volume de dados gerado por este portal demanda estratégias paralelas de armazenamento, acesso e processamento dos dados, com o objetivo de proporcionar um resultado válido em tempo hábil para a tomada de decisões.

2.4.1. *Dataflow*

O *dataflow* da aplicação é composto de 4 atividades conforme apresentado na Figura 1:

[1] http://ogando.linea.gov.br/

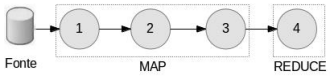


Figure 1. Dataflow

(1) **Leitura de dados**: Obtenção dos dados;

(2) **Tratamento dos dados**: Preparação para formato esperado pelo *SkyMap*;

(3) **SkyMap**: Execução da aplicação *SkyMap* que com base na indicação do posicionamento de cada objeto no arquivo de entrada, colore o ponto respectivo em um histograma da região do céu sendo analisada. Sua implementação na etapa *Map* realiza a leitura do arquivo gerado na atividade anterior e gera um outro arquivo ".pkl" gravado em um sistema de arquivos global. Nestes experimentos foi utilizado o sistema de arquivos distribuído NFS (*Network File System*). Os arquivos assim gerados são passados para a etapa de *Reduce*.

(4) **SkyMapAdd**: Execução da aplicação *SkyMapAdd* que consolida os diversos histogramas gerados localmente em um único. Uma instância do *Reduce* acessa os vários ".pkls" gerados na atividade (3) e produz o histograma final;

Sua implementação no *framework* Hadoop requer o mapeamento do *dataflow* em duas etapas *Map* e *Reduce*. Sendo assim as atividades (1), (2) e (3) fazem parte da etapa *Map* e a (4) compõe a etapa *Reduce*.

3. Estratégias de Armazenamento de Dados

Nesta seção apresentam-se as estratégias de armazenamento investigadas neste trabalho.

3.1. Banco de Dados Centralizado

Esta estratégia é a mais comum e se traduz no armazenamento dos dados inteiramente em um único BD. Considera-se que instâncias paralelas de *dataflows* requisitam subconjuntos disjuntos dos dados armazenado no BD centralizado.

Do ponto de vista da arquitetura de processamento de *dataflow*, apenas a estratégia de paralelismo de processos se mostra possível na medida em que o *dataflow* executa em nós remotos em relação ao BD centralizado. Desta forma, ao paralelizar a execução do *dataflow* em diferentes nós do *cluster*, tem-se que cada um submeterá uma consulta ao BD centralizado recuperando sua partição dos dados.

O subconjunto de interesse é transferido do SGBD ao nó de processamento do *dataflow*, iniciando seu processamento. Na Figura 2 (A) apresenta-se a arquitetura centralizada. De acordo com a Figura, quando a execução da aplicação é iniciada, as várias instâncias do *dataflow* realizam em paralelo a leitura de seus respectivos dados do SGBD. Uma vez que as instâncias do *Dataflow* executando em paralelo requisitam suas partições de dados de um mesmo servidor, pode-se facilmente concluir que existirá concorrência na leitura dos dados.

3.2. Banco de Dados Particionado

Nesta estratégia, os dados são gerenciados por instâncias de BD instaladas em cada nó do *cluster*. Sendo assim, cada instância recebe um subconjunto dos dados e gerencia

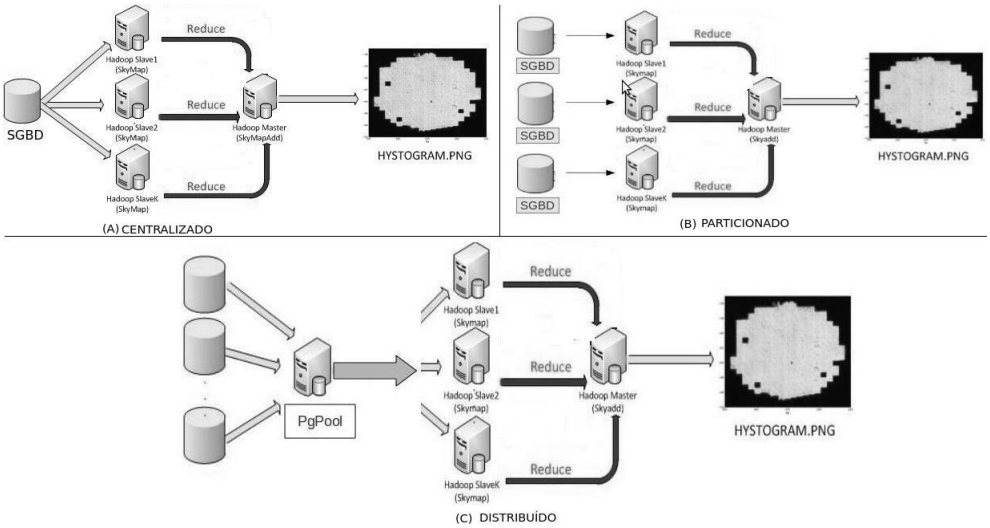


Figure 2. Arquitetura com BD Centralizado, Particionado e Distribuído

sua partição. Pode-se perceber que não há uma visão global dos dados e os *dataflows* conhecem o critério de particionamento. Note-se ainda que essa estratégia coincide com o fato dos *dataflows* acessarem uma grande partição dos dados e terem interesse em uma leitura do tipo varredura. Nesta estratégia podem-se adotar arquiteturas de processamento de *dataflow* tanto baseadas em paralelismo de processo quanto de dados.

Na Figura 2 (B) apresenta-se a arquitetura particionada que foi utilizada para execução da aplicação. Observa-se que os dados deste modelo estão armazenados em cada nó trabalhador. Quando a execução da aplicação é iniciada, cada instância do *dataflow* recupera o conjunto de dados de interesse acessando o SGBD local. Com isso, não existe tráfego de dados na rede nem concorrência de acesso aos dados.

3.3. Banco de Dados Distribuídos

A estratégia de BD distribuído apresenta características de ambas as estratégias anteriores. Assim como o BD centralizado, todas as requisições são submetidas a um nó Mestre que as recebe, redireciona aos nós que contêm dados para a consulta, recebe os resultados e re-envia ao *dataflow*. Quando avalia-se a estratégia de armazenamento à luz da estratégia de processamento de *dataflows* percebe-se que a alocação distribuída dos dados é comprometida pela necessidade de se re-compor o resultado no nó mestre e retornar o conjunto aos nós de processamento.

Na Figura 2 (C) apresenta-se a arquitetura distribuída que foi utilizada para execução da aplicação. Neste modelo, os dados encontram-se distribuídos pelos servidores gerenciados pelo PgPool-II. Desta forma, quando a execução da aplicação é iniciada, as várias instâncias do *dataflow* realizam a leitura de seus respectivos dados através do pgpool. Este por sua vez acessará os dados nos bancos que lhe estão associados. Com isso, existe tráfego de dados na rede e os resultados demonstram que quanto mais máquinas concorrem em solicitações ao pgpool, pior é o tempo médio de acesso. Uma vez que o pgpool é um centralizador das consultas, pode-se concluir que a execução paralela de instâncias do *dataflow* produzirá um gargalo de acesso aos dados.

3.4. Sistema de Arquivos Distribuído

Este modelo é semelhante ao utilizado pelo BD particionado com relação à localidade dos dados, pois os processos *MAP* podem ser executados sobre os mesmos nós de armazenamento.

Neste cenário todo o BD foi importado para o HDFS através do SQOOP [Kimball 2009]. O SQOOP é uma ferramenta de importação e exportação de dados de um SGBD para o HDFS. Neste modelo é necessário que a aplicação realize o tratamento dos dados dentro da fase de *Map*. Uma vez que todos os dados do BD são colocados como arquivos no HDFS, a aplicação deve ler os arquivos e filtrar somente os dados necessários.

Uma outra configuração de teste sobre a distribuição de dados usando o HDFS foi realizada. Nesta configuração, a atividade de tratamento de dados é silenciada. A ferramenta SQOOP importa para o HDFS apenas o sub-conjunto de dados relevante ao *dataflow*. Estes são denominados nos experimentos como $HDFS\_RA$.

4. Experimentos

4.1. Ambiente

Com o propósito de realizar a avaliação experimental,foi preparado um ambiente de processamento de alto desempenho com as seguintes características:

(1) *Hardware*: *Cluster* SGI com 81 nós de processamento, sendo 1 nó mestre (com dois processadores Intel Xeon X5650, 2.67GHz, 6 núcleos reais por processador, com possibilidade de *Hyper-threading*, 24GB memória, 1,5TB de disco rígido) e 80 nós trabalhadores com a mesma configuração do nó mestre, porém contando com 500GB de disco rígido localmente.

(2) *Software*: *Framework* Hadoop 0.20; BD Centralizado (PostgreSQL 8.4), BD Particionado (PostgreSQL 8.4), BD Distribuído (PgPool-II), HDFS e HIVE; Dados para aplicação foram obtidos do catálogo do Laboratório LIneA com aproximadamente 30 milhões de objetos.

Embora as máquinas utilizadas para os testes possuam mais de um núcleo, a fase de *Map* inicia um único processo em cada máquina utilizando somente um núcleo.

4.2. Análises de Resultados

Os experimentos sobre as estratégias de acesso foram realizados com três configurações: 20, 40 e 80 máquinas. Cada estratégia e cada configuração foram executadas 10 vezes. São apresentadas a seguir as médias obtidas dos máximos, mínimos e desvio padrão obtidas.

Cada estratégia é representada na discussão pelo rótulo "$< ESTRATEGIA > n$", que é uma execução de n processos *Map* onde cada um obtém $1/n$ dos dados. Por exemplo, $CENT20$ representa a execução no BD centralizado com 20 *Maps*.

4.2.1. Classificação de Desempenho

Como apresentado na Figura 3 o HDFS20 apresenta o pior desempenho na atividade (2) quando comparado aos demais. Isto se deve à necessidade de se eliminar dados não

acessados pelo *dataflow*, ou seja, refere-se ao custo associado à leitura dos dados em cada partição e subsequente projeção vertical para extração dos atributos de interesse, uma vez que a atividade *SkyMap* apenas utiliza alguns dos atributos contidos no arquivo de entrada. As demais estratégias não requerem tal operação.

Observa-se que o custo de execução da atividade "tratamento de dados" domina o tempo de execução nos experimentos com HDFS, quando a projeção é necessária. Neste cenário, a leitura dos dados em SGBD apresenta vantagem sobre aquela realizada em sistemas de arquivos distribuídos, conforme observado por [Stonebraker et al. 2010]. Essa vantagem, no entanto, é mitigada quando o SGBD é acessado concorrentemente pelas instâncias do *dataflow*, como nos cenários de SGBD centralizado.

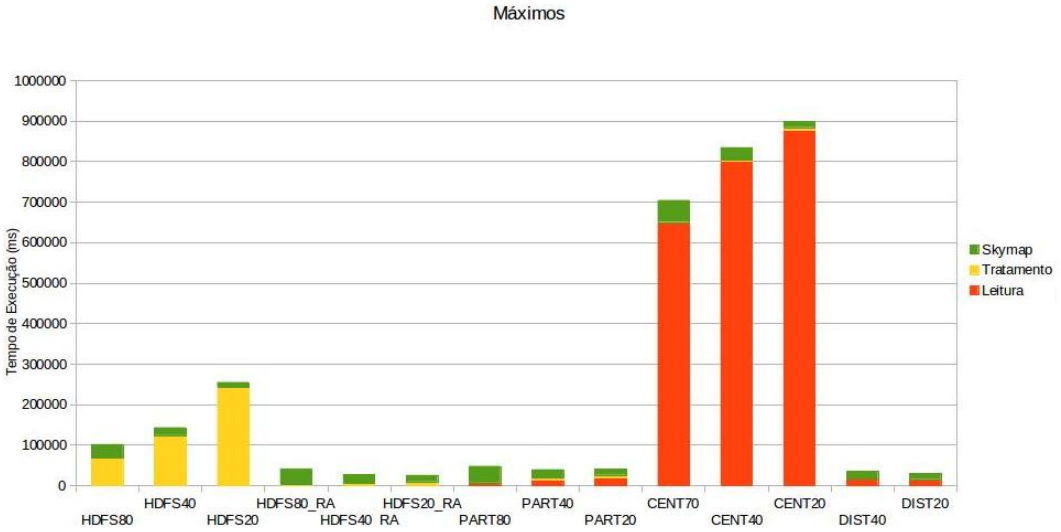


Figure 3. Valores Máximos das Estratégias

Pela Figura 3, o tempo gasto na atividade de Leitura no CENT70 é claramente pior devido à concorrência no acesso aos dados, que só ocorre nesta estratégia. Esta concorrência gera uma maior variação no tempo de resposta às requisições, uma vez que todas as máquinas solicitam os dados a um nó central, algumas máquinas recebem os dados mais rapidamente e outras demoram mais a receber gerando um maior desvio padrão que pode ser observado na Figura 4. Existe tolerância à falhas no nível de dados somente na estratégia com HDFS, pois este automaticamente realiza a replicação dos dados, assegurando a obtenção completa dos resultados.

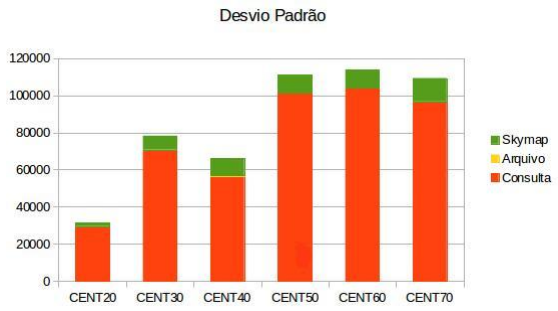


Figure 4. Desvio Padrão da Estratégia Centralizada

4.2.2. Análise de *Speedup*

Ainda observando a Figura 3, a estratégia CENT, nos cenários entre 20 e 70 partições, apresenta um *speedup* de 0.77, para uma redução de leitura por cada instância do *dataflow* de um fator de 3.5. Neste cenário, ao reduzir o número de dados a serem lidos, aumenta-se de forma inversa o número de instâncias do *dataflow* que acessam concorrentemente o SGBD. Esta última mitiga o ganho obtido na redução de dados lidos por cada instância, impactando a intenção de ganho de desempenho com o paralelismo de processo.

Pode-se notar também que no ambiente particionado a etapa Skymap fica mais lenta a medida em que se aumenta o número de nós *Map*. Como já descrito anteriormente, a fase de Map gera arquivos ".pkl" a serem lidos pela fase *Reduce*. Uma vez em que os arquivos gerados em qualquer dos nós de processamento *Map* podem vir a ser acessados pelo processo de *Reduce*, é necessário gravá-los em um sistema de arquivos distribuído. Neste trabalho, optou-se pelo sistema NFS. Consequentemente, a concorrência de acesso ao NFS será proporcional ao número de processos *Map*. A gravação de arquivos intermediários passou a ser um gargalo e por isso a etapa de *SkyMap* apresenta pior desempenho com o aumento do número de *Maps*.

Ainda, a quantidade de dados por nó diminui quando aumenta-se o número de nós. Porém o tamanho dos arquivos ".pkl" se mantém constante, ou seja, quando existe um nó executando a fase de Map será gerado um arquivo ".pkl" com tamanho de 50 MB, quando existem 70 nós executando a fase de *Map* serão gerados 70 arquivos ".pkl" com tamanho de 50 MB cada. Isto é uma característica da aplicação de *SkyMap* adotada neste experimento. Neste processo, uma matriz é preenchida com valores de acordo com os dados da astronomia. Neste contexto, quanto maior o número de processos *SkyMap*, maior será a quantidade de ".pkl" em transferência pela rede. Por último, o *Reduce* se encarrega de realizar a união das matrizes (pkls). Com isso, verificaram-se dois problemas na aplicação *SkyMap* quando o número de nós aumenta: gravação concorrente (NFS) e aumento da quantidade de dados a serem escritos.

Na Figura 3 fica visível o impacto da co-alocação de atividades sobre um mesmo Map. Neste exemplo simples, as etapas de consulta e gravação de arquivos armazenam localmente os dados, apresentando uma redução linear associada ao tempo em relação ao número de nós envolvidos. Já a etapa de *Skymap* que grava no NFS, sofre as consequências da concorrência entre os nós levando, a execução em maior número de nós a apresentar o pior desempenho. Em uma solução de *dataflow* tradicional, esse fenômeno aconteceria a cada etapa, quando da gravação de resultados intermediários.

No experimento com o Hive com 80 máquinas executando a fase de Map o tempo de execução foi de 2500000. Embora o Hive realize a consulta em paralelo através do Hadoop, ele não atende requisições em paralelo. Isto prejudicou muito os resultados do Hive com relação às demais estratégias.

5. Conclusão

O desenvolvimento das ciências *in-silico* tem apresentado novos desafios no processamento de grandes volumes de dados. Em particular, este trabalho investiga a integração de arquiteturas de gerência de dados e estratégias de paralelismo de *dataflows* científicos. Observou-se que as diversas alternativas atualmente disponíveis apresentam vantagens e

desvantagens. Principalmente, observa-se que o consumo de dados da fonte tem impacto relevante no custo total do processamento, assim como aquele relativo à gravação de resultados intermediários nas atividades tem impacto no *dataflow*. Adicionalmente, o perfil de leitura/gravação de dados pelas atividades têm impacto significativo. Este trabalho abre perspectivas para classificação de arquiteturas para processamento intensivos de dados em ciência e na concepção de novos modelos de custo que reflitam diferente alternativas.

6. Agradecimentos

Os autores gostariam de agradecer ao Laboratório Interinstitucional de Astronomia (LineA) por haver motivado essa pesquisa e fornecido o cenário para o experimento. Este trabalho foi parcialmente financiado pelo CNPq, Bolsa de Produtividade em Pesquisa, Processo:309494/2012-5.

7. Referências

References

Dean, J. and Ghemawat, S. (2008). Map reduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113.

Fadika, Z., Govindaraju, M., Canon, R., and Ramakrishnan, L. (2012). Evaluating hadoop for data-intensive scientific operations. In *Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing*, CLOUD '12, pages 67–74, Washington, DC, USA. IEEE Computer Society.

HDFS (2012). http://hadoop.apache.org/docs/stable/hdfs-design.html.

Kimball, A. (2009). Introducing Sqoop. Online.

Loebman, S., Nunley, D., Kwon, Y., Howe, B., Balazinska, M., and Gardner, J. P. (2009). Analyzing massive astrophysical datasets: Can Pig/Hadoop or a relational DBMS help? pages 1–10.

Özsu, M. T. and Valduriez, P., editors (2011). *Principles of Distributed Database Systems*. Springer, New York, 3 edition.

PgPool (2012). http://www.pgpool.net/docs/latest/pgpool-en.html.

Stonebraker, M., Abadi, D., DeWitt, D. J., Madden, S., Paulson, E., Pavlo, A., and Rasin, A. (2010). Mapreduce and parallel dbmss: friends or foes? *Commun. ACM*, 53(1):64–71.

Thusoo, A., Sarma, J. S., Jain, N., Shao, Z., Chakka, P., Zhang, N., Antony, S., Liu, H., and Murthy, R. (2010). Hive - a petabyte scale data warehouse using Hadoop. In *ICDE '10: Proceedings of the 26th International Conference on Data Engineering*, pages 996–1005. IEEE.

Wang, D. L., Monkewitz, S. M., Lim, K.-T., and Becla, J. (2011). Qserv: a distributed shared-nothing database for the lsst catalog. In *State of the Practice Reports*, SC '11, pages 12:1–12:11, New York, NY, USA. ACM.

White, T. (2009). *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition.

paper:77

Captura e Consulta a Dados de Proveniência Retrospectiva Implícita Intra-Atividade

Wellington Oliveira[1,2], Vitor C. Neves[1], Kary Ocaña[3], Leonardo Murta[1], Daniel de Oliveira[1], Vanessa Braganholo[1]

[1]Instituto de Computação – Universidade Federal Fluminense (UFF)
Niterói – RJ – Brasil

[2]Departamento Acadêmico de Ciência da Computação – Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais – Rio Pomba – MG – Brasil

[3]COPPE - Universidade Federal do Rio de Janeiro (UFRJ)

{wellmor, vcneves, leomurta, danielcmo, vanessa}@ic.uff.br;
kary@cos.ufrj.br

Abstract. *Executing scientific experiments in large scale generates huge amounts of data that should be captured for future query. However, most of the available tools collect only part of the provenance (i.e., only that which is explicitly defined in the workflow specification), leading to loss of important information to the analysis of the experiment. This work presents an approach for collecting, storing, and querying intra-activity provenance obtained by monitoring the directories and files manipulated by activities of the workflow. Therefore, information not referenced in the workflow specification are captured and related with previously specified provenance.*

Resumo. *A execução de experimentos científicos em larga escala gera uma grande quantidade de dados que devem ser capturados para consulta futura. Porém, a maioria das ferramentas disponíveis coletam apenas parte da proveniência (i.e., aquela que é explicitada na especificação do workflow), levando à perda de informações importantes para a análise do experimento. Este trabalho apresenta uma abordagem para coleta, armazenamento e consulta da proveniência intra-atividade obtida pelo monitoramento dos diretórios e arquivos manipulados pelas atividades do workflow. Assim, informações não referenciadas na especificação do workflow são capturadas e relacionadas com proveniência previamente especificada.*

1. Introdução

A execução de experimentos científicos modelados como *workflows* produz resultados que necessitam de validação e que sejam passíveis de reprodução por terceiros [Freire *et al.* 2008]. Para tal, é necessário consultar o histórico das transformações dos dados desde sua origem até os resultados gerados na execução de um *workflow* científico. Esse histórico é conhecido como proveniência [Freire *et al.* 2008]. A proveniência pode ser classificada como prospectiva, quando relacionada à especificação do *workflow*, ou retrospectiva, quando associada à execução do *workflow* [Freire *et al.* 2008]. Podemos ainda subdividir a proveniência retrospectiva em explícita, que diz respeito aos elementos referenciados na configuração do *workflow*, e implícita, que trata de elementos não referenciados na definição do *workflow* [Marinho *et al.* 2011]. Por

exemplo, se uma atividade do *workflow* invoca um *script* externo que recebe como parâmetro um arquivo f_1, mas também modifica os arquivos f_2 e f_3, Sistemas de Gerência de *Workflows* Científicos (SGWfC), tais como SciCumulus [Oliveira *et al.* 2010], Vistrails [Callahan *et al.* 2006] e Kepler [Ludäscher *et al.* 2006], somente capturariam a alteração do arquivo f_1. Todo o histórico da modificação dos arquivos f_2 e f_3 será perdido. Neste caso temos um fluxo de dados implícito (*i.e.* proveniência retrospectiva implícita) em relação a f_2 e f_3.

Tomemos como exemplo concreto o *workflow* SciPhy da área de bioinformática, apresentado na Figura 1. O SciPhy possui um fluxo de dados implícito gerado durante sua execução. O SciPhy recebe como entrada sequências de DNA, RNA ou aminoácidos e gera uma árvore filogenética, que representa a semelhança entre organismos [Ocaña *et al.* 2011]. Conforme pode ser observado na Figura 1, os arquivos *MFS*, *MAS*, *PH*, *Modelo*, *EM*, e *Árvore Filogenética* são acessados e/ou modificados por uma ou mais atividades. Intuitivamente, uma primeira estratégia de captura da proveniência seria verificar a transformação de cada arquivo no final da execução da atividade que o manipula. Essa estratégia é interessante do ponto de vista prático, mas pode levar à perda de informações importantes. Por exemplo, conforme requisitos pré-determinados pelo cientista, a atividade *Alinhamento de Sequência* poderia organizar as sequências no formato *Fasta* (arquivo *MFS*) por tamanho e/ou eliminar algumas delas (*i.e.*, aquelas repetidas ou muito similares) antes de alinha-las (na atividade *Alinhamento de Sequência*). Nesse caso, o SGWfC não estaria ciente dessa modificação. Outro exemplo que pode ser observado é na leitura e manipulação do arquivo *EM* pela atividade *Busca do Modelo Evolutivo*. Nesta atividade podem ocorrer duas operações sobre o mesmo arquivo: uma operação de limpeza sobre os dados do arquivo e outra operação de gravação do modelo evolutivo extraído do arquivo *Modelo*. Assim, em uma estratégia de coleta de dados de proveniência intra-atividade, poderíamos coletar não somente a última transformação do arquivo (*i.e.*, última versão do arquivo produzida pela atividade), mas também todo o fluxo implícito (*i.e.*, sequência de modificações) ocorrido durante a execução de cada atividade.

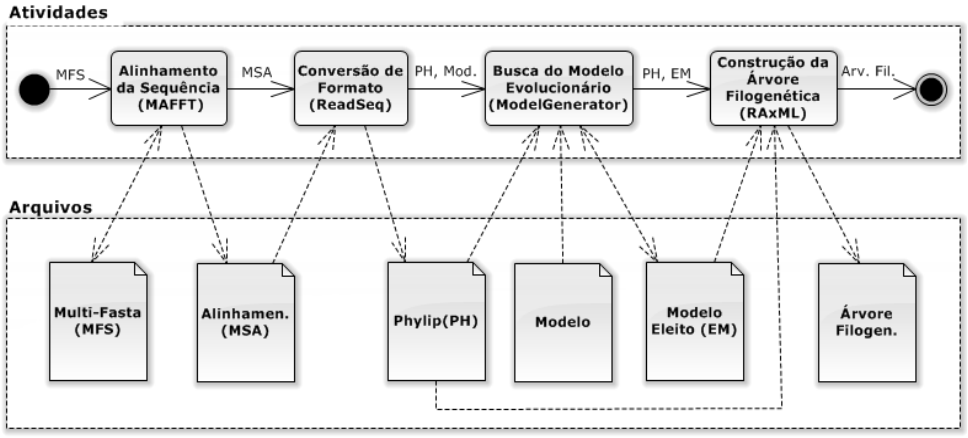


Figura 1. Atividades e arquivos de uma execução do *workflow* SciPhy

Esta captura da proveniência intra-atividade permite coletar e analisar a proveniência gerada sob as transformações de todos os arquivos consumidos e produzidos por cada atividade em um determinado diretório raiz, em seus subdiretórios ou em um espaço de trabalho gerenciado (*workspace*). Este registro em um nível mais detalhado possibilita entender melhor o funcionamento interno da atividade (mesmo

sem conhecer seu código fonte) e verificar a compatibilidade do processamento dos dados com os objetivos da atividade. Após a análise dos dados, é possível corrigir eventuais erros na configuração da atividade, definindo melhor seus dados de entrada ou a ordem de execução de cada operação de criação, modificação ou exclusão.

Para capturar a proveniência, algumas abordagens têm utilizado mecanismos de Gerência de Configuração de *Software* (GCS) [Koop *et al.* 2010; Neves *et al.* 2013]. A Gerência de Configuração é a área responsável pela gerência da evolução do *software* [Estublier 2000]. Estas abordagens atuam no máximo na coleta da proveniência obtida sob a perspectiva da atividade (unidade de versionamento). Entretanto, não permitem capturar todas as transformações dos artefatos manipulados durante a execução de cada atividade, conforme ocorre na execução do SciPhy (Figura 1). Apenas o resultado final da execução de uma atividade é capturado.

Outras abordagens [Frew *et al.* 2008; Holland *et al.* 2008] trabalham sobre os dados de proveniência em um grão mais fino que a atividade do *workflow*, coletando a proveniência a partir dos processos de sistema operacional que manipulam arquivos e através do próprio sistema de arquivos. Entretanto, a proveniência coletada fica desconectada dos dados de proveniência prospectivos (representação conceitual do *workflow*). Associar esses dados à proveniência prospectiva pode não ser uma tarefa trivial. Além disso, esse tipo de trabalho pode ser computacionalmente intenso, uma vez que o sistema operacional trabalha sobre múltiplos processos de forma simultânea.

Dessa forma, esse artigo apresenta uma abordagem de captura da proveniência retrospectiva intra-atividade gerada pelo fluxo de dados implícitos ou explícitos durante toda a execução de cada atividade, e não apenas ao final de cada atividade. A solução proposta, denominada ProvWatcher, relaciona a proveniência capturada com a proveniência prospectiva e retrospectiva sob a gerência dos SGWfCs. Para tal, ProvWatcher monitora diretórios e arquivos manipulados pelas atividades do workflow e registra as modificações em arquivos em um banco de dados relacional em tempo real, o que possibilita consultá-las e analisá-las durante o curso de execução do *workflow*. A abordagem proposta nesse artigo captura a proveniência implícita de forma análoga a proposta por Neves *et al.* [2013], entretanto captura as edições parciais dos arquivos feitos por cada atividade do *workflow*. Dessa forma, pode ser obtido um histórico mais detalhado sobre a edição de arquivos durante a execução de cada atividade.

O artigo está estruturado como segue. A Seção 2 oferece um referencial teórico sobre os conceitos relativos à proveniência de dados científicos. Na Seção 3 é apresentada a abordagem para a captura de proveniência sobre diretórios e arquivos. Um estudo de caso com o uso da abordagem é apresentado na Seção 4. A Seção 5 discute os trabalhos relacionados. Por fim, a Seção 6 conclui o artigo, apontando possibilidades de trabalhos futuros.

2. Proveniência de Dados Científicos

O termo "proveniência de dados científicos" pode ser definido como a origem ou linhagem do dado que ajuda a compreender o resultado de um experimento científico [Freire *et al.* 2008]. Com esse entendimento é possível obter resultados mais confiáveis e também garantir a reprodutibilidade do experimento. De acordo com Freire *et al.* [2008], a proveniência obtida de *workflows* pode ser classificada como: (i) *Proveniência Prospectiva*, que é aquela que captura a especificação de tarefas computacionais, e

corresponde aos passos a serem seguidos para chegar a um resultado, e (ii) *Proveniência Retrospectiva*, que é aquela que captura os passos executados e as informações sobre o ambiente utilizado para derivar um resultado, consistindo em um histórico estruturado e detalhado de uma execução de tarefas computacionais.

Freire *et al.* [2008] classifica também os mecanismos de captura de proveniência em três tipos: (i) Baseados em *workflow*, que monitoram o *workflow* como um todo, são construídos para SGWfC e dependentes deles (fortemente acoplados aos mesmos); (ii) Baseados em atividades, em que as atividades coletam sua própria proveniência. Esses mecanismos podem ser independentes de SGWfC, mas exigem adaptação das atividades do *workflow*. Por fim, os mecanismos de captura também podem ser (iii) Baseados em chamadas ao Sistema Operacional, que capturam somente a proveniência retrospectiva em grão fino, mas em geral não capturam a proveniência prospectiva, ou, quando o fazem, não relacionam proveniência prospectiva e retrospectiva. Devido à grande quantidade de informação gerada, o processo de captura e consulta pode ser complexo.

A proveniência retrospectiva também pode ser subdividida em explícita e implícita [Marinho *et al.* 2011]. A proveniência retrospectiva explícita refere-se às informações que foram devidamente explicitadas na especificação do *workflow*. Por outro lado, a proveniência retrospectiva implícita refere-se aos fluxos de dados implícitos, ou seja, os fluxos que não foram explicitados na especificação do *workflow*. Este tratamento de proveniência implícita permite obter informações como a identificação da atividade que criou ou modificou um determinado arquivo, a data em que o arquivo foi alterado, etc. A Figura 1 ilustra a ocorrência destes dois tipos de fluxos de dados: um fluxo de dados e processos que ocorre no domínio do SGWfC e é coletado por ele (proveniência retrospectiva explícita) e outro que ocorre no domínio do SO e que normalmente não é coletado pelo SGWfC (proveniência retrospectiva implícita). Assim, na especificação do SciPhy é definido que a atividade *Conversão de Formato* será executada logo após a atividade *Alinhamento da Sequência*, mas não há informação sobre o compartilhamento das informações contidas nos arquivos nem sobre os processos de edição dos arquivos que ocorrem durante a execução de cada uma delas.

3. ProvWatcher: Captura da Proveniência Intra-Atividade

A análise do processo de transformação dos dados gerados e consumidos por uma atividade é de fundamental importância para sua manutenção e evolução. Dado que uma atividade pode manipular um ou vários arquivos e cada um destes arquivos pode ser modificado mais de uma vez durante a execução do *workflow*, faz-se necessário realizar o monitoramento dos diretórios onde estes dados se encontram. Assim, para monitorar e recuperar estas informações, foi desenvolvida a abordagem chamada ProvWatcher. Nessa abordagem o diretório onde um determinado *workflow* é executado é monitorado em tempo real durante a execução de cada uma de suas atividades. A cada novo evento de criação, alteração ou exclusão de arquivos, os dados de proveniência são capturados, independente se são referenciados na especificação do workflow ou não. Esses dados (nome do arquivo, diretório, tipo de modificação e data) são armazenados no repositório de proveniência e relacionados à atividade que está sendo executada naquele momento.

A Figura 2 exibe uma visão geral do ProvWatcher. Nessa visão geral, o *SGWfC* gerencia o *workflow* científico e realiza a coleta da proveniência prospectiva e retrospectiva explícita. O *Repositório de Proveniência* armazena a proveniência

prospectiva e retrospectiva capturada pelo *SGWfC* e a proveniência retrospectiva implícita capturada pelo componente *ProvWatcher*. As setas pontilhadas indicam a comunicação entre os componentes da arquitetura em diferentes domínios (*Domínio do Workflow* e *Domínio do Sistema de Arquivos*), enquanto as setas cheias indicam a comunicação entre os componentes da arquitetura dentro do mesmo domínio. O *Diretório Raiz* é onde se encontram os arquivos manipulados pelos programas mapeados na especificação do *workflow*. Os *Arquivos 1 ... N* representam os arquivos de diversos formatos manipulados pelos programas definidos pelo *SGWfC*.

O *ProvWatcher* é responsável pelo monitoramento do *Diretório Raiz*, pela coleta da proveniência retrospectiva implícita e pela associação da proveniência implícita com a proveniência capturada pelo *SGWfC*. Finalmente, o *ProvDB* realiza a leitura e escrita de dados no *Repositório de Proveniência*. Apesar do *ProvWatcher* ser independente do *SGWfC*, seu componente *ProvDB* é desenvolvido conforme o modelo do repositório de dados do *SGWfC* em questão. O *Repositório de Proveniência* utilizado na arquitetura pode adotar ou não padrões como o PROV [Moreau and Missier 2013] ou OPM [Moreau *et al.* 2011]. Caso seja necessário realizar a conversão de dados de proveniência para o padrão PROV, ferramentas de intercâmbio de proveniência entre SGWfC heterogêneos podem ser utilizadas [Oliveira *et al.* 2014].

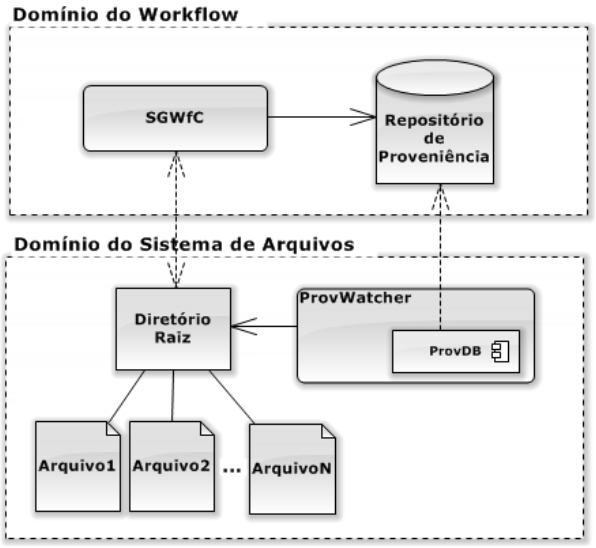


Figura 2. Arquitetura da abordagem ProvWatcher

O *ProvWatcher* e o *ProvDB* foram desenvolvidos na linguagem Java devido ao fato dela ser multiplataforma e possuir uma API própria para notificação de mudanças de arquivos denominada *WatchService* [Oracle 2014]. Essa API opera sobre a *Java Virtual Machine* (JVM) e fornece uma abstração do sistema de arquivos e do SO utilizado, facilitando a portabilidade da aplicação entre diferentes SO. Por outro lado, ela não é capaz de capturar acessos de leitura aos arquivos e diretórios. Durante o monitoramento iterativo dos diretórios do *workflow,* para cada notificação disparada pela API *WatchService*, o *ProvWatcher* realiza uma chamada ao *ProvDB* por meio de sua interface. Em seguida, o *ProvDB* realiza a leitura do repositório de proveniência e retorna o identificador do *workflow* e da atividade em execução. Após a consulta, o *ProvWatcher* combina os dados de proveniência retrospectiva explícita com os dados de proveniência implícita recuperados no monitoramento do diretório. Por fim, o *ProvWatcher* envia os dados combinados para o componente *ProvDB* que os armazena

na tabela *changefile* criada pelo *ProvDB* no início da sua execução. A tabela *changefile* é uma extensão adicionada pelo *ProvDB* ao modelo de proveniência utilizado pelo *SGWfC* (se o SGWfC também representar a proveniência usando o modelo relacional).

Para ilustrar o acoplamento do *ProvDB* com um *SGWfC*, a Figura 3 exibe o relacionamento da tabela *changefile* com as tabelas *hworkflow* e *hactivity*, utilizadas pelo *SGWfC* SciCumulus para armazenar proveniência. A tabela *hworkflow* armazena dados sobre os workflows, enquanto a tabela *hactivity* armazena dados sobre as atividades dos workflows. A Tabela 1, Tabela 2 e a Tabela 3 descrevem os atributos das relações *changefile, hworkflow e hactivity,* respectivamente.

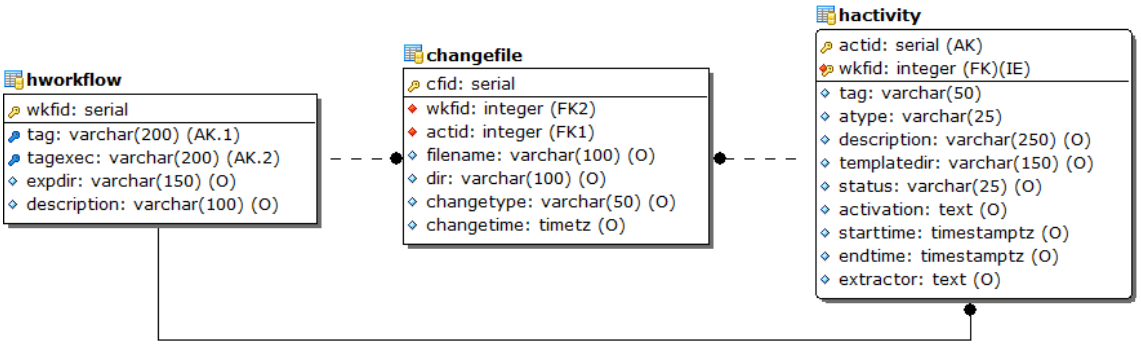


Figura 3. Tabelas hworkflow, changefile e hactivity com seus atributos e relacionamentos.

A estrutura do *ProvWatcher* permite o monitoramento de arquivos de diferentes SGWfC por meio da integração do componente de leitura e escrita (*ProvDB*), desenvolvido conforme o esquema do repositório de dados de proveniência em questão. Desta forma, através de diferentes configurações do componente *ProvDB*, nossa abordagem pode ser utilizada para capturar a proveniência implícita de variados *SGWfC*. Assim, torna-se possível capturar a proveniência retrospectiva intra-atividade, combiná-la com outros tipos de proveniência (retrospectiva explícita e prospectiva) e disponibilizá-la para uma análise mais profunda e fiel ao histórico de mudanças ocorridas durante a execução de cada atividade. Para realizar a análise dos dados de proveniência gerados, pode-se criar consultas em SQL que retornam informações de proveniência de diferentes tipos e granularidades, devidamente relacionadas entre si.

Tabela 1. Descrição dos atributos da tabela *changefile*.

Atributo	Descrição
cfid	Identificador da mudança ocorrida no arquivo
wkfid	Identificador do *workflow* em execução ou executado
actid	Identificador da atividade em execução ou executada
filename	Nome do arquivo modificado
dir	Diretório onde ocorreu a modificação
changetype	Tipo de modificação ocorrida
changetime	Data e horário da modificação

Tabela 2. Descrição dos atributos da tabela *hworkflow*.

Atributo	Descrição
wkfid	Identificador do *workflow* em execução ou executado
tag	Nome do *workflow*
tagexec	Nome da execução do *workflow*
expdir	Diretório onde os dados produzidos serão armazenados
description	Descrição do *workflow*

Tabela 3. Descrição dos atributos da tabela *hactivity*.

Atributo	Descrição
actid	Identificador da atividade em execução ou executada
wkfid	Identificador do *workflow* em execução ou executado
tag	Nome da atividade
atype	Tipo de atividade
description	Descrição da atividade
templatedir	Diretório do *template* de script da atividade
status	Estado atual da atividade em execução ou executada
activation	Comando ou script de execução da atividade
starttime	Data e hora de início da execução da atividade
endtime	Data e hora de término da execução da atividade
extractor	Nome e diretório do programa extrator de dados

4. Estudo de Caso

Para avaliar a viabilidade da abordagem ProvWatcher foi desenvolvido um estudo de caso utilizando o mecanismo de execução de *workflow* em ambiente de nuvem denominado SciCumulus [Oliveira *et al.* 2010]. Além de executar o *workflow* em paralelo, o SciCumulus captura os dados de proveniência durante a execução de cada atividade. Estes dados são persistidos em um banco de dados relacional PostgreSQL e, a partir dele, consultas em SQL podem ser realizadas para a obtenção de informações sobre a origem dos resultados e suas correlações.

No experimento aqui apresentado, o SciCumulus executou o *workflow* SciPhy num ambiente real centralizado. Um diretório raiz denominado *exp_SciPhy* foi criado no diretório raiz ("*/*") do sistema operacional Linux (Ubuntu 13.04). Os arquivos utilizados pelo *workflow* SciPhy foram então transferidos de um servidor remoto para o diretório local "*exp_SciPhy"*. Para cada atividade em execução, o SciCumulus altera o estado (campo *status* de *hactivity* na Figura 3) de cada atividade no banco de dados de proveniência para "RUNNING" em seu início, e esse estado é novamente alterado para "FINISHED" ao fim da execução. Para capturar a proveniência implícita gerada, foi necessário registrar o diretório no ProvWatcher e configurar o componente ProvDB para a conexão e acesso ao banco de dados do SciCumulus.

Assim que o SciPhy é executado, o ProvWatcher inicia o processo de monitoramento e captura dos eventos de manipulação dos arquivos utilizando o método não bloqueante *take()* da API *WatchService*. Quando um novo evento é capturado, o ProvWatcher faz uma chamada ao componente ProvDB, solicitando o identificador do *workflow* e da atividade em execução (*wkfid* e *actid,* respectivamente). Então, o ProvDB consulta a base de dados do SciCumulus, retorna as informações solicitadas pelo ProvWatcher e cria uma nova tabela (*changefile*) para armazenar a proveniência capturada (somente na primeira iteração). Em seguida, o ProvWatcher relaciona as informações capturadas (nome do arquivo, diretório, tipo de modificação e data e horário) com as informações da atividade que está sendo executada naquele momento (com estado "RUNNING" na tabela *hactivity* do SciCumulus) e passa estas informações ao ProvDB. Após a obtenção destes dados, o ProvDB persiste-os na tabela *changefile*. Todo processo é repetido até o final da execução do *workflow*.

Durante a execução do SciPhy, os cientistas podem realizar consultas SQL sobre o banco de dados para obter informações sobre a manipulação de dados pelos processos que estão sendo executados. A Tabela 4 exibe dois exemplos de consulta que podem ser

executados sobre a base de dados do SciCumulus em tempo de execução. A consulta do item 1 retorna informações da atividade em execução e das modificações realizadas até aquele momento sobre os respectivos arquivos utilizados. Além de obter informações importantes para o entendimento dos processos internos de cada atividade, este tipo de consulta também pode apoiar o processo de direcionamento do *workflow,* conforme a análise realizada sobre os mesmos em tempo de execução. A consulta do item 2 retorna os arquivos que foram modificados mais de uma vez. Esse tipo de consulta não pode ser executado nas abordagens atuais. Nela podemos identificar quais arquivos foram modificados ao longo da execução das atividades, independente da especificação do workflow. Assim, o cientista pode depurar o *workflow*, a fim de verificar inconsistências e prováveis pontos de melhoria. Esta consulta também nos permite obter a proveniência em diferentes grãos, relacionando a proveniência implícita capturada no nível intra-atividade (*changefile*) com a proveniência prospectiva e retrospectiva no nível da própria atividade (*hactivity*) e do workflow (*hworkflow*). O resultado obtido pela execução do ProvWatcher e o retorno das consultas acrescentam um maior nível de detalhamento sobre a proveniência retrospectiva, uma vez que estas informações não eram capturadas e disponibilizadas pelo SciCumulus. Assim, os dados gerados pelo monitoramento do ProvWatcher puderam satisfazer as consultas propostas possibilitando a validação do experimento executado.

Tabela 4. Consultas sobre a base de dados de proveniência

Item	Consulta
1	**Select** w.wkfid. w.tag, a.actid, a.tag, a.status, c.filename, c.dir, c.changetype, c.changetime **from** hactivity a, h*workflow* w, **changefile c** **where** a.status = 'RUNNING' **and** w.wkfid = a.wkfid **and** a.actid = c.actid **order by** c.changetime
2	**Select** w.wkfid, w.tag, w.tagexec, c.filename, count(c.filename) **from** h*workflow* w, hactivity a, **changefile c** **where** w.wkfid = a.wkfid **and** a.actid = c.actid **and** w.wkfid = 90 **group by** w.wkfid, w.tag, w.tagexec, c.filename **HAVING COUNT**(c.filename)>1

5. Trabalhos Relacionados

Nas abordagens PASS [Holland *et al.* 2008] e ES3 [Frew *et al.* 2008], a proveniência é capturada no nível do SO. Porém, apesar de coletar informações sobre a proveniência retrospectiva implícita, estas abordagens recuperam uma grande quantidade de informações que dificultam o trabalho de análise. Além disto, elas dependem de um sistema operacional específico para serem executadas e exigem um pós-processamento para interligar a proveniência implícita com a proveniência retrospectiva. Por outro lado, ProvWatcher captura a proveniência retrospectiva implícita e a interliga à proveniência prospectiva por meio do relacionamento existente entre a tabela *changefile* e as tabelas que representam os workflows e as atividades no banco de dados de proveniência do SGWfC (*hworkflow* e *hactivity* no caso do SciCumulus). O processo de monitoramento da nossa abordagem coleta os dados sobre diretórios e arquivos diretamente relacionados à atividade que está sendo executada naquele momento. Isto

implica numa quantidade menor de informações, porém mais especializadas, favorecendo o processo de análise.

Algumas abordagens têm utilizado técnicas e ferramentas de Gerência de Configuração de *Software* (GCS) para coletar a proveniência retrospectiva [Koop *et al.* 2010, Neves *et al.* 2013]. Koop *et al.* [2010] propõem um arcabouço que utiliza o GIT com o objetivo de criar ligações fortes (chamados de *strong links*) entre o experimento e seus dados de proveniência. Nesta abordagem é necessário informar na especificação do *workflow* (proveniência prospectiva) quais arquivos deverão ser capturados, o que é dispensado na abordagem proposta nesse artigo. Assim, Koop *et al.* [2010] capturam a proveniência retrospectiva explícita, enquanto nossa abordagem é capaz de capturar a proveniência retrospectiva implícita.

Neves *et al.* [2013] apresentam uma proposta de monitoramento através do versionamento de artefatos após a execução de cada atividade. Nesta abordagem, Neves *et al.* [2013] utilizam o sistema de gerência de configuração GIT. Com ele um *workspace* é criado para armazenar os arquivos (*clonados* a partir do repositório central) que serão gerados e/ou consumidos pelas atividades que compõe o *workflow*. Cada execução de um *workflow* é associada a um novo ramo e no final da execução de cada atividade um *commit* é disparado para gravar todos os eventos de leitura e escrita sobre os dados, gerando novas versões. Após a execução de todas as atividades o comando *push* é executado para retornar os dados modificados ao repositório central. Isto possibilita a obtenção e rastreamento do histórico dos arquivos no contexto de cada atividade através de uma análise *intra* (diferentes atividades sob um *trial*) e *inter-trial* (mesma atividade em diferentes *trials*) [Neves *et al.* 2013]. Por outro lado, este histórico não traz o detalhamento do que aconteceu durante a execução da atividade, pois captura somente a última modificação, sobre os artefatos, realizada por cada atividade. O ProvWatcher, por sua vez, captura as modificações (criação, alteração e exclusão) dos arquivos durante a execução de cada atividade (intra-atividade) sem identificar as alterações internas (linhas) do arquivo como ocorre no trabalho de Neves *et al.* [2013].

6. Conclusão e Trabalhos Futuros

A abordagem de captura de proveniência ProvWatcher faz uso de monitoramento para verificar as alterações sobre um diretório que armazena dados de entrada e saída gerados, alterados ou consumidos por atividades de *workflows*. Cada ação realizada sobre um determinado arquivo é persistida em um banco de dados relacional para a realização de consultas durante e/ou após a execução do *workflow*. Esta abordagem permite que o cientista tenha um maior controle sobre o fluxo de informações de entrada e saída dentro de cada atividade. Ela também garante um controle em granularidade mais fina do que a própria atividade, uma vez que podemos conhecer as ações realizadas no decorrer de sua execução.

A abordagem proposta nesse artigo foca em capturar informações de proveniência relacionadas às operações de criação, modificação e exclusão de arquivos, (uma vez que a API *WatchService* não fornece apoio a operações de acesso) e relaciona-las aos dados de proveniência prospectiva e retrospectiva do *workflow*. Encontra-se em andamento o desenvolvimento de uma nova versão do ProvWatcher onde serão contemplados os acessos de leitura aos arquivos de dados, além da captura do conteúdo das versões intermediárias dos arquivos.

Agradecimentos. Os autores agradecem ao CNPq e à FAPERJ pelo financiamento parcial deste trabalho.

Referências

Callahan, S. P., Freire, J., Santos, E., et al. (2006). VisTrails: Visualization Meets Data Management. In *SIGMOD*. ACM.

Estublier, J. (2000). Software configuration management: a roadmap. In *Proceedings of the Conference on The Future of Software Engineering*. ACM.

Freire, J., Koop, D., Santos, E. and Silva, C. T. (2008). Provenance for Computational Tasks: A Survey. *Computing in Science Engineering*, v. 10, n. 3, p. 11–21.

Frew, J., Metzger, D. and Slaughter, P. (2008). Automatic capture and reconstruction of computational provenance. *Concurrency and Computation: Practice and Experience*, v. 20, n. 5, p. 485–496.

Holland, D. A., Seltzer, M. I., Braun, U. and Muniswamy-Reddy, K.-K. (2008). PASSing the provenance challenge. *Concurrency and Computation: Practice and Experience*, v. 20, n. 5, p. 531–540.

Koop, D., Santos, E., Bauer, B., et al. (2010). Bridging Workflow and Data Provenance Using Strong Links. In: Gertz, M.; Ludäscher, B. [Eds.]. *Scientific and Statistical Database Management*. LNCS. Springer Berlin Heidelberg. p. 397–415.

Ludäscher, B., Altintas, I., Berkley, C., et al. (2006). Scientific workflow management and the Kepler system. *Concurrency and Computation: Practice and Experience*, v. 18, n. 10, p. 1039–1065.

Marinho, A., Mattoso, M., Werner, C., Braganholo, V. and Murta, L. (2011). Challenges in managing implicit and abstract provenance data: experiences with ProvManager. In *TaPP*. USENIX Association.

Moreau, L., Clifford, B., Freire, J., et al. (2011). The Open Provenance Model core specification (v1.1). *Future Generation Computer Systems*, v. 27, n. 6, p. 743–756.

Moreau, L. and Missier, P. (2013). PROV-DM: The PROV Data Model. http://www.w3.org/TR/2013/REC-prov-dm-20130430/, [acessado em 17 de fevereiro de 2014].

Neves, V. C., Braganholo, V. and Murta, L. (2013). Implicit Provenance Gathering through Configuration Management. In *SE-CSE*. IEEE.

Ocaña, K. A. C. S., Oliveira, D. De, Ogasawara, E., et al. (2011). SciPhy: A Cloud-Based Workflow for Phylogenetic Analysis of Drug Targets in Protozoan Genomes. In: Souza, O. N. De; Telles, G. P.; Palakal, M. [Eds.]. *Advances in Bioinformatics and Computational Biology*. LNCS. Springer Berlin Heidelberg. p. 66–70.

Oliveira, D., Ogasawara, E., Baião, F. and Mattoso, M. (2010). SciCumulus: a lightweight cloud middleware to explore many task computing paradigm in scientific workflows. In *3rd International Conference on Cloud Computing*. IEEE.

Oliveira, W., Oliveira, D. and Braganholo, V. (2014). Experiencing PROV-Wf for Provenance Interoperability in SWfMSs. In *IPAW*.

Oracle (2014). Watching a Directory for Changes. http://docs.oracle.com/javase/tutorial/essential/io/notification.html, [acessado em 18 de maio de 2014].

paper:80

CARLO: Modelo Ontológico de Contexto para Recomendação de Objetos de Aprendizagem em Ambientes Pervasivos

Guilherme Medeiros Machado, José Palazzo Moreira de Oliveira

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

{guimmachado,palazzo}@inf.ufrgs.br

Abstract. *Pervasive environments are places that have abundant resources and as a consequence the analysis and selection of a small set of interesting resources may become a difficult task. Learning objects are important resources for teaching and so are the targets resources of the recommendation process proposed here. To recommend such objects an ontological context model consisting of four dimensions of information and a rule-based filter that allows the performance of recommendation directly in the model is presented. To validate this model, we have developed a system for Android platforms and a scenario-based case study is presented.*

Resumo. *Ambientes pervasivos são locais que possuem abundante quantidade de recursos e por conta disso a análise e seleção de um conjunto pequeno de recursos de interesse pode tornar-se uma tarefa difícil. Objetos de aprendizagem são recursos importantes para o ensino e por isso são os recursos alvos do processo de recomendação aqui proposto. Para recomendar tais objetos é apresentado um modelo ontológico de contexto, composto por quatro dimensões de informação e um filtro baseado em regras que possibilita a realização da recomendação diretamente no modelo. Para validar tal modelo, além de ter-se desenvolvido um sistema para plataformas Android, é apresentado um estudo de caso, baseado em cenário.*

1. Introdução

Ambientes pervasivos são locais onde usuários móveis utilizam elementos computacionais integrados em suas vidas que muitas vezes passam despercebidos, além disso, tais ambientes proporcionam fácil comunicação entre dispositivos e disponibilizam informações a seus usuários em qualquer lugar e a qualquer momento [Ark and Selker 1999]. Além de recursos computacionais, tais ambientes são compostos por recursos não computacionais e pessoas que utilizam ambos os tipos de recursos. Um exemplo de ambiente que pode se tornar pervasivo é o campus universitário e um dos recursos mais abundantes nesse ambiente são os objetos de aprendizagem, definidos como qualquer entidade, digital ou não, que pode ser utilizada para aprendizagem, educação ou treinamento. Devido a sua importância, objetos de aprendizagem são utilizados como recursos alvos do problema que este trabalho se propõe a resolver.

Com o intuito de auxiliar usuários a descobrirem objetos de aprendizagem que estejam próximos, que correspondam a seus interesses e guiá-los até tais objetos, é proposto e implementado neste trabalho um modelo de contexto que contempla quatro dimensões de informação encontradas na literatura. Além dessas dimensões o modelo proposto integra um filtro descrito em formato de regras SWRL (*Semantic Web Rule*

Language) que tem por objetivo torná-lo apto a selecionar apenas os objetos de interesse próximos ao usuário. Tal modelo é validado através de um estudo de caso implementado com base em um cenário. O restante deste artigo está organizado da seguinte maneira, a Seção 2 apresenta alguns conceitos fundamentais, a Seção 3 apresenta alguns trabalhos relacionados e uma análise dos mesmos, a Seção 4 apresenta o modelo proposto e desenvolvido neste trabalho, a Seção 5 apresenta o filtro de objetos de aprendizagem desenvolvido e um cenário de uso e a Seção 6 apresenta as conclusões.

2. Conceitos

Dey define contexto como qualquer informação relevante para caracterizar a situação de uma entidade [Dey 2001]. Tal entidade é uma pessoa, lugar ou objeto que é relevante para a interação entre o usuário e a aplicação, incluindo os próprios usuários e aplicação. Ciente desta definição, Dourish propõe a classificação do contexto em duas vertentes, ora é encarado como um problema representacional ora como um problema interacional [Dourish 2004]. Como problema representacional o contexto é uma informação que é delineável, pode ser prevista em uma determinada situação e é estável, ou seja, os elementos de contexto não variam de instância para instância em uma mesma situação.

Como problema interacional o contexto é visto como uma propriedade relacional que acontece entre objetos ou atividades, ao invés de ser considerado um tipo de informação. Por conta disso, o contexto não é delineável, já que o escopo das características contextuais é definido dinamicamente. Neste trabalho o contexto é encarado como um problema representacional, já que nesta vertente é possível delimitar-se o escopo de seus elementos. Consequentemente podemos modelar mais facilmente tal informação para ser utilizada em uma aplicação.

Segundo Bettini as técnicas de modelagem de contexto podem ser classificadas em: i) baseadas em papéis-objetos; ii) baseadas em modelos espaciais e iii) baseadas em ontologias [Bettini et al. 2010]. A técnica baseada em papéis objetos originou-se da tentativa de criar modelos suficientemente formais para suportar o processamento de consultas e raciocínio. Uma das principais linguagens utilizadas nesta técnica é a *Context Modelling Language* (CML), que é uma linguagem gráfica suficientemente formal para ser mapeada diretamente para um sistema de gerenciamento de contexto. Contudo uma desvantagem desta linguagem é que ela apresenta um modelo de informação "raso" já que qualquer tipo de informação de contexto é representada da mesma maneira, não sendo possível, por exemplo, representar uma hierarquia. Na técnica baseada em modelos espaciais a informação de localização é o elemento principal de contexto, que pode estar associado a uma entidade física ou não física. A última técnica é baseada em ontologias, que são definidas por [Gruber 1993] como uma "especificação formal e explícita de uma conceituação compartilhada". A linguagem mais utilizada para a construção de ontologias é a *Web Ontology Language* (OWL), que em sua versão DL e Lite possibilitam o uso de motores de inferência que utilizam lógica descritiva e são capazes de checar a consistência do modelo desenvolvido, inferir novos relacionamentos entre classes e descobrir conhecimentos anteriormente implícitos. Além disso, modelos ontológicos de contexto são adequados para compartilhamento de conhecimento, já que eles provêm uma especificação formal de dados de contexto semânticos. Esta característica é particularmente importante em ambientes pervasivos, onde entidades diferentes devem interagir com o intuito de compartilhar informações de contexto do usuário.

3. Trabalhos Relacionados

Modelos de contexto para ambientes pervasivos são temas centrais de vários trabalhos encontrados na literatura. Tais trabalhos podem ser classificados de acordo com o objetivo de aplicação dos modelos em: i) modelos de alto nível e ii) modelos aplicados. Os modelos de alto nível têm por objetivo dar suporte à construção dos modelos aplicados, por isso são dotados de conceitos mais genéricos e que podem ser utilizados por várias aplicações de um mesmo domínio. Os modelos aplicados, em contrapartida, são mais específicos e dificilmente são totalmente utilizados por várias aplicações.

O trabalho de [Chen et al. 2004] apresenta uma ontologia a ser utilizada por aplicações pervasivas, batizada de *Standard Ontology for Ubiquitous and Pervasive Applications* (SOUPA) a ontologia é composta de dois conjuntos distintos de ontologias relacionadas, o *SOUPA Core* (de alto nível) e o *SOUPA Extension* (aplicado). As categorias de informação manipuladas pelas ontologias do conjunto *Core* são relativas à pessoa, políticas e ações, agentes, tempo, espaço e eventos. As categorias manipuladas pelas ontologias do conjunto *Extension* são relativas à reuniões e agenda, documentos e documentos digitais, captura de imagem e localização.

O trabalho de [Wang et al. 2004] propõe um modelo ontológico de contexto de alto nível batizado de *CONtext ONtology* (CONON), construída em OWL a ontologia modela quatro conceitos de alto nível principais, são eles, Pessoa, Localização, Atividade e Entidades Computacionais, dos quais podem derivar conceitos específicos.

O trabalho de [Pantsar-Syvaniemi et al. 2010] propõe um modelo ontológico de alto nível que tem por objetivo adaptar o comportamento de um sistema pervasivo. Os autores propõem a manipulação das informações de contexto em três níveis: i) físico, descreve informações do ambiente, tais como tempo, localização, equipamentos de rede; ii) digital, funde as informações vindas do primeiro nível a fim de descrever situações tais como "Acidente próximo", neste nível o modelo é instanciado e iii) humano, funde informações do nível anterior para encontrar relacionamentos entre situações.

O trabalho de [Hervás and Bravo 2011] propõe um modelo ontológico de contexto aplicado que tem por objetivo apoiar a adaptação de interfaces de software, utilizando para isto informações de ambientes pervasivos. O modelo é composto de 4 dimensões de informação: i) o usuário, incluindo sua situação atual (localização, atividades, papéis e objetivos) e seus relacionamentos sociais; ii) os dispositivos relevantes, com suas características, associações e dependências, iii) o ambiente físico, que define a distribuição espacial e iv) o serviço, relacionada às características relativas a visualização de informações.

O trabalho de [Al-Mekhlafi et al. 2009] propõe um modelo ontológico de contexto aplicado que tem por objetivo apoiar a recomendação de sentenças utilizadas no cotidiano de um estudante da língua chinesa de acordo com seu nível de aprendizado. O modelo é dividido em quatro dimensões: i) tempo, descreve o status do sistema (ativado, desativado ou sincronizando); ii) localização, descreve o lugar onde o estudante se encontra; iii) atividade, descreve as atividades planejadas na agenda do estudante e iv) nível, descreve o nível que um estudante pode estar no aprendizado da língua, bem como o nível de uma sentença armazenada no banco de dados.

Através da leitura e análise dos trabalhos apresentados anteriormente nota-se um padrão de categorias de informações modeladas pelas aplicações pervasivas. Todos os

trabalhos modelam pelo menos informações sobre o perfil do usuário, sobre o ambiente e sobre o domínio tecnológico.

Além do padrão de informações nota-se também que em nenhum dos trabalhos apresentados explicita suas "regras da aplicação" junto com o modelo ontológico de contexto. Os trabalhos limitam-se apenas a modelar informações de contexto e domínio na ontologia. As "regras da aplicação", que são o conjunto de ações que determinam o comportamento da aplicação frente a um determinado contexto, acabam sendo descritas somente no código fonte da aplicação.

A partir do momento que tais "regras da aplicação" são extraídas do código fonte e armazenadas no nível do modelo de contexto, este conhecimento torna-se explícito e entendível por agentes de software, como previsto no clássico artigo de [Berners-Lee et al. 2001], facilitando dentre outras coisas o compartilhamento dessas informações inclusive entre agentes que não foram projetados para trabalharem em conjunto. Tal característica tem fundamental importância em um ambiente pervasivo, onde elementos computacionais estão espalhados e devem a todo momento compartilhar informações e conhecimento entre si [Bettini et al. 2010].

Outra vertente a se revisar são os trabalhos que realizam recomendações de objetos de aprendizagem. Alguns trabalhos encontrados na literatura são [Bouzeghoub et al. 2009], [Pernas et al. 2012] e [Wang and Wu 2011]. Em todos eles os autores propõem a utilização de informações de contexto com o objetivo de realizar recomendações.

Nesses trabalhos a recomendação é realizada através da correspondência das características da pessoa e do objeto de aprendizagem. Entretanto, seus modelos de contexto poderiam ser estendidos para abarcar mais conceitos sobre o perfil do usuário e sobre os objetos de aprendizagem. Esse maior detalhamento do perfil e dos objetos de aprendizagem proporcionaria maior flexibilidade aos algoritmos e regras na escolha das características a serem correspondidas para realizar a recomendação.

4. Modelo Ontológico de Contexto para Recomendação de Objetos de Aprendizagem em Ambientes Pervasivos

O modelo ontológico de contexto desenvolvido neste trabalho tem por objetivo dar suporte à recomendação de objetos de aprendizagem em um ambiente pervasivo. Batizado de CARLO – *Model for **C**ontext-**A**ware **R**ecommendation of **L**earning **O**bjects* (Modelo para Recomendação Sensível ao Contexto de Objetos de Aprendizagem) o modelo leva em consideração o padrão de dimensões de informações encontrados nos trabalhos apresentados na Seção 3, além de acrescentar a dimensão de objetos de aprendizagem para realizar as recomendações dos mesmos. O modelo CARLO tem o usuário como seu elemento central e está dividido em quatro dimensões de informação contextuais i) informações sobre o perfil do usuário; ii) informações sobre localização; iii) informações sobre os elementos tecnológicos e iv) informações sobre os objetos de aprendizagem. Além dos elementos de domínio e de contexto, junto ao modelo CARLO são descritas regras semânticas (regras da aplicação) responsáveis pela seleção dos objetos de aprendizagem presentes no ambiente pervasivo. A união do modelo instanciado com as regras semânticas e um motor de inferências lógicas, forma a base de conhecimento utilizada para realizar a recomendação sensível ao contexto dos objetos de aprendizagem em um ambiente pervasivo.

Na Figura 1 é apresentada uma visão geral e resumida das classes do modelo CARLO, note que o usuário é o elemento central e o mesmo se relaciona com as quatro dimensões de informações contextuais identificadas. Um usuário tem um perfil, tem uma localização e usa uma tecnologia e por causa das características presentes nesses três relacionamentos ele terá um objeto de aprendizagem recomendado, fato exibido através da relação do tipo propriedade objeto "temOARecomendado".

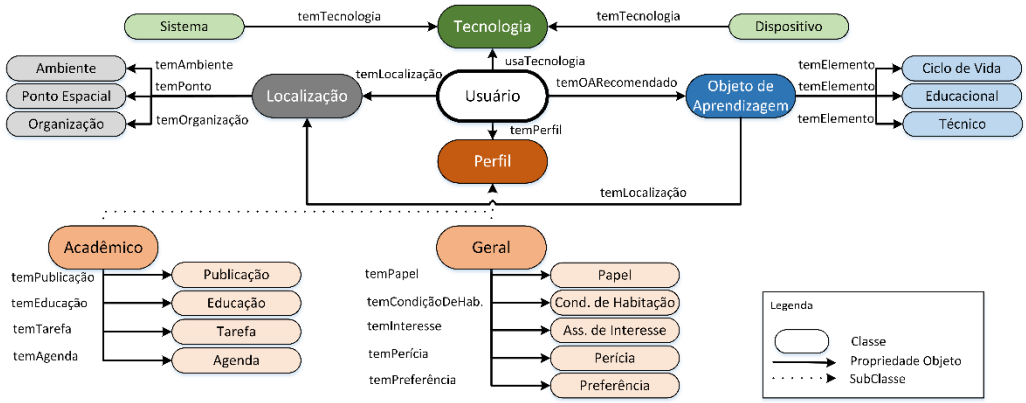


Figura 1. Visão geral sumarizada do modelo CARLO

Devido à complexidade em definir as informações pertinentes ao perfil do usuário o trabalho de Brusilovsky propõe cinco características populares e úteis em modelos de usuários de sistemas adaptativos, são elas: i) o conhecimento do usuário; ii) os interesses do usuário; iii) os objetivos e tarefas; iv) o plano de fundo do usuário; e v) os traços individuais [Brusilovsky and Millán 2007].

No modelo CARLO são apresentados conceitos relativos às características 'i', 'ii', 'iii' e 'iv', a característica 'v' não foi modelada por representar características intrínsecas do usuário, sendo assim um desafio além do escopo deste trabalho, já que tais características são de difícil identificação e podem variar muito de um usuário para outro. Por serem bastante amplas, quando implementadas, as características selecionadas resultaram em uma grande quantidade de conceitos ontológicos, por isso, optou-se por dividir-se a dimensão de perfil do usuário em "Perfil Acadêmico" e "Perfil Geral".

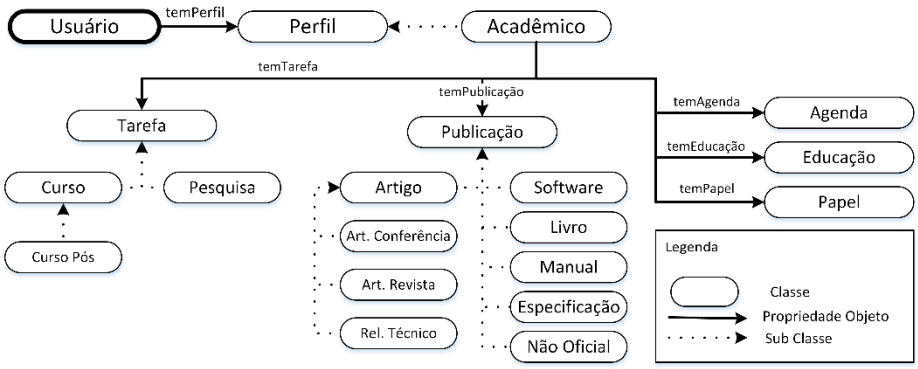


Figura 2. Modelo da subdimensão de informação do perfil acadêmico do estudante

Na Figura 2 é exibido o modelo da dimensão de informação do perfil do usuário, mais especificamente é exibida a subdimensão do perfil acadêmico. Nesta subdimensão são apresentados conceitos relativos às informações acadêmicas do usuário, tais como suas atividades e papel desempenhado dentro da universidade. Esta subdimensão apre-

senta conceitos que modelam as atividades do usuário dentro da universidade, a posição ocupada e sua formação acadêmica.

Na Figura 3 é exibida a subdimensão de informação do perfil geral do usuário. Nesta subdimensão são apresentados conceitos relativos a informações gerais do usuário, tais como habilidades, preferências e interesses. Esta subdimensão apresenta conceitos relativos às características de plano de fundo, gostos e conhecimentos do usuário.

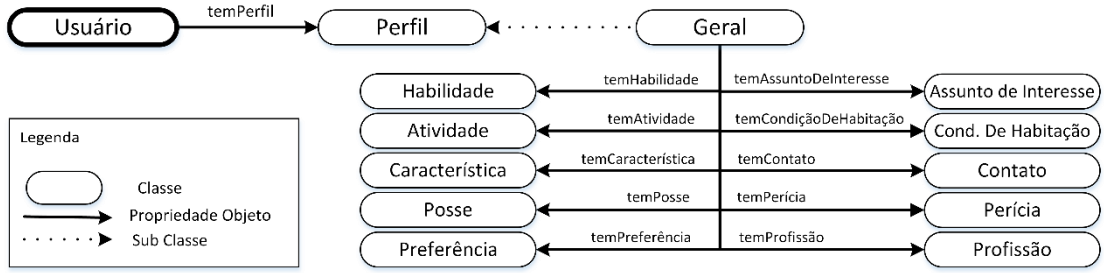


Figura 3. Modelo da subdimensão de informação do perfil geral do estudante

A união das duas subdimensões "Perfil Acadêmico" e "Perfil Geral" forma a dimensão do perfil do usuário. Pela importância da dimensão de perfil do usuário seus conceitos são exibidos na Tabela 1 com uma breve descrição e na coluna "Brusilovsky e Millán" é exibida a característica a qual tal conceito se refere.

Tabela 1. Descrição dos conceitos da dimensão de perfil do usuário e seu relacionamento com as características de Brusilovsky e Millán

Conceito	Descrição	Brusilovsky e Millán
Agenda	Agenda de compromissos do usuário	iii
Educação	Título de grau do usuário (Bsc.; Msc.; Dr.)	iv
Papel	Papel do usuário na universidade (ex. prof.; aluno)	iv
Publicação	Produção científica do usuário	iv
Tarefa	Itens que serão inseridos na agenda	iii
Habilidade	Habilidades e limitações físicas e mentais do usuário	iv
Atividade	Atividades e passatempos do usuário	ii
Característica	Características físicas do usuário	iv
Contato	Pessoas que o usuário se relaciona	iv
Perícia	Conhecimentos que o usuário demonstra maestria	i
Cond. de Habitação	Local e tipo de residência do usuário	iv
Assunto de Interesse	Interesses acadêmicos e passatempos do usuário	ii
Posse	Coisas vivas e não vivas, possuídas pelo usuário	iv
Preferência	Gostos do usuário	ii
Profissão	Ocupação profissional do usuário	Iv

As informações ambientais, ou de localização, constituem a segunda dimensão contextual do modelo. Esta dimensão é tão importante que Brusilovsky afirma que os sistemas sensíveis ao contexto para dispositivos móveis focam-se naturalmente na localização do usuário [Brusilovsky and Millán 2007]. A informação de localização é utilizada para determinar um conjunto pequeno de objetos de interesse próximo a localização do usuário [Brusilovsky and Maybury 2002]. Este subconjunto define o que deve ser apresentado ou recomendado ao usuário.

Na Figura 4 é exibida a dimensão de informação contextual de localização. Nesta dimensão são apresentados conceitos que têm por objetivo descrever a posição atual

do usuário, tais como suas coordenadas simbólicas e geométricas [Bettini et al. 2010], além da organização do ambiente.

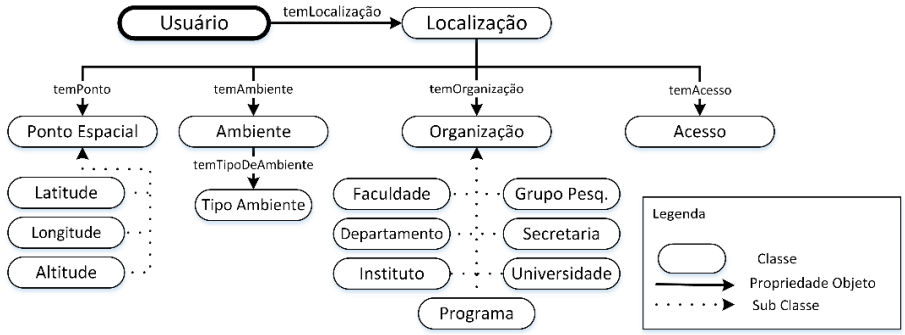


Figura 4. Modelo da dimensão de informação de localização

Os conceitos desta dimensão são: i) Ponto Espacial, descreve informações de GPS relativas a localização do usuário e do objeto de aprendizagem, para isso utiliza os conceitos de Latitude, Longitude e Altitude; ii) Ambiente, descreve os ambientes/áreas da universidade; iii) Tipo de Ambiente, descreve um ambiente como público ou privado; iv) Organização, descreve o papel desempenhado por um conjunto de ambientes da universidade; v) Acesso, descreve o tipo de acesso ao ambiente como permitido ou negado. A dimensão de localização se relaciona com o conceito Ambiente, Usuário e a dimensão de objetos de aprendizagem através da propriedade objeto "temLocalização".

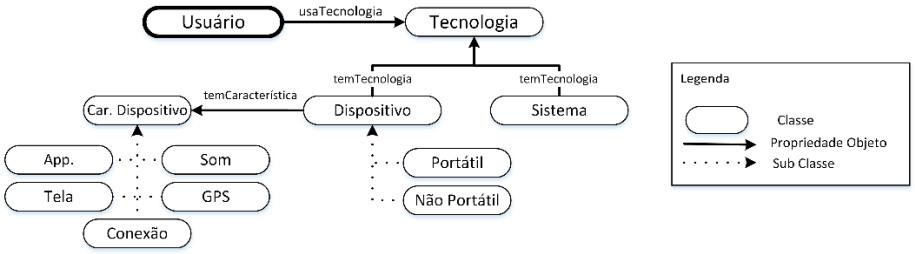


Figura 5. Modelo da dimensão de informação de tecnologia

Na Figura 5 são exibidos os conceitos modelados na dimensão de tecnologia são eles: i) Sistema, descreve o próprio sistema de recomendação e os demais sistemas presentes no ambiente, apresenta informações como nome do sistema e versão; ii) Dispositivo, descreve o dispositivo utilizado para acessar o sistema de recomendação, bem como todos os dispositivos espalhados pela universidade. É pai dos conceitos Portátil e 'Não Portátil'; iii) Características do Dispositivo, descreve características importantes do dispositivo móvel onde o sistema de recomendação está instalado. É pai dos conceitos Conexão (tipos de conexões disponíveis no dispositivo – importante caso o GPS venha a falhar o usuário pode ser localizado através do sinal Wi-fi), Som (modo sonoro do dispositivo – importante ao emitir alertas de objetos de aprendizagem recomendados), Tela (tamanho da tela – importante para adaptar a interface do sistema ao dispositivo atual) e GPS (modelo do GPS utilizado no dispositivo do usuário).

Na Figura 6 é exibida a dimensão de informação de objetos de aprendizagem, que descreve um objeto de aprendizagem através de nove conceitos definidos pelo padrão de metadados de objetos de aprendizagem IEEE LOM ["IEEE Standard for Learning Object Metadata" 2002]. Tais conceitos foram modelados a partir do documento definido pela IEEE.

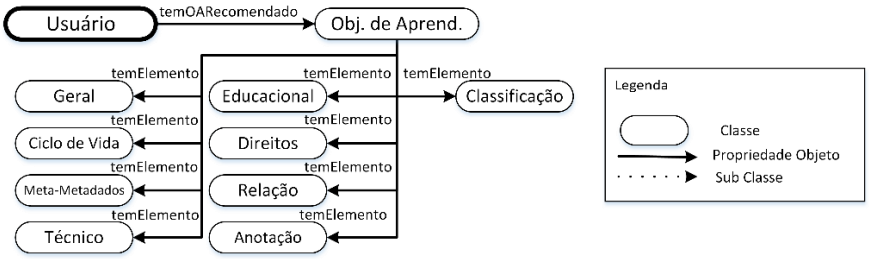


Figura 6. Modelo da dimensão de informação de objetos de aprendizagem

O modelo foi construído tomando por base, além do modelo de metadados LOM, os conceitos definidos na ontologia LUBM (*Lehigh University Benchmark*) [Guo et al. 2005] e na ontologia de perfil definida em [Golemati et al. 2007]. Nas ontologias LUBM e na de perfil foi realizado um alinhamento manual entre os conceitos equivalentes além de serem adicionados vários conceitos necessários para o domínio da aplicação do modelo que não estavam presentes nos modelos originais.

5. Caso de Uso

Com o intuito de demonstrar a execução de recomendações dos objetos de aprendizagem através do modelo CARLO, nesta seção é desenvolvido um caso de uso baseado em um cenário de aplicação. Neste cenário tem-se um estudante universitário William que necessita implementar um algoritmo de ordenação de vetores em Java, entretanto o estudante não tem conhecimento algum de tal linguagem. O estudante então receberá recomendações de objetos de aprendizagem próximos à sua localização que poderão auxiliar na execução desta tarefa. Com o intuito de tornar o modelo apto a realizar recomendações foi desenvolvido um filtro semântico, que utiliza regras escritas em SWRL (*Semantic Web Rule Language*) e faz uma correspondência entre as características do perfil do usuário, do objeto de aprendizagem e da localização de ambos. Tais regras são integradas ao modelo CARLO que após ser instanciado, pode ser utilizado junto a um motor de inferências lógicas, que raciocine sobre regras SWRL, para extrair recomendações de objetos de aprendizagem adequadas às preferências e ao contexto.

Em (1) é descrita a regra genérica aqui proposta, note que a regra considera um usuário '?u', uma característica do perfil desse usuário '?c' e verifica se o usuário está em uma localização '*?l*' e tem em seu perfil a característica '*?c*', após verificar o usuário a regra se preocupa em verificar se um objeto de aprendizagem '*?o*' está em uma localização '*?x*' (próxima à localização '*?l*') e se o objeto tem a mesma característica '*?c*' que o usuário. Caso todas essas condições sejam verdadeiras, o objeto de aprendizagem '*?o*' é então recomendado ao usuário '*?u*'.

$$usuário(?u) \wedge característicaASerCorrespondida(?c) \wedge temLocalização(?l,?u) \wedge temCaracterística(?c,?u) \wedge objetoDeAprendizagem(?o) \wedge temLocalização(?x,?o) \wedge temCaracterística(?c,?o) \rightarrow temOARecomendado(?o,?u) \quad (1)$$

Um exemplo prático de aplicação desta regra é exibido em (2), onde a característica do usuário selecionada para ser correspondida com a característica do objeto de aprendizagem é o Assunto de Interesse. Note que as instâncias de localização do usuário e do objeto de aprendizagem são diferentes, porém estão relacionadas a área próxima à biblioteca do campus.

$$usuário(?u) \wedge assuntoDeInteresse(?c)$$

$$\wedge\, temLocalização(PertoBiblioteca, ?u) \wedge temAssuntoDeInteresse(?c, ?u) \wedge objetoDeAprendizagem(?o) \wedge temLocalização(Biblioteca, ?o) \wedge temAssuntoDeInteresse(?c, ?o) \rightarrow temOARecomendado(?o, ?u) \quad (2)$$

Para ilustrar tal cenário o modelo CARLO, foi implementado em OWL (*Web Ontology Language*) com o auxílio do software *Protégé* e integrado a um sistema de recomendação de objetos de aprendizagem desenvolvido para plataformas *Android*. O motor de inferência utilizado para validar a consistência do modelo e raciocinar sobre suas regras foi o *Pellet*, por este apresentar mais expressividade em lógica descritiva, ser sólido, completo, justificar suas inferências, ser integrável ao *Jena*, compatível com o *Protégé*, ser implementado em *Java* e ser *Open Source* [Dentler et al. 2011].

Os objetos de aprendizagem extraídos do modelo são apresentados ao usuário em uma interface que os agrupa em prédios próximos à localização do usuário. Desta forma diminui-se a frequência que o usuário é notificado sobre recomendações dos objetos de aprendizagem, pois ao invés de notificá-lo a cada vez que um objeto relevante é encontrado, o usuário somente será notificado quando um grupo de objetos estiver disponível em um prédio próximo. Diminuindo tais notificações, contribui-se para diminuir a sobrecarga de recomendações dadas a um único usuário. Uma das vantagens ao se considerar o contexto do usuário no processo de recomendação é que ao invés de se apresentar todos os objetos de aprendizagem que o usuário possa ter interesse, apresentam-se somente os objetos que ele tem interesse no contexto atual, refinando assim o processo de seleção dos objetos.

6. Conclusão

Ambientes pervasivos são locais cheios de recursos computacionais e não computacionais. Por conta da grande quantidade, a análise e seleção apenas dos recursos que sejam de interesse do usuário torna-se uma tarefa difícil. Este trabalho apresenta um modelo ontológico de contexto a ser utilizado em sistemas de recomendação de objetos de aprendizagem em ambientes pervasivos. O modelo se apresenta dividido em quatro dimensões de informações, encontradas na literatura. Além das dimensões de informação o modelo também integra um filtro baseado em regras para seleção dos objetos de aprendizagem relevantes ao contexto do usuário. Por conta deste filtro é possível realizar a recomendação dos objetos de aprendizagem diretamente do modelo de contexto. Tais regras utilizam as características do perfil do usuário e as correspondem com as características dos objetos de aprendizagem próximos do mesmo. Um estudo de caso, baseado em cenário, é apresentado para validar o modelo e o filtro de objetos de aprendizagem desenvolvido. Alguns trabalhos futuros são a integração de estilos cognitivos ao modelo com o intuito de refinar as recomendações e a preocupação com aspectos de privacidade e segurança.

Referências

Al-Mekhlafi, K., Hu, X. and Zheng, Z. (2009). An Approach to Context-Aware Mobile Chinese Language Learning for Foreign Students. In *Int. Conf. on Mobile Business*.

Ark, W. S. and Selker, T. (1999). A look at human interaction with pervasive computers. *IBM Systems Journal*, v. 38, n. 4, p. 504–507.

Berners-Lee, T., Hendler, J., Lassila, O. and Others (2001). The semantic web. *Scientific american*, v. 284, n. 5, p. 28–37.

Bettini, C., Brdiczka, O., Henricksen, K., et al. (2010). A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, v. 6, n. 2, p. 161–180.

Bouzeghoub, A., Do, K. N. and Wives, L. K. (2009). Situation-Aware Adaptive Recommendation to Assist Mobile Users in a Campus Environment. In *Int. Conf. on Advanced Information Networking and Applications*. AINA. IEEE.

Brusilovsky, P. and Maybury, M. T. (2002). From adaptive hypermedia to the adaptive web. *Communications of the ACM*, v. 45, n. 5, p. 30–33.

Brusilovsky, P. and Millán, E. (2007). User Models for Adaptive Hypermedia and Adaptive Educational Systems. In: *The Adaptive Web Methods and Strategies of Web Personalization*. Springer Berlin Heidelberg. p. 3–53.

Chen, H., Perich, F., Finin, T. and Joshi, A. (2004). SOUPA: standard ontology for ubiquitous and pervasive applications. In *International Conference on Mobile and Ubiquitous Systems: Networking and Services, MOBIQUITOUS 2004.* IEEE.

Dentler, K., Cornet, R., Ten Teije, A. and De Keizer, N. (2011). Comparison of reasoners for large ontologies in the OWL 2 EL profile. *Semantic Web*, v. 2, p. 71.

Dey, A. K. (2001). Understanding and Using Context. *Personal and Ubiquitous Computing*, v. 5, n. 1, p. 4–7.

Dourish, P. (2004). What we talk about when we talk about context. *Personal and Ubiquitous Computing*, v. 8, n. 1, p. 19–30.

Golemati, M., Katifori, A., Vassilakis, C., Lepouras, G. and Halatsis, C. (2007). Creating an ontology for the user profile: Method and applications. In *Int. Conf. on Research Challenges in Information Science*.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, v. 5, n. 2, p. 199–220.

Guo, Y., Pan, Z. and Heflin, J. (2005). LUBM: A benchmark for OWL knowledge base systems. *Web Semantics: Science, Services and Agents on the WWW*, v. 3, p. 158.

Hervás, R. and Bravo, J. (2011). Towards the ubiquitous visualization: Adaptive user-interfaces based on the Semantic Web. *Interacting with Computers*, v. 23, p. 40–56.

IEEE Standard for Learning Object Metadata (2002). *IEEE Std 1484.12.1-2002*

Pantsar-Syvaniemi, S., Simula, K. and Ovaska, E. (2010). Context-awareness in smart spaces. In *The IEEE symposium on Computers and Communications*. IEEE.

Pernas, A. M., Diaz, A., Motz, R. and Oliveira, J. P. M. De (2012). Enriching adaptation in e-learning systems through a situation-aware ontology network. *Interactive Technology and Smart Education*, v. 9, n. 2, p. 60–73.

Wang, S.-L. and Wu, C.-Y. (2011). Application of context-aware and personalized recommendation to implement an adaptive ubiquitous learning system. *Expert Systems with Applications*, v. 38, n. 9, p. 10831–10838.

Wang, X. H., Zhang, D. Q., Gu, T. and Pung, H. K. (2004). Ontology based context modeling and reasoning using OWL. In *IEEE Annual Conf. on Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second*. IEEE.

paper:84

Data Extraction from Structured Databases using Keyword-based Queries

Mariana Soller Ramada, João Carlos da Silva, Plínio de Sá Leitão-Júnior

[1]Instituto de Informática – Universidade Federal de Goiás (UFG)
Caixa Postal 131 – 74.001-970 – Goiânia – GO – Brazil

{mariana,jcs,plinio}@inf.ufg.br

Abstract. *Relational databases are used to store a large quantity of data scattered around the world. However, users face difficulties in accessing such data for lack of a more natural way of specifying queries. Techniques that use natural language words to search different information sources on the Web are now very common, but they cannot be employed to search relational databases. This work proposes a method that allows user to submit keyword-based queries, which are then semantically analysed and enriched before being mapped for the database language. By analysing keyword-based queries, the method considers different factors, e.g. the proximity between keywords, query segmentation, and the use of aggregate functions.*

1. Introduction

The use of keywords for retrieving information consists of a simple search technique. The last decade has witnessed the growing use of this technique, which has in fact become a standard for user interaction with the World Wide Web (WWW). However, it cannot be applied to all storage media, like relational databases, for instance, which store a vast amount of valuable information. Querying in relational databases requires prior knowledge of storage structures and of the syntax of a structured language, such as SQL. However, the majority of users do not have such knowledge, which limits access to the stored data. In the last few years, great efforts have been made in research and development activities to extend the abilities of keyword-based search to data sources that follow the relational paradigm. Nevertheless, the existing techniques reveal some drawbacks.

The first drawback is that, according to such approaches, every keyword plays a given role in the database and each keyword is mapped out for a corresponding database structure. Consider the schema of the relation *Employee (Id, Name, Address, Salary, Super_id, Department_id)*. The query *"higher salary"* expects a statistic, the higher salary, as a result, not a set of interconnected tuples containing the keywords *higher* and *salary*. The second drawback is that the query is segmented in a way that each keyword represents a single role in the database. Referring once again to the relation *Employee*, a possible interpretation for the query *"employee Houston Tx"* is the employee who lives at the address Houston Tx. Therefore, the keywords *Houston* and *Tx* are expected to be mapped together for the attribute *Address* of the table *Employee*, instead of being mapped separately for a database structure. The third drawback concerns the fact that several studies fail to consider the interdependence between keywords. Even though a query is made up of a simple list of keywords, the meaning of each keyword is not independent from the

meaning of the others; together, they all represent the concepts intended by the user when creating a query.

This paper focuses on the semantic approach to keyword queries. The drawbacks listed above are considered and solutions are provided and implemented. A new keyword-based search method for relational databases was defined. For a given keyword query, the method proposed converts it into corresponding SQL queries, all of which are submitted to the underlying database. An SQL query is a structured query in which tables, attributes, and their conditions are accurately specified, whereas a keyword query comprises imprecise terms that express the user's need for information. Therefore, this study introduces semantics to this conversion process, to provide a clearer idea of the meaning intended by the query and to construct SQL expressions that represent the user's real intent, returning results in order of relevance. The Keymantic-based approach [Bergamaschi et al. 2010] was chosen as a starting point for meeting the aims of this study, in view of the fact that it deals partially with the third drawback previously mentioned.

The remainder of this paper is organized as follows. Section 2 presents related works. Section 3 shows the architecture of the keyword query method proposed, as well as the way it operates. Section 4 explains how the *Keymantic* system works, and Sections 5 and 6 reveal the modifications proposed and their impacts. Section 7 provides some conclusions.

2. Related Works

The literature reveals two main approaches to keyword queries in relational databases: one based on *Candidate Networks* and another based on *Steiner Trees*. Both conceive a database as a network of interconnected tuples and focus on detecting tuples that contain the keywords of a given query. Query processing returns connected components based on the way these tuples are associated. *DBXplorer* [Agrawal et al. 2002] and *DISCOVER* [Hristidis and Papakonstantinou 2002] implement the *Candidate Networks* approach, whereas *BANKS* [Aditya et al. 2002] applies the *Steiner Tree* approach. All of these systems pose the three drawbacks mentioned in Section 1.

To solve the second drawback, FRISK system [Pu and Yu 2009] uses a dynamic programming algorithm to compute the query's best segmentations and then present them to the user, who in turn chooses the one that better suits his/her intent.

Keymantic [Bergamaschi et al. 2010] and *Keyword++* [Ganti et al. 2010] solve the third drawback by taking into account the query's ambiguity and seeking the completeness and accuracy of results. Completeness consists of returning all relevant results, whereas accuracy refers to the most relevant results which match user intent.

A query may return more than one result, in which case it becomes necessary to order them according to their relevance. Ranking functions assign scores to each result and then classify results according to these scores. Some researchers have employed Data Retrieval metrics for calculating ranking functions, such as Luo et al. [Luo et al. 2008] and Hristidis et al. [Hristidis et al. 2003]. In *DISCOVER* [Hristidis and Papakonstantinou 2002] and *DBXplorer* [Agrawal et al. 2002], results are ranked by simple methods, e.g. based on the number of joins. In *Labrador* [Mesquita et al. 2007] a ranking is computed using a Bayesian network model. As for

BANKS [Aditya et al. 2002], calculating result scores takes into account the edge weight of the data graph.

3. Architecture

The method proposed converts a keyword query into corresponding SQL queries, which are submitted to the underlying database. As a result to a keyword query, the method returns the results obtained by running the SQL queries, listed in order of relevance.

Figure 1 shows an overview of the method's architecture.

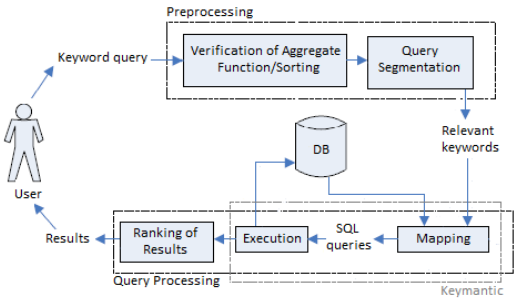


Figure 1. Architecture of the method proposed

3.1. Preprocessing

This stage is responsible for identifying keywords in the query that do not provide a direct meaning in the database structure, but other forms of semantics, e.g. the use of aggregate functions and data sorting. In addition, this stage determine's the query's best segmentation regarding the values comprising more than one keyword.

3.1.1. Verification of Aggregate Function/Sorting

This stage allows the identification of keywords which represent the intention of using aggregate functions and sorting.

Identifying keywords which suggest the use of aggregate functions and sorting is performed through a list of *"reserved"* words, which are basically predefined words based on the way users produce their queries. Seven groups were created and a set of keywords was defined for each of them. Each set consists of synonyms obtained by integrating a thesaurus as a semantic resource. Listed below are the groups and their respective sets:

- maximum = synonym(higher)
- minimum = *synonym(lower)*
- mean = *synonym(average)*
- sum = *synonym(total)*
- count = *synonym(quantity)*
- grouping = *synonym(for each)*
- order = *synonym(sorted)*

After identifying which keyword suggests the use of an aggregate function or sorting, it is necessary to establish which attribute the function will be applied on. Based on the fact that users create queries in which related words are close to one another, once a keyword that suggests the use of an aggregate function or sorting is identified, the function it suggests is applied to the term represented by the keyword that immediately follows it.

3.1.2. Query Segmentation

In general, search engines support the use of delimiters to group multiple words to a single concept. Many concepts are represented by a phrase rather than by a single word. Google and Yahoo! are examples of search engines that allow syntax for phrase searching. Identifying keywords which must be regarded collectively to form an attribute value is performed by the use of single inverted commas.

3.2. Query Processing

Once a keyword query has been submitted, it is necessary to create several SQL queries that will be run in the underlying database. This process is carried out by mapping the keywords for the database terms. Once the SQL queries have been run, results are returned to the user in order of relevance.

3.2.1. Mapping

Returning a keyword query in relational databases requires understanding the meaning of each keyword and the construction of an SQL query that provides a coherent interpretation of the original query. It is necessary to map the keywords for the database structures, e.g. relations, attributes, and attribute values. Several techniques and tools have been proposed to solve the problem of keyword queries in relational databases, as was pointed out in Section 2. However, most of these proposals fail to consider the many interpretations which a keyword query may pose.

The mapping process proposed in this paper is based on the semantic analysis implemented by *Keymantic*, which explores the relative positions (order) of keywords within the query, as well as the database schema and other auxiliary external sources. The *Keymantic* approach, which grounds the present work, is described in Section 4.

3.2.2. Execution and Ranking of Results

Each SQL query generated in the previous stage is now run in the corresponding database, and the results of these queries represent the results expected for a keyword query.

The results produced are not equally significant, as some of them represent the semantics intended by the query more effectively. In this regard, it is interesting to generate a ranking of results. Considering that the *Keymantic* system computes a score for each generated SQL query, the results returned to the user may then be listed firstly by the score of their corresponding SQL query, and secondly by the size of the join path, based on the number of joins required to create the SQL query.

4. Semantic Query Analysis

Keymantic explores the relative positions of query keywords together with external sources, to produce a more accurate assumption of the semantics represented by the query. This statement is grounded on the fact that the meaning of each keyword is not independent from the meaning of the others; all of them collectively represent the concepts the user had in mind when creating the query. Moreover, not all keywords represent instance values. Many are used as metadata of adjacent keywords.

The mapping process performed by *Keymantic* comprises five stages. A special data structure, known as weight matrix, is used during this process. The value of a cell represents the weight related to the mapping performed between keyword and database term. Two sub-matrices may be distinguished in the weight matrix. The first, called SW, corresponds to the database terms related to schema elements, i.e. relations and attributes. The second, called VW, corresponds to attribute values, i.e. elements which belong to attribute domains.

The first step is *Intrinsic Weight Computation*. The relevance between each query keyword and each database term is calculated by exploring and combining a number of similarity techniques. In the next step, *Selection of Best Mappings to Schema Terms*, a serie of mappings is generated based on the intrinsic weights of sub-matrix SW. Each mapping associates a certain number of keywords to the database schema terms. Keywords that remain unmapped are considered at a later stage during value term mapping. Only mappings that reach the highest score are selected. In the third step, *Contextualization of VW and Selection of Best Mappings to Value Terms*, the unmapped keywords are now mapped based on each partial mapping generated in the previous stage.

Then a total mapping of keywords for the database terms form a configuration at step *Generation of the Configurations*. The configuration's score is the sum of weights in the weight matrix of elements $[i, j]$, where i is a keyword and j is the database term to which the keyword was mapped. Finally, once the best configurations have been computed, the interpretations of the keyword query, i.e. SQL queries, may be generated at step *Generation of the Interpretations*. The score of each SQL query is that of its respective configuration. A configuration is simply a mapping of the keywords to database terms. The presence of different join paths between these terms leads to multiple interpretations.

5. Contributions to Semantic Analysis

Keymantic is grounded on the assumption that every keyword plays a role in a query, i.e. each keyword represents a term within the database. As each keyword must represent a database term, it is not possible to interpret queries which suggest the use of aggregate functions or sorting. Even if a query has, for instance, the keyword *higher*, the system will map it to a database term instead of interpreting it as an indicator of the use of the aggregate function *max*.

Additionally, *Keymantic* regards the mapping process as an injective function, in which there is no image element that shows correspondence with more than one domain element, the domain being the set of keywords and the image, the set of database terms. In other words, two keywords cannot be mapped to the same database term. However, as mentioned in Section 1, it is possible for a value to be composed by more than one

keyword. In this sense, it is necessary to map all the keywords which compose this value for the same attribute domain.

Even though *Keymantic* computes weights which refer to each interpretation, it does not provide a classification of the obtained results. The paper which describes this tool suggests that a classification based on each interpretation's score and on the size of the join path may be carried out.

To make feasible the mapping of query keywords for aggregate function, sorting or composite values, as well as to allow sorting results, the method proposed in this paper implements the functionalities which refer to the modules *Verification of Aggregate Function/Sorting*, *Query Segmentation*, and *Ranking of Results* shown in Figure 1, all of which are external to *Keymantic*. In addition, some modifications within *Keymantic*, particularly during *Mapping*, were also performed to yield greater quality to the returned results. Such modifications regard the stages *Intrinsic Weight Computation of Schema Database Terms*, *Intrinsic Weight Computation of Value Database Terms*, and the contextualization process.

Use of Synonyms as Keywords

During *Intrinsic Weight Computation of Schema Database Terms*, *Keymantic* employs a series of techniques that measure similarity between keywords and database terms, selecting the one which produces the best result. One of these techniques is string similarity. For string similarity *Keymantic* employs a number of different similarity metrics such as Jaccard, Hamming, Levenshtein, etc. The tool also assesses the relationship between a given keyword and a database schema term based on their semantic relationship, and for that uses ontologies and dictionaries. For each measuring technique (string similarity, semantic relationship, etc.), the similarity between a keyword and a database term is calculated in a 0-1 interval. The highest value returned is then multiplied by 100 and selected as intrinsic weight. If none of the values returned is higher than a predefined threshold, then the weight is set to 0.

The method here proposed regards, at first, only the string similarity technique. Three similarity metrics are used for this technique: Jaccard, Levenshtein, and Cosine. Each metric has two strings as input and returns the similarity between them in a 0-1 interval. Computing the intrinsic weight of a keyword and a database term is based on the average of similarities returned for each metric. If the average is greater than the threshold, it is multiplied by 100 and selected as intrinsic weight. If not, the similarity between the synonyms of the keyword and the database term will be calculated. That involves using the *WordNet* [1] dictionary to return the synonyms of each query keyword. The average of similarities returned for each metric is computed for each synonym found. The greater average value returned will be taken into account. If this value is greater than the threshold, the average is multiplied by 90 and selected as intrinsic weight. If it is lower than the threshold, then the intrinsic weight is set to 0.

Because users normally expect keywords from the query to appear in the results, the reason for considering the use of a thesaurus at a later stage is that it is necessary to give greater relevance to keyword queries. In other words, the existence of the keyword in the database must have greater weight than the existence of its synonym.

[1] http://wordnet.princeton.edu/

Weight normalization for sub-matrix VW

The modification applied to the step *Intrinsic Weight Computation of Value Database Terms* is performed because computing value term weights is regarded in binary form. Therefore, whereas the weights of sub-matrix SW - which refer to schema terms - are found in a 0-100 interval, the weights of sub-matrix VW are 0 or 1. Thus, given that the weights of matrix VW are binary, the value attributed to the total configuration score by the keywords mapped to value terms is minimum. That being said, it is necessary to compare the value of value term intrinsic weights with that of schema term weights. Hence, after calculating the intrinsic weights for the value terms in the method proposed, weights that have value 1 are multiplied by a predefined constant.

Proximity between keywords

The last modification performed within *Keymantic* took place during contextualization. *Keymantic* takes into account the interdependence between query keywords, in which mapping a given keyword may increase or reduce the probability that another keyword, as yet unmapped, corresponds to a given database term. In addition to considering the interdependences between keywords, the method proposed takes into account their relative positions in the query based on their proximity. The farther a mapped keyword is from an unmapped one, the smaller its influence over this word's mapping process. This fact led to the following modification during contextualization: instead of adding a constant to intrinsic weights, the value added to the weight is proportional to the distance between keywords.

6. Comparison with Keymantic

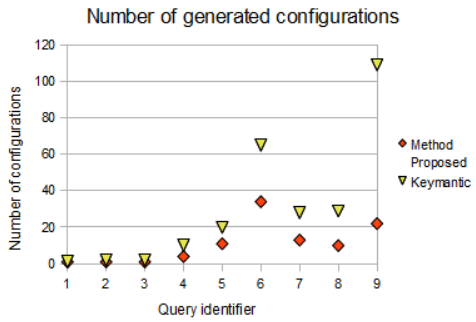
This section describes the impacts caused to the set of results presented to the user after applying modifications to the method proposed in *Keymantic*. Internal modifications were performed to attribute further semantics to the mapping process, in order to generate a smaller but more relevant set of interpretations in view of user intent. Precision and MRR metrics were used for this analysis.

Using a company database, both systems were implemented and compared based on the results returned by each after running a set of keyword queries. Table 1 shows the set of keyword queries selected from a query pool and its user intended semantics.

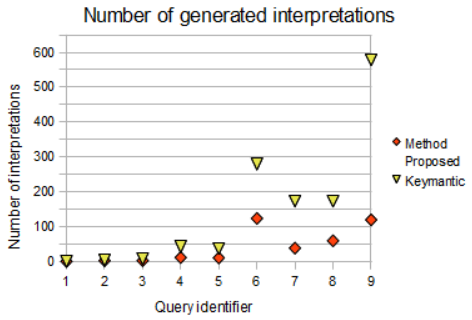
Table 1. Set of keyword queries

	Keyword Queries	Intended Semantics
1	*department*	*List of information regarding the company's departments*
2	*department employee*	*List of each department's employees*
3	*department project*	*List of each department's projects*
4	*department project newbenefits*	*Information of the department responsible for the Newbenefits project*
5	*employee dependent daughter*	*Information regarding employees who have a daughter as a dependent*
6	*project name employee John*	*Name of the project for which employee John works*
7	*employee address project 1*	*Address of employee who works in project 1*
8	*hours employee works project productz*	*Number of working hours of each employee in project ProductZ*
9	*project location employee salary 30000*	*Location of project for which the employee who earns a 30000 salary works*

Figure 2 compares and constrasts the number of configurations and interpretations (SQL queries) generated by both systems. Values are shown for each of the queries, whose identification is the same as that used in Table 1. Internal modifications, performed during the first three stages of *Keymantic's* mapping process, influenced the results obtained by our method, leading to lower values compare with *Keymantic* as regards the number of configurations and interpretations. This reduction has a major impact on the cost of the process and allows the user to deal with less results for his/her query.



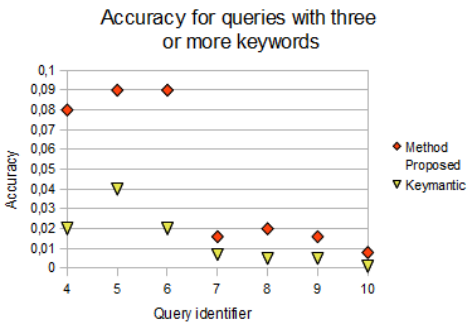
(a) Number of configurations generated by both systems



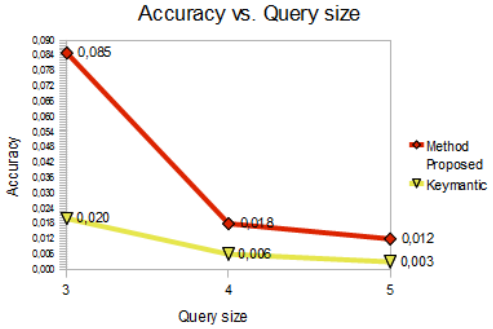
(b) Number of interpretations generated by both systems

Figure 2. Number of configurations and interpretations generated by both systems

The graphs in Figure 3 explore queries with three or more keywords, as well as present each system's relative accuracy. This metric reveals the relevant fraction of the obtained interpretations, hence its status as a quality indicator commonly described in the literature. As shows Subfigure 3(a), the method proposed was more effective, showing that despite the reduction in the number of interpretations generated, the resulting set is more relevant to the query. Subfigure 3(b) exhibits accuracy values which refer to query size, i.e. the number of keywords comprising the query. Accuracy decreases in both systems as the number of query keywords increases. However, results reveal, once more, better accuracy values when compared with the original method.



(a)



(b)

Figure 3. Accuracy metric for both systems

As regards the external stage *Ranking of Results*, *Keymantic* is known to show results according to the order in which they are generated, with no guarantee that they will be presented in a descending score order.

The metric Mean Reciprocal Rank (MRR) was used to assess the stage *Ranking of Results*. This metric measures how near the top the first relevant result ranks within the set of results. Figure 4 shows the MRR for both systems based on query length. In *Keymantic*, the longer the query, the farther from the top of the ranking appears the first relevant result. In the method proposed, even though the first result becomes distant from the top as the query becomes longer, it remains one of the top-ranked results.

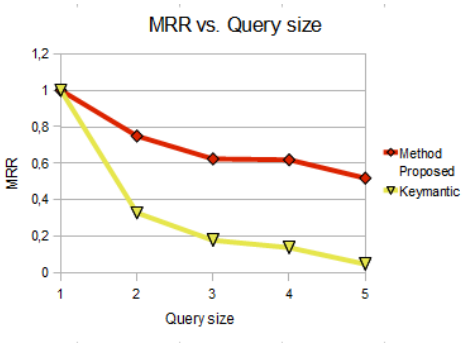


Figure 4. MRR for both systems (inspired in [Fakhraee and Fotouhi 2012])

Threats to Validity

Three limitations were identified with regard to the results obtained with the proposed method.

The first limitation is related to the cardinality of the set of queries from which the results were obtained. A more extensive set of queries could lead to a higher confidence level when comparing results with Keymantic. The second limitation is related to how the queries were obtained, which are designed for the analysis of the proposed method instead of queries obtained from real systems. The last limitation is related to the fact that the proposed method was compared solely with *Keymantic*. A stronger argument could be made in favor of the proposed approach if its performance were to be compared with other existing state-of-the-art tools that allow searching relational databases using keyword queries.

7. Conclusion

This paper proposed a method for querying relational databases with keywords to simplify access to these data, given the fact that such queries use natural language words. This method considered factors such as query segmentation, aggregate functions and sorting, as well as the user's intended semantics when creating the query. The semantic analysis of keyword queries was based on the approach provided by the *Keymantic* tool.

Other resources were added to *Keymantic's* original proposal, such as the possibility of dealing with queries using aggregate functions and values comprising more than one keyword. To offer new querying possibilities, the method proposed implemented new external functionalities and performed internal improvements to the tool.

During the experiments, it became clear that the internal modifications promoted a smaller and more significant set of results for the keyword query submitted, whereas the external modifications allowed the specification of queries that had up until then not

been considered, such as those that employed aggregate functions, sorting, and values comprising more than one keyword.

Along the course of this research, we identified some aspects that could be complemented on to further discussion on this topic. Among them is the use of ontologies. During the process of intrinsic weight computation of schema terms, *Keymantic* uses ontologies, in addition to the synonyms obtained by WordNet, hence adding greater semantics to these stages. Another aspect to be considered is more effective query segmentation. In the method proposed, composite values are identified via the use of single inverted commas, which requires the user's a priori knowledge to construct the query in a suitable way.

References

Aditya, B., Bhalotia, G., Chakrabarti, S., Hulgeri, A., Nakhe, C., Parag, and Sudarshanxe, S. (2002). Banks: Browsing and keyword searching in relational databases. In *VLDB '02: Proceedings of the 28th Intl. Conference on Very Large Databases*, pages 1083 – 1086. Morgan Kaufmann, San Francisco.

Agrawal, S., Chaudhuri, S., and Das, G. (2002). Dbxplorer: a system for keyword-based search over relational databases. In *Data Engineering, 2002. Proceedings. 18th Intl. Conference on*, pages 5 –16.

Bergamaschi, S., Domnori, E., Guerra, F., Orsini, M., Lado, R. T., and Velegrakis, Y. (2010). Keymantic: semantic keyword-based searching in data integration systems. *Proc. VLDB Endow.*, 3:1637–1640.

Fakhraee, S. and Fotouhi, F. (2012). Dbsemsxplorer: semantic-based keyword search system over relational databases for knowledge discovery. In *Proceedings of the Third Intl. Workshop on Keyword Search on Structured Data*, KEYS '12, pages 54–62, New York, NY, USA. ACM.

Ganti, V., He, Y., and Xin, D. (2010). Keyword++: a framework to improve keyword search over entity databases. *Proc. VLDB Endow.*, 3:711–722.

Hristidis, V., Gravano, L., and Papakonstantinou, Y. (2003). Efficient ir-style keyword search over relational databases. In *Proceedings of the 29th Intl. conference on Very large data bases - Volume 29*, VLDB '2003, pages 850–861.

Hristidis, V. and Papakonstantinou, Y. (2002). Discover: Keyword search in relational databases. In Bernstein, P. A., Ioannidis, Y. E., Ramakrishnan, R., and Papadias, D., editors, *VLDB '02: Proceedings of the 28th Intl. Conference on Very Large Databases*, pages 670 – 681. Morgan Kaufmann, San Francisco.

Luo, Y., Wang, W., and Lin, X. (2008). Spark: A keyword search engine on relational databases. In *Data Engineering, 2008. ICDE 2008. IEEE 24th Intl. Conference on*, pages 1552 –1555.

Mesquita, F., da Silva, A. S., de Moura, E. S., Calado, P., and Laender, A. H. F. (2007). Labrador: Efficiently publishing relational databases on the web by using keyword-based query interfaces. *Inf. Process. Manage.*, 43(4):983–1004.

Pu, K. and Yu, X. (2009). Frisk: Keyword query cleaning and processing in action. In *Data Engineering, 2009. ICDE '09. IEEE 25th Intl. Conference on*, pages 1531 –1534.

paper:7

Debugging Scientific Workflows with Provenance: Achievements and Lessons Learned

Daniel de Oliveira[1], Flavio Costa[2], Vítor Silva[2], Kary Ocaña[2] and Marta Mattoso[2]

[1]Institute of Computing – Fluminense Federal University (IC/UFF)

[2]COPPE – Federal University of Rio de Janeiro (UFRJ)

danielcmo@ic.uff.br, {flscosta, silva, kary, marta}@cos.ufrj.br

Abstract. *Scientific Workflow Management Systems manage experiments in large-scale and deliver provenance data. Provenance data represents the workflow execution behavior, allowing for tracing the data-flow generation. When provenance is extended with performance execution data, it becomes an important asset to identify and analyze errors that occurred during the workflow execution (i.e. debugging). Debugging is essential for workflows that execute in parallel in large-scale distributed environments since the incidence of errors in this type of execution is high and difficult to track. By debugging at runtime, scientists can identify errors and take the necessary actions, while the workflow is still running. We present provenance based debugging, in real use cases, running in parallel, with virtual machines in clouds. In these experiences scientists use provenance data to query domain and execution data to detect errors, especially when the execution concludes without any system error message.*

1. Introduction

Many scientific experiments are modeled as scientific workflows to represent the flow of data, transformed by programs, each of which consumes a set of parameters and input data. Scientific workflows are managed by powerful engines in Scientific Workflow Management Systems (SWfMS), such as Pegasus (Lee *et al*. 2008), Swift/Turbine (Wozniak *et al*. 2012), Tavaxy (Abouelhoda *et al*. 2012) and SciCumulus (Oliveira *et al*. 2010), which have parallel execution support. SWfMS are used in domains, such as, engineering, bioinformatics and astronomy (Hey *et al*. 2009). Furthermore, several scientific workflows execute in large scale, since they process large amounts of data (Pennisi 2011) requiring parallel execution in High Performance Computing (HPC) environments, such as clusters, grids, and more recently, clouds. These SWfMS also provide provenance capabilities to record the historical information about the workflow execution behavior (Freire *et al*. 2008) to help workflow analysis, reproducibility and validation. Although several SWfMS provide sophisticated mechanisms for executing large-scale scientific workflows in distributed HPC environments, most of them execute the workflow in an "offline" way (Ailamaki *et al*. 2010), i.e. as a black-box. Online monitoring and debugging may save significant amounts of workflow execution time, when unexpected behavior can be detected way before the end of workflow execution. When debugging can analyze the relation between execution data, workflow data and

[1] This work is partially sponsored by CNPq and FAPERJ.

domain data, scientists have a much clear picture of the experiment dataflow (Mattoso *et al*. 2013). When this analysis is done at runtime, significant time may be saved.

We consider here three types of unexpected execution behavior, one is related to execution performance, the second is aware of the workflow stage of execution and the third is related to data-flow generation, including domain data analysis (Gonçalves *et al*. 2012). Execution performance may indicate that a task is taking longer than expected. It also identifies failures from node, task or program crashes, among others. Workflow execution may be unexpected when a data is generated and does not produce the expected result value. In fact, this may prevent the next activity to consume it, generating an anomaly. For example, the specification of a filter value may not be adequate, generating an excessive or too restrictive data-flow. Since these results would not cause any failure on the execution, scientists would only realize the filter configuration problem by the end of the whole workflow execution. By further analyzing domain data, scientists may evaluate if domain data is not compatible with the defined filter value. Therefore, data-flow monitoring also helps to improve workflow performance and should be together with domain data and execution monitoring.

To help workflow execution monitoring and debugging, related work like Stampede (Gunter *et al*. 2011) and Panda (Ikeda *et al*. 2012) support data-oriented scientific workflow analysis using runtime execution data. These systems are focused on workflow activities and their corresponding jobs and tasks executions, using performance analyzers, such as, NetLogger to associate execution behavior to activities. They represent performance metrics and file generations in a relational database so that users can query and identify execution failures in scientific workflows, at runtime, even when the workflow execution does not crash. Stampede has a visual interface where execution behavior may be visualized and analyzed with R package (Vahi *et al*. 2013).

Nevertheless, these powerful execution performance monitoring and debugging do not allow for domain data-flow generation debugging. This would require access to provenance data-flow and some related domain data, also at runtime. When provenance data and execution monitoring data are in separate databases, it is difficult to relate execution behavior with the data that caused an unexpected activity execution. For example, if the user detects that a specific task failed or is taking too long, it might be helpful to check the inputs and parameters of this task, or in which workflow iteration this failure occurred. To track this, the user would need first to query the execution monitoring database to find the instance of the failed task, and then query the domain data-flow, to manually try to relate them all together, *i.e*., the execution instance task with its corresponding input data-flow.

In another related work, MTCProv (Gadelha *et al*. 2012) stores performance execution time in the same database as provenance data and it helps scientists in data-flow debugging through provenance queries. However, in MTCProv, online monitoring is not supported, therefore debugging queries can only be submitted after the workflow finishes its execution. Unexpected data-flow might not generate a failure occurrence, however it would still represent an anomalous execution requiring aborting the execution as soon as possible. Waiting until the end to debug, considering performance and data-flow, change the workflow specification and resubmitting it takes too long.

Differently from the current mainstream, Chiron (Ogasawara *et al*. 2011) and its extension to cloud environments, SciCumulus, are the only SWfMS that support

debugging based on execution performance and domain data-flow generation. They obtain that by integrating all these data at their provenance database, available for traversal analysis at runtime. In this paper we go deep into specific details of the data-flow algebraic approach (Ogasawara *et al*. 2011), adopted in both Chiron and SciCumulus, to show the potential of data analysis in real applications. Therefore, this paper presents our achievements and lessons learned from users exploring the runtime provenance support of SciCumulus in debugging real large-scale bioinformatics workflows (Ocaña *et al*. 2011, Ocaña *et al*. 2014).

This paper is organized as follows: Section 2 explains the importance of provenance in scientific workflow monitoring and debugging; Section 3 describes case of studies for workflow debugging at runtime using SciCumulus; and finally, Section 4 presents our conclusions and future research directions.

2. Data-flow Runtime Analysis

The workflow data centric algebraic approach (Ogasawara *et al*. 2011), that drives the execution engine of both Chiron and SciCumulus, uses relations to store the workflow specification as a prospective provenance and the workflow execution as a retrospective provenance, all in the same database. In other words, prospective provenance presents a workflow composition (such as activities and your dependencies), while retrospective provenance determines properties about workflow execution (such as the directory to execute workflow or database hostname) (Freire *et al*. 2008). This algebraic approach introduces a set of operators to the relational algebra (*Map*, *Reduce*, *Filter*, *SplitMap*, *MRQuery* and *SRQuery*), which considers both workflow activities and data-flow as operands. Each activity of the workflow is associated to an operator, which has operand relations as input and all results of this activity execution are stored at an output relation, which can be the input operand relation to the operator associated to the next workflow activity.

To present the potential of monitoring and debugging, we show the algebraic view of two real workflows from the bioinformatics domain using SciCumulus. One is SciPhy, for phylogeny analyses (Ocaña *et al*. 2011) and the other is SciDock for molecular docking analyses (Ocaña *et al*. 2014). SciPhy and SciDock are complex workflows composed by several data and computing-intensive activities that may last for days, weeks or even months, to generate results. We simplify them here and show the algebraic relations only for the most representative activities (programs) for both workflows: RAxML for SciPhy and autodock for SciDock.

SciPhy, partially represented in Figure 1, is modeled as two algebraic expressions: $R_m \leftarrow$ *Map*(RAxML, R_i); $R_o \leftarrow$ *Reduce*(TreeView, "model", R_m). RAxML is followed by the phylogenetic analysis activity TreeView. Workflow parameters from the *Command line* are now attributes of input relation R_i and their values are inserted in R_i tuples (Figure 1.a). RAxML is executed n times with different input data for three parameters (*i.e*. the last attributes in R_i) generating n resulting tuples in the output relation R_m. Each tuple of R_m is related to the consumption of each R_i tuple (Figure 1.b). Attribute k relates R_i with R_m. TreeView executes a computational simulation by grouping R_m tuples by *model* (*Reduce*) into output relation R_o to obtain a phylogenetic tree and for analyzing/visualizing the quality of the tree (Figure 1.c).

Scientists may query R_m at runtime, while the output tuples are still being generated. For example, a query may retrieve tuples with attributes *Result = true* and *model = GTR*, which were generated from a runtime calculated average bootstrap attribute value in R_i that are above a threshold. This would indicate an issue in the obtained model that will influence the topology of the trees. In another example, based on data-flow analysis, scientists decide that they are only interested on result models that generate trees where bootstrap *quality-bp (AVG) > 90*, and they may submit relational queries to R_o to filter out unwanted results for the next steps. Estimating adequate thresholds or filters before evaluating the data-flow is complex and anticipating the "right" value might not be possible.

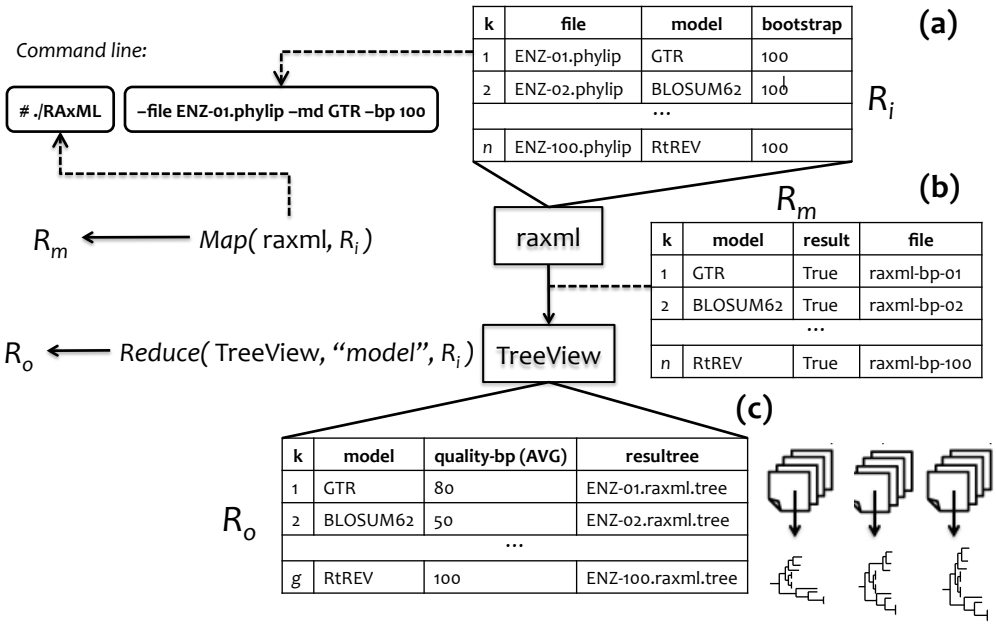


Figure 1. Reduced algebraic workflow representation of SciPhy

SciDock is a workflow for molecular docking that attempts to mimic the process of bringing together a protein and a ligand to form a non-covalent complex and to reveal the electrostatic and steric complementarity between the protein and the ligand. Figure 2 shows the most representative activities of SciDock (autodock4) and the following docking analysis (Vina). Vina executes a computational simulation for analyzing/ visualizing the quality of the molecular docking for the receptor-ligand pairs (True, if it exists/False, otherwise), reinforcing the biological inferences attributed to each obtained molecular docking and statistical results. The algebraic workflow would be: $R_x \leftarrow Map$ (autodock4, R_e); $R_s \leftarrow Map$ (Vina, R_x) ; represented by Figure 2.a and Figure 2.b.

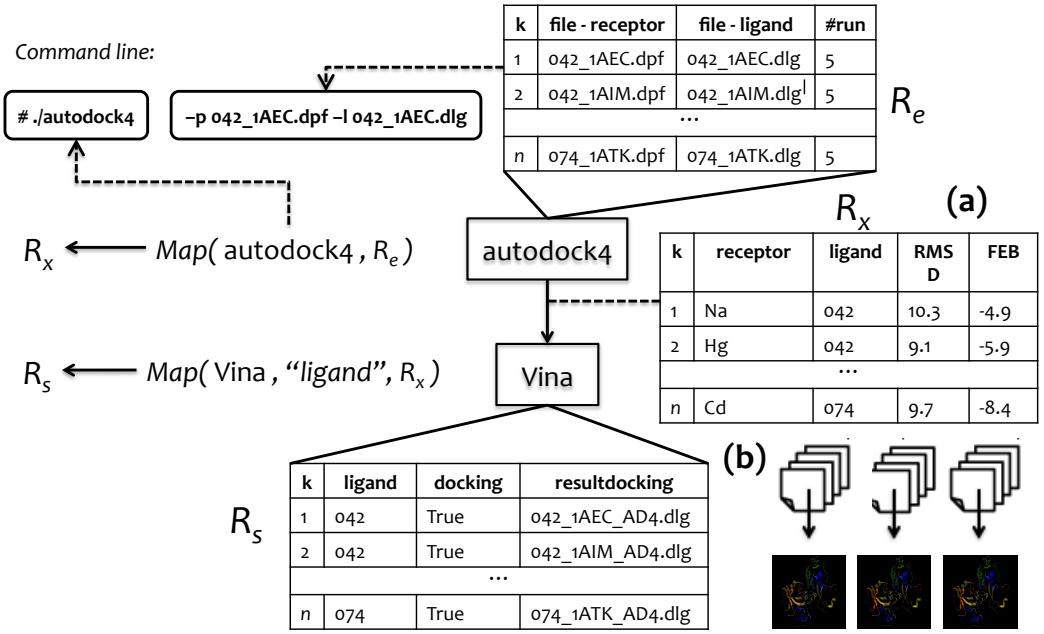


Figure 2. Reduced algebraic workflow representation of SciDock

Scientists may query R_x data-flow at runtime and check if it presents negative average FEB values and RMSD > 4 for any tuple. This would indicate an issue with the obtained ligand/receptor biophysical characteristics that will influence the existence (or not) of the docking process, depending on some the quality values (*i.e.*, FEB, RMSD).

3. Debugging Bioinformatics Workflows Using Execution and Domain Data

During SciPhy and SciDock executions, SciCumulus provenance database was queried to detect possible unexpected situations from workflow executions. We discuss here queries related to performance execution time and data-flow generation for detecting domain-specific problems that would not be identified using monitoring and debugging mechanisms based only on performance execution data.

Domain-data relations with parameters consumed and produced by activities (*i.e.* R_i and R_m) are stored in the same provenance database that has execution data and typical workflow data. The workflow algebraic approach fostered this provenance data integration. Our provenance relations follow the PROV-Wf model (Costa *et al.* 2013), an extension of the W3C PROV standard (Groth and Moreau 2013). Using PROV-Wf, scientists are able to query performance execution time, domain and data-flow from workflow execution, as soon as they are generated, at runtime. PROV-Wf is composed by a set of classes (each one has an associated relation at the provenance database) to represent workflow concepts. In Figure 3 we show some of them, used in our queries: *hworkflow*, *hactivity* for prospective provenance and *hactivation*, *relation*, *hfile*, and *hkeyspace* for retrospective provenance. The instance of one *hactivity* of an *hworkflow* execution is represented as a tuple in *hactivation* through attribute *actid*. Table *relation* has its name and attributes defined by the user to store domain data values for input parameters that are consumed by an activity execution, like R_i is input to RAxML, and *hfile* provides access from activities to domain files, like ENZ-01.phylip in R_i and raxml-bp-01 in R_m.

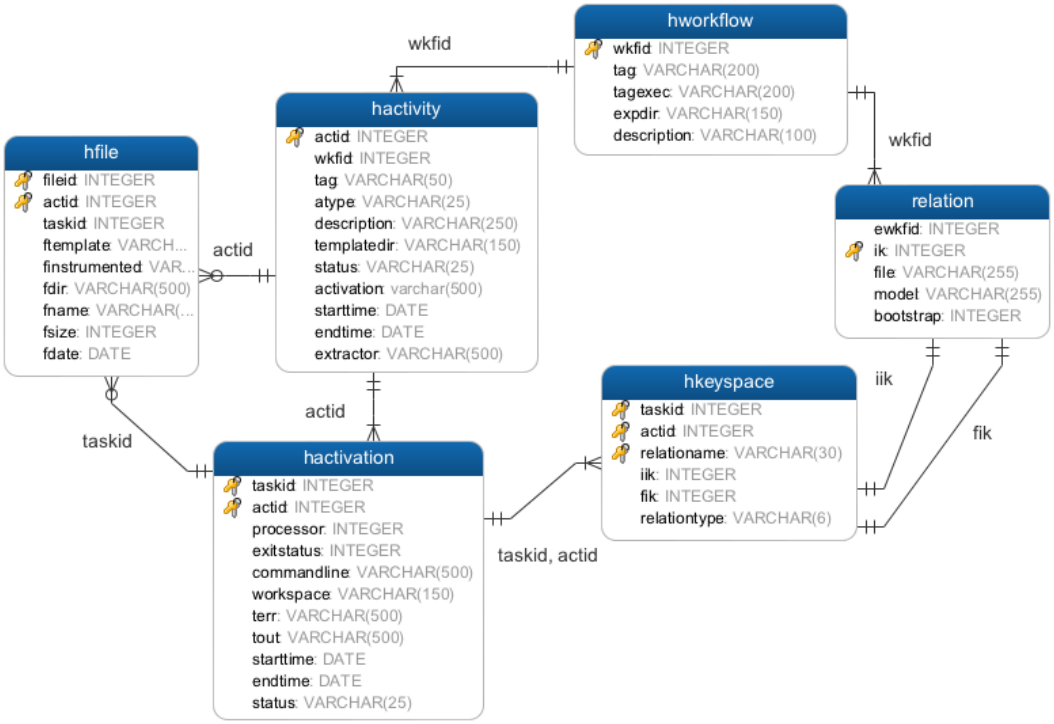


Figure 3. Simplified view of PROV-Wf classes, adapted from (Costa *et al.* 2013)

All experiments presented in this section were performed in the Amazon EC2 cloud environment using 32 Amazon's large VMs (EC2 ID: m1.large – 7.5 GB RAM, 850 GB storage, 2 cores). Provenance data is managed using PostgreSQL database system, stored at Amazon S3. The execution of SciPhy generated 1,000 activations for 200 input files. SciDock generated 1,798 activations for 200 input multi-fasta files. This means that for each execution of SciPhy 1 row is inserted in *hworkflow* table, 5 in *hactivity*, 1,000 in *hactivation*, 1,000 in *relation*, 1,000 in *hkeyspace* and 5,315 in *hfile*. In the case of SciDock, for each execution consuming 200 input data, 1 row is inserted in *hworkflow*, 9 in *hactivity*, 1,798 in *hactivity*, 1,798 in *hrelation*, 1,798 in *hkeyspace* and 10,380 rows in hfile. In total, the provenance database contains the following amount of rows *per* table: *hworkflow:* 757, *hactivity:* 3265, *hactivation:* 67,039, *hfile:* 272,218, *relation:* 67,035, and *hkeyspace:* 67,035. The following selected queries were considered in our analysis to represent three types of the most frequent queries: execution, data-flow and monitoring queries. In this case, our monitoring query aims to identify activity executions with errors.

Query 1 (Q1) – Execution time. SciPhy workflow is executed 1,000 times, varying a large number of parameters or data input. Errors can be identified at runtime, when scientists are monitoring and tracing one specific task. Using task execution time information, from historic provenance, users are able to identify performance variations that may indicate execution outliers. If one task is consuming more time to execute than expected (*e.g.* more than average execution time), there is an indication of alert. Scientists have to investigate if this performance variation is an error. Related data files are identified through provenance queries and then analyzed. In SciPhy, queries may check: the number of sequences in the input fasta file (*i.e.*, if there are less than three or more than 100 fasta sequences); if the format (for amino acid sequences) is recognized; or if the default number for bootstrap replication is 100. When scientists are able to

analyze performance metrics with domain-specific data, they can evaluate if the execution is likely to be correct or not. This execution time debugging with domain data saved 15% of time, since waiting until the end of the execution would require additional 3.1 hours (Ocaña *et al*. 2011).

```
Q1: SELECT w.tag, a.tag, t.taskid, t.exitstatus, t.processor, t.workspace, t.status, t.endtime, t.starttime,
    extract ('epoch' from (t.endtime-t.starttime))||',' as duration, r.file, r.bootstrap
    FROM hworkflow w, hactivity a, hactivation t, hkeyspace k, relation r
    WHERE w.wkfid = a.wkfid  AND a.actid = t.actid AND t.taskid = k.taskid AND r.ik >= k.iik
    AND r.ik <= k.fik
    AND w.tag like '%SciPhy%'
    AND r.bootstrap <> 100
    AND duration > (SELECT avg(extract ('epoch' from (t.endtime-t.starttime))||',')-
                        2.4*stddev(extract('epoch' from (t.endtime-t.starttime))||',')
                        FROM hactivation ac WHERE ac.actid=t.actid)
```

Query 2 (Q2) – Data-flow. In SciDock workflow, approximately 30% of executions failed. Problems involving environment and workflow specification were checked and discarded (*i.e*. any problems related to virtual machines or environment configuration). Therefore, we concluded that failures were associated with input data (*i.e*. ligands or receptor structures). Querying the provenance database, the errors were constantly traced back to a specific set of ligand structures (*i.e*. very small ligands), which were not compatible with receptors of the cysteine protease family. Then, these ligands were detected and excluded for *a posteriori* executions. At the end, results were re-verified with Q2 to confirm that the ".dlg" files (containing the final docking results) were correctly generated (*i.e*. as Successful Completion), and consequently, the docking quality scores (*e.g*. FEB and RMSD) analyzed by specialists. This data-flow query saved 30% of time, since waiting the execution end would require another 21.2 hours. However, even after waiting for the workflow completions, if this direct access to the set of ligand structures was not available, it would be very difficult to identify that the anomalous ligands all shared a common property, related to the failures.

```
Q2: SELECT w.tag, a.tag, f.fname, f.fsize, f.fdir, r.RMSD, r.FEB
    FROM hworkflow w, hactivity a, hactivation t, hfile f, hkeyspace k, relation r
    WHERE w.wkfid = a.wkfid AND a.actid = t.actid AND f.taskid = t.taskid AND t.taskid = k.taskid
    AND r.ik >= k.iik AND r.ik <= k.fik
    AND w.tag like '%SciDock%' AND f.fname like '%dlg%'  AND t.exitstatus = 0
```

Query 3 (Q3) Unexpected program errors. By default, programs like AutoDock and AutoDock Vina do not recognize a set of atoms (*e.g., Na, Hg, Ac, Cd, Ce, V, K*) producing the following error message: "*Unknown receptor type: 'Hg'*". The problem is that users cannot specify *a priori* which atoms are inside each ligand/receptor structure. However, in structures of enzyme families, as cysteine protease, used in our experiments, some receptor structures can contain mercury (*Hg*). This scenario was detected by querying the provenance database at runtime. Q3 searches for tasks that present some failures in the execution (*i.e*. that produced error messages as outputs) and whose input data has the atom *Hg* in its cysteine protease structures. When identifying these tasks, the *Hg* molecule was included to be recognized in cysteine protease receptors. Finally, all executions that failed due to this problem were successfully identified and re-executed. This data-flow query saved 3% of time, since waiting until the end of the execution would require 3.9 hours. Again, debugging without provenance would be even more time consuming.

```
Q3: SELECT w.wkfid, w.tag
    FROM hworkflow w, hactivity a, hactivation t, hkeyspace k, relation r
    WHERE w.wkfid = a.wkfid AND a.actid = t.actid AND t.taskid = k.taskid AND r.ik >= k.iik
    AND r.ik <= k.fik
    AND w.tag like '%SciDock%'
    AND a.tag = 'autogrid4'
    AND r.receptor == 'Hg'
    AND (t.terr <> '' OR t.exitstatus <> 0)
```

The unexpected behavior detected by queries 1, 2 and 3 could only be identified by associating performance execution data with provenance data and domain-specific data. Using relational queries, scientists can check on thousands of executions related to hundreds of thousands of domain data files. This represents a step forward to scientists regarding related work. While querying the integrated database at run time, users do not have to manually find the required data files, download them and analyze each file individually (without data flow context) to detect failures and debug unexpected behavior in their workflow execution. Typical queries are also available as simpler parameterized query templates.

4. Conclusions and Final Remarks

Debugging mechanisms in HPC scientific workflows are essential to support the exploratory nature of science and the dynamic process involved in scientific analyses. Large-scale experiments can benefit from debugging features to reduce the incidence of errors, decrease the total execution time and sometimes reduce the financial cost involved. In most of the executions, typical HPC profiling tools are too low level, not aware of workflow resources. The debugging process has to explore the content of input data, iteratively searching for the identification of what caused the anomalous execution. Since each workflow execution may consist of hundreds or even thousands of activities that are executed in parallel, it is unviable to perform a manual monitoring and debugging on data-flow generation.

This paper discussed the importance of W3C provenance data enriched with performance and domain-specific data. Furthermore, real bioinformatics workflow executions present how to make fine-tuning debugging using the provenance data at runtime. The actions taken from debugging provenance queries improved the quality of results and time spent to complete all computations in bioinformatics scenarios. Similar results were obtained in different domains such as deep water oil exploitation (Ogasawara et al. 2011), numerical reduced order models (Dias et al. 2011) and text mining (Dias et al. 2013).

Data storage and access from thousands workflow executions are also important issues. When workflow execution and provenance data are in the same database, queries are simpler to formulate and a lot of data redundancy is avoided. However, access to this database may become a bottleneck. Using well-known parallel relational database techniques and new lighter SQL systems may leverage this. This integrated provenance database only has metadata for one specific execution. When the workflow finishes its execution, this database may be loaded into a larger provenance warehouse. We believe that database technology has an unexplored potential to further improve existing solutions for workflow debugging in parallel large data-flow executions.

References

Abouelhoda, M., Issa, S., Ghanem, M., (2012), "Tavaxy: Integrating Taverna and Galaxy workflows with cloud computing support", *BMC Bioinformatics*, v. 13, p. 77.

Ailamaki, A., Kantere, V., Dash, D., (2010), "Managing scientific data", *Communications of the ACM*, v. 53, n. 6 (Jun.), p. 68–78.

Costa, F., Silva, V., de Oliveira, D., Ocaña, K., Ogasawara, E., Dias, J., Mattoso, M., (2013), "Capturing and Querying Workflow Runtime Provenance with PROV: A Practical Approach". In: *EDBT/ICDT '13 Workshops*, p. 282–289, New York, NY, USA.

Dias, J., Ogasawara, E., Oliveira, D., Porto, F., Coutinho, A., Mattoso, M., (2011), "Supporting Dynamic Parameter Sweep in Adaptive and User-Steered Workflow". In: 6th WORKS, p. 31–36, Seattle, WA, USA.

Dias, J., Ogasawara, E., de Oliveira, D., Porto, F., Valduriez, P., Mattoso, M., (2013), "Algebraic dataflows for big data analysis". In: *2013 IEEE International Conference on Big Data2013 IEEE International Conference on Big Data*, p. 150–155

Freire, J., Koop, D., Santos, E., Silva, C. T., (2008), "Provenance for Computational Tasks: A Survey", *Computing in Science and Engineering, v.10*, n. 3, p. 11–21.

Gadelha, L. M. R., Wilde, M., Mattoso, M., Foster, I., (2012), "MTCProv: a practical provenance query framework for many-task scientific computing", *Distributed and Parallel Databases*, v. 30, n. 5-6 (Oct.), p. 351–370.

Gonçalves, J. C. A. R., Oliveira, D., Ocaña, K. A. C. S., Ogasawara, E., Mattoso, M. (2012) "Using Domain-Specific Data to Enhance Scientific Workflow Steering Queries", In: IPAW 2012, p. 152-167

Gunter, D., Deelman, E., Samak, T., Brooks, C. H., Goode, M., Juve, G., Mehta, G., Moraes, P., Silva, F., et al., (2011), "Online workflow management and performance analysis with Stampede". In: *Network and Service Management (CNSM), 2011 7th International Conference on*, p. 1 –10

Hey, T., Tansley, S., Tolle, K., (2009), *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research.

Ikeda, R., Cho, J., Fang, C., Salihoglu, S., Torikai, S., Widom, J., (2012), "Provenance-Based Debugging and Drill-Down in Data-Oriented Workflows". IEEE 28th International Conference on Data Engineering (ICDE), p. 1249–1252

Lee, K., Paton, N. W., Sakellariou, R., Deelman, E., Fernandes, A. A. A., Mehta, G., (2008), "Adaptive Workflow Processing and Execution in Pegasus". In: *3rd International Conference on Grid and Pervasive Computing*, p. 99–106, Kunming, China.

Mattoso, M., Ocaña, K., Horta, F., Dias, J., Ogasawara, E., Silva, V., de Oliveira, D., Costa, F., Araújo, I., (2013), "User-steering of HPC workflows: state-of-the-art and future directions". In: *2nd SWEET ACM SIGMOD Workshop*, p. 1–6, New York, NY, USA.

Ocaña, K. A. C. S., Oliveira, D., Ogasawara, E., Dávila, A. M. R., Lima, A. A. B., Mattoso, M., (2011), "SciPhy: A Cloud-Based Workflow for Phylogenetic Analysis of Drug Targets in Protozoan Genomes". In: *BSB*, p. 66–70, Berlin, Heidelberg.

Ocaña, K., Benza, S., Oliveira, D., Dias, J., Mattoso, M., (2014), "Exploring Large Scale Receptor-Ligand Pairs in Molecular Docking Workflows in HPC Clouds". In: *13th IEEE International Workshop on High Performance Computational Biology (HiComb 2014)*, IPDPS, Phoenix, Arizona, USA, p. 536-545.

Ogasawara, E., Dias, J., Oliveira, D., Porto, F., Valduriez, P., Mattoso, M., (2011), "An Algebraic Approach for Data-Centric Scientific Workflows", *PVLDB Endowment*, v. 4, n. 12, p. 1328–1339.

Oliveira, D., Ocaña, K. A. C. S., Ogasawara, E., Dias, J., Gonçalves, J., Baião, F., Mattoso, M., (2013), "Performance evaluation of parallel strategies in public clouds: A study with phylogenomic workflows", *Future Generation Computer Systems*, v. 29, n. 7 (Sep.), p. 1816–1825.

Oliveira, D., Ogasawara, E., Baião, F., Mattoso, M., (2010), "SciCumulus: A Lightweight Cloud Middleware to Explore Many Task Computing Paradigm in Scientific Workflows". In: *3rd International Conference on Cloud Computing*, p. 378–385, Washington, DC, USA.

Pennisi, E., (2011), "Will Computers Crash Genomics?", *Science*, v. 331, n. 6018 (Feb.), p. 666–668.

Vahi, K., Harvey, I., Samak, T., Gunter, D., Evans, K., Rogers, D., Taylor, I., Goode, M., Silva, F., et al., (2013), "A Case Study into Using Common Real-Time Workflow Monitoring Infrastructure for Scientific Workflows", *Journal of Grid Computing*, v. 11, n. 3 (Sep.), p. 381–406.

Wozniak, J. M., Armstrong, T. G., Maheshwari, K., Lusk, E. L., Katz, D. S., Wilde, M., Foster, I. T., (2012), "Turbine: a distributed-memory dataflow engine for extreme-scale many-task applications", p. 1–12

paper:82

Definição e Avaliação de uma Abordagem para Extração e Rotulação de Conteúdo Obtido da *DeepWeb*

Augusto Ferreira de Souza, Ronaldo dos Santos Mello

Departamento de Informática e Estatística–Centro Tecnológico – Universidade Federal de Santa Catarina (INE/CTC/UFSC) Caixa Postal 476– Florianópolis, SC - Brasil.

augustofs@gmail.com, r.mello@ufsc.br

Abstract. *This paper presents an approach for the extraction and labeling of data presented in Deep Web databases. Such a data are extracted from a set of HTML pages generated as the result of a query posed on the hidden database through a Web form. Data labeling (and persistence) aims at providing further structured queries over this hidden content. Preliminary experiments had demonstrated that the proposed approach is promising, if compared with baselines. Other contributions are a joint-process for simultaneous data extraction and labeling, an automatic approach with the support of a knowledge base, and a labeling process of extracted records with content self-filling support for attributes with missing values.*

Resumo. *Este artigo apresenta uma solução para a extração e rotulação de dados contidos em bancos de dados na Deep Web. Esses dados são extraídos de um conjunto de páginas HTML gerado como resultado de uma consulta submetida ao banco de dados através de um formulário Web. A rotulação (e conseqüente persistência) destes dados viabiliza futuras consultas estruturadas sobre este conteúdo escondido. Uma avaliação preliminar demonstrou a eficácia da abordagem proposta em relação a baselines. Outros diferenciais deste trabalho são a realização simultânea de um processo de extração e de rotulação de dados, uma abordagem automática com suporte de uma base de conhecimento e um processo de rotulação de registros extraídos com suporte ao auto-preenchimento de atributos com valores ausentes.*

1. Introdução

O aumento do volume de dados disponíveis na *Deep Web* [Halevy et al. 2009] faz com que também aumente o interesse no acesso a essas informações por parte dos usuários. A *Deep Web* representa, dentre outras fontes de dados, bancos de dados disponíveis na *Web* cuja estrutura e conteúdo tornam-se visíveis (ou parcialmente visíveis) apenas quando apresentados em páginas dinâmicas criadas a partir do resultado de uma consulta definida sobre um formulário *Web* [Bergman, 2001]. O formulário *Web* é a principal interface de busca para estes bancos de dados ditos "escondidos" (BDEs).

Para se ter acesso às informações de um BDE são necessárias soluções para a sua descoberta na Web, submissão automática de consultas através dos seus formulários, assim como técnicas de extração e rotulação dos dados exibidos nas páginas de resultado, viabilizando o posterior consumo humano. A atividade de extração, em particular, é complexa devido à existência de uma grande variedade de *Web sites* com padrões diferenciados para a exibição do conteúdo de BDEs, bem como a

existência de muitas informações irrelevantes (menus, anúncios, etc.) que dificultam o reconhecimento do que é realmente relevante [Hong, 2010][Oro et al. 2011]. Já as abordagens de rotulação de dados geralmente utilizam um dicionário para ajudar na comparação dos dados para uma correta rotulação, porém não são capazes de detectar novas informações e assim construir registros com conteúdo mais completo [Zhao et al. 2008][Silva et al. 2011].

Assim sendo, o objetivo deste artigo é detalhar uma abordagem chamada *DeepEL* (*Deep Web Extraction and Labeling Process*) que realiza de forma automática a extração e rotulação de conteúdo obtido da *Deep Web*. O processo de rotulação é capaz de detectar informações que não estão disponíveis nos registros extraídos. Esta detecção é possível com o suporte de uma base de conhecimento (BC) construída para alguns domínios da *Deep Web*. A intenção da *DeepEL* é a construção sistemática de um BD sobre a *Deep Web* que possa servir de base para diversos serviços, como por exemplo, sistemas de busca na *Deep Web* e a construção de catálogos de BDEs. As principais contribuições deste artigo são:

- A definição de um processo de extração de conteúdo existente em BDEs e que não estão necessariamente presentes em rótulos e valores dos seus formulários de busca;
- Experimentos preliminares que indicam que o processo de extração é promissor, se comparado com *baselines* conhecidos na literatura;
- A definição de um processo de rotulação cujo benefício é uma melhoria de qualidade para futuras buscas estruturadas sobre dados de BDEs, uma vez que o conhecimento sobre os mesmos fica enriquecido com a complementação do conteúdo dos seus atributos.

Este artigo está organizado conforme segue. A seção 2 sumariza os principais trabalhos relacionados à extração e rotulação de dados na *Web*. A seção 3 detalha a abordagem *DeepEL*. A seção 4 apresenta os experimentos preliminares e a seção 5 é dedicada às conclusões e trabalhos futuros.

2. Trabalhos Relacionados

2.1 Extração de Dados

Soluções para extração de dados na *Web* definem algoritmos capazes de identificar e recuperar informações presentes em páginas *Web* [Kaiser and Miksch 2005]. Nenhum dos trabalhos relacionados ao tratamento desta problemática inclui um processo de rotulação dos dados extraídos, nem um processo automatizado quando há o suporte de uma ontologia ou BC [Embley et al. 1998] [Muslea et al. 2001] [Liu et al. 2003] [Phan et al.2005]. Essas limitações motivaram o desenvolvimento da *DeepEL*.

Dentre as soluções existentes, este trabalho considerou dois *baselines* durante a realização de experimentos: *Road Runner* [Muslea et al. 2001] e MDR [Liu et al. 2003]. Estes métodos são comumente referenciados na literatura [Hong, 2010][Oro et al. 2011]. *Road Runner* é uma abordagem automática que compara conteúdos HTML a fim de definir uma expressão regular para um conjunto de páginas de amostra. Depois de resolver incompatibilidades, uma expressão comum é utilizada para extrair registros de

dados de outras páginas na *Web*. O método MDR (*Mining Data Records*) gera uma árvore DOM de uma página HTML e a percorre de maneira *bottom-up* verificando a similaridade entre nodos adjacentes. A partir desta análise de similaridade, ele determina regiões de dados que serão consideradas para fins de extração.

2.2 Rotulação de Dados

Com relação à rotulação de dados, o trabalho de [Zhao et al. 2008] emprega uma técnica de aprendizado de máquina para rotulação de texto. Ele utiliza tabelas com dados estruturados como amostra para treinamento. Durante a fase de treinamento, a rotulação de conteúdos textuais é definida a partir de tabelas de referência. Assumindo que as sequências de texto a serem segmentadas vêm em lotes, elas devem estar em conformidade com a ordem dos atributos definidos no treinamento, para serem devidamente identificadas e rotuladas.

O trabalho de [Silva et al. 2011] introduz a abordagem JUDIE. Ela recebe como entrada um texto contendo registros de dados. Após, ocorre a segmentação dos dados e uma rotulação de valores potenciais, realizados através da comparação com dados relacionados ao domínio mantidos em uma BC. Após a segmentação e a primeira rotulação, um algoritmo baseado em um modelo de posicionamento e sequenciamento de dados rotula-os novamente, confirmando ou efetuando possíveis correções nos rótulos inicialmente rotulados.

Limitações desses trabalhos que motivaram o desenvolvimento da *DeepEL* são:

- No trabalho de [Zhao et al. 2008], o esquema assume uma ordem fixa conforme a amostra, prejudicando a rotulação dos dados que não seguem este padrão;
- Nenhuma abordagem realiza a complementação de informações que, apesar de pertencerem ao domínio, não estão presentes na fonte de dados a ser rotulada.

3. DeepEL

A abordagem *DeepEL* está centrada em dois processos: extração e rotulação de dados da *Deep Web*. A Figura 1 apresenta a arquitetura da abordagem e os componentes considerados. O componente externo *Motor* (1) encapsula o processo de descoberta de BDEs na *Web*, preenchimento de formulários *Web* e submissão de consultas, bem como a geração de páginas HTML com os resultados obtidos. Esse conjunto de páginas (2) é a entrada considerada pela *DeepEL*.

A *DeepEL* utiliza uma BC (4) como auxílio nos processos de extração e rotulação de dados. Ela foi especificada no formato XML e, nesta primeira versão, mantém conhecimento relativo a dois domínios populares na *Deep Web*: *Automóveis* e *Livros*. A modelagem da BC considerou a coleta dos principais atributos de cada domínio, conforme descrito no *corpus* do repositório colaborativo de dados *Freebase*[1], bem como o enriquecimento semântico deste conhecimento através da coleta (e posterior cadastro na BC) de termos sinônimos presentes na base de dados léxica *WordNet*[2]. Apesar da estrutura complexa no formato XML, a BC tende a ser reduzida,

[1] *http://www.freebase.com/*
[2] *http://wordnet.princeton.edu/*

pois mantém apenas atributos mandatários do domínio e exemplos de valores de atributos relevantes. Outras BCs disponíveis são muito mais volumosas, requerendo a indexação de muitos dados. Maiores detalhes sobre a modelagem da BC são omitidos devido a restrições de espaço. A Figura 2(b) apresenta um fragmento da BC com conteúdos de rótulos de atributos presentes no domínio de *Automóveis*, inclusive com sinônimos para alguns rótulos enriquecidos a partir do *WordNet*.

Os demais componentes da arquitetura são comentados à medida que os processos de Extração (3) e Rotulação (6) são detalhados, conforme segue.

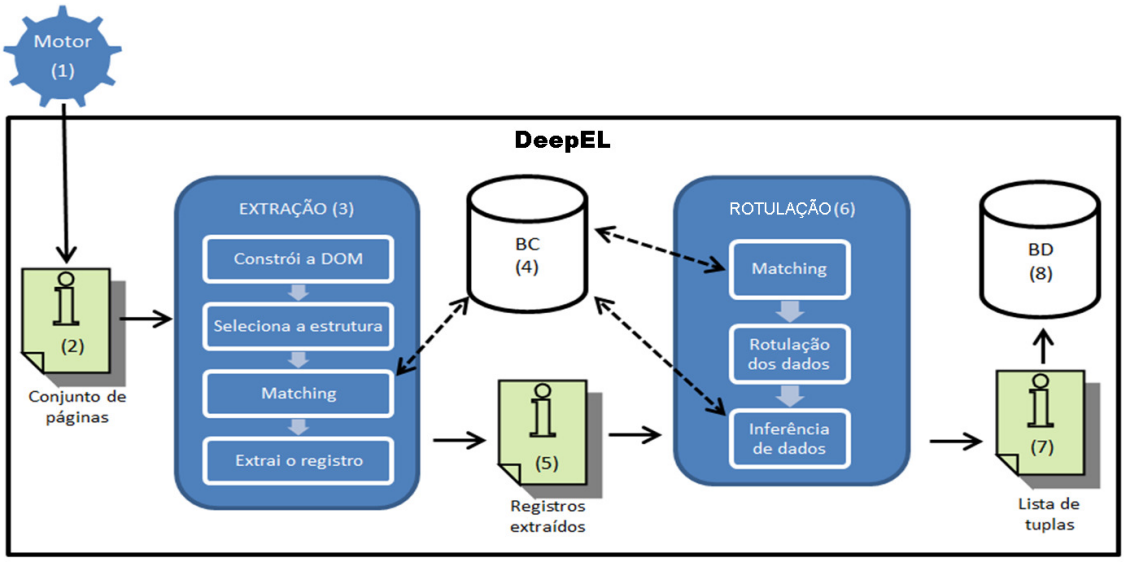


Figura 1. Arquitetura da Abordagem DeepEL.

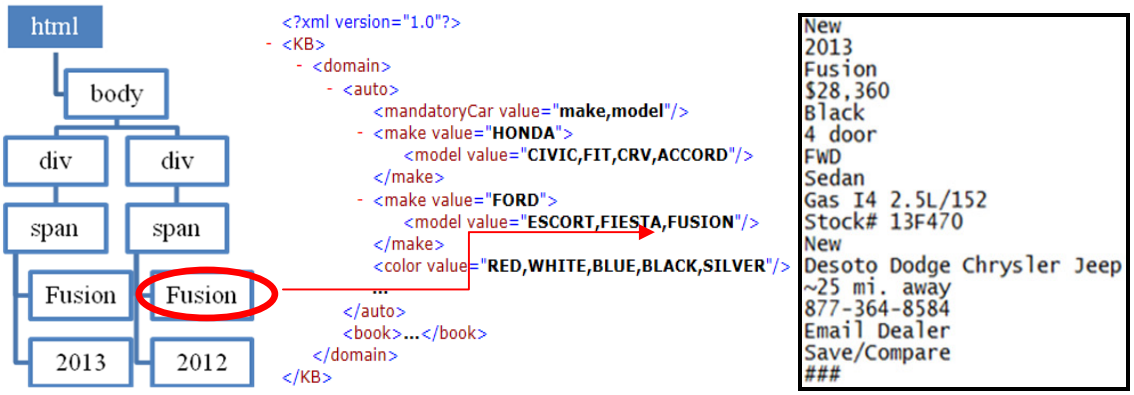


Figura 2. Página HTML no formato DOM (a). Exemplo parcial da BC no formato XML (b). Exemplo de arquivo com os registros extraídos (c).

3.1 Processo de Extração

O processo de extração adotado pela *DeepEL* é baseado nas Heurísticas 1 a 3, que visam melhorar a qualidade da recuperação dos registros de dados de BDEs presentes nas páginas HTML. Estas heurísticas são definidas a seguir.

Definição 1 (Heurística da Estrutura Irrelevante - HEI). As estruturas HTML definidas pelas *tags* "*script*", "*select*" e "*option*" são descartadas, pois provavelmente são funções ou campos de formulários sem conteúdo relevante.

Definição 2 (Heurística de Atributo Mandatário - HAM). Um atributo mandatário A_m é um atributo significativo de um determinado domínio, servindo para caracterizar o domínio de um registro que possua um valor de A_m.

Exemplo. O atributo *make* é um atributo mandatário no domínio de *Automóveis*, pois é uma propriedade inerente a qualquer registro de automóvel.

Definição 3 (Heurística da Estrutura Relevante - HER). A estrutura de representação de dados que mais se repetir na página HTML é onde provavelmente estão localizados os registros relevantes advindos do BDE.

A heurística HER é a mais importante para o bom funcionamento do extrator, pois é ela que decide o que é considerado relevante. Ela foi definida com base no trabalho de [Hong, 2010], porém, a *DeepEL* tem como diferencial o suporte da BC para confirmar se a estrutura mais frequente contém realmente os dados desejados. Com a heurística HEI analisa-se de forma mais minuciosa a estrutura HTML e desconsidera-se *tags* que não possuem conteúdo útil.

O Algoritmo 1 detalha o processo de extração. Ele recebe como entrada uma página HTML e considera o suporte da BC. Na linha 5, a página é instanciada no modelo DOM e na linha 6 é aplicada a HEI que remove estruturas irrelevantes. Na linha 7 é realizada uma busca por atributos mandatórios de um determinado domínio disponíveis na BC (HAM).

Algoritmo 1: Método de Extração.

```
1:  Entrada: páginaHTML;
2:  Início
3:    CaminhosCandidatos←{ }
4:    CaminhosRelevantes←{ }
5:    PDOM ← DOM_Parser(pagina)
6:    SDOM ← HEI(PDOM )
7:    para cada valor de HAM(v)∈ BC faça
8:      para cada termo t de um nodo folha n ∈ SDOM faça
9:        se matching(v, t) então
10:         CaminhosCandidatos←CaminhosCandidatos + Caminho (n)
11:       fim se
12:     fim para
13:   fim para
14:   CaminhosRelevantes← HER (CaminhosCandidatos)
15:   para ∀ termo t de um nodo folha n ∈ CaminhosRelevantes faça
16:     extrai(t)
17:   fim para
18: Fim
```

As Figuras 2a e 2b exemplificam o processo de *matching* entre o valor de um atributo mandatório da BC e um termo presente nas folhas de um nodo DOM da página HTML (linha 9). A função de similaridade utilizada aqui foi a *Jaro-Winkler* [Winkler,

1990], por ter obtido melhor desempenho nos experimentos preliminares. Vale ressaltar que, mesmo a BC sendo extensa, essa comparação por similaridade é realizada somente com o conteúdo das *tags* correspondentes a atributos mandatários dos domínios presentes na BC, como é o caso da *tag model*, que é um atributo mandatário no domínio de *Automóveis*. Após a identificação de tais registros, o caminho da *tag* raiz que representa o registro de dado até os nodos folha é considerado um *caminho candidato* (linha 10). Na sequência, na linha 14, é aplicada a HER sobre os caminhos candidatos para selecionar apenas os caminhos relevantes.

Por fim, na linha 16, ocorre a extração dos conteúdos que se encontram nos caminhos relevantes, gerando um arquivo de saída com os registros extraídos (5), conforme ilustrado na Figura 2c. Cada linha do arquivo é o valor de um campo do registro extraído de um nodo. Cada registro, por sua vez,é delimitado por "###".

3.2 Processo de Rotulação

O processo de rotulação adota uma abordagem de análise e caracterização de segmentos de texto, utilizando também a BC como apoio. Ele recebe como entrada o arquivo com os registros extraídos de uma página HTML e analisa cada um deles, campo a campo, para fins de construção de uma tupla a ser rotulada. Para tanto, o conteúdo de cada campo é comparado com os conteúdos da BC. O Algoritmo 2 detalha este processo. Inicialmente, na linha, 3 é considerada uma heurística para detecção do domínio ao qual pertencem os registros a serem rotulados. Ela é definida a seguir.

Definição 4 (Heurística de Detecção de Domínio - HDD). Uma detecção de um domínio *D* ocorre quando conteúdos de um registro extraído casam, por similaridade, somente com valores de atributos mandatários de *D* presentes na BC.

Algoritmo 2: Método de Rotulação.

```
1:  Entrada: conjunto de registros extraídos RegEx;
2:  Início
3:    d ← HDD(RegEx)
4:    se d ≠ nulo então
5:      para cada atributo com tag t e valor v ∈ BC(d) faça
6:        para cada registro r ∈ RegEx faça
7:          para cada campo c ∈ r faça
8:            se matching(v, c) então
9:              rotula(c,t,d)
10:           Senão
11:             se detectaPadrao(c) então
12:               rotula(c,t,d)
13:             fim se
14:           fim se
15:         fim para
16:         para cada atributo t da tupla rotulada tp |valor (t,tp) = nulo faça
17:           itemDetectado ← HDC (t)
18:           se itemDetectado ≠ nulo então
19:             rotula(itemDetectado,t,d)
20:           fim se; fim para; fim para; fim para; fim se
21: Fim
```

A HDD se baseia na hipótese considerada pela heurística HAM. Uma vez que os atributos mandatários são específicos de cada domínio, eles possibilitam a detecção deste domínio. Se um domínio *D* é detectado (linha 4), cada registro é lido e seus campos comparados com os termos da BC pertencentes a *D* (linhas 5 a 7). As similaridades entre termos da BC e os campos dos registros são identificadas (linha 8) utilizando-se, mais uma vez, a função *Jaro-Winkler*. Quando uma correspondência é identificada, o termo e o seu significado (*tag*) são armazenados na tabela do BD destino (8) correspondente ao domínio *D* (linha 9).

O processo de rotulação considera o reconhecimento de alguns padrões de valores para determinados atributos de domínios (linha 11). Esse reconhecimento é realizado através de expressões regulares e funções específicas. Para o domínio de *Automóveis*, por exemplo, padrões descobertos são ano, preço e quilometragem. Ainda, um diferencial deste processo é a detecção de conteúdos para atributos nulos de registros a serem rotulados. Isto ocorre através da identificação das correspondências entre campos de registros e amostras de dados da BC para determinados domínios, como é o caso do modelo *Fusion* para a marca *Ford* mostrada na Figura 2b. Neste caso, se o modelo é conhecido e a marca não, é possível determinar o valor da marca caso esta dependência de valores esteja presente na BC, conforme rege a heurística HDC definida a seguir. Desta forma, enriquece-se o conteúdo deste registro no ato da rotulação.

Definição 5 (Heurística de Detecção de Conteúdo - HDC). Uma detecção de conteúdo de atributo ocorre quando existe uma dependência de valor entre atributos X e Y, sendo X o atributo determinante e Y o atributo determinado na hierarquia da BC. Neste caso, se o valor de Y é rotulado, o valor de X também o será.

Esta heurística é considerada na linha 17 após a verificação da existência de algum valor de atributo nulo na tupla construída para fins de rotulação (linha 16). Caso ocorra a detecção de conteúdo, o mesmo é rotulado (linha 19). Por fim, as tuplas criadas (7) são armazenadas em um BD relacional (8). A Figura 3 mostra exemplos, gerados pelo processo de rotulação para o domínio de *Automóveis*, de registros e campos devidamente preenchidos e prontos para serem persistidos como tuplas no BD relacional. Detalhes sobre o projeto deste BD são omitidos por restrições de espaço.

Nr:1	make: FORD	model: FUSION	door: 4	price: 28.360	year: 2013	color: BLACK	
Nr:2	make: FORD	model: FUSION	door: 2	price: 22.888	year: 2012	color: GRAY	mileage: 10.054
...	...	...	...	...	...	...	...

Figura 3. Exemplos de registros rotulados para o domínio de *Automóveis*.

4. Experimentos

As amostras de dados utilizadas nos experimentos foram obtidas a partir do repositório de páginas *Web* mantido pelo sistema de busca *Deep Peep* [Barbosa et. al 2010]. *Deep Peep* é uma máquina de busca para formulários *Web* pertencentes a alguns domínios da *Deep Web*. Os dois domínios escolhidos para os experimentos são os que possuem maior volume de dados indexados por esta máquina de busca. A partir disso, realizou-se o acesso a 18 páginas desses domínios e a submissão de consultas aos formulários presentes nelas. As consultas geraram 60 páginas de resultado, que foram consideradas como entrada para a *DeepEL*. Essas páginas geraram 1376 registros no domínio de

Automóveis e 352 registros no domínio de *Livros*. Uma pequena amostra destes dados foi utilizada também na população da BC.

Os experimentos avaliaram a qualidade da extração e da rotulação de dados nos dois domínios. Primeiramente, realizaram-se experimentos somente para a etapa de extração. Nesta etapa, os resultados obtidos foram comparados com os resultados gerados pelos *baselines* MDR e *Road Runner*. A justificativa para a comparação da *DeepEL* com essas duas abordagens é que, apesar delas não considerarem aspectos semânticos em seus métodos, como o suporte de uma BC, elas se enquadram dentre as poucas que realizam o processo de extração de forma automática, assim como a *DeepEL*, e possuem códigos-fonte disponíveis para *download*.

O desempenho da *DeepEL* foi também avaliado através de experimentos para a etapa de rotulação utilizando os registros recuperados durante a etapa de extração. Cabe salientar que um trabalho anterior definiu informalmente o processo de rotulação e fez uma avaliação apenas dos dois *baselines* [Souza e Mello, 2013]. Este artigo propõe um processo de extração para a *DeepEL*, o compara com os *baselines* e ainda define formalmente os dois processos em termos de heurísticas consideradas.

A Tabela 1 apresenta os resultados do processo de extração da *DeepEL* em comparação com os *baselines*. A tabela apresenta os resultados em termos de precisão (P), revocação (R) e medida F (F), bem como o tempo de processamento necessário para a extração dos registros. Utilizou-se *make* e *model* como atributos mandatórios para o domínio de *Automóveis*, bem como *publisher* e *author* para o domínio de *Livros*.

Tabela 1. Resultados do processo de extração de dados.

Abordagem	Domínio	P	R	F	Tempo (s)
Road Runner	Automóveis	0,94	0,95	0,95	18,10
MDR		0,92	0,95	0,93	13,56
DeepEL (Extração)		0,94	**0,96**	0,95	**10,09**
Road Runner	Livros	0,84	0,91	0,88	4,95
MDR		0,81	0,90	0,85	3,62
DeepEL (Extração)		0,87	0,90	**0,89**	**2,88**

Nota-se que a etapa de extração da *DeepEL* apresenta resultados próximos à precisão e à revocação dos *baselines*. Obteve-se a melhor revocação para o domínio de *Automóveis*, ou seja, o melhor percentual de acertos em termos de recuperação de registros desejados presentes nas páginas de resultado, pois este domínio apresenta propriedades mais homogêneas, facilitando assim a identificação dos registros desejados. Outro ponto positivo da *DeepEL* foi a melhor medida F para o domínio de *Livros*. Além disso, ela realizou a extração com rapidez (aproximadamente 20% mais veloz que o *baseline* mais rápido), mesmo considerando o acesso a uma BC que, por ser enxuta, pode ser manipulada totalmente em memória.

A Tabela 2 apresenta o desempenho da *DeepEL* em termos de rotulação, ou seja, os valores de precisão, revocação e medida F considerando o que é rotulado em relação à quantidade de registros extraídos disponíveis nos arquivos. Os bons resultados (medida F superior à 90%) se justificam pelo fato de grande parte dos registros rotulados possuírem algum tipo de informação presente na BC, como por exemplo, ocorrências de modelos de carros para o domínio de *Automóveis*. Cabe ressaltar que, mesmo com a BC contendo uma pequena amostra de valores de alguns atributos

mandatórios dos domínios, isso já torna possível a rotulação correta de um conjunto de dados muito mais volumoso, pois basta haver um casamento parcial de conteúdo dos dados da amostra com o registro extraído para ocorrer a rotulação.

Tabela 2. Avaliação da rotulação dos conteúdos extraídos pela *DeepEL*.

Domínio	Precisão	Revocação	Medida F
Automóveis	0,94	0,96	0,95
Livros	0,90	0,93	0,91

Por fim, a Tabela 3 apresenta a quantidade de valores de campos que foram complementados para o conjunto de registros considerado nos experimentos. Observa-se aqui que, aplicando o mecanismo de detecção de conteúdo, foi possível um ganho de qualidade na rotulação de registros de mais de 10%. Este percentual não foi elevado pois grande parte dos registros a serem rotulados possuía valores para todos os campos previstos no domínio. Mesmo assim, é uma contribuição da abordagem *DeepEL*.

Tabela 3. Ganho com o enriquecimento de informação durante a rotulação.

Domínio	#Registros	Valores de Campos Detectados	Ganho em Termos de Registros Complementados
Automóveis	1376	150	10,90 %
Livros	352	25	7,10 %

5. Conclusão

Diferente do estado da arte sobre extração e rotulação de dados da *Deep Web*, que não trata essas atividades como um processo único e possui limitações em termos da extração distinta de metadados e valores, além da falta de complementação de informações, este artigo define e avalia uma abordagem, denominada *DeepEL*, para extração de registros de dados estruturados de páginas de resultados de consultas submetidas a BDEs da *DeepWeb*, bem como a rotulação e complementação de conteúdo destes registros. A *DeepEL* é uma contribuição para a problemática de tornar visível, estruturado e contextualizado o conteúdo presente em BDEs, contando ainda com o suporte de uma BC projetada para manter metadados e amostras de valores mais significativos dos principais domínios da *DeepWeb*. A BC é utilizada na comparação e inferência de conteúdo para registros a serem extraídos.

Os resultados obtidos através de uma avaliação experimental indicaram que a *DeepEL* executa no mínimo com qualidade similar a dos principais métodos de extração automáticos, porém com um melhor desempenho de tempo e de qualidade de rotulação nos domínios considerados. Além disso, ela proporcionou um ganho de até 10% no enriquecimento do conteúdo dos registros rotulados. Estima-se que este ganho seja maior para um volume maior de registros em domínios cujo conteúdo seja mais homogêneo, como é o caso do domínio de *Automóveis*, um dos mais extensos da *Deep Web*. Deseja-se executar novos experimentos com um maior volume de dados extraídos de páginas *Web* para validar esta hipótese.

Alguns trabalhos futuros estão em planejamento, entre eles: *(i)* avaliar o desempenho da *DeepEL* em outros domínios da *DeepWeb*; *(ii)* permitir a extensão da BC a partir de novos conteúdos extraídos, visando ampliar o seu conhecimento; *(iii)* comparar o desempenho da *DeepEL* com outros métodos de rotulação, caso se obtenha acesso às suas implementações. Por fim, cabe salientar que a *DeepEL* não está livre de

problemas de ambigüidade durante o processo de rotulação. Entretanto, os exemplos na BC e seu posterior enriquecimento, assim como uma análise de posicionamento de conteúdos dos registros, conforme utilizado por [Silva et al. 2011], podem minimizar esta questão, que também é alvo de investigações futuras.

Referências

Barbosa, L., Nguyen, H., Nguyen, T., Pinnamaneni, R., and Freire, J. (2010). Creating and Exploring Web Form Repositories. In: ACM SIGMOD Int. Conf. on Management of Data, p.1175-1178.

Bergman, M. K. (2001). White Paper: The Deep Web: Surfacing Hidden Value. Journal of Electronic Publishing, v.7, n.1.

Embley, D. W., Campbell, D. M. and Smith, R. D. (1998). Ontology-Based Extraction and Structuring of Information from Data-Rich Unstructured Documents. In: Int. Conf. on Information and Knowledge Management, p. 52-59.

Halevy, A., Madhavan, J., Afanasiev, L. and Antova, L. (2009). Harnessing the Deep Web: Present and Future. In: Int. Conf. on Innovative Data Systems Research.

Hong, Jun; He, Zhongtian and Bell, David A. (2010). An Evidential Approach to Query Interface Matching on the Deep Web. Information Systems, v.35, n.2, p.140-148.

Kaiser, K. and Miksch, S. (2005). Information Extraction. A Survey. Technical Report,Vienna University of Technology.

Liu, B., Grossman, R. and Zhai, Y. (2003). Mining Data Records in Web Pages. In: ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining, p.601-606.

Muslea, I., Minton, S. and Knoblock, C. (2001). Hierarchical Wrapper Induction for Semistructured Information Sources. Autonomous Agents and Multi-Agent Systems Archive, v.4, n.1, p.93-114.

Oro, E. and Ruffolo, M. (2011). SILA: a Spatial Instance Learning Approach for Deep Web Pages. In: Int. Conf. on Information and Knowledge Management, p.2329-2332.

Phan, X., Horiguchi, S. and Ho, T. (2005). Automated Data Extraction from the Web with Conditional Models. Int. Journal of Business Intelligence and Data Mining, v.1, n.2, p.194-209.

Silva, A. S., Cortez, E., Oliveira, D., Moura, E. S. and Laender, A. H. F. (2011). Joint Unsupervised Structure Discovery and Information Extraction. In: ACM SIGMOD Int. Conf. on Management of Data, p.12-16.

Souza, A. F. and Mello, R. S. (2013). DeepEC: An Approach for Deep Web Content Extraction and Cataloguing. In: European Conf. on Information Systems.

Winkler, W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. In: Section on Survey Research Methods, American Statistical Association, p.354-359.

Zhao, C., Mahmud, J. and Ramakrishnan, I. V. (2008). Exploiting Structured Reference Data for Unsupervised Text Segmentation with Conditional Random Fields. In: Int. Conf. on Data Mining, p.420-431.

paper:81

Gerência de Proveniência Multigranular em Linked Data com a Abordagem *ETL4LinkedProv*

Rogers Reiche de Mendonça[1,3], Sérgio Manuel Serra da Cruz[2], Maria Luiza M. Campos[3]

[1] Petróleo Brasileiro S.A - Petrobras

[2] Universidade Federal Rural do Rio de Janeiro - PET-SI/PPGMMC/UFRRJ

[3] Universidade Federal do Rio de Janeiro - PPGI/UFRJ

rogers.mendonca@petrobras.com.br, serra@ufrrj.br, mluiza@ppgi.ufrj.br

Abstract. *This paper presents the ETL4LinkedProv approach to manage the collection and publication of provenance metadata with different levels of granularity, as Linked Data. The approach uses ETL workflows and a novel component named Provenance Collector Agent. Its application in a real scenario is presented and the impact of the fine-grained provenance in the ETL workflow runtime and in the number of published RDF triples is evaluated.*

Resumo. *Este trabalho apresenta a abordagem ETL4LinkedProv para gerenciar a coleta e publicação dos metadados de proveniência com diferentes níveis de granulosidades, sob a forma de Linked Data. A abordagem utiliza workflows ETL e um componente denominado Agente Coletor de Proveniência. Um exemplo de utilização da abordagem em um cenário real é descrito e o impacto da proveniência de granulosidade fina no tempo de execução do workflow ETL e no número de triplas RDF publicadas é avaliado.*

1. Introdução

Nos últimos anos, um número crescente de provedores de dados vem adotando um conjunto de tecnologias e boas práticas da Web Semântica para publicar e interligar dados estruturados na Web, formando assim a Web de Dados. Os princípios determinados pela abordagem Linked Data (LD) e adotados pela Web de Dados oferecem simplicidade e flexibilidade para que dados sejam representados, interligados e possam interoperar na Web [Bizer et al. 2009; Heath and Bizer 2011].

O acelerado crescimento da Web de Dados incentivou o surgimento de uma nova geração de aplicações para consumir os dados publicados e explorar o potencial por eles oferecido [Heath and Bizer 2011]. Por exemplo, iniciativas governamentais nacionais [Breitman et al. 2012] e internacionais [Sheridan and Tennison 2010], fomentadas por acordos de transparência e colaboração, passaram a publicar seus dados abertos seguindo os princípios de LD. Ações desta natureza, aliadas à propagação de clientes móveis e serviços de acesso público, trouxeram uma nova perspectiva de empoderamento e participação aos cidadãos [Cordeiro et al. 2011a].

Haja vista este crescimento e relevante uso da Web de Dados, questões pertinentes à qualidade do LD passaram a desempenhar um papel cada vez mais importante. Sem subsídios para a avaliação da qualidade e da confiabilidade dos dados publicados como LD, o consumo e a exploração dos mesmos podem ser comprometidos. Sendo assim, os

metadados de proveniência surgem também neste contexto com uma função importante de apoio para a avaliação da qualidade e da confiabilidade dos dados [Gil et al. 2010].

Este trabalho tem como objetivo discutir a abordagem *ETL4LinkedProv* para coleta e publicação de metadados de proveniência multigranular e seu impacto no processo de publicação de LD realizado dentro dos limites de uma organização, denominado *Processo de Preparação e Transformação dos Dados*. A abordagem *ETL4LinkedProv* amplia trabalhos anteriores [Campos and Guizzardi 2010; Mendonça et al. 2013], que introduziram *workflows* de Extração, Transformação e Carga (ETL) de dados para publicar LD. A abordagem contempla um componente, denominado Agente Coletor de Proveniência (ACP), que captura, interliga e armazena metadados de proveniência de diferentes granulosidades, relacionados tanto à composição (proveniência prospectiva), quanto à execução (proveniência retrospectiva) do processo de publicação. Tais metadados também são publicados como LD, possibilitando não só a exploração conjunta dos dados de domínio e dos metadados de proveniência por meio de consultas SPARQL, como também investigações sobre a qualidade do LD.

Este artigo está organizado da seguinte forma: a seção 2 discute trabalhos relacionados; a seção 3 introduz a abordagem *ETL4LinkedProv* e sua implementação; a seção 4 ilustra um caso de uso da abordagem e discute análises quantitativas realizadas com diferentes configurações do nível de granulosidade dos metadados de proveniência e, por fim, a seção 5 apresenta considerações finais e trabalhos futuros.

2. Trabalhos Relacionados

A proveniência de dados representa a ancestralidade de um objeto e pode ser descrita em diferentes termos, dependendo do domínio abordado [Gil et al. 2010]. Em geral, ela fornece a documentação que é essencial para registrar a origem, a qualidade e a autoria dos dados, assim como a capacidade de reproduzir e validar resultados de processos comerciais ou científicos. Os diferentes tipos de proveniência (prospectiva – relativa à especificação do processo, ou retrospectiva – relativa a sua execução) e seu nível de detalhamento (fina ou grossa) [Cruz et al. 2009] são facetas que podem ser exploradas no contexto da qualidade de LD. Quanto menor a granulosidade, mais refinadas serão as possibilidades de investigação entre os dados publicados e seus metadados de proveniência.

Dentre os principais trabalhos que tratam do apoio semântico à proveniência em LD, destacamos o VoID [Alexander et al. 2009], um vocabulário que permite a descrição de metadados sobre *datasets* RDF (metadados gerais, de acesso, estruturais e *linksets*), e o modelo OPM [Moreau et al. 2011], concebido para permitir a interoperabilidade dos metadados de proveniência entre sistemas. O modelo OPM foi suplantado recentemente pela especificação PROV do W3C, cujo modelo de dados [Moreau and Missier 2013] é fundamentado nos conceitos *Entidade* (estado imutável de um objeto), *Atividade* (ação executada sobre ou com entidades) e *Agente* (responsável por atuar em uma atividade).

Dentre os trabalhos sobre a proveniência em LD, poucos discutem as questões do rastreamento das transformações de dados que ocorrem dentro dos limites de uma organização (agência governamental, empresa pública ou privada), antes que um *dataset* seja exposto como LD. Frequentemente, no contexto das organizações, os *datasets* são heterogêneos, não se alinham com os formatos desejados pelos analistas e, portanto, necessitam passar por processos de preparação e pré-integração antes de serem

triplificados. Relacionada a esta questão, Cordeiro et al. (2011b) propuseram uma abordagem inovadora para apoiar a exposição, a partilha e a associação de *datasets* de governo na forma de LD. Além de utilizarem *workflows* ETL para orquestrar o processo de publicação de LD, eles desenvolveram estratégias para capturar a proveniência sobre os esforços de transformação dos dados. Posteriormente, Freitas et al. (2012) e Omitola et al. (2012) investigaram perspectivas complementares desta abordagem. Freitas et al. (2012) definiram um modelo de três camadas para representação da proveniência do processo de publicação de LD e Omitola et al. (2012) enfatizaram o potencial do uso de ferramentas interativas de transformação de dados (IDT) para apoiar os esforços de transformação de dados e definiram uma arquitetura de gerenciamento dos metadados de proveniência baseada no modelo de representação definido por Freitas et al. (2012).

Por fim, é importante destacar que os trabalhos supracitados, diferentemente da abordagem aqui descrita, não discutem os métodos de coleta e publicação de dados legados. Além disso, não consideram a especificação PROV para LD, nem as distintas granulosidades dos metadados de proveniência relacionados às atividades das etapas do ciclo de vida de LD definido pelo projeto LOD2 [Auer et al. 2012].

3. *ETL4LinkedProv*: Coleta e Publicação de Proveniência como Linked Data nas Organizações

Nossa abordagem foca nos diferentes tipos de metadados de proveniência produzidos na etapa de *Extração* do ciclo de vida do LOD2 e amplia um trabalho anterior [Mendonça et al. 2013], que abrange metadados de proveniência das demais. Outras etapas intimamente associadas ao nosso trabalho são a de *Classificação / Enriquecimento de Dados* – que visa a anotação dos dados publicados em ontologias para facilitar integração, fusão e posteriores buscas e a etapa de *Análise de Qualidade* – que explicita que o suporte analítico pode se beneficiar de metadados de proveniência, de contexto e de estrutura.

Vale ressaltar que a abordagem *ETL4LinkedProv* atua em uma extensão da etapa de *Extração* do ciclo de vida do LOD2. Conforme mencionado anteriormente, no contexto de organizações, atividades complementares de limpeza, consolidação, agregação e pré-integração dos dados frequentemente necessitam ser executadas entre a extração dos dados e a triplificação dos mesmos, para publicá-los como LD. Sendo assim, a abordagem *ETL4LinkedProv* utiliza *workflows* ETL para orquestrar a publicação de fontes de dados heterogêneas como LD, com a captura e publicação dos metadados de proveniência por meio de triplas RDF anotadas em ontologias pré-existentes.

3.1. Agente Coletor de Proveniência

O Agente Coletor de Proveniência (ACP) é um componente de software que encapsula o *workflow* ETL responsável pela publicação de LD. Ele captura os metadados de proveniência prospectiva e retrospectiva e faz a interligação aos dados de domínio, durante a execução do processo de publicação. A execução de um *workflow* ETL consiste na execução de um conjunto de passos interligados, por onde os dados fluem de maneira unidirecional. Cada passo corresponde ao conceito *Atividade* da especificação PROV e é responsável pela execução de um processo de extração, transformação ou carga de dados; pela execução de atividades auxiliares como transferir arquivos e enviar *emails* ou pela execução de um *sub-workflow*. As tarefas realizadas pelo ACP podem ser agrupadas em 3 etapas distintas: *captura*, *interligação* e *armazenamento temporário* dos metadados de proveniência.

Na etapa de captura, o ACP monitora eventos relacionados ao *workflow* ETL e, sempre que um dos eventos ocorre, realiza a coleta dos metadados de proveniência de diferentes granulosidades. Os eventos monitorados são: (i) o início e o término do *workflow* principal ou de um dos seus *sub-workflows*; (ii) o início e o término de cada passo executado; e (iii) a leitura de dados por determinado passo. Um subconjunto de tipos de passos pode ser selecionado, a fim de que somente os passos relacionados aos tipos selecionados tenham a proveniência com nível de granulosidade fina capturada pelo ACP. Esta seleção de tipos de passos visa minimizar os impactos do grande volume de metadados gerado pela estratégia de coleta da proveniência com granulosidade fina, sem, no entanto, negligenciar a coleta dos metadados de proveniência das atividades mais relevantes do processo de publicação de LD.

Na etapa de interligação, o ACP interliga os metadados de proveniência coletados na etapa de captura. Os metadados de proveniência interligados referem-se tanto ao processo de publicação, quanto aos dados de domínio publicados. Além disso, os metadados de proveniência retrospectiva são interligados aos seus respectivos metadados de proveniência prospectiva. Por fim, na etapa de armazenamento temporário, os metadados são armazenados para, posteriormente, também serem extraídos, processados e publicados como triplas RDF pelo *Processo de Preparação e Transformação dos Dados*. O modelo conceitual de dados do repositório temporário utilizado pelo ACP (Figura 1) é formado por 13 entidades, que separam e relacionam os metadados sobre a composição e os metadados sobre a execução do processo de publicação.

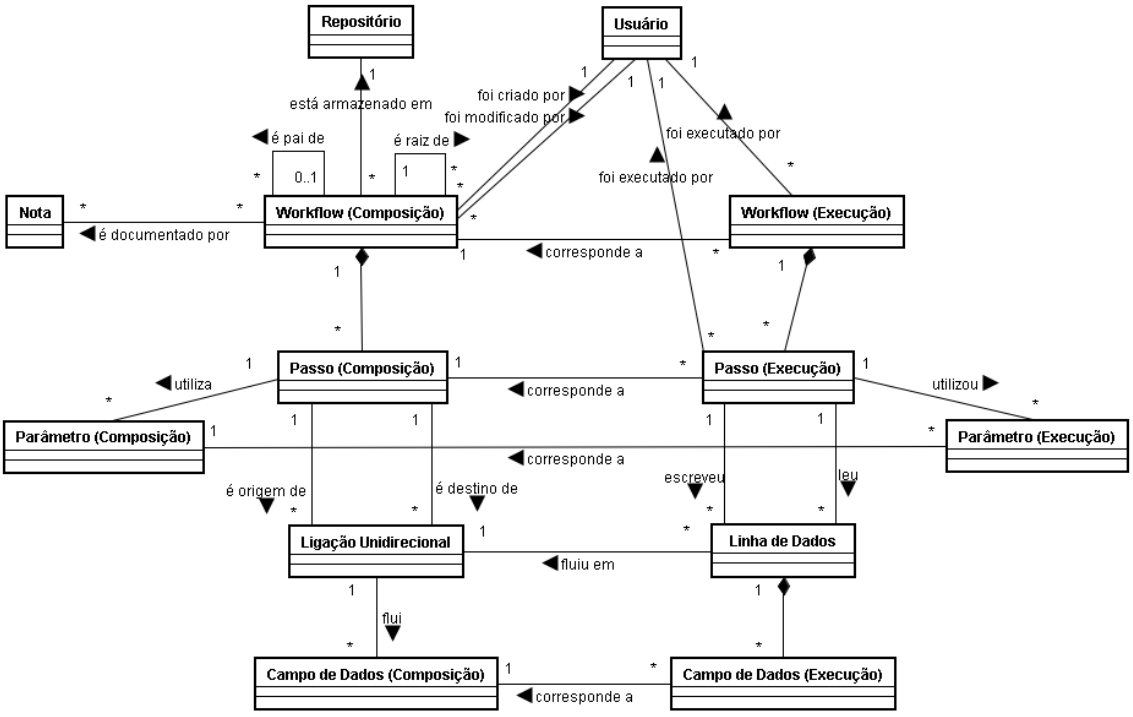


Figura 1. Modelo conceitual do repositório temporário utilizado pelo Agente Coletor de Proveniência para armazenar os metadados de proveniência.

Para apoiar semanticamente a publicação dos metadados de proveniência armazenados temporariamente pelo ACP, a abordagem *ETL4LinkedProv* utiliza um conjunto de ontologias já existentes, que estendem sucessivamente o detalhamento da representação da proveniência no contexto de LD. A ontologia PROV-O

(http://www.w3.org/ns/prov#) é utilizada como base para representar a semântica do *workflow* e possibilitar a interoperabilidade da proveniência na Web de Dados. A ontologia OPMW (http://www.opmw.org/ontology/) é utilizada como extensão da PROV-O para distinguir a semântica sobre a composição do *workflow* (proveniência prospectiva) e a semântica sobre a execução do *workflow* (proveniência retrospectiva). A ontologia Cogs (http://vocab.deri.ie/cogs#) é utilizada como extensão da PROV-O para representar de maneira expressiva os conceitos do processo de ETL. Adicionalmente, uma ontologia específica ao domínio da aplicação e os vocabulários Dublin Core (DC) e FOAF podem ainda ser utilizados para complementar a representação dos metadados de proveniência publicados.

3.2. Implementação da *ETL4LinkedProv* no Kettle

ETL4LinkedProv foi concebida para ser implementada independente da particularidade de uma ferramenta ETL. Neste protótipo, utilizamos o Kettle (Pentaho Data Integration) [Casters et al. 2010]; com o Kettle, dois tipos de componentes (*transformations* e *jobs*) podem ser utilizados para especificar o *workflow* ETL. Um *transformation* consiste de um conjunto de passos conectados, onde cada passo, denominado *step*, é responsável por uma atividade de extração, transformação ou carga de dados. A conexão (*hop*) entre dois *steps* permite que os dados fluam em um único sentido e de maneira assíncrona. Um *job* também consiste de um conjunto de passos conectados. No entanto, os passos de um *job*, denominados *job entries*, são responsáveis por executar um *transformation*, outro *job* ou atividades auxiliares como transferir arquivos ou executar operações de validação. A conexão (*hop*) entre dois *job entries* determina a ordem de execução deles, que, diferente dos passos do *transformation*, são executados de maneira síncrona. Tanto um *transformation*, quanto um *job* podem possuir notas de documentação e os metadados de suas especificações podem ser armazenados em tabelas de um repositório do tipo banco de dados ou em arquivos XML de um repositório do tipo sistema de arquivos.

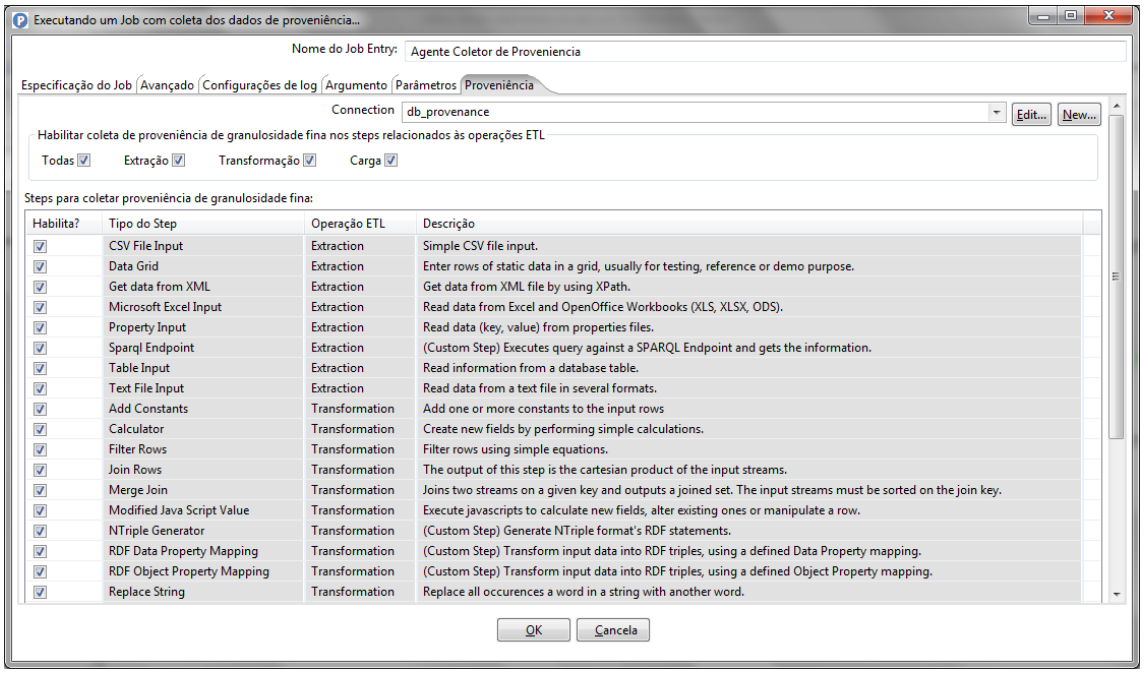


Figura 2. Interface de configuração do Agente Coletor de Proveniência implementado no Kettle.

Apesar de oferecer um conjunto significativo de *steps* e *job entries* que realizam uma série de atividades de extração, transformação e carga de dados, o Kettle não oferece, em sua instalação padrão, passos específicos para o contexto de LD. Assim, para suprir esta carência e possibilitar a implementação da abordagem *ETL4LinkedProv*, foi desenvolvido um conjunto de *steps*, denominado ETL4LOD, que permite a publicação tanto dos dados de domínio, quanto dos metadados de proveniência, em um banco de triplas RDF. O ACP também foi implementado com a API Java do Kettle, mas como um *job entry*, cujo tipo foi denominado *Provenance Collector Agent*. Sua interface de configuração (Figura 2) apresenta uma aba denominada "Proveniência", que possibilita a configuração da conexão com o banco de dados relacional utilizado para armazenamento temporário dos metadados de proveniência e a seleção dos tipos de *steps*, cujos metadados de proveniência com nível de granulosidade fina serão coletados. Os *steps* ETL4LOD e o *job entry Provenance Collector Agent* encontram-se disponíveis em http://greco.ppgi.ufrj.br/lodbr/index.php/principal/etl4linkedprov.

4. Caso de Uso – Integração de Dados de Agências Governamentais

Esta seção apresenta o caso de uso adotado neste trabalho. No cenário da pesquisa científica nacional, encontramos hoje dificuldades de reutilizar, consumir conjuntamente e explorar os dados das diferentes organizações de fomento à pesquisa. Por exemplo, tarefas como correlacionar os financiamentos concedidos por organizações distintas com a produtividade em publicações científicas geralmente se deparam com diversos obstáculos. Considerando este problema, um exemplo de aplicação da *ETL4LinkedProv* foi desenvolvido envolvendo dados reais do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Rede Nacional de Ensino e Pesquisa (RNP).

4.1. Integração de dados do CNPq e da RNP

Caso o CNPq e a RNP publicassem suas informações como LD por meio da abordagem *ETL4LinkedProv*, além da vantagem de ter seus dados interligados e possibilitar o consumo conjunto destes dados por aplicativos da Web Semântica, seria possível utilizar os metadados de proveniência para uma posterior verificação da linhagem dos dados de domínio publicados. Esta verificação seria um subsídio importante de apoio à validação da consistência entre os dados publicados pelo CNPq e pela RNP, promoveria um melhor entendimento dos seus dados de domínio e permitiria a exploração conjunta com qualidade e confiabilidade.

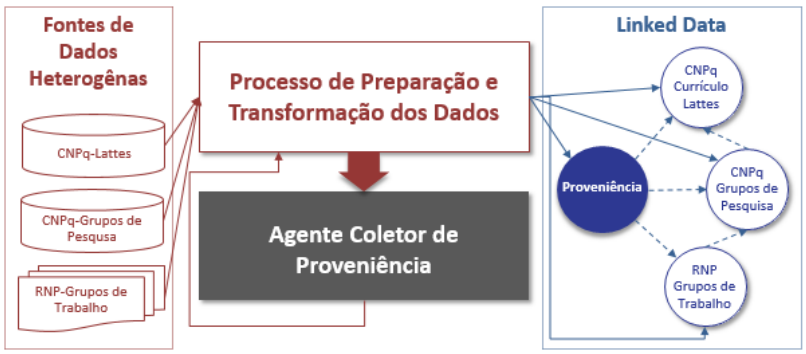


Figura 3. Visão geral do cenário utilizado neste trabalho.

Neste cenário, o protótipo da *ETL4LinkedProv* é capaz de encapsular um *workflow* ETL, que integra e publica os dados de duas fontes de dados do CNPq e uma

fonte de dados da RNP. As duas fontes de dados do CNPq, ambas armazenadas em banco de dados relacional, disponibilizam informações sobre os CV Lattes e sobre os Grupos de Pesquisa respectivamente. Já a fonte de dados da RNP, armazenada em um repositório XML, disponibiliza informações sobre os projetos financiados via Grupos de Trabalho do programa Grupos de Trabalho da RNP. A Figura 3 oferece uma visão geral do cenário.

Após a execução, os dados do CNPq e da RNP encontram-se publicados e interligados através de três grafos RDF: o primeiro contém as triplas RDF dos CV Lattes; o segundo contém as triplas RDF dos Grupos de Pesquisa do CNPq e o terceiro contém as triplas RDF dos Grupos de Trabalho da RNP. Um quarto grafo contém as triplas RDF referentes aos metadados de proveniência capturados pelo ACP. O grafo RDF de proveniência encontra-se interligado aos três grafos RDF de domínio, permitindo que consultas SPARQL de exploração conjunta entre os dados do CNPq, os dados da RNP e seus respectivos metadados de proveniência sejam realizadas.

4.2. Análise quantitativa de desempenho e do volume de triplas RDF

A partir da relação entre os dados extraídos e o número de triplas RDF publicadas pela abordagem *ETL4LinkedProv*, é possível obter métricas que evidenciam o impacto da granulosidade da proveniência no desempenho e no volume de dados publicados pelo *Processo de Preparação e Transformação dos Dados*. Deste modo, 3 configurações distintas do nível de granulosidade da proveniência capturada foram aplicadas no caso de uso descrito na sub-seção anterior.

Na primeira configuração, o *workflow* ETL que publicou os dados das fontes CNPq-Lattes, CNPq-Grupos de Pesquisa e RNP-Grupos de Trabalho foi executado sem estar encapsulado pelo ACP. Nas outras, o mesmo *workflow* foi executado encapsulado pelo ACP, com a diferença de que, na segunda configuração, nenhum tipo de passo foi selecionado para ter a proveniência coletada com granulosidade fina e, na terceira configuração, todos os tipos de passos foram selecionados para terem a proveniência coletada com granulosidade fina. A infraestrutura utilizada nas execuções do exemplo de aplicação contemplou a edição comunitária do SGBD MySQL (versão 5.0.51a), banco de triplas Virtuoso (versão 7.0.0), Kettle (versão 4.3.0-stable) e Java Standard Edition (versão 1.7.0_13). Essas plataformas estavam instaladas localmente em um *notebook* com processador Intel Core i3-2370M de 2.4 GHz e 6 GB de memória principal, executando o sistema operacional Windows 7 na versão 64 bits. O número de triplas RDF publicadas por meio da *ETL4LinkedProv*, tanto de dados de domínio, quanto de metadados de proveniência, segue o modelo representado pela fórmula:

$$T = \sum_{i=1}^{n} (I_i * A_i + I_i * P_i) + \sum_{\substack{i=1 \\ j=1}}^{n} R_{ij}$$

onde, T é o número de triplas RDF publicadas, n é o número de entidades extraídas da fonte de dados pelo *workflow* ETL, I_i é o número de instâncias extraídas da entidade i, A_i é o número de tipos RDF relacionados à entidade i, P_i é o número de propriedades extraídas da entidade i e R_{ij} é o número de relacionamentos entre as instâncias da entidade i e as instâncias da entidade j.

A Tabela 1 apresenta o resultado da primeira análise quantitativa, que consistiu em comparar os tempos de execução dos *sub-workflows* ETL de publicação das fontes de

dados do CNPq e da RNP em LD, considerando as 3 configurações distintas do nível de granulosidade da proveniência. t_1 é o tempo de execução sem atuação do ACP e t_2 e t_3, os tempos de execução com atuação do ACP, respectivamente, com nenhum e com todos os tipos de passos selecionados para captura da proveniência com granulosidade fina.

Tabela 1. Tempos de execução dos *sub-workflows* ETL de publicação das fontes do CNPq e da RNP em Linked Data.

Extração						Carga	Nível 1	Nível 2	Nível 3
Fonte	**Entidade**	**I**	**A**	**P**	**∑ *R***	**T**	**t_1 mm:ss**	**t_2 mm:ss**	**t_3 mm:ss**
CNPq - Lattes	Pesquisador	7	1	5	478	2242	00:02	00:03	06:10
	Produção	217	1	5					
	Projeto	42	1	9					
CNPq - Grupos de Pesquisa	Grupo de Pesquisa	3	1	2	158	425	00:01	00:02	02:10
	Pesquisador	69	1	2					
	Coordenador	4	1	0					
	Aluno	47	1	0					
RNP - Grupos de Trabalho	Grupo de Trabalho	63	1	6	652	2267	00:02	00:04	09:50
	Instituição	18	1	2					
	Pesquisador	362	1	2					
	Coordenador	34	1	0					
Total		866			1288	4934	00:05	00:09	18:10

Usualmente os *sub-workflows* ETL seriam executados em uma infraestrutura de maior desempenho e iriam extrair um volume maior de dados, no entanto, o impacto da atuação do ACP nos tempos de execução do *workflow* ETL encapsulado estaria igualmente relacionado à configuração do nível de granulosidade da proveniência coletada. Sem o nível de granulosidade fina habilitado, ou seja, somente com coleta de metadados de proveniência mais genéricos sobre o processo de publicação, a duração (t_2) do *workflow* ETL e de seus *sub-workflows* é muito próxima do tempo de execução sem a atuação do ACP (t_1). Em contrapartida, o tempo de execução cresce exponencialmente de acordo com a quantidade de passos configurados para ter a proveniência coletada no nível de cada dado lido e manipulado pelos passos. Como normalmente ocorrem repetições de dados e existência de dados pouco relevantes, ao se aplicar a *ETL4LinkedProv*, torna-se recomendável realizar uma análise prévia sobre qual grão da proveniência é mais adequado para ser configurado no ACP. Esta configuração deve atender aos requisitos de exploração conjunta entre os dados de domínio e seus metadados de proveniência e, ao mesmo tempo, não sobrecarregar desnecessariamente o tempo de execução do processo de publicação de LD.

Com relação ao volume de triplas RDF de proveniência e à duração do *sub-workflow* ETL que as publicou, a Tabela 2 apresenta os resultados obtidos, considerando as 2 últimas configurações do nível de granulosidade da proveniência aplicadas no ACP. O número de tipos RDF (A) relacionados às entidades do repositório temporário utilizado pelo ACP (Figura 1) e o respectivo número de propriedades (P) foram contabilizados para cada ontologia empregada para representar os metadados de proveniência no contexto de LD. As entidades Repositório, Nota e Ligação Unidirecional do modelo conceitual não são representadas por classes das ontologias de proveniência, mas seus atributos são utilizados como objetos de propriedades de outras entidades. Para cada nível de

granulosidade da proveniência, foram registrados o número de instâncias extraídas (I) do repositório temporário, o somatório de relacionamentos entre as instâncias ($\sum R$) e o tempo de execução do *sub-workflow* ETL de publicação da proveniência (t).

Tabela 2. Volume de triplas RDF de proveniência publicadas e duração do *sub-workflow* ETL de publicação.

		PROV-O		OPMW		Cogs		DC		Nível 2				Nível 3			
	Entidade	**A**	**P**	**A**	**P**	**A**	**P**	**A**	**P**	**I**	**$\sum R$**	**T**	**t_4 mm:ss**	**I**	**$\sum R$**	**T**	**t_5 mm:ss**
Composição	Workflow	1	2	1	4	0	0	0	4	4	480	1727	00:09	4	551396	1141388	20:13
	Passo	0	0	1	0	1	0	0	2	105				105			
	Parâmetro	0	0	2	0	1	0	0	1	0				1428			
	Campo de Dados	0	0	2	0	1	0	0	1	0				517			
Execução	Workflow	2	2	1	4	0	0	0	1	4				4			
	Passo	1	2	1	2	0	0	0	1	105				105			
	Parâmetro	1	1	1	1	0	0	0	0	0				831			
	Linha de Dados	2	0	0	0	1	0	0	0	0				31055			
	Campo de Dados	1	1	1	1	0	0	0	0	0				121119			
	Usuário	1	0	0	0	0	0	1	2	1				1			

De maneira semelhante ao resultado da primeira análise, o impacto do ACP no tempo de execução do *sub-workflow* ETL de publicação da proveniência, assim como no volume de triplas RDF publicadas, está diretamente ligado ao nível de granulosidade configurado. Este impacto se intensifica na análise dos metadados de proveniência retrospectiva sobre a execução do *workflow* ETL.

5. Conclusão

Metadados de proveniência têm sido reconhecidos como mecanismo fundamental no apoio à avaliação de qualidade e consistência de dados e, de forma geral, como complemento aos esforços de integração e interoperabilidade. Em especial, no contexto de LD, dados de diferentes origens sendo manipulados, transformados e interligados podem certamente se beneficiar da captura e posterior publicação de metadados de proveniência, para que possam ser explorados em conjunto com os dados correspondentes. Neste trabalho, ao apresentar um processo sistemático apoiado por um sistema de *workflow* para captura e disponibilização de proveniência na forma de LD, procuramos explorar, em especial, os diferentes níveis de detalhe desses descritores e suas implicações em termos de desempenho e número de triplas complementares geradas. O exemplo de aplicação apresentado evidencia que é possível tirar vantagem da flexibilidade de configurar o processo de captura, de maneira a adequar o volume e desempenho às necessidades de cada situação.

Como trabalhos futuros, apontamos novas estratégias de interligação entre dados e metadados de proveniência, explorando as possibilidades de reificação de triplas e subgrafos, assim como mecanismos de indexação que possam servir de base para o processo de recuperação das informações disponibilizadas. Além destes, pretende-se testar a distribuição do processamento das consultas, no caso do uso da proveniência de granulosidade fina, tirando vantagem do paralelismo e de maior capacidade de processamento.

Agradecimentos. Maria Luiza M. Campos possui financiamentos parciais do CNPq (308934/2012-1) e FAPERJ (E-26/110.492/2012).

Referências

Alexander, K. et al. (2009) "Describing Linked Datasets - On the Design and Usage of voiD, the 'Vocabulary of Interlinked Datasets'", In: Proc. of the 2nd LDOW.

Auer, S. et al. (2012) "Managing the Life-Cycle of Linked Data with the LOD2 Stack", In: Proc of the 11th ISWC 2012, Part II, p. 1–16. Springer.

Bizer, C. et al. (2009) "Linked Data - The Story So Far", In: International Journal on Semantic Web and Information Systems 5, 3, p. 1–22.

Breitman, K. et al. (2012) "Open Government Data in Brazil", In: IEEE Intelligent Systems 27, June, p. 45–49.

Campos, M.L.M. and Guizzardi, G. (2010) "GT-LinkedDataBR – Exposição, compartilhamento e conexão de recursos de dados abertos na Web (Linked Open Data)". www.rnp.br/pd/gts2010-2011/gt_linkeddatabr.html.

Casters, M. et al. (2010) "Pentaho Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration". Wiley Publishing Inc., USA.

Cordeiro, K. F. et al. (2011a) "Empowering Citizens and Government with Collaboration on Linked Open Data", In: ESWC/Workshop Semantics in Governance and Policy Modelling, p. 33–37.

Cordeiro, K. F. et al. (2011b) "An approach for managing and semantically enriching the publication of Linked Open Governmental Data", In: WCGE do SBBD2011, p. 82–95.

Cruz, S.M.S. et al. (2009) "Towards a Taxonomy of Provenance in Scientific Workflow Management Systems.", In: Congress on Services I, IEEE, p. 259–266.

Freitas, A. et al. (2012) "Representing Interoperable Provenance Descriptions for ETL Workflows", In: Proc. of the 3rd SWPM 2012.

Gil, Y. et al. (2010) "Provenance XG Final Report", W3C, www.w3.org/2005/Incubator/prov/XGR-prov.

Heath, T. and Bizer, C. (2011) "Linked Data: Evolving the Web into a Global Data Space". Morgan & Claypool.

Mendonça, R.R. et al. (2013) "LOP - Capturing and Linking Open Provenance on LOD Cycle", In: Proc. of the 5th SWIM, ACM.

Moreau, L. et al. (2011) "The Open Provenance Model: Core Specification (v1.1)", In: Future Generation Computer Systems 27, 6, p. 743–756.

Moreau, L. and Missier, P. (2013) "PROV-DM: The PROV Data Model", W3C, www.w3.org/TR/prov-dm/.

Omitola, T. et al. (2012) "Capturing interactive data transformation operations using provenance workflows", In: 3rd SWPM 2012.

Sheridan, J. and Tennison, J. (2010) "Linking UK Government Data", In: Proc. of the 3rd LDOW.

Halite$_{ds}$: Agrupamento de Dados em Subespaços de Séries Temporais Multidimensionais *

paper:74

Afonso E. da Silva[1], **Robson L. F. Cordeiro**[1]

[1]Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo (USP)
Avenida Trabalhador São-Carlense, 400 – Centro – São Carlos – SP – Brasil

aexpedito@grad.icmc.usp.br, robson@icmc.usp.br

Abstract. *Given a data stream with many attributes, how to cluster similar events? For example, how to cluster measurements of tens of climatic attributes to aid in forecasting the climate and extreme events? The task of clustering data with many attributes is known as* **subspace clustering***. Today, there exists a need for algorithms of this type well-suited to process data streams. This paper proposes the new algorithm* Halite$_{ds}$ *for subspace clustering in data streams. The new algorithm improves upon one existing technique, the method* Halite*, which was originally designed to process static datasets. Compared to using the base algorithm in data streams, the new algorithm takes advantage of the knowledge obtained from clustering past data to easy clustering data in the present, thus shrinking the runtime. Experiments using a synthetic stream, as well a real climatic stream indicate that the new algorithm is in average* 4.2 *times faster than the base algorithm, still obtaining similar accuracy of results.*

Resumo. *Dada uma série temporal com muitos atributos, como agrupar eventos similares? Por exemplo, como buscar grupos em medições de dezenas de atributos climáticos para previsão climática e de eventos extremos? O agrupamento de dados com muitos atributos é conhecido como* **agrupamento em subespaços***. Há hoje uma carência de algoritmos adequados a séries temporais. Este artigo propõe o novo algoritmo* Halite$_{ds}$ *para agrupamento em subespaços de séries temporais. É utilizada como base a técnica* Halite*, originalmente voltada à análise de dados estáticos. Em comparação ao uso do algoritmo base em séries temporais, o novo algoritmo permite que o conhecimento obtido dos dados do passado facilite o agrupamento dos dados no presente, diminuindo o tempo de análise. Experimentos em uma série sintética e em uma série climática real indicam que o novo algoritmo é em média* $4,2$ *vezes mais rápido do que o algoritmo base, e ainda obtém acurácia similar de resultados.*

1. Introdução

O volume de informação gerada ou coletada em formatos digitais nas mais diversas áreas de aplicação vem crescendo não somente na quantidade de objetos, mas também na quantidade e na complexidade dos atributos que descrevem cada objeto [Kang et al. 2014, Cordeiro et al. 2013a, Cordeiro et al. 2011]. Além disso, a coleta de dados é em muitos casos um processo contínuo, repetitivo e potencialmente infinito, no qual são registrados os valores dos atributos de interesse seguindo intervalos de tempo determinados. O conjunto de valores resultante forma uma **série temporal multidimensional**[1]. Nesse cenário, a busca por agrupamentos é um dos principais meios que nos auxiliam nas tarefas de análise, compreensão e extração de conhecimento de tais dados. Por exemplo, como analisar décadas de medições freqüentes de dezenas de atributos climáticos, como temperatura, precipitação de chuva,entre outros, a fim de identificar grupos de medições similares voltado ao auxílio à previsão climática e de eventos extremos?

*Este trabalho foi financiado por FAPESP, CAPES e CNPq.

[1]Assume-se aqui que uma série temporal multidimensional é formada por n séries temporais (unidimensionais), em que n é o número de atributos ou dimensões, cujos valores são coletados simultaneamente em todas as séries.

Nota-se nesse cenário que, sempre que um período de tempo finito para análise é especificado, qualquer série temporal pode ser vista como uma base de dados estática. Assim, é possível a análise de séries temporais feita por algoritmos tradicionais, originalmente voltados à mineração de dados estáticos, em que são analisados dados correspondentes a janelas (i.e., intervalos) de tempo passado específicas, de forma que a análise de janelas consecutivas permita entender a evolução temporal dos dados. Entretanto, uma limitação relevante desta abordagem é o fato de que os resultados da análise de cada janela são comumente ignorados durante a análise de janelas temporais subsequentes, de forma que o conhecimento prévio obtido dos dados no passado não é utilizado para facilitar a análise dos dados no presente, aumentando assim o custo computacional da análise e impossibilitando e/ou dificultando, por exemplo, a análise das séries em tempo real.

Como conseqüência, têm sido desenvolvidos novos algoritmos de mineração específicos para a análise de séries temporais com o intuito principal de minimizar e/ou eliminar as limitações encontradas na abordagem anteriormente descrita. Em específico, para a tarefa de agrupamento de dados com muitos atributos – conhecida na literatura como **agrupamento em subespaços** [Kriegel et al. 2009], nota-se hoje uma carência de algoritmos voltados à análise de séries temporais multidimensionais, visto que são raros os trabalhos que abordam este tema na literatura.

Este artigo propõe um novo algoritmo de agrupamento de dados em subespaços bem adequado à análise de séries temporais de média a alta dimensionalidade – o algoritmo *Halite*$_{ds}$. É utilizada como base a técnica *Halite* [Cordeiro et al. 2013a, Cordeiro et al. 2013b], a qual integra o estado da arte para agrupamento em subespaços de dados estáticos de média a alta dimensionalidade por permitir a análise precisa, rápida e escalável de tais dados. O algoritmo *Halite* analisa por multiresolução os dados de entrada identificando variações na densidade dos dados representados em um espaço multidimensional por meio de varreduras em uma estrutura de dados que armazena contagens de pontos incidentes em hipercubos multidimensionais de variados tamanhos – uma *quad-tree* multidimensional. Nesse contexto, a principal contribuição do novo algoritmo *Halite*$_{ds}$ é permitir a análise de janelas consecutivas de séries temporais, sem reconstruir, para cada nova janela, a estrutura de dados utilizada. Permite-se assim que o conhecimento prévio obtido dos dados agrupados no passado facilite o agrupamento dos dados no presente, diminuindo consideravelmente o custo computacional total da análise.

São reportados experimentos realizados em uma série temporal sintética, e em uma grande série de dados climáticos reais referente a quase um século de medições diárias de atributos climáticos. Para analisar tais séries, o novo algoritmo *Halite*$_{ds}$ demandou tempo de processamento em média $4,2$ vezes menor do que o tempo requerido pelo o algoritmo base *Halite*, e, ainda assim, obteve acurácia de resultados similar.

O restante deste artigo segue uma organização tradicional: os principais trabalhos relacionados são discutidos na Seção 2; o novo método *Halite*$_{ds}$ é descrito na Seção 3, e; experimentos são reportados na Seção 4. Por fim, a Seção 5 apresenta conclusões.

2. Trabalhos Relacionados

Nota-se hoje uma carência de algoritmos de **agrupamento em subespaços** voltados à análise de séries temporais multidimensionais, visto que são raros os trabalhos que abordam este tema na literatura. Em [Hassani et al. 2013], é proposta a medida *SubCMM* (*Subspace Cluster Mapping Measure*) para a avaliação de acurácia de resultados, a qual detecta erros ocasionados por surgimento, movimentação e divisão do subespaço de cada grupo da série analisada. Em [Miller and Hu 2012], é proposta a técnica de agrupamento em subespaços *StreamPreDeCon* com foco específico na análise de séries de pacotes de dados em redes de comunicação para detecção de pacotes anômalos. Em [Zhang et al. 2007], é proposto um algoritmo incremental de agrupamento em subespaços sobre janelas deslizantes, o qual inclui um modelo para descrever grupos e detectar mudanças de

subgrupos em séries temporais em relação a subconjuntos específicos de eventos. Por fim, em [Lee and Lee 2008, Park and Lee 2007] é proposto um algoritmo de mineração em subespaços que combina o agrupamento por *grids* com a mineração de conjuntos freqüêntes em séries temporais.

Nota-se ainda a existência de um trabalho correlato não referente a agrupamento de dados. Em [Sousa et al. 2006] utiliza-se uma *quad-tree* multidimensional para analisar séries temporais, em que também são propostas operações de auto-ajuste da estrutura. Entretanto, o foco do trabalho é a análise fractal e de dimensão intrínseca dos dados.

2.1. O Algoritmo *Halite*

O algoritmo *Halite* [Cordeiro et al. 2013b] permite a busca precisa, rápida e escalável por agrupamentos em subespaços de dados estáticos de média a alta dimensionalidade. O algoritmo possui duas fases principais: a primeira é responsável por representar os objetos de entrada em uma estrutura de dados de tipo *quad-tree* multidimensional; na segunda etapa, a estrutura criada é analisada em busca de grupos de objetos similares formados em subespaços do espaço original.

É importante notar que as mudanças propostas neste artigo para permitir a análise eficiente de séries temporais afetam apenas a primeira fase do algoritmo base *Halite*. Por esse motivo, apresenta-se aqui uma descrição detalhada da construção da estrutura de dados utilizada – primeira fase – e uma descrição sucinta da fase subsequente que se mantém inalterada – a busca por grupos. Detalhes são encontrados em [Cordeiro et al. 2013b].

2.1.1. Primeira fase

A fase inicial do algoritmo envolve a construção em memória primária de uma estrutura de dados de tipo *quad-tree* multidimensional, referenciada como a *Árvore de Contagens*. A estrutura representa uma base de dados ${}^dS$ contendo $n = |\ {}^dS\ |$ objetos com d atributos cada. Cada atributo refere-se a um valor real normalizado para o intervalo $[0, 1)$. Portanto, assume-se que a base de dados de entrada está inclusa no hipercubo unitário $[0, 1)^d$.

O número H de níveis (ou resoluções) da árvore é dado de acordo com a necessidade da aplicação. O valor sugerido pelos autores é $H = 4$. O nó raiz (nível 0) representa a base de dados como um todo, o qual é subdividido em 2^d hipercubos no próximo nível (nível 1), dividindo-se ao meio o espaço de dados em relação a cada uma das d dimensões de forma que cada novo hipercubo tenha lados com a metade do tamanho dos lados do hipercubo original. Novamente, o nível 2 apresenta 2^d hipercubos cada, tendo lados com a metade do tamanho dos lados dos hipercubos do nível anterior. Segue-se a divisão até o último nível $H - 1$. Cada hipercubo é representado como uma célula de contagem. A estrutura da célula contém os campos loc, n, $P[d]$, $usedCell$ e ptr. O campo loc é um identificador único para uma célula, em relação a todas as células que compartilham a mesma "célula pai" no nível anterior. n é o número total de pontos incidentes na célula, e $P[d]$ é um vetor que armazena a contagem de pontos na metade inferior da célula, em relação a cada uma das d dimensões. Por fim, $usedCell$ é uma variável booleana, utilizada apenas na fase de agrupamento, e ptr é o ponteiro para o próximo nível da árvore.

Os objetos de dados são representados por pontos no espaço d-dimensional, sendo contabilizados nas respectivas células em que incidem em cada um dos níveis da árvore. Uma célula a em um nível h é indicada por a_h. Para exemplificar a divisão do espaço, a Figura 1 ilustra o espaço 2-dimensional dividido em células de diferentes níveis da árvore, além de seus respectivos valores para o campo loc. A célula em destaque – com textura em linhas diagonais – está no nível 3, e contém campo $loc = 11$. Nos níveis 2 e 1, encontram-se sua "célula mãe" com campo $loc = 11$ e sua "célula avó" com campo $loc = 01$, respectivamente. Portanto, um ponto incidente sobre a célula em destaque $a_3.loc = 11$ é também contabilizado nas células correspondentes em todos os demais níveis, nesse

caso, nas células identificadas por $a_1.loc = 01$ e $a_2.loc = 11$.

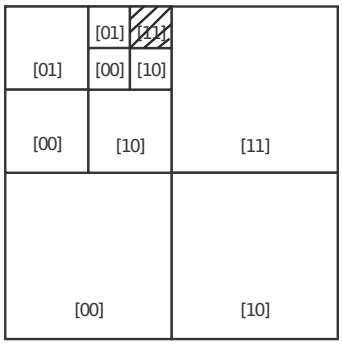


Figura 1. Células no espaço 2-dimensional.

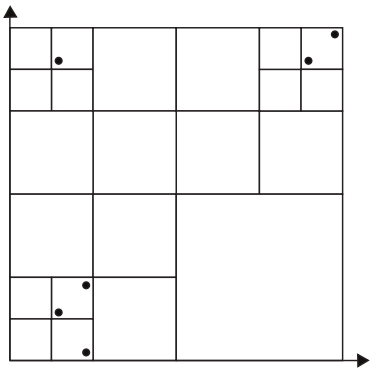


Figura 2. Pontos incidentes em células no espaço 2-dimensional.

A contagem de pontos é ilustrada nas Figuras 2 e 3. Seis pontos são apresentados no espaço 2-dimensional da Figura 2. A *Árvore de Contagens* correspondente é ilustrada na Figura 3, em que omite-se o campo $usedCell$ de cada célula para não poluir a ilustração. Têm-se células representadas em nós de três níveis distintos da árvore, em que cada nó apresenta $2^d = 4$ células.

2.1.2. Segunda fase

A segunda fase do algoritmo *Halite* busca por grupos de objetos similares em subespaços do espaço original dos dados de entrada. Apenas as contagens presentes na *Árvore de Contagens* são avaliadas nesta fase – não há novas leituras aos objetos de dados de entrada. O algoritmo encontra grupos com base na variação de densidade dos dados em subespaços. Uma análise por convolução é aplicada a cada nível da árvore, por meio de filtros de Laplace, a fim de localizar concentrações expressivas de pontos no espaço com todas as dimensões, as quais podem indicar grupos de dados existentes em subespaços do espaço original. Por ainda não terem sido validados, estes grupos em potencial são chamados de β-grupos. Em seguida, são analisados em separado os pontos de cada β-grupo, projetados em dimensões individuais, a fim de determinar os subespaços onde a concentração de pontos é mais evidente. Um teste estatístico é então aplicado para validar cada β-grupo, em relação ao seu subespaço de maior evidência, e os β-grupos com maior possibilidade de terem sido formados por tendências reais de agrupamento de dados são identificados e reportados.

3. Método proposto

Nesta seção é descrito o novo método *Halite*$_{ds}$ de agrupamento em subespaços de séries temporais multidimensionais. O método proposto adiciona três novas funcionalidades à *Árvore de Contagens* do algoritmo base, as quais são descritas nas subseções seguintes.

3.1. Inclusão de janelas de eventos

O algoritmo base *Halite* obtém os objetos de um banco de dados estático, de forma que é necessário ler cada objeto e inseri-lo na *Árvore de Contagens* sequencialmente, e só então o processo de agrupamento pode ser iniciado. Entretanto, séries temporais são potencialmente infinitas. Para analisá-las, deve-se permitir o agrupamento de subconjuntos dos dados, ao invés de aguardar a leitura do conjunto completo. Assume-se neste artigo que a série de dados de entrada é composta por uma seqüência de janelas de eventos, comumente referentes a intervalos temporais, como, por exemplo, janelas mensais, semestrais ou anuais, e que a memória primária disponível permite o armazenamento de M janelas. O novo algoritmo *Halite*$_{ds}$ lê e insere na *Árvore de Contagens* as M janelas iniciais, e então

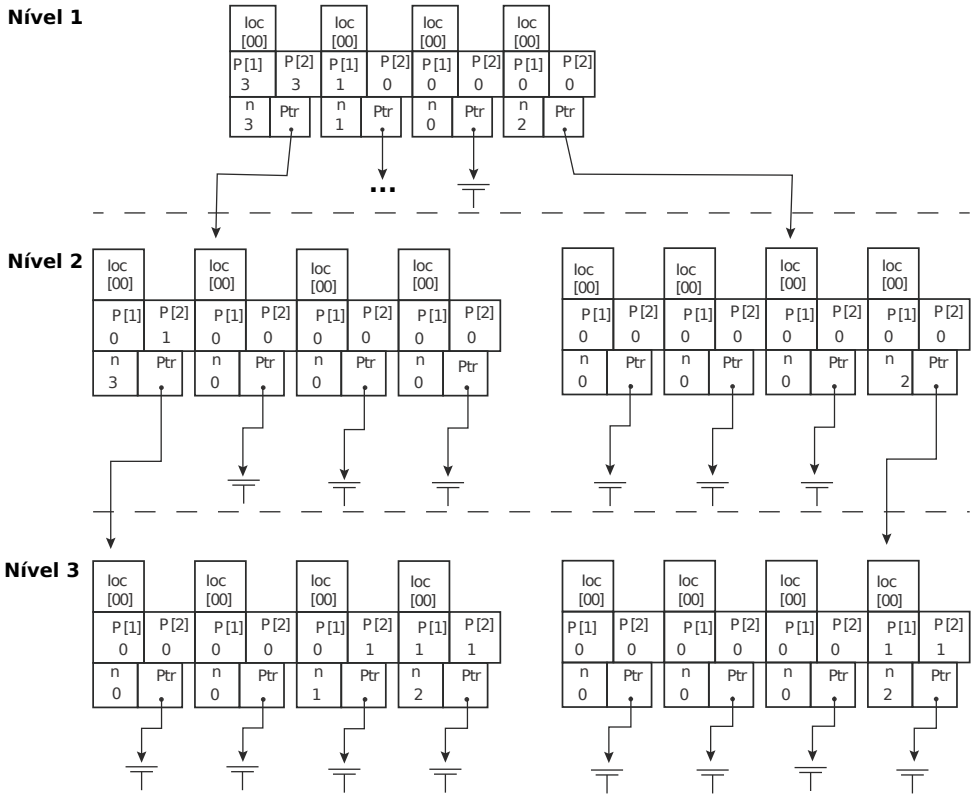


Figura 3. *Árvore de Contagens* referente aos pontos da Figura 1.

as analisa por agrupamento. Cada nova janela lida sobrescreve a janela mais antiga na árvore, e uma nova análise por agrupamento é realizada nas M janelas mais recentes. Por exemplo, se $M = 3$ janelas cabem em memória, o algoritmo lê e armazena a primeira janela, lê a segunda janela e a armazena, lê a terceira e a armazena, e só então busca por grupos nas três janelas lidas. Ao ler a quarta janela, os dados da primeira janela são sobrescritos e um novo processo de agrupamento é iniciado, considerando apenas a segunda, terceira e quarta janelas. Efetuam-se nesse processo análises por agrupamento conforme novas janelas são lidas, sempre em referência às M janelas mais recentes da série de dados. Permite-se assim a análise eficiente da evolução temporal dos dados.

O modelo de janelas empregado permite o suporte eficiente a aplicações que recebem eventos intermitentemente para análise sem a necessidade de realizar operações relativamente caras, originalmente propostas para a análise de bases de dados estáticas. Especificamente, o novo algoritmo *Halite*$_{ds}$ permite analisar as M janelas mais recentes de uma série temporal sem a necessidade de reconstruir por completo a *Árvore de Contagens* a cada nova janela lida. Vale notar que experimentos reportados na Seção 4 indicam que a construção da arvore é a fase com maior custo computacional do algoritmo base.

São propostas neste artigo alterações na árvore descrita na Subseção 2.1.1, algumas delas referentes aos campos de suas células. O campo n, que armazena a contagem de pontos incidentes em uma célula, passa a considerar também a janela temporal a que esses pontos pertencem. Na implementação atual do novo algoritmo *Halite*$_{ds}$ esse campo é representado por um vetor com posições referentes às contagens de pontos de janelas específicas. Adiciona-se também mais uma dimensão ao campo $P[i]$ para representar janelas, o qual armazena a contagem de pontos na metade inferior de cada célula em relação a cada atributo i. A implementação atual desse campo é feita por uma matriz 2-dimensional $P[i][j]$, em que i representa um atributo e j uma janela.

O Algoritmo 1 descreve a operação de inserção de uma janela na *Árvore de Contagens*. As

operações de expansão e contração são descritas nas seções posteriores.

Algoritmo 1: insereJanela($eventosDaJanela, idDaJanela$)

Entrada: Eventos da nova janela e número identificador da nova janela
Saída: Árvore de contagem com a nova janela

```
1  início
2      Descarta dados da janela mais antiga na árvore;
3      para cada evento e da janela faça
4          se e está fora do hipercubo representado por raiz então
5              Expande(e); // expande a árvore para representar o evento e
6          fim
7          Insere o evento e na árvore, com referência à janela atual;
8      fim
9      se há uma única célula no nível 1 da árvore então
10         Contrai( ); // contrai a árvore
11     fim
12 fim
```

3.2. Análise de dados não normalizados

O algoritmo base *Halite* recebe objetos de dados com valores de atributos normalizados entre 0 e 1, e os insere na árvore com seu nó raiz sempre representando o hipercubo unitário $[0, 1)^d$. Os limites espaciais mínimo e máximo de cada célula da árvore são sempre invariáveis e pré-definidos, além de serem sempre os mesmos para todas as dimensões. No entanto, em diversas séries temporais reais os valores limites mínimo e máximo de cada atributo não são previamente conhecidos, inviabilizando qualquer normalização. É então necessário capturar e representar na árvore eventos não normalizados, assumindo que seus atributos podem conter qualquer valor real.

Para permitir a análise de séries temporais é proposta neste artigo mais uma alteração na árvore descrita na Subseção 2.1.1. No novo algoritmo *Halite*$_{ds}$ o nó raiz da árvore representa um hipercubo com tamanho variável $r0$ para cada lado. O valor inicial de $r0$ é definido pelo domínio ativo dos atributos da primeira janela de dados lida. Especificamente, $r0 = max(maior_i - menor_i)$, em que $maior_i$ e $menor_i$ se referem respectivamente ao maior e ao menor valor do domínio ativo do atributo i em relação à primeira janela. A primeira janela também define valores iniciais para os atributos L_i e U_i, os quais se referem respectivamente aos valores limites mínimo e máximo do espaço de dados representado pelo nó raiz da árvore para cada atributo i. A inicialização de L_i é dada por $L_i = menor_i$ e, após isso, U_i é inicializado com $U_i = L_i + r0$. Note que os limites do espaço de representação da árvore são variáveis e dinâmicos, e, portanto, assim também são os valores de L_i e U_i para cada atributo i e de $r0$, os quais são sempre armazenados junto à árvore para correta interpretação do espaço de dados representado.

3.3. Expansão e contração do espaço de representação

Dado que atributos de séries temporais podem assumir quaisquer valores dos números reais, propõe-se para o algoritmo *Halite*$_{ds}$ o auto-ajuste do espaço de representação da *Árvore de Contagens*, por meio de duas novas operações de expansão e contração de espaço. Em uma expansão, um novo nó raiz é criado na árvore para representar um espaço maior, enquanto o antigo raiz se torna "filho" do novo nó. De modo contrário, em uma contração o nó raiz é excluído por ter apenas um nó "filho", enquanto o "filho" assume o papel de nova raiz para representar um espaço menor. Permite-se assim inserir eventos que não incidem sobre o espaço até então representado.

No novo algoritmo *Halite*$_{ds}$, a primeira janela de eventos determina o intervalo inicial que o nó raiz da árvore irá adotar para cada dimensão. Na chegada de um evento e de uma nova janela,

Algoritmo 2: Expande(e)

Entrada: Coordenadas e_i de um evento e **Saída:** Árvore expandida

```
1  início
2      se e está fora do hipercubo representado por raiz então
3          // incrementa o nível inicial de busca por grupos (para a segunda fase)
4          nivelInicial = nivelInicial + 1;
5          H = H + 1; // incrementa a altura da árvore
6          para cada coordenada e_i do evento e faça
7              A_i = L_i; // valor antigo de L_i para uso posterior
8              se e_i < L_i então
9                  L_i = L_i − r0;
10             senão
11                 U_i = U_i + r0;
12             fim
13         fim
14         r0 = r0 * 2;  aux = raiz;  raiz = new(nó);  raiz.ptr = aux;
15         para cada dimensão i faça
16             para cada janela j faça
17                 se e_i < A_i então
18                     raiz.P[i][j] = aux.n[j];
19                 senão
20                     raiz.P[i][j] = 0;
21                 fim
22             fim
23         fim
24         Expande(e); // chamada recursiva para uma possível nova expansão
25     fim
26 fim
```

se e contiver um valor de atributo e_i tal que $e_i < L_i$ ou $e_i > U_i$, em relação a pelo menos um atributo i, é realizada uma operação de expansão na árvore para possibilitar sua representação, e só então, o evento é inserido. Como pode ser visto no Algoritmo 2, a operação de expansão é realizada recursivamente até que todas as coordenadas do evento possam ser representadas. Por outro lado, a contração (Algoritmo 3) realiza o procedimento contrário à expansão para a eliminar da memória todas as células vazias da árvore, causadas por remoção de uma janela antiga. Exemplos de expansão e de contração são ilustrados nas Figuras 4 e 5 respectivamente.

4. Experimentos

Esta seção reporta os experimentos realizados com o novo método *Halite*$_{ds}$. Em comparação ao uso do método base *Halite* em séries temporais, visa-se responder:

- **Q1** – Qual é o ganho em tempo de processamento obtido pelo novo método?
- **Q2** – Qual é o correspondente impacto na acurácia dos resultados obtidos?

Os experimentos utilizaram um processador de $2,0$ GHz de uma máquina com SO Linux e 2 GB de memória principal. Ambas as técnicas *Halite* e *Halite*$_{ds}$ foram configuradas com os valores padrão para seus parâmetros de entrada, $H = 4$ e $\alpha = 1.0E-10$, seguindo o que sugere a proposta original de *Halite*. Note que α é utilizado apenas na busca por grupos – segunda fase – que se mantém inalterada neste artigo.

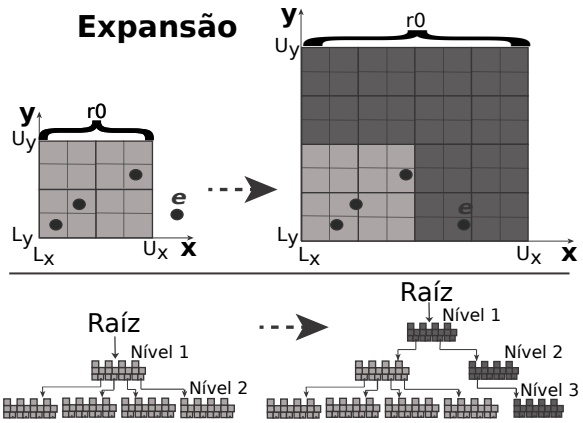


Figura 4. Exemplo de expansão.

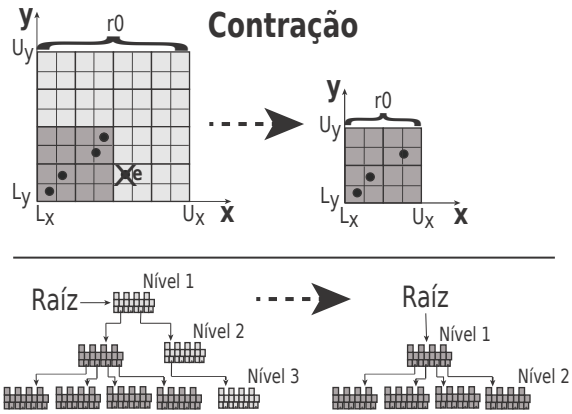


Figura 5. Exemplo de contração.

Algoritmo 3: Contrai( )

Saída: Árvore contraída

início
  se *há uma única célula no nível* 1 *da árvore* $\wedge$
    $H >$ *valor de* H *originalmente informado pelo usuário* **então**
      // decrementa o nível inicial de busca por grupos (para a segunda fase)
      $nivelInicial = nivelInicial - 1$;
      $H = H - 1$; // decrementa a altura da árvore
      $c =$ célula única do nível 1 da árvore; $r0 = r0/2$;
      para *cada dimensão* i **faça**
        se c *está na metade superior de* $raiz$*, em relação a* i **então**
          $L_i = L_i + r0$;
        senão
          $U_i = U_i - r0$;
        fim
      fim
      para *cada dimensão* i **faça**
        para *cada janela* j **faça**
          $raiz.P[i][j] = c.P[i][j]$;
        fim
      fim
      $aux = raiz$; $raiz = c.ptr$; $delete(aux)$;
  fim
  $Contrai(\,)$; // chamada recursiva para uma possível nova contração
fim

4.1. Experimentos com dados sintéticos

Foi gerada uma série temporal multidimensional sintética de 15 atributos (i.e., 15 séries unidimensionais com eventos simultâneos) dividida em 100 janelas, cada janela contendo $\sim 10k$ eventos. Cada janela inclui 5% de *outliers* e 10 grupos existentes apenas em subespaços do espaço original de 15 dimensões, com subespaços escolhidos aleatóriamente. Cada grupo segue uma distribuição normal em seu correspondente subespaço, definida com média e desvio padrão aleatórios. Visto que as janelas devem representar evoluções temporais de uma mesma base de dados, cada grupo da primeira janela tem seu par correspondente em cada uma das janelas posteriores, seguindo a mesma média e

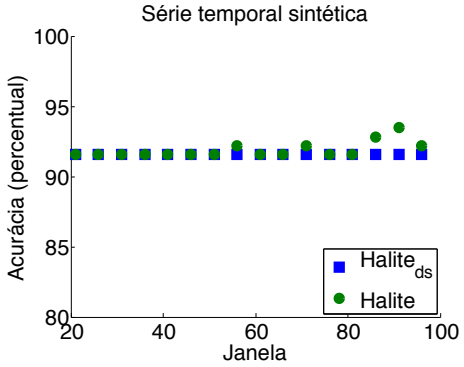


Figura 6. Acurácia em série sintética.

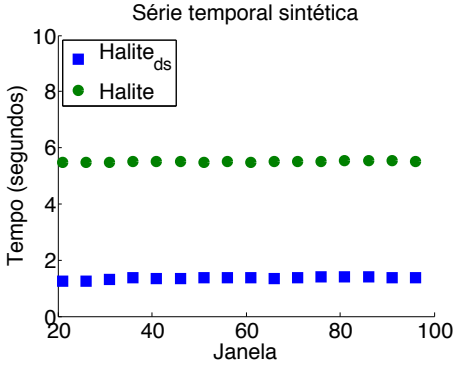


Figura 7. Tempo em série sintética.

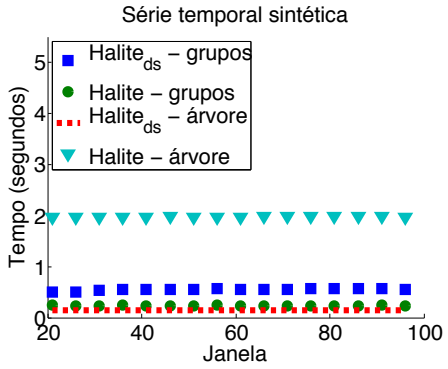


Figura 8. Tempo em série sintética – árvore e grupos.

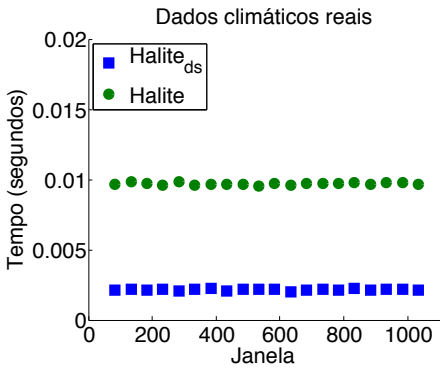


Figura 9. Tempo em série climática real.

desvio padrão na geração de dados. Utilizou-se aqui o algoritmo de geração de dados do algoritmo base, o Algoritmo 6 em [Cordeiro et al. 2013b].

Os resultados em dados sintéticos são vistos nas Figuras 6, 7 e 8. $M = 20$ foi utilizado. A Figura 6 reporta a acurácia (percentual) dos resultados em relação à cada janela lida. Como pode ser visto, a acurácia se mantém similar em todos os casos para ambos os algoritmos. Para computar a acurácia, utilizou-se a mesma estratégia baseada em valores de precisão e revocação definida na proposta original do algoritmo base, Seção 8.1 em [Cordeiro et al. 2013b]. Nesse processo, valores de precisão e de revocação foram computados comparando-se os resultados de agrupamento apresentados por cada algoritmo com o resultado ideal (i.e., "verdade absoluta" ou *ground truth*) previamente conhecido apenas nos dados sintéticos.

A Figura 7 reporta os respectivos tempos de processamento. Como pode ser visto, o novo método *Halite*$_{ds}$ é em média 4 vezes mais eficiente do que o método base *Halite*. Esses resultados são detalhados na Figura 8 para mostrar o tempo gasto em duas fases específicas: a construção da árvore e a busca por grupos. Como pode ser visto, o tempo de construção da árvore do algoritmo base *Halite* equivale em média a $89,3\%$ do tempo total da análise, sendo este o grande gargalo no processamento. Justificam-se assim os ganhos expressivos da nova abordagem proposta.

4.2. Experimentos com dados reais

Uma grande série climática real multidimensional foi estudada contendo quase um século (i.e., anos entre 1917 e 2010) de medições diárias de três de atributos climáticos, temperaturas máxima e mínima e precipitação de chuva, efetuadas em uma estação climática real – a estação da ESALQ-USP localizada no município de Piracicaba[2]. A série temporal completa contém 33.217 eventos. Os resultados obtidos são reportados na Figura 9. Janelas mensais e $M = 60$ foram utilizados. É reportado o tempo

[2] Dados gentilmente fornecidos por colaboradores do CEPAGRI-UNICAMP e CPTEC-INPE.

de processamento para cada algoritmo e cada janela. Como pode ser visto, o novo método *Halite*$_{ds}$ é novamente mais eficiente do que o método base *Halite*, sendo neste caso $4,5$ vezes mais rápido.

5. Conclusão

Este artigo propõe um novo algoritmo de agrupamento de dados em subespaços de séries temporais de média a alta dimensionalidade – o algoritmo *Halite*$_{ds}$. É utilizada como base a técnica *Halite*. A principal contribuição do novo algoritmo *Halite*$_{ds}$ é permitir a análise de janelas consecutivas de séries temporais, sem reconstruir, para cada nova janela, a estrutura de dados utilizada. Permite-se assim que o conhecimento prévio obtido dos dados agrupados no passado facilite o agrupamento dos dados no presente, diminuindo consideravelmente o custo computacional total da análise. São reportados resultados de experimentos realizados uma série temporais sintética, e também em uma grande série de dados climáticos reais referente quase um século de medições diárias de atributos climáticos. Para analisar tais séries, o novo algoritmo *Halite*$_{ds}$ foi em média $4,2$ vezes mais rápido do que o algoritmo base *Halite*, e, ainda assim, obteve acurácia de resultados similar.

Referências

Cordeiro, R., Faloutsos, C., and Traina Jr., C. (2013a). *Data Mining in Large Sets of Complex Data*. SpringerBriefs in Computer Science Series. Springer-Verlag New York Incorporated.

Cordeiro, R. L. F., Jr., C. T., Traina, A. J. M., López, J., Kang, U., and Faloutsos, C. (2011). Clustering very large multi-dimensional datasets with mapreduce. In Apté, C., Ghosh, J., and Smyth, P., editors, *KDD*, pages 690–698. ACM.

Cordeiro, R. L. F., Traina, A. J. M., Faloutsos, C., and Traina Jr., C. (2013b). Halite: Fast and scalable multiresolution local-correlation clustering. *IEEE TKDE*, 25(2):387–401.

Hassani, M., Kim, Y., Choi, S., and Seidl, T. (2013). Effective evaluation measures for subspace clustering of data streams. In Li, J., Cao, L., Wang, C., Tan, K., Liu, B., Pei, J., and Tseng, V., editors, *Trends and Applications in Knowledge Discovery and Data Mining*, volume 7867 of *Lecture Notes in Computer Science*, pages 342–353. Springer Berlin Heidelberg.

Kang, U., Meeder, B., Papalexakis, E. E., and Faloutsos, C. (2014). Heigen: Spectral analysis for billion-scale graphs. *IEEE Trans. Knowl. Data Eng.*, 26(2):350–362.

Kriegel, H.-P., Kröger, P., and Zimek, A. (2009). Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM TKDD*, 3(1):1:58.

Lee, J. W. and Lee, W. S. (2008). A coarse-grain grid-based subspace clustering method for online multi-dimensional data streams. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 1521–1522, New York, NY, USA. ACM.

Miller, Z. and Hu, W. (2012). Data stream subspace clustering for anomalous network packet detection. *J. Information Security*, 3(3):215–223.

Park, N. H. and Lee, W. S. (2007). Grid-based subspace clustering over data streams. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 801–810, New York, NY, USA. ACM.

Sousa, E. P., Traina, A. J., Traina Jr., C., and Faloutsos, C. (2006). Measuring evolving data streams' behavior through their intrinsic dimension. *New Generation Computing*, 25(1):33–60.

Zhang, Q., Liu, J., and Wang, W. (2007). Incremental subspace clustering over multiple data streams. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 727–732.

paper:75

Improving Pairwise Preference Mining Algorithms using Preference Degrees

Juliete A. Ramos Costa[1], **Sandra de Amo**[1]

[1]Faculdade de Computação
Universidade Federal de Uberlândia
Uberlândia-MG, Brazil

juliete@mestrado.ufu.br,deamo@ufu.br

Abstract. *Different preference mining techniques designed to predict a preference order on objects have been proposed in the literature, with very good accuracy results. In this paper, we propose to consider not only the fact that the user prefer an item i_1 to an item i_2 but also the degree of his preference on the two items. We propose the algorithm FuzzyPrefMiner designed to predict fuzzy preferences and show through a series of experiments that it outperforms pairwise preference mining techniques whose training phase do not include information on preference degrees.*

1. Introduction

Most known recommendation systems, following a collaborative filtering or a content-based approach, rely on explicit user preference feedback data in order to provide recommendations on items not yet seen by the user. These explicit preference data are given by the *utility matrix* which contains at each position i, j the rating given by user i to item j. However this explicit way to get user feedback lacks of effectiveness, since most users are unwilling to rate items while browsing the web. Moreover, the information obtained in that way may be biased, since it does not include preferences from users who are not keen on rating things. Another drawback of using explicit user feedback is related to the fact that the rating scale is fixed (normally going from 0 to N, where N is the maximum rating permitted). Let us suppose that a user u rated a movie f_1 as N since he/she loved it. Sometime later u saw the movie f_2 which he/she liked more than f_1. In this case, since past given ratings cannot be modified, u is forced to give equal ratings (namely N) to both movies, even if f_2 is largely preferred to f_1.

For those reasons, we argue that preference data obtained in an implicit manner (by inferring user preferences from his/her browsing behavior) and following a *pairwise* representation (the user preference is represented by a set of pairs of items (i_1, i_2) telling that i_1 is preferred to i_2) are more desirable. For instance, let us suppose that, while browsing the IMDB website[1], user u spent 30 seconds on the page of movie f_1 and 10 minutes on the page of f_2. One can infer that u is far more interested on film f_2 than on film f_1. One can go a step further and assume that u prefers f_2 to f_1 and represent this preference as a pair (f_2, f_1). Pairwise preference data obtained in this implicit way do not contain sufficient information for inferring *specific ratings* on items. However, they are sufficient for inferring a *ranking* of the top-k films the user would probably like, and consequently, recommend those films to him/her [Cohen et al. 1999].

[1]www.imdb.com

Most preference mining methods proposed in the literature rely on explicit ratings provided by the user as input data represented by means of a *score function* [Cohen et al. 1999, Crammer and Singer 2001, Burges et al. 2005]. *Pairwise* representation [Jiang et al. 2008, Koriche and Zanuttini 2010, de Amo et al. 2013, Amo et al. 2012] and *Ranking* representation [2] [Joachims 2002,Freund et al. 2003] are receiving more attention in recent research. Preference mining techniques that use input data following a pairwise representation usually obtain this input data directly from the user, using a pre-processing step for transforming explicit user rating into pairs[3]. For instance, if the user evaluated item i_1 as 5 and i_2 as 1, this information is transformed into the pair (i_1, i_2). We claim that in this transformation, a lot of information has been lost. Indeed, if the ratings were 5 and 4, the same pair (i_1, i_2) would be obtained. In the first situation, the user prefers object i_1 to i_2 more *intensely* than in the second situation.

In our previous work [de Amo et al. 2013] we proposed the algorithm CPrefMiner for mining a *crisp* contextual preference model from a set of preference samples represented as pairs of alternatives. The model is said to be *crisp* since the input data provided by the user is based on *yes/no* alternatives (either he/she prefers object i_1 to object i_2 or vice-versa) and the output model allows to predict if a new object i_3 is preferred to a new object i_4 or vice-versa. The mined preference model is said to be *contextual* since it is constituted by a set of *contextual preference rules* ([Wilson 2004]) of the form *IF <context> then I prefer 'this' to 'that'* . For instance: For films directed by Spielberg I prefer Action to Comedy whereas for films directed by Woody Allen I prefer Comedy to Action. The preference model extracted by CPrefMiner is capable to infer an *order relation on objects* (a task a classifier would not be able to achieve !) that is, it is capable to infer *transitivity* relatioships (if $i_1 \succ i_2$ and $i_2 \succ i_3$ then $i_1 \succ i_3$ [4]) and also satisfying *irreflexibility* ($i_1 \not\succ i_1$).

The main hypothesis of this paper is that *"the more information is enclosed in the input preference data, the more efficient is the mined preference model"*. We propose the method FuzzyPrefMiner to mine *fuzzy* contextual preference model from a set of preferences samples represented as triples (i_1, i_2, n) where i_1 and i_2 are items evaluated by a user u and n is the *degree of preference* ($0 \leq n \leq 1$) of i_1 w.r.t. i_2 provided by user u . An extensive set of experiments comparing CPrefMiner and FuzzyPrefMiner validates this hypothesis: FuzzyPrefMiner is far more accurate then CPrefMiner to predict user preferences. Besides, the preference knowledge extracted by FuzzyPrefMiner is more refined then the one provided by CPrefMiner, since it is capable to predict not only if $i_1 \succ i_2$ but also the degree of preference of i_1 with respect to i_2. For lack of space, we present in this paper only a small subset of the experiments carried out to validate our hypothesis.

Fuzzy preference modeling and reasoning has been extensively studied in the last decade [Chiclana et al. 1998, Herrera-Viedma et al. 2004, Ma et al. 2006, Xu et al. 2013]. However, to the best of our knowledge this kind of preference model has not yet been exploited in the preference mining research.

[2] The user provides a sequence of objets in decreasing order of preference.

[3] Notice that, as argued before, pairwise preference data could also be obtained indirectly by examining the user's browsing behavior.

[4] the symbol $\succ$ standing for "is preferred to"

2. Problem Formalization

In this section we formalize the problem of mining fuzzy contextual preferences, introducing the notion of *fuzzy preference model (FPM)*, how it is used to order things and some quality measures to evaluate its order predicting capability.

A *Fuzzy Preference Relation* over a set of objects $X = \{x_1, ..., x_n\}$ is a $n \times n$ matrix P such that $P_{ij} \in [0, 1]$ represents the *degree of preference (dp)* of u over objects x_i and x_j. Fuzzy preference relations have been introduced in [Chiclana et al. 1998] and should satisfy certain conditions (see [Chiclana et al. 1998] for details), the most important being the reciprocity property $P_{ij} = 1 - P_{ji}$ (which entails that $P_{ii} = 0.5$). One way of obtaining fuzzy preference relations from a set of item ratings provided by the user is to consider $P_{ij} = f(\frac{r_i}{r_j})$, where r_i is the rating u assigned to item x_i and f is a function belonging to the family $\{f_n : n \geq 1\}$, $f_n(x) = \frac{x^n}{x^n+1}$. This family of functions produces matrices satisfying all the properties required for fuzzy preference relations. Fig. 1(b) illustrates the fuzzy preference relation corresponding to the item-rating database of Fig. 1(a). Here, function f_2 has been considered for *fuzzification*.

A *Fuzzy Preference Database (FPD)* over a relation schema $R(A_1, ..., A_n)$ is a finite set $\mathcal{P} \subseteq$ Tup(R) $\times$ Tup(R) $\times$ [0,1] which is *consistent*, that is if $(i, j, n) \in \mathcal{P}$ and $(j, i, m) \in \mathcal{P}$ then $m = 1 - n$. The triple (i, j, n), represents the fact that the user prefers *the tuple i to the tuple j* with a degree of preference n. Fig. 1(c) illustrates a fuzzy preference database corresponding to a *subset* of the information contained in the matrix depicted in Fig. 1(b) (the triples presented in (c) corresponds to the underlined positions of (b)).

Pre-processing the FPD. Let $\mathcal{P}$ be a fuzzy preference database. First of all, triples with $dp < 0.5$ are transformed into triples with $dp \geq 0.5$. For instance, triple $(i_2, i_3, 0.31)$ in Fig. 1(c) becomes $(i_3, i_2, 0.69)$. After doing that, let dp_{min} (resp. dp_{max}) be the smaller (resp. the larger) dp appearing in (the transformed) $\mathcal{P}$.
Partition of $\mathcal{P}$ into k layers : First, let us partition the interval $I = [dp_{min}, dp_{max}]$ into k disjoints and contiguous subintervals $I_1, ..., I_k$, such that $\bigcup_{i=1}^{k} I_i = I$. Second, partition $\mathcal{P}$ into k disjoint subsets of triples $\mathcal{P}_1$,..., $\mathcal{P}_k$. Each $\mathcal{P}_i$ is obtained by inserting into $\mathcal{P}_i$ all triples (w_1, w_2, n) with $n \geq 0.5$ such that (w_1, w_2, n) or $(w_2, w_1, 1 - n) \in \mathcal{P}$. Figure 1(d) illustrates this step.
Layers can be viewed as classes: Let $I_i = [a_i, b_i]$ and $\lambda_i = \frac{a_i+b_i}{2}$. Triples (w_1, w_2, n) in I_i are characterized by tuples w_1 and w_2 having an average "intensity" of preference λ_i between them. For instance, triples of layer T_1 in 1(d) are associated to class λ_1 = 0.72 (average of 0.73, 0.69,0.8 and 0.64).

The Fuzzy Preference Model (FPM): The fuzzy preference model we propose in this paper is a structure $\mathcal{F}_k$ = <$\mathcal{M}$,$PNet_1$,...,$PNet_k$>, where $\mathcal{M}$ is a classification model extracted from the set of layers $I_1, ..., I_k$ (considering elements in I_i as (w_1, w_2, λ_i) with the extra "class" dimension λ_i), and $PNet_i$ is a *Bayesian Preference Network (BPN)* extracted from layer I_i, for all $i = 1, ..., k$. BPNs have been introduced in [de Amo et al. 2013] for mining *crisp* preference models from a set of pairwise preference data. In the crisp scenario, a *unique* BNP is extracted from the input data. This unique BPN is used to predict if (yes or no) a tuple w_1 is preferred to a tuple w_2. In the fuzzy scenario, we will mine several BPNs, one for each layer. Fig. 1(e) illustrates a FDM extracted from the pre-processed preference input data of Fig. 1(d). BPNs are defined as follows:

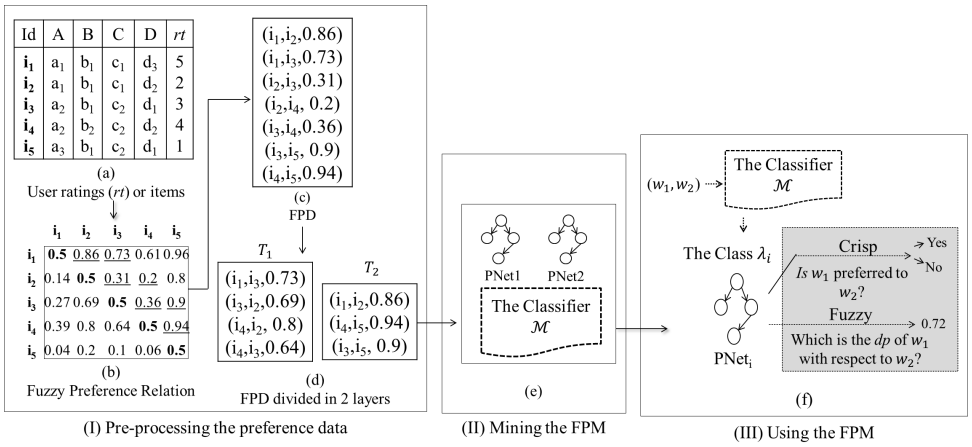


Figure 1: Fuzzy Contextual Preference Mining Process

Definition 1 ([de Amo et al. 2013]) A *Bayesian Preference Network* (BPN) over a relational schema R(A_1, ..., A_n), consists in (1) a directed acyclic graph G whose nodes are attributes in $\{A_1, ..., A_n\}$ and the edges stand for *attribute dependency* and (2) a mapping θ that associates to each node of G a finite set of conditional probabilities.

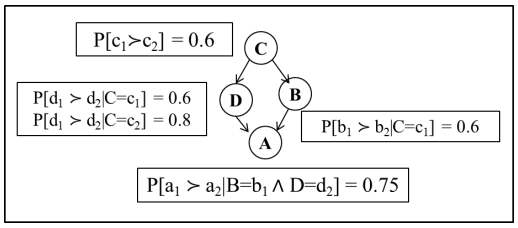


Figure 2: A Bayesian Preference Network

Fig. 2 illustrates a BPN **PNet**$_1$ over the relational schema $R(A, B, C, D)$. Notice that the preference on values for attribute B *depends* on the *context* C: if $C = c1$, the probability that value b_1 is preferred to value b_2 for the attribute B is 60%. The following example illustrates how BNPs are used to predict the order between two new tuples in the crisp scenario:

Example 1 Let us consider the BPN **PNet**$_1$ depicted in Fig. 2. In order to compare two **new** tuples $w_1 = (a_1, b_1, c_1, d_1)$ and $w_2 = (a_2, b_2, c_1, d_2)$, we proceed as follows: (1) Let $\Delta(w_1, w_2)$ be the set of attributes for which w_1 and w_2 differ. In this example, $\Delta(w_1, w_2)$ = $\{A, B, D\}$; (2) Let min($\Delta(w_1, w_2)$) $\subseteq \Delta(w_1, w_2)$ such that the attributes in min(Δ) have no ancestors in Δ (according to graph G underlying the BPN **PNet**$_1$). In this example min($\Delta(w_1, w_2)$) = $\{D, B\}$. The necessary and sufficient conditions for w_1 to be preferred to w_2 are: $w_1[D] \succ w_2[D]$ and $w_1[B] \succ w_2[B]$; (3) Compute the following probabilities: p_1 = probability that $w_1 \succ w_2 = P[d_1 \succ d_2 | C = c_1] * P[b_1 \succ b_2 | C = c_1]$ = 0.6 * 0.6 = 0.36; p_2 = probability that $w_2 > w_1 = P[d_2 \succ d_1 | C = c_1] * P[b_2 \succ b_1 | C = c_1]$ = 0.4 * 0.4 = 0.16. In order to compare w_1 and w_2 we select the higher between p_1 and p_2. In this example, $p_1 > p_2$ and so, we infer that w_1 is preferred to w_2. If $p_1 = p_2$, a random choice decides if w_1 is preferred to w_2 or vice-versa. So, a BPN is capable to compare any pair of tuples: either it *effectively* infer the preference ordering without randomness or it makes a random choice.

How a FDM can be used to predict preference degree between tuples. Let $\mathcal{F}_k$ = $<\mathcal{M}$,$PNet_1$,...,$PNet_k>$ be a k-layer FDM and let w_1, w_2 be two new tuples. First of all, we have to infer the intensity of preference between these tuples. The classifier $\mathcal{M}$ is responsible for this task: it is executed on (w_1, w_2) and returns a class λ_i for this pair. Then the BPN $PNet_i$, ***specific*** for this layer, is executed on (w_1, w_2) to decide which one is preferred. This execution follows the same steps as illustrated in Example 1. If $PNet_i$ decides that w_1 is preferred to w_2 (resp. w_2 is preferred to w_1), the final prediction is : w_1 *is preferred to* w_2 *with degree of preference* λ_i (resp. $1 - \lambda_i$). Remark that the classifier $\mathcal{M}$ only decides how intense the preference between the tuples is. The BPN decides the *preference ordering* between them. This prediction may be *effective* or *random* as remarked at the end of Example 1. Fig. 1(f) illustrates how a FPM can be used to predict both crisp and fuzzy preferences.

Measuring the quality of a fuzzy preference model. Since an exact match between the predicted dp (dp_p) and the *real* one (dp_r) is quite unlikely to occur, we consider a threshold $\sigma > 0$ to measure how good is the prediction. We consider that the predicted dp_p is *correct* if the following conditions are both verified: (1) $\mid dp_r - dp_p \mid \leq \sigma$ and (2) ($dp_r \geq 0.5$ and $dp_p \geq 0.5$) or ($dp_r < 0.5$ and $dp_p < 0.5$). We say that the inference is effective (resp. random) depending on the effectiveness or randomness of the BPN prediction. The quality of a FDM $\mathcal{F}_k$ is evaluated by calculating the following measures on a *test* FPD $\mathcal{P}$:
The ***Accuracy****(acc)*: defined by acc($\mathcal{F}_k$,$\mathcal{P}$,σ)=$\frac{N_e+N_r}{M}$, where M is the number of triples in $\mathcal{P}$, N_e (resp. N_r) is the amount of triples $(t_1, t_2, dp_r) \in \mathcal{P}$ for which the dp_p inferred by $\mathcal{F}_k$ is effectively (resp. randomly) correct ;
The ***Recall****(rec)*: defined by rec($\mathcal{F}_k$,$\mathcal{P}$,σ)= $\frac{N_e}{M}$;
The ***Randomness Rate****(rr)*: defined by rr($\mathcal{F}_k$,$\mathcal{P}$,σ)= $\frac{N_r}{M}$;
The ***Precision****(prec)*: defined by $prec$($\mathcal{F}_k$,$\mathcal{P}$,σ)= $\frac{N_e}{N_c}$, where N_c is the number of triples $(t_1, t_2, dp_r) \in \mathcal{P}$ that have been effectively compared by $\mathcal{F}_k$ (correctly or not);
The ***Comparability Rate****(cr)*: defined by rec($\mathcal{F}_k$,$\mathcal{P}$,σ))= $\frac{N_c}{M}$.

Example 2 Let us consider tuples w_1 and w_2 of Example 1. Let us also suppose the input FPD depicted in Fig. 1(d) with its 2-layer partitions T_1 and T_2. The corresponding classes are λ_1 = 0.72 (average dp of [0.64, 0.8]) and λ_2 = 0.9 (average dp of [0.83, 0.9]). Let us suppose that $dp_r(w_1, w_2) = 0.76$ and that the classifier $\mathcal{M}$ assigns the class λ_1 to (w_1, w_2). Notice that the classifier only predicts that the insensity of preference between the two tuples is 0.72. It doesn't say anything about the *ordering* of preference between them. This task is achieved by the BPN_1 associated to partition T_1. Let us suppose that BPN_1 is the same as the one illustrated in Fig. 2. It compares w_1 and w_2 as described in Example 1, by calculating the probabilities $p_1 = 0.36$ and $p_2 = 0.16$. As $p_1 > p_2$ then the predicted dp is $dp_p = \lambda_1 = 0.72$. So, both dp_r and dp_r are ≥ 0.5 (condition (2) of the quality test is verified). Let us consider parameter $\sigma = 0.05$ as in the experiments of Section 5. Since $\mid 0.76 - 0.72 \mid = 0.04 < 0.05$, we conclude that the fuzzy model correctly and effectively predicted the dp of w_1 with respect to w_2

The Problem of Mining Fuzzy Contextual Preferences. This problem consists in, given a fuzzy preference database and $k > 0$, return a k-layer fuzzy preference model $\mathcal{F}_k$ having good quality with respect to $acc, prec, rec, cr$ and rr measures and a given threshold σ.

3. Algorithm FuzzyPrefMiner

The FuzzyPrefMiner algorithm we propose to solve the Fuzzy Contextual Preference Mining Problem follows the same strategy of the algorithm CPrefMiner proposed in [de Amo et al. 2013] to solve the crisp counterpart problem. The general framework of the algorithm is showed in Alg. 1. In this section we present the details only of the parts of the method that have been modified in order to treat the fuzzy information. The specific procedures which have been modified are indicated inside a rectangle, in boldface.

Algorithm 1: The FuzzyPrefMiner Algorithm

Input: A Fuzzy Preference Database $\mathcal{P}_i$ over relational schema $R(A_1, ..., A_n)$ *% corresponding to a layer in the entire training FPD* $\mathcal{P}$;
Output: A BPN **PNet**

1 *Extract*($\mathcal{P}$, G);
2 **for** *each* $i = 1, \ldots, n$ **do**
3 *Parents*(G, A_i) = $[A_{i_1}, ..., A_{i_m}]$;
4 ***CPTable***$(A_i$***, Parents***$(A_i, G))$ = $[CProb_1, ..., CProb_l]$;
5 **return** *PNet*

3.1. Learning the Network Structure

Procedure *Extract*(Alg. 2) is responsible for learning the network topology G from the training preference database $\mathcal{P}$. G is a graph with vertices in $A_1, ..., A_n$. An edge (A_i, A_j) in G means that the preference on values for attribute A_j depends on values of attribute A_i. That is, A_i is part of the context for preferences over the attribute A_j. This learning task is performed by a genetic algorithm which generates an initial population **P** of graphs with vertices in $A_1, ..., A_n$ and for each graph $G \in$ **P** evaluates a fitness function **score** on G. Full details on the codification of individuals and on the crossover and mutation operations are presented in [de Amo et al. 2013].

Algorithm 2: *Extract* Procedure - [de Amo et al. 2013]

Input: $\mathcal{P}$: a fuzzy preference database over relational schema $R(A_1, ..., A_n)$; β: population size; ϑ: number of generations; γ: number of attribute orderings
Output: A directed graph $G = (V, E)$, with vertices $V \subseteq \{A_1, ..., A_n\}$ that fits best to the fitness function

1 **for** *each* $i = 1, \ldots, \gamma$ **do**
2 Randomly generate an attribute ordering;
3 Generate an initial population I_0 of random individuals;
4 ***Evaluate individuals in*** I_0***, i.e, calculate their fitness***;
5 **for** *each* j*-th generation,* $j = 1, \ldots, \vartheta$ **do**
6 Select $\beta/2$ pairs of individuals from I_j;
7 Apply crossover operator for each pair, generating an offspring population I'_j;
8 Evaluate individuals of I'_j;
9 Apply mutation operator over individuals from I'_j, and evaluate mutated ones;
10 From $I_j \cup I'_j$, select the β fittest individuals through an elitism procedure;
11 Pick up the best individual after the last generation;
12 **return** *the best individual among all* γ *GA executions*

The Fitness Function. This is the only part of the *Extract* procedure where the degrees of preference included in the input data are relevant. The degree of preference affects primarily the way one decides if a given structure is good or not.

The main idea of the fitness function is to assign a real number (a score) in $[-1, 1]$ for a candidate structure G, aiming to estimate how good it captures the dependencies between attributes in a fuzzy preference database $\mathcal{P}$. In this sense, each network arc is "punished" or "rewarded", according to the matching between each arc (X, Y) in G and the corresponding *degree of dependence* of the pair (X, Y) with respect to $\mathcal{P}$.

Algorithm 3: The degree of dependence between a pair of attributes

Input: $\mathcal{P}$: a fuzzy preference database; (X, Y): a pair of attributes; two thresholds $\alpha_1 \geq 0$ and $\alpha_2 \geq 0$.
Output: The Degree of Dependence of (X, Y) with respect to $\mathcal{P}$

1 **for** *each pair* $(y, y') \in$ ***dom***$(Y) \times$ ***dom***(Y), $y \neq y'$ *and* (y, y') *comparable* **do**
2 **for** *each* $x \in$ ***dom***(X) *where* x *is a cause for* (y, y') *being comparable* **do**
3 Let $f_1(S_{x|(y,y')}) = \max\{N, 1 - N\}$, where
$N = \dfrac{\{\sum dp \in DP(S_{x|(y,y')})\} : t \succ t' \wedge (t[Y] = y \wedge t'[Y] = y')}{\sum dp \in DP(S_{x|(y,y')})}$
4 Let $f_2(T_{yy'}) = \max\{f_1(S_{x|(y,y')}) : x \in \mathbf{dom}(X)\}$
5 Let $f_3((X, Y), \mathcal{T}) = \max\{f_2(T_{yy'}) : (y, y') \in \mathbf{dom}(Y) \times \mathbf{dom}(Y), y \neq y', (y, y')$ comparable$\}$
6 **return** $f_3((X, Y), \mathcal{T})$

The *degree of dependence* of a pair of attributes (X, Y) with respect to a FPD $\mathcal{P}$ is a real number that estimates how preferences on values for the attribute Y are influenced by values for the attribute X. Its computation is carried out as described in Alg. 3. In order to facilitate the description of Alg. 3 we introduce some notations as follows: **(1)** For each $y, y' \in \mathbf{dom}(Y)$, $y \neq y'$ we denote by $T_{yy'}$ the subset of pairs $(t, t') \in \mathcal{P}$, such that $t[Y] = y \wedge t'[Y] = y'$ or $t[Y] = y' \wedge t'[Y] = y$; **(2)** If S is a set of triples in $\mathcal{P}$, we denote by $DP(S)$ the (multi)set constituted by the dp appearing in the triples of S (repetitions are considered as different elements); **(3)** We define support$((y, y'),\mathcal{P})$ = $\frac{\sum dp \in DP(T_{y,y'})}{\sum dp \in DP(\mathcal{P})}$. We say that the pair $(y, y') \in$ **dom** $(Y) \times$ **dom** (Y) is *comparable* if support$((y, y'),\mathcal{P}) \geq \alpha_1$, for a given threshold α_1, $0 \leq \alpha_1 \leq 1$; **(4)** For each $x \in \mathbf{dom}(X)$, we denote by $S_{x|(y,y')}$ the subset of $T_{yy'}$ containing the triples (t, t', n) such that $t[X] = t'[X] = x$; **(5)** We define support$((x|(y, y')),\mathcal{P})$ = $\frac{\sum_{dp \in DP(S_{x|(y,y')})} dp}{\sum_{z \in \mathbf{dom}(X), dp \in DP(S_{z|(y,y')})} dp}$; **(6)** We say that x is a *cause for* (y, y') *being comparable* if support$(S_{x|(y,y')},\mathcal{P}) \geq \alpha_2$, for a given threshold α_2, $0 \leq \alpha_2 \leq 1$.

The fitness score associated to G is calculated as in [de Amo et al. 2013] by the formula $\dfrac{\sum\limits_{X,Y} g((X, Y), G)}{n(n-1)}$ where function g is calculated as follows: **(1)** If $f_3((X, Y), \mathcal{P}) \geq 0.5$ and edge $(X, Y) \in G$, then $g((X, Y), G) = f_3((X, Y), \mathcal{P})$; **(2)** If $f_3((X, Y), \mathcal{P}) \geq 0.5$ and edge $(X, Y) \notin G$, then $g((X, Y), G) = -f_3((X, Y), \mathcal{P})$; **(3)** If $f_3((X, Y), \mathcal{P}) < 0.5$ and edge $(X, Y) \notin G$, then $g((X, Y), G) = 1$; **(4)** If $f_3((X, Y), \mathcal{P}) < 0.5$ and edge $(X, Y) \in G$, then $g((X, Y), G) = 0$. More details on the motivation behind this computation ***in the crisp scenario***, please see [de Amo et al. 2013].

3.2. Calculating the Probability Tables Associated to each Vertex

Procedure *Parents(A_i, G)* returns the list of the parents of vertex A_i in the directed graph G. Procedure *CPTable*(A_i, Parents(A_i)) returns, for each vertex A_i of the graph returned by Procedure *Extract*, a list of conditional probabilities $[CProb_1, ..., CProb_l]$. Each conditional probability $CProb_i$ is of the form $Pr[E_1|E_2]$ where E_2 is an event of the form $A_{i_1} = a_1 \wedge ... \wedge A_{i_l} = a_l$ and E_1 is event of the form $(B = b_1) \succ (B = b_2)$.

The second point in the FuzzyPrefMining algorithm where the degree of preference is relevant is in procedure *CPTable*. We calculate the maximum likelihood estimates for each conditional probability distribution of our model. The underlying intuition of this principle uses frequencies as estimates; for instance, if want to estimate $P(A = a \succ A = a'|B = b, C = c)$ we need to calculate $\frac{\sum n \in DP(S(a,a'|b,c))}{\sum m \in DP(S(a,a'|b,c)) + \sum m' \in DP(S(a',a|b,c))}$, where $S(a, a'|b, c)$ = set of triples (t, t', dp) such that $t[B] = t'[B] = b$ and $t[C] = t'[C] = c$ and $t[A] = a$ and $t'[A] = a'$.

4. Experimental Results

4.1. Preparing the Experiments

The Fuzzy Preference Datasets (FPD) The datasets used in the experiments have been obtained from the GroupLens Project[5]. The original data are of the form (UserId, FilmId, rating). More details about the movies, namely Genre, Actors, Director, Year and Language have been obtained by means of a crawler which extracted this information from the IMDB website[6]. Six datasets of film evaluations, corresponding to six different users have been considered. Details about them are shown in Fig. 3(a). For instance, dataset D1 is a set of 34.526 triples (w_1, w_2, n) where w_1, w_2 are taken from a pool of 300 movies.

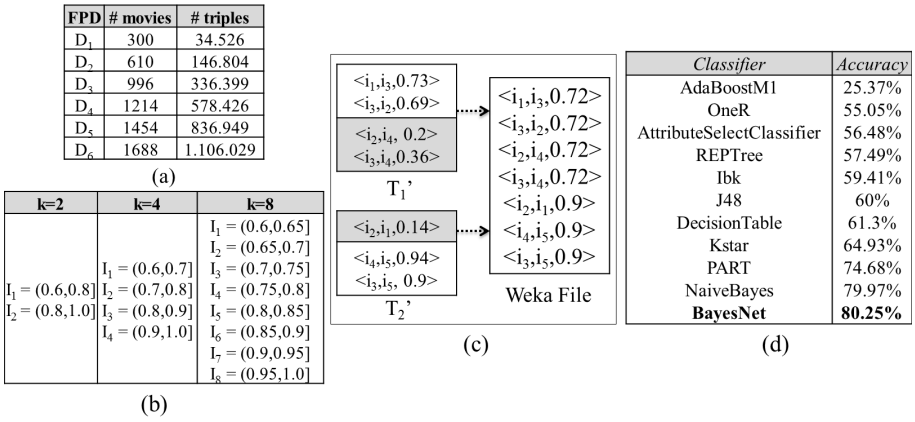


Figure 3: (a) The FPDs, (b) The Preference Layers, (c) Preparing the training data for classification, (d) Classifier's Accuracy

Training a Classifier. A classifier is trained on each dataset in order to predict the "intensity of preference" existing between two movies. A preliminary pre-processing is needed before the classifier training phase. Each preference layer should contain 50% of triples with $dp \geq 0.5$ and 50% with $dp < 0.5$ in order to contain equal amount of pairs (w_1, w_2) with "yes" and "no" answers to "*is* w_1 *preferred to* w_2 *?*". This pre-processing is illustrated in Fig. 3(c), where the preference layers T_1 and T_2 of Fig. 1(d) have been transformed into layers T_1' and T_2'. After the calculation carried out by the *fuzzification* function f_2 (as explained in Section 3) we have $dp_{min} = 0.6$ and $dp_{max} = 1.0$ for each FPD considered in our experiments. The interval I = $[0.6, 1.0]$ is partioned into k layers, and triples of each layer receives an extra class attribute with value λ_i, the average dp for the layer. The triples of the Weka File of Fig. 3(c) are depicted with their respective "classes" λ_1 = 0.72 and λ_2 = 0.9. The Weka classifiers have been trained on a FPD of 31713 triples partioned into 4 layers: T_1 = $(0.6, 0.7]$, T_2 = $(0.7, 0.8]$, T_3= $(0.8, 0.9]$ and T_4

[5]www.grouplens.org
[6]imdb.com

= $(0.9, 1.0]$. The accuracy results are shown in Fig. 3(d). The Bayesian classifiers *Naive Bayes* and *BayesNet* presented the best results. Besides they execute faster then the other ones. Thus, the *BayesNet* has been chosen for the remaining experiments concerning FuzzyPrefMiner evaluation.

4.2. Performance and Scalability Results

A *30-cross-validation* protocol has been considered in the experiments with FuzzyPrefMiner. FuzzyPrefMiner was implemented in Java and all the experiments were performed on a core i7 3.20GHZ processor, with 32GB RAM, running on operation system Linux Ubuntu 12.10. The following default values have been set for the parameters involved in the algorithm: (1) *Extract* Procedure: $\beta = 50$, $\vartheta = 100$ and $\gamma = 20$; (2) Algorithm 3: $\alpha_1 = 0.2$ e $\alpha_2 = 0.2$. For evaluating the $acc, prec, rec, cr, rr$ measures, a threshold σ = 0.05 has been considered.

Experiments have been carried out in order to test our original claim: *the more information is contained in the input data the better are the performance results*. We compared FuzzyPrefMiner (acting in the *crisp* and *fuzzy* scenarios) with the algorithm CPrefMiner [de Amo et al. 2013]. Both algorithms return BNPs, but CPrefMiner learns its BNP from a *crisp* training set and FuzzyPrefMiner from a *fuzzy* one. Fig. 4(a) and Fig. (b) present the results comparing CPrefMiner and FuzzyPrefMiner (validated in both scenarios *Crisp* and *Fuzzy*). A partioning with 8 layers has been considered in this group of experiments for FuzzyPrefMiner.

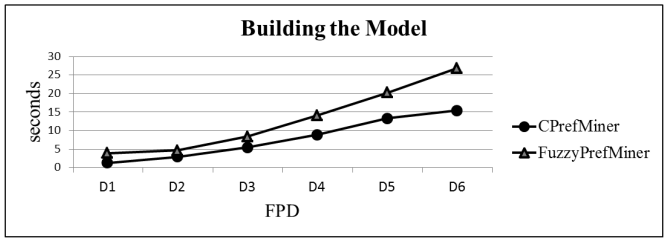
These results show the superiority of FuzzyPrefMiner over CPrefMiner, even when validated in the *fuzzy* scenario. As expected the performance of FuzzyPrefMiner in the *fuzzy* scenario decreases with respect to its performance in the *crisp* scenario, since the fuzzy validation protocol requires predicting the degree of preference between two tuples and not only their relative preference ordering. As far as measures cr and rr are concerned, the results do not differ significantly, as shown in 4(b).

FPD	CPrefMiner			FPMiner: Crisp			FPMiner: Fuzzy		
	acc	*rec*	*prec*	*acc*	*rec*	*prec*	*acc*	*rec*	*prec*
D1	81.24%	79.45%	82.34%	**87.41%**	**85.64%**	**88.80%**	81.60%	79.93%	82.77%
D2	80.48%	79.46%	81.18%	**88.42%**	**87.17%**	**89.45%**	85.90%	84.73%	86.96%
D3	83.85%	83.32%	84.32%	**90.09%**	**89.21%**	**90.83%**	88.33%	87.41%	89.13%
D4	81.79%	80.44%	82.71%	**89.96%**	**89.23%**	**90.56%**	88.16%	87.44%	88.76%
D5	81.36%	80%	82.25%	**89.25%**	**88.50%**	**89.85%**	86.97%	86.18%	87.65%
D6	81.29%	80.86%	81.56%	**89.48%**	**88.70%**	**90.07%**	86.96%	86.18%	87.58%

(a)

FPD	CPrefMiner		FPMiner: Crisp		FPMiner: Fuzzy	
	cr	*rr*	*cr*	*rr*	*cr*	*rr*
D1	95.50%	1.78%	96.45%	1.76%	**96.47%**	**1.67%**
D2	**97.91%**	**1.02%**	97.46%	1.25%	97.43%	1.17%
D3	**98.72%**	**0.63%**	98.21%	0.89%	98.07%	0.92%
D4	97.26%	1.35%	**98.53%**	0.73%	98.51%	**0.72%**
D5	97.29%	1.36%	**98.49%**	**0.75%**	98.31%	0.79%
D6	**99.14%**	**0.43%**	98.48%	0.77%	98.40%	0.77%

(b)



(c)

Using the Model	
CPrefMiner	FuzzyPrefMiner
35 ms	50 ms

(d)

Figure 4: (a) *acc*, *prec*, *rec* results; (b) *cr*, and *rr* results; (c) time spent for building the models; (d) time spent for using the model on 1000 pairs of tuples.

Execution Time. Fig. 4(c) presents the time spent by CPrefMiner and FuzzyPrefMiner to build their respective models. In the case of FuzzyPrefMiner the total time includes the time spent for pre-processing the input data for the classifier (involving I/O operations),

the time spent by the classifier BayesNet to build the classification model and the time spent to mine the 8 BNPs, one for each partition. As expected, FuzzyPrefMiner is more costly than CPrefMiner. The Fig. 4(d) shows the time spent to predict the preference for 1000 pairs of tuples. Again, FuzzyPrefMiner is a little more costly than CPrefMiner taking nearly $1.4T_3$, where T_3 is the time spent by CPrefMiner to accomplish the same task.

5. Conclusion

We studied the influence of the *degree of preference* in the performance of Preference Mining techniques specific for learning contextual preference models and propose the algorithm FuzzyPrefMiner for this task, with very promising results. Our future research concentrate on the design of techniques for *repairing inconsistency* in the fuzzy relation produced by FuzzyPrefMiner. Although the original BPN introduced in our previous works is capable to infer a strict partial order on the set of tuples, the fuzzy relation produced by FuzzyPrefMiner does not verify some transitivity properties of fuzzy relations such as weak transitivity or additive transitivity [Herrera-Viedma et al. 2004].

Acknowledgements. We thank the Brazilian Research Agencies CNPq and FAPEMIG for supporting this work.

References

Amo, S., Diallo, M., Diop, C., Giacometti, A., Li, H., and Soulet, A. (2012). Mining contextual preference rules for building user profiles. In *Data Warehousing and Knowledge Discovery*, volume 7448, pages 229–242. Springer Berlin Heidelberg.

Burges, C. J. C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. N. (2005). Learning to rank using gradient descent. In *ICML*, pages 89–96.

Chiclana, F., Herrera, F., and Herrera-Viedma, E. (1998). Integrating three representation models in fuzzy multipurpose decision making based on fuzzy preference relations. *Fuzzy Sets and Systems*, 97(1):33–48.

Cohen, W. W., Schapire, R. E., and Singer, Y. (1999). Learning to order things. *J. Artif. Intell. Res.*, 10:243–270.

Crammer, K. and Singer, Y. (2001). Pranking with ranking. In *NIPS*, pages 641–647.

de Amo, S., Bueno, M. L. P., Alves, G., and da Silva, N. F. F. (2013). Mining user contextual preferences. *JIDM*, 4(1):37–46.

Freund, Y., Iyer, R., Schapire, R. E., and Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969.

Herrera-Viedma, E., Herrera, F., Chiclana, F., and Luque, M. (2004). Some issues on consistency of fuzzy preference relations. *European Journal of Operational Research*, 154(1):98–109.

Jiang, B., Pei, J., Lin, X., Cheung, D. W., and Han, J. (2008). Mining preferences from superior and inferior examples. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 390–398. ACM.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *KDD*, pages 133–142.

Koriche, F. and Zanuttini, B. (2010). Learning conditional preference networks. *Artif. Intell.*, 174(11):685–703.

Lops, P., de Gemmis, M., Semeraro, G. : Content-based Recommender Systems: State of the Art and Trends. In Recommender Systems Handbook, Springer 2011, pp 73-105.

Ma, J., Fan, Z.-P., Jiang, Y.-P., Mao, J.-Y., and Ma, L. (2006). A method for repairing the inconsistency of fuzzy preference relations. *Fuzzy Sets and Systems*, 157(1):20–33.

Wilson, N. (2004). Extending cp-nets with stronger conditional preference statements. In *AAAI*, pages 735–741.

Xu, Y., Patnayakuni, R., and Wang, H. (2013). The ordinal consistency of a fuzzy preference relation. *Information Sciences*, 224(0):152–164.

paper:78

Mineração da opinião sobre aspectos de candidatos a eleições em comentários de notícias

Leonardo Augusto Sápiras, Karin Becker

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{lasapiras,karin.becker}@inf.ufrgs.br

Resumo. *A classificação automática da opinião pública sobre aspectos de candidatos a eleições, a partir de opiniões disponíveis na web, é um problema complexo para a Mineração de Opiniões. Este artigo descreve uma experiência de mineração de opiniões em nível de aspecto no contexto de comentários de leitores de jornais sobre notícias eleitorais. Nosso desafio é identificar e sumarizar a opinião baseada em aspectos a partir de fontes de opiniões fracamente estruturadas. Mostramos ser possível identificar, classificar a polaridade e sumarizar a opinião de leitores de um jornal sobre os aspectos Saúde e Educação relacionados a candidatos de uma eleição municipal.*

Abstract. *The automatic classification of opinions about aspects of political candidates, from public web data, is a complex Opinion Mining problem. This paper describes a case study of aspect-based opinion mining in the context of comments that newspaper readers express about political news. Our challenge is to identify and summarize opinions on aspects of election candidates, using an ill-structured source of opinion. Our case study propose techniques that can be used to identify, classify and summarize opinions on Health and Education issued by readers about political candidates.*

1. Introdução

A opinião pública sobre entidades e seus aspectos expressa em documentos textuais pode ser analisada e sintetizada através de Mineração de Opiniões, que realiza essas atividades combinando técnicas de mineração de dados com processamento de linguagem natural [Liu 2012]. Os trabalhos pioneiros nessa área buscaram identificar a opinião global contida em revisões sobre produtos e serviços, escritas por usuários [Pang and Lee 2008, Tsytsarau and Palpanas 2012]. Como um substancial avanço, a opinião passou a ser detalhada em termos dos diferentes aspectos de produtos/serviços [Hu and Liu 2004, Guo et al. 2009, Qiu et al. 2011, Liu et al. 2013], tais como o preço, capacidade ou desempenho de um computador. Essas opiniões detalhadas são extremamente úteis e influenciam potenciais consumidores.

Revisões de produto são fontes de opinião bem estruturadas e possibilitam obter a opinião pública sobre entidades e seus aspectos mais facilmente [Tsytsarau and Palpanas 2012, Liu 2012], porque o alvo da opinião é a entidade sendo revisada. Por outro lado, as opiniões expressas em fontes de opiniões menos estruturadas, tais como redes sociais [Tumasjan et al. 2010, Castellanos et al. 2011], blogs e notícias [Balahur et al. 2010], são mais difíceis de serem mineradas porque as tarefas de

encontrar o conteúdo de opinião, e o seu alvo, são bem mais complexas. Ao contrário do contexto de revisão de produtos, no qual o objeto da revisão é a entidade alvo da opinião, documentos nessas mídias podem conter opiniões sobre múltiplas entidades, sobre aspectos específicos destas, ou mesmo podem não conter nenhuma opinião.

Este artigo descreve uma experiência de mineração de opiniões em nível de aspecto no contexto de comentários de leitores de jornais sobre notícias eleitorais. Nosso desafio é identificar e sumarizar a opinião baseada em aspectos a partir de fontes de opiniões fracamente estruturadas. Aplicamos a mineração de opiniões em fontes de dados relacionadas com eleições, já que a plataforma eleitoral de candidatos inclui propostas relevantes para a população em diversas áreas, tais como saúde, educação, segurança, as quais são exploradas durante uma campanha com o intuito de obter votos. Essas áreas são tratadas neste trabalho como os aspectos de um candidato sobre os quais os leitores podem opinar. Assim, partimos da premissa que, além da percepção global de um candidato, é possível identificar a opinião pública em relação a aspectos específicos deste. Por exemplo, deseja-se identificar que a percepção sobre um candidato X em relação à saúde é mais positiva que a do candidato Y, mas no que se refere à educação a percepção é mais negativa.

A contribuição do presente trabalho é uma abordagem para minerar e sumarizar, em nível de *aspecto*, opiniões sobre candidatos a eleições, a qual foi aplicada a um estudo de caso. Ela integra resultados parciais de experimentos sobre extração de aspectos [Sapiras and Becker 2014], e de mineração de opiniões em nível de sentença sobre a mesma fonte de opiniões fracamente estruturada [Tumitan and Becker 2013, Tumitan and Becker 2014], a saber, opiniões expressas por leitores de jornal em comentários como reações a notícias. Em nosso conhecimento, este é um trabalho pioneiro de mineração de opiniões em nível de aspecto fora do contexto de revisões de produtos.

O restante deste artigo está estruturado como segue: a Seção 2 descreve os trabalhos relacionados; a Seção 3 detalha um estudo de caso para minerar a opinião sobre aspectos de entidades políticas; a Seção 4 apresenta conclusões e trabalhos futuros.

2. Trabalhos relacionados

A mineração de opiniões pode ser dividida em três grandes tarefas [Tsytsarau and Palpanas 2012]: a) identificação de conteúdo subjetivo e do alvo das opiniões; b) classificação da polaridade das opiniões; c) sumarização das opiniões através de métricas e/ou recursos visuais. A mineração de opiniões pode ser realizada em nível de documento, sentença ou aspecto, sendo este último o foco deste trabalho.

Métodos para classificação da polaridade de opiniões incluem abordagens baseadas em dicionário, aprendizagem de máquina, estatísticas e semânticas [Tsytsarau and Palpanas 2012], sendo as duas primeiras as mais frequentemente empregadas [Liu 2012]. A abordagem baseada em dicionários requer bons léxicos sentimento para a língua alvo, e pesquisas mostram a importância de dicionários de domínio (e.g. [Hu and Liu 2004]). Já para bons resultados usando aprendizagem de máquina, é necessário um extenso corpus anotado para treino.

O trabalho pioneiro na identificação de opinião em nível de aspecto foi o de [Hu and Liu 2004], o qual utiliza regras de associação envolvendo substantivos. [Qiu et al. 2011] utilizam dependências sintáticas para identificar o alvo da opinião, mas

bons analisadores sintáticos não estão disponíveis para o português. Modelos mais sofisticados de co-ocorrência (e.g. [Guo et al. 2009]) também foram propostos com o mesmo fim. Essas pesquisas exploram revisão de produtos, que são fontes de opinião mais estruturadas, pois: (i) geralmente apresentam opiniões sobre uma única entidade, (ii) explicitam a entidade-alvo da opinião, (iii) pertencem ao mesmo domínio (e.g. informática, cinema). Não se pode assumir tais características em fontes de dados menos estruturados, como blogs ou redes sociais, pois esses tipos de documentos podem apresentar opiniões sobre múltiplos alvos (ou mesmo não conter opinião) e em domínios sem relação entre si.

Trabalhos que usam fontes de opinião fracamente estruturadas (e.g. tweets, comentários) identificam o alvo de opinião no momento da extração, usando um conjunto de termos que representam a entidade alvo. Trabalhos que se propõem a fazer previsões sobre eleições políticas baseadas em sentimentos usando tweets [Tumasjan et al. 2010, O'Connor et al. 2010] ou comentários de jornais [Tumitan and Becker 2014] usam essa estratégia e realizam a mineração de opiniões em nível de documento ou sentença. LCI [Castellanos et al. 2011] monitora o sentimento em tweets, agrupando os termos mais frequentes em tópicos e sumarizando as opiniões envolvendo esses termos. O Observatório da Web[1] monitora tweets sobre vários assuntos, incluindo eleições, sumarizando os principais tópicos, mas sem envolver mineração da opiniões.

Modelos visuais para a tarefa de sumarização são propostos em trabalhos como [Hu and Liu 2004, Castellanos et al. 2011]. Em nível de aspecto, um modelo comum mostra a quantidade de opiniões positivas/negativas para cada um dos aspectos de um produto frequentemente citados [Hu and Liu 2004]. LCI [Castellanos et al. 2011] usa este modelo de sumarização para entidades e seus aspectos, além de propor uma nuvem de termos para os aspectos mais comentados.

O presente artigo integra contribuições desenvolvidas em trabalhos prévios dos autores. Experimentos para extrair aspectos de comentários de notícias foram reportados em [Sapiras and Becker 2014], mas não envolveram mineração de opiniões. Observamos como comportamento dos leitores que os comentários não necessariamente têm relação com o conteúdo da notícia (e.g. uma notícia sobre transportes pode ter comentários opinando sobre saúde, transportes ou corrupção). Assim, optamos por considerar todos comentários, e experimentamos diferentes técnicas probabilísticas de co-ocorrência para identificar aspectos. O processo de mineração de opiniões para o mesmo estudo de caso foi detalhado em [Tumitan and Becker 2013], incluindo a extração e pré-processamento de comentários, identificação dos candidatos alvo, e classificação da opinião em nível de sentença. A classificação da polaridade usando léxicos de sentimento e aprendizado supervisionado foi comparada em [Tumitan and Becker 2014], também em nível de sentença. O presente trabalho diferencia-se destes ao propor uma abordagem para minerar e sumarizar opiniões em nível de aspecto para a mesma fonte de opiniões fracamente estruturada, a saber, comentários de jornais sobre candidatos a eleições.

3. Estudo de caso

3.1. Objetivo

Neste estudo de caso realizamos uma experiência pioneira de mineração de opiniões em nível de aspecto fora do contexto de revisão de produtos. Tal como em

[1] http://www.observatorio.inweb.org.br/eleicoes2012/destaques/

[Tumitan and Becker 2013], usamos como fonte de opinião comentários sobre notícias envolvendo eleições municipais de 2012 da cidade de São Paulo, e os três candidatos a prefeito mais comentados (i.e. Serra, Russomanno e Haddad). As notícias e comentários foram extraídos da Folha de São Paulo, e o período analisado corresponde ao mês que antecede o primeiro turno da eleição. O objetivo do estudo foi propor uma abordagem que permitisse detalhar a opinião dos leitores em relação a aspectos específicos destes candidatos (e.g. a percepção de X sobre saúde é mais positiva que a dos candidatos Y e Z no mesmo aspecto), e aplicá-la sobre o corpus escolhido. Consideramos os mesmos aspectos já explorados em [Sapiras and Becker 2014]: saúde e educação.

3.2. Abordagem proposta

A abordagem proposta para o estudo de caso é apresentada na Figura 1. As caixas com fundo cinza correspondem às técnicas discutidas em trabalhos prévios dos autores e as com fundo verde correspondem às etapas desenvolvidas no presente artigo. Uma vez extraídos os comentários dos jornais relevantes ao estudo de caso [Tumitan and Becker 2013], dois procedimentos são realizados: a) a identificação dos comentários envolvendo os aspectos alvo de acordo [Sapiras and Becker 2014]; e b) a mineração de opiniões em nível de sentença, para sentenças mencionando os candidatos observados [Tumitan and Becker 2014]. O nível de sentença foi adotado para a polarização da opinião porque cada comentário pode envolver opiniões sobre mais de um candidato. Então, a relação entre as sentenças polarizadas e os respectivos comentários é analisada para verificar se se referem a algum aspecto específico. Finalmente, a opinião dos leitores é sumarizada por candidato e por seus aspectos. Na discussão que segue, sejam $A = \{a_i\}$ um conjunto de aspectos, $E = \{e_j\}$ um conjunto de candidatos monitorados e $C = \{c_k\}$ um conjunto de comentários.

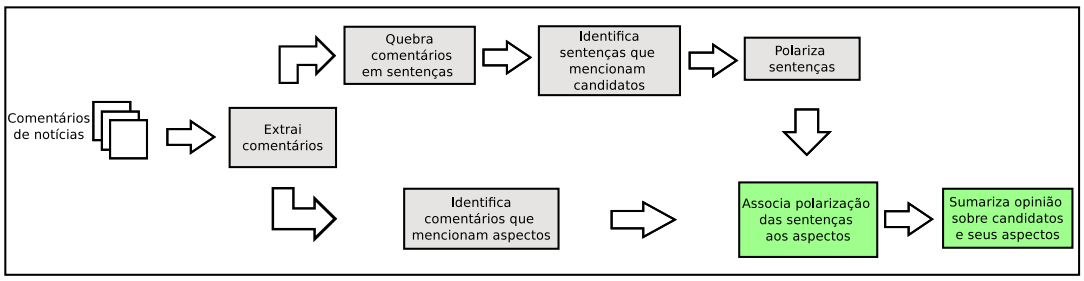


Figura 1. Abordagem de mineração de opiniões em nível de aspecto.

Extração de comentários: extração de comentários sobre notícias eleitorais, identificadas usando rótulos pré-definidos em um indexador de notícias (Google Reader). O pré-processamento elimina comentários duplicados (similaridade superior a 85% usando cosseno), ou muito curtos (até 3 palavras). Os detalhes da extração e limpeza de comentários são relatados em [Tumitan and Becker 2013].

Identificação de comentários que mencionam aspectos: identificação dos comentários $C_a \subseteq C$ que contêm termos representativos de cada aspecto $a \in A$, de acordo o método detalhado na Seção 3.4.

Divisão de comentários em sentenças: utilizando o módulo *punkt* do NLTK [2], treinado para português, os comentários são divididos em sentenças. A relação entre

[2]Natural Language Toolkit - http://www.nltk.org/

sentença e comentário de origem é armazenada, criando o conjunto de sentenças $S = \{< s_i, c_k >: c_k \in C\}$.

Identificação de sentenças que mencionam entidades eleitorais: filtragem de todas sentenças de S que fazem menções explícitas aos candidatos $e \in E$, criando um conjunto $SM = \{< s_i, e_j, c_k >: e_j \in E \wedge < s_i, c_k > \in S\}$. As menções são identificadas usando os nomes dos candidatos (e.g. Serra, Russomano) e suas variações (e.g. vampissera, tarussomano), encontradas a partir de expressões regulares.

Polarização de sentenças: as sentenças com menções SM são polarizadas em três classes, gerando o conjunto $SP = \{< s_i, e_j, pol, c_k >: pol \in \{pos, neut, neg\} \wedge < s_i, e_j, c_k > \in SM\}$. Os detalhes relacionados à classificação das opiniões são discutidos na Seção 3.5.

Associação das sentenças polarizadas aos aspectos: as sentenças com opiniões sobre aspectos específicos são separadas, usando o seu relacionamento com os respectivos comentários. Para cada aspecto $a \in A$, buscamos as sentenças $SP_a = \{< s_i, e_j, pol, c_a >: c_a \in C_a\}$, $SP_a \subseteq SP$, considerando estas como opiniões sobre o aspecto a do candidato e_j.

Sumarização da opinião sobre as entidades e seus aspectos: sumarização das opiniões sobre os candidatos em geral, e sobre seus aspectos, de acordo com os mecanismos apresentados na Seção 3.6. Agregamos as sentenças por comentário e candidato, para representar o número de pessoas emitindo opiniões sobre cada candidato.

3.3. Corpus e Gold Standard

O corpus utilizado consiste de comentários de notícias sobre as eleições municipais de São Paulo relativos aos meses de setembro e outubro de 2012. Após pré-processamento, o corpus foi reduzido a 14.848 comentários, divididos em 79.752 sentenças. Deste corpus, foram extraídos dois subconjuntos de dados a serem usados como *Gold Standard*.

Todos os dados foram anotados por três anotadores, em que somente anotações com no mínimo duas concordâncias foram consideradas. Os anotadores foram orientados a basear sua avaliação apenas no conteúdo explicitamente escrito, sem usar julgamento próprio ou conhecimento do domínio político para inferir entendimento. A concordância dos anotadores para polaridade foi 91,81%, e para aspectos, 85,75%.

Os *datasets* 1 e 2 são subconjunto dos comentários sobre eleições municipais de 2012, enquanto que o *Dataset3* foi criado para as pesquisas relatas em [Tumitan and Becker 2014], e envolve eleições de 2010. A identificação de aspectos foi validada usando o *Dataset1*. Os *datasets* 2 e 3 foram utilizadas para treinar e testar os classificadores de opinião.

- *Dataset1*: aleatoriamente, foram selecionadas 407 notícias eleitorais do corpus, as quais foram anotadas em relação aos tópicos que evocavam (i.e. saúde ou educação). Dos respectivos comentários, foram selecionadas também aleatoriamente 2072 sentenças, em que cada anotador deveria avaliar se a sentença evocava um dos tópicos, quais candidatos eram explicitamente mencionados, se expressava uma opinião e a respectiva polaridade. Assumiu-se que todo comentário que contivesse pelo menos uma sentença anotada como evocando um dado tópico, por transitividade, também mencionava aquele tópico.

- *Dataset2:* aleatoriamente foram selecionadas 1071 sentenças de comentários do mesmo corpus, as quais foram anotadas no tocante à polaridade, possuindo 732 sentenças negativas, 180 neutras e 159 positivas.
- *Dataset3:* considerando as eleições presidenciais e governamentais de São Paulo de 2010, foram selecionadas aleatoriamente 990 sentenças de comentários extraídos segundo o mesmo processo, que, após anotação quanto à polaridade, resultaram em 356 sentenças negativas, 480 neutras e 154 positivas.

3.4. Identificação de Aspectos

A Figura 2 detalha a abordagem utilizada para encontrar os aspectos [Sapiras and Becker 2014], a qual visa identificar os comentários que mencionam determinado aspecto com base em um conjunto de *termos representativos* do mesmo. Dado um conjunto de notícias sobre um dado aspecto (*documentos de domínio*), os termos representativos são encontrados com base na co-ocorrência entre termos encontrados nestes documentos (*palavras candidatas*), e um conjunto de *palavras-semente*.

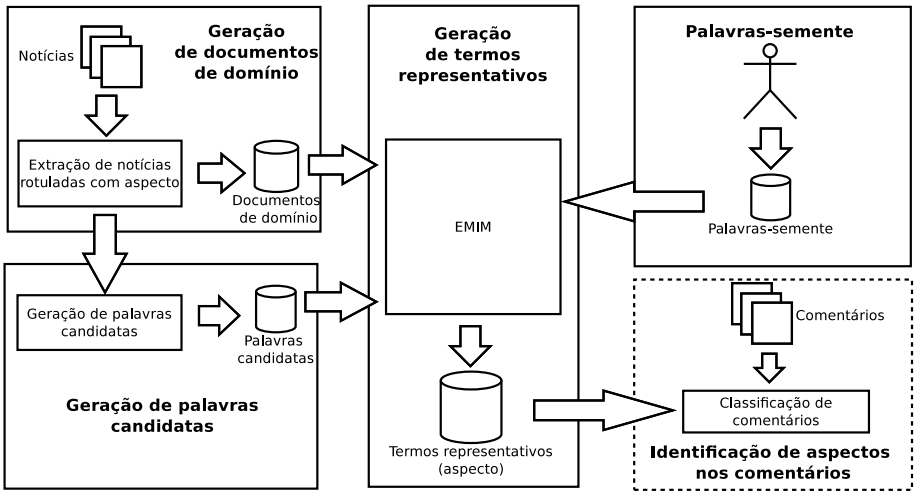


Figura 2. Processo de identificação de aspectos.

No estudo de caso, as palavras-semente foram definidas pelos próprios autores a partir do conhecimento do domínio. Exemplos de palavras-semente para Saúde são "hospital", "médico" e "vacinação", e para o aspecto Educação, "aula" e "escola".

O processo de identificação dos termos representativos tem início com a extração de documentos específicos do domínio de cada aspecto. Nosso estudo de caso utiliza um *corpora* de notícias classificadas pelo Jornal Folha de São Paulo para os aspectos saúde e educação. O processo de extração desse *corpora* resultou em 1000 notícias com o rótulo Educação, e 1000 notícias com o rótulo Saúde.

De cada um desses *corpora*, foram extraídas, usando NLTK, todas as palavras existentes, junto com suas respectivas frequências e classes gramaticais (*part-of-speech tags*). *Stopwords* foram eliminadas. Foram consideradas como *palavras candidatas* apenas os substantivos que apareciam no respectivo corpus (e.g. notícias sobre saúde) e que não apareciam no outro (e.g. notícias sobre educação).

Dentre as técnicas de co-ocorrência examinadas, os melhores resultados foram obtidos com EMIM (Expected Mutual Information Measure), detalhada na Equação 1.

Dado que *x* representa uma palavra candidata e *y* representa uma palavra-semente, *a* é número de vezes que as palavras *x* e *y* co-ocorrem em um documento; *b* é o número de vezes que *x* ocorre em um documento e *y* não ocorre; *c* é o número de vezes que *y* ocorre e *x* não ocorre, e *d* é o número de vezes que nem *x* nem *y* ocorrem um documento.

$$EMIM_{(x,y)} = \log_2 \frac{a(a+b+c+d)}{(a+b)(a+c)} \quad (1)$$

Um termo *x* é considerado como representativo quando: a) $EMIM(x, y) > 0$, e b) *x* e *y* co-ocorrem em, no mínimo, 10 documentos do domínio, em que estes limites foram fixados empiricamente. Em nossos experimentos, melhor detalhados em [Sapiras and Becker 2014], obtivemos 73,83% de precisão, 79,8% de revocação e 76,7% de F-score para o aspecto Saúde. Já para o aspecto Educação, obtivemos 70,54% de precisão, 80,53% de revocação e 75,21% de F-score. Também experimentamos as técnicas LSI e Phi-squared, mas essas apresentaram resultados inferiores.

3.5. Polarização de Opinião

Usamos a experiência desenvolvida em [Tumitan and Becker 2014] para polarizar as sentenças, com a diferença de que consideramos três classes: positivo, neutro e negativo. Optamos por utilizar apenas técnicas de aprendizagem de máquina, testando diferentes classificadores no WEKA [Hall et al. 2009]. Os melhores resultados foram obtidos usando o algoritmo SMO para treinar um classificador Support Vector Machine (SVM). Para preparação de *features*, utilizamos unigramas, representação de pesos usando TF-IDF e seleção de atributos utilizando a técnica BestFirst. Testamos várias outras preparações (e.g. n-gramas, *stopwords*, representação binária e frequência, outras funções de seleção), mas os resultados foram inferiores e não são descritos aqui.

A Tabela 1 apresenta os resultados usando Precisão, Revocação, Medida-F e a respectiva média ponderada. Foram feitos dois experimentos: com validação cruzada, e com conjuntos de treino e teste distintos. Na abordagem SVM (Cross-validation), mesclamos os *datasets* 2 e 3, e verificamos o desempenho do classificador utilizando validação cruzada com 10 *folds*. Já na segunda abordagem, utilizamos o *Dataset3* como conjunto de treinamento e o *Dataset2* como conjunto de teste. Os resultados são animadores quanto à precisão, mas com baixa revocação para algumas classes, em particular a neutra. O pior desempenho da segunda abordagem é devido a *overfiting*, já que termos distintos foram usados em cada eleição. Por exemplo, nas eleições de 2010 existiam várias referências às primeiras candidatas à presidência (e.g. "guerreira", "presidenta"), enquanto que, em 2012, foram evocados feitos passados dos candidatos e os escândalos de seus partidos.

Tabela 1. Resultados da polarização das sentenças com Precisão (P), Revocação (R), F-score (F), respectivas médias ponderadas (WP, WR, WF) e Acurácia (A).

Abordagem	Polaridade	P(%)	R(%)	F(%)	WP(%)	WR(%)	WF(%)	A(%)
SVM (Cross-validation)	Positivo	86,6	51,9	64,9	83,9	82,5	79,5	82,46
	Neutro	93,7	26,8	41,7				
	Negativo	81,6	98,5	89,3				
SVM (Com conjunto de teste)	Positivo	76,2	48,4	59,2	79,3	77,3	73,2	77,31
	Neutro	92,5	20,6	33,6				
	Negativo	76,8	97,5	85,9				

Os melhores resultados foram obtidos para a classe negativa, já que há um número muito maior de sentenças negativas, quando comparado às demais classes. No futuro, reavaliaremos os resultados com classes mais equilibradas. A sumarização dos resultados descrita na próxima seção utiliza SVM com validação cruzada, pois foi a que apresentou o melhor desempenho.

3.6. Sumarização

O modelo visual de sumarização adotado, exemplificado nas Figuras 3 e 4, é semelhante ao apresentado em [Castellanos et al. 2011]. Através dele, é possível visualizar a percepção da população sobre cada candidato em geral, os aspectos analisados em particular, bem como os termos mais comentados em cada tópico.

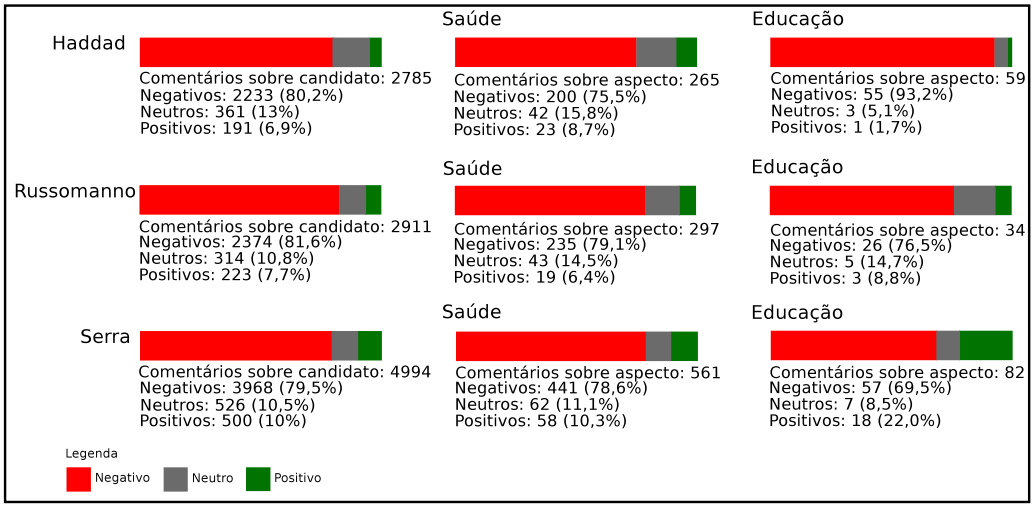


Figura 3. Modelo visual de sumarização de opiniões.

A opinião sobre cada candidato é sumarizada usando o número absoluto e percentual de comentários positivos, negativos ou neutros. O mesmo é feito especificamente para cada aspecto. Os resultados mostrados na Figura 3 correspondem à aplicação da abordagem proposta no *corpus* completo (i.e. 14.848 comentários).

Agregamos as opiniões por comentários para representar *pessoas* emitindo opiniões. O uso de sentenças, como em [Tumitan and Becker 2014], provocaria distorções nos resultados, e.g. um comentário com diversas sentenças negativas teria mais peso que um comentário com uma única sentença positiva. Classificamos um comentário $c_k \in C$ como positivo em relação a uma entidade e_j, se contiver mais sentenças positivas $sp_i \in P$ relacionadas a c_k que mencionem e_j, do que negativas. Caso o número de sentenças negativas seja superior, o comentário é classificado como negativo, se não, ele é neutro. Note-se que o mesmo comentário pode ser contabilizado para mais de um candidato, caso expresse opiniões sobre múltiplos candidatos. Neste caso, são contabilizadas as sentenças referentes a cada candidato. A mesma lógica foi aplicada na sumarização dos comentários por aspecto.

Com base na sumarização apresentada na Figura 3, concluímos que os leitores expressam um número substancialmente maior de opiniões negativas, tanto para os candidatos em geral, quanto para seus aspectos. Observamos também que, apesar de existir

uma grande quantidade de comentários com opinião sobre os candidatos, a proporção em relação ao total de comentários foi relativamente baixa (51%). Além disso, a quantidade de opinião da população sobre os aspectos analisados (saúde e educação) também foi baixa. Isso está relacionado ao fato de que o conteúdo dos comentários apresenta opiniões sobre outros aspectos (e.g. corrupção) ou opiniões diretas aos candidatos. Considerando que dois candidatos haviam sido ex-ministros de educação e saúde, é interessante interpretar os comentários por aspecto. Os leitores têm uma percepção mais positiva sobre educação do ex-ministro da saúde do que o ex-ministro da educação sobre educação.

A nuvem de palavras (Figura 4) mostra os termos representativos para cada aspecto, em que o tamanho das palavras está relacionado à frequência de ocorrência de tais termos nos comentários.



(a) Saúde (b) Educação

Figura 4. Nuvem de palavras para termos representativos de cada aspecto.

4. Conclusões

Este artigo apresentou uma proposta de mineração de opinião pública em nível de aspecto para entidades eleitorais, considerando uma fonte de opinião fracamente estruturada. Desenvolvemos um estudo de caso completo, no qual, identificamos, classificamos e sumarizamos a percepção dos leitores de um jornal sobre os aspectos saúde e educação de candidatos, baseado em seus comentários. Em nosso conhecimento, trata-se de um trabalho pioneiro de mineração de opiniões em nível de aspecto fora do contexto de revisões de produtos. No estudo de caso, foi possível observar que apesar de existir uma grande quantidade de opiniões em relação aos candidatos, a quantidade em relação aos aspectos analisados é pequena. Isso se deve ao fato da população apresentar opiniões sobre outras dimensões destes candidatos.

Este trabalho apresenta limitações, que serão tratadas em trabalhos futuros, tais como a extensão do estudo de caso para outros aspectos, o uso de corpus balanceado para classificação de opiniões, o tratamento de opiniões irregulares e implícitas e a identificação automática das entidades eleitorais nos comentários. Também é importante a identificação automática de aspectos. Para resolver o problema de uma sentença expressar opiniões sobre vários candidatos (e.g. "X é horrível, votarei em Y"), a solução seria polarizar a opinião em nível de cláusulas. Pretendemos também aplicar nossa abordagem para as eleições presidenciais de 2014 e utilizar outros meios além de comentários de notícias, tais como *posts* em redes sociais.

Referências

Balahur, A., Steinberger, R., Kabadjov, M., Zavarella, V., Van Der Goot, E., Halkia, M., Pouliquen, B., and Belyaeva, J. (2010). Sentiment analysis in the news. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC)*, volume 10, page 2216.

Castellanos, M., Dayal, U., Hsu, M., Ghosh, R., Dekhil, M., Lu, Y., Zhang, L., and Schreiman, M. (2011). Lci: a social channel analysis platform for live customer intelligence. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, SIGMOD '11, pages 1049–1058, New York, NY, USA. ACM.

Guo, H., Zhu, H., Guo, Z., Zhang, X., and Su, Z. (2009). Product feature categorization with multilevel latent semantic association. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 1087–1096, New York, NY, USA. ACM.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

Liu, B. (2012). *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Liu, Q., Gao, Z., Liu, B., and Zhang, Y. (2013). A logic programming approach to aspect extraction in opinion mining. In *Proceedings of the 2013 IEEE/WIC/ACM International Conferences on Web Intelligence*, pages 276–283.

O'Connor, B., Balasubramanyan, R., Routledge, B. R., and Smith, N. A. (2010). From tweets to polls: Linking text sentiment to public opinion time series. *ICWSM*, 11:122–129.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.

Qiu, G., Liu, B., Bu, J., and Chen, C. (2011). Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37(1):9–27.

Sapiras, L. and Becker, K. (2014). Identificação de aspectos de candidatos eleitorais em comentários de notícias. In *Anais do III Brazilian Workshop on Social Network Analysis and Mining - BraSNAM 2014*.

Tsytsarau, M. and Palpanas, T. (2012). Survey on mining subjective data on the web. *Data Min. Knowl. Discov.*, 24(3):478–514.

Tumasjan, A., Sprenger, T. O., Sandner, P. G., and Welpe, I. M. (2010). Predicting elections with twitter: What 140 characters reveal about political sentiment. In *ICWSM'10*.

Tumitan, D. and Becker, K. (2013). Tracking Sentiment Evolution on User-Generated Content: A Case Study on the Brazilian Political Scene. In *Anais do XXVIII Simpósion Brasileiro de Banco de Dados*, pages 135–144.

Tumitan, D. and Becker, K. (2014). Sentiment-based features for predicting election polls: a case study on the brazilian scenario. In *Proceedings of the 2014 IEEE/WIC/ACM International Conferences on Web Intelligence*, page 8p. IEEE Computer Society.

paper:79

Mineração de emoções em textos multilíngues usando um *corpus* paralelo

Aline Graciela Lermen dos Santos, Karin Becker, Viviane Moreira

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

{aglsantos,karin.becker, viviane}@inf.ufrgs.br

Abstract. *Multilingual Opinion Mining deals with the analysis of opinions, regardless of the language in which they are written. Works in this area focus on the classification of the polarity of opinions extracted from texts, and less attention has been paid to the classification of emotions. This work proposes the use of Multilingual Opinion Mining techniques for emotion mining using parallel corpora. We developed experiments with two goals: 1) to compare two approaches for emotion classification: lexicon-based and machine learning-based; 2) to analyze whether a specific language produces better classification results. We developed experiments with a parallel corpus composed by lyrics in their original language (English) and their translations to Portuguese. The results show that machine learning is superior to the use of sentiment lexicon, and that there is no statistical difference regarding the languages used for classification.*

Resumo. *A área de Mineração de Opinião Multilíngue surgiu para extrair e analisar opiniões em textos, independente do idioma no qual estão escritos. Os trabalhos na área concentram seus esforços em classificar a polaridade de opiniões extraídas de texto, mas a classificação das emoções ainda é pouco explorada. Este trabalho se propõe a usar técnicas de Mineração de Opinião Multilíngue para minerar emoções em corpora paralelos. Desenvolvemos experimentos com dois objetivos: 1) comparar duas abordagens de classificação de emoções, a saber, baseada em léxico de sentimentos e em aprendizado de máquina; 2) na classificação de emoções baseada em aprendizado de máquina, verificar se algum idioma produz melhores resultados em isolado ou combinado. Os experimentos utilizaram um corpus paralelo composto por letras de música no idioma original (inglês) e suas traduções para o português. Os resultados mostram que a abordagem aprendizado de máquina é superior ao uso de léxico de sentimento e que não há diferença estatística entre as classificações segundo os idiomas.*

1. Introdução

Com a expansão da Internet, é fácil para as pessoas expressarem seus sentimentos sobre diversos assuntos na rede, através de textos postados em blogs, redes sociais, fóruns, entre outros. Esses textos são uma fonte de dados rica para a Análise de Sentimentos (AS), também denominada Mineração de Opinião (MO), que tem como objetivo identificar, classificar e sumarizar sentimentos em textos a respeito de um alvo. Dentre as aplicações mais comuns de AS se encontram sumarização de revisões de produtos, análise de popularidade de marcas, organizações ou pessoas para relações públicas ou

marketing, previsão de indicadores a partir de sentimentos (por exemplo, resultados de eleições), entre outras.

A maioria dos textos encontrados na Web está em inglês. Assim, os recursos desenvolvidos para AS são voltados para esse idioma, tornando a análise em outros idiomas mais desafiadora. A Mineração de Opinião Multilíngue (MOM) [Banea et al. 2008] surgiu para analisar textos contendo opiniões, independentemente do idioma no qual estão escritos. Os trabalhos de MOM se concentram em duas grandes frentes: 1) realizar a MOM combinando recursos já existentes para outros idiomas e tradução automática; 2) desenvolver técnicas de MOM que independam de uma linguagem alvo.

Um sentimento representa uma atitude, opinião ou emoção que o autor da opinião tem sobre o alvo da mesma [Liu 2012]. Frequentemente, este sentimento é medido através de sua polaridade, que procura classificá-lo como positivo, negativo ou neutro. Outra forma de classificar o sentimento é através de classes de emoções [Plutchik 1984][Ekman 1982], tais como alegria ou tristeza. Os trabalhos em MO se concentram em classificar a polaridade, visto ser mais fácil de tratar do que emoções. Contudo, trabalhos em previsão baseada em sentimento mostram que as emoções são indicadores bem mais discriminantes no comportamento do que a simples polaridade (e.g. [Asur and Huberman 2010][Tumasjan et al. 2010][Bollen et al. 2011]), além de existirem várias aplicações importantes baseadas na detecção de emoções (e.g. diagnóstico de distúrbios psicológicos) [Mohammad 2012]. Considerando-se as mesmas carências da MO para idiomas diferentes do inglês, pretende-se contribuir com a área através de técnicas para classificar emoções explorando *corpora* multilíngues.

Neste trabalho, desenvolvemos experimentos para comparar duas abordagens de MOM para Mineração de Emoções em um *corpus* paralelo. Os experimentos consideram dois objetivos: 1) comparar duas abordagens de classificação de emoções, a saber, baseada em léxico de sentimentos e em aprendizado de máquina; 2) na classificação baseada em aprendizado de máquina, verificar se algum idioma produz melhores resultados em isolado (monolíngue) ou combinado (multilíngue). Os resultados mostram que o aprendizado de máquina é superior ao uso de léxico de sentimento, o que é interessante, já que léxicos estão restritos a linguagens específicas. Não há diferença significativa entre os idiomas testados. Contudo, a classificação baseada em treinamento multilíngue teve desempenho marginalmente superior ao treinamento monolíngue. Estes experimentos são importantes para compreender a contribuição de técnicas multilíngues de MO para a classificação de emoções considerando textos em línguas tradicionalmente sem recursos léxicos/sintáticos, como a língua portuguesa.

O artigo segue estruturado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados estudados para o desenvolvimento do trabalho; a Seção 3 descreve a estrutura dos experimentos, incluindo o *corpus* e recursos utilizados, o método de classificação, os resultados e suas considerações; o trabalho é encerrado com conclusões e trabalhos futuros na Seção 4.

2. Trabalhos Relacionados

MO utiliza duas principais estratégias para classificar o sentimento: a abordagem léxica: baseada no uso de léxico de sentimentos, ou em aprendizado de máquina (que necessita de dados rotulados para treinamento [Tsytsarau e Palpanas 2012]). Léxicos (ou dicionários) de sentimentos consistem em um conjunto de palavras rotuladas para uma

medida de sentimento, sendo polaridade a mais encontrada. Dentre os algoritmos de aprendizado mais usados na MO, se encontram Naïve Bayes e SVM [Liu 2012][Tsytsarau e Palpanas 2012]. Entre as dificuldades da MOM, podem ser destacadas a carência de léxicos de sentimentos em idiomas diversos e de *corpora* anotados para aprendizado supervisionado.

Os trabalhos de MOM em sua maioria lidam com geração de recursos para outros idiomas usando como medida polaridade. É o caso do trabalho de Banea et al. [2008], que a partir de tradução automática de recursos existentes em inglês cria recursos para MOM em espanhol e romeno. Outros trabalhos optam por fazer análise de sentimentos sem recursos dependentes de idioma, como o de Narr et al. [2013], que utiliza *emoticons* para criar dados de treinamento a partir de *tweets* em diversos idiomas. Entretanto, esses trabalhos estão restritos ao uso de polaridade.

As limitações da polaridade de sentimento são consideradas em trabalhos que agregam emoção à polaridade para desenvolver modelos preditivos melhores (e.g. bilheteria de cinema [Asur and Huberman 2010], movimentação de ações [Bollen et al. 2011], eleições [Tumasjan et al. 2010]). Mohammad [2012] identifica emoções em livros e e-mails através da co-ocorrência de palavras de emoção usando um léxico de sentimentos em língua inglesa. Martinazzo e Paraiso [2010] classificam emoções em notícias curtas em português nas classes de emoções básicas de Ekman, exploradas a seguir, utilizando *Latent Semantic Analysis*. Esses trabalhos, no entanto, tratam apenas um idioma.

Como são diversas as classes de emoção (como alegria e raiva), é necessário selecionar um conjunto para o estudo; entretanto os trabalhos envolvendo emoção não entram em consenso quanto a um conjunto básico de emoções, cada um utilizando um conjunto arbitrário [Ortony and Turner 1990]. Um dos conjuntos mais utilizados é o definido por Ekman [1992], que define como básicas as classes de emoção *joy*, *sadness*, *anger*, *fear*, *disgust* e *suprise*[1]. Outro conjunto utilizado é o de Plutchik [1984], que adiciona ao conjunto de Ekman as emoções *trust* e *anticipation*[2].

Diferente da polaridade, onde uma categoria se opõe à outra (positiva versus negativa), nem toda emoção possui uma emoção oposta, o que complica a análise envolvendo negação no texto. Outro aspecto importante é que uma emoção pode ser considerada positiva ou negativa, dependendo do contexto: surpresa durante um filme pode ser considerada positiva, enquanto surpresa no funcionamento de um produto pode ser negativa se o produto não apresentou o funcionamento esperado.

Os trabalhos relacionados ou lidam com emoções ou trabalham com MOM, havendo uma escassez de trabalhos que abordem os dois temas. Em um estudo preliminar [Santos et al. 2014] com corpora comparáveis, concluímos que a tradução automática do corpus é uma abordagem viável para MOM. O presente trabalho visa contribuir para a área comparando abordagens distintas para minerar emoções em textos multilíngues: uso de léxico de sentimentos e uso de aprendizado de máquina, utilizando um *corpus* multilíngue.

[1] Termos traduzidos: alegria, tristeza, raiva, medo, repugnância e surpresa.
[2] Termos traduzidos: confiança e expectativa.

3. Experimentos

Usando um corpus paralelo, os experimentos relatados neste trabalho têm dois objetivos: a) fazer uma comparação entre duas abordagens de MOM para classificação de emoções: baseada em léxico de sentimentos e baseada em aprendizado de máquina; b) comparar os resultados de classificação supervisionada por idioma, ou combinação de idiomas.

Para a classificação das emoções, serão utilizadas as 8 categorias de Plutchik, por serem aquelas adotadas pelo dicionário de sentimentos utilizado, a saber, NRC (detalhado na seção 3.1). Para a abordagem baseada em aprendizado supervisionado, será utilizada a classificação Naïve Bayes, por apresentar bons resultados na área, como amplamente relatado na literatura [Liu 2012][Tsytsarau e Palpanas 2012].

Uma das limitações da área é que existem poucos léxicos de emoções, havendo divergências entre as classes de emoções consideradas. Pela classificação de Plutchik ser bastante citada, assim como o uso de NRC para este fim, adotamos estas 8 categorias. Contudo, esta falta de consenso nas classes de emoção é uma das motivações de uma abordagem independente de recursos específicos, como a baseada em aprendizado supervisionado. Assim, com o primeiro objetivo buscamos responder a questão: "*É melhor depender de um recurso específico para um idioma, como o NRC, ou de uma abordagem independente de linguagem, como o classificador Naïve Bayes*?".

Os idiomas envolvidos são inglês (*corpus* e léxico) e português (*corpus*). Optou-se pelo idioma português por não haver variedade de recursos para este idioma, e pela facilidade de obtenção de um *corpus* paralelo.

Considerando os idiomas envolvidos e o uso de um *corpus* paralelo, procura-se saber, a partir da abordagem de aprendizado de máquina no segundo experimento, se "*é melhor classificar os textos a partir do idioma inglês, português ou da combinação de ambos?*". Como não há uma versão em português do léxico de sentimento usado, não há como fazer essa verificação na abordagem de léxico de sentimento.

Ambas as abordagens consideram a classificação da emoção nos textos em português, explorando o *corpus* paralelo. Para a classificação baseada em léxico, busca-se o texto em inglês correspondente, classificando-o com o léxico NRC, e os rótulos de emoção resultantes são projetados para o texto em português. Já na abordagem de aprendizado de máquina, um classificador para cada emoção baseado no algoritmo Naïve Bayes é treinado usando os textos do *corpus*. Os termos contidos nos textos em português, em inglês ou na combinação de ambos são usados. Os experimentos comparam as duas abordagens, bem como o uso de idiomas específicos ou de combinações destes no aprendizado supervisionado.

3.1. *Corpus* e recursos utilizados

O *corpus* utilizado para os experimentos é composto por estrofes de letras de música do site letras.mus.br, tendo sido coletadas músicas de 3 bandas arbitrariamente escolhidas por possuírem uma vasta quantidade de músicas traduzidas. As letras extraídas estão no idioma inglês (idioma original) e português (tradução feita por usuários do site), como mostra a Figura 1. O *corpus* obtido é paralelo, isto é, os textos obtidos são a tradução um do outro, possuindo a mesma estrutura. Pode-se expandir esse *corpus* para outros idiomas, contanto que seja possível realizar o alinhamento. A principal vantagem deste

tipo de *corpus* é que, por ser paralelo, não é necessário anotar o texto em cada idioma, visto que os textos são os mesmos e a anotação pode ser projetada para as traduções.

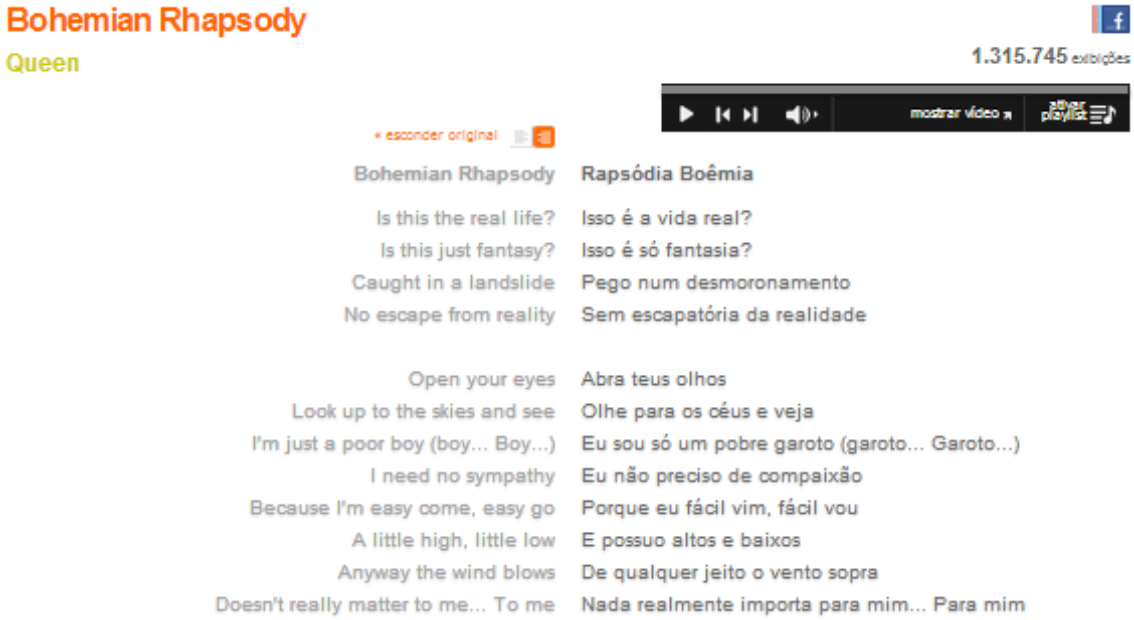


Figura 1. Exemplo de letra de música - retirado do site letras.mus.br.

A maior parte das letras é dividida em estrofes de diferentes tamanhos. Optou-se por realizar a análise em nível de estrofe para facilitar a identificação das emoções, já que podem variar ao longo da música. Assim, esperamos atingir um nível maior de consenso na anotação. As estrofes são utilizadas em seu estado bruto, sem passar por pré-processamento, como corretor gramatical ou remoção de *stopwords*.

Foram anotadas 850 estrofes no idioma português, que classificaram cada uma das 8 emoções presentes no NRC segundo a seguinte escala: "não evoca" (NE), "evoca vagamente" (EV) ou "evoca claramente" (EC). Ao final, é feita a uniformização dos valores das anotações, para validar os classificadores, na qual os casos discordantes recebem a intensidade definida pela maioria. Se não há concordância, a intensidade é convertida pra "evoca vagamente". A Tabela 1 mostra a concordância média entre os anotadores A1, A2 e A3, calculadas sobre todas as 8 emoções. A Figura 2 mostra a distribuição nas estrofes das emoções segundo cada intensidade. Nota-se que as classes são desbalanceadas. Um total de 152 estrofes não evocam nenhuma emoção.

Tabela 1. Concordância média e desvio padrão entre os anotadores.

	A1xA2	**A2xA3**	**A1xA3**	**A1xA2xA3**
Concordância média	0,682	0,703	0,712	0,576
Desvio padrão	0,158	0,171	0,163	0,177

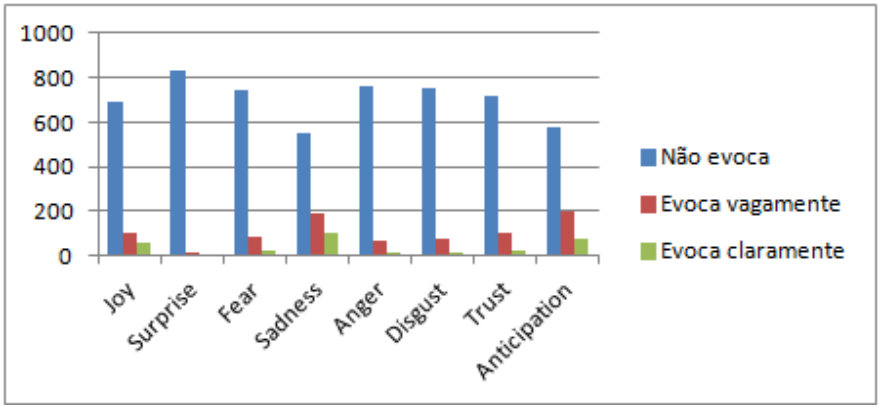


Figura 2. Gráfico de distribuição nas estrofes de emoções por intensidade.

Utilizamos o léxico NRC Emotion Lexicon [Mohammad and Turney 2013], que contém 14000 palavras em inglês rotuladas para polaridade e 8 classes de emoções (Emoções = {*joy*, *sadness*, *anger*, *fear*, *disgust*, *suprise*, *trust*, *anticipation*}). Para cada palavra contida no NRC, há uma indicação (0 ou 1) para cada uma das categorias de emoção e de polaridade (positiva e negativa), conforme mostrado na Figura 3. Um termo pode ter valor 1 para mais de uma categoria (e.g. *lovely*).

love	positive:1	negative:0	anger:0	anticipation:0	disgust:0	fear:0	joy:1	sadness:0	surprise:0	trust:0
lovely	positive:1	negative:0	anger:0	anticipation:1	disgust:0	fear:0	joy:1	sadness:1	surprise:1	trust:1
lovemaking	positive:1	negative:0	anger:0	anticipation:0	disgust:0	fear:0	joy:1	sadness:0	surprise:0	trust:1
lover	positive:1	negative:0	anger:0	anticipation:1	disgust:0	fear:0	joy:1	sadness:0	surprise:0	trust:1
loving	positive:1	negative:0	anger:0	anticipation:0	disgust:0	fear:0	joy:1	sadness:0	surprise:0	trust:1
lower	positive:0	negative:1	anger:0	anticipation:0	disgust:0	fear:0	joy:0	sadness:1	surprise:0	trust:0

Figura 3. Trecho do léxico NRC.

Foi utilizado o classificador Naïve Bayes disponível na ferramenta Weka [Hall et al. 2009], em conjunto com a linguagem Java.

3.2. Classificação

As emoções das estrofes foram classificadas usando dois tipos de classificadores: léxico e aprendizado supervisionado. Em ambos os casos, as emoções identificadas nos textos dependem das palavras usadas. A princípio, cada estrofe recebe valor 0 para cada categoria de emoção; ao final da classificação, ela recebe rótulos de emoção, compostos pelas emoções e suas respectivas intensidades, isto é, *x*NE, *x*EV, *x*EC, onde $x \in$ Emoções.

As Figuras 4 e 5 mostram um esquema simplificado dos classificadores de cada abordagem. Para a abordagem utilizando léxico de sentimentos, as estrofes no idioma português são classificadas usando suas correspondentes em inglês, mesmo idioma do NRC. Cada palavra contida na estrofe em inglês é buscada no dicionário, e se encontrada, a emoção correspondente é incrementada nos valores associados ao termo no dicionário. Ao final da busca, a soma final dos valores de emoção da estrofe é normalizado pelo total de palavras da estrofe, resultando em um valor entre 0 e 1; é associada uma intensidade à emoção conforme o valor obtido: para valores entre 0 e 0,05, a intensidade é "NC"; para valores entre 0,05 e 0,1, "EV"; e valores acima de 0,1 recebem a intensidade "EC". Esses rótulos são então projetados para a respectiva estrofe em português.

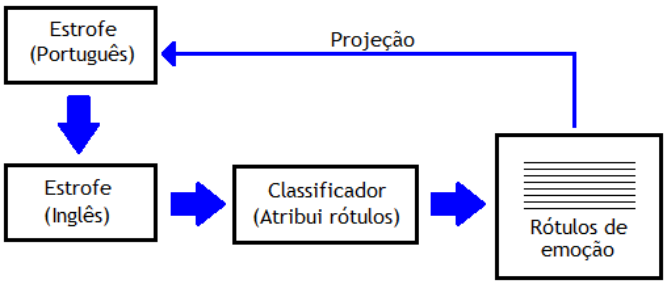


Figura 4. Classificação baseada em léxico de sentimento.

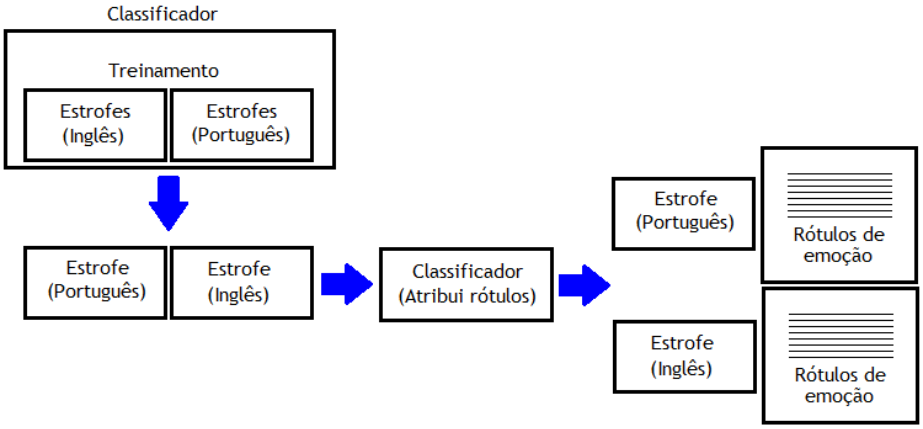


Figura 5. Classificação baseada em aprendizado de máquina.

Para a abordagem baseada em aprendizado de máquina, são construídos 8 classificadores, um para cada emoção, cujas classes são as intensidades de cada emoção (NE, EV, EC). Para o treinamento e validação dos classificadores, foram feitos experimentos com os termos extraídos das estrofes em português, em inglês e da concatenação de ambos. Para preparação dos dados foram testadas várias opções, entre elas: unigramas versus n-gramas; uso ou remoção de *stopwords*; representação de termos binária ou com pesos (frequência e TF-IDF); seleção de atributos. A configuração que trouxe melhores resultados foi o uso de unigramas, representação de peso usando TF-IDF; uso de *stopwords*, e seleção de atributos com *Information Gain*.

Para comparar uso de léxico de sentimento e aprendizado de máquina, são analisadas as estrofes classificadas em português. Já para a comparação entre classificação monolíngue versus multilíngue, são analisadas as estrofes classificadas pelo Naïve Bayes em cada idioma, assim como a combinação deles. Os experimentos são validados utilizando 10-*fold cross-validation*. Os resultados são apresentados a seguir.

3.3. Resultados e Considerações Finais

Devido ao desbalanceamento das classes, os resultados foram analisados utilizando as medidas: precisão, revocação e medida-F ponderadas em função do número de instâncias das classes de cada emoção (*x*NE, *x*EV e *x*EC, onde $x \in$ Emoções). A Figura 6 mostra os resultados, comparando a classificação baseada em léxico e em aprendizado de máquina, esta última usando as estrofes em português.

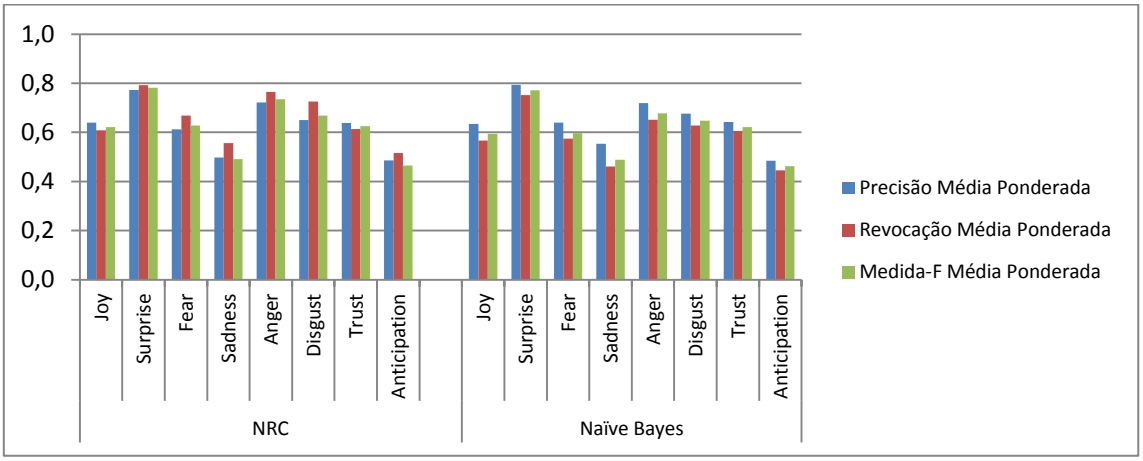


Figura 6. Gráfico de comparação entre as abordagens de léxico de sentimento (uso de NRC) e aprendizado de máquina (uso de Naïve Bayes).

É possível observar que os resultados utilizando Naïve Bayes são muito superiores, analisando precisão, revocação e medida-F há uma diferença estatisticamente significativa segundo o teste-t (p = 2,75^{-37}). O classificador Naïve Bayes não está limitado aos termos contidos no NRC, não é dependente de recursos linguísticos, e permite explorar um pré-processamento nos dados não aplicável à abordagem baseada em léxico. Vale ressaltar que o classificador usando NRC depende de um *corpus* paralelo ou de tradução automática para tratar outros idiomas, diferente do Naïve Bayes, que necessita apenas de textos rotulados para os idiomas tratados. A simplicidade da abordagem, aliada ao fato de não necessitar de recursos específicos, adicionam vantagens ao aprendizado de máquina. Por outro lado, seus resultados são bastante dependentes do tamanho e qualidade do corpus de treinamento. O corpus utilizado, ainda de tamanho pequeno, exigiu um esforço considerável. Esse esforço, contudo, é menor do que o trabalho necessário para a criação de um dicionário de sentimentos.

A Figura 7 mostra os resultados da segunda parte do experimento, no qual comparamos os resultados de classificadores Naïve Bayes testando separadamente os idiomas português, inglês e a combinação dos dois idiomas.

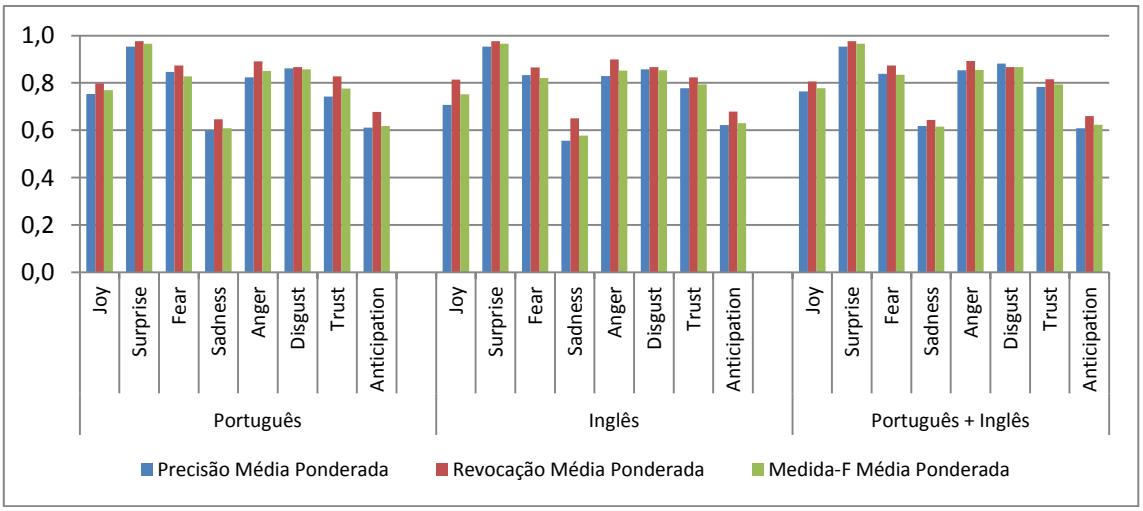


Figura 7. Gráfico de comparação da classificação entre os idiomas português e inglês, usando Naïve Bayes.

Os resultados analisando todas as métricas mostram que a classificação de textos em português são minimamente superiores aos resultados obtidos com o idioma inglês para a maioria das emoções (*joy*, *sadness*, *disgust*, *trust* e *anticipation*), entretanto, não há melhoria estatisticamente significativa segundo teste-t (p = 0,5). A diferença poderia ser maior caso lematizadores fossem usados para ambos os idiomas, visto que o português possui mais flexões, mas este é um recurso escasso para língua portuguesa.

Com a combinação da estrofe original e a tradução, esperava-se que os resultados fossem melhores, visto que a disponibilidade de textos em múltiplos idiomas costuma auxiliar a desambiguação [Banea et al. 2010]. Com efeito, pode-se ver que a revocação da classificação que usa a combinação das estrofes em português e em inglês é maior para todos os casos, enquanto a precisão é melhor para a maioria das emoções, o que confirma que uma análise multilíngue é superior à análise monolíngue. Entretanto, as melhoras não são significativas quando comparadas com os resultados individuais de português e inglês, segundo o teste-t (p = 0,257 e p = 0,055, respectivamente).

Entre as limitações do trabalho está o tratamento de negação. Como os classificadores lidam com cada palavra, pode ser identificada uma emoção que não deveria estar presente devido a uma negação na frase. Entretanto, diferente da polaridade, onde uma negação pode levar o sentimento de positivo/negativo para negativo/positivo, não há como definir emoções contrárias no conjunto analisado. Por exemplo, *joy* e *sadness* podem ser consideradas contrárias, mas é difícil definir as emoções opostas a, por exemplo, *surprise* ou *anticipation*. Por esta razão, optou-se por não tratar negação.

A MOM frequentemente tem seus resultados dependentes da qualidade da tradução automática. Para evitar esta dependência, o *corpus* paralelo foi extraído de um site de músicas brasileiro, onde a princípio as traduções são feitas por usuários do mesmo. Espera-se que sejam superiores, mas não há como garantir a qualidade das traduções obtidas e uma validação manual não é viável, devido ao esforço necessário.

Para cada emoção, o *corpus* é desbalanceado. Cada estrofe possui, em média, duas emoções presentes, geralmente classificadas como "evoca vagamente". Em poucas estrofes a emoção foi anotada "evoca claramente". Isto faz com que o restante das emoções não estejam presentes em cada estrofe, resultando que a melhor representada para todas as emoções é "não evoca". Isso se deve à escolha das músicas não levar em consideração quais emoções estariam presentes.

4. Conclusões e Trabalhos Futuros

Este artigo analisou qual abordagem, entre uso de léxico de sentimentos e aprendizado de máquina, possui melhores resultados para classificação de emoções em textos multilíngues. Os resultados mostram que aprendizado de máquina é superior.

Outro objetivo foi verificar qual idioma, entre inglês e português, possui melhores resultados na classificação de idiomas, usando Naïve Bayes para isso. Os resultados mostram que não há diferença na classificação a partir de textos em português e inglês. Ainda neste ponto, verificou-se que fazer a análise combinando ambas as versões da estrofe produz resultados marginalmente melhores.

Em experimentos futuros, pretendemos investigar se o uso de um lematizador pode melhorar a qualidade da classificação Para isto, é necessário que este recurso esteja disponível para ambos os idiomas. Além disso, aumentaremos a quantidade de idiomas tratados, alinhando traduções das letras de músicas que compõem o *corpus* paralelo. Sobre o *corpus* usado, pretendemos expandi-lo, o que demanda grande esforço manual. Outro ponto a ser investigado é a avaliação de outros algoritmos de aprendizado de máquina, como o SVM, por exemplo.

Agradecimentos

Este trabalho foi parcialmente financiado pelo CNPq projeto 478989/2012/6. A primeira autora recebeu bolsa do CNPQ.

Referências

Asur, S., & Huberman, B. A. (2010, August). Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT),* (Vol. 1, pp. 492-499). IEEE.

Banea, C., Mihalcea, R., Wiebe, J., & Hassan, S. Multilingual subjectivity analysis using machine translation. In *EMNLP* (pp. 127–135), 2008.

Banea, C., Mihalcea, R., & Wiebe, J. (2010, August). Multilingual subjectivity: are more languages better?. In *Proceedings of the 23rd International Conference on Computational Linguistics* (pp. 28-36). Association for Computational Linguistics.

Bollen, J., Mao, H., & Pepe, A. (2011, July). Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. In *ICWSM*.

Ekman, P. An argument for basic emotions. Cognition & Emotion 6.3-4: 169-200, 1992.

Hall, Mark et al. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, v. 11, n. 1, p. 10-18, 2009.

Liu, B. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1), 1–167, 2012

Martinazzo, B., & Paraiso, E. C. (2010). Identificação de Emoções em Notícias Curtas. In *CLEI-Conferência Latino-Americana de Informática* (Vol. 1, pp. 1-10).

Mohammad, S. M. (2012). From once upon a time to happily ever after: Tracking emotions in mail and books. *Decision Support Systems*, *53*(4), 730-741.

Mohammad, S. M. & Turney, P. D. NRC Emotion Lexicon. *NRC Technical Report*, 2013, December.

Narr, S., Hülfenhaus, M. & Albayrak, S. Language-independent twitter sentiment analysis. *KDML*, 2012.

Ortony, A., & Turner, T. J. (1990). What's basic about basic emotions? *Psychological review*, 97(3), 315.

Plutchik, R. (1984). Emotions: A general psychoevolutionary theory. *Approaches to emotion*, *1984*, 197-219.

Santos, A. G. L., Becker, K., & Moreira, V. Um estudo de caso de mineração de emoções em textos multilíngues. *Anais do Brasnam*, Brasília, 2014. 12p.

Tsytsarau, M., & Palpanas, T. Survey on mining subjective data on the web. *Data Mining and Knowledge Discovery*, 24(3), 478–514, 2012.

Tumasjan, A., Sprenger, T. O., Sandner, P. G., & Welpe, I. M. (2010). Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment. *ICWSM*, *10*, 178-185.

paper:76

oLinDa: Uma Abordagem para Decomposição de Consultas em Federações de Dados Interligados

Danusa R. B. da Cunha[1], **Bernadette Farias Lóscio**[1]

[1]Centro de Informática
Universidade Federal de Pernambuco (UFPE)
Av. Jornalista Anibal Fernandes, s/n - 50.740-560 - Recife - PE - Brasil

{dbrc,bfl}@cin.ufpe.br

Abstract. *Providing integrated access to data distributed over Linked Data federations has become a major research challenge, mainly due to heterogeneity problems. This work proposes a solution for query decomposition over Linked Data Federations available on the Web of Data, i.e., sets of RDF data sources published according to the principles of Linked Data. Our main contribution lies in the definition and implementation of a process for query decomposition, considering that the data sources have structurally distinct ontologies, which describe their schemas. In order to evaluate the proposed approach, a prototype was implemented and some experiments were performed.*

Resumo. *Prover uma visão integrada de dados distribuídos em diferentes conjuntos de dados RDF, interligados entre si e acessíveis por meio de consultas SPARQL tornou-se um grande desafio de pesquisa devido, principalmente, aos problemas de heterogeneidade. Este trabalho define e implementa um processo para decomposição de consultas, considerando que as fontes de dados possuem esquemas estruturalmente distintos, com o objetivo de auxiliar o processo de integração de dados em conjuntos de dados que seguem os padrões Linked Data. Nossa abordagem manipula consultas SPARQL e utiliza mapeamentos heterogêneos com o objetivo de tratar aspectos referentes às diferenças estruturais entre as ontologias que descrevem os conjuntos de dados a serem integrados. Para avaliar a abordagem proposta, um protótipo foi implementado e alguns experimentos foram realizados com fontes de dados que seguem os princípios Linked Data.*

1. Introdução

Um dos principais objetivos de um sistema de integração de dados é facilitar a recuperação e o compartilhamento de dados heterogêneos e distribuídos [Halevy et al. 2006]. Muitas soluções de integração de dados foram propostas no decorrer dos anos, como a arquitetura de mediadores, os PDMS (Peer Data Management Systems) e sistemas de integração de dados pay-as-you-go [Sung et al. 2005, Halevy et al. 2006, Franklin et al. 2005]. Por outro lado, a adoção das tecnologias da Web Semântica, como RDF (*Resource Description Framework*) e OWL (*Ontology Web Language*), combinado com os princípios de *Linked Data* [Bizer et al. 2009], têm motivado o desenvolvimento de novas soluções para integração de dados com o objetivo de superar alguns problemas específicos encontrados neste contexto.

No cenário da Web de Dados, onde as fontes de dados são conjuntos de dados RDF que podem ser acessados a partir de consultas SPARQL (*SPARQL Protocol and RDF Query Language*), os sistemas de consultas federadas destacam-se como uma solução de integração de dados [Rakhmawati et al. 2013]. Nestes sistemas, o *federator* desempenha um papel similar a um mediador, sendo responsável por decompor uma consulta em uma ou mais subconsultas, direcionar as subconsultas às múltiplas fontes de dados e, finalmente, integrar os resultados das subconsultas. Além disso, o *federator* pode possuir um esquema que provê o vocabulário necessário para a construção das consultas.

Um dos principais desafios a serem solucionados quando se deseja oferecer uma visão integrada de dados distribuídos em múltiplas fontes de dados diz respeito à decomposição de consultas. Este desafio torna-se relevante devido, principalmente, à heterogeneidade dos esquemas das fontes de dados a serem integradas (esquemas locais). Além disso, considerando a presença de um esquema global para descrever a visão integrada oferecida pelo sistema de integração, torna-se necessária a especificação de mapeamentos entre o esquema global e os esquemas locais. Estes mapeamentos especificam como as instâncias que representam a visão integrada podem ser obtidas a partir das instâncias das fontes de dados locais.

Este trabalho propõe uma abordagem, denominada *oLinDa* (*Query Decomposition **o**ver **Lin**ked **Da**ta Federation*), para decomposição de consultas em sistemas de consultas federadas, que busca oferecer uma visão integrada de dados distribuídos em múltiplas fontes de dados que seguem os princípios de Linked Data. Como principais diferenciais da abordagem proposta, destacam-se: (i) focamos no problema de decomposição de consultas em federações de dados interligados; (ii) são utilizados tanto mapeamentos homogêneos quanto heterogêneos definidos por [Sacramento et al. 2010] com o objetivo de lidar com as estruturas distintas dos esquema das fontes de dados da federação que, em sua maioria, têm como esquema ontologias distintas; (iii) são consideradas consultas SPARQL que fazem uso dos construtores FILTER e OPTIONAL e (iv) é possível, por meio dos mapeamentos, encontrar informações complementares, ou seja, não é obrigatório cada fonte de dados responder toda a consulta dada como entrada. Caso ela responda apenas partes dela, o resultado será considerado. Para avaliar a abordagem proposta, um protótipo foi implementado e alguns experimentos foram realizados com fontes de dados que seguem os princípios *Linked Data*.

O restante do artigo está organizado como se segue. Na Seção 2 apresentamos a caracterização do problema. Na Seção 3 discorremos sobre algumas definições preliminares para um melhor entendimento deste trabalho. Na Seção 4 descrevemos a abordagem proposta para decomposição de consultas. Na Seção 5 apresentamos os principais trabalhos relacionados. Na Seção 6 descrevemos o protótipo desenvolvido, discorrendo sobre suas principais funcionalidades, aspectos de implementação e resultados experimentais. Finalmente, na Seção 7 apresentamos algumas considerações sobre a pesquisa, as conclusões e os trabalhos futuros.

2. Caracterização do Problema

Com o intuito de ilustrar o problema abordado neste trabalho, nesta seção apresentamos um exemplo de um cenário típico de integração de dados a partir de múltiplas fontes de dados que seguem os princípios *Linked Data*. As fontes são do domínio de dados

bibliográficos e armazenam dados sobre artigos, autores, conferências, periódicos e livros da área de Ciência da Computação. Esses dados podem ser encontrados em diversas fontes, porém, foram escolhidas as fontes de dados *DBLP*[1] e *Kisti*[2]. Por fim, uma fonte de dados com temas variados, *DBpedia*[3], foi escolhida para dar um maior suporte às informações contidas nas duas primeiras fontes, como descrição pormenorizada sobre autores.

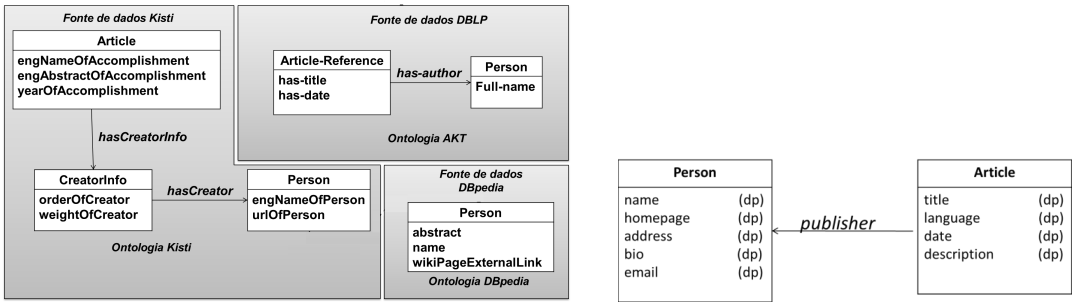


Figura 1. (a) Fontes de dados *Kisti*, *DBLP* e *DBpedia* e (b) Ontologia *SWRC*.

Cada uma das fontes de dados mencionadas está associada a uma ontologia que descreve o vocabulário da fonte de dados. A fonte de dados DBLP é descrita pela ontologia AKT[4], uma ontologia criada para descrever entidades relacionadas com a comunidade de Ciência da Computação. As fontes de dados *Kisti* e *DBpedia* utilizam suas próprias ontologias, *Kisti* e *DBpedia*, respectivamente. A Figura 1a apresenta parte das ontologias das fontes de dados. A ontologia *SWRC*[5] (SemanticWeb for Research Communities), apresentada parcialmente na Figura 1b, é uma ontologia que descreve entidades relacionadas com a comunidade de pesquisadores tais como pessoas, organizações, publicações e seus relacionamentos. Esta ontologia desempenha o papel do esquema do *federator*, ou seja, procura fornecer uma visão integrada dos dados disponíveis nas diferentes fontes locais.

Suponha, então, que um usuário deseja obter a resposta para a seguinte consulta *Q* (*Consulta exemplo referenciada ao longo do texto*): *Recupere os títulos de artigos publicados pelo autor Alon Y. Halevy que contenham a palavra Data Integration, bem como um resumo dos mesmos. Além disso, caso exista, recupere o ano de publicação desses artigos. Retorne também uma pequena biografia do autor.*

Considere que o usuário deseja obter a resposta para a consulta Q sem ter que se preocupar sobre a localização dos dados e com a heterogeneidade existente entre as ontologias locais. Logo, para encontrar uma resposta para a consulta *Q*, é necessário identificar em quais fontes de dados a resposta poderá ser encontrada, seja de maneira total ou parcial. Para isso, é preciso realizar a decomposição da consulta *Q* em várias subconsultas que serão submetidas às fontes de dados capazes de contribuir com a res-

[1]http://dblp.rkbexplorer.com/sparql/
[2]http://kisti.rkbexplorer.com/sparql/
[3]dbpedia.org
[4]http://swl.slis.indiana.edu/repository/owl/aktportal.owl
[5]http://ontoware.org/swrc/

posta à pergunta. Durante a decomposição de consultas, dois principais desafios precisam ser enfrentados: (i) heterogeneidade entre os esquemas, neste caso, entre as ontologias e (ii) a recuperação de informações parciais. No primeiro caso, diferentes representações podem ser utilizadas para descrever o mesmo conceito do mundo real. No segundo caso, as fontes de dados podem responder apenas parte da consulta Q. Para ilustrar o caso (i), considere a propriedade de objeto *s:publisher* a qual relaciona um artigo com seu(s) autor(es). Observe que na ontologia *Kisti*, a associação entre artigo e autor não é feita de maneira simples e direta, pois um artigo é primeiro relacionado com uma instância da classe *k:CreatorInfo* e somente quando é recuperada a instância desta classe, é que, então, pode-se recuperar a instância de autor por meio da propriedade de objeto *k:hasCreator*. Isto, por outro lado, é feita de forma direta na ontologia *AKT*. Notamos, aqui, que isto acontece devido às ontologias possuírem estruturas diferentes caracterizando uma heterogeneidade estrutural. Já para o caso (ii), suponha que nas fontes de dados *Kisti* e *DBLP* é possível encontrar as informações desejadas sobre os artigos publicados pelo autor Alon Y. Halevy. Porém, dados sobre o mesmo só poderão ser recuperados na fonte de dados *DBpedia*, por meio das propriedades *d:abstract* e *d:wikiPageExternalLink*. Neste caso, as subconsultas deverão ser criadas a partir das informações que cada fonte de dados pode fornecer. Diante de tais desafios, nota-se que são necessários mecanismos eficientes para a decomposição de consultas com o objetivo de superar os problemas de heterogeneidade e permitir que fontes de dados possam oferecer respostas parciais ou incompletas.

Neste contexto, o processo de decomposição de consultas proposto neste trabalho visa solucionar o problema descrito a seguir: *Seja uma ontologia de domínio O_D que representa o esquema do federator; $S = \{S_1, \ldots, S_n\}$ um conjunto de fontes de dados RDF; $O_L = \{O_{S_1}, \ldots, O_{S_n}\}$ um conjunto de ontologias, chamadas de ontologias locais, onde cada O_{S_i} descreve o vocabulário da fonte de dados S_i e um conjunto de mapeamentos M entre O_D e cada ontologia local O_{S_i}*. O problema de decomposição de consultas consiste em como decompor uma consulta global Q, definida em termos de O_D, em uma ou mais consultas $\{Q_1, \ldots, Q_k\}$ a serem submetidas em uma ou mais fontes de dados de S, considerando que as ontologias que descrevem os esquemas podem possuir estruturas distintas.

3. Definições preliminares

No processo de decomposição de uma consulta é necessário caracterizar os mapeamentos entre as ontologias que descrevem as fontes de dados e o esquema do *federator*. Na nossa abordagem, os mapeamentos são descritos por regras definidas em Datalog e são obtidos conforme a estratégia descrita em [Leme et al. 2009, Sacramento et al. 2010]. Tal estratégia divide-se em duas etapas principais: (i) *matching contextualizado de vocabulários*, que define um modelo de *matching* para representação das correspondências entre as entidades de ontologias distintas, e (ii) geração das regras de mapeamento, que induz a um conjunto de regras a partir do modelo obtido na etapa (i).

O processo de *matching* proposto em [Leme et al. 2009] é dito contextualizado porque as classes que compõem o domínio das propriedades são representadas explicitamente nas correspondências resultantes do processo de *matching*. Esta representação explícita das classes é chamada de contexto. A fim de tratar alguns tipos de heterogeneidade, foi necessário estender o *matching* contextualizado de vocabulários com uma nova definição para representar correspondências que tratam caminhos de propriedades.

A definição de caminho adotada neste trabalho foi inspirada na definição de caminho proposta em [Vidal et al. 2001] e encontra-se a seguir:

Definição 1 *Caminho de Propriedades: Dado um conjunto de propriedades p_1, ..., p_n de uma ontologia O, μ é um caminho, $\mu = p_1, \ldots, p_n$ se: (i) cada propriedade p_i, $1 \leq i \leq n$, é uma propriedade da ontologia O e (ii) o range de p_i é igual ao domain de p_{i+1}, $\forall\ p_i$ tal que $1 \leq i \leq n$.*

Desta forma temos que: seja O_L uma ontologia que descreve o esquema de uma fonte de dados, chamada de ontologia local, e O_D uma ontologia de domínio, tal que V_L e V_D representam seus respectivos vocabulários. Sejam ainda C_L e C_D os conjuntos de classes, e P_L e P_D os conjuntos de propriedades em V_L e V_D, respectivamente. O conceito de *matching contextualizado* de caminho entre O_L e O_D é um *matching* cujas correspondências obtidas podem ser representadas através de um conjunto finito S de quádruplas (v_1, e_1, v_2, e_2), tal que:

- se $v_1 \in P_D$ e v_2 é um caminho $\mu = p_1. \ldots . p_n$, onde cada propriedade $p_i \in P_L$, $1 \leq i \leq n$, então e_1 e e_2 são classes em C_L e C_D, as quais devem ser subclasses dos *domains*, os próprios *domains* ou os *domains* restritos das propriedades v_1 e p_n, respectivamente. Se v_1 é uma propriedade de dados ou de objeto então p_n é uma propriedade de dados ou de objeto, respectivamente, e o (*domain*) de v_1 possui uma correspondência com (*domain*) de p_1.

Como exemplo de quádrupla obtido do *matching* realizado entre a ontologia de domínio O_D e a ontologia local da fonte de dados *DBPL*, O_{dblp}, temos: (*s:title*, *s:Article*, *a:has-title*, *a:Article-Reference*). Esta quádrupla indica que as propriedades *s:title* e *a:has-title* são correspondentes devido aos seus respectivos contextos, ou seja, às suas classes *domain s:Article* e *a:Article-Reference*, respectivamente, também possuírem uma correspondência, neste caso, (*s:Article*, $\top$, *a:Article-Reference*, $\top$), onde $\top$, indica que não há restrições sobre as respectivas classes. Outro exemplo de quádrupla obtido como resultado do *matching* é: (*s:publisher*, *s:Article*, *k:hasCreatorInfo.k:hasCreator*, *k:Accomplishment*). Aqui temos um exemplo do *matching* que é realizado quando v_2 é um caminho de propriedades. No caso, na ontologia O_D a propriedade de objeto *s:publisher* é correspondente ao caminho v_2 = *k:hasCreatorInfo.k:hasCreator*. Assim sendo, *s:publisher* corresponde ao caminho *k:hasCreatorInfo.k:hasCreator* devido a classes *domain* de *s:publisher*, *s:Article*, ser corresponde a classe *domain* da propriedade *k:hasCreatorInfo* do caminho *k:hasCreatorInfo.k:hasCreator* que é *k:Accomplishment*.

Após a criação das correspondências entre os termos das ontologias é necessário gerar as regras de mapeamentos. De maneira geral, as regras de mapeamentos seguem a forma $\psi(w) \leftarrow \phi_1(t_1) \wedge \ldots \wedge \phi_n(t_n)$, onde $\phi_1(t_1) \wedge \ldots \wedge \phi_n(t_n$, chamado de corpo da regra, é um átomo ou uma conjunção de átomos e cada $\phi_i(t_i)$ é uma classe ou uma propriedade pertencente a uma das ontologias locais O_{S_i}. $\psi(w)$, chamado de cabeça da regra, é um átomo, onde $\psi(w)$ pode ser uma classe ou uma propriedade na ontologia de domínio O_D e w é uma sequência de termos. A Tabela 1 apresenta alguns exemplos de regras de mapeamentos entre as ontologias *SWRC* e *AKT*.

4. Visão Geral da Abordagem oLinDa

Nesta seção, explicamos, brevemente, as principais atividades do processo de decomposição de consultas proposto neste trabalho. O processo possui três atividades

Tabela 1. Exemplo de Regras de Mapeamentos

Regra de Mapeamento entre as ontologias SWRC, AKT, Kisti e DBpedia
s:Article(ar) ← a:Article-Reference(ar); k:Accomplishment(ar)
s:title(ar,t) ← a:has-title(ar,t), a:Article-Reference(ar); k:engNameOfAccomplishment(ar,t), k:Accomplishment(ar)
s:publisher(ar,pb) ← a:has-author(ar,pb), a:Article-Reference(ar); k:hasCreatorInfo(ar,z).k:hasCreator(z,pb), k:Accomplishment(ar)

descritas a seguir: (i)**Atividade 1** - Geração do Grafo de Consulta: Ao receber uma consulta *Q*, na linguagem SPARQL, como entrada, um PARSER é responsável por converter *Q* para um grafo de consultas G_Q; (ii) **Atividade 2** - Geração dos grafos das subconsultas: Uma vez que o grafo de consultas G_Q foi criado, a Atividade 2 consiste em verificar, no conjunto de mapeamentos, se existem regras de mapeamentos que possam ser utilizadas para decompor G_Q em um ou mais grafos $G_{Q_{S_1}}, \ldots, G_{Q_{S_n}}$, onde cada $G_{Q_{S_i}}$ corresponde a um grafo de consulta para a fonte de dados S_i e (iii) **Atividade 3** - Geração das subconsultas em SPARQL: A última atividade consiste em transformar cada grafo gerado na Atividade 2 em subconsultas SPARQL $\{G_{Q_{S_1}}, \ldots, G_{Q_{S_n}}\}$. Salienta-se que estas consultas estão de acordo com o respectivo esquema da fonte S_i.

Um Grafo de Consultas é um par ordenado $G_Q = (V, E)$, onde *V* é um conjunto de vértices, constituído por sujeitos (*s*) e objetos (*o*) e *E* representa um conjunto de predicados (*p*), de cada padrão de tripla $TP \in BGP$, de uma consulta SPARQL *Q* [Bondy 1976].

Para ilustrar nossa abordagem, considere que o conjunto de fontes de dados S_i = *DBLP, Kisti, DBpedia* são as que compõe a federação de dados interligados. Além disso, considere o conjunto de mapeamentos *M* resumidamente ilustrado na Tabela 1. Suponha uma consulta *Q*, submetida de acordo com o vocabulário da ontologia de domínio O_D: *Recupere os títulos e os resumos dos artigos publicados pelo autor Alon Y. Halevy que possuem a palavra "Data Integration"no título. Além disso, caso exista, recupere o ano desses artigos. Retorne também uma pequena biografia do autor. Retorne também uma pequena biografia do autor.* A definição em SPARQL da consulta *Q* é representada na Tabela 2.

Tabela 2. Consulta SPARQL Q.

```
PREFIX s:http://swrc.ontoware.org/ontology
SELECT ?title ?year ?abstract ?bio
WHERE{
    ?article rdf:type s:Article; s:abstract ?abstract; s:title ?title; s:publisher ?author .
    ?author s:description ?bio; s:name "Alon Y. Halevy".
    FILTER regex{?title, "Data Integration", "i"}
    OPTIONAL {?article s:year ?year }
}
```

Ao recebermos como entrada a consulta SPARQL *Q*, o primeiro passo é transformá-la em um grafo de consultas G_Q. Para isso, cada sujeito e objeto, neste caso, representados por variáveis, são transformados em vértices e cada predicado é transformado em aresta. Se o padrão de tripla é opcional, ou seja, é do tipo *OPTIONAL TP*, então,

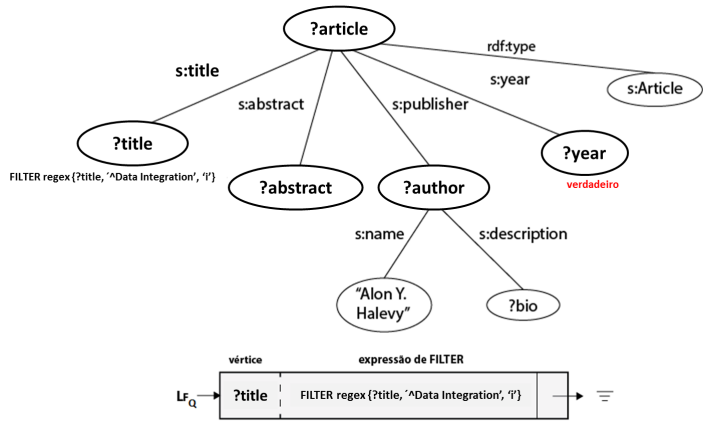


Figura 2. Grafo de Consulta e Lista de Filtro

TP sofre uma pequena modificação, passando a ter uma anotação sobre o nó que recebe o valor do objeto, ficando com a seguinte representação: *TP(s, p , <o, op>)*, onde *op* é uma variável booleana que possui valor verdadeiro. Após todo este procedimento, é feita uma análise da consulta *Q* a fim de identificar a presença do construtor FILTER. Para capturar os construtores FILTER do *BGP* de *Q* foi necessário representar este construtor por meio de uma estrutura constituída por dois atributos: uma expressão e um conjunto de variáveis manipuladas ou mencionadas no atributo expressão. Basicamente, percorremos a expressão do construtor FILTER de onde são extraídas a expressão em si e as variáveis que são manipuladas ou mencionadas nela. Como artefato final deste procedimento, é retornada uma lista de filtros que contém todos os construtores FILTER encontrados no *BGP*. Para exemplificar como os construtores FILTER são extraídos do BGP e armazenados em LFQ, considere o filtro aplicado na consulta Q: FILTER regex{?titulo, "Data Integration", "i"} tem-se que:

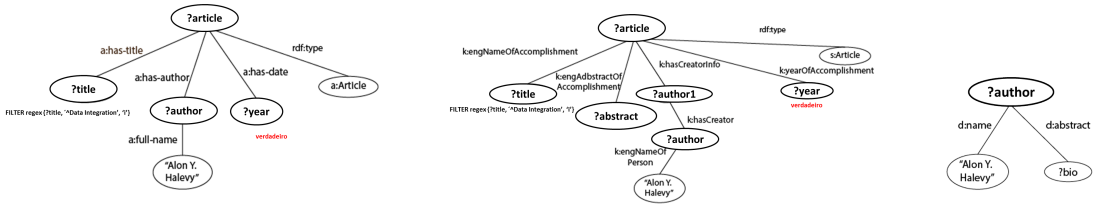
Expressão: regex{?title, "Data Integration", "i"} e **Variável:** ?title

Tanto a expressão quanto a variável são armazenadas em um nó de uma lista L_{F_Q} conforme ilustrado na Figura 2.

Uma vez que o grafo de consultas G_Q foi criado, a Atividade 2 consiste em verificar, no conjunto de mapeamentos *M*, se existe alguma regra de mapeamento que possa ser utilizada para decompor G_Q em um ou mais grafos $G_{Q_{S_1}}$, . . ., G_{Q_S}, onde cada $G_{Q_{S_i}}$ corresponde a um grafo de consulta para a fonte de dados S_i. De maneira geral, para cada fonte de dados S_i é gerado um novo grafo vinculado a esta fonte de dados. Por exemplo, a Figura 3 apresenta os grafos de consultas gerados para cada uma das fontes de dados.

Como é possível observar, somente o padrão de tripla que faz referência a autor foi mapeado para o grafo da fonte de dados *DBpedia* visto que apenas esta fonte possui informações sobre autores. Note que para capturar as heterogeneidades estruturais entre a ontologia *SWRC* e a ontologia *Kisti*, o padrão de tripla *<?article s:author ?author>* que faz referência à propriedade *s:publisher* foi traduzido em dois padrões de triplas: *<?article kisti:hasCreatorInfo ?author1>* e *<?author1 kisti:hasCreator ?author>*. Esta transformação ocorre devido ao mapeamento apresentado na linha 3 da Tabela 1.

Com os grafos gerados para cada fonte de dados é possível criar consultas



(a) (b) (c)

Figura 3. Grafos de Consulta para as fontes de dados (a) DBLP, (b) Kisti e (c) DBpedia.

SPARQL para cada um deles. Para realizar esta atividade, cada nó e cada aresta dos grafos de consultas são percorridos e transformados em padrões de triplas a fim de dar origem aos BGPs das consultas já levando em consideração se um objeto possui alguma anotação. Além disso, a lista de filtros LFQ é percorrida a fim de aplicar os filtros nos devidos grafos de consultas. As consultas geradas para cada grafo de consultas obtido são apresentadas na Figura 4.

Consulta DBLP	Consulta Kisti	Consults DBpedia
SELECT ?title ?year WHERE { ?article rdf:type a:Article-Reference . ?article a:has-title ?title . ?article a:has-author ?author . ?author a:full-name "Alon Y. Halevy". OPTIONAL {?article a:has-date ?year.} FILTER regex {?title , "Data Integration", "i"} }	SELECT ?title ?resumo ?ano WHERE { ?article rdf:type a:Article . ?article k:engNameOfAccomplishment ?title . ?article k:engAbstractOfAccomplishment ?abstract ?article k:hasCreatorInfo ?author1 . ?author1 k:hasCreator ?author . ?author k:engNameOfPerson "Alon Y. Halevy". OPTIONAL {?article k:yearOfAccomplishment ?year .} FILTER regex {?title , "Data Integration", "i"} }	SELECT ?bio WHERE { ?author d:name "Alon Y. Halevy". ?author d:abstract ?bio . }

Figura 4. Subconsultas Decompostas

É importante deixar claro que o foco deste trabalho não consiste em tratar aspectos de estratégias para execução, recuperação, composição dos resultados e otimização de consultas em SPARQL.

5. Trabalhos Relacionados

Nesta seção, apresentamos brevemente os trabalhos relacionados com nossa abordagem, neste caso, os propostos em [Correndo et al. 2010, Lee et al. 2010, Makris et al. 2012]. As abordagens [Correndo et al. 2010] e [Makris et al. 2012] lidam diretamente com SPARQL. No entanto, ambos não tratam expressões de filtro, além de utilizarem apenas mapeamentos homogêneos. Além disso, [Makris et al. 2012] não apresenta a implementação da ideia proposta, enquanto [Correndo et al. 2010] não discute aspectos teóricos, mostrando apenas um estudo de caso. Já a estratégia proposta em [Lee et al. 2010], embora não seja aplicada diretamente sobre fontes de dados no padrão *Linked Data* apresenta um algoritmo de decomposição de consultas utilizando ontologias distintas e distribuídas, cenário semelhante ao encontrado na Web de Dados. A abordagem utiliza mapeamentos homogêneos, expressos em lógica DL-safe e não lida com construtores como FILTER e OPTIONAL.

6. Implementação e Experimentos

Com o objetivo de avaliar a abordagem de decomposição de consultas proposta neste trabalho, foi construído um protótipo, chamado oLinDa, o qual permite que o usuário,

por meio de uma interface gráfica amigável, formule consultas e manipule mapeamentos entre as ontologias.

Para a avaliação experimental, utilizamos o Jena TDB[6] como repositório RDF para armazenar triplas e para realizar consultas sobre estes dados. Para avaliar o protótipo, definimos um conjunto de 20 (vinte) consultas, dentre as quais 16 (dezesseis) foram extraídas do *benchmark SP2Bench*[7]. Além disso, criamos um repositório de dados *RD* que contém os dados das fontes de dados *DBLP*, *Kisti*, *DBpedia* e *Swego* com o objetivos de termos algum parâmetro de referência que indique quais deveriam ser os resultados de cada uma das vinte consultas.

A partir do repositório *RD*, foi possível avaliar se o processo de decomposição proposto neste trabalho está funcionando de maneira adequada. Cada consulta foi submetida tanto sobre o repositório *RD* quanto foi executada no protótipo. Para cada consulta, comparamos os resultados com o objetivo de saber se os resultados obtidos pelas subconsultas criadas pelo nosso processo de decomposição de consultas são corretos e completos, se comparados com os resultados obtidos, caso cada consulta fosse executada no repositório de dados integrados *RD*. Destacamos que a integração dos resultados das subconsultas foi realizada de forma manual, a fim de poder comparar com o resultado do repositório *RD*.

Como resultado, constatamos que a maioria das consultas foram decompostas corretamente e quando foram submetidas tanto em *RD* quanto nas fontes de dados os resultados foram iguais. Aquelas que não obtiveram sucesso na decomposição deriva-se do fato de serem consultas que solicitam propriedades que não estão no escopo da ontologia de domínio e por possuírem, dentro dos construtores OPTIONAL e UNION, mais de um padrão de tripla. Consultas com tais características são decompostas de maneira incorreta, pois ainda estão fora do escopo da abordagem.

7. Conclusões e Trabalhos Futuros

Neste trabalho, apresentamos uma solução para o problema de decomposição de consultas no contexto de integração de dados na Web de Dados. Dentre os principais diferenciais da nossa abordagem, destaca-se o fato de lidar com o problema de decomposição de consultas entre ontologias que apresentam estruturas distintas.

É importante destacar que, para a avaliação da abordagem proposta, implementamos um protótipo e realizamos experimentos considerando diversos tipos de consulta, as quais abrangem: (i) os construtores FILTER e OPTIONAL da linguagem SPARQL e (ii) ontologias heterogêneas. Como resultado desta avaliação, verificou-se que a maioria das consultas propostas foram decompostas de maneira correta por nossa abordagem.

Algumas direções que podem ser exploradas em trabalhos futuros são: aumentar o escopo da gramática da linguagem SPARQL e realizar estudos mais aprofundados sobre estratégias de execução de consultas levando em consideração algoritmos de junção que exploram imprevisibilidade do tempo de resposta dos SPARQL *endpoints*, bem como os que objetivam reduzir a transmissão de dados intermediários para o *federator*.

[6]http://jena.apache.org/documentation/tdb/

[7]http://dbis.informatik.uni-freiburg.de/index.php?project=SP2B/queries.php

Referências

Bizer, C., Heath, T., and Berners-Lee, T. (2009). Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22.

Bondy, J. A. (1976). *Graph Theory With Applications*. Elsevier Science Ltd.

Correndo, G., Salvadores, M., Millard, I., Glaser, H., and Shadbolt, N. (2010). Sparql query rewriting for implementing data integration over linked data. In *Proceedings of the 2010 EDBTICDT Workshops*, EDBT '10, pages 41–411, New York, NY, USA.

Franklin, M., Halevy, A., and Maier, D. (2005). From databases to dataspaces a new abstraction for information management. *SIGMOD Rec.*, 34(4):27–33.

Halevy, A., Rajaraman, A., and Ordille, J. (2006). Data integration the teenage years. In *Proceedings of the 32nd international conference on Very large data bases*, VLDB '06, pages 9–16. VLDB Endowment.

Lee, J., Park, J.-H., Park, M.-J., Chung, C.-W., and Min, J.-K. (2010). An intelligent query processing for distributed ontologies. *J. Syst. Softw.*, 83(1):85–95.

Leme, L. A. P. P., Casanova, M. A., Breitman, K. K., and Furtado, A. L. (2009). Instance-based owl schema matching. In *Enterprise Information Systems*, volume 24, pages 14–26. Springer Berlin Heidelberg.

Makris, K., Bikakis, N., Gioldasis, N., and Christodoulakis, S. (2012). Sparql-rw transparent query access over mapped rdf data sources. In *Proceedings of the 15th International Conference on Extending Database Technology*, EDBT '12, pages 610–613, New York, NY, USA. ACM.

Rakhmawati, N. A., Umbrich, J., Karnstedt, M., Hasnain, A., and Hausenblas, M. (2013). Querying over federated sparql endpoints - a state of the art survey. *CoRR*, abs/1306.1723.

Sacramento, E. R., Vidal, V. M. P., de Macêdo, J. A. F., Lóscio, B. F., Lopes, F. L. R., and Casanova, M. A. (2010). Towards automatic generation of application ontologies. *JIDM*, 1(3):535–550.

Sung, L. G. A., Ahmed, N., Blanco, R., Li, H., Soliman, M. A., and Hadaller, D. (2005). A survey of data management in peer-to-peer systems.

Vidal, V. M. P., Lóscio, B. F., and Salgado, A. C. (2001). Using correspondence assertions for specifying the semantics of xml-based mediators. In *Workshop on Information Integration on the Web*, pages 3–11.

paper:73

Principais componentes na ordenação de anúncios: um experimento em ambiente real de publicidade computacional

André Casimiro, Marcos Broinizi, João Eduardo Ferreira

[1] Departamento de Ciências de Computação
Universidade de São Paulo (USP) – São Paulo, SP – Brasil

{casimiro,mbroinizi,jef}@ime.usp.br

Abstract. *Computational Advertising is a multibillion dollar market that promotes the web's development. A major challenge is to simultaneously optimize the interests of users, advertisers and publishers, which depends on the order of ads retrieved. Many research papers explored different factors to solve the problem of ad ordering, such as page context, ad historical performance, maximum cost per click, and user's related data. However, there is a knowledge gap regarding the evaluation of these factors in a real environment. To reduce this gap, we present an experiment of computational advertising in the largest Brazilian news portal. More concretely, in this paper, we evaluated the correlation and how each one of these factors affects system performance. According to our results, these aspects complement each other, and when used together, they are able to optimize the interests of users, advertisers and publishers.*

Resumo. *Publicidade computacional é um mercado multibilionário que proporciona o desenvolvimento da Internet. Um dos grandes desafios é otimizar simultaneamente os interesses dos usuários, anunciantes e publicadores, o que depende da ordem dos anúncios recuperados. Muitos trabalhos exploram diferentes fatores para resolver o problema de ordenação de anúncios, tais como o contexto da página, o desempenho histórico do anúncio, o valor máximo de um clique e os dados do usuário. Entretanto, há uma lacuna de entendimento com relação a avaliação desses fatores em um ambiente real. Para reduzi-la, apresentamos um experimento de publicidade computacional no maior portal brasileiro de notícias. Mais concretamente, avaliamos a correlação e como cada um desses fatores afeta o desempenho do sistema de anúncios. De acordo com nossos resultados esses aspectos são complementares e, quando usados em conjunto, otimizam os interesses dos usuários, anunciantes e publicadores.*

1. Introdução

Na Internet, é impossível não notar a presença de anúncios publicitários, eles estão presentes em sites de busca, portais de informação, notícias, redes sociais e até mesmo blogs pessoais. A exibição de publicidade é um mercado multibilionário [IAB 2013] que estimulou o desenvolvimento da Internet, pois permite que produtores de conteúdo e desenvolvedores de aplicativos rentabilizassem seus negócios. Nesse cenário, os modelos e os contratos de publicidade desenvolveram-se muito nos últimos anos. Iniciando com a alocação de anúncios visuais (*banners*) por um valor fixo, seguida pela busca patrocinada em máquinas de busca e, posteriormente, pelas redes de anúncios (*ad networks*) expandindo a veiculação de conteúdo publicitário a qualquer página da Internet.

Os sistemas de publicidade computacional possuem um repositório de anúncios e de anunciantes. Cada requisição feita por um navegador para exibição do conteúdo de um site gera também uma requisição ao sistema de anúncios. Os anúncios são recuperados, ordenados e devolvidos para exibição. Essa ordenação depende de diversos fatores e visa balancear os interesses das partes envolvidas, a saber: o usuário, o anunciante e o publicador (que é o site no qual os anúncios são veiculados). O equilíbrio desses interesses é essencial para a continuidade do sistema, pois a participação de cada um depende da atuação dos outros. Por exemplo, uma ordenação que favorecesse o interesse financeiro do publicador aumentaria o lucro mas, a longo prazo, provavelmente menos usuários se interessariam pelos anúncios veiculados, uma vez que os mesmos seriam pouco relacionados ao conteúdo da página, o que implicaria na degradação da efetividade do sistema de publicidade. A definição de uma ordenação que equilibre os interesses se torna ainda mais desafiadora ao considerar o fato de que a quantidade de espaço publicitário em cada página comporta poucos anúncios, enquanto existem milhares disponíveis para exibição.

A seleção e ordenação de anúncios é frequentemente modelada como um problema de recuperação de informação [Wang et al. 2009] e leilões de segundo preço [Edelman et al. 2005]. Alguns trabalhos têm utilizado técnicas de aprendizado de máquina para resolver esse problema, indo desde regressões lineares [Bendersky et al. 2010] e logísticas [Agarwal et al. 2009] até técnicas mais avançadas como programação genética [Lacerda et al. 2006] e SVM [Broder et al. 2007]. Bons resultados tem sido alcançados com esses modelos, por outro lado seu uso enfrenta algumas restrições quando aplicados a um sistema real. Uma das principais restrições é a manutenção do sistema conforme o ambiente evolui, devido à sua complexidade o desenvolvimento e implantação de novos modelos pode não acompanhar as constantes mudanças a que o sistema está submetido.

Os sistemas de anúncios têm restrições de escala da ordem de bilhões de requisições por hora e milhões de usuários únicos por mês. Para satisfazê-las, são construídos semelhantes a sistemas de busca, realizando ações como *crawling* da página destino, indexação dos anúncios cadastrados, recuperação, ordenação e exibição dos anúncios recuperados. Nosso trabalho é baseado em um sistema de anúncios que neste modelo. Ele atende às restrições de escala e possui uma estratégia de mercado para ordenação de anúncios que utiliza quatro variáveis principais: a aposta, ou o valor máximo pago por um anunciante por um clique em seu anúncio (BID), a taxa histórica de clique do anúncio (CTR, ou *Click Through Rate*), o *score* de relevância com o contexto da página (CTX) e o *score* de relevância com o usuário (BTU, ou *behavioral targeted user*). Podemos descrever os interesses de cada participante em termos dessas variáveis como segue:

- o usuário acessa a Internet com a intenção de obter informações (CTX), esperando ter uma boa experiência pessoal (BTU) e tem potencial de adquirir algo (CTR);
- o anunciante disponibiliza informação sobre um bem ou serviço que deseja oferecer a um usuário e possui recursos para investir em publicidade, esperando receber acessos e vendas (ou conversões) advindos dessa publicidade (CTR);
- o publicador exerce papel de intermediário, possui alta audiência e deseja aumentar a receita exibindo publicidade (BID) e a audiência atraindo usuários (CTR).

No melhor de nossa pesquisa, nenhum estudo foi ainda conduzido para analisar a importância e a contribuição dessas variáveis individualmente. Nosso trabalho traz na

seção 2 uma descrição dos principais trabalhos relacionados ao desenvolvimento da publicidade computacional nos últimos anos. Para análise das variáveis, foram realizados dois experimentos no ambiente de produção do maior portal de conteúdo da Internet brasileira. A descrição do ambiente é feita na seção 3, seguida pela descrição dos experimentos e seus resultados na seção 4. Por fim, as conclusões e trabalhos futuros estão na seção 5.

2. Trabalhos relacionados

A história da publicidade computacional nos remete a própria história da Internet. Conforme a Internet foi crescendo e se consolidando como parte integrante da vida das pessoas, maior a atração de se colocar conteúdo publicitário em suas páginas. Logo, naturalmente, quanto mais recursos foram investidos, maior o desenvolvimento da área. Ao longo dos anos, surgiram novos modelos e formatos de publicidade, enriquecendo a experiência do usuário e satisfazendo melhor suas necessidades de consumo [Oberoi 2013].

O primeiro modelo de publicidade computacional foi fortemente baseado na publicidade tradicional vinda de outras mídias. Nele um anunciante paga por um espaço publicitário fixo em um site que julgue ser relevante para o público que deseja alcançar. Esse modelo é utilizado até hoje e geralmente o custo é fixado por tempo ou por milhares de exibições/impressões, também conhecido como CPM (*cost per mille*). O grande divisor de águas para a publicidade computacional foi o surgimento da busca patrocinada (*sponsored search*), em que usuários de máquinas de busca são expostos a publicidade relacionada à consulta que realizaram. Os anunciantes apostam em palavras e concorrem em um leilão de segundo preço para conseguirem as melhores posições de exibição [Edelman et al. 2005]. Além de CPM, o modelo de pagamento também pode ser por clique no anúncio, que é conhecido como CPC (*cost per click*). Em 1998, a Overture, uma das empresas pioneiras em publicidade computacional que foi comprada pelo Yahoo em 2003, utilizava apenas a variável BID para ordenação dos anúncios. Assim, quem pagava mais aparecia nas melhores posições. O Google, que era conhecida por fornecer os resultados de busca mais relevantes, usou a mesma estratégia para ordenação dos anúncios e acrescentou uma variável de qualidade, introduzindo o conceito de recomendação de anúncios. Em fevereiro de 2002, essa variável nada mais era do que o CTR do anúncio, hoje certamente a qualidade inclui diversos outros fatores [Google 2002].

Há uma generalização teórica dessa variável de qualidade que é usada para o problema de ordenação de anúncios e que modela o equilíbrio de interesses que citamos anteriormente. Podemos definir a fórmula de ordenação de anúncios apenas em função de dois valores, a aposta do anunciante (BID) e a probabilidade de se atingir o objetivo esperado com a veiculação do anúncio.

$$rank = BID * \Pr(sucesso)$$

No modelo de CPM, o objetivo é simplesmente a impressão do anúncio na tela. Como não há risco de o anúncio não ser exibido, temos que $\Pr(impr) = 1$ e logo:

$$rank_{CPM} = BID * \Pr(impressao) = BID$$

Para o modelo CPC, a probabilidade de um usuário clicar em um anúncio depende de diversos fatores e sua fórmula de rank seria então:

$$rank_{CPC} = BID * \Pr(clique)$$

Há ainda o modelo CPA (*cost per action*), em que o anunciante só paga quando o usuário, após clicar no anúncio, faz determinada ação em seu site, como por exemplo comprar um produto ou preencher um formulário. Podemos representar sua fórmula como:

$$rank_{CPA} = BID * \Pr(clique) * \Pr(acao)$$

A taxa de usuários que clica em um anúncio exibido geralmente é menor do que 1% e a proporção desses que também realizam uma ação (e.g. compram um produto) é menor ainda. Como a escala da web é de milhões de usuários, esse funil de conversão ainda é atrativo para os anunciantes. O trabalho de Bagherjeiran et al. explora uma abordagem de otimização para o modelo de CPA [Bagherjeiran et al. 2010]. Há inúmeros aspectos que podem ser considerados para estimar $\Pr(clique)$ e $\Pr(acao)$ com relação ao usuário e o que o influencia em seu engajamento com conteúdo publicitário. Algumas das questões que buscaram ser analisadas são: o anúncio é relevante com o interesse do usuário naquele momento? [Agarwal et al. 2009] A exibição excessiva de anúncios irrita o usuário? [Broder et al. 2008] O aspecto visual do anúncio está adequado? [Azimi et al. 2012]

As páginas de busca são bons pontos de partida para navegação na Internet, mas o tempo que o usuário gasta navegando nelas é pequeno se comparado ao tempo gasto consumindo conteúdo na página alvo de sua busca. Para suprir essa demanda de exibição de publicidade surgiu o que chamamos de publicidade contextualizada (*contextual advertising*), que é a exibição de anúncios relacionados em qualquer página da Internet. Uma solução em escala para esse problema foram as redes de anúncios (*ad networks*) que agregaram em uma única plataforma anunciantes e publicadores. Qualquer anunciante pode cadastrar um anúncio e qualquer publicador pode integrar seu site para exibição de publicidade. Quando um usuário requisita uma página do site do publicador, os anúncios são requisitados ao servidor de anúncios e inseridos na área reservada a publicidade.

O problema de publicidade contextualizada é uma generalização do problema de busca patrocinada no sentido que extende o desafio de contextualização dos anúncios a qualquer página da Internet, não somente a resultados de buscas. Além do significativo aumento de tráfego, a maior dificuldade está em estimar $\Pr(clique)$ e $\Pr(acao)$ com precisão. Na busca patrocinada temos uma informação precisa do interesse do usuário naquele instante: a consulta que acabou de realizar. Em publicidade contextualizada não temos isto, precisamos inferir tal informação a partir do contexto, como o conteúdo da página e o comportamento do usuário. A variável CTX que apresentamos é uma forma de capturar a relevância de um anúncio com relação ao conteúdo da página e é obtida no processo de recuperação de anúncio. As redes de anúncios são estruturadas de forma semelhantes às máquinas de busca, fazem *crawling* das páginas dos publicadores e indexação dos anúncios em um índice invertido. Quando um usuário acessa a página de um publicador, é feita uma requisição ao sistema de anúncios identificada pela url da página e uma consulta é feita ao índice de anúncios utilizando palavras extraídas da página pelo crawler. O CTX é o *score* obtido na recuperação dos anúncios. Os trabalhos de Agarwal et al. mostram como utilizar o CTX para estimar $\Pr(clique)$ [Agarwal et al. 2009] e também para obter maior eficiência computacional e precisão semântica [Agarwal and Gurevich 2012]. O trabalho de Bendersky melhora eficiência da recuperação dos anúncios propondo um modelo de dados organizado por campanhas publicitárias [Bendersky et al. 2010] e o trabalho de Ribeiro-Neto usa métodos de expansão de consulta para evitar falta de correspondência entre o vocabulário das páginas e dos anúncios [Ribeiro-Neto et al. 2005].

Uma tendência mais recente da área de publicidade computacional tem sido a de melhorar seu conhecimento sobre o usuário que está navegando, não de forma generalista, mas individualizada. Conhecer o perfil do usuário (sexo, faixa etária, região geográfica, interesses pessoais, etc...) permite que a recomendação de publicidade seja mais precisa, fornecendo ao usuário anúncios de seu interesse e melhorando o retorno obtido pelos anunciantes. A obtenção desse tipo de dado pode ser feita por meio de redes sociais, requisitando ao usuário acesso aos dados; ou por rastreamento de comportamento (*behavioral tracking* ou *behavioral targeting*), área que utiliza aprendizado de máquina para analisar padrões de navegação e classificar usuários em perfis. Essa é uma área promissora, que tem obtido melhorias significativas no engajamento do usuário [Tang et al. 2011]. A variável BTU que utilizamos em nossos experimentos representa essa dimensão.

3. Ambiente

Nesta seção descrevemos o sistema de publicidade computacional utilizado para realização de nosso experimento, bem como características da fonte de coleta de dados e o processo de obtenção de cada variável. Nosso experimento foi realizado na rede de anúncios do maior portal de conteúdo da Internet brasileira, responsável por 5 bilhões de requisições de anúncios por mês. Há três tipos principais de publicadores de anúncios, os canais, os parceiros e os afiliados. Os canais são seções do portal de conteúdo específico como notícias, esporte, entretenimento, etc.; os parceiros são sites de nicho que possuem parceria de tráfego de usuários; e os afiliados são quaisquer sites que integraram às suas páginas uma seção para exibição de conteúdo publicitário por meio da rede. Essa gama de sites que exibem os anúncios atrai anunciantes dos mais variados, empresas de pequeno, médio e grande porte de todos os ramos de atuação.

3.1. Sistema de anúncios

O principal objetivo do sistema de anúncios utilizado é exibir a um usuário que está visitando uma das páginas dos publicadores os melhores anúncios dentre os disponíveis. O conceito de melhor anúncio é implementado como critérios de ordenação, ou *ranking*, para selecionar uma pequena quantidade de anúncios dentre o grande volume disponível. Esses critérios usualmente são representados por fórmulas ou funções que utilizam valores mensuráveis, como as quatro váriaveis já descritas. A ordenação busca atender simultaneamente os interesses dos usuários, anunciantes e publicadores. Além disso, o sistema está submetido a fortes restrições de desempenho, como volume de requisições e tempo de resposta.

Para atender ao seu objetivo e respeitar as restrições impostas, o sistema de anúncios em estudo foi concebido em duas fases. A primeira utiliza recuperação de informação para selecionar anúncios em um índice, no qual anúncios são tratados como documentos a serem recuperados. Uma consulta de anúncios é construída com os termos e assuntos da página visitada, os assuntos de interesse do usuário, os dados demográficos, a geo-localização do usuário, entre outros. A consulta é submetida e os anúncios mais relevantes ou similares são recuperados, utilizando uma ordenação tradicional de recuperação de informação no modelo vetorial, com ponderação de frequência e raridade, tf-idf. Essa etapa calcula os valores das variáveis CTX e BTU.

A segunda fase é chamada de estratégia de reordenação e reordena os anúncios selecionados na etapa inicial utilizando uma fórmula que combina as variáveis BID,

CTX, BTU e CTR e exibe os anúncios mais bem posicionados. Exploramos apenas essa segunda fase, testando o desempenho de diferentes estratégias sobre a plataforma de experimentação descrita em mais detalhes por Broinizi e Ferreira [Broinizi et al. 2014].

4. Análise de dados

Nesta seção descrevemos a análise de dados conduzida para determinar quais são os fatores mais relevantes na ordenação de anúncios e como sua combinação pode contribuir para os interesses do usuário, do anunciante e do publicador. Realizamos dois experimentos em ambiente de produção para comparação do desempenho de diferentes estratégias de reordenação de anúncios; também, fizemos uma análise de componentes principais para entender o relacionamento entre as variáveis disponíveis: BID, CTR, CTX e BTU.

A avaliação das estratégias com relação à sua utilidade para os participantes foi feita por meio do desempenho em duas métricas chave:

- o CTR (*click through rate*), que é o número de cliques dividido pelo número de impressões, representa os interesses do usuário e do publicador;
- e o ECPM (*estimated cost per mille*), que é o valor financeiro arrecadado a cada mil impressões, representa o interesse do publicador;

Vale notar que gostaríamos de utilizar como métrica específica para o anunciante o ROI (*return over investment*), ou seja o retorno recebido pelo que foi investido em publicidade, mas esse dado pertence ao anunciante e nem sempre está disponível. Entretanto, o CTR também é uma métrica de interesse do anunciante uma vez que indica que o mesmo atraiu a atenção do usuário para seu produto ou site.

4.1. Experimento 1: comparação de estratégias puras

O primeiro experimento avaliou cinco estratégias de reordenação de anúncios: stRND, stBID, stCTR, stCTX e stBTU. A estratégia stRND é uma reordenação aleatória dos anúncios e será utilizada como *baseline*. Todas as outras estratégias representam uma ordenação pura pela variável que as descreve, ou seja, stBID reordena os anúncios apenas pela variável BID, stCTR apenas por CTR, e assim por diante. As cinco estratégias foram avaliadas simultaneamente ao longo de uma semana, todas receberam uma amostra aleatória e de mesmo tamanho de todas as requisições feitas ao sistema de anúncios, totalizando mais de 300 milhões de anúncios exibidos. Uma limitação de nossa plataforma que pretendemos aprimorar futuramente é o fato de não isolar as estratégias por grupos de usuários, assim o mesmo usuário é exposto a várias estratégias diferentes.

A Figura 1 apresenta o gráfico normalizado do desempenho diário de cada estratégia em relação às métricas de CTR e ECPM. A estratégia com a melhor taxa de clique foi stCTR e a que gerou mais retorno financeiro foi stBID. A alta taxa de clique em stCTR a fez superar o patamar de retorno financeiro de stRND, em contrapartida, mesmo stBID tendo a pior taxa de clique, excedeu em muito as outras estratégias em termos de ECPM. Enquanto a estratégia stCTX obteve resultado levemente superior a stRND tanto em CTR quanto em ECPM, stBTU se mostrou bem semelhante: em termos de CTR sem um padrão definido; e em termos de ECPM equiparados em um patamar baixo.

O fato de stCTR desempenhar tão bem em CTR indica que podemos olhar para o problema de exibição de publicidade do ponto de vista de recomendação, no qual usuários recomendam anúncios para outros que tenham comportamento semelhante, o que corrobora com a tendência apontada por Garcia-Molina [Garcia-Molina et al. 2011].

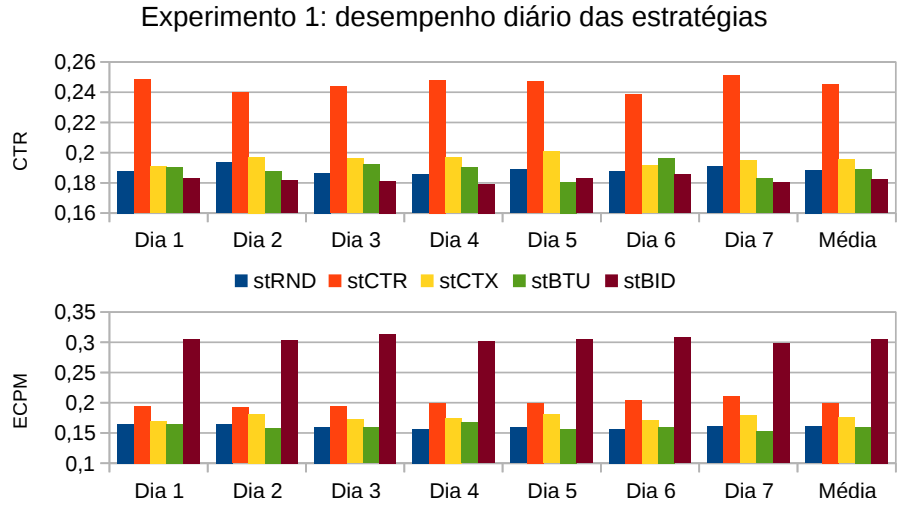


Figura 1. Desempenho diário das estratégias do experimento 1 em termos de CTR e ECPM normalizados. A média de todos os dias é apresentada à direita.

4.2. Experimento 2: comparação de estratégias tradicionais

O segundo experimento é semelhante ao primeiro e analisou comparativamente as seguintes estratégias: stCTR, stBID, stTRAD, stUSER, stMKT. As estratégias stCTR e stBID são as mesmas do experimento 1; stTRAD é a fórmula tradicional de reordenação criada pelo Google em 2002 e nada mais é que uma multiplicação de CTR e BID; stUSER é a soma dos valores normalizados das variáveis relevantes apenas para o usuário (CTR, CTX, BTU); e stMKT é uma solução comercial que combina as variáveis disponíveis. A coleta de dados durou sete dias e mais de 300 milhões de anúncios foram exibidos.

A Figura 2 apresenta o gráfico normalizado do desempenho diário de cada estratégia em relação às métricas de CTR e ECPM. A estratégia stUSER obteve resultado levemente inferior ao de stCTR com relação à taxa de clique e equivalente em ECPM, o que valida que as variáveis CTR, CTX e BTU impactam diretamente no interesse do usuário. A estratégia stTRAD se mostrou a mais equilibrada com relação às duas métricas de desempenho. A estratégia stMKT obteve o melhor resultado em ECPM, superando stBID não somente em desempenho financeiro, mas também em taxa de clique (quase duas vezes superior). Isso ratifica que a reordenação de anúncios baseado nessas variáveis pode produzir ganhos para cada um dos participantes simultaneamente.

4.3. Análise de componentes principais (PCA)

O PCA é uma técnica estatística usada em análise multivariada que realiza uma mudança de base do espaço analisado, de modo que os vetores da nova base (componentes principais) estejam alinhados com os eixos de maior dispersão dos dados [Shlens 2014]. Para entender a correlação entre as variáveis analisadas, realizamos a análise de componentes principais com uma amostra de dados coletada durante os experimentos. Utilizamos apenas dados de anúncios que foram clicados, ou que foram exibidos junto a algum outro clicado. Essa heurística visa aumentar a qualidade dos dados, removendo anúncios que possivelmente não foram nem vistos pelos usuários, e foi necessária para obtermos uma análise conclusiva. A Figura 3 apresenta a projeção dos vetores da base original (BID, CTR, CTX e BTU) em dois planos definidos pela nova base (PC1, PC2, PC3 e PC4).

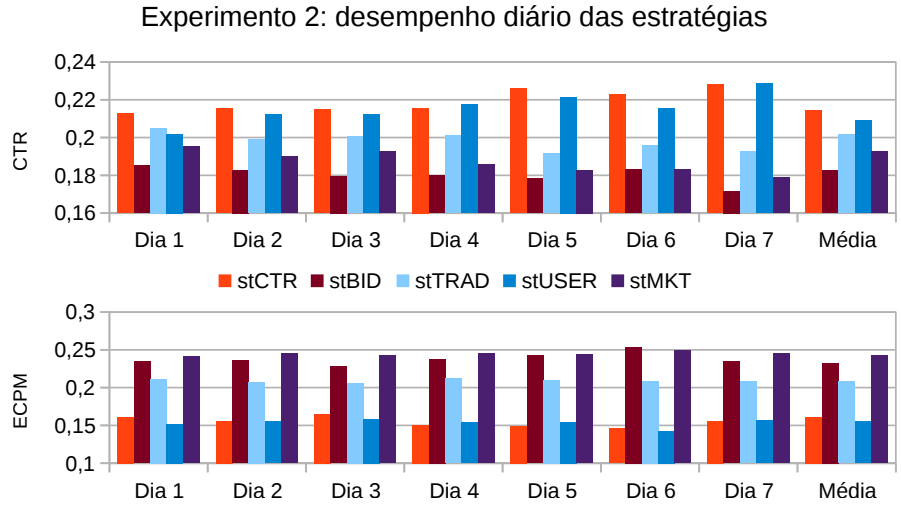


Figura 2. Desempenho diário das estratégias do experimento 2 em termos de CTR e ECPM normalizados. A média de todos os dias está a direita.

No plano de PC1×PC2, vemos que as duas primeiras componentes principais são compostas por uma combinação de BID, CTX e BTU, e uma expressão menor de CTR. A diferença de direção entre BID, CTX e BTU revela que essas variáveis são pouco correlacionadas, já o alinhamento de CTR com BTU revela a utilidade da captura de interesses do usuário para atrair sua atenção. As duas primeiras componentes principais cobrem apenas 56,3% de toda a variância, portanto necessitamos analisar também as duas últimas. No plano de PC3×PC4, vemos que a terceira componente (PC3) é composta principalmente por CTR, o que, em conjunto com a diferença de direção de BID, CTX e BTU no plano de PC1×PC2, revela que as variáveis capturam aspectos diferentes do contexto e que, portanto, nenhuma delas deve ser descartada no problema de reordenação de anúncios. A quarta componente (PC4) está em uma direção de dispersão residual das variáveis BID, CTX e BTU, e revela a existência de uma relação direta entre o valor pago pelo anunciante (BID) e a contextualização entre o anúncio e a página (CTX).

5. Conclusão

A ordenação de anúncios para exibição de publicidade contextualizada é um problema de otimização com múltiplos interesses envolvidos. Muitos aspectos de contexto podem ser explorados, desde o conteúdo da página ao padrão de navegação do usuário. Nosso estudo sobre as variáveis BID, CTR, CTX e BTU mostrou que elas possuem impacto direto no desempenho do sistema de anúncios e validamos que a reordenação de anúncios por meio da combinação delas traz ganhos para o usuário, o anunciante e o publicador. Descobrimos também que há pouca correlação entre as variáveis e que usá-las em conjunto amplia o contexto capturado e as chances de recomendar anúncios melhores para o usuário. Pretendemos evoluir a plataforma de experimentação para permitir que grupos de usuários sejam expostos sempre à mesma estratégia, isso isolará a variável de usuário e viabilizará um estudo nos moldes estatísticos A/B, bem como nos permitirá analisar o efeito do tempo no desempenho das estratégias. Além disso, pretendemos estudar o efeito de sazonalidades, como feriados e datas comemorativas, em que o comércio é fortemente aquecido e a disposição do usuário para conteúdo publicitário é maior.

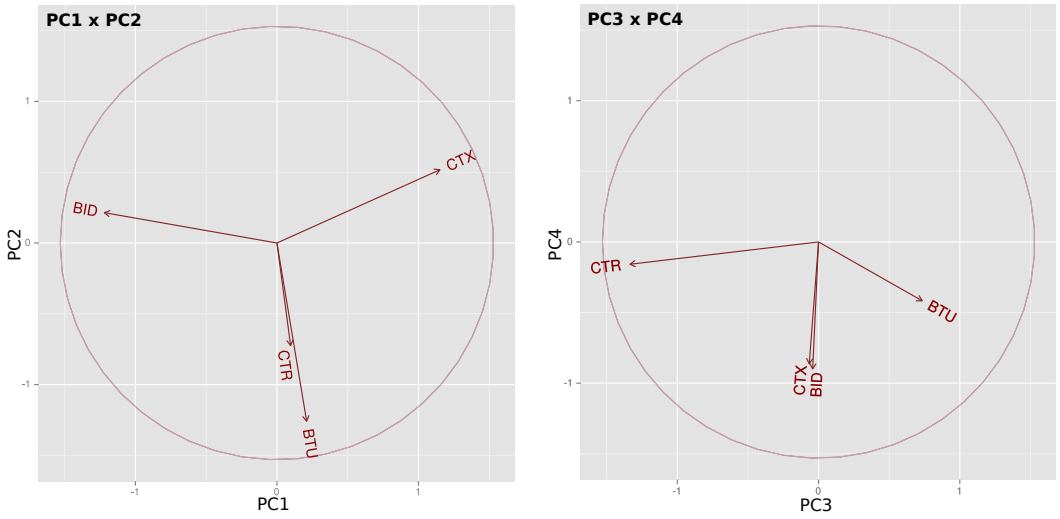


Figura 3. *Loading plots* das componentes principais. A preservação da variância original é de: 30.5% em PC1, 25.8% em PC2, 24.8% em PC3 e 18.6% em PC4.

Agradecimento

Este trabalho foi parcialmente financiado por UOL, Universo Online S.A. e também contou com o apoio institucional do CNPq, processo nro. 304150/2012-6.

Referências

Agarwal, D., Gabrilovich, E., Hall, R., Josifovski, V., and Khanna, R. (2009). Translating relevance scores to probabilities for contextual advertising. In Cheung, D. W.-L., Song, I.-Y., Chu, W. W., Hu, X., and Lin, J. J., editors, *CIKM*, pages 1899–1902. ACM.

Agarwal, D. and Gurevich, M. (2012). Fast top-k retrieval for model based recommendation. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 483–492, New York, NY, USA. ACM.

Azimi, J., Zhang, R., Zhou, Y., Navalpakkam, V., Mao, J., and Fern, X. (2012). Visual appearance of display ads and its effect on click through rate. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 495–504, New York, NY, USA. ACM.

Bagherjeiran, A., Hatch, A. O., and Ratnaparkhi, A. (2010). Ranking for the conversion funnel. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 146–153, New York, NY, USA. ACM.

Bendersky, M., Gabrilovich, E., Josifovski, V., and Metzler, D. (2010). The anatomy of an ad: structured indexing and retrieval for sponsored search. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 101–110, New York, NY, USA. ACM.

Broder, A., Ciaramita, M., Fontoura, M., Gabrilovich, E., Josifovski, V., Metzler, D., Murdock, V., and Plachouras, V. (2008). To swing or not to swing: Learning when (not) to advertise. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 1003–1012, New York, NY, USA. ACM.

Broder, A., Fontoura, M., Josifovski, V., and Riedel, L. (2007). A semantic approach to contextual advertising. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 559–566, New York, NY, USA. ACM.

Broinizi, M., Mutti, D., and Ferreira, J. E. (2014). Application configuration repository for adaptive service-based systems: Overcoming challenges in an evolutionary online advertising environment. In *Proceedings of the 21th International Conference on Web Services*, ICWS '14.

Edelman, B., Ostrovsky, M., Schwarz, M., Fudenberg, T. D., Kaplow, L., Lee, R., Milgrom, P., Niederle, M., and Pakes, A. (2005). Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97.

Garcia-Molina, H., Koutrika, G., and Parameswaran, A. (2011). Information seeking: Convergence of search, recommendations, and advertising. *Commun. ACM*, 54(11):121–130.

Google (2002). Google introduces new pricing for popular self-service online advertising program. http://googlepress.blogspot.com.br/2002/02/google-introduces-new-pricing-for.html , Fevereiro.

IAB (2013). Internet advertising revenue report 2012 full year results. Technical report. http://www.iab.net/media/file/IAB_Internet_Advertising_Revenue_Report_FY_2012_rev.pdf , Abril.

Lacerda, A., Cristo, M., Gonçalves, M. A., Fan, W., Ziviani, N., and Ribeiro-Neto, B. (2006). Learning to advertise. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 549–556, New York, NY, USA. ACM.

Oberoi, A. (2013). The history of online advertising. http://www.adpushup.com/blog/the-history-of-online-advertising , Julho.

Ribeiro-Neto, B., Cristo, M., Golgher, P. B., and Silva de Moura, E. (2005). Impedance coupling in content-targeted advertising. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 496–503, New York, NY, USA. ACM.

Shlens, J. (2014). A tutorial on principal component analysis. *CoRR*, abs/1404.1100.

Tang, J., Liu, N., Yan, J., Shen, Y., Guo, S., Gao, B., Yan, S., and Zhang, M. (2011). Learning to rank audience for behavioral targeting in display ads. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pages 605–610, New York, NY, USA. ACM.

Wang, H., Liang, Y., Fu, L., Xue, G.-R., and Yu, Y. (2009). Efficient query expansion for advertisement search. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 51–58, New York, NY, USA. ACM.

paper:71

Ranking de Pesquisadores em Cenários Ambíguos: uma Abordagem Probabilística

Luciana B. Maroun[1], **Mirella M. Moro**[1]

[1]Departamento de Ciência da Computação, Universidade Federal de Minas Gerais

{lubm,mirella}@dcc.ufmg.br

Resumo. *Diversas situações acadêmicas necessitam obter um ranking de pesquisadores, em geral manualmente com grande investimento de tempo e dinheiro. Automatizar tal tarefa requer associar autores a artigos, o que por sua vez adiciona incerteza aos dados devido à ocorrência de ambiguidade. Neste trabalho, propõe-se empregar uma abordagem probabilística, vinculando uma métrica de incerteza a cada posição. O uso de um desambiguador probabilístico permite explorar alguns dos casos possíveis e aplicar o desvio padrão das posições de dado autor como indicador de incerteza. Os resultados experimentais mostram que o método alcança uma boa estabilidade, em torno de* 80%*, com poucas iterações em comparação ao tamanho do conjunto de situações viáveis.*

Abstract. *Different academic scenarios require ranking researchers, which is usually done manually with high investment of time and money. Automating such task requires associating authors to articles, which in turn adds uncertainty to the data due to ambiguity. In this work, we propose using a probabilistic approach, attaching an uncertainty measure to each position. Using a probabilistic disambiguation method enables to explore some of the possible cases and to use the standard deviation of the positions of a given author as an uncertainty index. Experimental results show that the method reaches a good stability, around* 80%*, with few iterations compared to the size of the set of viable situations.*

1. Introdução

Instituições envolvidas na produção científica frequentemente requerem a definição de um ranking de pesquisadores para concessão de bolsas, de prêmios, análise evolutiva, entre outros motivos. Geralmente, um comitê especializado produz tal ranking, como ocorre, por exemplo, na distribuição de bolsas de produtividade em pesquisa pelo CNPq[1]. Essa tarefa, além de possuir alto custo, torna-se impraticável com o aumento do número de pesquisadores envolvidos. Com isso, técnicas automizadas são necessárias.

Para gerar um ranking de pesquisadores utilizam-se indicadores como volume de artigos, total de citações e h-index [Lima et al. 2013]. Para calculá-los, em geral, é preciso identificar os artigos de cada pesquisador. Técnicas de casamento de cadeias de caracteres não são suficientes nesse caso, pois pode haver diversas representações para o nome de um mesmo indivíduo, isto é, *sinônimos*. Além disso, acadêmicos distintos podem possuir uma nomenclatura em comum, situação de *homônimos*.

A dificuldade em determinar precisamente a qual pesquisador o autor de um artigo se refere pode levar a situações falhas. Exemplos de tais situações podem ser encontrados

[1]CNPq: http://www.cnpq.br/

em bibliotecas digitais, como na DBLP[2]. Pesquisando-se por *Ana Vasconcelos*, por exemplo, uma autora denominada *Ana Vasconcelos* possui majoritariamente artigos de uma professora da Universidade de Sheffield no Reino Unido na área de gerência de informação, porém um dos artigos[3] é de uma outra pesquisadora vinculada à Fraunhofer Portugal na área de IHC para jogos. Este é um exemplo de homônimos, pois duas pessoas diferentes usaram a mesma assinatura em artigos e, por isso, foram consideradas como sendo a mesma entidade. Além disso, duas autoras da lista, *Ana T. R. de Vasconcelos* e *Ana Tereza Ribeiro de Vasconcelos*, dividem os artigos de uma única pesquisadora, configurando uma situação de sinônimos. Em um ranking, a autora com publicações particionadas tende a ser prejudicada, enquanto aquela com artigos agregados, a ser favorecida.

Este trabalho propõe a modelagem probabilística da incerteza inerente à tarefa de associação de pesquisadores a artigos em cenários ambíguos. Em tais situações há pouca informação, em geral apenas o nome do autor é indicador direto de sua identidade, e demanda-se a integração dos dados de diferentes publicações. Além de ser um modelo genérico para representar imprecisão dos dados, tratando tanto casos de sinônimos quanto de homônimos, a utilização de probabilidades é amplamente defendida pela literatura quando existe incerteza [Dong et al. 2009, Magnani and Montesi 2010].

As contribuições deste artigo podem ser resumidas em: (1) definição de um método probabilístico de desambiguação de autores; (2) elaboração de uma métrica, denominada *simcoaut*, relevante para a tarefa de desambiguação; (3) composição de uma método de amostragem e de mapeamento entre mundos possíveis; e (4) determinação de um ranking probabilístico, que contém as incertezas das posições.

2. Definição do Problema

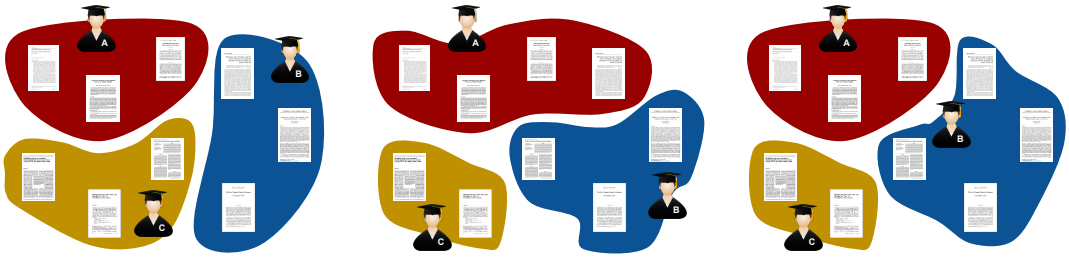
Define-se *referência* como o nome de um autor em uma publicação mais os dados bibliográficos como título e veículo. Seja $\{p_1, p_2, ..., p_m\}$ um conjunto de publicações. A partir das publicações, extrai-se um conjunto de referências a autores $R = \{r_1, r_2, ..., r_n\}$. Além disso, suponha que existam k autores, identificados por inteiros de 1 a k. O objetivo de um *método de desambiguação* consiste em obter um particionamento das referências $W = \{a_1, a_2, ..., a_k\}$, tal que $a_i = \{r \in R | r\ refere\ ao\ autor\ i\}$ [Ferreira et al. 2012b].

No contexto probabilístico, a divisão em autores é flexível, isto é, obtém-se um conjunto de particionamentos $\{W_1, W_2, ..., W_p\}$. Um exemplo de diferentes partições do conjunto de referências pode ser visto na Figura 1. Seja $S = [b_1, b_2, ..., b_k]$ um ranking em que o autor b_j esteja na posição j e corresponda a algum $a_i \in W$. Cada particionamento implica em um ranking distinto, isto é, há um conjunto de resultados $\{S_1, S_2, ..., S_p\}$. A partir desse conjunto, é possível calcular incertezas e obter um ranking probabilístico $S_p = [(b_1, u_1), (b_2, u_2), ..., (b_k, u_k)]$, em que u_i é a incerteza para a posição i, desde que se defina uma métrica de incerteza a partir do conjunto de rankings.

A semântica dessa modelagem pode ser descrita por meio de *mundos possíveis (possible worlds)*. Um mundo possível consiste em um dos estados em que um banco

[2]DBLP: `http://www.informatik.uni-trier.de/~ley/db/`

[3]O artigo é entitulado *Using the Smartphone Accelerometer to Monitor Fall Risk while Playing a Game: The Design and Usability Evaluation of Dance! Don't Fall*. A ambiguidade foi verificada analisando-se a página pessoal da primeira pesquisadora e a página no Linked-in da segunda.



(a) (b) (c)

Figura 1. Exemplos de mundos possíveis para um conjunto de referências.

de dados probabilístico pode estar [Suciu et al. 2011]. No caso de um banco de dados de referências a pesquisadores, o mesmo consiste em cada um dos elementos de $\{W_1, W_2, ..., W_p\}$. A Figura 1 apresenta exemplos de diferentes mundos possíveis.

3. Arcabouço Probabilístico

A geração do ranking probabilístico é composta por três partes: uma etapa de desambiguação probabilística; outra de amostragem dos mundos possíveis; e uma última etapa de obtenção do ranking com cálculo de incertezas. A descrição da sequência procedimental desse arcabouço é apresentada no Algoritmo 1 e detalhada nas próximas seções.

Algoritmo 1: Arcabouço Probabilístico.

```
Data: conjunto_treino, conjunto_teste, iterações
Result: um ranking probabilístico
vetores ← [ ];
foreach bloco ∈ conjunto_treino do
    vetores_b ← obter_vetores_sim(bloco, rotulado=true);
    vetores ← vetores + vetores_b;
end
preditor ← ajustar_regressao_logistica(vetores);
matrizes_prob ← [ ];
foreach bloco ∈ conjunto_teste do
    vetores ← obter_vetores_sim(bloco, rotulado=false);
    probabilidades ← predizer(preditor, vetores);
    matriz_prob ← obter_matriz_prob(probabilidades);
    matrizes_prob ← matrizes_prob + [matriz_prob];
end
mundo_base ← obter_mundo_base(matrizes_prob);
ranking_base ← obter_ranking(mundo_base);
rankings_alt ← [ ];
foreach i ∈ [1, ..., iterações] do
    mundo_alt ← obter_mundo_alt(matrizes_prob);
    ranking_alt ← obter_ranking(mundo_alt);
    rankings_alt ← rankings_alt + [ranking_alt];
end
emparelhamentos ← emparelhar_rankings(ranking_base, rankings_alt);
ranking_prob ← criar_ranking_prob(ranking_base, rankings_alt, emparelhamentos);
return ranking_prob;
```

3.1. Desambiguação Probabilística

O processo de desambiguação probabilística aplica um algoritmo de aprendizado de máquina supervisionado para calcular a probabilidade de duas referências a autores corresponderem ao mesmo pesquisador. Cada par de referências é modelado como um vetor chamado *vetor de similaridade*, que reflete aspectos comparativos relevantes para a determinação de correferência, sendo definido como $(stfidf, simcoaut, intcoaut, simtitle, eqvenue)$ em que:

- $stfidf$ corresponde à métrica soft-TFIDF, uma flexibilização do TFIDF que considera casamentos aproximados entre as palavras [Moreau et al. 2008].
- $simcoaut$ é uma métrica de coautoria proposta aqui que utiliza regras de associação calculadas em toda a base de dados usando o algoritmo *Apriori* [Agrawal and Srikant 1994]. O cálculo é feito somando-se as confianças das regras de associação do tipo $A \rightarrow D_j$ e $B \rightarrow C_i$, em que A e B são os nomes de autor em questão, C_i o i-ésimo coautor de A e D_j o j-ésimo coautor de B, $i = 1, ..., m$ e $j = 1, ..., n$, havendo m coautores de A e n de B. Em suma, são somadas as confianças das coautorias de A com os coautores de B e vice versa.
- $intcoaut$ contabiliza a interseção de coautores no par de referências em questão, sendo utilizado também em [Gurney et al. 2012, Treeratpituk and Giles 2009].
- $simtitle$ corresponde ao coeficiente de similaridade de Jaccard entre os títulos (já processados para conter somente radicais) dos artigos.
- $eqvenue$ é um valor booleano que indica se os veículos são iguais ou não, sendo também aplicado em [Torvik et al. 2005].

As referências são inicialmente particionadas em *blocos ambíguos*, que agrupam aquelas que potencialmente correspondem à mesma entidade. Somente entre pares de referências que estão dentro do mesmo bloco são criados vetores de similaridade, com isso reduz-se o custo de comparação ao desconsiderar os pares claramente distintos. O agrupamento de pesquisadores com inicial e último nome em comum foi o critério adotado, similarmente a desambiguadores do estado-da-arte [Ferreira et al. 2012b].

Um modelo de regressão logística é empregado para calcular as probabilidades de correferência. Esse modelo corresponde à função $y = \frac{1}{1+e^{-x\beta}}$, em que x é o vetor de similaridade (acrescido de uma dimensão cujo valor é 1 para considerar o intercepto), β é um vetor estimado durante o treino que define os pesos de cada parâmetro, e y é a variável que se deseja estimar, no caso a probabilidade de correferência. A vantagem da regressão logística consiste em fornecer saídas no intervalo $[0, 1]$, o que permite a interpretação como probabilidade. Além disso, é simples e apresenta bons resultados em trabalhos prévios de desambiguação (acurácia acima de 90% em [Treeratpituk and Giles 2009]).

Para o treino, um conjunto de vetores de similaridade rotulados é necessário. Os rótulos podem ser 1, indicando que os pares de referência correspondem ao mesmo autor, ou 0, caso contrário. Após o treino, que obtém β, dado um vetor de similaridade, o modelo provê um valor $p \in (0, 1)$, que reflete a confiança de duas referências pertencerem ao mesmo indivíduo e serve como uma estimativa para a probabilidade de correferência. Apesar de esse algoritmo ter sido usado em outras abordagens da literatura, neste trabalho é proposta a interpretação dos valores estimados como *probabilidades*.

3.2. Amostragem de Mundos Possíveis

Investigar todos os particionamentos de referências possíveis é uma tarefa inviável, pois cada bloco ambíguo possui uma quantidade expressiva de particionamentos. Essa quantidade é denominada *Número de Bell (Bell Number)* e é definida pela fórmula de recorrência $B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_n$, em que n é o número de elementos do bloco [Becker and Riordani 1948]. Esse valor possui um crescimento considerável. Com isso, torna-se imperativo formular um método que aproxime a distribuição de mundos possíveis, evitando sua completa inspeção.

Uma vez calculadas as probabilidades, é obtida uma matriz de probabilidades M, tal que $M[r_i][r_j]$ contém a probabilidade da referência r_i pertencer ao mesmo autor que r_j. Em seguida, é utilizado o algoritmo *DBScan* [Theodoridis and Koutroumbas 2008] para particionar as referências em autores. Esse método agrupa os elementos atendendo critérios de tamanho mínimo para um grupo e de distância mínima entre aqueles agrupados. O número de grupos não precisa ser definido como parâmetro, e o problema de ausência de transitividade é resolvido razoavelmente bem, como descrito em [Huang et al. 2006]. A transitividade é uma propriedade desejável para a relação de correferência, uma vez que, supondo que sejam correferentes os pares (a, b) e (b, c), consequentemente a e c devem também corresponder à mesma pessoa por definição. O particionamento obtido nesta fase corresponde ao base e é somente a partir desse que a versão determinística do ranking final, (isto é, excluída a informação de incerteza) é construído. É neste processo, portanto, que se define o número de pesquisadores a serem considerados.

Como a regressão logística produz uma probabilidade de correferência entre cada par de autores do mesmo bloco ambíguo, é praticável simular possíveis situações executando-se um *experimento de Bernoulli* para cada par. Um experimento dessa forma recebe como entrada a probabilidade p e retorna 1 com probabilidade p ou 0 com probabilidade $1 - p$. Repetindo essa execução estocástica em uma série de iterações, aproxima-se da distribuição dos mundos possíveis — a taxa de ocorrência de 1 para cada par é próxima da probabilidade p. Para cada iteração estocástica, é aplicado o algoritmo *k-medoids* [Theodoridis and Koutroumbas 2008] a fim de agrupar as referências. Esse algoritmo possibilita a utilização das métricas pré-computadas pela regressão logística. O seu comportamento é de minimizar as distâncias intra-cluster a um elemento central, o medoid, que é escolhido dentre as referências existentes. Com isso, na etapa de particionamento, obtém-se um que serve como base para o ranking, por meio do DBScan, e outros que derivam rankings alternativos, por meio do k-medoids.

3.3. Obtenção do Ranking Probabilístico

A partir do particionamento base, é possível gerar um ranking determinístico. Posteriormente, deve-se calcular um valor de incerteza para cada posição desse ranking. Para fazê-lo é necessário, primeiramente, obter relacionamentos biunívocos entre os autores do particionamento base e dos alternativos a fim de observar a variação das posições. Cria-se, para isso, um grafo bipartido completo para cada particionamento alternativo, em que os vértices de um conjunto correspondem aos autores do particionamento base e os do outro são os autores do alternativo em questão. O peso entre cada par de autores de diferentes mundos possíveis corresponde ao número de referências em comum. Com isso, aplica-se o algoritmo *Hungariano* [Kuhn 1955] que obtém o emparelhamento máximo e, portanto, retorna o melhor mapeamento entre o par de mundos possíveis considerado.

Como última etapa do método, é calculado o desvio padrão das posições de um autor ao longo de diferentes rankings, que são associados por meio do mapeamento obtido por emparelhamento máximo. Para o cálculo do desvio padrão, considera-se a posição no ranking base como a posição média, já que o objetivo é identificar a variação em torno desse valor. A apresentação das incertezas em uma escala fixa torna sua informação mais compreensível; por isso, optou-se por utilizar números inteiros de 0 a 10. A normalização utiliza $m - 1$ como valor máximo de desvio padrão em um ranking de tamanho m.

Pode-se ilustrar o funcionamento desta etapa utilizando os mundos possíveis da Figura 1. Nesse caso, obtém-se os rankings $[A, B, C]$ (considerado o ranking base), $[A, B, C]$ e $[B, A, C]$, utilizando o volume de artigos como função de ordenação do ranking, a ordem lexicográfica como critério de desempate e nomeando de acordo com o emparelhamento máximo. Supondo que, além desses, existam ainda os rankings alternativos $[B, A, C]$ e $[B, C, A]$. Calculando-se o desvio padrão de cada pesquisador obtém-se $1,095$ para o A, $0,774$ para o B e $0,447$ para o C. Normalizando em relação ao desvio padrão máximo, $m - 1 = 2$, e projetando em uma escala de 0 a 10, obtém-se o ranking probabilístico $[(A, 5), (B, 4), (C, 2)]$, conforme definido na Seção 2.

4. Avaliação Experimental

Os dados utilizados foram retirados de [Ferreira et al. 2012a] e consistem em um conjunto de 4287 referências rotuladas a 220 pesquisadores distintos. Cada referência contém os nomes dos autores, o título, o veículo e um identificador do pesquisador na base. Após a modelagem, obtiveram-se $1.175.609$ vetores de similaridade. Finalmente, 25% das referências foram aleatoriamente selecionadas para constituir o conjunto de treino.

A avaliação experimental é dividida em três partes. A primeira apresenta três análises do desambiguador probabilístico: (*i*) acurácia – permite a comparação do desambiguador proposto com outros métodos e indica seu potencial para modelar correferências; (*ii*) qualidade de estimação – denota o poder preditivo do modelo em sua versão probabilística; e (*iii*) importância dos parâmetros – as covariáveis definidas para o vetor de similaridade são analisadas a fim de verificar se são realmente importantes para a estimação de probabilidades ou se adicionam um viés indesejado. Na segunda parte, a amostragem de mundos possíveis e o ranking probabilístico são avaliados por meio do cálculo da estabilidade das incertezas. Esse experimento analisa a consistência do método, representada pela variação dos resultados entre diferentes execuções. Investigar a estabilidade é importante devido à aproximação da distribuição de mundos possíveis ser efetuada por meio de um procedimento estocástico. Na terceira parte, mede-se o tempo de execução variando o tamanho da entrada e o número de iterações estocásticas.

Acurácia do Desambiguador Determinístico. A acurácia do modelo de regressão logística foi calculada considerando sua versão determinística, em que se classifica como 0 se a probabilidade for menor do que $0,5$ e como 1, caso contrário. Obteve-se $90,3\%$ em um esquema de validação cruzada com 5 partições. Esse resultado é muito próximo de abordagens da literatura [Ferreira et al. 2012b], porém a acurácia varia muito de acordo com os dados, o que dificulta a comparação. Em [Ferreira et al. 2012a], a mesma base de dados é utilizada e a acurácia máxima alcançada dentre diferentes abordagens exploradas foi de $92,0\%$. O desambiguador proposto neste trabalho, além disso, possui flexibilidade devido ao formato probabilístico da saída.

Tabela 1. Dados da estimação de parâmetros para o modelo de desambiguação.

Parâmetro	Valor estimado	Desvio Padrão	Valor Z
$stfidf$	$1,9960$	$0,0103$	$166,75$
$simcoaut$	$7,7544$	$0,0120$	$223,92$
$intcoaut$	$3,4642$	$0,0346$	$98,39$
$simtitle$	$15,4557$	$0,0818$	$188,85$
$eqvenue$	$2,4636$	$0,0352$	$70,05$

Qualidade da Estimação de Probabilidades. As probabilidades também foram avaliadas em um processo de validação cruzada com 5 partições aplicando-se a métrica *Brier score* [Brier 1950], obtendo-se $0,077$. Essa medida calcula a qualidade das probabilidades estimadas por meio da expressão $BS = \frac{1}{n}\sum_{i=1}^{n}(p_i - c_i)^2$, em que n é o número de vetores de similaridades, p_i é a probabilidade prevista do i-ésimo vetor, c_i é a informação verdadeira de presença ou não de correferência do i-ésimo vetor e $i = 1, 2, ..., n$. Quanto menor esse valor, menor o erro das predições e melhor a qualidade.

Importância dos Parâmetros do Vetor de Similaridade. Para cada parâmetro do vetor de similaridade foi aplicado o *Teste de Wald (Wald Test)* [Wasserman 2010], que consiste em um teste de hipótese no formato $H_0 : \theta_i = \theta_0 \; vs. \; H_1 : \theta_i \neq \theta_0$, em que H_0 é a hipótese nula; H_1, a hipótese alternativa; e θ_i o i-ésima covariável do modelo. Esse teste foi feito usando $\theta_0 = 0$ e, portanto, verifica se determinado parâmetro é importante para o modelo. Obtiveram-se p-valores (que correspondem à probabilidade de rejeitar a hipótese nula quando a mesma é verdadeira) abaixo de $2 \cdot 10^{-16}$, levando à rejeição da hipótese nula e à conclusão de que todas as covariáveis são relevantes para a desambiguação. As estimativas dos parâmetros, os desvios padrões e os valores Z, que são as estatísticas do Teste de Wald, são apresentados na Tabela 1. Tais estatísticas do teste seguem uma normal padrão e, portanto, quanto maior o seu módulo, maior a distância do valor obtido à média e maior a indicação de rejeição da hipótese nula. Com isso, o ranking de importância dos parâmetros é: $simcoaut$, $simtitle$, $stfidf$, $intcoaut$, $eqvenue$. Logo, a covariável mais discriminativa é a métrica $simcoaut$ proposta neste trabalho.

Estabilidade das Incertezas. Uma qualidade desejável das incertezas calculadas consiste na presença de estabilidade, isto é, não deve haver uma variação considerável em seus valores para diferentes execuções. Esse problema pode acontecer devido à utilização de um procedimento estocástico de amostragem de mundos possíveis. A métrica de estabilidade utilizada foi o *coeficiente de correlação de ranking de Spearman* [Kendall 1948] sobre os autores dos rankings ordenados pela incerteza, já que a comparação relativa é a informação mais importante. Essa métrica foi escolhida por, além de observar as mudanças de posições, também considerar o quanto variam. O valor final consiste na média dessa métrica calculada para 10 pares de rankings (obtidos em 20 execuções), para dado número de iterações estocásticas. A Figura 2 mostra a variação da estabilidade com o número de iterações com intervalos de confiança de 95%. Nota-se que, com o aumento do número de iterações, a estabilidade tende a crescer e o intervalo de confiança a diminuir, apesar de haver certa oscilação devido à natureza estocástica do processo. Isso é uma evidência de que a solução do método gradativamente aproxima do resultado do verdadeiro (que considera todos os mundos possíveis) com o aumento do número de iterações. A partir de aproximadamente 500 iterações, a estabilidade obtida foi acima de 80%, sendo o valor máximo em torno de 85%. O alto custo em relação ao tempo de execução para

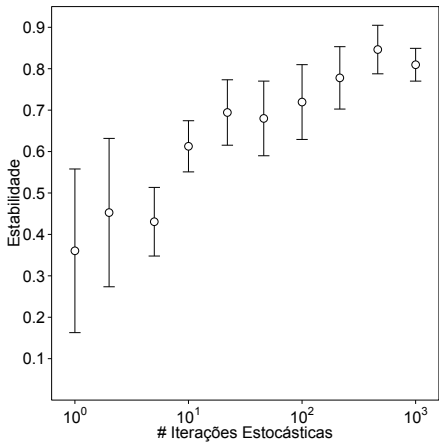


Figura 2. Estabilidade em função do número de iterações estocásticas.

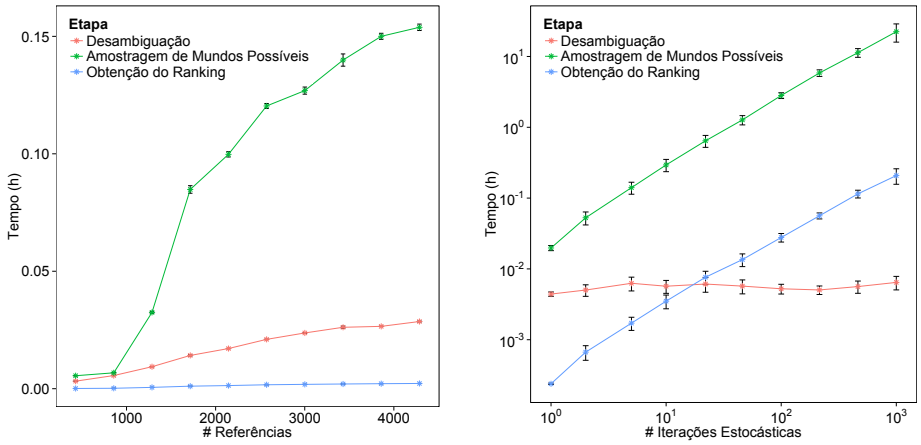
essa etapa, conforme discutido a seguir, limita o estudo de mais iterações. O tamanho do espaço de mundos possíveis, calculado aplicando a fórmula do número de Bell para cada bloco de referências, é da ordem de 10^{122867}; portanto, alcançou-se uma alta estabilidade considerando apenas uma fração muito pequena das situações possíveis.

Tempo de Execução. O tempo de execução foi avaliado considerando a variação do tamanho da entrada na fase de teste e do número de interações estocásticas, como é apresentado na Figura 3 juntamente com intervalos de confiança (IC) de 95%. Na Figura 3(a), alguns ICs são muito pequenos e difíceis de serem visualizados. As medições foram realizadas em um computador com processador Intel Xeon 2.40 GHz e 16GB de memória RAM. Nota-se que a etapa de desambiguação é a mais custosa em relação a ambos os recursos. Quanto à variação do tamanho da entrada, observa-se que o tempo de desambiguação cresce, uma vez que mais pares de referências existem para o cálculo de probabilidades; na fase de amostragem de mundos possíveis, mais elementos pertencem a cada um desse, o que aumenta igualmente o custo; já na etapa de composição do ranking, apesar de haver mais elementos a serem ordenados, a taxa de crescimento é muito baixa.

Analisando o tempo de execução em função do número de iterações (apresentado em escala log-log no gráfico), observa-se que não há efeito observável sobre o tempo de desambiguação, o que é esperado já que esse parâmetro não é utilizado nesse momento. No processo de desambiguação, por sua vez, existe um crescimento aproximadamente linear: cada nova iteração implica em um novo mundo possível a ser investigado. Na obtenção do ranking probabilístico também há um crescimento linear, pois o número de posições a serem consideradas para o cálculo de incerteza cresce proporcionalmente.

5. Trabalhos Relacionados

Diversas soluções da literatura existem para desambiguação de nomes de autores, conforme trabalho de revisão recente [Ferreira et al. 2012b], sendo divididas em duas abordagens principais: métodos de agrupamento de autores, que particionam as referências usando alguma métrica de similaridade; e métodos de atribuição de autores, que classificam as referências como pertencentes a determinado autor. Como exemplo do primeiro método, em [Huang et al. 2006] é utilizado o algoritmo DBScan para agrupar o conjunto de referências em que a distância é calculada por um algoritmo SVM ativo e online. Já



(a) Tamanho da entrada (b) Número de iterações estocásticas

Figura 3. Tempo em função de diferentes variáveis.

a abordagem descrita em [Han et al. 2004] é um exemplo do segundo caso, em que são aplicados dois métodos de aprendizado supervisionado — Naive Bayes e SVM — para classificar o autor correspondente a uma referência. Porém, nenhuma das propostas apresentadas na revisão realiza a desambiguação no formato probabilístico.

Um ranking probabilístico pode ser interpretado sob diferentes perspectivas [Ilyas et al. 2008]: resultados aproximados para dados certos; resultados probabilísticos para dados incertos; e ordenação de dados incertos. No primeiro caso, não existe incerteza nos dados, mas na satisfabilidade de um registro a uma consulta. No segundo, retornam-se múltiplos resultados a uma consulta ordenados pela probabilidade. O ranking probabilístico considerado neste trabalho trata do último, isto é, os objetos de um banco de dados incertos são ordenados segundo alguma função. Entretanto, não foi encontrado outro trabalho na literatura que propagasse a informação de incerteza para o ranking.

6. Conclusões e Trabalhos Futuros

Em situações em que os dados possuem algum tipo de incerteza envolvida, a representação dessa informação na saída é bastante válida, agregando informação que pode ser útil. Em um ranking probabilístico, é possível identificar os elementos cujas posições são menos confiáveis. A novidade deste trabalho consiste em utilizar probabilidades para derivar incertezas associadas às posições do ranking, apresentando uma boa estabilidade com poucas iterações em relação ao número de mundos possíveis existentes. Melhorias podem ser exploradas como a utilização de um método de agrupamento que melhor estime o número de pesquisadores; a aplicação de outras métricas no cálculo de incertezas; e a definição de uma nova semântica de mapeamento dos particionamentos, mesmo entre aqueles com diferentes quantidades de partições.

Agradecimentos. Trabalho parcialmente financiado pelo CNPq e InWeb, Brasil.

Referências

[Agrawal and Srikant 1994] Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Procs. of VLDB*, San Francisco, CA, USA.

[Becker and Riordani 1948] Becker, H. W. and Riordani, J. (1948). The arithmetic of bell and stirling numbers. *Amer. J. Math.*, 70:385–394.

[Brier 1950] Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3.

[Dong et al. 2009] Dong, X. L., Halevy, A., and Yu, C. (2009). Data integration with uncertainty. *The VLDB Journal*, 18(2):469–500.

[Ferreira et al. 2012a] Ferreira, A. A. et al. (2012a). Active associative sampling for author name disambiguation. In *Procs. of JCDL*, Washington, DC, USA.

[Ferreira et al. 2012b] Ferreira, A. A. et al. (2012b). A brief survey of automatic methods for author name disambiguation. *SIGMOD Rec.*, 41(2):15–26.

[Gurney et al. 2012] Gurney, T. et al. (2012). Author disambiguation using multi-aspect similarity indicators. *Scientometrics*, 91(2):435–449.

[Han et al. 2004] Han, H. et al. (2004). Two supervised learning approaches for name disambiguation in author citations. In *Procs. of JCDL*, Tucson, USA.

[Huang et al. 2006] Huang, J., Ertekin, S., and Giles, C. L. (2006). Efficient name disambiguation for large-scale databases. In *Procs. of PKDD*, Berlin, Germany.

[Ilyas et al. 2008] Ilyas, I. F. et al. (2008). A survey of top-k query processing techniques in relational database systems. *ACM Comput. Surv.*, 40(4):11:1–11:58.

[Kendall 1948] Kendall, M. (1948). *Rank correlation methods*. C. Griffin.

[Kuhn 1955] Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quartely*, 2:83–97.

[Lima et al. 2013] Lima, H. et al. (2013). Aggregating productivity indices for ranking researchers across multiple areas. In *Procs. of JCDL*, Indianapolis, USA.

[Magnani and Montesi 2010] Magnani, M. and Montesi, D. (2010). A survey on uncertainty management in data integration. *J. Data and Information Quality*, 2(1):5:1–5:33.

[Moreau et al. 2008] Moreau, E., Yvon, F., and Cappé, O. (2008). Robust similarity measures for named entities matching. In *Procs. of COLING*, Manchester, UK.

[Suciu et al. 2011] Suciu, D. et al. (2011). *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

[Theodoridis and Koutroumbas 2008] Theodoridis, S. and Koutroumbas, K. (2008). *Pattern Recognition*. Synthesis Lectures on Data Management. Academic Press, 4 edition.

[Torvik et al. 2005] Torvik, V. I. et al. (2005). A probabilistic similarity metric for medline records: A model for author name disambiguation: Research articles. *JASIST*, 56(2):140–158.

[Treeratpituk and Giles 2009] Treeratpituk, P. and Giles, C. L. (2009). Disambiguating authors in academic publications using random forests. In *Procs. of JCDL*, Austin, USA.

[Wasserman 2010] Wasserman, L. (2010). *All of Statistics: A Concise Course in Statistical Inference*. Springer Publishing Company, Incorporated.

paper:72

Towards Supporting Systematic Mappings Studies: An Automatic Snowballing Approach

Fábio Bezerra[1], Carlos H. Favacho[1], Rafael Souza[4], Cleidson de Souza[2,3]

[1] ICIBE-UFRA – Belém-Pará-Brasil

[2]ITV-DS – Belém-Pará-Brasil

[3]Faculdade de Computação-UFPA – Belém-Pará-Brasil

[4]IC-UNICAMP – Campinas-São Paulo-Brasil

fabio.bezerra@ufra.edu.br

cleidson.desouza@acm.org

Abstract. *Systematic mapping is a secondary research method that aims to summarize and synthesize the current state of an area, providing a general map of the field. It is a technique that requires the execution of a series of steps, many of them repetitive, which makes this technique time-consuming and error-prone. To address these issues, this paper presents an algorithm for automatic selection of references based on both backward snowballing (from the list of references) and forward snowballing (finding citations to the papers). Our algorithm is especially useful for supporting the selection phase of a systematic mapping study, therefore it represents an effort towards a tool for facilitating systematic mapping research. In order to assess its efficacy and efficiency, we evaluated the algorithm in a set of experiments using data collected from a semester-long graduate course about Computer-Supported Cooperative Work (CSCW).*

1. Introduction

As a research area evolves through years, the number of studies in such area often increases. This can be noticed by the number of papers published in conferences and/or journals and even with the creation of new conferences focusing on that particular area. At some point, it then becomes important to summarize the current state of the area. Such an overview is helpful to guide new researchers as well as to help the field itself to assess its evolution, providing then new directions for future research.

An overview of a research area can be provided by the so-called secondary studies [Kitchenham and Charters 2007]. A secondary study aims to review all primary studies relating to a specific research question in order to integrate and synthesize evidences about this question [Kitchenham and Charters 2007]. Among the existing secondary studies, two stand out for having a well-defined methodology: systematic review and systematic mapping. These methods adopt a rigorously defined process in order to reduce the bias of their conclusions [Petersen et al. 2008, Kitchenham and Charters 2007, Scannavino 2012] and, thus, they are known as **systematic studies**. A systematic mapping is a method of secondary research that aims to show the state of the art of the analyzed area through a **general map**, usually presented as diagrams, charts and statistics [Petersen et al. 2008, da Silva et al. 2012]. On the other hand, a systematic review is a

secondary research method used to provide a comprehensive and clear assessment of the state of a research area, relevant to a particular topic of interest [Felizardo et al. 2012].

Systematic reviews and systematic mappings differ in terms of goals, breadth and depth, and their usage have different implications for the classification of the topic being investigated and the research approach. According to [Kitchenham and Charters 2007] systematic mapping studies "are designed to provide a wide overview of a research area, to establish if research evidence exists on a topic and provide an indication of the quantity of the evidence" whereas the systematic review is used "to identify, analyze and interpret all available evidence related to a specific research question in a way that is unbiased and repeatable".

1.1. Motivation

The execution of a systematic study is a process quite costly and error-prone. First, it employs activities like studying, reading and sorting large amounts of articles. Second, it requires the cooperation of many researchers during the process. For that reason, it is essential that the search of papers return only the most relevant ones. Then, to obtain the desired quality in the activity of identifying articles, methods of strictly planned search are commonly used.

In this context, it is worth to notice that it is a common practice among researchers to select a set of the most important papers in a certain area of research and, from these papers, identify the relevant related work. This survey practice is the basis of the method of selection known as snowballing[Jalali and Wohlin 2012]. The most common snowballing approach works in the following way: from a relevant paper, references from this paper are selected, and then from these selected papers, new references are selected in a iterative process.

Actually, the snowballing is a more general method of selection of papers, but it also requires a set of initial articles, called seeds, to start the process. In this case, from the seeds, there are two approaches to selection: backward snowballing, which selects papers referenced by the seeds. This is the most common and intuitive adopted approach. The other approach is forward snowballing, which selects papers that do cite the seeds. The selected papers from an iteration of the snowballing algorithm compose the group of articles, which will be the seeds for the following iteration. The process continues iteratively until a stopping criterion is satisfied.

Given this description, it is clear that the snowballing approach requires tool support, because, otherwise, it would demand an effort that would make the approach infeasible, if manually conducted. For instance, just the selection of a large number of articles, in both snowballing approaches (backward and forward), can be a large, complex and and time-consuming task. Therefore, a tool that automates all, or part, of the snowballing process presents itself as an interesting alternative support to the selection of papers, because it would decrease the time needed for the implementation of the process, facilitate the work of the researchers involved, and, therefore, provide benefits for the entire research community.

1.2. Objective and Outline

This paper presents an algorithm that automates the snowballing method for selecting papers in a systematic study. The algorithm presented in this paper was implemented in a web system called Ramani [de Souza et al. 2013], developed for supporting a systematic mapping project. [de Souza et al. 2013] focus on the collaborative aspects of the tool, while this paper focuses on the algorithm description, implementation, and evaluation.

The rest of this paper is organized as follows: we present in Section 2 the theoretical background that informed the creation of the automatic snowballing algorithm and guided its implementation in the Ramani tool. Details of this algorithm are described in Section 3, while its evaluation is described in Section 4. Finally, in Section 5 we present our final comments and plans for future work.

2. Background and Related Work

In this section, we first describe the tasks necessary for supporting the systematic mapping process (Section 2.1). This overview is important at this point so that the reader can understand the complexity and rigor of a systematic study. In the context of this paper, the search step is especially important, for it is the core of snowballing algorithm. Moreover, searching is a step that requires a lot of effort from the researchers. Finally, in Section 2.2 we present some related work.

2.1. The Systematic Mapping Process

Figure 1 presents the phases of a systematic mapping study proposed in [Petersen et al. 2008]. In the first phase, the **definition of the research question**, questions should be formulated based on the objective of the research, always focusing in the ultimate goal of a systematic mapping: to produce an overview of a research area. As an outcome, the scope of the review is defined. This scope is used as an input for the search, the next step of the systematic mapping process.

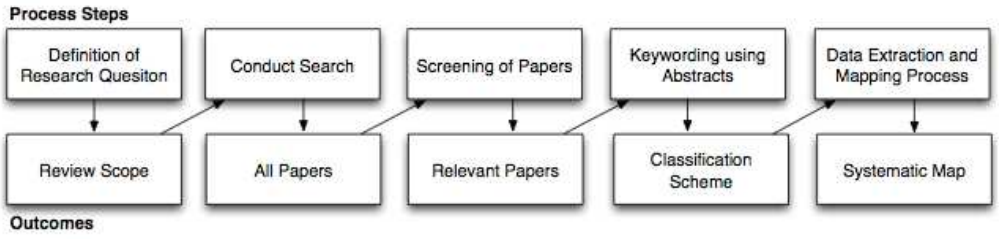


Figure 1. Systematic Mapping Process

The second step is to **conduct a search for primary studies**. These studies are identified by using search strings on scientific libraries / databases or browsing manually through relevant conference proceeding or journal publications. After obtaining the initial set of papers, they should be **screened to select the relevant papers** that help to answer the research questions, applying inclusion and exclusion criteria, i.e., criteria that define whether a paper should be included or excluded from the list of relevant papers. The following phase, **keywording of abstracts**, is often done in two steps. First, the reviewers read abstracts from the papers and look for keywords and concepts that reflect

the contribution of the paper. When the final set of keywords has been chosen, they can be clustered and used to form the categories for the map.

Different categories of papers can be used depending on the research question. We illustrate an interesting category proposed by Petersen et al. (2008) that focuses on the type of contribution reported in the paper: a tool, a process, a method, etc. This category is based on an existing classification of research types and described in [Wieringa et al. 2006]:

Solution Proposal papers the paper presents a new solution (method or tool), but does not evaluate it;

Validation Research papers the paper presents a new solution (method or tool) and evaluates it in a simulated or fictitious scenario;

Evaluation Research papers the paper presents a new solution (method or tool) and evaluates it in a real scenario;

Philosophical papers the paper proposes a taxonomy or a conceptual framework of the field;

Opinion papers the paper express a personal opinion of a solution (method or tool) that already exists, but it does not report an evaluation; and

Experience papers the paper reports the use of a solution (method or tool) that already exists and relates an experience assessment of the solution.

Finally, in the **data extraction and mapping** phase, once the classification scheme is developed, one must extract the data from each paper including year of the publication, authors, venue, categories, etc and document this in a format that can be later processed (e.g., a spreadsheet). With this information, the frequencies of publications in each category can be computed. The analysis of the results focuses on presenting the frequencies of publications for each category allowing one to find out which categories have been emphasized in past research and, as a consequence, to identify gaps and possibilities for future research.

2.2. Related Work

Sytematic studies are mainly based on search strings in databases [Kitchenham and Charters 2007, Dieste and Padua 2007, Petersen et al. 2008, Kitchenham et al. 2009], but there are some efforts based on the selection of the list of references and snowballing [Webster and Watson 2002, Runeson and Skoglund 2009, Jalali and Wohlin 2012]. Whatever the chosen approach, what it is really important in a systematic review is to find as many primary studies relating to the research question as possible [Kitchenham and Charters 2007]. In this context, even systematic reviews based on search strings recommend the selection of **relevant** primary studies as another source of selection.

[Jalali and Wohlin 2012] report a comparison between using snowballing and search strings as a way for conducting a systematic study. An important result of this work is that "despite the differences in the included papers, the conclusions and the patterns found in both studies are quite similar". That is, for the context of this work, snowballing appears as a good automatic approach. For example, in [Webster and Watson 2002] the authors recommend snowballing as the main method to find relevant literature. More specifically, they suggest the use of relevant papers from leading journals in the beginning

of the method as seeds. However, comparing to our approach, both of these works do not deal with snowballing in an automatic way.

The work of [Runeson and Skoglund 2009] is closer related to the snowballing approach presented here. These authors present a search strategy based on four components: (i) a "take-off paper"; (ii) papers referenced by the "take-off paper"; (iii) identification of "cardinal papers"; and (iv) papers from external sources referencing the "cardinal papers". The "take-off paper" is a paper regarded as very relevant on the topic – the authors argue that researchers conducting a systematic review should easily be able to select such a relevant paper, based on their pre-understanding of the research question. On the other hand, "cardinal papers" are those papers referenced more than others – the authors believe that those papers are more likely to be referenced also from the relevant papers available in external resources. However, different from the snowballing algorithm presented in Section 3, the first two components of this approach select just the list of references from the "take-off paper". As we will explain later, this is equivalent to only the first iteration of our snowballing algorithm. Similarly, the last two components of [Runeson and Skoglund 2009]'s approach adopt a procedure alike forward snowballing, but limited to one iteration. As we will describe in the following section, our algorithm conducts forward and backward snowballing during several iterations.

3. An Automatic Snowballing Algorithm

As mentioned before, the main contribution of this paper is an approach for selecting papers based on snowballing, which we call automatic snowballing. Such an approach is called automatic because it does not require the interaction of the researcher during the selection of papers. The automatic snowballing simultaneously uses both backward and forward snowballing during its execution, which ends when the algorithm is not able to find additional articles from the group of referenced or cited papers. Therefore, in order to minimize the stress of processing, this algorithm considers as an input parameter the definition of one or more conferences (or journals) to limit the search space or scope, i.e., this works as a filter for the selection of articles referenced or cited. This algorithm is described in the following section.

3.1. The Algorithm

Our algorithm requires four input parameters: (i) **seeds**, which represent the list of known relevant papers; (ii) **conferences**, which are used to filter the papers selected during the process, limiting the search space and contextualizing the topic of interest; and finally, the last two input data, (iii) **project** and (iv) **collaborator**, which are used to save the selection (in a given project), and associate the collaborators or researchers in the given project that can have access to the returned papers. The result is saved in a variable **groupsOfSeeds**, and it is a "list of list of papers", that is, each iteration of the algorithm returns a *list of papers*, which is added in the *groupsOfSeeds* and contains the seeds of next iteration.

The algorithm is described on Algorithm 1. The seeds used as input are added in the result set (**groupsOfSeeds**) at the very beginning of algorithm, then they are saved as a partial selection for the collaborator in the project. The snowballing process ends when there are not seeds to be processed anymore, that is, in Line 4 when the list of papers (**seeds**) is empty. Otherwise, the lists of references and citations of each paper in the

Algorithm 1: Automatic Snowballing

```
Input: seeds
Input: conferences
Input: project
Input: collaborator
Output: groupsOfSeeds
1  groupsOfSeeds ← {};
2  groupsOfSeeds.add(seeds);
3  select(seeds, project, collaborator);
4  while not seeds.isEmpty() do
5      seeds ← createSeeds(seeds);
6      if seeds not null then
7          seeds ← removeDuplication(groupsOfSeeds, seeds);
8          seeds ← filterByConferences(seeds, conferences);
9          groupsOfSeeds.add(seeds);
10         select(seeds, project, collaborator);
```

seeds list are selected, creating the seeds to be used in the next iteration (**createSeeds** in Line 5).

It is worth to note that **createSeeds** is the function responsible for extracting the data from the digital library, that is, in an implementation of this algorithm it should include a mechanism for collecting this data, like a web-crawler. The papers found by that function may have already been selected before, so it is important to remove the duplications (Line 7). Moreover, a filtering is applied over the resulted list of papers (Line 8), removing those articles that are not published in the given conference list (**conferences**).

3.2. Implementation issues

The automatic snowballing algorithm is available as a selection function in the Ramani, a collaborative software tool [de Souza et al. 2013], which was designed over a set of free and popular technologies like Java, JSF, Primefaces, JPA and MySQL. To use Ramani, the user should upload a file with the initial seeds and the conference list used to define the scope of the review. This file is added within a specific project, which can be accessed by a specific set of collaborators.

Furthermore, as mentioned before, Ramani implements a crawler that reads information from specific sites, namely the ACM or IEEE digital libraries, since the current implementation of the crawler supports only these two digital libraries. During the execution of the crawler, when an article is found in the list of references or citations from a seed, this article is first searched in the local database. If this article is not in the local database, then its data is extracted from the digital libraries. By conducting queries in the local database first, we aim to optimize the processing time of our tool. In addition, to avoid being blocked by the digital libraries, we randomly set a delay in seconds for the next query.

4. Automatic Snowballing Algorithm: Assessment

For the assessment of the proposed algorithm, we have used the data described in Section 4.1, while the results of test are presented in Section 4.2.

4.1. Materials and Methods

The materials used for the assessment of the automatic snowballing algorithm were collected in a graduate class on 2012. This class focused on the topic of Computer-Supported Collaborative Work, or simply CSCW. To be more specific, during this class, a review study was conducted by 10 participants: seven students, a faculty, and the first two authors. The faculty is the last author of this paper, while collaborators are from UFRA.

The objective of that study was to create a mapping of the studies about collaborative software engineering in the context of the CSCW area, i.e., using the papers available in the ACM CSCW Conference, through the systematic mapping methodology (see Section 2.1). For that, all papers published in the previous 16 editions of CSCW conferences were analyzed, i.e., the period from 1986 to 2012. This included an universe of 639 full and short papers, which were filtered by the participants to select those which satisfy the following pre-requisites: (i) papers that focused on the development of a particular software collaboratively; (ii) papers that described empirical studies on how software engineers worked; or (iii) papers that described software tools or methods that allowed software development to be performed collaboratively, or that allowed the construction of tools that supported collaborative software development. We present in Table 1 the 31 papers selected in that study. Due to space limitations, we present only the DOI (Document Object Identify) number for each paper.

Table 1. Selected papers about collaborative software development

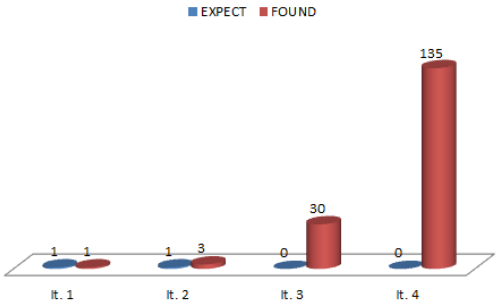
Year	DOI	Year	DOI	Year	DOI
2012	10.1145/2145204.2145401	2008	10.1145/1460563.1460654	2004	10.1145/1031607.1031612
2012	10.1145/2145204.2145403	2008	10.1145/1460563.1460581	2004	10.1145/1031607.1031620
2012	10.1145/2145204.2145345	2008	10.1145/1460563.1460565	2004	10.1145/1031607.1031622
2011	10.1145/1958824.1958851	2006	10.1145/1180875.1180883	2002	10.1145/587078.587080
2011	10.1145/1958824.1958923	2006	10.1145/1180875.1180882	2000	10.1145/358916.359004
2011	10.1145/1958824.1958889	2006	10.1145/1180875.1180884	1990	10.1145/99332.99352
2010	10.1145/1718918.1718972	2006	10.1145/1180875.1180906	1990	10.1145/99332.99356
2010	10.1145/1718918.1718958	2006	10.1145/1180875.1180929	1990	10.1145/99332.99347
2010	10.1145/1718918.1718973	2004	10.1145/10316071031704	1986	10.1145/637069.637071
2010	10.1145/1753326.1753677	2004	10.1145/10316071031621		
2010	10.1145/1718918.1718971	2004	10.1145/1031607.1031611		

This list of papers allowed us to identify the initial seeds used in the assessment of our automatic snowballing algorithm, as well as the efficacy and efficiency of the algorithm, i.e., the parameters for evaluation. For evaluating the efficacy, we expected as the result of our algorithm to obtain the same set of papers presented in Table 1. On the other hand, for evaluating efficiency or performance, we expected to obtain such papers in a short number of iterations – but at most four iterations for the context of this assessment.

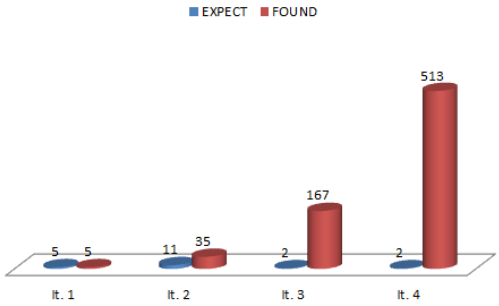
Because our automatic snowballing algorithm applies both backward and forward snowballing, we decided to assess three kinds of initial seeds, namely: (i) seeds from the beginning of the whole period used in the search (1986); (ii) seeds from the middle period under analysis, i.e., papers from 2006; and (iii) seeds from the final of the period (papers

from 2012[1]). These seeds were chosen because testing the papers of 1986 help to assess the forward snowballing component, while the papers of 2012 help to assess the backward snowballing component. Finally, seeds from the middle help us to assess both backward and forward aspects of the snowballing algorithm.

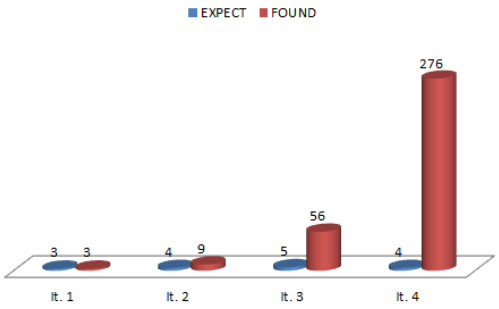
4.2. Results and Discussion



(a) Iterations for seeds of 1986



(b) Iterations for seeds of 2006



(c) Iterations for seeds of 2012

		Iteration 1	Iteration 2	Iteration 3	Iteration 4
1986	EXPECTED	1	1	0	0
	FOUND	1	3	30	135
2006	EXPECTED	5	11	2	2
	FOUND	5	35	167	513
2012	EXPECTED	3	4	5	4
	FOUND	3	9	56	276

(d) Data from three different set of seeds

Figure 2. Results for automatic snowballing using three different sets of seeds

We present in Figure 2 the results of the algorithm assessment. In Figure 2(d) we report the data used for plotting the three graphics, which report, for each iteration, both the number of found papers and the number of expected papers among the found ones. The first iteration represents the initial set of seeds, and because they were composed from the papers of Table 1, the number of found papers is equal to the number of expected ones.

Among the three sets of seeds, the one that helped find more new expected papers in four iterations was the set of papers of 2006 (see Table 2). Such a set of seeds helped find 15 new expected papers, while the set of seeds of 1986 helped find just one new expected paper, and the set of seeds of 2012 helped find 13 new expected papers. Moreover, at the end of the forth iteration the seeds of 2006 helped select 720 papers of the ACM CSCW Conference, while the seeds of 1986 helped select 169 papers and the seeds of 2012 helped select 276 papers. Therefore, because it intensively explored both backward and forward search, since this set of seeds was in the middle of the search space, the seeds of 2006 helped select more papers of the given search context (CSCW

[1]When this paper was being written, the last edition of the CSCW Conference happened on March, 2014, while in the data available for the evaluation of the algorithm the last edition is 2012, since that is when the class who conducted the systematic mapping took place.

Conference), so this helped find more new expected papers. As reported in Table 2, the seed of 2006 has bigger recall (it helped find more papers than were expected).

Table 2. Data after four iterations

	Initial Seeds	Found	Expected	New Expected	Recall	Precision
1986	1	169	2	1	6.45%	1.18%
2006	5	720	20	15	64.52%	2.78%
2012	3	344	16	13	51.61%	4.65%

Combining the results of each set of seeds, we identified that 814 papers of CSCW Conference were selected. Among these, 22 papers are part of the set of expected papers presented in Table 1. It is worth to notice that the number of available papers in the CSCW Conference increased on 2014, so the universe of papers is bigger than the 639 availables on 2012.

A problem of the current implementation of this algorithm is its processing time. At each iteration, because more seeds are found, the processing time increases in a polynomial way. For that reason, the results presented herein are limited to four iterations, which required, for each assessed set, approximately 12h of processing. Therefore, future extensions of this algorithm should consider a more accurate selection of papers that will compose the seeds of the next iteration.

5. Conclusions and Future Work

Because a systematic study demands the cooperation of many researchers during the process and it is quite costly and error-prone,a tool that supports part or the whole process of conducting systematic studies seems to be interesting. In this context, Ramani, which is a collaborative web system tool, appears as a solution for such a demand. This work presented an automatic snowballing algorithm helpful for supporting the selection phase in Ramani. The goal of the automatic snowballing algorithm is to return only the most relevant papers in the context of a systematic review.

The evaluation of the proposed algorithm was based on data collected during a graduate class about CSCW that took place on the Spring of 2012. In that class, 10 participants selected, in a peer review process, 31 papers about collaborative software engineering, published until 2012 in the ACM CSCW Conference. From these papers, we selected three sets of seeds for assessment, the papers published in 1986 (the beginning of the series), 2006 (the middle of the series) and 2012 (the end of the series). After four iterations of executions for each set of seeds, the algorithm did not achieve a 100% of accuracy. Moreover, it returned better results with the seeds of the middle of the series, which helped find 15 new papers.

In the future, in order to get a broader assessment of algorithm, we plan to evaluate other seeds. Such an assessment may help us find what we call *golden seeds* for a given context (research topic or area). These are the seeds with better accuracy (precision and recall) and efficiency (performance). Moreover, others extensions for the automatic (or semi-automatic) snowballing approach may be developed, for instance, an interative snowballing approach or a filter based on an ontology to better select the references and citations. Finally, we plan to extend the support of crawler to other digital libraries.

References

da Silva, F. Q. B., Suassuna, M., França, A. C. C., Grubb, A. M., Gouveia, T. B., Monteiro, C. V. F., and dos Santos, I. E. (2012). Replication of empirical studies in software engineering research: a systematic mapping study. *Empirical Software Engineering*, 19(3):501–557.

de Souza, R., Lopes, C., Bezerra, F., and de Souza, C. R. B. (2013). Ramani: Uma ferramenta de apoio à colaboração durante a execução de estudos sistemáticos. In *Proceedings of the X Brazilian Symposium in Collaborative Systems*, SBSC '13, pages 144–147, Manaus, Amazonas, Brazil. Sociedade Brasileira de Computação.

Dieste, O. and Padua, O. (2007). Developing search strategies for detecting relevant experiments for systematic reviews. In *Empirical Software Engineering and Measurement, 2007.*, ESEM 2007, pages 215–224, Madrid. IEEE.

Felizardo, K. R., MacDonell, S. G., Mendes, E., and Maldonado, J. C. (2012). A systematic mapping on the use of visual data mining to support the conduct of systematic literature reviews. *Journal of Software*, 7(2):450–461.

Jalali, S. and Wohlin, C. (2012). Systematic literature studies: database searches vs. backward snowballing. In *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*, ESEM '12, pages 29–38, Lund, Sweden. ACM-IEEE.

Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., and Linkman, S. (2009). Systematic literature reviews in software engineering: a systematic literature review. *Information and Software Technology*, 51(1):7 – 15.

Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical report, Keele University and Durham University Joint Report. Tech. Rep. EBSE 2007-001.

Petersen, K., Feldt, R., Mujtaba, S., and Mattsson, M. (2008). Systematic mapping studies in software engineering. In *International Conference on Evaluation and Assessment in Software Engineering*.

Runeson, P. and Skoglund, M. (2009). Reference-based search strategies in systematic reviews. In *The Proceedings of the 13th International Conference on Evaluation and Assessment in Software Engineering*, Durham, England. British Computer Society.

Scannavino, K. R. F. (2012). *Evidence-based software engineering: systematic literature review process based on visual text mining*. PhD thesis, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos.

Webster, J. and Watson, R. T. (2002). Analyzing the past to prepare for the future: writing a literature review. *Management Information Systems Quarterly*, 26(2):13–23.

Wieringa, R., Maiden, N., Mead, N., and Rolland, C. (2006). Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements Engineering*, 11(1):102–107.

paper:83

Uma Abordagem para Recomendação no apoio à Modelagem de Processos de Negócio

Mateus Barcellos Costa e Thanner Soares Silva

Lab. de Engenharia de Software e Comércio Eletrônico - SEEC
Informática - Instituto Federal do Espírito Santo
Serra – ES – Brasil

mcosta@ifes.edu.br, thannersoares@gmail.com

Abstract. *In this paper, a methodology for Business Process Modeling support through Recommendation based on Similarity of Behavior is discussed and a set of techniques to support this methodology is provided. These techniques include the determination of the modeling solution space based on processe's declarative specifications and the generation of recommendations by using this solution space as the reference. A prototype to support the methodology was developed and performed tests have shown that recommendations can be obtained in a computationally efficient way. The methodology also shows feasible to accommodate different modeling concerns in the recommendation process.*

Resumo. *Neste artigo, uma metodologia para o apoio à Modelagem de Processos de Negócio por meio de Recomendação baseada em Similaridade de Comportamento é discutida e um conjunto de técnicas para apoiar esta metodologia é apresentado. Estas técnicas incluem a obtenção do Espaço de Soluções da modelagem por meio de especificações declarativas de processos e a geração de recomendações considerando este espaço de soluções como referencial. Um protótipo para apoiar a metodologia foi desenvolvido e testes realizados com o mesmo mostram que recomendações podem ser obtidas de forma computacionalmente eficiente. A metodologia também mostrou-se adequada à acomodação de diferentes preocupações de modelagem no processo de recomendação.*

1. Introdução

A Modelagem de Processos de Negócio permite definir e ilustrar como as atividades em um sistema devem ser conduzidas para que seus objetivos sejam alcançados [Aguilar-Saven 2004]. Embora a finalidade principal da modelagem seja, tipicamente, estabelecer um ordenamento apropriado para a execução das atividades envolvidas nas operações do sistema, a noção do que significa *apropriado* está vinculada a diferentes preocupações. Dentre estas preocupações encontram-se, por exemplo, a conformidade do modelo com relação a restrições do sistema [Sadiq et al. 2007], relações de custo/benefício [Barba et al. 2013], a mitigação de riscos [Lakshmanan et al. 2011] e o grau de flexibilidade [Gröner et al. 2013].

Para garantir a criação de modelos efetivos no contexto de ambientes complexos (ex. uma planta industrial, um hospital), as informações a respeito dessas preocupações devem chegar ao modelador fornecendo a este um referencial para modelagem, por exemplo, por meio de conhecimento tácito ou de um conjunto de documentos descritivos do

processo. Entretanto, a medida que aumentam estas preocupações e a complexidade do sistema, a obtenção desse referencial fica mais difícil [Chan et al. 2011]. Decidir entre as diferentes opções de modelagem torna-se então um problema complexo e trabalhoso.

Neste contexto, o uso de Sistemas de Recomendação surge como uma alternativa capaz de fornecer, ao modelador, direcionamento sem perda de flexibilidade. De acordo com a literatura pesquisada, tais sistemas têm buscado realizar recomendações de fragmentos de processos, e levam em conta a estrutura desses fragmentos para verificar se são adequados [Koschmider et al. 2011, Wang e Wu 2011, Li et al. 2014]. Alternativamente, neste trabalho discutimos uma metodologia para obtenção desses sistemas baseada em similaridade de comportamento [Dijkman et al. 2011]. Esta metodologia utiliza um conjunto de técnicas para especificar o comportamento aceitável de um processo, de forma a determinar o conjunto de modelos aceitáveis para o mesmo (espaço de soluções), e obter recomendações, tendo este espaço de soluções como referencial.

O restante do artigo está organizado da seguinte maneira: na Seção 2, é discutida a ideia de recomendação baseada em similaridade de comportamento e são analisados os modelos adotados na metodologia proposta. Na Seção 3, são introduzidas as regras Especificações de Restrições, utilizadas para especificação preliminar do espaço de soluções da modelagem. Na Seção 4, são apresentados os mecanismos utilizados para o mapeamento de regras Especificações de Restrições em Sistemas de Transições. Na Seção 5, o método de obtenção de recomendações é apresentado e são discutidos aspectos de implementação da metodologia. Na Seção 6, o trabalho é concluído e trabalhos futuros são apresentados.

2. Recomendação baseada em Similaridade de Comportamento

A aplicação do conceito de Recomendação [Resnick e Varian 1997], na modelagem de processos de negócio, é recente. Neste contexto, o termo recomendação tem sido utilizado para caracterizar sistemas interativos, de apoio ao processo de modelagem, que propiciam, por exemplo, a indicação de estruturas de modelagem corretas [Betz et al. 2006], a adequação a padrões e a fragmentos de processo considerados apropriados [Koschmider et al. 2011, Wang e Wu 2011, Li et al. 2014], e a alocação de recursos e serviços[Chan et al. 2011]. De acordo com a literatura pesquisada, as abordagens para obtenção destas recomendações têm se baseado, principalmente, em comparações entre fragmentos de modelos de processos, as quais recaem sobre técnicas de Similaridade Estrutural entre modelos [Dijkman et al. 2011].

A alternativa para obtenção de recomendações que propomos é por meio da análise de Similaridade de Comportamento [Dijkman et al. 2011]. O processo de recomendação, neste caso, implica em capturar o comportamento do processo que se deseja obter e, considerando este comportamento, dado um modelo de processo em construção, recomendar próximos passos de modelagem para o mesmo. A análise de similaridade de comportamento pode ser uma alternativa oportuna para a recomendação ao se considerar as hipóteses de que, no contexto de modelagem: (1) não haverá, *a priori*, um modelo ou fragmento de modelo, com estrutura específica, que venha a servir como referência para o processo de recomendação; (2) recomendações possam ser obtidas a partir de descrições de restrições e dos aspectos desejáveis que o processo deve respeitar e atender durante a sua execução. Neste caso, tais descrições definem, portanto, não uma estrutura de mo-

delo, mas o comportamento desejado para o processo. A título de simplificação, ao longo deste artigo o termo *processo desejado* será utilizado para denominar estas descrições e o termo *modelo candidato* será usado para denominar modelos que estão sendo criados para representar os processos desejados.

Uma técnica que permite determinar similaridade de comportamento entre dois processos é a Bissimulação ou Equivalência Observável [Milner 1989]. Bissimulação é uma relação que quando observada implica que dois processos possuem a mesma semântica operacional [De Nicola e Vaandrager 1995]. Ou seja, os processos são equivalentes do ponto de vista de um observador externo. A verificação de bissimulação fundamenta-se na análise do comportamento dos processos, a qual pode ser feita utilizando-se representações de processos baseadas em sistemas de transições [Finkel e Schnoebelen 2001].

Na abordagem proposta, a relação de bissimulação deve ser verificada, não entre o processo desejado e o modelo em construção, mas sim entre um comportamento aceitável pelo processo desejado e o comportamento do modelo em construção. Este aspecto se justifica pelo fato de que, em geral, o processo desejado apresenta-se como uma descrição baseada no domínio do problema, para a qual um ou mais comportamentos aceitáveis podem ser verificados. Para ilustrar esta situação, considere um processo desejado constituído de duas atividades $A\ e\ B$, as quais podem executar em qualquer ordem sem a imposição de nenhuma restrição ou exigência. Neste caso, são comportamentos aceitáveis $A \rightarrow B$ (A executa antes de B), $B \rightarrow A$ (B executa antes de A) e $A|B$ (A e B executam simultaneamente). Logo, para que se tenha um modelo de processo válido, basta que apenas um dos três comportamentos aceitáveis seja verificado no modelo candidato.

2.1. Sistemas de Transições

Um sistema de transições é uma estrutura contendo um conjunto de estados e um conjunto de transições. Esta estrutura pode conter ainda elementos adicionais, tais como, estados iniciais, estados finais, relações causais e durações associadas às transições [Finkel e Schnoebelen 2001]. Qualquer série de transições válidas, iniciada a partir de um estado inicial, corresponde a um caminho de execução do sistema. A partir da análise desses caminhos pode-se inferir o comportamento do processo representado pelo mesmo. Para efeito desta análise, embora o limite superior do número de estados e transições seja exponencial com relação ao número de tarefas do processo, observa-se com facilidade que na modelagem de processos, quando exercida de forma pragmática, as restrições do sistema modelado tendem a reduzir significativamente o número de estados alcançáveis e transições.

Sistemas de transições são representáveis por meio de matrizes de transições. Uma matriz de transições T é uma matriz $2^n \times 2^n$, onde n é o número de tarefas do processo e 2^n é o número de estados do modelo. Cada estado é representado por uma cadeia binária de tamanho n. Para o estado $e_x = [t_n..t_1]$, $t_i = 0$ indica que a tarefa t_i não se encontra executada em e_x; $t_i = 1$ indica que a tarefa t_i está executada no estado e_x. As transições são indicadas pelos valores das células da matriz. A título de ilustração, considere um processo com 3 tarefas $(t1,\ t2\ e\ t3)$ onde a tarefa t2 só pode ser executada após a tarefa t1 e a tarefa t3 somente após a tarefa t2. Para este processo o sistema de transições é

$$\mathrm{T} = \begin{array}{c} \\ 000 \\ 001 \\ 011 \end{array} \begin{array}{c} \begin{array}{ccc} 001 & 011 & 111 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

Figura 1. Matriz de transições T.

representável pela matriz T da Figura 1. Esse exemplo ilustra que matrizes de transições são representáveis por matrizes esparsas[1]. Quanto maior o número de restrições de um processo, mais esparsa será T. Novamente, considerando essa relação, pode-se induzir que matrizes de transições são representáveis de forma computacionalmente eficiente.

2.2. Representação de Processos Desejados e Modelos Candidatos

A representação de processos por meio de sistemas de transições permite a análise do comportamento de processos de forma computacionalmente eficiente (ver Sec. 2.1). Entretanto, tal representação não é conveniente para o consumo humano sendo necessários outros modelos para a descrição de processos desejados e modelos candidatos, bem como a construção de mapeamentos entre estes modelos e sistemas de transições.

Para a descrição de processos desejados optou-se pelo uso de modelos declarativos. Um modelo declarativo de um processo pode ser definido por meio de um conjunto de *tarefas*[2] e um conjunto de restrições sobre estas tarefas que devem ser respeitadas pelos caminhos de execução do processo. Exemplos destas restrições são relações de dependência de execução ou de não-coexistência entre tarefas e estruturas de repetição. O modelo declarativo adotado, discutido na Seção 3, foi denominado de Especificação de Restrições.

Considerou-se que modelos candidatos serão construídos utilizando-se alguma notação ou linguagem de modelagem imperativa. Em adição, a modelagem foi restringida a processos de negócio estruturados [Kiepuszewski et al. 2000]. Um processo de negócio estruturado é um processo representável por meio de um diagrama de workflow no qual, para todo elemento de processamento ou desvio contendo n fluxos de saída, existe um elemento correspondente contendo n fluxos de entrada. A título de simplificação, consideramos ainda que estes processos podem ser representados por um grafo onde os nós representam tarefas (elementos de processamento) e desvios, e os arcos são as relações de fluxo de controle. Esta simplificação, entretanto, não restringe a abordagem adotada dado que é possível, para a análise de comportamento, a representação de modelos contendo especializações e estruturas complementares utilizando esta forma[3].

3. Especificações de Restrições

Modelos declarativos têm sido utilizados, principalmente, na modelagem de processos flexíveis, que necessitam ser constantemente replanejados, e de processos que não seguem rigorosamente um modelo de workflow (processos não estruturados) [Pesic et al. 2007].

[1]Os estados inalcançáveis foram omitidos.

[2]Conforme discutido ainda nesta seção, adotaremos neste trabalho uma visão simplificada de processos onde diferentes tipos de atividades são vistos indistintamente como tarefas.

[3]Simplificações similares tem sido utilizadas na literatura, por exemplo no trabalho de Polyvyanyy [Polyvyanyy et al. 2012].

De acordo com a literatura estudada, este modelos têm sido aplicados, principalmente, para complementar a modelagem de processos específicos apoiando a descrição de aspectos que não podem ser modelados por modelos imperativos. Como a aplicação desses modelos em nossa abordagem não compartilha este mesmo objetivo, a título de prova de conceito, optamos pela definição de uma notação própria para as regras de Especificações de Restrições (ER).

A notação proposta foi inspirada em padrões de Workflow típicos como, por exemplo, dependências de execução, não coexistência e laços. Também foram definidas regras para expressar comportamentos alternativos. Para este fim foram introduzidas as regras de Chave, Exclusão Absoluta e Módulo. Foi também introduzida a regra de Dependência Circunstancial, apropriada a diversos cenários de modelagem analisados. As regras de ER são apresentadas resumidamente na Tabela 1, considerando que o processo possui um conjunto de tarefas T.

Regra	Sintaxe	Descrição	Exemplos
Dependência Simples	$D : t_x - T'$	A tarefa t_x depende da execução prévia do subconjunto de tarefas $T' \subset T$.	$D : t_3 - t_1\ t_2$, $D : t_2 - t_1$
Dependência Composta	$D : t_x - T' \vee T'' \vee \ldots$	A tarefa t_x depende da execução de ao menos um subgrupo de tarefas de T.	$D : t_3 - T_1 \vee T_2$, $D : t_3 - t_1 \vee t_2$
Dependência Circunstancial	$D : t_x - T'*$	Se o subconjunto $T' \subset T$ vier a aparecer no modelo, então a tarefa t_x deve, obrigatoriamente, depender de sua execução.	$D : t_2 - t_1*$
OU	$O : T' - T'' - \ldots$	Ao menos o subconjunto T' ou T'' deve ser executado em um dos caminhos de execução permitidos pelo modelo. Ou seja, um modelo contendo um caminho de execução que gere uma das sequências verdadeiras do OU deve ser aceito.	$O : t_1 - t_2\ t_3$, $O : t_1 - t_2 - t_3$
Escolha	$C : T' - T'' - \ldots$	Os subconjuntos de tarefas $T' - T'' - \ldots$ contidos em T são disjuntos e ambos não podem vir a ocorrer em um mesmo caminho de execução permitido pelo modelo.	$C : t_1 - t_2\ t_3$, $C : t_1 - t_2 - t_3$
Exclusão Absoluta	$A : T' - T'' - \ldots$	O subconjunto T' não pode estar em um mesmo modelo com subconjunto $T'' \subset T - T'$. T' e T'' são conjuntos disjuntos.	$A : t_1 - t_2$, $A : t_1 - t_2 - t_3$
Chave	$K : T'$	Uma regra de chave indica que o subconjunto de tarefas $T' \subset T$ deve ser executado para o processo atingir um estado final. Cada caminho de execução permitido pelo modelo deve ser consistente com pelo menos uma regra de chave.	$K : t_1\ t_2\ t_3\ t_4$, $K : t_1$
Laço	$L : T' - t_x$	Após a execução de alguma tarefa $t \in T'$ a tarefa t_x deve ser re-executada.	$L : t_2 - t_1$
Módulo	$M : T'$	A regra de Módulo permite explicitar um conjunto mínimo de tarefas T' que deve existir em um modelo.	$M : t_1\ t_2\ t_3\ t_4$, $M : t_1$

Tabela 1. Regras de Especificação de Restrições.

Devido a limitações de espaço, as regras de especificação de restrições não serão discutidas neste artigo de maneira aprofundada. Entretanto, o detalhamento das mesmas, assim como um guia de utilização e exemplos ilustrativos podem ser obtidos no relatório técnico de apoio [Silva e Costa 2014]. Na próxima seção introduzimos o método utilizado para realizar o mapeamento entre ERs e sistemas de transições.

4. Obtenção do Espaço de Soluções como um Sistema de Transições

Um sistema de transições fornece uma representação expandida do espaço de soluções, representado por uma ER, que explicita todos os caminhos de execução permitidos pelo processo desejado. Para a obtenção desses caminhos, o método proposto realiza, num primeiro passo, a combinação sistemática das restrições definidas por uma ER, com o objetivo de descrever as relações de dependência e exclusão específicas de cada caminho de execução. Estas dependências são representadas por uma matriz de restrições R conforme definição apresentada na Figura 2.

$R = [r_{ij}]_{N \times N};\ r_{ij} \in \{0, 1, 2\},\ 1 \leq i, j \leq N$, onde N é o número de tarefas do processos e,

para um conjunto de tarefas $T = \{t_1 \ldots t_N\}$, temos que $r_{ij} = \begin{cases} 0, & \text{se } t_i \text{ não depende de } t_j; \\ 1, & \text{se } t_i \text{ depende de } t_j; \\ 2, & \text{se } t_i \text{não é executada no caminho de execução.} \end{cases}$

Figura 2. Definição da Matriz de Restrições.

4.1. Geração da Árvore de Restrições

No método proposto, a obtenção do sistema de transições equivalente a uma ER requer a obtenção de todas as matrizes de restrições correspondentes aos diferentes caminhos de execução permitidos pela ER, de maneira unificada, sem adicionar inconsistências nem promover a perda de informação. Para tanto, as matrizes de restrições são geradas e dispostas em uma estrutura denominada de Árvore de Restrições. A idéia básica da utilização dessa estrutura é a obtenção de matrizes que representem as combinações das dependências, relativas a um caminho de execução, obtidas ao se processar as regras da ER. Na árvore de restrições isto se reflete ao se derivar novas matrizes de restrições a partir do conjunto atualizado de matrizes de restrições adicionando a estas, novas restrições oriundas do processamento da ER. Ao final desse processo será produzida uma árvore de matrizes de restrições cujos nós finais (nós folha) corresponderão às matrizes de restrições equivalentes à ER.

Para ilustrar o método de geração de árvores de restrição utilizaremos a ER ilustrada na Figura 3. A obtenção de uma árvore de restrições inicia-se com uma matriz com

1. D 2 - 1	5. K 4
2. D 3 - 1	6. L 4 - 1
3. D 4 - 2 V 3	7. M 1 2 4
4. A 2 - 3	8. M 1 3 4

Figura 3. Exemplo de ER.

zero restrições, na qual são registradas todas as relações de dependência simples. Em seguida, para cada regra de Dependência Composta, OU , Escolha, Exclusão Absoluta e Dependência Circunstancial, tomando-se cada nó folha da árvore, derivam-se novas matrizes introduzindo as restrições correspondentes à regra. Por exemplo, a regra 3 da ER da Figura 3 possui duas relações de dependência (4-2 e 4-3), as quais geram as matrizes do nível 2 da árvore de restrições (Figura 4).

4.2. Geração da Matriz do Sistema

Uma matriz do sistema (MS), é uma matriz de transições que reflete todos os modelos de processo aceitáveis pela ER que a originou. Nela, as transições são identificadas de acordo com o vocabulário σ conforme definição apresentada na Figura 5.

A geração de uma MS inicia-se por uma matriz $2^N \times 2^N$ com zero transições. No primeiro passo, o processo insere as transições possíveis a partir do estado 0 (zero) observando as matrizes de restrições (nós folha da árvore de restrições). A matriz MS_1 (Figura 6) representa esse passo para a árvore de restrições da Figura 4. Neste processo, para cada matriz de restrições de entrada é gerada uma matriz intermediária[4].

[4]Para uma melhor visualização das matrizes, os estados inalcançáveis e os zeros (transições inexistentes) foram omitidos.

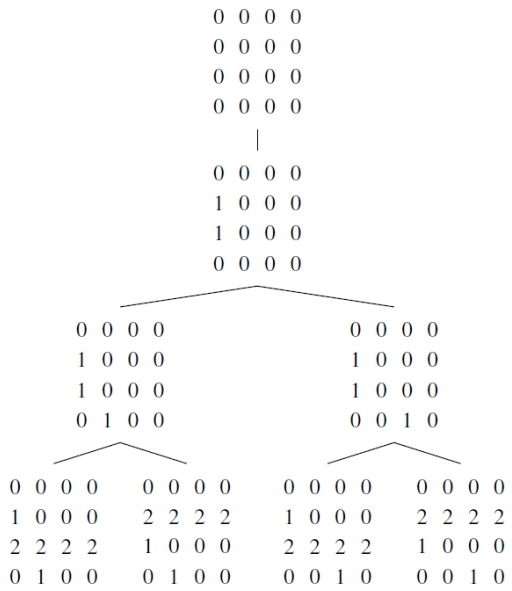


Figura 4. Árvore de Restrições da ER da tabela 3.

$$MS = [s_{ij}]_{2^N \times 2^N};\ 1 \leq i, j \leq N, N \in \mathbb{N}^+,$$
$$s_{ij} = \{a_1, a_2, \ldots, a_k\}, a_i \in \sigma, \text{ onde}$$

$$\sigma = \begin{cases} 0, & \text{se não há transição entre os estados } i \text{ e } j; \\ 1, & \text{se há uma transição entre os estados } i \text{ e } j; \\ F, & \text{há uma transição entre os estados } i \text{ e } j \text{ e esta transição é final;} \\ L, & \text{há uma transição entre os estados } i \text{ e } j \text{ e esta transição é um laço de retorno.} \end{cases}$$

Figura 5. Definição Matriz do Sistema.

$$\text{MS}_1 = \begin{array}{c} \\ 0000 \\ 1000 \\ 1010 \\ 1100 \end{array} \begin{array}{c} \begin{array}{ccccc} 1000 & 1010 & 1011 & 1100 & 1101 \end{array} \\ \begin{bmatrix} 1 & & & & \\ & 1 & & 1 & \\ & & 1 & & \\ & & & & 1 \end{bmatrix} \end{array} \qquad \text{MS}_2 = \begin{array}{c} \\ 0000 \\ 1000 \\ 1010 \\ 1011 \\ 1100 \\ 1101 \end{array} \begin{array}{c} \begin{array}{ccccc} 1000 & 1010 & 1011 & 1100 & 1101 \end{array} \\ \begin{bmatrix} 1 & & & & \\ & 1 & & 1 & \\ & & F & & \\ L & L & & & \\ & & & & F \\ L & & & L & \end{bmatrix} \end{array}$$

Figura 6. Matriz Parcial e Matriz do Sistema Final.

Posteriormente, estas matrizes são unificadas em uma única matriz. Finalmente, sobre a matriz resultante são aplicadas as regras que ficaram de fora da montagem da árvore de restrições, que são as regras de laço e regras de chave. As Regras de Chave permitem identificar as transições finais. A matriz MS_2 (Figura 6) é a matriz do sistema completa para a ER da Tabela 3. Exemplos maiores de geração de matrizes do sistema, bem como o algoritmo utilizado para sua construção e o detalhamento de seu funcionamento são apresentados em [Silva e Costa 2014].

5. Processo de Recomendação Interativa

Para que seja possível gerar recomendações de modelagem, o comportamento do modelo candidato é mapeado interativamente em uma matriz de transições binária que denominamos Matriz do Processo. O processo de recomendação ocorre por meio da análise da matriz do processo e de buscas por novas transições válidas considerando uma matriz do sistema. Ao se detectar estas transições, o modelo equivalente é recomendado.

Para ilustrar o processo de recomendação, considere a matriz MS_2 da Figura 6. A partir de um modelo candidato inicialmente vazio o processo inicia-se determinando as transições possíveis a partir do estado 0000. Conforme mostra a matriz MS_2, este conjunto de transições leva à recomendação da inclusão da tarefa T_1 no modelo. Com a inclusão dessa tarefa pelo modelador, a matriz do processo equivalente será a matriz MP_1 (Figura 7). Em seguida, os novos estados alcançáveis são checados e novas recomendações são geradas. No exemplo, as recomendações correspondem à inclusão das tarefas T_2 e T_3, correspondentes aos estados alcançáveis e considerando ainda as tarefas já existentes no modelo. Com a inserção da tarefa T_3 (matriz MP_2 da Figura 7), e da tarefa T_4 conectada a um fluxo divergente da tarefa T_3, teremos um sistema de transições equivalente à matriz do processo MP_3.

Caso o modelador adicione a tarefa T_4 a partir de um fluxo divergente de T_1 a matriz de processo correspondente seria MP_4 (Figura 7). Neste caso, como a transição incorporada não existe na matriz do sistema, o processo informa ao modelador que o modelo possui um defeito. Durante o processo de recomendação, as regras de exclusão absoluta são verificadas por meio da análise da matriz do processo. A medida que o modelo candidato vai sendo alterado, o processo verifica se não existem caminhos de execução com transições correspondentes à execução de tarefas que estejam em uma regra de Exclusão Absoluta. O processo permite também obter recomendação com base em propriedades do modelo completo. Neste processo é verificado: (a) se todos os caminhos levam a um estado final e, (b) se alguma regra de módulo foi satisfeita.

$$
\begin{array}{c}
\begin{array}{c} \\ 0000 \end{array}
\begin{array}{c} \begin{array}{c} 1000 \end{array} \\ \left[\begin{array}{c} 1 \end{array}\right] \end{array} \\
MP_1
\end{array}
\qquad
\begin{array}{c}
\begin{array}{c} \\ 0000 \\ 1000 \end{array}
\begin{array}{c} \begin{array}{cc} 1000 & 1010 \end{array} \\ \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right] \end{array} \\
MP_2
\end{array}
\qquad
\begin{array}{c}
\begin{array}{c} \\ 0000 \\ 1000 \\ 1010 \end{array}
\begin{array}{c} \begin{array}{ccc} 1000 & 1010 & 1011 \end{array} \\ \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \end{array} \\
MP_3
\end{array}
\qquad
\begin{array}{c}
\begin{array}{c} \\ 0000 \\ 1000 \\ 1010 \end{array}
\begin{array}{c} \begin{array}{cccc} 0001 & 1000 & 1010 & 1011 \end{array} \\ \left[\begin{array}{cccc} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array} \\
MP_4
\end{array}
$$

Figura 7. Matriz do Processo 1, 2, 3 e 4.

5.1. Recomendação baseada em Custos

A matriz do sistema permite realizar, com facilidade, decomposições e especificações, em termos de seus estados e transições, de valores de métricas que possam estar relacionadas às preocupações de modelagem. Com base nesta possibilidade, como parte dos experimentos foi implementado um modelo de custos associados a tarefas. Neste modelo, a execução ou não de uma tarefa impacta no custo de execução das outras tarefas [Teixeira et al. 2014]. Ou seja, dado que um estado na matriz do sistema representa o conjunto de tarefas executadas até o momento, o custo de uma tarefa é uma função do estado a partir do qual ocorre a sua execução. Ao se integrar tal modelo à matriz do sistema, pode-se diferenciar transições com relação a seu custo e, por conseguinte, recomendar as disposições de tarefas considerando o modelo de custos fornecido. Analogamente, outras métricas como, por exemplo, grau de risco e tempo, podem ser integrados ao mecanismo de recomendação.

5.2. Aspectos de Implementação

Os algoritmos de geração da matriz do sistema e o processo de validação foram implementados como um protótipo utilizando o software de computação numérica Scilab. Foram realizados testes de validação funcional e experimentos para verificação do método

em termos de desempenho. A Figura 8 ilustra o resultado de um destes experimentos. Cada curva no gráfico representa o tempo de geração da matriz do sistema para um dado número de tarefas (de 1 a 10), e um número de restrições de dependência simples fixo, de 0 a 9. Nota-se a redução de tempo expressiva com o aumento do número de restrições.

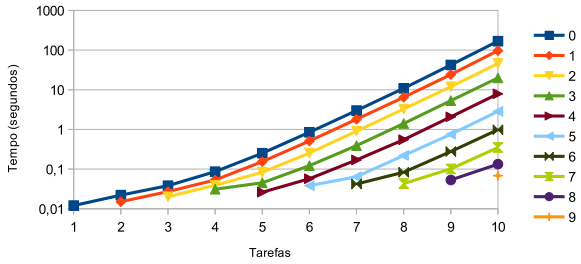


Figura 8. Desempenho do método de geração da Matriz do Sistema.

6. Conclusão

A modelagem de Processos de Negócio é uma atividade cognitiva complexa que envolve a percepção, a memória e a compreensão da realidade constituída pelos diversos elementos que determinam a essência do fenômeno ou sistema que está sendo modelado. Para apoiar esta atividade, neste trabalho foi apresentada uma metodologia para obtenção de recomendações de modelagem. A metodologia proposta, baseada em similaridade de comportamento, visa capturar e recomendar ações de modelagem, considerando o comportamento dos processos desejados, e permitir a acomodação de diferentes preocupações de modelagem no processo de recomendação. Pudemos observar, no estudo realizado, que por meio de uma representação e análise eficiente desse comportamento, no caso, por meio de sistemas de transições, é possível determinar recomendações a respeito da estrutura de modelos imperativos candidatos. Foi possível ainda integrar, ao processo de recomendação, um modelo de custos para tarefas, preservando a mesma técnica de recomendação. O uso de similaridade de comportamento mostrou-se importante para este fim pois permite introduzir, com facilidade, os aspectos dinâmicos das métricas relacionadas às preocupações de modelagem. O trabalho mostrou-se promissor e diversas iniciativas estão sendo realizadas a título de trabalhos futuros, dentre elas, o estudo de recomendações orientadas ao gerenciamento de riscos, o apoio à modelagem de processos descentralizados por meio de recomendações e a integração do protótipo desenvolvido a um ambiente gráfico de modelagem de processos.

Referências

Aguilar-Saven, R. S. (2004). Business Process Modelling: Review and Framework. *International Journal of Production Economics*, 90(2):129–149.

Barba, I., Del Valle, C., Weber, B., and Jiménez, A. (2013). Automatic Generation of Optimized Business Process Models from Constraint-Based Specifications. *International Journal of Cooperative Information Systems*, 22(02).

Betz, S., Klink, S., Koschmider, A., and Oberweis, A. (2006). Automatic user support for business process modeling. In *Proceedings of the Workshop on Semantics for Business Process Management*, pages 1–12.

Chan, N. N., Gaaloul, W., and Tata, S. (2011). Context-based service recommendation for assisting business process design. In *E-Commerce and Web Technologies*, pages 39–51. Springer.

De Nicola, R. and Vaandrager, F. (1995). Three Logics for Branching Bisimulation. *Journal of the ACM (JACM)*, 42(2):458–487.

Dijkman, R., Dumas, M., Van Dongen, B., Käärik, R., and Mendling, J. (2011). Similarity of Business Process Models: Metrics and Evaluation. *Information Systems*, 36(2):498–516.

Finkel, A. and Schnoebelen, P. (2001). Well-Structured Transition Systems Everywhere! *Theoretical Computer Science*, 256(1):63–92.

Gröner, G., Bošković, M., Silva Parreiras, F., and Gašević, D. (2013). Modeling and Validation of Business Process Families. *Information Systems*, 38(5):709–726.

Kiepuszewski, B., ter Hofstede, A. H. M., and Bussler, C. J. (2000). On Structured Workflow Modelling. In *Advanced Information Systems Engineering*, pages 431–445. Springer.

Koschmider, A., Hornung, T., and Oberweis, A. (2011). Recommendation-based Editor for Business Process Modeling. *Data & Knowledge Engineering*, 70(6):483–503.

Lakshmanan, G. T., Duan, S., Keyser, P. T., Curbera, F., and Khalaf, R. (2011). Predictive Analytics for Semi-structured Case Oriented Business Processes. In *Business Process management Workshops*, pages 640–651. Springer.

Li, Y., Cao, B., Xu, L., Yin, J., Deng, S., Yin, Y., and Wu, Z. (2014). An Efficient Recommendation Method for Improving Business Process Modeling. *Industrial Informatics, IEEE Transactions on*, 10(1):502–513.

Milner, R. (1989). *Communication and Concurrency*. Prentice-Hall, Inc.

Pesic, M., Schonenberg, H., and van der Aalst, W. M. (2007). Declare: Full support for Loosely-Structured Processes. In *Enterprise Distributed Object Computing Conference, 2007. EDOC 2007. 11th IEEE International*, pages 287–287. IEEE.

Polyvyanyy, A., García-Bañuelos, L., and Dumas, M. (2012). Structuring Acyclic Process Models. *Information Systems*, 37(6):518–538.

Resnick, P. and Varian, H. R. (1997). Recommender Systems. *Communications of the ACM*, 40(3):56–58.

Sadiq, S., Governatori, G., and Namiri, K. (2007). Modeling Control Objectives for Business Process Compliance. In *Business process management*, pages 149–164. Springer.

Silva, T. S. and Costa, M. B. (2014). Tutorial de Apoio à Recomendação na Modelagem de Processos de Negócios. [Online] Disponível em: http://www.sr.ifes.edu.br/∼seec/recomendacao/Tutorial.

Teixeira, R., Miranda, G. M., Vencioneck, L., Silva, T. S., Filho, J. M., and Costa, M. B. (2014). Caracterização de Processos de Negócio utilizando Sistemas de Transições. In *Anais do VIII Workshop Brasileiro de Gerência de Processos de Negócio*.

Wang, H. J. and Wu, H. (2011). Supporting Process Design for e-Business via an Integrated Process Repository. *Information Technology and Management*, 12(2):97–109.

paper:9

Uso de SGBDs NoSQL na Gerência da Proveniência Distribuída em *Workflows* Científicos

Guilherme R. Ferreira, Carlos Filipe Jr. e Daniel de Oliveira

Instituto de Computação – Universidade Federal Fluminense (UFF)
{guilherme_rangel, cfmtjunior}@id.uff.br, danielcmo@ic.uff.br

Resumo. *Um fator fundamental na gerência de experimentos modelados como workflows científicos são seus dados de proveniência. Esses dados basicamente são usados para garantir a reprodutibilidade, porém nos últimos anos eles também vêm sendo usados para tarefas de monitoramento e escalonamento de atividades. Como essas tarefas demandam consultas em tempo real, conforme a quantidade de dados de proveniência aumenta, mecanismos eficazes para armazenamento e consulta se fazem necessários. Uma das opções mais comuns é utilizar os SGBDs relacionais para gerenciar a proveniência, dada a tradição da tecnologia. Porém, novas tecnologias como os SGBDs NoSQL tem ganhado bastante atenção nos últimos anos e podem ser de grande valia nesse cenário, principalmente em ambientes distribuídos onde escalabilidade é essencial. Este artigo realiza um estudo comparativo entre SGBDs relacionais e um SGBD NoSQL (Cassandra) no que tange a gerência dos dados de proveniência. Apresentamos um estudo com um workflow real de bioinformática usando a máquina de workflows para nuvens SciCumulus.*

1. Introdução

Os experimentos científicos com base em simulações computacionais são comumente compostos por várias invocações de programas em sequência, cada uma das quais com seu próprio conjunto de parâmetros e dados de entrada (Mattoso *et al.* 2010). Esses experimentos podem ser modelados como *workflows* científicos e gerenciados pelos chamados Sistemas de Gerência de *Workflows* Científicos (SGWfC). Como esses *workflows* são comumente computacionalmente intensivos e processam uma grande quantidade de dados, eles requerem capacidades de paralelismo e uso de ambientes de processamento de alto desempenho (PAD). Vários SGWfCs apresentam mecanismos de execução paralela em ambientes de PAD (Ogasawara *et al.* 2013, Oliveira *et al.* 2010, Taylor *et al.* 2007) como *clusters* (Dantas 2005), grades (Foster e Kesselman 2004) ou nuvens (Vaquero *et al.* 2009). Além de gerenciarem a execução dos *workflows*, esses SGWfCs coletam informação fundamental para o experimento: os dados de proveniência (Freire *et al.* 2008). Os dados de proveniência representam a história e os caminhos dos dados de entrada, desde o início do experimento até o seu fim, e são fundamentais para garantir a sua consequente reprodução e validação (Freire *et al.* 2008).

[1] O trabalho apresentado nesse artigo foi parcialmente financiado pelo CNPq (Processo 478878/2013-3) e FAPERJ (Processo E-26/111.370/2013)

Nos últimos anos, o uso da proveniência tem sido extrapolado para outras áreas como o monitoramento (Pintas *et al.* 2013), a tolerância a falhas (Costa et al. 2012) e o escalonamento adaptativo de atividades (Oliveira *et al.* 2012). Assim, os SGWfCs que são orientados a proveniência como o SciCumulus (Oliveira *et al.* 2010) e o Chiron (Ogasawara *et al.* 2013) agora necessitam realizar consultas frequentes ao repositório de proveniência durante a execução do *workflow* seja para monitora-lo ou para planejar próximas ações. Diferentemente da análise realizada após a execução do *workflow* (*i.e. post mortem*) a qual não necessariamente preza pela eficiência, as análises realizadas em tempo de execução tendem a requerer uma maior velocidade tanto para armazenar os dados quanto para consultá-los de forma que não atrapalhe a execução paralela do *workflow*. Dessa forma, conforme a escala do *workflow* aumenta (seja na quantidade de atividades ou na quantidade de dados processados) o volume de dados de proveniência gerenciados também aumenta proporcionalmente e mecanismos eficientes de armazenamento e consulta a esses dados se fazem necessários de forma que não interfiram no desempenho da máquina de *workflows*. Mais ainda, *workflows* executados em ambientes distribuídos como as nuvens de computadores podem aumentar a quantidade de recursos computacionais usados na execução do *workflow*, aumentando assim a quantidade de dados de proveniência a serem gerenciados o que faz com que a abordagem escolhida para gerenciar os dados de proveniência seja capaz de aumentar a sua capacidade também.

De fato, esse problema da gerência dos dados de proveniência se assemelha muito aos problemas de *Big Data* (Bertino et al. 2012). O termo *Big Data* é usado para descrever qualquer coleção de conjuntos de dados que são grandes e complexos que se torna difícil trabalhar com esses dados usando as técnicas de processamento de dados tradicionais. E principalmente, em problemas de *Big Data* o aumento e diminuição da demanda em tempo real também é um problema relevante e ainda sem solução definitiva. Assim, podemos considerar a gerência da proveniência distribuída como um desses problemas de *Big Data* já que o volume de dados pode ser grande e a demanda por armazenamento e consulta pode aumentar e diminuir ao longo do tempo. Por exemplo, se considerarmos uma base de dados de proveniência que siga o modelo PROV-Wf (Costa *et al.* 2013, Oliveira *et al.* 2014), para uma base que contenha 10.000 execuções de *workflows*, teremos em média mais de 100.0000 registros de atividades, mais de 1.000.000 de registros de execuções de atividades e mais de 5.000.000 de registros de arquivos gerados. Em relação ao espaço em disco ocupado, essa base não é extremamente grande (12GB), porém como possui muitos registros, as consultas executadas pela máquina de execução podem ser demoradas, o que também caracteriza um problema de *Big Data*.

Atualmente, muitos dos SGWfCs optam por armazenar os dados de proveniência em bancos de dados relacionais. Essa é uma solução natural, uma vez que a tecnologia presente nos SGBDs relacionais está bem sedimentada e estável. Os SGBDs relacionais atuais são bastante eficientes, porém eles possuem a desvantagem de não escalarem facilmente. Ou seja, se a demanda por armazenamento e consulta aumentar rapidamente (*i.e. flash crowd*) não é trivial aumentar a capacidade do SGBD tão rapidamente. Em paralelo, novos SGBDs não-relacionais vem sendo propostos como soluções escaláveis e "vendidos" como soluções ideais principalmente para os problemas de *Big Data*. Os SGBDs não relacionais (a partir de agora chamados somente de NoSQL) não garantem as tradicionais propriedades ACID dos SGBDs relacionais, porém focam em prover a

chamada escalabilidade horizontal (aumento ou diminuição da capacidade de acordo com a demanda). Os SGBDs NoSQL passam a ser uma alternativa atraente para armazenar a proveniência de *workflows* executados em ambientes distribuídos.

Assim, esse artigo apresenta um estudo preliminar sobre o uso de SGBDs NoSQL frente aos tradicionais SGBDs relacionais no que tange a gerência dos dados de proveniência desses *workflows*. O principal aspecto a ser analisado nesse artigo é o impacto no SGWfC quando o tipo de SGBD utilizado é modificado. Apresentamos a migração dos dados de proveniência de um banco de dados no PostgreSQL para o SGBD NoSQL Cassandra, que possui a vantagem de ser mais escalável e eficiente para grandes volumes de dados. O banco de dados de proveniência migrado é o mesmo utilizado pelo SGWfC SciCumulus (Oliveira *et al.* 2010) que executa *workflows* em paralelo em ambientes de nuvem. Além da migração da base de dados para uma base não-relacional, esse artigo também apresenta uma análise comparativa entre os resultados obtidos pela versão anterior no que se refere ao impacto da mudança de tecnologia no desempenho da máquina de *workflow*.

Além da introdução, esse artigo possui outras 4 seções. A Seção 2 apresenta o desenvolvimento e a migração da base de dados de proveniência do PostgreSQL para o Cassandra. A Seção 3 apresenta uma análise dos resultados experimentais. A Seção 4 apresenta os trabalhos relacionados e a Seção 5 conclui este artigo.

2. Mapeamento do Banco de Dados de Proveniência

Nesse artigo, escolhemos o Cassandra como o SGBD NoSQL a ser usado. O Cassandra é um SGBD de natureza distribuída que utiliza o modelo de registros particionados com consistência configurável. Esses registros são armazenados em tabelas chamadas de *Column Families*. Nessas tabelas, a primeira parte da chave primária é chamada chave de partição e serve para definir em qual partição o registro será armazenado. Dentro de uma partição, os registros são agrupados pela chave de agrupamento ou *clustering keys*. Um ponto importante que deve ser ressaltado é a não existência de chaves estrangeiras no modelo do Cassandra. Isso permite que tabelas sejam criadas, removidas ou alteradas em tempo de execução, sem bloqueios para atualizações ou consultas. Desta forma, não existe o conceito de junção e sub-consultas.

O Cassandra distribui os dados de forma transparente, bastando o usuário definir na hora da criação de um *keyspace* em quantos nós (máquinas) os dados devem ser distribuídos e qual política de replicação utilizar. Além disso, o usuário deve definir qual máquina será a *seed* que tem como função armazenar a lista dos nós pertencentes ao conjunto de nós disponíveis para que os outros nós possam encontrá-los. O Cassandra não utiliza SQL como linguagem de consulta. Sendo assim, ele possui a sua própria linguagem de consulta e manipulação de dados chamada CQL (*Cassandra Query Language*). A linguagem possui pontos similares com o SQL, porém possui algumas limitações. Por exemplo, não é possível realizar uma seleção em uma coluna que seja referenciada na cláusula WHERE sem que a mesma faça parte da chave primária ou possua um índice secundário declarado. Ainda em relação à cláusula WHERE, no caso da chave de partição ser composta, sempre que uma de suas partes aparecer na projeção, as outras devem, obrigatoriamente, se fazer presentes.

Existem outras restrições importantes, como por exemplo a restrição do uso dos operadores "=" e "IN" apenas para os campos da chave de partição. Para os outros

campos que não fazem parte da chave de partição, outros operadores lógicos podem ser usados com a exceção de "<>". Existem também os índices secundários, que permitem que a consulta seja realizada sobre algum atributo que não esteja declarado na chave primária, porém quando um atributo que possui um índice secundário aparece em uma consulta, é necessário visitar todos os nós em que a *keyspace* se faz presente, não sendo, portanto, uma consulta otimizada quanto uma realizada apenas sobre os atributos da chave primária. Ou seja, é aconselhável evitar a criação de índices secundários sem que haja real necessidade. Outro ponto de diferença é a questão da ordenação. Apesar de utilizar a cláusula ORDER BY como em SQL, no CQL somente é possível ordenar pelos atributos presentes na chave de agrupamento e, é necessário respeitar sua ordem de declaração.

Dadas as restrições impostas pelo Cassandra, iniciamos o mapeamento do banco de dados proveniência relacional para o modelo do Cassandra. Para tal, escolhemos mapear o banco de dados proveniência do SciCumulus se encontra atualmente no PostgreSQL e é baseado no modelo PROV-Wf (Costa *et al.* 2013) que estende o padrão PROV (Moreau e Missier 2013) do W3C. Logo, é necessário mapear o esquema levando-se em conta as restrições do Cassandra. O modelo utilizado no PostgreSQL foi adaptado de forma que não houvesse perda de semântica. Dessa forma, cada tabela que existia no PostgreSQL foi mapeada para uma *Column Family* no Cassandra, sendo que sempre que houvesse uma restrição como uma das anteriormente citadas, teríamos que adaptar o *script* de criação dos objetos. Devido às restrições de espaço, exemplificaremos apenas uma tabela do sistema, a *hactivation*. Entretanto, o mapeamento completo do banco de dados do SciCumulus para o seu equivalente no Cassandra se encontra disponível em *https://s3.amazonaws.com/SBBD-Cassandra/Mapeamento-PostgreSQL-Cassandra.zip*. A Tabela 1 apresenta o *script* de criação apenas da tabela *hactivation* no PostgreSQL e no Cassandra.

Tabela 1 Mapeamento do *script* de criação da tabela *hactivation*

PostgreSQL:

```
CREATE TABLE hactivation (
  taskid integer NOT NULL DEFAULT,
  actid integer NOT NULL,
  processor integer,
  exitstatus integer,
  commandline text,
  workspace character varying(150),
  terr text,
  tout text,
  starttime timestamp with time zone,
  endtime timestamp with time zone,
  status character varying(25),
  CONSTRAINT htask_pk PRIMARY KEY (actid,
taskid),
  CONSTRAINT htask_actid_fk FOREIGN KEY
(actid)
      REFERENCES hactivity (actid) MATCH
SIMPLE
      ON UPDATE NO ACTION ON DELETE NO
ACTION)
```

Cassandra:

```
CREATE TABLE hactivation (
  id uuid,
  taskid timeuuid,
  actid uuid,
  status text,
  seqnum int,
  processor int,
  exitstatus int,
  commandline text,
  workspace text,
  terr text,
  tout text,
  starttime TIMESTAMP,
  endtime TIMESTAMP,
  extractor text,
  PRIMARY KEY (id, taskid)
);
CREATE INDEX ON hactivation(status);
CREATE INDEX ON hactivation(actid);
CREATE INDEX ON
hactivation(workspace);
CREATE INDEX ON hactivation(seqnum);
```

Na Tabela 1 podemos perceber algumas diferenças importantes. Primeiramente, no Cassandra temos a criação de um UUID (*i.e. Universally Unique Identifer*) que é um

identificador universal que evita que ocorram colisões de identificadores no momento do armazenamento. Basicamente, quando não utilizamos o UUID são realizadas diversas atualizações em uma tabela de metadados para se recuperar o último identificador utilizado de cada tabela, o que acaba impactando no tempo final de execução do *workflow*. Além disso, não é possível executar mais de um *workflow* concorrentemente, pois existe o caso de uma instância do SGWfC consultar a tabela de metadados para resgatar o último identificador e uma outra instância do SGWfC consultar a tabela antes que a anterior pudesse gravar o último identificador utilizado. Nesse caso, teríamos duas instâncias de SGWfC utilizando os mesmos identificadores e existiriam informações conflitantes, e como o SciCumulus é orientado aos dados de proveniência, a execução de um *workflow* acabaria interferindo na outra.

Além disso, índices foram criados para todos os campos que faziam parte de predicados de seleção de alguma consulta executada pelo SciCumulus. Por exemplo, no caso do campo *status*, o SciCumulus verifica quais tarefas já executaram via proveniência e isso requer uma consulta onde *status* = 'FINISHED', logo um índice sobre esse campo se faz necessário no Cassandra. Além disso, antes de criamos as *Column Families* no Cassandra, devemos definir uma *keyspace* a ser usada. No caso do experimento realizado nesse artigo criamos uma *keyspace* específica para as *Columns Families* do SciCumulus conforme apresentado na Tabela 2. Na Tabela 2 podemos perceber que a *keyspace* criada está replicada em um nó utilizando a estratégia *SimpleStrategy* que posiciona a primeira réplica do banco de proveniência em um nó determinado pelo particionador e as réplicas adicionais são colocadas no próximo nó no sentido horário do anel de nós que fazem parte da rede de replicação do Cassandra. Utilizando os bancos de dados relacional e NoSQL executamos então um estudo comparativo entre as abordagens para verificar as vantagens e as desvantagens da utilização de SGBDs NoSQL e seu impacto nas máquinas de execução de *workflows*.

Tabela 2 Criação da *keyspace* no Cassandra

```
CREATE KEYSPACE sciphyComUUID WITH REPLICATION = {'class' :
'SimpleStrategy','replication_factor' : 1 };
USE sciphyComUUID;
```

3. Resultados Experimentais

A fim de comparar o desempenho do SGWfC SciCumulus utilizando o PostgreSQL e o Cassandra, é necessário estabelecer as mesmas condições de execução do SciCumulus para ambos os SGBDs. Para tal, foi configurado o ambiente com a mesma capacidade de processamento tanto para o SciCumulus-PostgreSQL quanto para o SciCumulus-Cassandra. Em ambos os casos utilizamos 16 máquinas virtuais do tipo *small* (EC2 ID: m1.small - 1.7 GB RAM, 1 núcleo de CPU equivalente a um 1.0-1.2 GHz 2007 Opteron, 160 GB de armazenamento) no ambiente da Amazon EC2, um ambiente de nuvem com recursos sob demanda.

Como o SciCumulus é orientado ao banco de proveniência, ele consulta o banco de dados para realizar as várias tarefas como escalonamento e monitoramento. Qualquer mudança de SGBD pode fazer com que a execução paralela do *workflow* também seja impactada negativamente. Para analisar o impacto da mudança do SGBD no SciCumulus, executamos o *workflow* SciPhy (Ocaña *et al.* 2011) utilizando tanto o PostgreSQL quanto o Cassandra. O SciPhy é um *workflow* de bioinformática que executa uma varredura de parâmetros em um conjunto de sequências de DNA, RNA ou

aminoácidos e gera diversas árvore filogenéticas, que apresentam o quanto um organismo se assemelha a outro evolutivamente. Para mais detalhes, favor referenciar Ocaña *et al.* (2011). O SciPhy foi executado com vários conjuntos de dados de entrada de forma que pudéssemos analisar como o SGBD NoSQL se comportaria com o aumento do volume de dados de entrada. A Figura 1 apresenta os tempos de execução do *workflow* para cada conjunto de dados de entrada com apenas uma máquina executando os SGBDs PostgreSQL e o Cassandra (instâncias rotuladas como 1N) e com três máquinas executando os SGBDs Cassandra e o PostgreSQL com o Pgpool (instâncias rotuladas como 3N).

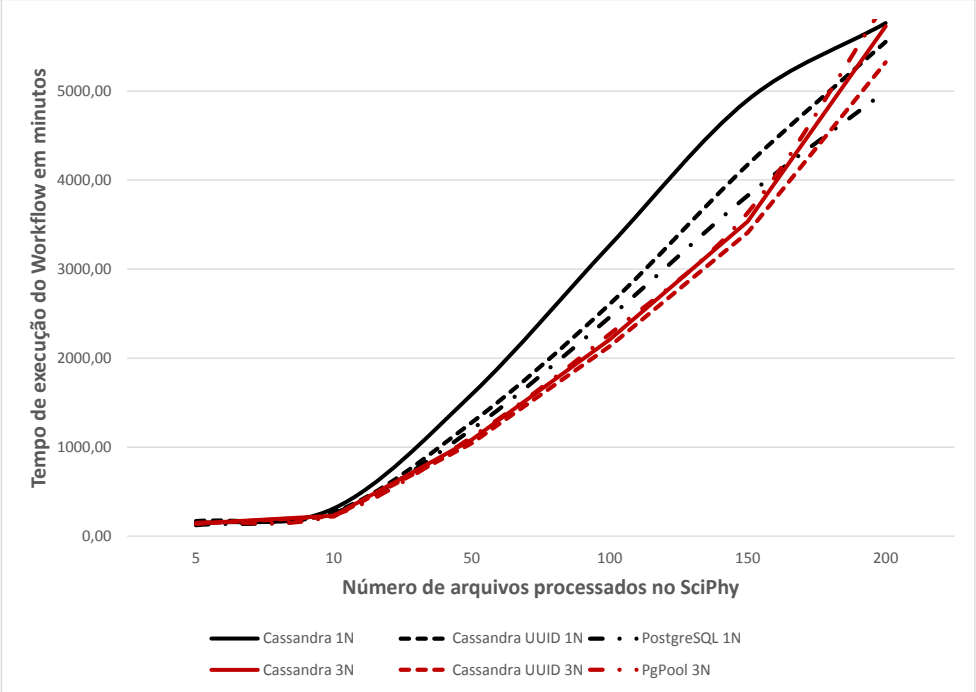


Figura 1 Tempo de execução do *workflow SciPhy*

Nos casos das execuções com os SGBDs em apenas um nó, podemos perceber que sem a distribuição efetiva dos dados o Cassandra insere um *overhead* significativo na execução do *workflow*. Por exemplo, quando processamos 200 arquivos de entrada o tempo de execução do SciCumulus com PostgreSQL dura 4.998 minutos, com o Cassandra sem UUID a duração é de 5.763 minutos e com o Cassandra com UUID a duração é de 5.552 minutos. Apesar de sabermos que a versão sem UUID é menos eficiente já que necessita realizar consultas sucessivas a tabela de metadados, mesmo com a utilização do UUID ainda temos uma degradação de 11,08% no tempo de execução do *workflow*, o que gera uma diferença absoluta de 9,2 horas. Essa diferença ocorre uma vez que o SciCumulus necessita executar diversas consultas aos dados de proveniência durante o curso de execução do *workflow*. Toda vez que uma consulta no PostgreSQL necessita realizar uma junção (*e.g.* saber quais são as atividades de um determinado *workflow*), mais de uma consulta deve ser realizada no Cassandra, o que faz com que a cada atividade a ser despachada, uma consulta seja executada com um *overhead* embutido.

Analisamos também o caso em que o SGBD se encontra distribuído em 3 nós (séries em vermelho na Figura 1). Nesses casos, tanto a execução do Cassandra com UUID quanto a sem o uso do UUID apresentaram um desempenho melhor do que o

PostgreSQL utilizando o Pgpool. Essa vantagem se deu já que cada nó executor do SciCumulus grava seus dados de proveniência o que faz com que tenhamos várias centenas de gravações concorrentes no SGBD. O Cassandra é otimizado justamente para esse fim (acelerar as inserções) o que fez com que seu desempenho tenha sido melhor. O ganho na inserção no Cassandra se deve a associação de um disco em separado para a gravação de *logs*. Quando as *Column Families* compartilham a mesma unidade de disco com o *log*, ocorrem deteriorações nas operações. No caso, a execução com o uso de UUID foi 19% mais eficiente do que com o uso do Pgpool. É importante ressaltar que acima de 175 arquivos de entrada para o SciPhy, o tempo das execuções com o SGBD em 3 nós foi praticamente equivalente a execução com 1 nó para o SGBD. Esse comportamento se deveu a um problema de implementação da máquina de *workflows*. A versão original do SciCumulus elegia apenas uma máquina do seu *cluster* virtual para capturar e armazenar os dados de proveniência (chamada de nó 0). Para os experimentos apresentados nesse artigo, foi necessário modificar a estrutura da máquina para que todos os nós pudessem gravar dados de proveniência. Essa alteração gerou *overheads* que impactaram na execução nos casos com mais instâncias de atividades existentes.

Em todos os casos apresentados anteriormente, executamos apenas uma instância do SciPhy. Para analisar como o Cassandra se comportaria com vários *workflows* executando concorrentemente, iniciamos 4 instâncias do SciPhy consumindo 200 arquivos de entrada cada uma e utilizando o Cassandra com UUID (já que sem o uso do UUID não faria sentido) e o PostgreSQL e Pgpool (Figura 2). O Cassandra apresentou um desempenho superior em todos os casos, mas principalmente quando utilizadas 3 nós para o SGBD. Esse desempenho se dá justamente devido a sua otimização de escrita e também ao controle de acessos concorrentes do Cassandra, que é otimizado. No caso, a execução do Cassandra com 3 nós foi 12% mais eficiente que a execução com o Pgpool o que nos dá uma diferença de aproximadamente 9 horas.

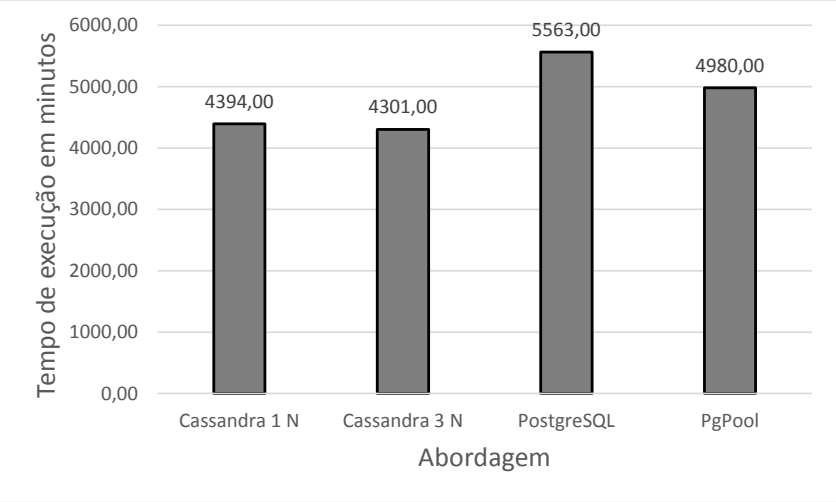


Figura 2 Tempo de execução de 4 *workflows* concorrentes

4. Trabalhos Relacionados

Apesar de existirem diversos trabalhos na literatura que focam na comparação entre SGBDs NoSQL (Abramova e Bernardino 2013, Tudorica e Bucur 2011, Wang e Tang

2012), eles não são específicos para os dados de proveniência. Entretanto, existem alguns trabalhos que comparam o uso de SGBDs relacionais com SGBDs NoSQL especificamente para o armazenamento de dados de proveniência. Cheah (2014) apresenta um estudo comparativo sobre a gerência de dados de proveniência em ambientes de alto desempenho. Similarmente, Vicknair *et al.* (2010) e McColl *et al.* (2014) avaliam o uso de SGBDs NoSQL para o armazenamento e consulta a dados de proveniência distribuída. Entretanto, todos esses trabalhos consideram apenas o uso de SGBDs baseados em grafos como o Neo4J e não baseados em registros distribuídos como o Cassandra e não avaliam o impacto dos SGBDs NoSQL nas máquinas de *workflow*. Desta forma, esse trabalho se diferencia ao explorar uma categoria diferente de SGBDs NoSQL e seu impacto em uma máquina de *workflows* orientada a proveniência.

5. Conclusões

Este artigo apresenta um estudo inicial e preliminar sobre o uso do SGBDs NoSQL para a gerência de dados de proveniência distribuídos e seu consequente impacto nas máquinas de execução de *workflows* científicos. Nesse artigo, escolhemos o Cassandra como SGBD NoSQL e apesar do Cassandra não apresentar todas as vantagens dos tradicionais SGBDs relacionais, o mesmo se mostrou uma solução interessante de acordo com o aumento da quantidade de inserções e acessos concorrentes ao SGBD. Como a captura e a gerência dos dados de proveniência podem ser considerados problemas de *Big Data*, principalmente quando vários *workflows* são executados concorrentemente, o Cassandra pode oferecer um benefício único, já que o mesmo é otimizado para múltiplas escritas e acessos concorrentes a base. Por exemplo, quando o *workflow* SciPhy foi executado com 200 arquivos de entrada o tempo necessário para a execução do *workflow* com o Cassandra em 3 nós foi 12% inferior ao tempo de execução utilizando o Pgpool em 3 nós de mesma capacidade de processamento. Esse resultado é um indicativo do potencial do Cassandra para cenários de múltiplos acessos concorrentes, entretanto, mais experimentos se fazem necessários. Os resultados, apesar de positivos em alguns cenários ainda não podem ser generalizados para qualquer classe de *workflow*. Os resultados obtidos são representativos apenas para uma classe de *workflows*, na qual o SciPhy está incluído. O SciPhy é um *workflow* no estilo MapReduce (Dean e Ghemawat 2010). Novos experimentos devem ser executados, principalmente no que tange as consultas *post mortem* que também são fundamentais para a análise dos resultados do experimento.

Referências Bibliográficas

Abramova, V., Bernardino, J., (2013), "NoSQL Databases: MongoDB vs Cassandra". In: *Proceedings of the International C* Conference on Computer Science and Software Engineering*, p. 14–22, New York, NY, USA.

Bertino, E., Bernstein, P., Agrawal, D., Davidson, S., Dayal, U., Franklin, M., Gehrke, J., Haas, L., Halevy, A., et al., (2012), "Challenges and Opportunities with Big Data"

Cheah, Y.-W., (2014), *Quality, Retrieval and Analysis of Provenance in Large-scale Data*, Indiana University, AAI3613730.

Costa, F., de Oliveira, D., Ocaña, K. A. C. S., Ogasawara, E., Mattoso, M., (2012), "Enabling Re-executions of Parallel Scientific Workflows Using Runtime Provenance Data". In: *Proceedings of the 4th International Conference on Provenance and Annotation of Data and Processes*, p. 229–232, Berlin, Heidelberg.

Costa, F., Silva, V., de Oliveira, D., Ocaña, K., Ogasawara, E., Dias, J., Mattoso, M., (2013), "Capturing and querying workflow runtime provenance with PROV: a practical approach". In: *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, p. 282–289, New York, NY, USA.

Dantas, M., (2005), "Clusters Computacionais", *Computação Distribuída de Alto Desempenho: Redes, Clusters e Grids Computacionais*, 1 ed.Rio de Janeiro: Axcel Books, p. 145–180.

Dean, J., Ghemawat, S., (2010), "MapReduce: a flexible data processing tool", *Communications of the ACM*, v. 53, n. 1, p. 72–77.

Foster, I., Kesselman, C., (2004), *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann.

Freire, J., Koop, D., Santos, E., Silva, C. T., (2008), "Provenance for Computational Tasks: A Survey", *Computing in Science and Engineering, v.10*, n. 3, p. 11–21.

Mattoso, M., Werner, C., Travassos, G. H., Braganholo, V., Ogasawara, E., Oliveira, D., Cruz, S. M., Martinho, W., Murta, L., (2010), "Towards supporting the life cycle of large scale scientific experiments", *International Journal of Business Process Integration and Management*, v. 5, n. 1, p. 79.

McColl, R. C., Ediger, D., Poovey, J., Campbell, D., Bader, D. A., (2014), "A Performance Evaluation of Open Source Graph Databases". In: *Proceedings of the First Workshop on Parallel Programming for Analytics Applications*, p. 11–18, New York, NY, USA.

Moreau, L., Missier, P., (2013). PROV-DM: The PROV Data Model. Disponível em: http://www.w3.org/TR/2013/REC-prov-dm-20130430/. Acesso em: 17 Feb 2014.

Ocaña, K. A. C. S., Oliveira, D., Ogasawara, E., Dávila, A. M. R., Lima, A. A. B., Mattoso, M., (2011), "SciPhy: A Cloud-Based Workflow for Phylogenetic Analysis of Drug Targets in Protozoan Genomes". In: *Advances in Bioinformatics and Computational Biology*, p. 66–70, Berlin, Heidelberg.

Ogasawara, E., Dias, J., Silva, V., Chirigati, F., Oliveira, D., Porto, F., Valduriez, P., Mattoso, M., (2013), "Chiron: A Parallel Engine for Algebraic Scientific Workflows", *Concurrency and Computation*, v. 25, n. 16, p. 2327–2341.

Oliveira, D., Ocaña, K., Baião, F., Mattoso, M., (2012), "A Provenance-based Adaptive Scheduling Heuristic for Parallel Scientific Workflows in Clouds", *Journal of Grid Computing*, v. 10, n. 3, p. 521–552.

Oliveira, D., Ogasawara, E., Baião, F., Mattoso, M., (2010), "SciCumulus: a lightweight cloud middleware to explore many task computing paradigm in scientific workflows". In: *3rd International Conference on Cloud ComputingInternational Conference on Cloud Computing*, p. 378–385, Washington, DC, USA.

Oliveira, W., Oliveira, D., Braganholo, V., (2014), "Experiencing PROV-Wf for Provenance Interoperability in SWfMSs". In: *IPAW*, Cologne, German.

Pintas, J., Oliveira, D., Ocaña, K. A. C. S., Ogasawara, E., Mattoso, M., (2013), "SciLightning: a Cloud Provenance-based Event Notification for Parallel Workflows". In: *Proceedings of the 3rd International Workshop on Cloud Computing and Scientific Applications (CCSA)3rd International Workshop on Cloud Computing and Scientific Applications (CCSA)*, p. 352–365, Berlin, Germany.

Taylor, I. J., Deelman, E., Gannon, D. B., Shields, M., (2007), *Workflows for e-Science: Scientific Workflows for Grids*. 1 ed. Springer.

Tudorica, B. G., Bucur, C., (2011), "A comparison between several NoSQL databases with comments and notes". In: *Roedunet International Conference (RoEduNet), 2011 10thRoedunet International Conference (RoEduNet), 2011 10th*, p. 1–5

Vaquero, L. M., Rodero-Merino, L., Caceres, J., Lindner, M., (2009), "A break in the clouds: towards a cloud definition", *ACM SIGCOMM Computer Communication Review*, v. 39, n. 1, p. 50–55.

Vicknair, C., Macias, M., Zhao, Z., Nan, X., Chen, Y., Wilkins, D., (2010), "A Comparison of a Graph Database and a Relational Database: A Data Provenance Perspective". In: *Proceedings of the 48th Annual Southeast Regional Conference*, p. 42:1–42:6, New York, NY, USA.

Wang, G., Tang, J., (2012), "The NoSQL Principles and Basic Application of Cassandra Model". In: *2012 International Conference on Computer Science Service System (CSSS)2012 International Conference on Computer Science Service System (CSSS)*, p. 1332–1335

29th Brazilian Symposium on Databases

demos

October 6th to 9th, 2014
Curitiba – PR – Brazil

DEMOS AND APPLICATIONS SESSION

PROCEEDINGS

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

Demos and Applications Session Chairs

José Antônio Macedo, UFC
Flávio Sousa, UFC

Editorial

The Brazilian Symposium on Databases (SBBD) is the official database event of the Brazilian Computer Society (SBC). It is the largest venue in Latin America for presentation and discussion of research results in the database domain. In its 29th edition, the symposium will be held in Curitiba, in the state of Paraná, on October 06th to 09th, 2014. SBBD brings together researchers, students and practitioners, from Brazil and abroad, to discuss problems related to the main topics in modern database technologies.

During SBBD, since 2004, a special session is dedicated to the presentation of database software aggregated with research solutions. This session, named Demonstrations and Applications Session (or Demos Session, for short), aims at revealing practical achievements among researchers, developers and professionals, from both academia and industry, interested in relevant and functional tools or applications that use data/information management technologies to solve non-trivial problems. The demonstrations held in the Demos Session concern relevant problems related to data management technologies, providing interaction between researchers and practitioners.

In 2014, we keep the *Minute Madness* presentation format. This presentation is a dynamic event where each author has 1 minute to present his/her research. Authors are challenged to employ creative presentation skills to get the audience attention during this quick and fun moment and to motivate people to visit his/her demonstration/application.

In this edition issue, we had 8 demo papers selected from a total of 24 submissions (an acceptance rate of 33%). Each paper was evaluated by 3 reviewers selected from a committee of 28 researchers from different Brazilian institutions. The Minute Madness got together demos and short papers. Then, during the live demonstration moments, the demos were assessed again by the award selection committee, which determined the Best Demo Paper Award.

We thank all authors of submitted papers for their interest in the event. We also thank the reviewers and the award selection committee for their prompt and careful evaluations. We would like to express our gratitude for their hard work, which was fundamental for this event to happen.

We also congratulate the SBBD 2014 organizers for the local arrangements that provide the necessary infrastructure for Demos exposition and for the Minute Madness, as well as to SBBD Steering for the continuous support given to SBBD Demos.

We hope the SBBD 2014 Demos Session can awake great insights and many interaction opportunities for the Brazilian Database community.

José Antônio Macedo, UFC
SBBD Demos and Applications Session 2014 Chair

Flávio Sousa, UFC
SBBD Demos and Applications Session 2014 Chair

29th Brazilian Symposium on Databases

October 6th to 9th, 2014
Curitiba – PR – Brazil

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

SBBD Steering Commitee

Ana Carolina Salgado, UFPE
Cristina Ciferri, ICMC-USP (chair)
José Palazzo Moreira de Oliveira, UFRGS
Marco A. Casanova, PUC-Rio
Mirella M. Moro, UFMG
Vanessa Braganholo, UFF

SBBD 2014 Commitee

Steering Committee Chair
Cristina Ciferri, ICMC-USP

Local Organization Chairs
Carmem Hara, UFPR and Vidal Martins, PUC-PR

Program Committee Chairs
Mirella M. Moro, UFMG and Renata Galante, UFRGS

Demos and Applications Session Chairs
José Antônio Macedo, UFC and Flávio Sousa, UFC

Workshop on Thesis and Dissertations in Databases Chairs
João Eduardo Ferreira, IME-USP and Fabio Porto, LNCC

Lectures Chair
Bernadette Farias Lóscio, UFPE

Tutorials Chair
Vanessa Braganholo, UFF

Local Organization Committee

Carmem Satie Hara, UFPR (chair)
Cristina Verçosa Pérez Barrios de Souza, PUC-PR
Daniel S. Kaster, UEL (proceedings)
Marcos A. Carrero, UFPR
Nadia Puchalski Kozievitch, UTFPR
Vidal Martins, PUC-PR (chair)

Demos and Applications Session Program Committee

Ana Paula Appel, IBM Research Brazil
Angelo Brayner, UNIFOR
Camila Barioni, UFABC
Carlos Eduardo Pires, UFCG
Damires Souza, IFPB
Daniel Kaster, UEL
Elaine Sousa, ICMC-USP
Evandro Baccarin, UEL
Fábio Porto, LNCC
Fernanda Baiao, UNIRIO
Fernando Trinta, UFC
Flávio Sousa, UFC (chair)
Humberto Razente, UFU
Javam Machado, UFC
Jonice Oliveira, UFRJ
José Antônio Macedo, UFC (chair)
José Maria Monteiro, UFC
Karin Becker, UFRGS
Marcela Ribeiro, UFSCar
Marcio Oikawa, UFABC
Mario Nascimento, University of Alberta
Renata Galante, UFRGS
Renato Bueno, UFSCar
Ronaldo Mello, UFSC
Sandra de Amo, UFU
Valeria Times, UFPE
Vânia Vidal, UFC
Windson Viana, UFC

Table of Contents (Demos Session)

Chameleon: The Performance Tuning Tool for MapReduce Query Processing Systems 203
Edson Ramiro Lucas Filho, Ivan Luiz Picoli, Eduardo Cunha de Almeida, Yves Le Traon

Have You Met VikS? A Novel Framework for Visual Diversity Search Analysis 209
Lúcio Fernandes Dutra Santos, Rafael Loosli Dias, Mônica Ribeiro Porto Ferreira, Marcela Xavier Ribeiro, Agma Juci Machado Traina, Caetano Traina Jr.

Litebase: A Database System for Mobile Devices 215
Juliana Carpes Imperial, Sérgio Lifschitz, Guilherme Campos Hazan, Bruno Muniz

pExtract: Uma abordagem probabilistica para a extração de esquemas de documentos XML 221
Geomar A. Schreiner, Denio Duarte

PhoCA: An extensible service-oriented tool for Photo Clustering Analysis 227
Yuri Almeida Lacerda, Johny Moreira da Silva, Leandro Balby Marinho, Cláudio de Souza Baptista

QualityStamp – Avaliando a Qualidade de Linked Datasets para Aplicações de Domínio Específico 233
Walter Travassos Sarinho, Bernadette Farias Lóscio, Damires Souza

SciCumulus 2.0: Um Sistema de Gerência de Workflows Científicos para Nuvens Orientado a Fluxo de Dados 239
Vítor Silva, Daniel de Oliveira, Marta Mattoso

SHRuB - Searching through Heuristics for the Better Query-Execution Plan .. 245
Marcos Vinicius Naves Bedo, Luiz Olmes Carvalho, Gabriel V. De Pierro, Caetano Traina Jr.

paper:38

Chameleon: The Performance Tuning Tool for MapReduce Query Processing Systems

Edson Ramiro Lucsa Filho[1], **Ivan Luiz Picoli**[2], **Eduardo Cunha de Almeida**[2], **Yves Le Traon**[1]

[1] University of Luxembourg

{edson.lucas,yves.letraon}@uni.lu

[2]Federal University of Paraná

{ilpicoli,eduardo}@inf.ufpr.br

Abstract. *Chameleon is a tuning advisor to support performance tuning decision-making of MapReduce administrators and users. In MapReduce query processing, a query is translated into a set of jobs, i.e., query plan. For administrators, Chameleon can be a powerful tool for observing query plan workloads and their impact in large-cluster machine setups in terms of computing resource consumptions. For users, Chameleon provides a set of functionalities to tune query plans and observe performance improvements while testing different tuning knobs. Chameleon embeds a tuning mechanism based on a hash index to map jobs to tuning knobs. The hash index allows users defining specific tuning knobs for jobs, while a clustering algorithm is responsible for finding jobs with similar resource consumptions to receive the same tuning. In this demonstration we outline the functionalities of Chameleon and allow users interacting with it by sending and tuning queries for auditing performance improvements.*

1. Introduction

In the last years, we have seen the rise of MapReduce Query Processing Systems that translate SQL-like queries into Map Reduce programs, such as Hive [Ashish Thusoo 2009], DryadLINQ [Isard et al. 2007], Pig [Gates et al. 2009], and Shark [Xin et al. 2013]. The advantage of these systems is that they make declarative programs leveraging the MapReduce benefits. For instance in Hive, every incoming query is translated into a set of jobs to be executed in distributed machines, where a job is a program with a set of query operators, the performance tuning knobs and data input. This set of jobs represents the query execution plan in the form of a Directed Acyclic Graph (DAG).

In these systems, performance tuning knobs are either defined in a central configuration file or included into the job source-code (similar to SQL hints). However, in either ways, the same tuning configuration is propagated to all jobs within the DAG, forbidding users to define specific tuning knobs for each job. The propagation of the same tuning can be treacherous, since distinct query operators and varied input data sizes lead to different usages of computing resources (the *TableScan* operator requires disk bandwidth, the *Sort* operator requires memory). Therefore, performance of MapReduce queries depends on specific tuning configuration for each job. Instead of coarse-grained tuning, at the level of a query, we promote a fine-grained tuning, with specific tuning knobs per job. The

http://www.inf.ufpr.br/erlfilho/chameleon.mp4

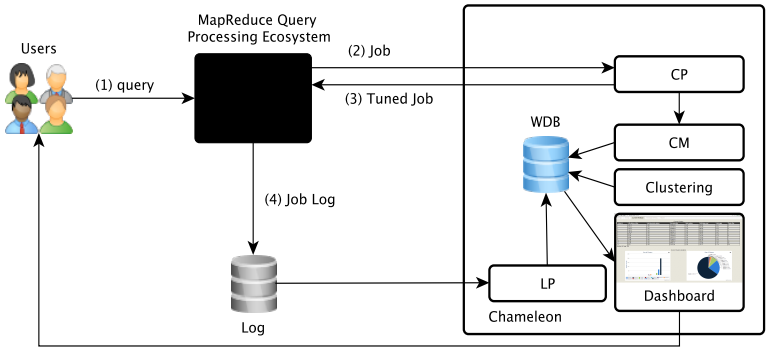


Figure 1. The Chameleon architecture and its integration with the MapReduce Ecosystem.

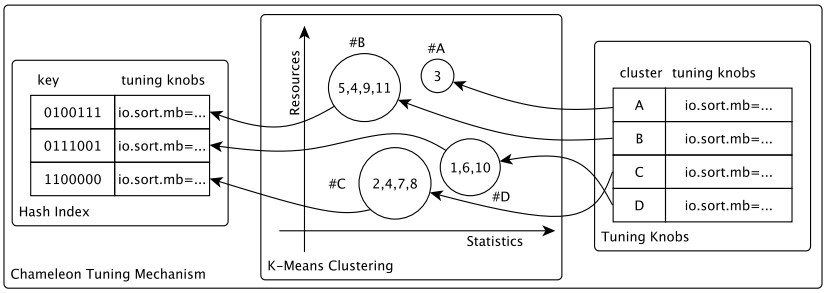


Figure 2. K-means updating the tuning knobs in the Chameleon Hash Index.

issue we address in this paper is the tool-environment for materializing such fine-grained tuning.

Chameleon is a tuning advisor to MapReduce Query Processing Systems. Its main goal is supporting MapReduce users to make performance tuning decisions. Chameleon uses a tuning mechanism based on a hash index. The index key is a canonical representation of a job in the form of a bitmap. The index values represent the tuning knobs. Chameleon implements the K-means unsupervised machine learning algorithm to cluster jobs with equivalent resource consumptions and update the tuning knobs. Clusters are computed from the resource consumption statistics of the execution log files. Once a new job gets its key, it is automatically mapped to a tuning setup. Chameleon does not calculate the actual tuning, rather than, it reuses precomputed tuning (from users or third party MapReduce tuning systems) and optimize queries upfront the execution. Indeed, Chameleon allows ad-hoc jobs receiving proper tuning without monitoring or sampling executions. In this demonstration we outline the functionalities of Chameleon and allow users interacting with it by sending and tuning queries for auditing performance improvements.

2. Architecture

Figure 1 illustrates the Chameleon architecture interacting with the MapReduce Query Processing Ecosystem. The architecture has six modules, described as follows:

- **Code Parser** (CP) is responsible for reading the source-code of each job and creating a canonical representation for it (i.e., the hash index key). Initially, Chameleon



Figure 3. Dashboard: The history list of executed queries.

assumes that the resource consumptions of different jobs can be said equivalent if they implement equivalent query operators. Then, a key maps jobs to specific tuning knobs. After clustering, the keys will be used to update the tuning knobs. Let τ be the query operators and the domain of τ be the list of operators $\{op_1, \ldots, op_m\}$. A key κ is a one-to-one mapping $(M : \tau \rightarrow \{< b_1, \ldots, b_{|\tau|} > | b_i \in \{0, 1\}, i = 1, \ldots, |\tau|\})$. The key is represented as a bitmap, called κ, with a bit turned on or off accordingly to the existence of the operator in the source code. For instance, let us say that a job j_0 implements only the *TableScan* operator, than, $\tau_0 = \{TableScan\}$ and, for instance, $|\tau| = 5$. The calculated key for this job is $\kappa_0 = \{00001\}$, where the bit for the $TableScan$ operator has been switched to 1 in τ_0. The number of query operators supported by the index is represented by τ. This number can be incremented as news programs are parsed. The query systems are evolving and new operators can be mapped without additional definitions, adding a new operator to the key and incrementing $|\tau|$.

- The **Clustering** module performs the K-means algorithm over the execution statistics gathered at the log files. Clustering is the foundation of the Chameleon tuning. K-means creates clusters for jobs with similar usage of computing resources. Once jobs in the same cluster present similar resource consumptions, they are allowed to receive the same tuning setup. Users may interact with K-means by setting 1 to 65 computing resources traced in the log files used. In addition, users can configure the execution of the K-means algorithm to update clustering information. The *refresh_time* configuration refers to the periodicity to perform K-means. The *number_of_classes* configuration refers to the number of clusters created. The *clustering_execs* configuration defines how many times the K-means algorithm is performed within the *refresh_time*.
- **Code Mapper** (CM) maps keys to clusters. For this, Chameleon computes the occurrences of keys per cluster. The algorithm is straightforward, the key maps to the cluster in which it appears the most. Figure 2 illustrates the keys matching the tuning knobs in the hash index. For instance, lets say jobs 1, 3 and 6 share the same key $\{0111001\}$, but in different clusters (#A and #D). Their key will map to cluster #D with two occurrences. This means, key $\{0111001\}$ will receive the tuning knobs from cluster #D. The goal is allowing upcoming jobs reusing precomputed tuning setup. Once a new job gets its key, it is mapped to a cluster and automatically receives the tuning setup from such cluster. In this way, Chameleon can reuse precomputed tuning (from users or third party MapReduce tuning systems) and optimize queries upfront the execution.
- **Log Parser** (LP) periodically reads the query log files traced by the MapReduce Query Processing Ecosystem in order to extract execution statistics; for instance,

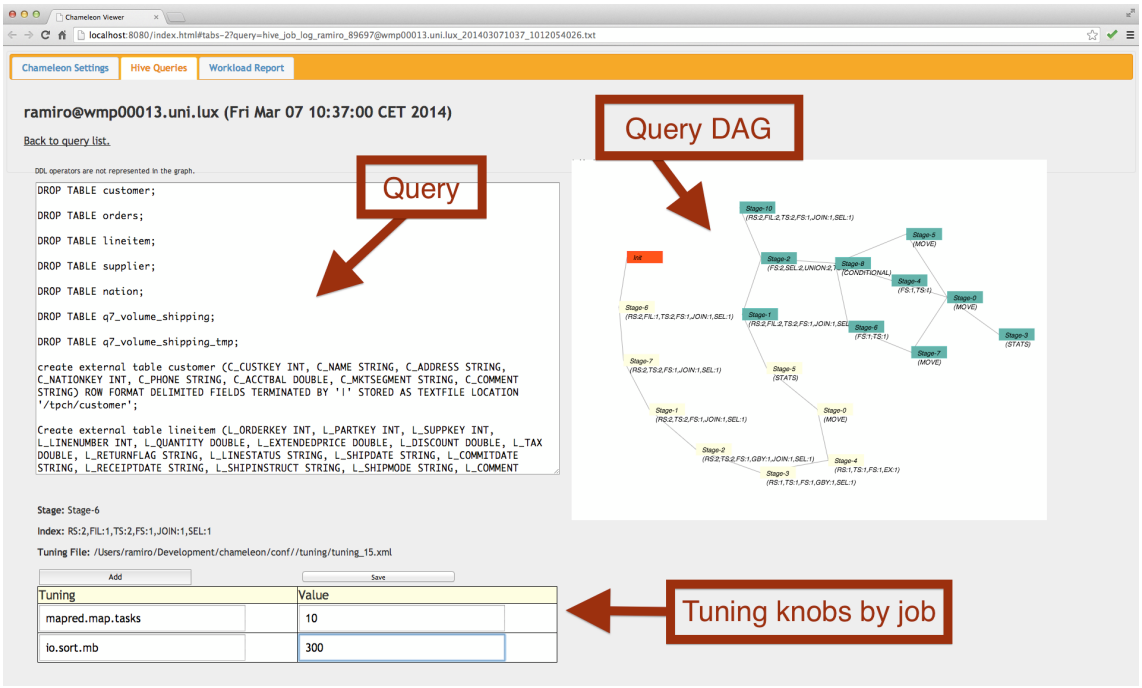


Figure 4. Dashboard: Query analysis interface.

Tuning Knobs	Range of Values
io.sort.mb	70-250
io.sort.factor	7-10
io.sort.record.percent	0.05-0.5
io.sort.spill.percent	0.80-0.99
io.file.buffer.size	4096-131072
mapred.child.java.opts	Xmx512m - Xmx1g Xms256m - Xms512m
mapred.job.shuffle.merge.percent	0.66-0.99
mapred.reduce.parallel.copies	5-10

Table 1. List of the Hadoop tuning knobs.

the number of bytes that was processed at the map and reduce phases.

- **Workload Database** (WDB) is the internal information database that hosts the hash index and tuning knobs computed by third party as depicted by Table 1. It also hosts the information required to perform K-means, such as: the *statistics* from the log files and the clusters.
- The **Dashboard** module provides the Chameleon web interface with three tabs. The *Chameleon Settings* tab is reserved to setup the internal configurations, such as: database connection, path to the MapReduce Query Processing Ecosystem, and K-means setup. The *Queries* tab, illustrated in Figure 3, summarizes the execution history. After selecting one of the queries for tuning, as we illustrate in Figure 4, the *Dashboard* module presents the query plan with its internal jobs and source-codes, and the tuning knobs applied. As illustrated in Figure 5, the *Workload Report* summarizes the workload characteristics of each cluster. Once

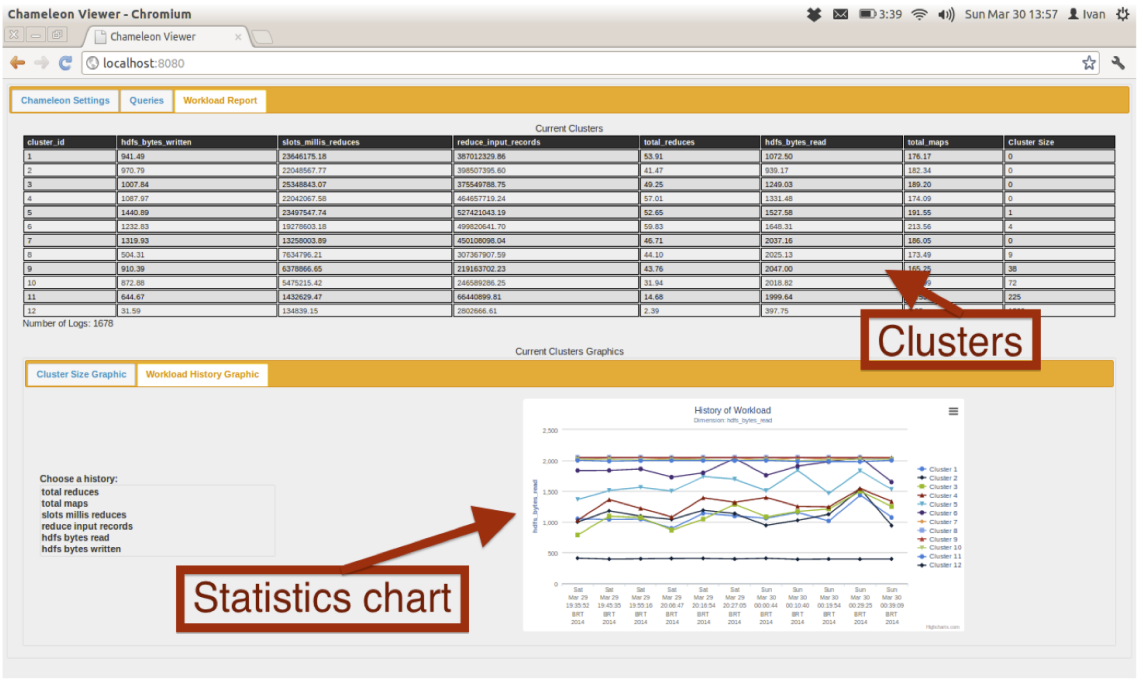


Figure 5. Dashboard: The workload analysis interface.

the users know the workload characteristics, they are able to define specific tuning parameters for each cluster.

3. Demonstration overview

We will present the Chameleon functionalities, allowing users to interact with it by sending queries via HiveQL command line interface and setting different tuning setups to observe performance improvements. Users will also interact with Chameleon via web interface to analyze the execution workloads. The current implementation of Chameleon supports the Apache Hive 0.11.0 and Apache Hadoop 1.2.1. The demonstration agenda is organized, as follows:

1. **Start Chameleon Internal Modules** We will launch the Chameleon modules that will be up and running in background during the demonstration. We will present a brief description of the Chameleon configuration files, including the K-means setup.
2. **Query submission**: The Apache Hive will have pre-loaded data of the TPC-H database. We will provide the 22 TPC-H queries written in HiveQL, but users will be able to submit any query via Hive Command Line Interface to see the functionalities of Chameleon. Users will observe Chameleon tuning each job generated by queries followed by the proper execution.
3. **Dashboard**: Users will be able to access the Dashboard and interact with the *Queries* tab to see the execution history of jobs. They will be able to interact with any submitted query by checking the query source-code, the query DAG with its internal jobs and the tuning knobs applied to each job. At this point, users will be able to interact with the *Workload Report* tab to analyze the charts and tables that describe the clusters and their workload.

Figure 5 depicts the clusters and their execution statistics. Users will visualize a combination of statistics used by the K-means algorithm and their respective values. The header plots the *statistics* chosen by the user in the Chameleon settings, where each column represents one of the *statistics*. The column values are the average *statistic* values computed by K-means. The last column represents how many jobs have been clustered. Users will visualize the *Statistics* chart, which drills down to the execution statistics of each cluster. This chart is a tool to check the ongoing workload in the MapReduce Query Processing Ecosystem. For instance, the number of bytes read and written by the HDFS.

4. **Tuning Query**: Users will be invited to redefine the tuning parameters for each cluster or import them from third party tuning systems. Then, users will be able to re-execute the same query via Hive Command Line Interface in order to observe performance improvements.

4. The Take-Away Message

Chameleon is a tuning advisor to support performance tuning decision-making of MapReduce users and administrators. For users, Chameleon provides a set of functionalities to tune query plans and observe performance improvements while testing different tuning knobs. Chameleon does not calculate the actual tuning, rather than, it reuses precomputed tuning (from users or third party MapReduce tuning systems) and performs intra-query optimization upfront the execution. It also enables jobs from different queries sharing the same tuning. For administrators, Chameleon can be a powerful tool for observing query workloads and their impact in large-cluster machine setups in terms of computing resource consumptions.

References

Ashish Thusoo, J. S. S. (2009). Hive- A Warehousing Solution Over a Map-Reduce Framework. *Proceedings of the VLDB Endowment*.

Gates, A. F., Natkovich, O., Chopra, S., Kamath, P., Narayanamurthy, S. M., Olston, C., Reed, B., Srinivasan, S., and Srivastava, U. (2009). Building a high-level dataflow system on top of Map-Reduce: the Pig experience. *Proceedings of the VLDB Endowment*, 2(2):1414–1425.

Isard, M., Budiu, M., Yu, Y., Birrell, A., and Fetterly, D. (2007). Dryad: distributed data-parallel programs from sequential building blocks. *ACM SIGOPS Operating Systems Review*, 41(3):59.

Xin, R. S., Rosen, J., Zaharia, M., Franklin, M. J., Shenker, S., and Stoica, I. (2013). Shark: SQL and rich analytics at scale. In *Proceedings of the 2013 international conference on Management of data - SIGMOD '13*, page 13, New York, New York, USA. ACM Press.

paper:4

Have You Met VikS? A Novel Framework for Visual Diversity Search Analysis*

Lucio F. D. Santos[1], Rafael L. Dias[2], Mônica R. P. Ferreira[1], Marcela X. Ribeiro[2], Agma J. M. Traina[1], Caetano Traina Jr.[1]

[1]Database and Images Group (GBDI)
Institute of Mathematics and Computer Science (ICMC)
University of São Paulo (USP)
São Carlos – SP – Brazil

{luciodb,monika,agma,caetano}@icmc.usp.br

[2]Department of Computing
Federal University of São Carlos (UFSCar)
São Carlos – SP – Brazil

{rafael.dias,marcela}@dc.ufscar.br

Abstract. *Searching images based on their pictorial content, or content-based image retrieval (CBIR), instead of using traditional tags and labeling attached to them has attracted considerable attention. However, this retrieval by content may often retrieve images too similar among themselves. Considering a diversity factor has been a way of improving the quality of results retrieved by user queries. There are still questions about how this factor is used in the searches. In this paper, we present VikS[1], a CBIR system that answers queries based on the similarity and diversity paradigms and supports visual data mining techniques, becoming the user an active agent in the query process and enhancing the understanding of the impact of the diversity in the k-nearest neighbor queries. This framework provides implementations of a wide suite of algorithms to compute and compare diverse results. Users can tune diversification parameters, combine similarity with diversity and see how diverse results are in a projecting space which highlight the distance distribution of the elements.*

1. Introduction

Most Content-Based Image Retrieval (CBIR) systems query image sets employing the similarity paradigm [Santos et al. 2013]. They retrieve images that are similar to the given query image based on images' pictorial content instead of using traditional data (text or numerical attributes annotated to the image). Typically, each image is represented by a feature vector, which is a numeric representation summarizing the image visual characteristics, such as color, shape or texture. To establish a similarity degree among images, the feature vectors are compared by a distance function. Based on a similarity criteria (either range or k-nearest neighbor), the system retrieves images that are closer/similar to the given query image. A range query (Rq) retrieves all images closer to or at the same

*This research was supported by FAPESP, CNPq, Capes, RESCUER, and SticAMSUD.
[1]https://db.tt/Px8yzUng

distance from the query image, while the k-nearest neighbor ($kNNq$) retrieves the k ones most similar to the query image.

However, in some scenarios, querying a dataset employing $kNNq$ and/or Rq criteria may often return images that present too much similarity among themselves, specially if the data has clusters. The problem of having elements too similar in result sets induces the user to think that the dataset does not have his/her answer, often requiring query reformulations, which can lead the user to give up the search. A solution is to provide a diversity degree between images in the results, aiming at approximating them to the user's expectation. The intuition behind diversity is generating a result set that includes elements not only similar to the query element, but also diverse among themselves.

Diversity has been explored in various ways in different areas [Drosou and Pitoura 2013]. Most approaches use metadata related to the elements besides the distances among them, which restrict their usage to datasets that have this information in advance [Vieira et al. 2011]. Others have pursued diversity without using extra information, called the distance-based approach. Here, similarity and diversity measurements must compete with each other, exploiting a user-defined diversity preference parameter. The result of traditional similarity algorithms is re-ranked, inducing diversity among elements based on a bi-criteria objective function, which results in an NP-hard problem [Santos et al. 2013]. To overcome their high computational cost, randomized or greedy algorithms are used to provide approximate results in feasible time.

In the literature, there are some prototypes aimed at optimizing and evaluating the distance-based approaches, such as DivDB [Vieira et al. 2011] and POIKILO [Drosou and Pitoura 2013]. In DivDB, the user can manipulate the *trade-off* between diversity and similarity and other query parameters. However, even though the user can participate more actively in the query process, the comprehension of how the diversity factor is being applied in the whole dataset is not explored [Dias et al. 2013]. In a similar way, POIKILO provides tools to tune several query parameters and diversity models. Geographical dataset produces some visual information about the diversification process, but there is no feedback regarding the relationship among elements in the image dataset. Despite their efforts, these systems do not provide information to help the user to understand the diversified answer.

Using visualization techniques together with user feedback is a way to bring the user as an active agent in the diversity query process. Visual Data Mining (VDM) techniques [Keim 2002] allow creating a process for knowledge extraction that combines computational efficiency with human judgment. Different visualization and interaction strategies were developed, turning easier the analysis and exploration of image datasets and related information by the user. An example of a VDM technique is the multidimensional projection LAMP (Local Affine Multidimensional Projection) [Joia et al. 2011] that employs the local mapping, using an initial sample subset to guide the position of other elements in the final projection. Thus, the user can manipulate the elements and bend the projection according to his/her knowledge about the correlation between the control points previously projected, driving interactively to the final projection.

In this paper we present the Visual Interactive k Diverse Neighbor System (VikS), a CBIR system that answers queries based on the similarity and diversity paradigms and

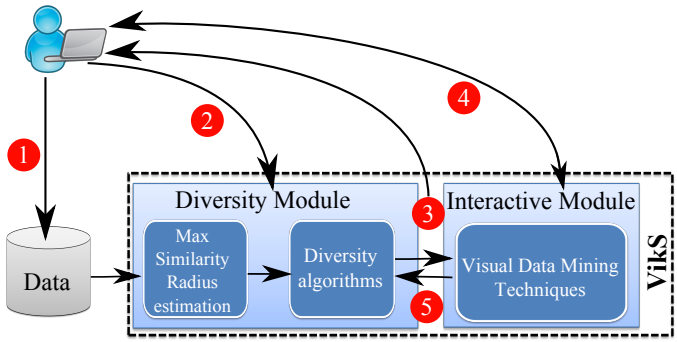


Figure 1. The VikS architecture.

supports VDM techniques to visualize and interact with query results, semantically improving them.

2. The VikS Framework

The *Visual Interactive k Diverse Neighbor System* – VikS – is a framework that helps users to understand and interact with the answers provided by similarity with diversity search algorithms. The framework takes advantage of VDM techniques to provide tools for the user to explore interactively the dataset, engaging more actively in the process and steering the results according to his/her needs. Therefore, the user becomes an active agent in the analysis process, providing feedback about the relationship of the elements in the search space.

Figure 1 presents an overview of VikS architecture. Starting VikS, the user should provide the dataset to be analyzed and the query image (Step 1). Due to the high computational cost to answer diversity search, the user must define the 'Max Similarity Radius Estimation' parameter in the 'Diversity Module' (Step 2) to restrict the search space for the most similar elements to the query image, since too much dissimilar elements may induce users to think that the query was misinterpreted. Therefore, the diverse candidate elements should not be farther than a distance threshold $MaxSim$ to the query image. However, the definition of this threshold for the user is not easy, because it can vary according to the distance distribution of the dataset. To surpass such problem, VikS takes advantage of an optimization technique that applies the dataset fractal dimension analysis to estimate the $MaxSim$ just using the number of elements expected in the candidate set, being more friendly for the user. Another parameter that should be provided in the 'Diversity Module' is the 'Diversity Algorithms'. In this parameter, the user chooses which algorithm will be used for search. Viks has the following literature algorithms available: Greedy with Marginal Contribution (GMC) [Vieira et al. 2011], Greedy Randomized with Neighborhood Expansion (GNE) [Vieira et al. 2011], Maximal Marginal Relevance (MMR) [Carbonell and Goldstein 1998], SWAP [Vieira et al. 2011], Max-Sum Dispersion (MSD) [Vieira et al. 2011], MOTLEY [Vieira et al. 2011] and a parallel version of the brute-force algorithm (OPD) [Dias et al. 2013]. Those algorithms differ in the result set strategy construction which impacts in the running time and the result quality.

The plain result (only the images) is presented to the user using the n-dimensional features space (Step 3), that is, the search was employed using the original space that

has the low-level features (basic characteristics such as color and shape) extracted by an automatic image processing algorithm, which may not entirely represent the semantic expected by the user cognition about the images. At this point, the user interacts with VikS executing queries with different adjustment and algorithms, as already happens in the other literature systems.

In addition , the user can activate the 'Interactive Module' (Step 4) and manipulate the search space, since VikS uses the LAMP VDM technique to map n-dimensional data in a 2-dimensional space, preserving most of the relationships between distances of elements from the original space. Thus, projections of the query image and initial control points are presented to the user, which allow interactive edition, such as shifting elements in the space and changing the position of the query image, following prior knowledge of the user about the dataset. To assist the activity of bending and edition of the initial projection space, the user can request to visualize the image identifier or the preview (thumbnail) of a projected instance. Besides that, each class of images (this information is retrieved from metadata) is identified by different colors, enhancing the user understanding about the relationship among the elements and their classes.

The edited projection space (Step 5) is sent back to the 'Diversity Module', which is used to generate a new diverse result set using the information provided by the user in the Step 4. As a result, VikS returns the plain images and the projection space with information about the elements selected in each space, n-dimensional (extracted features) and the edited one. Aimed at highlighting the advantages of using the user's knowledge in the query process, VikS provides information about which images were considered as candidate subsets to the diversity algorithm, i.e, depicts the 'Max Similarity Radius' estimated to prune the search space, as well as the proximity (similarity) in which they are positioned in projected space.

To the best of our knowledge, VikS is the first framework allowing users to compare different diversifying algorithms and participate in the query process bending and steering the results through visual data mining techniques, providing a better understanding of the impact of the diversity factor in the $kNNq$ and how those algorithms differ from each other.

3. Demonstration of the ViKs Framework

Aimed at providing portability, the ViKs framework was coded in Java EE. Its main interface is presented in Figure 2. Users are allowed to submit queries, see results diversified, tune the diversification parameters and interact with the search space steering the result to its needs. To allow a download-and-run experience, two real images datasets are preloaded in the system: MRIBalan [Balan et al. 2005] and Faces [Vieira et al. 2011]. The former is composed of 704 images of magnetic resonance imaging obtained from the Clinical Hospital at Ribeirão Preto of University of São Paulo (USP). The image feature vectors were obtained by the method proposed in [Balan et al. 2005]. The latter is a collection of 300 gray images of 30 people's faces with slight variations of the head positions. The image feature vectors were extracted using gray-level histogram. For both datasets, the feature vectors are compared using the Euclidean distance function (L_2) evaluated over the full feature vectors extracted.

Upon starting the framework, a panel is presented, where the user should config-

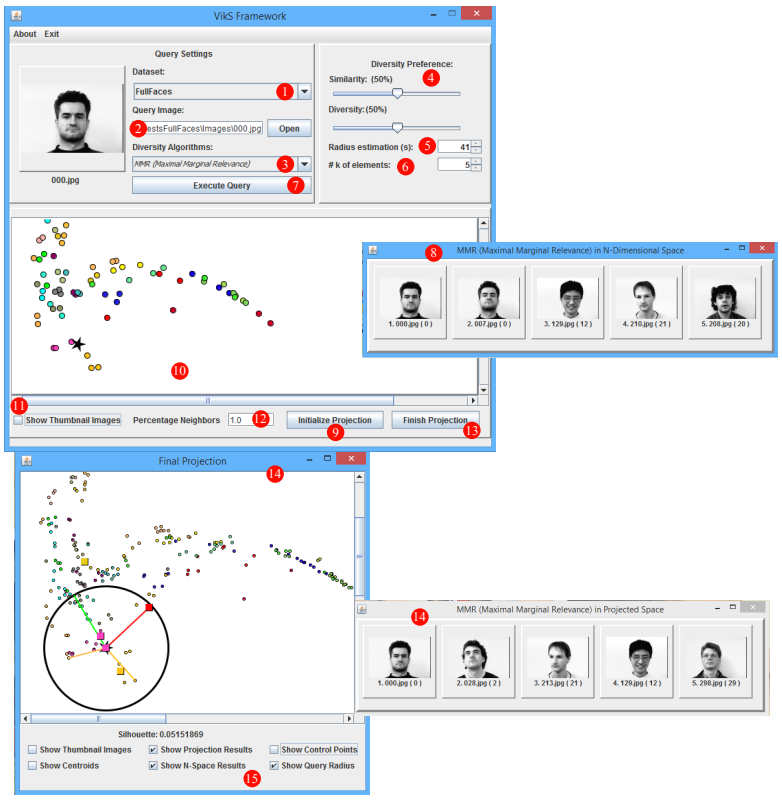


Figure 2. The VikS framework interface.

ure the query settings, selecting the dataset (Step 1), the query image (Step 2), and the diversity algorithm to be executed (Step 3). Then, the user defines which is his/her preference for the results, starting from pure similarity to full diversity answers (Step 4). The number $MaxSim$ (s) of elements to be estimated by the max similarity radius parameter (Step 5) and the number of k elements to be returned (Step 6) should also be specified. The diversity algorithm, the diversity preference and the $MaxSim$ are the diversification parameters of the 'Diversity Module' (Figure 1). At this point, the user can submit the query to the system (Step 7) using only the n-dimensional space without providing any more feedback. The diversified result is showed in the result window (Step 8).

Another option is to activate the 'Interactive Module' clicking on the 'Initialize Projection' button (Step 9), which provides feedback about the similarity of the elements in the search space and the position of the query image (Step 10). The colors indicate the class of each image, and the user can see the preview (thumbnail images) marking the check box (Step 11) as well as defining the number of neighbors to control the projection (Step 12), which has 1.0 as the default value.

The user provides feedback about the search space handling/bending the information presented in the initial projection (Step 10). It is also possible to redo the query using the new distribution of the images (Step 13). Finally, VikS returns the result in a projected space (Step 14). At this point, the user have the option to tune the visualization by zooming-in or out of the presented projection. It is also possible to highlight the factors involved in this process (Step 15) showing the results in the n-dimensional space, the projection results, the max similarity radius, the clusters centroids, the thumbnail images

and the element color indicating the different classes of the dataset (this information must be defined in advance).

4. Conclusions

Querying by image content shows up as a complement to text-based query. In certain situations, there is a greater interest in analyzing the image by its visual features than just using the textual information related to them. However, the manipulation of information concerning images' visual content can create some drawbacks to the user analysis when the returned images for a query are too similar to the query image and among themselves. One way to reduce the problem is by adding a diversity factor to the query, using besides the similarity between the query image and other images in the dataset, some dissimilarity in the resulting subset. However, specifying dissimilarity is not an easy task, since the impact of diversification process in the dataset is almost always unclear to the user.

In this paper, we presented the VikS framework, which uses the user's knowledge to assist in the query diversification process. Additionally, visualization and interaction techniques were used in order to engage the user in a more active role in the query process, enabling a better understanding of the results and relationships between images and categories. To the best of our knowledge, VikS is the first framework to integrate similarity with diversity search to visual data mining technique, enhancing the understanding of the impact of the diversity in the k-nearest neighbor queries.

References

Balan, A., Traina, A. J. M., Traina, A., and Azevedo-Marques, P. (2005). Fractal analysis of image textures for indexing and retrieval by content. In *Proc. of CBMS*, pages 581–586.

Carbonell, J. G. and Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. ACM SIGIR*, pages 335–336, Melbourne, Australia. ACM.

Dias, R. L., Bueno, R., and Ribeiro, M. X. (2013). Reducing the complexity of k-nearest diverse neighbor queries in medical image datasets through fractal analysis. In *Proc. CBMS.*, pages 101–106.

Drosou, M. and Pitoura, E. (2013). Poikilo: A tool for evaluating the results of diversification models and algorithms. *Proc. VLDB Endow.*, 6(12):1246–1249.

Joia, P., Coimbra, D., Cuminato, J. A., Paulovich, F. V., and Nonato, L. G. (2011). Local affine multidimensional projection. *IEEE Trans. Visual. Comput. Graphics*, 17(12):2563–2571.

Keim, D. A. (2002). Information visualization and visual data mining. *IEEE Trans. Visual. Comput. Graphics*, 8(1):1–8.

Santos, L. F. D., Oliveira, W. D., Ferreira, M. R. P., Cordeiro, R. L. F., Traina, A. J. M., and Jr., C. T. (2013). Evaluating the diversification of similarity query results. *Journal of Information and Data Management*, 4(3):188–203.

Vieira, M. R., Razente, H. L., Barioni, M. C. N., Hadjieleftheriou, M., Srivastava, D., Jr., C. T., and Tsotras, V. J. (2011). Divdb: A system for diversifying query results. *PVLDB*, 4(12):1395–1398.

paper:8

Litebase: A Database System for Mobile Devices

Juliana C. Imperial[1], Sérgio Lifschitz[2], Guilherme C. Hazan[1], Bruno Muniz[1]

[1]TotalCross Global Mobile Platform – Fortaleza – CE – Brazil

[2]Departamento de Informática – PUC-Rio, Rio de Janeiro, RJ – Brazil

{juliana.imperial,guich,bruno}@totalcross.com, Sergio@inf.puc-rio.br

Abstract. *Litebase is a database system developed to run on TotalCross, a mobile development platform for smartphones. This paper describes the database system features, currently work, and future enhancements so that it is reliable, fast in all operations, has an efficient memory management, and an efficient storage space usage. The first requirement is challenging because applications might be killed by the device system anytime, and the last three requirements are necessary since even the modern and powerful nowadays devices are not as powerful as computers.*

1. Introduction

Nowadays, the number of mobile devices, such as smartphones, is very huge and may overcome the number of computers (desktops, notebooks, and netbooks) in this year. Therefore, it is interesting to invest in program development for these devices, such as Androids, iPhones, iPads, and Windows Phones [Morgan Stanley 2010].

In order to facilitate the programming for mobile devices with different hardware and operational systems, TotalCross (totally-cross platform) [TotalCross 2014] was created. It is a program development platform for mobile devices which focus on portability, so that the same program can be executed in different platforms without changing its code, and without needing third-party software or hardware to create an executable to the target platform.

TotalCross is currently used on thirty four thousand devices and is present in 17 countries around the world, being used by huge enterprises. Its current version runs on Java, Windows XP/Vista/7/8, Androids, iOSs, Linux, and Windows Phone 8. Previous TotalCross versions used to run on Palm, BlackBerry, Windows CE/Mobile, and previous desktop Windows versions, which are not supported anymore because these platforms were discontinued.

A TotalCross programmer only needs to know to program in Java, since TotalCross uses one subset of this widely used language and some additional libraries. When running a TotalCross program on Java, it uses the Java virtual machine (VM), whereas on the other devices, the TotalCross VM is used.

Since its first version, TotalCross comes with a portable relational database system called Litebase [Litebase 2014][1].

[1] Link for a demonstration video: http://www.julianaimperial.com.br/demo.rar.

Litebase implements an almost standard SQL subset. It was written in such a way that a database created on one device can be used on another one without any conversion. Although it is embedded in TotalCross, it is also possible to use it in applications without user interface, but the use of TotalCross is necessary. However, on Java, a programmer can use it without using TotalCross, using it as a server.

Litebase was designed to be fast and to have an efficient memory management and storage space usage. In the past these requirements were even more necessary as the old devices where it used to run, such as Palms and BlackBerries, had little RAM memory, small storage space, and very slow processors. Nowadays, these requirements are still necessary as the modern devices are not as fast as computers.

In spite of being used by many customers, Litebase still lacks much functionality that widely used database systems implement, such as transactions (which is being implemented) and concurrency. All of them are in the roadmap to be implemented in the future to improve the system.

The rest of this paper is organized as follow: section two describes Litebase features, section three describes Litebase limits and limitations, section four shows some implementation details, section five compares Litebase with SQLite, a widely used database system on mobile devices, and the last section concludes this work and shows its development roadmap.

2. Litebase Features

Litebase has the following features:

- SQL used keywords are treated as reserved words.
- Can issue error messages in English or Portuguese, and it is easy to add another language to it. However, internal error messages from the underlining operational system are issued in the language used by the device.
- Supports case-insensitive strings using the keyword `nocase`.
- Strings can be stored in unicode (default) or ascii to save disk space.
- Supports cryptography, shuffling table files data so that they become unreadable.
- Supports `like`, some SQL data and aggregate functions, `order` and `group by`.
- Supports simple and composed primary keys and indices. There are no foreign keys.
- Supports null and default values.
- Supports inner joins, there are no outer joins yet.
- Supports adding columns, and column and table rename using `alter table`.
- Supports multiple connections and multiple threads (with some limitations because the system doesn't support concurrency yet) with the same or different databases.
- Has error and operation logging.
- Has a specific configuration for huge-data loading in a table in order to speed up the total time taken for all operations in the database. When transaction implementation

is finished, this will not be necessary anymore because the flushes are also only done when the memory cache is full.

- Eases the synchronization task from the device to the server, a very important feature to mobile applications. Therefore, this is one of the most important Litebase features. The database system provides a way to mark a row as synchronized, new, updated, or deleted, without additional columns in a table. It also provides an easy way to search this information and send it to another database system on a server.
- The deletion of records and index keys are always logical. When the programmer issues a purge operation in a Litebase table, all the wasted space (due to deletions and updates), are recovered, and the indices, rebuilt.
- Each table has at least two files: one for row (fixed-length records) data, and another for the strings and blobs stored in the table. Moreover, each table index, including primary keys, is stored in a different file.
- The string types in Litebase are `char` and `varchar` and they are treated in the same way by the database system.
- Index files have a cache of 512 bytes (maximum node size), whereas the table files have a cache of at least 2048 bytes. This is greater if the system must read more than this size at once, for example, when a blob, string, or table record to be read is greater than that. Those values were found empirically.
- When the index is created or re-created, all internal nodes are full so that the queries will be faster. This is done by sorting the keys before inserting them.

3. Litebase Limits and Limitations

Litebase has the following limits and limitations:

- The first data of a row is always an internal 4-bytes field called rowid, which can only be used for searches, but can never be updated. rowid has a unique value for each line and is never reused, even if a row is deleted. The first two more significant bits of the four bytes store information for synchronization, indicating if a row is new, updated, deleted (a removed row is never marked as synchronized), or synchronized. The other bits represent a natural number with 30 bits, which can have more than 500 million different values (beginning with 1). This value is huge even for a database in modern mobile devices and does not to be increased soon.
- Each table can have at most 255 columns including rowid, a limit never reached by well-designed databases.
- The maximum number of parameters in a prepared statement is 254.
- A where clause can use at most 32 indices.
- The maximum number of simple indices is given by the number of table columns, while the maximum number of composed indices is also 32.
- The maximum number of columns used in a where or having clause is also 254, where this number takes into consideration repeated values. That is, if a where clause uses only the column `A` repeated 255, an exception will be raised. This

limitation would not be important if Litebase supported the IN operator. Without it, the programmer must use an OR with all the possible values of the column.

- Since Litebase transactions are still being implemented, during normal usage, all table modifications are flushed immediately, since the database system does not have transactions yet. This guarantees that no data will be lost. However, when the programmer uses the mode to load huge amounts of data in a table, the flush is only done when necessary, when the file caches are full. Therefore, if a TotalCross program using Litebase in this way is stopped suddenly (due to a crash or the system closes it), to be used again the table must pass through a recovery process, where a corrupted row is marked as deleted and has its rowid set to 0, and all indices are rebuilt.
- The currently maximum blob size is 10 Mb. Even this small size might be too huge and cause out of memory problems. In the future it is intended to increase this size and enable to use streams to store and query blobs, so that it is not necessary to have all its content in memory. Streams will also be used to improve some internal operations in order to improve memory management.
- The maximum size of the file that stores strings and blobs is 2 Gb, because the value stored as the string or blob position in the record has four bytes (`int`). To increase the file size, this value must be changed to be stored in 8 bytes (`long`).
- There can only be 65535 nodes in a Litebase index. If the index is created on a `short` (two bytes) column, it will be able to hold about two million keys if they are added after the index creation, and about four million if the index is created or recreated. This seems to be a very large number for a mobile device. However, if the key is of a larger type, or the index is composed, the number of keys it can hold will be much smaller and this might turn a specific index usage impossible if the table is big enough. This limit will be increased soon.
- There are no triggers, views, sub-queries, and constrains.

4. Implementation Details

Litebase has two different codes which do the same work: one implemented in Java to run on the Java VM and another one which has only a very small part in Java and all the rest in C to run on the TotalCross VM on the supported devices. The parts in Java are only to load the Litebase native library and to deal with some synchronization problems when trying to load it. This way, almost all Litebase execution is native, without using the TotalCross VM, improving its performance.

Although Litebase uses native code, it was written in order to make it as platform independent as possible, using TotalCross internal native functions to deal with platform dependent details, such as file treatment. Therefore, it is very easy to port Litebase to new mobile platforms that support code written in C.

The only parts of Litebase code that are platform dependent are the ones that deal with paths (on Windows Phone 8, for instance, path strings use unicode characters while the others use ascii), and when it has to deal with a POSIX limitation concerning the maximum number of files that can be opened at the same time.

The problem with the approach of having many files in a database is that iOSs, Androids, and Linux, have a maximum number of files that can be opened at the same time by all currently running programs. On iOSs, this number is very small, about 256, whereas on Androids, it's about 1024. To solve this problem, Litebase will only open 128, 512, and 1024 files on iOSs, Androids, and Linux, respectively, so that Litebase won't exhaust the system resources. If the programmer opens too many tables so that it's necessary to use too many files at the same time, Litebase will close the last used files when it needs to open another one and re-opens it again if needed without any extra effort from the programmer.

5. Comparison with SQLite

SQLite [SQLite] is a widely used database system and adopted as a standard on Androids and iOSs. The current TotalCross version also has an embedded SQLite so that the programmers have another option to store data. SQLite has many sponsors, has many more features and supports much more of the SQL standard than Litebase. However, Litebase has some advantages:

- Litebase can issue error messages in English or Portuguese and it is easy to add another language to it whereas SQLite only supports English.
- Litebase cryptography is free although it is not strong.
- Using Litebase it is possible to log all used SQL commands and some other operations such as purge and use the log files to reproduce a specific Litebase usage in order to find bugs in the database system or in the TotalCross program.
- Strings can be stored in ascii to save the storage space.
- Litebase is strong-typed, which is less error-prone than the weak-typed SQLite, where any type of data can be inserted in any another type of column (for instance, inserting a blob in an integer field).
- In SQLite the synchronization task must be programmed using an extra field, for instance, whereas Litebase provides it. Moreover, deletions will have to be delayed because the deleted data space might be reused at any time.
- SQLite does not provide a way to drop all indices in a single command, whereas Litebase supports `drop index * on table_name`.
- There is no way to add or drop a primary key on an already created table in SQLite, whereas Litebase supports these commands via `alter table`.
- Litebase result sets hold the number of rows that result from a query. Moreover, it is possible to navigate to any row of the result set. This is useful when loading data to a grid from a result set dynamically so that only the results shown on screen are loaded. When going backward on the grid, it is only necessary to navigate the result set backwards. It is also possible to jump to absolute and relative positions in it. SQLite only allows going forward. To go backward, it must be reset.
- Litebase does not support concurrency yet, but since all table are stored in different files, when writing data it will not be necessary to issue an exclusive lock in the entire database such as in SQLite. Litebase locks will be in table level.

- Litebase is faster than SQLite for huge-data load. A simple sample inserting 50.000 rows of a simple table with only one string field (using the configuration for huge-data load on Litebase and a transaction on SQLite), where half of the insertions use a prepared statement, takes 7 % less time on Android devices. For huge-data load, the difference of time taken with each database system can be significant.

Nevertheless, SQLite has the following advantages:

- There is no wasted time to check types when inserting data.
- The space used to store data is smaller because each column data only needs the number of bytes used to represent the current data. For instance, a column declared as `int` will always need four bytes in Litebase even if it is one or null whereas in SQLite one will only need one byte and null will not need any byte at all.
- There are more people working on it, and using and sponsoring it. So, it is more reliable and new features are implemented much faster.

Finally, the TotalCross software LaudoMovel [LaudoMovel] has versions using SQLite and Litebase their performance seemed to be the same.

6. Conclusion, Currently, and Future Work

The number of devices using TotalCross is increasing and will increase even more as it is ported to new mobile platforms such as FirefoxOS and MacOS. Therefore, the number of Litebase users will also increase, turning necessary the implementation of new features in Litebase to satisfy the programmers' needs.

A feature that is being currently implemented is transactions, so that the recovery process will not be needed anymore. The transactions will deal with that. Some future enhancements include concurrency implementation, stronger cryptography, table data compression, outer joins, subqueries, more column types such as `byte` and `decimal`, providing a JDBC API, etc.

References

LaudoMovel Site (2014), https://caixa.laudomovel.com.br/.

Litebase Companion v 2.8 (2014), http://www.totalcross.com.br.

Morgan Stanley Internet Trends Report (2010), http://pt.slideshare.net/alfonsogu/crecimiento-del-uso-de-internet-en-el-mvil.

SQLite Online Documentation (2014), http://www.sqlite.org/docs.html.

TotalCross Companion v 3.0 (2014), http://www.totalcross.com.br.

Ramakrishnan, R. and Gehrke, J. (2003), Database Management Systems, MacGraw-Hill, 3rd edition.

paper:3

pExtract: Uma abordagem probabilística para a extração de esquemas de documentos XML

Geomar A. Schreiner[1], **Denio Duarte**[2]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)

[2]Universidade da Fronteira Sul (UFFS) – Campus Chapecó

geomarschreiner@gmail.com.br, duarte@uffs.edu.br

Abstract. *Data exchange still relies on XML document, mainly in Web applications. However, mostly of these documents are not valid, i.e., there not exists a schema associated to them. Schemas are usefull to optimize XML processing operations. In this context, a tool that extracts a schema from a XML document collection helps applications to process documents efficiently. This paper presents pExtract : a tool to extract XML schema from XML document collection based on bayesian network and extended context-free grammar. We conduct experiments to show pExtract correcteness and efficiency.*

Resumo. *XML é o padrão preferido para troca de dados entre aplicações. Muitos dos documentos XML não possuem esquemas associados tornando assim otimizações de consultas e de armazenamento mais complexas. Ferramentas de extração de esquemas XML são úteis quando se necessita realizar tais otimizações. Este trabalho apresenta o pExtract : uma ferramenta para extração de esquemas XML baseada em rede Bayesiana e em gramática livre de contexto estendida.*

1. Introdução

Documentos XML (*Extensible Markup Language*) ainda são utilizados tanto na área de desenvolvimento quanto na interoperabilidade entre aplicações [Moro et al. 2009]. Porém, a maioria dos documentos são gerados sem um esquema associado [Barbosa et al. 2005]. Um esquema permite que o processamento dos documentos seja menos custoso computacionalmente. Assim, esquemas são úteis pois: (i) auxiliam na consulta dos dados, (ii) otimizam o processamento das consultas, (iii) são base para formas eficientes de armazenamento dos dados, (iv) permitem a criação de classes de documentos e (v) facilitam o processo de mapeamento do documento em outros formatos ($e.g.$, dados relacionais). Neste contexto, ferramentas que extraem esquemas a partir de uma coleção de documentos XML têm um papel importante no processamento eficiente de tais documentos pelas aplicações.

Este artigo apresenta uma ferramenta para extração de esquemas, nomeada pExtract [1]. O pExtract realiza a extração do esquema baseada em uma abordagem probabilística. A abordagem utilizada transforma a coleção de documentos (possivelmente unitária) de um mesmo domínio (estruturalmente semelhantes) como entrada em

[1]Demo disponível em http://pextract.blogspot.com.br

uma rede Bayesiana baseada em crença (grafo direcionado etiquetado com as probabilidades) e a partir deste grafo gera um gramática livre de contexto estendida ($eCFG$ - extended Context-Free-Grammar) que descreve a estrutura da coleção. Posteriormente, o pExtract transforma esta $eCFG$ em uma XSD ($XML\text{-}Schema$).

O restante deste trabalho está organizado da seguinte forma: na próxima seção são apresentadas brevemente algumas abordagens de extração de esquema encontradas na literatura. A Seção 3 apresenta a estrutura do pExtract e seu funcionamento. Na Seção 4 são apresentados alguns experimentos. Finalmente, a Seção 5 apresenta as conclusões deste trabalho.

2. Trabalhos Relacionados

Na literatura são propostas várias abordagens para a extração de esquemas de documentos XML. Em [Garofalakis et al. 2000] é proposta a ferramenta XTRACT que extrai um esquema de um documento XML D criando para cada elemento e de D, um conjunto de sequências de subelementos de e. O conjunto de sequências é submetido a um processo de inferência de expressões regulares. As expressões regulares R são simplificadas para compor o esquema final descrito por uma DTD. [Xing and Parthepan 2011] propõe uma melhoria no processo de simplificação do XTRACT utilizando autômatos finitos para selecionar as melhores R dentre as candidatas. A abordagem proposta em [Min et al. 2003] assemelha-se, também, em parte à abordagem do XTRACT, porém cada um dos elementos de D tem seu conteúdo extraído e, então, é inferida uma regra com notação própria que represente o conteúdo do elemento avaliado. Em seguida, as regras são validadas em relação aos elementos no documento. Embora a gramática possa ser transformada em XSD ou DTD nenhuma transformação é proposta. O XStruct ([Hegewald et al. 2006]) combina as duas abordagens anteriores. Essa abordagem utiliza uma variação do algoritmo proposto em [Min et al. 2003] para realizar a extração de uma gramática G que gere a estrutura do documento. G é simplificada pelo processo de fatoração proposto por [Garofalakis et al. 2000]. A ferramenta extrai atributos e tipos básicos para compor o esquema gerado em XSD.

Uma abordagem diferente é proposta em [Chidlovskii 2001] baseada em inferência gramatical, isto é, para um dado documento XML é inferida uma gramática que descreve sua estrutura. As regras são geradas a partir da árvore XML e depois fusionadas com base no contexto e no conteúdo a fim de simplificar a gramática gerada. Um esquema XSD é gerado a partir das regras extraídas.

A abordagem proposta neste trabalho estende a abordagem proposta em [Pasquali and Duarte 2008] que propõe um algoritmo que constrói um grafo G a partir de uma árvore XML t. Os nós de G representam os elementos e os arcos a probabilidade de ocorrer um relacionamento (pai, filho ou irmão) entre os elementos. Em seguida, G é transformado em uma $eCFG$. O algoritmo proposto não foi implementado e assim não é possível verificar a eficácia e corretude do mesmo. pExtract implementa esse método transformando a coleção XML em uma rede Bayesiana baseada em crença R sem a necessidade de construir uma árvore e em seguida transforma R em uma $eCFG$.

Assim, pExtract difere dos métodos apresentados por criar uma rede Bayesiana baseada em crença para gerar uma gramática e propor um esquema XSD que descreve a coleção de documentos XML de entrada.

3. pExtract

A abordagem proposta é baseada em três passos que serão brevemente explicados a seguir (supondo que a coleção de documentos XML Δ é unitária).

1. Inicialmente, o pExtract cria a rede Bayesiana baseada em crença a partir da leitura de Δ utilizando a API SAX[2]. O parser SAX faz a leitura do documento baseado em eventos, ou seja, a cada elemento encontrado pelo parser são invocados métodos pré-definidos. O início do documento invoca o método *startDocument* e quando o parser alcança o final do documento é invocado o método *endDocument*. Cada abertura de uma *tag* produz uma chamada para um método *startElement* e seu fechamento uma chamada para *endElement*. Esses métodos são utilizados para criar a rede Bayesiana. Por exemplo, a cada chamada do método *startElement* é criado um nodo na rede e são criados os relacionamentos que este nodo possui com os demais. Ao final deste passo, a rede Bayesiana B terá sido criada contendo todos elementos da coleção e conhecimentos probabilísticos das relações entre os elementos (*i.e.*, pai, filho, filho mais à direita e irmão). B é um grafo $G = (V, E)$, onde os nós V representam os elementos de D e as arestas E representam o conhecimento probabilísticos entre os nós de V. Se existir uma aresta entre os nós (elementos) n_1 e n_2 então existe uma relação de parentesco entre n_1 e n_2. Assim, E armazena o conhecimento probabilístico de n_1 ser pai, filho, filho mais à direita ou irmão de n_2. A Figura 2 apresenta um exemplo de B.

2. A rede Bayesiana de crença gerada é utilizada para propor uma $eCFG$ que contém a estrutura do documento. O nó inicial na rede Bayesiana (sem arcos entrantes) representa o nó raiz da árvore do documento XML e tem o rótulo transformado na cabeça de uma regra da $eCFG$ a ser gerada. Este rótulo é o símbolo inicial da gramática (nó de cor vermelha na Figura 2). A produção desta regra é formada pelos nós que possuem uma aresta do tipo "c" ($child$) partindo do nó raiz. Após a criação da regra de produção do nó raiz, todos os outros nós são visitados executando o mesmo processo. Para o nó etiquetado $PLANT$ da Figura 2, por exemplo, é criada a seguinte regra de produção $plant \leftarrow common\ botanical\ zone\ light\ price\ availability$ pois $PLANT$ tem arestas para os nós etiquetados com os elementos do corpo da regra e os nós que compõem o corpo da regra tem a relação de irmão ("s" - $sibling$) entre si.

3. O último passo transforma a $eCFG$ em um esquema na linguagem XSD. As regras utilizadas para a conversão dos elementos da gramática são:
3.1. As cabeças das regras da $eCFG$ gerada são transformadas em elementos;
3.2. Regras cuja produção é vazia (*i.e.*, ϵ) geram elementos simples do tipo *string*;
3.3. Regras cujas produções são compostas (*i.e.,* $A \rightarrow (BC|D)$) geram elementos de tipo complexo;
3.4. Produções concatenadas (unidas pelo operando *e*) são declaradas dentro de delimitadores de grupo do tipo *sequence*;
3.5. Produções unidas pelo separador | (operador *ou*) são declaradas em delimitadores de grupo do tipo *choice*;
3.6. Produções que possuem o fecho *positivo* (respectivamente *negativo*) de Kleene ($^+$) (respectivamente (*)) devem possuir $minOccurs = 1$ (respectivamente $minOccurs = 0$) e $maxOccurs = unbounded$, não limitando o número de repetições aceitas;
3.7. Produções que são seguidas do símbolo "?" devem possuir *minOccurs* = 0;

[2] `http://www.saxproject.org/`

O pExtract foi implementado na linguagem Java. A utilização do sistema pode ser feita utilizando um modo gráfico (GUI) ou em modo texto. As formas de utilização do sistema (com interface ou sem) não interferem no resultado gerado, já que ambas utilizam as mesmas classes para realizar a extração.

A Figura 1 apresenta a interface criada para o pExtract . Na interface, o usuário seleciona a coleção (documento ou pasta) XML para extração do esquema (Figura 1a). O campo $output$ apresenta o nome e o local onde o esquema será gravado (Figura 1b). A Figura 1c apresenta a área de visualização que dependendo do contexto pode ser a coleção ou a XSD gerada. Ao clicar no botão *Run*, o processo de extração de esquema é iniciado. O painel localizado à esquerda da interface do pExtract (Figura 1d) apresenta as etapas que o pExtract realiza. A aba *Main* é a janela principal. A aba *XML* mostra a coleção de entrada para a extração. A aba *Graph* apresenta a rede Bayesiana de Crença criada a partir da coleção XML (Figura 2). Na aba *eCFG* é apresentada a $eCFG$ criada a partir do grafo. E a aba *XSD* apresenta a XSD gerada.

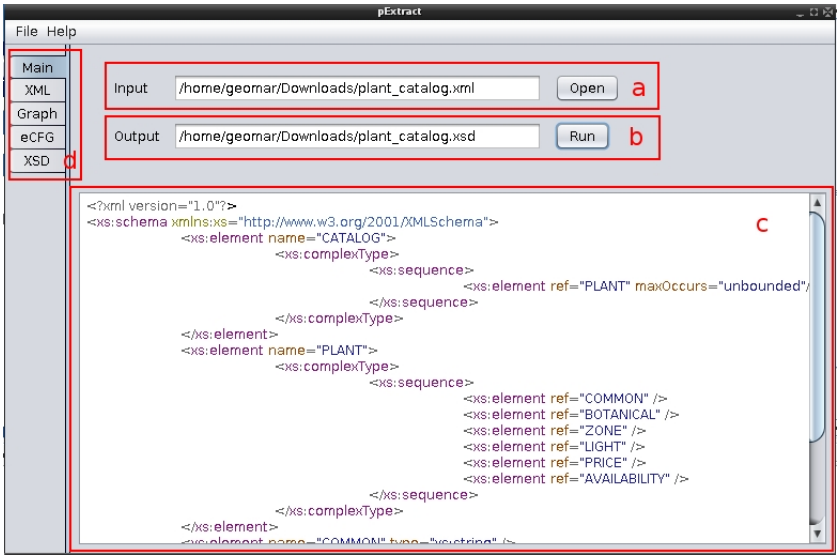


Figura 1. Interface pExtract - Visualização XSD

A Figura 2 apresenta a visualização da aba *Graph* onde é exibido o grafo gerado a partir de um documento XML (Figura 1a). O grafo apresenta um nó na cor vermelha (*CATALOG*) que representa a raiz do documento. Os demais nós do grafo (elementos internos) são azuis. Verifica-se também pela figura, que as arestas do grafo apresentam duas cores. As arestas verdes representam relacionamentos do tipo filho entre a origem da aresta e seu destino. Já as arestas alaranjadas representam relacionamentos do tipo irmão à direita. Para a melhor visualização do grafo optou-se por não exibir auto-relacionamentos (esse tipo de relacionamento indica que o elemento representado pelo nó será representado pelo fechamento positivo ou negativo de *Kleene* na gramática).

Em termos de complexidade de tempo, a transformação da coleção de documentos C no grafo da rede Bayesiana B é $O(n)$, onde n é o número de elementos de C. A construção da $eCFG$ a partir de B é também $O(k)$, onde k é o número de nós de B (elementos distintos de C). Finalmente, a transformação da $eCFG$ no esquema é $O(r)$, onde r é o número de símbolos em $eCFG$.

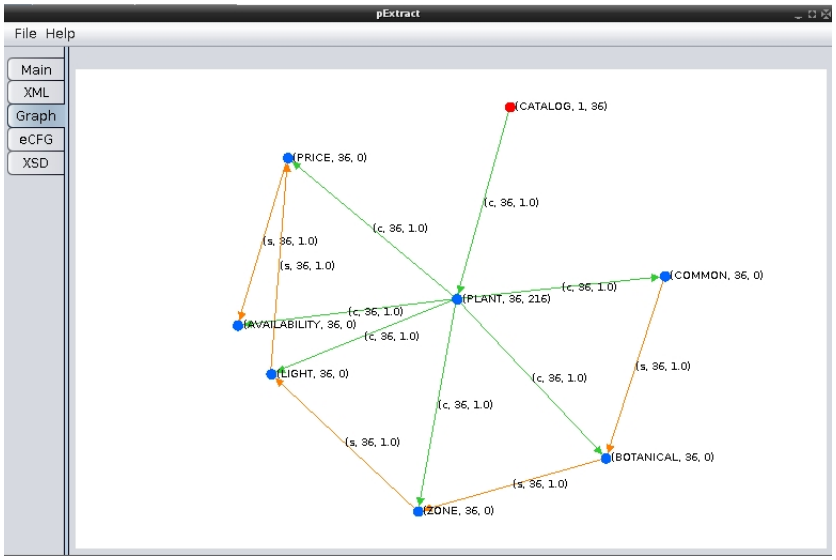


Figura 2. Interface pExtract - Visualização grafo

4. Experimentos

O gráfico (Figura 3) apresenta resultados de experimentos realizados utilizando o pExtract . A série *Documentos P. 4* representa quatro documentos XML contendo 8 elementos distintos, combinados com profundidade 4 e com o total de 146, 508.488, 20.156.021 e 4.528.060.093 elementos, respectivamente. Já *Documentos P.10* segue a mesma lógica mas com 19 elementos distintos, combinados com profundidade 10. O gráfico apresenta em seu eixo Y a quantidade de segundos que o teste levou para ser executado. No eixo X é apresentado o número de elementos (com repetições) de cada experimento. Percebe-se que os documentos com maior número de elementos tiveram tempos de extração superior a 200 segundos. Esses documentos têm tamanhos $7Gb$ e $22Gb$, respectivamente. Através do gráfico percebe-se que o pExtract realiza a extração do documento em um tempo relativamente pequeno, somente ultrapassando a marca de 1 minuto para documentos com um número elevado de elementos.

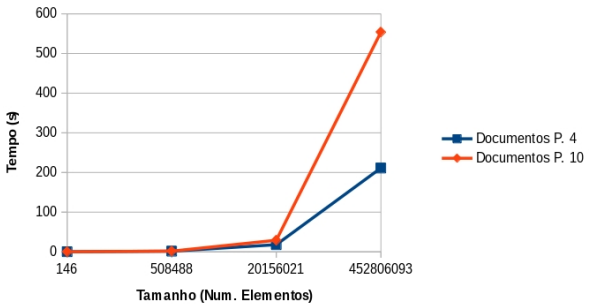


Figura 3. Grafico experimento

O experimento foi realizado em uma estação com sistema operacional Ubuntu Linux 13.04 (32 bits) com um processador Intel Core i5 - 480M (2,66GHz, Cache 3MB) com 4GB de memória RAM. Para computar os tempos foi utilizado o comando *time* do Linux. Os documentos utilizados como entrada foram gerados a partir de um esquema

conhecido utilizando o software OxygenXML[3]. Os resultados foram comparados de maneira manual (visualmente) com o esquema original pois não foram encontradas métricas de verificação. Para todos os teste realizados o pExtract gerou um esquema compatível com o original. Como não foram encontradas implementações dos outros métodos, não foi possível fazer a comparação dos mesmos com o pExtract .

5. Conclusões

Documentos XML ainda são muito utilizados para transferência de dados entre aplicações na web. A maior parte dos documentos XML não possuem um esquema associado ou não respeitam as restrições impostas por ele. Esquemas permitem que sejam realizadas otimizações no processamento de documentos XML. Neste contexto, este trabalho apresentou o pExtract , uma ferramenta que realiza a extração de esquemas de uma coleção (possivelmente unitária) de documentos XML. O pExtract realiza a extração do esquema baseado em 3 passos: (i) realiza o parse da coleção criando uma rede Bayesiana baseada em crença, (ii) com base na rede é criada uma $eCFG$ e (iii) a $eCFG$ é convertida em um esquema XML na linguagem XSD. O pExtract cumpre seu propósito, gerando um esquema XML compatível com o esquema original do documento XML com um baixo custo computacional pois sua complexidade de tempo é linear. Em [Schreiner 2014], pExtract é descrito com mais detalhes.

Referências

Barbosa, D., Mignet, L., and Veltri, P. (2005). Studying the XML Web: gathering statistics from an XML sample. *World Wide Web*, pages 1–32.

Chidlovskii, B. (2001). Schema Extration from XML Data: A Grammatical Inference Approach. *KRDB01 Workshop (Knowledge Representation and Databases)*.

Garofalakis, M., Gionis, A., and Rastogi, R. (2000). XTRACT: a system for extracting document type descriptors from XML documents. *ACM SIGMOD*, pages 165–176.

Hegewald, J., Naumann, F., and Weis, M. (2006). XStruct: Efficient schema extraction from multiple and large XML documents. In *Proceedings of ICDEW '06*. IEEE.

Min, J.-K., Ahn, J.-Y., and Chung, C.-W. (2003). Efficient extraction of schemas for XML documents. *Information Processing Letters*, 85(1):7–12.

Moro, M. M., Braganholo, V., Dorneles, C. F., Duarte, D., Galante, R., and Mello, R. S. (2009). Xml: Some papers in a haystack. *SIGMOD Rec.*, 38(2):29–34.

Pasquali, F. and Duarte, D. (2008). O Uso da Probabilidade para DescreverArvores (XML). *XXIII Simposio Brasileiro de Banco de Dado*, (Xml).

Schreiner, G. A. (2014). Extração de esquemas de documentos XML: Uma abordagem probabilística. TCC, Universidade Federal da Fronteira Sul.

Xing, G. and Parthepan, V. (2011). Efficient schema extraction from a large collection of XML documents. *Proceedings of the 49th ACM-SE*, page 92.

[3] `www.oxygenxml.com`

paper:5

PhoCA: An extensible service-oriented tool for Photo Clustering Analysis

Yuri A. Lacerda[1,2], Johny M. da Silva[2], Leandro B. Marinho[1], Cláudio de S. Baptista[1]

[1] Laboratório de Sistemas de Informação
Federal University of Campina Grande (UFCG) Campina Grande, PB – Brazil

[2] Federal Institute of Education, Science and Technology of Ceará (IFCE)
Crato, CE – Brazil

{yurilacerda,johny.moreira}@ifce.edu.br,
{lbmarinho,baptista}@dsc.ufcg.edu.br

Abstract. *Clustering algorithms are at the core of most of the research work in knowledge discovery using geo-tagged photos, e.g., a cluster of photos might identify existing or new points (or areas) of interest. But before clustering the data, researchers need to devote a lot of time and effort for collecting, pre-processing, and analyzing the data. For assisting researchers in these tasks, we propose PhoCA[1] (Photo Clustering Analyzer), an extensible web-based tool for photo clustering analysis. It aims at assisting the researcher in tasks such as data collection, clustering analysis, and visualization. PhoCA features tools for collecting and extracting metadata of geo-referenced photos; suites of state-of-the-art photo clustering algorithms; spatial visualization tools; etc.*

1. Introduction

The current generation of domestic and professional digital cameras enables the production of high quality photos enhanced with rich contextual information, such as the date, time, and location of the photo at shooting time, as well as the direction of the camera in the moment of shooting [Lacerda et al. 2012].

The large amount of photos available in the Web, combined with their rich contextual metadata, opens new opportunities for knowledge discovery and data mining applications. Some recent areas of research worth mentioning concern the automatic organization of personal photo collections [Figueirêdo et al. 2012]; the automatic planning of touristic routes [Popescu and Grefenstette 2009]; the automatic discovery of points of interest (POI) [Lacerda et al. 2012]; etc.

In all these applications, clustering algorithms appear as one of the main tools. For example, most of the existing research on detection of points of interest from digital photos is based on new or modified versions of existing clustering algorithms [Lacerda et al. 2012][Yang et al. 2011][Kisilevich et al. 2010]. In the area of recommender systems, the authors of [Matyas and Schlieder 2009][Marinho et al. 2012] propose to first identify areas of interest through clustering of geo-referenced photos, and then recommend the detected areas of interest to the users.

[1] A screencast and the PhoCA installation files are available on: http://www.yurilacerda.com/phoca.

However, before one can finally run clustering algorithms for investigating the aforementioned applications, one needs to first collect and pre-process the data, which is very time consuming, especially for new researchers in the area. This involves studying the data access API of the photo repository of interest and writing specific scripts for collecting the photos and extracting the metadata of interest. Thereby, it would be of great value to have applications that could assist the researchers in these tasks.

In order to fulfill this need, we propose PhoCA (Photo Clustering Analyzer) an extensible web-based tool for photo clustering analysis. PhoCA aims at assisting researchers in this field through the most important tasks of the KDD process, such as data collection, pattern recognition, analysis, and visualization.

This work is organized as follows. In the next section, the related work is presented. Section 3 presents the architecture and describes the modules that comprise PhoCA. Section 4 presents a case study using PhoCA. Finally, the Section 5 concludes the paper and discusses opportunities for future work.

2. Related Work

Although there are many works about using clustering algorithms for different applications, to the best of our knowledge PhoCA is the first specialized tool for assisting researchers in most of the steps of the KDD process for this particular domain. Some of the most popular problems in this domain are: the point of interest detection ; identification of representative images of landmarks; and automatic planning of touristic routes.

Regarding points of interest detection, most of the approaches cluster the photos based on their geographic distances. The assumption is that many photos taken close to each other tend to indicate points of interest. The most common algorithms found in the literature for this end are: k-means, mean shift, DBSCAN [Kriegel et al. 2012], and spectral clustering [Yang et al. 2011].

Some studies use DBSCAN to cluster geo-referenced photographs [Lacerda et al. 2012][Kisilevich et al. 2010], however, the original DBSCAN algorithm does not allow clusters with density variations [Han et al. 2011]. In [Kisilevich et al. 2010], it is proposed P-DBSCAN, a new approach that uses the DBSCAN algorithm, taking into consideration just a local density, that is, allowing the creation of clusters with different sizes and densities.

In [Lacerda et al. 2012] we proposed a new clustering algorithm that uses photo orientation and geotags of photographs to discover new points of interest. The algorithm creates line segments using photo orientation and uses the points of interceptions of these lines and some photo positions as input for the clustering process. All these algorithms are available in PhoCA.

In the reviewed literature, we did not find any tool that is similar to PhoCA. Instead, most of the data mining tools available are of general purpose, such as Weka [Hall et al. 2009].

However, differently from them, PhoCA handles the specific task of clustering large collections of photos. Thus, most of the functionalities of PhoCA are not featured by the aforementioned tools.

3. System Architecture

Figure 1 presents the architecture of PhoCA. The system has a service-oriented architecture, which means that its functionalities are available through the invocation of Web services. This enables the access of the functionalities of PhoCA from other software, independent of programming languages and system platform.

PhoCA is a RESTful application. The user interface was developed using HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) and Javascript. Also, the JQuery Javascript Library was used to enable rapid Web development, handling events and Ajax interactions with the services.

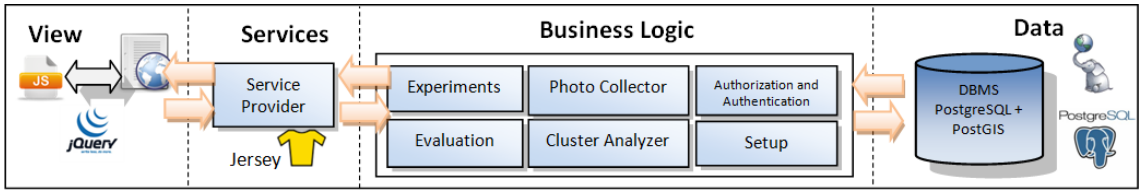


Figure 1. PhoCA Architecture

The model layer has the java classes responsible for the business logic. It divides the system into six modules: Photo Collector; Experiments; Cluster Analyzer; Evaluation; Authorization and Authentication; and Setup.

The first module is responsible for the extraction of metadata from local photos or Flickr. The system allows the storage and metadata extraction of photos uploaded by the users. In case of photos from Flickr, the user must inform a specific bounding box area to extract photo metadata and a keyword. Depending on the chosen parameters, in especial the bounding box size, this process could be expensive, e.g., taking many hours or even days. Therefore, there are some parameters (eg: number of threads used on the photo collector) on the Setup Module that must be tuned for performance.

The Experiments module allows the execution of a clustering algorithm for a specified collection. PhoCA provides five clustering algorithms, however the user could easily add new algorithms using some object-oriented programming elements and adding some lines in a XML file.

The next module, Cluster Analyzer, is responsible for presenting summary statistics related to the computed clusters. Beside this, the Evaluation module allows the users to evaluate if a specific photo is correctly associated to a group using a Graphical User Interface (GUI). Moreover, this module calculates the metric of precision based on the percentage of photos clustered correctly. This is a common metric in information retrieval field.

The Summary Statistics module is responsible for presenting simple statistics for a specified experiment, such as: the experiment identification, cluster method and parameters used; data and time of execution; running time (in milliseconds); and the mean, minimum and maximum cluster size. Moreover, this module is in charge of presenting the size of each cluster, allowing the visualization of these elements in a map, Figure 2.

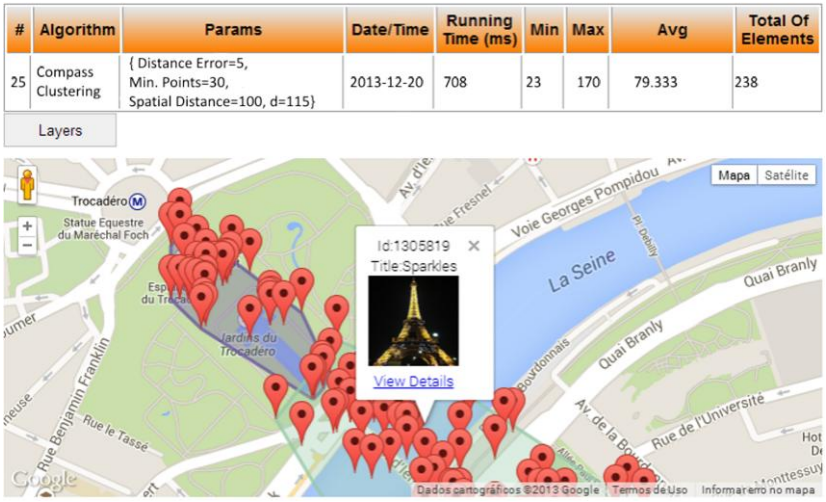


Figure 2. The Summary Statistic Module

The Authorization and Authentication module is in charge of the system's security, controlling to resources and functionalities. Furthermore, it allows the creation and manipulation of system users. There are two user profiles: administrator and default user. Only the administrator profile has privileges to create new users and changing the system's settings. The default user profile has privileges to use the other modules through system's GUI or invoking its Web services from other application.

The Setup module is responsible for setting the database access, Flickr API Key and performance tunings. PhoCA must be installed on a Web container. A Web container is a component of a Web server that implements the Java Servlet specification. Otherwise, it is not possible to access its functionalities. PhoCA uses as database management systems the PostgreSQL[2] and the spatial extension Postgis[3].

4. Case Study

In this section, we present a case study in order to evaluate the proposed tool. This case study runs over all the knowledge discovery process, i.e., it starts with the collection and storage of photos, and culminates with the execution of experiments using the Compass Clustering algorithm proposed in [Lacerda et al. 2012] for landmark detection.

We consider a photo related to a landmark as a photo captured inside the landmark area or that contains the landmark in the image content. The idea here is basically to check which size of the line segment, one the parameters required by Compass Clustering, provides the best results for each data set.

It was extracted three photo collection of a bounding box area around 500 meters of 3 well-known POIs: Eiffel Tower (Paris, France), Statue of Liberty (New York, USA) and Colosseum (Rome, Italy). We extracted metadata from 36,230 geo-tagged photos from which 1,109 are oriented. These photos were filtered manually to discard invalid landscape photos. Finally, we used 704 valid landscape photos in our evaluation: Eiffel Tower (316 photos), Colosseum (277 photos) and Statue of Liberty (111 photos). These

[2] http://www.postgres.org
[3] http://www.postgis.org

collections had 71,51%, 85,92% and 84,68% of its photos related to respective landmark. The valid landscape photos contain correct data about orientation and geo-location and they haven't focus in a specific object. We made a manual inspection for each photo.

We executed the experiments using the Compass Clustering algorithms with the line segmentation size (d) varying between 20 and 500 meters with steps of 20 meters. It was used the DBSCAN algorithm as the internal clustering algorithm of Compass Clustering. We used the parameters minimum points = 30 and spatial distance = 100 meters. These values were inferred by subjective observation. We set the constant distance error = 5 meters. This constant does not consider the interception points of photos that are less than the distance error. This was motivated because of GPS imprecision of cameras and smartphones GPS chips.

We analyzed the metrics precision and recall. Precision was defined as the ratio between the total of photos related to landmark in the cluster and the total of elements of the cluster itself. Recall is computed as the ratio between the number of photos related landmark in the cluster and the total number of photos related landmark.

Figure 3 presents the average of precision p and recall r of all clusters for the three collections: Eiffel Tower (p1 and r1), Colloseum (p2 and r2) and Statue of Liberty (p3 and r3). The precision and recall have become almost constant to values of d > 360 meters. In other words, the increasing of d hasn't difference to values higher.

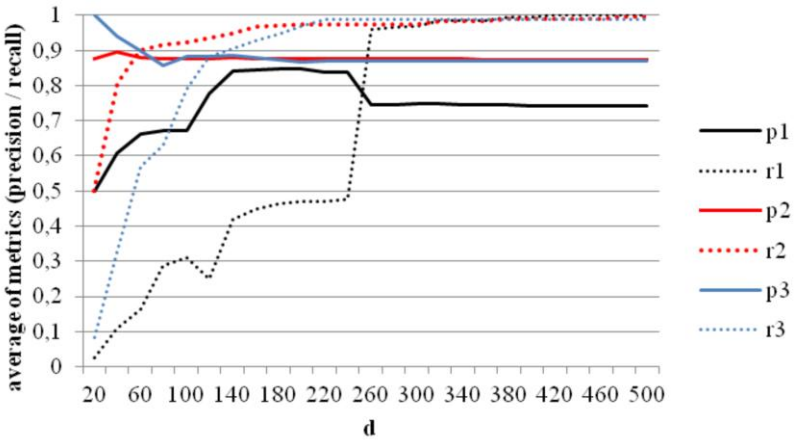


Figure 3. Precision and Recall of Experiments

5. Conclusion and Future Works

This paper presented PhoCA, an extensible web-based tool for photo clustering analysis. This tool features: public access through web services; extracting metadata of photos from Flickr or local photo repositories; clustering photos using state-of-art algorithms; and the analysis of clusters through a geo-spatial visualization tool.

Among the reviewed related work, none of them proposes a specific tool for photo clustering analysis, including tools for all the steps of the KDD process. We presented an experiment for landmark detection using the algorithm Compass Custering. The algorithm presented the influence of the parameter line segmentation.

Future work include to extend the suite of clustering algorithms available with other algorithms recently used for knowledge discovery in large photo collections (e.g., mean-shift) and to perform comparisons between several algorithms. We also plan to enhance the Cluster Analyzer module with more sophisticated filters, such as the visualization of the trajectory of the user while shooting photos, and the visualization of photos that were shot within a given period of time.

6. Acknowledgment

The authors are grateful to the CNPq and IFCE for funding this research project.

7. References

Figueirêdo, H. F., Lacerda, Y. A., Paiva, A. C., Casanova, M. A. and Baptista, C. S. (2012). PhotoGeo: a photo digital library with spatial-temporal support and self-annotation. *Multimedia Tools and Applications*, 59 (1): 279-305.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1), 10-18.

Han, J., Kamber, M. and Pei, J. (2011). Data Mining: Concepts and Techniques. Elsevier, 3th edition.

Kisilevich, S., Mansmman, F., and Keim, D.A. (2010) P-dbscan: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos. In *Proceedings of COM.Geo'10*.

Kriegel, H.-P., Sander, J., Ester, M. and Xu. X (2012) A density-based algorithm for discovering clusters in large spatial database with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226-231.

Lacerda, Y. A., Feitosa, R., Esmeraldo, G. A., Baptista, C. de S. and Marinho, L. B. (2012) Compass clustering: a new clustering method for detection of points of interest using personal collections of georeferenced and oriented photographs. In *Proceedings of the Webmedia '12*. pages 281-288.

Marinho, L. B., Sandholm, T., Baptista, C. de S., Nunes, I., Nóbrega, C. and Araújo, J. (2012) Extracting Geospatial Preferences Using Relational Neighbors. In *Journal of Information and Data Management (JIDM)*, 3 (3): 364-377.

Matyas, C. and Schlieder, C. (2009) A spatial user similarity measure for geographic recommender systems. In *Proceedings of Third International Conference on GeoSpatial Semantics*, vol. 5892 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pages 122–139.

Popescu, A.; Grefenstette, G. (2009) Deducing trip related information from flickr. In *Proceedings of WWW'09*, pages 1183–1184.

Yang, Y., Gong, Z. and Hou, L. (2011) Identifying Points of Interest by Self-Tuning Clustering. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval (SIGIR'11)*, Pequim, China.

paper:2

QualityStamp – Avaliando a Qualidade de *Linked Datasets* para Aplicações de Domínio Específico

Walter Travassos[1], Damires Souza[2], Bernadette Lóscio[1]

[1]Centro de Informática/UFPE, Brasil

[2]Instituto Federal de Educação, Ciência e Tecnologia da Paraíba/IFPB, Brasil

wts@cin.ufpe.br, damires@ifpb.edu.br, bfl@cin.ufpe.br

Abstract. *The increasing structure of the Web of Data enables a set of new applications that can make use of multiple Linked Datasets. However, the choice of datasets that meet the application´s requirements is not an easy task. The large number of datasets available on the web and the lack of information on the provenance and quality of data have brought a major challenge: how to identify the best Linked Datasets to a given application? With this in mind, this paper presents the QualityStamp, a tool which evaluates the quality of Linked Datasets, by taking into account the application requirements.*

Resumo. *A estrutura crescente da Web de Dados possibilita uma série de novas aplicações que podem fazer uso de múltiplos Linked Datasets. Contudo, a escolha de datasets que atendam aos requisitos de uma aplicação não é uma tarefa fácil. O grande número de datasets disponíveis na Web, bem como a falta de informações sobre a proveniência e a qualidade destes dados, traz a tona um grande desafio: como identificar os melhores Linked Datasets para uma determinada aplicação? Com base nisso, este trabalho apresenta a QualityStamp, uma ferramenta que avalia a qualidade de Linked Datasets considerando os requisitos da aplicação.*

1. Introdução

A adoção dos princípios de *Linked Data* tem contribuído para o crescimento da Web de Dados, uma vez que estes recomendam a publicação de dados em um formato estruturado (modelo RDF) e incentivam a criação de ligações entre os recursos descritos. A facilidade de navegação entre conjuntos de dados distintos é um dos diferenciais dos *Linked Datasets*, *i.e.*, conjuntos de dados definidos de acordo com os princípios de *Linked Data*. A Web de Dados, por sua vez, possibilita uma série de novas aplicações que podem fazer uso de múltiplos *Linked Datasets* (por simplificação, chamados aqui de *datasets*) a fim de prover informações úteis para comunidades de usuários interessados em um domínio de dados em particular, por exemplo, "Dados bibliográficos", "Saúde", "Música" (Heath e Bizer, 2011). Porém, a escolha de *datasets* relevantes para uma determinada aplicação, ou seja, *datasets* que atendam aos requisitos da aplicação não é uma tarefa fácil. De uma maneira geral, o grande número de *datasets* atualmente disponíveis na Web, bem como a falta de informações sobre a proveniência e a qualidade destes dados, traz a tona um grande desafio: como avaliar a qualidade de *datasets* que serão usados por uma dada aplicação.

Neste contexto, este trabalho propõe uma ferramenta, denominada *QualityStamp*[1], para avaliação da qualidade de *Linked Datasets*. Neste trabalho, o

[1] *Screencast*: http://youtu.be/yXtB-mJnkeU

termo "qualidade" é associado à descrição geral de "adequação ao uso" proveniente da definição clássica de Qualidade da Informação (Wang e Strong, 1996). A avaliação, então, tem como objetivo auxiliar a escolha dos *datasets* mais adequados para uma dada aplicação, ou seja, os *datasets* capazes de contribuir com dados relevantes para a mesma. A avaliação proposta leva em consideração múltiplos critérios e, para cada um dos possíveis critérios, uma métrica foi devidamente especificada e implementada. A avaliação de qualidade é guiada pelos requisitos da aplicação, os quais são representados por meio de um conjunto de consultas e dos requisitos não funcionais, que correspondem aos critérios de qualidade mais indicados para a aplicação. Dessa forma, no momento da avaliação, um especialista do domínio (ED) deverá escolher os critérios de qualidade que mais se adequam à aplicação.

Este trabalho está organizado da seguinte forma: a Seção 2 apresenta a abordagem *QualityStamp*. A Seção 3 descreve a ferramenta e suas funcionalidades por meio de um exemplo. A Seção 4 apresenta alguns trabalhos relacionados. A Seção 5 tece algumas considerações e indica trabalhos futuros.

2. Abordagem de Avaliação da Qualidade de *Linked Datasets*

A Qualidade da Informação (QI) é um aspecto multidimensional definido a partir de um conjunto de critérios (Naumann, 1998). Como a qualidade é uma condição dependente do contexto em que é empregada, além de algumas vezes depender do ponto de vista do usuário, é necessário definir mecanismos adequados de medição desses critérios. Por isso, para cada critério de QI é associada uma métrica de avaliação.

A abordagem proposta faz o uso de critérios com métricas objetivas, ou seja, que não necessitam da intervenção do usuário para serem aferidas, com o objetivo de avaliar três importantes características dos *datasets*: (*i*) o desempenho: diz respeito à capacidade do *dataset* em responder às solicitações da aplicação, determinado pelos critérios de disponibilidade, tempo de resposta e escalabilidade; (*ii*) a completude: está relacionada a quanto um *dataset* contribui para responder a um conjunto de consultas da aplicação, *i.e.*, o quanto da informação presente nos *datasets* candidatos é útil do ponto de vista dos requisitos da aplicação; e (*iii*) *interlinking*: diz respeito ao grau de interligações que um dado *dataset* possui com outros. Quanto mais ligações possuir, mais o *dataset* poderá fornecer informações relacionadas aos seus recursos.

Um dos diferenciais da abordagem é que a avaliação da qualidade é realizada levando-se em consideração os requisitos da aplicação, os quais podem ser capturados de duas formas: (*i*) por um conjunto de consultas relevantes para a aplicação, *i.e.,* consultas realizadas com frequência, e (*ii*) por um conjunto de requisitos não funcionais, os quais são definidos por meio dos critérios de qualidade escolhidos para a avaliação.

A avaliação da qualidade pode ser dividida em duas etapas principais. A primeira refere-se à identificação dos requisitos da aplicação, onde são obtidos os seguintes dados iniciais: (*i*) o conjunto de *linked datasets* candidatos que serão avaliados; (*ii*) um conjunto de consultas *select* SPARQL relevantes para a aplicação (que definem os requisitos de dados); e (*iii*) os critérios de QI (requisitos não funcionais) que serão utilizados no processo com seus respectivos pesos. A segunda etapa realiza, de fato, a avaliação da qualidade, a partir dos critérios e métricas explicados na Tabela 1. Ao final destas etapas, será gerada uma classificação dos *datasets* candidatos.

Tabela 1 – Categorias, critérios e métricas.

Categoria	Critério	Métrica	Detalhes
Desempenho	Disponibilidade	$\sum_{i=1}^{n} \frac{\lvert A_i \rvert}{n}$ onde, $\lvert A_i \rvert$ é o total de consultas SPARQL padrão respondidas (*true*) para um *linked dataset*; e *n* é o total de consultas SPARQL padrão submetidas	Uma consulta SPARQL padrão, *i.e. ASK WHERE {?s ?p ?o}*, é utilizada para verificar a disponibilidade do *dataset*. Caso a resposta seja *true*, o dataset está disponível. Esta verificação é realizada por um período, assim, é gerado um conjunto $\lvert A_i \rvert$ contendo todos os momentos que o *dataset* esteve disponível.
	Tempo de Resposta	$\sum_{i=1}^{n} \frac{\lvert RT_i \rvert}{n}$ onde, $\lvert RT_i \rvert$ é a média do conjunto de tempos de resposta para uma consulta SPARQL padrão; e *n* é o total de consultas SPARQL padrão submetidas.	É o tempo que um *dataset* leva para responder uma dada consulta. Da mesma forma que a disponibilidade, esta avaliação é realizada por um determinado período de tempo. Assim, será realizada uma média entre os valores de vários tempos de resposta aferidos por um dado período.
	Escalabilidade	$((teste\ 1 * w1) + (teste\ 2 * w2) + (teste\ 3 * w3))$ onde, w_i são pesos atribuídos a cada teste. A soma dos três pesos deve ser igual a 1.	É verificado se o tempo de resposta de uma única consulta é maior que a média de tempo de *n* consultas lançadas em rajada com um intervalo de 1 segundo entre elas. O escore é gerado pela porcentagem de variação entre *a média dos tempos de respostas de n consultas* e o tempo de resposta de *uma única consulta*. Assim, para gerar o escore são executados três testes com diferentes valores para *n*: 20, 30 e 50 consultas. Ao final, somam-se os valores dos testes.
Completude	Completude de Esquema	$\frac{\sum_{k=1}^{n} \lvert Rs_{i(k)} \rvert}{\sum_{j=1}^{n} \lvert PT_{A(j)} \rvert}$ onde, $\lvert RS_i \rvert$ é o número total de consultas que um *dataset* s_i respondeu para os padrões de triplas procurados; $\lvert PT_A \rvert$ é o total de padrões de triplas sem literais; *k* representa cada padrão de tripla sem literal encontrado no *dataset* e *j* representa cada um dos padrões de triplas sem literais extraídos do conjunto de consultas.	A completude do esquema irá mensurar o quanto cada *dataset* consegue responder um conjunto de padrões de triplas sem literais extraídos do conjunto de consultas *select* definido. Cada padrão de tripla sem literal irá formar uma consulta ASK SPARQL. Assim, o escore desta completude é a razão entre o número de consultas que retornarem verdadeiro pelo total de padrões de triplas sem literais.
	Completude de Literal	$\frac{\sum_{k=1}^{n} \lvert Rs_{i(k)} \rvert}{\sum_{i=1}^{n} \lvert PTL_{A(i)} \rvert}$ onde, $\lvert RS_i \rvert$ é o número total de consultas que um *dataset* s_i respondeu para os padrões de triplas procurados; $\lvert PTL_A \rvert$ é o total de padrões de triplas com literais; *k* representa cada padrão de tripla com literal encontrado no *dataset* e *i* representa cada um dos padrões de triplas com literais extraídos do conjunto de consultas.	A completude de literal irá verificar em cada *dataset* a presença de padrões de triplas com literais extraídos do conjunto de consultas *select* definido. A avaliação da completude de literal irá gerar um escore de quanto cada *dataset* contribui para responder o conjunto de padrões de triplas com literais.
	Completude de Instância	$\frac{\sum_{k=1}^{n} \lvert Rs_{i(k)} \rvert}{\sum_{i=1}^{n} \lvert TDV_{A(i)} \rvert}$ onde, $\lvert RS_i \rvert$ é o número total de consultas que um *dataset* s_i respondeu para os padrões de triplas procurados; $\lvert TDV_A \rvert$ é o total de termos descritos por meio de um vocabulário; *k* representa cada recurso de dado encontrado no *dataset* e *i* representa cada um dos recursos de dados extraídos do conjunto de consultas.	Mensura o quanto cada *dataset* contribui com recursos que representam instâncias de classes (ou recursos de dados) procuradas pelo conjunto de consultas *select* definido, ou seja, o quanto cada *linked dataset* pode responder se os termos do conjunto de Termos Descritos por um Vocabulário (TDV_A) estão presentes entre suas triplas.
Interlinking	Interlinking	$\frac{total\ de\ triplas\ com\ predicado\ de\ interlinking}{total\ de\ triplas\ do\ dataset}$	O escore do *interlinking* é gerado por meio das propriedades de *interlinking*: *owl:sameAs, owl:equivalentClass* e *owl:equivalentProperty*. A ferramenta conta quantas triplas com predicado de *interlinking* existem e, em seguida, divide esse valor pelo total de triplas existentes no *dataset*.

A avaliação dos critérios de qualidade relacionados ao desempenho faz uso de consultas SPARQL a fim de identificar se um conjunto de dados está disponível ou não, bem como avaliar o tempo de resposta para uma consulta. Por outro lado, a avaliação dos critérios relacionados à completude requer uma análise mais detalhada e, para isso, torna-se necessário uma avaliação individual dos padrões de triplas que compõem o BGP (*Basic Graph Pattern*) das consultas SPARQL informadas na primeira etapa do processo de avaliação. A completude pode ser dividida em completude do esquema e completude de dados, sendo esta última representada pela completude de literais e pela completude de instâncias. A completude do esquema irá avaliar quanto um *dataset* pode

contribuir para responder um conjunto de padrões de triplas que não possuem literais. A completude de literais avalia o quanto um *dataset* pode responder a um conjunto de padrões de triplas com literais. E a completude de instâncias mensura o quanto o *dataset* pode responder a um conjunto de recursos de dados. A partir do BGP de cada consulta *select*, três conjuntos são extraídos: (*i*) conjunto de padrões de triplas sem literais; (*ii*) conjunto de padrões de triplas com literais; e o (*iii*) conjunto dos termos que estão descritos por meio de algum vocabulário. Neste último, cada termo é um elemento (sujeito, predicado ou objeto) não repetido, existente nos padrões de triplas extraídos das consultas *select*. Estes elementos não são nem uma classe, nem uma propriedade, mas instâncias de classe sendo representadas por meio de um vocabulário. Estes três conjuntos são utilizados nas métricas associadas ao critério de completude.

Como a *QualityStamp* adota uma abordagem baseada em múltiplos critérios, encontrar a medida de qualidade de cada *dataset* candidato se torna um problema de decisão com múltiplos critérios (MCDM - *Multi-Criteria Decision Making*) (Naumann *et al.*, 1998). Para isso, após a geração dos escores de cada critério utilizado na avaliação, é utilizado o método de decisão *SAW* (*Simple Additive Weighting*) para encontrar a medida de qualidade de cada *dataset*. Este método permite a ponderação sobre os critérios definidos e faz uma classificação a partir da influência de cada critério na análise global de cada *dataset*, identificando cada critério como positivo ou negativo. Neste sentido, tanto o tempo de resposta quanto a escalabilidade são critérios negativos, *i.e.*, quanto menor seus escores, melhor. Já os critérios de disponibilidade, completude e *interlinking* são critérios positivos, *i.e.*, quanto maior os escores, melhor. Os pesos que o ED atribuiu a cada critério de QI no início da avalição são utilizados neste momento para gerar a medida geral de qualidade. Para isso, os valores dos escores obtidos são normalizados em valores comparáveis (entre zero e um) e a ponderação é realizada. A próxima seção apresenta a ferramenta que automatiza esse processo.

3. A Ferramenta *QualityStamp* na Prática

A ferramenta *QualityStamp* foi implementada no formato de aplicação Web e está disponível para uso[2]. Para o desenvolvimento, foi utilizada a linguagem JSP, juntamente com a API Jena[3] e o pacote org.json[4].

A seguir, é apresentado um exemplo de utilização da ferramenta *QualityStamp*. Para isso, uma aplicação do domínio de "dados bibliográficos" está sendo usada como base para a avaliação dos *datasets* candidatos. Esta aplicação submete consultas utilizando a ontologia de referencia AKT[5] e busca por títulos de artigos a partir do nome do autor, o ano da publicação e o nome da conferência. Os requisitos da aplicação são identificados e, na ferramenta, seu nome e domínio são definidos. Além disso, os *datasets* candidatos são inseridos, como mostra a Figura 1. Os requisitos de dados da aplicação, representados pelas consultas *select* SPARQL que a aplicação pode executar, são então inseridos. Como exemplo, a Figura 2 apresenta duas destas consultas, porém, mais consultas podem ser inseridas, conforme a necessidade.

[2] http://qualitystamp.waltertravassos.cloudbees.net/
[3] https://jena.apache.org/
[4] http://www.json.org/java/
[5] http://www.aktors.org/ontology/portal#



Figura 1 – Definição dos dados da Aplicação e dos *datasets*

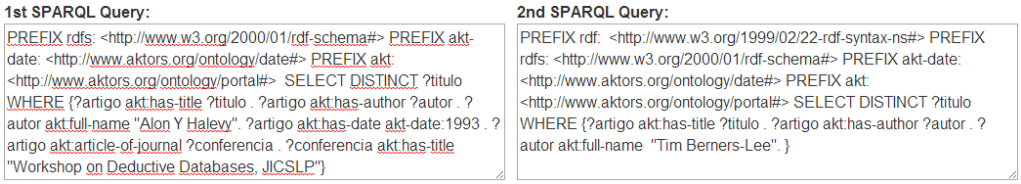


Figura 2 – Fragmento do conjunto de consultas da aplicação.

O ED define os requisitos não funcionais da aplicação, a partir da configuração dos critérios e métricas utilizados. Neste exemplo, os critérios são aqueles apresentados na Tabela 1. O ED então define os pesos que devem ser considerados na avaliação (Figura 3). Observa-se que critério completude recebe um peso maior devido à sua importância no tocante à capacidade do *endpoint* de responder um conjunto de consultas.

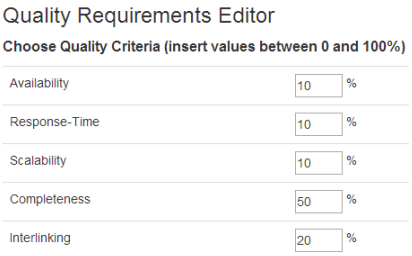


Figura 3 – Configuração dos Critérios de Qualidade.

Ao final da avaliação, é gerada a classificação ordenada dos candidatos, como mostra a Figura 4. Nesta figura, a medida geral de cada *dataset* é mostrada juntamente com sua classificação. O resultado mostra o DBLP como o *dataset* mais adequado aos requisitos da aplicação. Este resultado pode sofrer variação de acordo com os critérios e pesos considerados pelo ED. Alguns *datasets* podem possuir alta disponibilidade, alto desempenho, mas possuir uma baixa completude do dado. Essa discrepância acarretaria numa diminuição do número de resultados das consultas submetidas pela aplicação, e, consequentemente, diminuiria o escore do resultado final da avaliação da qualidade.

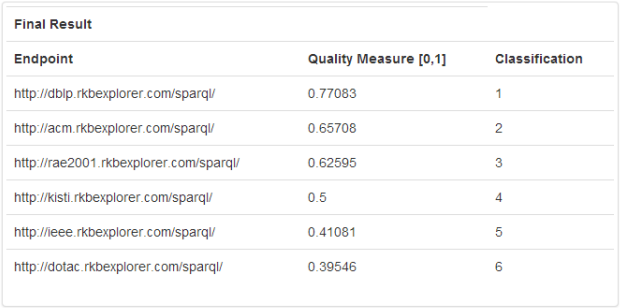


Figura 4 – Classificação dos endpoints candidatos.

4. Trabalhos Relacionados

Até o momento, nenhum trabalho com o objetivo de gerar uma medida de qualidade para um *linked dataset* segundo os requisitos de uma dada aplicação foi encontrado. Porém, alguns trabalhos serviram de base para compor a abordagem proposta. Zaveri *et al*., (2012) elencaram 26 métricas para avaliar *linked datasets*. Apenas as métricas objetivas elencadas por Zaveri *et al*., (2012) foram estudadas e implementadas com o objetivo de se adequar à abordagem *QualityStamp*. Buil-Aranda *et al*., (2013) fizeram um estudo para verificar se a estrutura da Web de Dados está madura para ser utilizada por aplicações. Sua metodologia de avaliação utilizou quatro critérios de qualidade: *discoverability*, *availability*, *interoperability*, *performance*. Neste trabalho, entretanto, não existe uma avaliação dos *linked datasets* utilizando todos os critérios propostos com o objetivo de gerar uma medida de qualidade semelhante à ferramenta *QualityStamp*. O *Sieve* (Mendes *et al*,. 2012), é um *framework* flexível que faz a avaliação da qualidade por meio de indicadores de qualidade (*timecloseness*, *preference*, *setMembership*, *Threshold* e *IntervalMembership)*. O *Sieve* foi criado como um módulo de qualidade intermediário para a integração de dados, ele não gera uma medida de avaliação da qualidade como a abordagem *QualityStamp*.

5. Considerações e Trabalhos Futuros

A Web de Dados continua crescendo e, a todo instante, novos *linked datasets* são disponibilizados enquanto outros se tornam indisponíveis. Neste cenário, verificar manualmente se um *dataset* atende aos requisitos de uma aplicação se torna uma tarefa trabalhosa e demorada. A *QualityStamp* se torna então uma opção automatizada para esta avaliação. Para tal, ela provê a avaliação da qualidade de *linked datasets* candidatos para aplicações de domínio específico. Atualmente estão sendo realizados experimentos para avaliação da abordagem e da ferramenta.

Referências

Buil-Aranda C., Hogan A., Umbrich J., Vandenbussche P. (2013) "SPARQL Web-Querying Infrastructure: Ready for Action?", In: 12th International Semantic Web Conference (ISWC 2013). Sydney, Australia, Outubro de 2013.

Heath T., Bizer C. (2011) "Linked Data: Evolving the Web into a Global Data Space" (1st edition). Synthesis Lectures on the Semantic Web: Theory and Technology, 1:1, 1-136. Morgan & Claypool.

Lóscio, B. F.; Batista, M. C. M.; Souza, D.; Salgado, A. C. (2012) "Using Information Quality for the Identification of Relevant Web Data Sources: A Proposal", In: iiWAS 2012.

Mendes, P., Mühleisen, H., and Bizer, C. (2012) "Sieve: Linked Data Quality Assessment and Fusion", In: Proceedings of LWDM (Março 2012).

Naumann F. (1998) "Data Fusion and Data Quality", In: New Techniques and Technologies for Statistics Seminar (NTTS'98). Sorrent, Italy, 1998.

Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., Auear, A. (2012) "Quality Assessment Methodologies for Linked Open Data", In: IOS Press 2012.

paper:6

SciCumulus 2.0: Um Sistema de Gerência de *Workflows* Científicos para Nuvens Orientado a Fluxo de Dados *

Vítor Silva[1], Daniel de Oliveira[2] e Marta Mattoso[1]

[1]COPPE – Universidade Federal do Rio de Janeiro (UFRJ)
[2]Instituto de Computação – Universidade Federal Fluminense (IC/UFF)
{silva, marta}@cos.ufrj.br, danielcmo@ic.uff.br

Resumo. *Ao contrário dos workflows de negócio, os workflows científicos são centrados no grande fluxo de transformação de dados. Entretanto, as abordagens dos sistemas de workflows em larga escala ainda são voltadas à gerência da execução paralela de tarefas ao invés de gerenciar as relações entre os dados ao longo fluxo de geração dos dados do workflow. Este artigo apresenta a execução do SciCumulus 2.0 e sua nova camada de submissão de execução paralela que oferece diferentes níveis para a modelagem de workflows, assim como a configuração do ambiente de paralelismo e consultas aos dados de domínio e de proveniência em tempo de execução.*

1. Introdução

Nos últimos anos, os experimentos científicos computacionais têm aumentado em complexidade em termos do número de execuções e do volume de dados produzidos. Esses experimentos podem ser modelados como um conjunto de atividades e um fluxo de dados entre elas. Cada atividade pode ser um programa de computador que processa um conjunto de dados de entrada e produz um outro conjunto de dados de saída, gerando o fluxo de dados dos chamados *workflows* científicos. Sistemas de Gerência de *Workflows* Científicos (SGWfC), como o Swift/T (Wozniak *et al*. 2013), executam workflows com processamento de alto desempenho (PAD), incluindo os ambientes de nuvens de computadores. Para prover paralelismo, as atividades do *workflow* são subdivididas em tarefas menores (que chamamos de ativações). Cada ativação atua sobre um elemento do conjunto de valores de parâmetros a ser consumido. Os SGWfC paralelos distribuem as ativações para os recursos computacionais nesses ambientes, respeitando as dependências de dados entre elas. A gerência da dependência do fluxo de dados do *workflow* é um dos diferenciais dos SGWfC em relação a soluções que programam esse controle por *scripts* ou Hadoop, ou de forma independente (manual).

Ao contrário dos *workflows* de negócio, os *workflows* científicos são centrados no fluxo de transformação dos dados (Davidson and Freire 2008). Entretanto, as abordagens dos SGWfC ainda são fortemente voltadas à gerência das ativações ao invés do fluxo de dados como um todo. Pode-se classificar os dados envolvidos na execução paralela de *workflows* científicos em quatro categorias: (i) dados do domínio, (ii) fluxo de dados gerados na execução do *workflow*, (iii) fluxo das atividades executadas e (iv) dados de desempenho da execução do *workflow*. Essas quatro categorias, embora relacionadas, são tratadas de modo independente pelos SGWfC. Apenas as categorias

* Este artigo foi parcialmente financiado pelo CNPq e FAPERJ.
Vídeo em: https://s3.amazonaws.com/SBBD-Demo/Video-Final.mp4.

(ii) e (iii) são interligadas pelos SGWfC por meio de dados de proveniência. SGWfC registram o fluxo de dados e atividades, capturando a proveniência prospectiva (estrutura e dependências das atividades) e retrospectiva (relacionados à execução do *workflow*) (Davidson and Freire 2008). Após a execução do *workflow*, cientistas podem consultar a base de dados de proveniência para analisar o fluxo dos dados até os resultados. Embora esse registro seja um grande avanço em relação às soluções que não utilizam proveniência, a análise de *workflows* com dados em larga escala necessita de um monitoramento do perfil computacional da geração do fluxo de dados (*e.g.* tempo de execução das atividades). É preciso ainda relacionar esse fluxo aos dados do domínio. Em dados de proveniência que contêm apenas metadados sobre os dados do domínio, o relacionamento aos dados de domínio ainda se dá de forma "manual". Dados do perfil computacional, em geral, são representados em *logs*. Entretanto, *logs* são difíceis de serem consultados e ferramentas do ambiente PAD são independentes do contexto do fluxo de transformação de dados, limitando o poder analítico.

Para realizar a integração das quatro categorias de dados, Ogasawara *et al*. (2011) propuseram uma abordagem algébrica, centrada em dados, para a execução de *workflows*. Essa álgebra faz uso da tecnologia de sistemas de bancos de dados relacionais para gerenciar o fluxo de transformação de dados em *workflows* de forma integrada, sendo tanto a definição do *workflow* quanto seus dados representados em relações dentro do banco de dados de proveniência. À medida que as ativações são executadas, esta base de dados de proveniência registra o perfil do desempenho das execuções em relações voltadas para apoiar o monitoramento via consultas. Da mesma forma, tuplas representam, como valores ou referência a arquivos de dados do domínio, os dados de entrada das atividades. Assim, a abordagem algébrica integra em uma base de dados de proveniência, as quatro categorias de dados do experimento científico, gerando uma ferramenta essencial na análise de dados científicos em larga escala.

Com o intuito de desenvolver um SGWfC que tirasse proveito dos ambientes de nuvens e dessa integração no banco de dados de proveniência, a abordagem algébrica foi incorporada à máquina de *workflows* que deu origem ao SciCumulus (Oliveira *et al*. 2010). A proveniência foi usada, de modo original, para que uma série de componentes oferecessem: monitoramento, tolerância a falhas, re-execuções, etc. (Costa *et al*. 2012). Além disso, o SciCumulus usou a proveniência para explorar a elasticidade na computação em nuvens, para a execução paralela de *workflows*, implementando um algoritmo adaptativo para a alocação dinâmica de recursos computacionais. Ao consultar o histórico via proveniência ele minimiza o custo financeiro, o tempo de execução e a confiabilidade dos recursos computacionais (Viana *et al*. 2011, Oliveira *et al*. 2012). Desde a sua primeira versão, o SciCumulus ainda não foi apresentado como ferramenta de demonstração. O SciCumulus 2.0 integra, em uma única ferramenta, diversos componentes anteriormente propostos de modo isolado, e cria uma camada de software para a modelagem de experimentos sem a necessidade de uma interação direta com o ambiente de PAD. Tal camada oferece diferentes níveis de abstração para a modelagem dos *workflows*, assim como a configuração do ambiente de PAD e consultas analíticas aos dados de proveniência em tempo de execução.

Embora seja grande o número de SGWfC com paralelismo (Bux e Leser 2013), nenhum deles oferece um acesso integrado às quatro categorias de dados envolvidas na execução e análise do *workflow* científico. Os sistemas mais relacionados a esse trabalho são o Swift/T e o Pegasus, devido ao paralelismo. Entretanto, nesses sistemas,

os dados de proveniência são disponibilizados apenas no final da execução do *workflow*. Para monitorar o desempenho durante a execução, o Pegasus disponibiliza uma base de dados relacional para consultas sobre o perfil de execução. Assim, o Pegasus opera com duas bases, uma para registrar as informações de execução das atividades e outra para os dados de proveniência do fluxo de dados e atividades do *workflow* executado. Além da redundância de dados, essa separação em duas bases, não permite consultas do tipo "quais os valores dos parâmetros das ativações com tempo de execução acima da média". Embora essa consulta seja possível na base de proveniência do Swift, ela só pode ser submetida quando a execução já terminou e muitas vezes sem acesso aos dados de domínio, como o parâmetro mencionado.

O restante deste artigo é organizado da seguinte forma: a Seção 2 apresenta o SciCumulus 2.0, incluindo a sua arquitetura com os mecanismos de interação e integração de dados via proveniência, enquanto que a Seção 3 conclui este artigo.

2. SciCumulus 2.0: um SGWfC Orientado a Fluxo de Dados

Esta seção apresenta uma visão geral do SciCumulus 2.0 e seus recursos de gerência de fluxos de dados guiada por dados de proveniência. Na álgebra de Ogasawara *et al*. (2011) cada atividade do *workflow* é associada a uma operação. Essas operações determinam como a atividade consome e produz dados. O conjunto de dados de entrada e saída das atividades são operandos, representados como relações. Cada ativação consome uma tupla da relação operando que contém parâmetros e referências a arquivos necessários para realizar o processamento da atividade (ativação) gerando uma tupla correspondente na relação de saída. Assim, o banco de dados de proveniência (Figura 1) se torna uma peça fundamental não só para a análise do experimento por parte do usuário, mas para a própria execução por parte do SGWfC, uma vez que atua como um catálogo de estatísticas que pode ser consultado e analisado.

A arquitetura do SciCumulus 2.0 é definida a partir de três camadas principais (Figura 1): *SciCumulus Starter*, *SciCumulus Core* e o Banco de Dados de Proveniência. O *SciCumulus Starter* configura o ambiente para a execução paralela do *workflow* e invoca a sua execução na nuvem. É implementado por uma aplicação independente que executa na máquina do cientista sem que seja necessário instalar nenhuma aplicação ou biblioteca (apenas copiar um *.jar*). Ele possui três componentes principais com as funções de: gerência do ambiente de nuvem, submissão de *workflows* e a validação do *workflow*. A gerência do ambiente de nuvem instancia e destrói um conjunto de máquinas virtuais (MVs) interconectadas na nuvem (*i.e*. *cluster* virtual). Ele inicializa ou exclui um conjunto de MVs, cria configurações de segurança necessárias (como grupos de segurança, chaves de acesso, entre outros), assim como define uma hierarquia dessas MVs, a fim de obedecer a arquitetura do *SciCumulus Core* (Oliveira *et al*. 2012) que é baseada em MPI e exige uma MV de controle para a distribuição das ativações. Essa instanciação inicial de MVs é baseada em consultas aos dados históricos de proveniência para que o *SciCumulus Starter* possa dimensionar o ambiente para a execução. O *SciCumulus Core* pode ainda instanciar novas MVs ou destruir, em tempo de execução de acordo com a demanda seguindo o algoritmo adaptativo proposto por Oliveira *et al*. (2012). Vale ressaltar que enquanto o SciCumulus original executava o controle do algoritmo adaptativo em uma máquina remota (que deveria possuir conexão permanente obrigatória), o SciCumulus 2.0 submete um pedido de execução para as

diversas MVs, dedicando uma das MVs de execução para o controle do algoritmo adaptativo (*i.e.* quando dimensionar o ambiente).

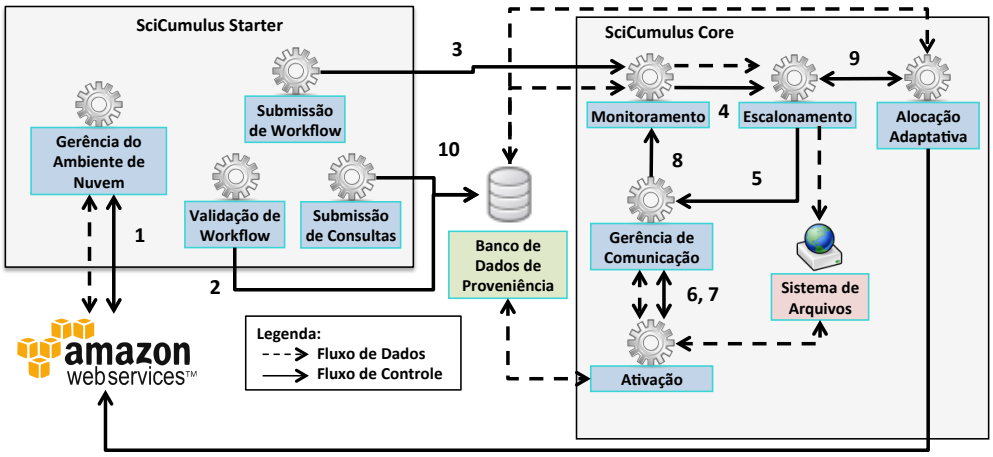


Figura 1. A Arquitetura do SciCumulus 2.0

A validação de *workflows* registra a especificação conceitual de *workflows* na base de dados de proveniência. Este componente valida um arquivo XML que contém a especificação do *workflow* e então registra no banco de proveniência a estrutura do *workflow* para ser usada no momento da execução. A submissão do *workflow* inicia a execução paralela de um *workflow* que já foi registrado na base de proveniência. A inserção do *SciCumulus Starter* na arquitetura do SciCumulus 2.0 é uma mudança importante em relação às primeiras versões do SciCumulus, cuja inicialização era realizada a partir de SGWfC de terceiros (*e.g.* VisTrails). Utilizando o *SciCumulus Starter*, a execução do *workflow* depende apenas do SciCumulus 2.0. O *SciCumulus Starter* é invocado por: java –jar SciCumulusStarter.jar <funcionalidade> <arquivo XML>. Os valores para o argumento de funcionalidade incluem: *-cc:* cria cluster de MVs na nuvem, *-dc:* exclui cluster de MVs; *-icw*: insere *workflow* na base de proveniência; *-ucw*: atualiza *workflow* na base de proveniência; *-dcw*: exclui *workflow* da base de proveniência; *-sew*: submete *workflow* para execução paralela e *-q*: consulta a base de proveniência. Além do argumento de funcionalidade, deve ser informado um arquivo de configuração, especificado de acordo com um XML *schema* próprio do SciCumulus. Cada *tag* do XML a ser informada se encontra no Quadro 1. Os elementos a serem informados no arquivo de configuração variam de acordo com a funcionalidade. Os termos estão em inglês para facilitar a interoperabilidade com a recomendação de padrão de proveniência PROV da W3C.

SciCumulus Starter ainda permite ao usuário executar consultas no banco de proveniência em tempo de execução por meio do componente de submissão de consultas. Além dos dados padrão de proveniência, tanto a consulta aos dados do domínio como do progresso da execução dos *workflows* podem ser submetidas via SQL especificado pelo usuário ou pré-configuradas no componente.

O *SciCumulus Starter* invoca o *SciCumulus Core* por meio do seu componente de submissão de *workflows*. O *SciCumulus Core* é uma aplicação paralela implementada em MPJ (MPI para linguagem Java) e que é executada em todas as MVs que fazem parte do *cluster* virtual. O *SciCumulus Core* gerencia a execução de *workflows* científicos, conforme a arquitetura apresentada na Figura 1. Quando um *workflow* é submetido ao *SciCumulus Core*, a primeira etapa (*#3* na Figura 1) consiste

na análise e envio dos dados previamente registrados na base de proveniência para esse *workflow* na nuvem. Já a segunda etapa (*#4*) consiste no envio dos dados de proveniência e a requisição, respectivamente, do início da execução do *workflow*. O *Monitoramento* analisa os dados de proveniência para verificar as atividades pendentes e que apresentam todas as suas dependências atendidas. Para essas atividades, o *Escalonamento* identifica as ativações que fazem parte dessa atividade (*#5*). Ao mesmo tempo, possíveis adaptações na quantidade de MVs são analisadas em tempo real pela *Alocação Adaptativa* (*#9*). Em caso de adaptações necessárias para atender a requisitos dos cientistas como tempo máximo de execução, são realizadas mudanças por esse componente, reportando ao *Escalonador* apenas essas mudanças de MVs.

Quadro 1. Elementos do arquivo de configuração no formato XML

Elementos	Descrição	Atributos	Dependência
SciCumulus	Elemento raiz	Nenhum	Nenhuma
credentials	Credenciais para a Amazon	access_key, secret_access_key	SciCumulus
environment	Configurações do ambiente	type, cluster_name	SciCumulus
binary	Informações de arquivos binários	directory, conceptual_version, execution_version	SciCumulus
machine	Configuração das máquinas virtuais	image, user, password	SciCumulus
vm	Tipos de máquinas virtuais disponíveis	Type, financial_cost, disk_space, ram, gflops, platform, cores	machine
constraint	Restrições do ambiente e do workflow	workflow_exectag, max_time, max_financial_cost, max_vm_amount, total_ram, total_disk, alfa1, alfa2, alfa3, cores	SciCumulus
workspace	Configurações para armazenamento de arquivos	upload, bucket_name, workflow_dir, compressed_workspace, compressed_dir	SciCumulus
database	Configurações da base de dados	name, username, password, port, server, path	SciCumulus
query	Especificação da consulta	sql	SciCumulus
conceptualWorkflow	Especificação do *workflow* conceitual	tag, description	SciCumulus
activity	Especificação das atividades	tag, descript, type, activation, template, extractor	conceptualWorkflow
relation	Especificação das relações	reltype, name	activity
field	Especificação dos campos	name, type, input, output, decimalplaces, operation	relation
executionWorkflow	Especificação do *workflow* de execução	tag, execmodel, expdir, max_failure, user_interaction, redundancy, reliability	SciCumulus
relation	Especificação das relações de entrada do *workflow* de execução	name, filename	executionWorkflow

O *Monitoramento* provê a tolerância a falhas no SciCumulus 2.0. Essa extensão foi inspirada no SciMultaneous (Costa *et al*. 2012) que verifica a ocorrência de erros por meio de consultas ao banco de proveniência. Três características importantes desse mecanismo são: a replicação de ativações em execução, quando um recurso computacional está ocioso e não possui ativações pendentes; o cálculo de confiabilidade de uma determinada MV em virtude do número de ativações com falhas de execução; e o número máximo de falhas permitido em uma ativação, para que a mesma seja desconsiderada da execução do *workflow*. Diferentemente do SciMultaneous, o SciCumulus 2.0 provê a tolerância a falhas em tempo de execução dentro da própria gerência de ativações (elemento *Ativação* da Figura 1). O SciCumulus 2.0 não necessita de uma MV extra para a recuperação de falhas, como no SciMultaneous.

Após a determinação da MV que deve executar (ou re-executar, caso uma falha seja identificada) uma ou mais ativações, o *Escalonamento* envia tal mapeamento (*i.e*. conjunto de MVs e ativações) para a *Gerência de Comunicação*, que envia (*#5*) e recebe (*#6*) mensagens (com dados e valores de parâmetros de cada ativação) entre as MVs. Ao receber uma ou mais ativações, a MV deve executá-la (s) pelo componente de *Ativações* (#6 e #7). Quando a MV central recebe ativações finalizadas pelas MVs executoras, os dados de proveniência dessas ativações são enviados ao *Monitoramento*, que assume o

controle e identifica as atividades pendentes e distribui novas ativações para as MVs que estiverem ociosas.

3. Conclusão

Experimentos científicos, mesmo se executados em paralelo, podem ter uma longa duração necessitando de monitoramento. Várias tentativas de configurações são necessárias para se chegar a uma conclusão final. SciCumulus (Oliveira *et al*. 2012), uma máquina de *workflows* capaz de gerenciar a execução de *workflows* em ambientes de nuvem, foi estendido e várias ferramentas desenvolvidas externamente foram incorporadas, de modo a facilitar sua instalação e execução, gerando o SciCumulus2.0. Uma característica que se mantém original em relação ao estado da arte é ser orientado ao fluxo de transformação de dados do *workflow*. Essa é uma característica importante, pois o banco de dados de proveniência serve de apoio à interface com a máquina de *workflows*, tanto para a configuração quanto para o monitoramento e análise de dados em larga escala por parte do cientista. Além disso, o SGWfC passa a ter acesso a um catálogo de estatísticas auxiliando a geração do plano de execução e estratégias adaptativas em tempo de execução. A demonstração objetiva a apresentação do SciCumulus 2.0, suas contribuições originais e evoluções.

Referências Bibliográficas

Bux, M., Leser, U., (2013), *Parallelization in Scientific Workflow Management Systems*, CoRR/arXiv:1303.7195.

Costa, F., Oliveira, D. de, Ocana, K., Ogasawara, E., Dias, J., Mattoso, M., (2012), "Handling Failures in Parallel Scientific *Workflows* Using Clouds". In: *WORKS workshop IEEE/ACM SC Companion*, p. 129–139.

Davidson, S. B., Freire, J., (2008), "Provenance and scientific *workflows*: challenges and opportunities". In: *ACM SIGMOD*, p. 1345–1350.

Lee, K., Paton, N. W., Sakellariou, R., Deelman, E., Fernandes, A. A. A., Mehta, G., (2008), "Adaptive *Workflow* Processing and Execution in Pegasus". In: *3rd International Conference on Grid and Pervasive Computing*, p. 99–106.

Ogasawara, E., Dias, J., Oliveira, D., Porto, F., Valduriez, P., Mattoso, M., (2011), "An Algebraic Approach for Data-Centric Scientific *Workflows*", In: *Proc. of VLDB Endowment*, v. 4, n. 12, p. 1328–1339.

Oliveira, D., Ogasawara, E., Baião, F., Mattoso, M., (2010), "SciCumulus: A Lightweight Cloud Middleware to Explore Many Task Computing Paradigm in Scientific Workflows". In: *3rd Int Conference on Cloud Computing*, p. 378–385.

Oliveira, D., Ocaña, K., Baião, F., Mattoso, M., (2012), "A Provenance-based Adaptive Scheduling Heuristic for Parallel Scientific *Workflows* in Clouds", In: *Journal of Grid Computing*, v. 10, n. 3, p. 521–552.

Wozniak, J. M., Armstrong, T. G., Wilde, M., Katz, D. S., Lusk, E., Foster, I. T., (2013), "Swift/T: Large-Scale Application Composition via Distributed-Memory Dataflow Processing". In: IEEE/ACM CCGrid, p. 95–102

paper:10

SHRuB - Searching through Heuristics for the Better Query-Execution Plan *†

Marcos Vinicius Naves Bedo[1], **Luiz Olmes-Carvalho**[1], **Gabriel Vicente Pierro**[1] **and Caetano Traina Jr.**[1]

[1]Department of Computer Science – University of São Paulo (USP)
P.O.Box 668 – CEP 13.560-970 – São Carlos, SP, Brazil

{bedo, olmes, gvpierro, caetano}@icmc.usp.br

Abstract. *An important aspect to be considered for systems aiming at integrating similarity-queries into RDBMS is how to represent and optimize query-plans that involve traditional and complex predicates. Toward developing facilities for such integration, we developed a technique to extract a canonical query-plan command tree from an similarity-extended SQL expression. The SHRuB tool, presented in this paper, is able to interactively represent a query parse-tree. We developed a catalog model which allows estimating the execution cost as well as provides hints for optimizing the query-plan by adopting a three stage heuristic. Through a case study and initial experiments, we have demonstrated that the tool is able to find a local-minimum query-execution plan. Moreover, SHRuB can be plugged on existing frameworks that support similarity queries or employed as a course-ware aid for database teaching.*

1. Introduction

Integrating similarity-queries into Relational Database Management Systems (RDBMS) is quite a challenge, mainly because the same resources available for scalar data (e.g., numbers, dates and characters strings) should be equally provided for complex data. Most of current RDBMS are able to provide query-plan cost estimation only for queries involving traditional data. Such estimation relies on selectivity measures, which are obtained by statistics collected by the RDBMS. Considering queries on complex data, the RDBMS-architecture should also provide an optimal query execution plan using: 1) Algebraic properties rules of complex predicates (rule-based optimization) and/or 2) Selectivity based on data distribution heuristics (cost-based optimization).

The first challenge to obtain optimal query-execution plans over complex data is defining a representation for query operators such as selection and join based on similarity predicates, in such a way that their execution order in a query can be evaluated. This problem has been studied by several authors [Kaster et al. 2010, Silva et al. 2010]. An initiative of special mention is [Ferreira et al. 2011], which proposes to employ similarity-based operations as extensions of the traditional predicates, leading to a set of algebraic properties that allows for order-execution manipulation subjected to the established rules. Once a definition on how to combine complex and traditional predicates is reached, another challenge emerges: *How to collect statistic measurements from complex and traditional data in order to estimate selectivity?*

*This research was supported by FAPESP, CNPq and CAPES.

†Screencast link: https://dl.dropboxusercontent.com/u/38049707/shrub.mp4

This paper presents the tool called SHRuB (Searching through Heuristics for the Better query-execution plan), which is being employed as a module of the SIREN *framework* [Barioni et al. 2006]. SHRuB enables that queries formulated in extended-SQL containing traditional and complex data can be visually represented by the parse-tree representation of a canonical query plan. It receives a SQL query command and generates: 1) a syntactic/semantic analysis; 2) the canonical query tree and the attributes and conditions table (ACT); and 3) the estimated cost of each node execution. The main advantages of SHRuB are that it relies on the robust and improved SIREN back-end to solve the extended-SQL, providing a tree optimization through a well-defined heuristic. Other desirable interface properties were also considered, such as the ability for on the fly tree construction and allowing the user to directly manipulate the query tree nodes. We highlight here the great educational potential of the tool for database learning. Considering this, the tool allow also the visualization of hypothetical queries plans, where the user can manipulate the node execution costs. This paper describes the implementation details of SHRuB and one example application, where the system is able to find an (sub) optimal query-execution plan. SHRuB is an initiative that contributes to the RDBMS' capability to handle query plans containing complex and traditional predicates.

2. Related Work

The predicates most employed in similarity search are the similarity range (Rng) and the $k-$Nearest Neighbors (k-NN). The former is able to select all tuples whose value in attribute S from relation T are not farther than a given threshold ξ from the query center s_q, considering a distance function d. Thus, the result of a range query is composed by $\sigma_{(\mathsf{S}\ Rng(d,\xi)\ s_q)}\mathsf{T} = \{t_i \mid d(t_i(\mathsf{S}), s_q) \leq \xi\}$. The later selects the tuples whose attribute S value is one of the k elements most similar to the query center s_q. The result of a k-nearest neighbor $\ddot{\sigma}_{(\mathsf{S}\ k\text{-}NN(d,k)\ s_q)}\mathsf{T}$ is composed by the tuples $t_i \in \mathsf{T}$ such that $t_i(\mathsf{S})$ is one of the k elements in the active domain S of S nearest to s_q. There are some prototype systems aiming at integrating the similarity paradigm into RDBMS, such as the FMI-SiR [Kaster et al. 2010] and POSTGRESQL-IE [Guliato et al. 2009]. However, no one of those systems provide a query optimization module. For instance, FMI-SiR allows the user to specify an index to be used in the query, but there is no clear query-optimization approach available. For predicates involving the traditional comparisons ($>, <, \geq, \leq, =$ and $\neq$), it is expected that the RDBMS generate a canonical query-plan which is further rewritten according to the defined heuristics.

Therefore, a query plan that addresses the problem of representing predicates based on Rng and k-NN that allow for optimization tasks should also be generated by systems aiming at supporting similarity queries. A further difficulty is that, unlike Rng, the k-NN predicate does not hold the commutative property, which leads to studies on how the relational algebra should properly handle it. In fact, a k-NN is one of the most computationally expensive predicates to execute. Thus, it may be desirable the query evaluator engine could benefit from techniques such as Metric Access Method (MAM) in order to speed up the k-NN processing. For this purpose, once k-NN predicates do not preserve the commutativity, they must be evaluated before the others [Traina Jr. et al. 2002]. Moreover, due to the fact the non-similarity-based predicates still remain, the final result may have fewer tuples than k. Performing the k-NN comparisons as the last operation produces the requested number of tuples – if enough tuples exists in the original rela-

tion. The work of [Ferreira et al. 2011] demonstrated that such execution has little or no impact in the overall query performance, whenever the query-plan optimizer is able to apply the rules of an extended-relational algebra. By adopting this extended-relational algebra, there are several permutations that can be applied over a parse-tree considering the costs of each node. RDBMS usually stores histograms to estimate selectivity and access plan costs. Obtaining a selectivity estimation and combining the operators through extended-relational algebra allow generating a good query execution plan.

Generically, a histogram H on attribute A is constructed by partitioning the data distribution of A in the relation T into β ($\beta \in \mathbb{N}^*$) disjoint subsets called *buckets* and approximating its number of occurrences by a measure of frequency values for each bucket. The definition of values in each bucket should follow a partition rule which approximates T. According to [Ioannidis 2003], every type of histogram can be characterized by the following properties: 1) Partition Class - The class of histograms that are represented by the rule, 2) Partition Constraint - The mathematical constraint that links a histogram to a partition class and 3) Sort and source parameter - Which represents derived and sorted parameters of value and frequency [Poosala et al. 1999]. The histograms most widely adopted by RDBMS are the *equi-depth* and the *equi-width* histograms. The last ones group continuous ranges of attributes into buckets, where each discrete value has accumulate frequency $1/\beta$. The buckets are equally partitioned and sorted by value, where the area refers to the cumulative probability distribution.

3. The Architecture of the SHRuB Application

SHRuB is a tool able to illustrate query-plans when performing similarity-queries using an extended-SQL language. It takes advantage of the SIREN back-end to interpret and validate user inputs. Currently, SHRuB is able to construct a canonical parse-tree providing also the attributes and conditions table (ACT) and the estimated selectivity for each tree node. Figure 1 illustrates the general SHRuB architecture. As the first step, SHRuB perform syntactical analysis of the users' input, validating it through the SIREN back-end (Step 2). This step requires accessing the database's catalog tables (Step 3). Once the expression is validated, SHRuB constructs the canonical query-plan (Step 4). For each node included in the query tree, if corresponding statistics are available in the catalog, SHRuB retrieves them (Step 5) and constructs an optimized query-plan considering the extended-relational algebra (Step 6). Finally, the optimized result is displayed to the user (Step 7) and sent to execution.

SHRuB employs a materialized query-plan analysis (i.e. each node has assigned a cost execution) by taking advantage of the extended-relational algebra rules to recombine nodes. In the execution phase, however, the RDBMS will employ pipelined operators. We propose here employing the *equi-width* histograms to estimate the selectivity considering non-indexed traditional attributes (i.e. numerical and character strings). These histograms are stored in SHRuB as Binary Large Objects (`BLOB`) in a catalog table. For complex attributes, we propose assuming the uniform distribution over the solution space, always evaluating k-NN as the last predicate and, therefore, not using indexes. The adopted heuristic is executed in three stages: (1) Perform the selections as earlier as possible (2) Perform the joins containing the highest selectivity predicates as earlier as possible, and (3) Perform the joins and selections involving two related tables as earlier as possible resulting in at most six distinct possibles query-plans, where a cheaper one to execute is

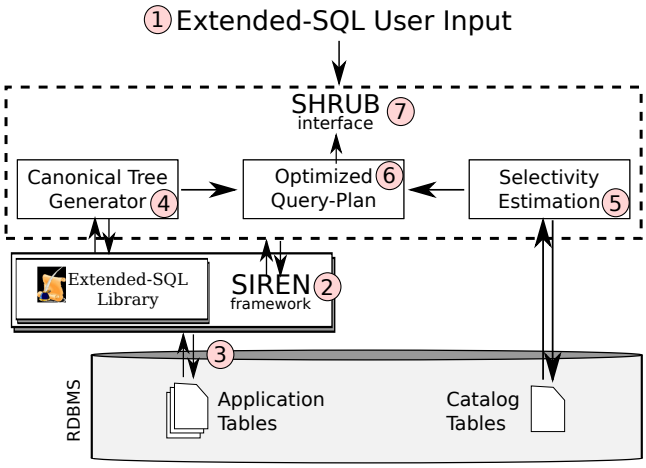


Figure 1. Overview of SHRuB architecture.

chosen. SHRuB is also able to simulate query-plans disregarding SIREN. In this case, steps 2 and 3 of the Figure 1 are ignored and a parse-tree for the SQL expression is immediately built. Thus, the user should manually provide the selectivity values for the selectivity estimation. This resource allow evaluating index creation and constraint definition, besides being an interesting teaching tool for database disciplines.

4. The SHRuB Interface - A case study

The SHRuB interface was coded in C++ using the cross-platform Qt 5.2[1] to provide portability. Consider that a database *schema* was already defined and populated using SIREN, including complex data such as images or audio, defined as attributes of the relations. The user may now query both traditional and complex data, using SIREN's extended-SQL language, which include the operators $>, <, \geq, \leq, =, \neq, Rng$ and k-NN. It is important that the search engine be capable to execute the query the faster the possible, even taking into account the time required to choose an optimal query-plan. Thus, we employed a three-stage heuristic, considering the extended-relational algebra.

Figure 2 illustrates the system interface. The user starts logging into SIREN to express queries through extended-SQL (1). Then, it is possible to ask for the system to construct a canonical parse-tree (2). The tree visualization (3) allows the user to interactively change the nodes in order to rearrange the query-plan. The attributes and conditions table (4) describes each node regarding its involved attributes and operators. Inter actor (5) provides a way to perform selectivity estimation for each node in the tree visualized. The results generated by the parse-tree can be visualized on the Results tab (6). SHRuB can generate a new parse-tree, optimized following its internal heuristics (7). The optimized tree replaces the one in the visualization tool (3).

For our case study, we used the `CANVAS` dataset, which is composed by 1348 public domain canvases of historic world painters available in Wikimedia Commons[2]. All data were inserted into normalized relations, allowing further queries-plans visualization combining traditional and complex predicates. Three main tables were created: Painter

[1]http://qt-project.org/

[2]http://commons.wikimedia.org/wiki/Category:Paintings_by_painter

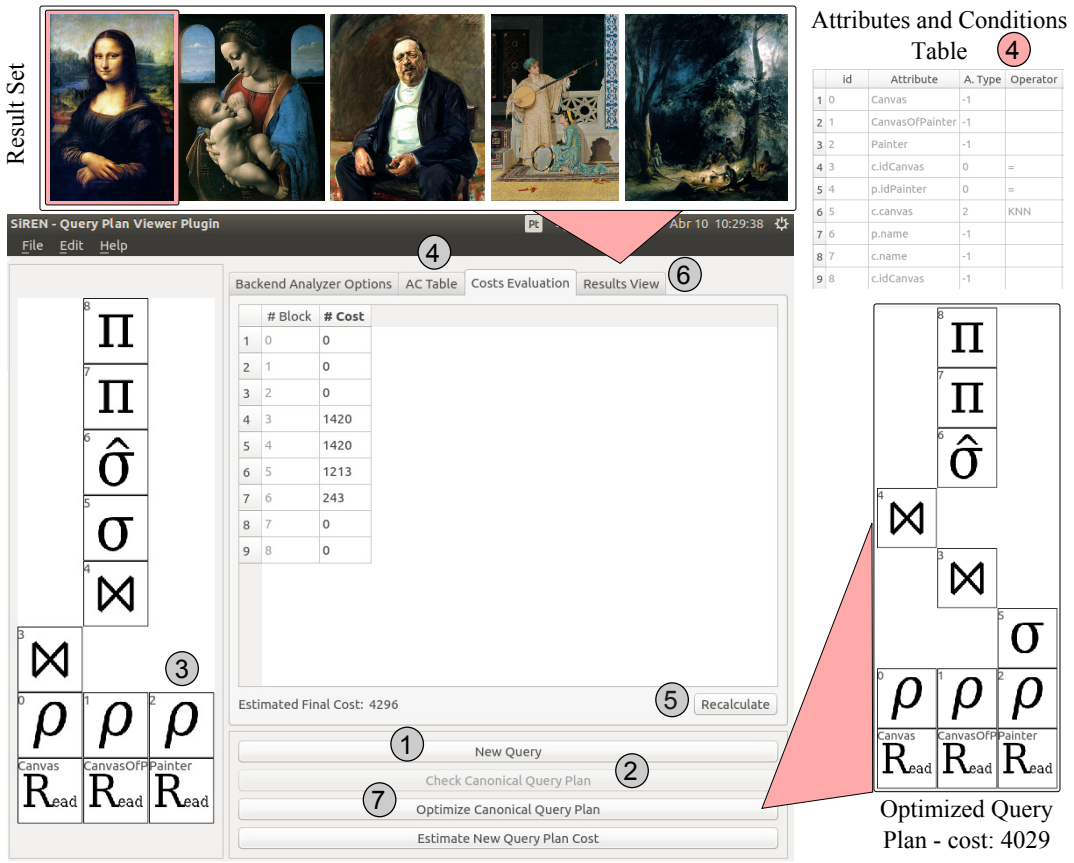


Figure 2. SHRuB interface. Pop-outs are displayed through mouse interactions.

(containing the painter information, such as name and nationality), Canvas (containing the canvas itself and details such as name and year), and CanvasOfPainter (since a canvas can be painted by two or more painters). We employed the color histogram extractor and the Euclidean distance function to compare canvases. A query example is: *"Retrieve the five paintings most similar to Da Vinci's 'Monalisa', considering only painters of the medieval and modern age."*. It is expressed in SIREN extended-SQL language as:

```
SELECT p.Name, c.Name, c.idCanvas FROM Canvas c JOIN CanvasOfPainter cv
ON c.idCanvas=cv.idCanvas JOIN Painter p ON p.idPainter=cv.idPainter
WHERE c.Canvas NEAR 'monalisa.jpg' STOP AFTER 5 AND p.birth < 1880;
```

Figure 2 (3) illustrates the canonical parse-tree generated for this SQL command. The initial execution cost, considering the statistics gathered (number of tuples in each table, the *equi-width histogram* for the birth attribute and uniform distribution for the canvas attribute) was 4296 comparisons and 9 disk reads. After applying the three stage heuristics, several operations were rearranged (Figure 2 (7)). The comparison birth $<$ 1880 is performed first, followed by joining CanvasOfPainter and Painter and them Canvas. Thereafter k-NN is performed, followed by the final projection. The plan cost now required 4029 comparisons and 8 disk reads. We evaluated the SHRuB tool using 100 combinations of traditional and complex predicates over `CANVAS`. In 32% of cases, the SHRuB strategy generated a significantly better query-execution plan. In the remaining cases, the canonical query-plan was already the local-minimum or the alternative query-plans did not present significant improvement. The required time to generated a few com-

binations of alternative parse-trees was always insignificant, in the order of the computer's time meter resolution. SHRuB allows the evaluation of theoretical query-plans. To do so, the user does not need to connect to SIREN (performed by unchecking the check box in the main interface (Figure 2 (8)). In this scenario, the optimizer heuristic is not able to generate an optimized query-plan, once the statistics are unknown. The parse-tree is now constructed considering user supplied attribute statistics, so it is necessary to manually inform each node cost. In this context SHRuB becomes as a potential pedagogical tool for databases studies, considering that either extended-SQL or traditional expressions can be represented in the materialized parse-tree.

5. Conclusions and Future Work

Providing full support for similarity-queries in RDBMS requires representation, rewriting and cost estimation of the query-plan. Several prototypes such as SIREN, FMI-SiR and POSTGRESQL-IE aims at providing support for query execution over complex data, but no generic query optimization approach was available yet. In this paper, we propose SHRuB, a new tool able to obtain a canonical query plan from extended-SQL expressions. Additionally, SHRuB can be *plugged* into SIREN, allowing obtaining syntactic/semantic query analyzes. By using properties of the extended-relational algebra, SHRuB employs a three-rules heuristic to estimate a better query execution plan. SHRuB optimizes queries involving traditional and complex predicates, offering a graphic and didactic support for query-plan analysis. SIREN (and every existing similarity-enabled DBMS prototype, by the way) currently does not execute similarity joins, so SHRuB performance gain tend to be small (in the order of 10%). However, as similarity joins are exceedingly costly, this is a resource required to enable them on real applications. Having SHRuB available, future works include implementing similarity join operators in a similarity-enabled database such as SIREN, and evaluating SHRuB performance on join processing.

References

Barioni, M. C. N., Razente, H. L., Traina, A. J. M., and Traina Jr., C. (2006). Siren: A similarity retrieval engine for complex data. In *VLDB*, pages 1155–1158.

Ferreira, M. R. P., Santos, L. F. D., Traina, A. J. M., Dias, I., Chbeir, R., and Traina Jr., C. (2011). Algebraic properties to optimize knn queries. *JIDM*, 2(3):385–400.

Guliato, D., de Melo, E. V., Rangayyan, R. M., and Soares, R. C. (2009). Postgresql-ie: An image-handling extension for postgresql. *J. Digital Imaging*, 22(2):149–165.

Ioannidis, Y. (2003). The history of histograms (abridged). In *Proceedings of the 29th International Conference on Very Large Data Bases*, pages 19–30. VLDB Endowment.

Kaster, D. S., Bugatti, P. H., Traina, A. J. M., and Traina Jr., C. (2010). Fmi-sir: A flexible and efficient module for similarity searching on oracle database. *JIDM*, 1(2):229–244.

Poosala, V., Ganti, V., and Ioannidis, Y. E. (1999). Approximate query answering using histograms. *IEEE Data Eng. Bull.*, 22(4):5–14.

Silva, Y. N., Aly, A. M., Aref, W. G., and Larson, P.-Å. (2010). Simdb: a similarity-aware database system. In *SIGMOD Conference*, pages 1243–1246.

Traina Jr., C., Traina, A. J. M., Faloutsos, C., and Seeger, B. (2002). Fast indexing and visualization of metric data sets using slim-trees. *IEEE TKDE*, 14(2):244–260.

29th Brazilian Symposium on Databases

wtdbd

October 6th to 9th, 2014
Curitiba – PR – Brazil

WORKSHOP ON THESIS AND DISSERTATIONS IN DATABASES

PROCEEDINGS

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

Workshop on Thesis and Dissertations Chairs

João Eduardo Ferreira, IME-USP
Fabio Porto, LNCC

Editorial

The Workshop of Thesis and Master Dissertations in databases is a traditional event co-located with the Brazilian Symposium on Databases. In the 2014 edition, the event takes place in Curitiba, Paraná, gathering professors and graduate students from different Universities in Brazil to present and discuss their most recent database research results.

The WTDBD is an excellent opportunity to receive feedback on-going graduate work from experienced researchers. All submitted papers received, at least, three reviews. Additionally, during the Workshop, students of selected papers have the opportunity to present their work to at least two senior researchers and to receive technical comments, as much as experimenting the challenge of presenting to an external committee. In this edition we have fifteen accepted works (thirteen master degree and two Phd) from many different Universities in Brazil.

The 2014 WTDBD Workshop chairs would like to thank the students and advisors that submitted their work to the workshop. Similarly, we are very grateful to the reviewers and the group of researchers that will take part in the jury during the workshop. Finally, the WTDBD coordinators would like to thank the SBBD 2014 organization for the support and excellent collaboration in preparing this edition of the event. We wish the community an excellent workshop and success in their works.

Joao Eduardo Ferreira, IME-USP
Workshop on Thesis and Dissertations in Databases 2014 Chair

Fabio Porto, LNCC
Workshop on Thesis and Dissertations in Databases 2014 Chair

29th Brazilian Symposium on Databases

October 6th to 9th, 2014
Curitiba – PR – Brazil

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

SBBD Steering Commitee

Ana Carolina Salgado, UFPE
Cristina Ciferri, ICMC-USP (chair)
José Palazzo Moreira de Oliveira, UFRGS
Marco A. Casanova, PUC-Rio
Mirella M. Moro, UFMG
Vanessa Braganholo, UFF

SBBD 2014 Commitee

Steering Committee Chair
Cristina Ciferri, ICMC-USP

Local Organization Chairs
Carmem Hara, UFPR and Vidal Martins, PUC-PR

Program Committee Chairs
Mirella M. Moro, UFMG and Renata Galante, UFRGS

Demos and Applications Session Chairs
José Antônio Macedo, UFC and Flávio Sousa, UFC

Workshop on Thesis and Dissertations in Databases Chairs
João Eduardo Ferreira, IME-USP and Fabio Porto, LNCC

Lectures Chair
Bernadette Farias Lóscio, UFPE

Tutorials Chair
Vanessa Braganholo, UFF

Local Organization Committee

Carmem Satie Hara, UFPR (chair)
Cristina Verçosa Pérez Barrios de Souza, PUC-PR
Daniel S. Kaster, UEL (proceedings)
Marcos A. Carrero, UFPR
Nadia Puchalski Kozievitch, UTFPR
Vidal Martins, PUC-PR (chair)

Workshop on Thesis and Dissertations Program Committee

André Santanchè, UNICAMP
Carmem Satie Hara, UFPR
Daniel de Oliveira, UFF
Duncan Ruiz, PUC-RS
Javam Machado, UFC
José Fernando Rodrigues Júnior, ICMC-USP
Leandro Wives, UFRGS
Luciano V. de Araujo, EACH-USP
Marcio K. Oikawa, UFABC
Maria Cláudia Cavalcanti, IME
Maria Luiza Machado Campos, DCC/UFRJ
Mirella M. Moro, UFMG
Ronaldo dos Santos Mello, UFSC
Sandra de Amo, UFU
Valeria Times, UFPE
Vaninha Vieira, UFBA

Table of Contents (WTDBD)

A strategy to preserve data confidentiality in cloud storage services 257
Eliseu Castelo Branco Junior, Javam Machado, José Maria da Silva Monteiro Filho

A Tuning Approach Based on Evolutionary Algorithm and Data Sampling for Boosting Performance of MapReduce Programs 266
Tiago Rodrigo Kepe, Eduardo Cunha de Almeida

Arquitetura Híbrida de Integração entre Banco de Dados Relacional e de Grafos: Uma Abordagem Aplicada à Biodiversidade 274
Patrícia R. N. Cavoto, André Santanchè

Auto Escalonamento Proativo para Banco de Dados em Nuvem 281
Victor A. E. Farias, Flávio R. C. Sousa, Javam Machado

Complex Pattern Detection and Specification to Support Biodiversity Applications 288
Jacqueline Midlej do Espírito Santo, Claudia Bauzer Medeiros

Data quality assessment of very large database through visualization system .. 295
João Marcelo Borovina Josko, João Eduardo Ferreira

Declarative Model Discovery in Provenance Data for Aiding in Scientific Experiment Planning 304
Mateus Ferreira Silva, Fernanda Baião, Kate Revoredo

Descoberta de Domínio Conceitual de Páginas Web 311
Gleidson Antônio Cardoso da Silva, Carina Friedrich Dorneles

Desenvolvimento de um método de acesso métrico dinâmico eficiente baseado em pivôs adicionais 318
Paulo Henrique Oliveira, Daniel S. Kaster

Mineração de Opinião Multilíngue para Comentários sobre Produtos 325
Anderson U. Kauer, Viviane P. Moreira

PVFS-Store - Um repositório chave-valor com garantia de localidade 332
Ricardo Masashi Maeda

Searching Time Series via Semantic Annotations 339
Lucas Oliveira Batista, Claudia Bauzer Medeiros

Sintonia fina baseada em ontologia: o caso de visões materializadas 346
Rafael Pereira de Oliveira, Sérgio Lifschitz

Uma Abordagem de Classificação não Supervisionada de Carga de Trabalho MapReduce Utilizando Análise de Log 353
Ivan Luiz Picoli, Eduardo Cunha de Almeida

WTMatcher: Uma Abordagem para Determinação de Similaridade entre Tabelas na Web 360
Filipe Roberto Silva, Ronaldo dos Santos Mello

paper:12

A strategy to preserve data confidentiality in cloud storage services

Eliseu Castelo Branco (Aluno)[1], **Javam de Castro Machado (Orientador)**[1], **José Maria da Silva Monteiro Filho (Coorientador)**[1]

[1]Programa de Mestrado e Doutorado em Ciência da Computação (MDCC)
Universidade Federal do Ceará (UFC)

{ecastelob, javam, monteiro}@lia.ufc.br

Level: PhD
Year of Program Entrance: 2012
Date of Qualifying Exam: August/2014
Expected Timing of Completion: November/2015
Completed Steps: Bibliographic Revision, Development of Theoretical Model
Future Steps: Submission of Propose , Development of Experiments, Presentation of Thesis and Final Report

Abstract. *Cloud storage is an increasingly popular class of services for archiving, backup and sharing of data. Despite the tremendous business and technical advantages of the cloud storage services, the data confidentiality concern has been one of the major hurdles preventing its widespread adoption. However, the solutions, in general, that were found in the literature, to ensure data confidentiality, use data encryption. Encryption is a useful tool for protecting the confidentiality of sensitive data. Nevertheless, when data is encrypted, performing queries becomes more challenging. In this paper, we present a strategy to preserve data confidentiality in cloud storage services. The proposed strategy uses information decomposition and dispersion to separate data into unrecognizable parts and store them in distributed hosts in the cloud. Besides this, our approach is not based on data encryption.*
keywords: *Data Fragmentation, Privacy, Cloud Database.*

1. Theoretical Foundation

Cloud computing is the next-generation architecture of IT enterprises, providing convenient remote access to massively scalable data storage and application services. Cloud storage is an increasingly popular class of online services for archiving, backup and sharing of data, besides it is also envisioned for future primary storage of enterprise data. Despite the tremendous business and technical advantages of the cloud storage services, the privacy concern has been one of the major hurdles preventing its widespread adoption.

The concept of privacy varies widely among countries, cultures and jurisdictions. So, a concise definition is elusive if not impossible. For the purposes of this discussion, privacy is "the claim of individuals, groups or institutions to determine for themselves when, how and to what extent information about them is communicated to others" [Camenisch et al. 2011]. Privacy protects access to the person, whereas confidentiality protects access to the data. So, confidentiality is the assurance that certain information that may include a subject's identity, health, lifestyle information or a sponsor's proprietary information would not be disclosed without permission from the subject (or sponsor). When dealing with cloud environments, confidentiality implies that a customer's data and computation tasks are to be kept confidential from both the cloud provider and other customers [Zhifeng and Yang 2013].

Actually, there are some techniques, proposed by the academic community, in order to ensure data protection, such as: generalization, suppression, shuffling, disturbance and data fragmentation.

Next, we present these techniques proposed by the academic community:

a) generalization: It substitutes the values of quasi-identifiers attributes by fewer specific values, but semantically consistent, which represent them. The technique categorizes the attributes, creating a taxonomy of values with levels of abstraction going from the particular level to the generic.
b) suppression: It erases some values of identifying attributes, quasi-identifying ones or even records from the table.
c) shuffling: It substitutes sensitive data (identifiers, quasi-identifiers and sensitive attributes) by encrypted data.
d) disturbance: this technique is used to replace values of real data by fictitious data. The general idea consistis in changing data to disguise confidential information while preserving the characteristics of data, which are critical for the data model.
e) data fragmentation: this technique combined use of encryption and data fragmentation has first been proposed in the context of data outsourcing. Privacy requirements are enforced by splitting information over some independent database servers (so to break associations of sensitive information) and by encrypting information whenever necessary.

Using and combining these techniques, two approaches to ensure data confidentiality were found in the literature:

a) Statistical Disclosure Control (SDC): It manages the protection of data that can be published without revealing confidential information which can be related to specific people to which the data correspond to. The protection that the SDC techniques provide, provoke, in some degree, modification in the published data, within the limits of no modification (maximum utility for the users and no data protection) and data encryption (maximum protection and no utility for the use without the cryptographic key) [Domingo-Ferrer 2008]. The SDC approach is appropriate when the data to be stored in the cloud need to be public.
b) Fragmentation-Redundancy-Scattering (FRS): is a technique which enables to recover erroneous data destroyed or contaminated by accidental faults or by intrusions, while preserving the confidentiality of sensitive data. This technique has been applied to different functions of distributed systems such as persistent file storage, security management and data processing. This approach consists in first cutting all sensitive information into fragments, in such a way that any isolated fragments do not contain significant information, and second scattering the fragments among the different sites of the distributed system, so that an intrusion into a part of the distributed system give access only to unrelated fragments. The FRS approach is appropriate when the data to be stored in the cloud does not need to be public.

Encryption is a useful tool for protecting the confidentiality of sensitive data. However, when data is encrypted, performing queries becomes more challenging. So, while data encrypting provides confidentiality, encrypted data is much less convenient to use than a cleartext data.

In this paper we present a strategy to preserve data confidentiality in cloud storage services. The proposed strategy uses information decomposition and dispersion to separate data into unrecognizable parts and store them in distributed hosts in the cloud. Besides this, our approach is not based on data encryption.

The remainder of this paper is organized as follows. Section 2 discusses the key concepts utilized to describe the proposed approach. Section 3 presents the current work status. Section 4 shows the related works. Finally, Section 5 concludes this article and points out directions for future research.

2. Characterization of Contribution

In order to correct the problem of lack of confidentiality in the storage of files in the cloud, we present the *I-OBJECT*, a solution of anonymisation by decomposition, fragmentation in k parts and dispersion of the file for n servers. The architecture proposed is presented in figure 1.

In this scenario, the provider is semi-honest, meaning, it executes the protocols of the system, but it is interested in obtaining extra information on data. In this paper, we are interested in the problem of ensuring the confidentiality of the data stored and processed in cloud computing environments.

In the proposed strategy, the data files are considered sets of information objects (i-objects). Our concept of i-object (information object) is a sequence of

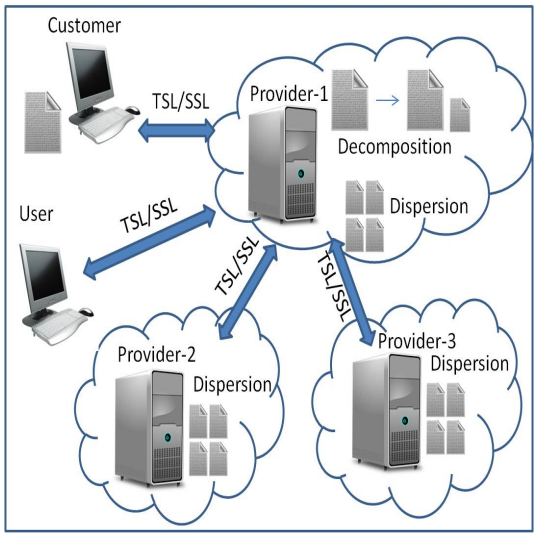


Figure 1. Data protection scenario

256 bytes from a digital file. Each object, according to [Hegel 1995], has three characteristics that determine It: quality, quantity and measure. In a digital file, we consider that quantity is the proportion in which each byte is present in the file, quality is composed by the bytes that can be present in the file and the measure is the order in which the bytes are arranged sequentially in the file. E.g., in a file of 256 bytes that just occurs bytes that represent the letters "A", "B", "C" and "D" in equal proportion, we have:

Data file: "ABCDABCDABCDABCDABCDABCDABCD...ABCD" (256 bytes)
Quantity Array: $\{64, 64, 64, 64\}$
Quality Array: $\{A, B, C, D\}$
Measure Array: $\{1^o, 5^o, 9^o, 13^o...253^o\}$,$\{2^o, 6^o, 10^o, 14^o..., 254^o\}$,$\{3^o, 7^o, 11^o, 15^o..., 255^o\}$,
. $\{4^o, 8^o, 12^o, 16^o, 256^o\}$

The contribution of the I-OBJECT system, in relation to the storage system of existing distributed information, such as: Oceanstore, Pergamun, POTSHARDS, PASIS, Gridsharing, Glacier, CleverSafe e Tahoe-LAFS, is the technical creation of decomposing data which is presented as follows: the decomposition technique extracts information from the data files on quantity, quality and measure. The processing steps for the decomposition of the file are described as follows:

a) The decomposition algorithm reads sequences of 256 bytes from the data file. We will refer, from now on, to this set of bytes as I-Object.
b) The algorithm extracts the information of quality ,quantity and measure from I-Object, storing these information in two arrays of 256 elements size each: the array of Quantity-Quality[256]gross and the array of Measure characters [256]. We will refer to these arrays as vectors from now on.
c) The vector Quantity-Quality[256] will store, for each one of the different bytes existent in I-Object, the number of times this byte is found in the I-Object. For example, if the byte (00001111)2 = (15)10 is present 20 times in the I-Object, the

item Quantity-Quality[15] will be equal to 20. If byte (15)10 is not present, the value of the item Quantity-Quality[15] will be equal to zero.

d) For each item of the vector Quantity-Quality, the algorithm of decomposition, converts the value of the item in a sequence of '1' bits, if the element of the vector is more than zero. Example: Quantity-Quality[25]=3 => VectorBits[25] = '111'. If the element of the vector Quantity-Quality is equal to zero, the VectorBits will not store any value.

e) The items of the VectorBits are concatenated as following: VectorBits[0]+'0'+ VectorBits[1]+'0'+,...+'0'+VectorBits[255], producing a vector of 512 elements, which is used as an entrance in a function which reads the vector in a sequence of 8 items and converts it for the correspondent ASCII presentation, creating a sequence of 64 bytes, which is recorded in the file quantity-quality.dec. Example: '01000001' is converted to the letter 'A' .

f) The vector of Measure[256]characters will store, for each element of the vector Quantity-Quality[256] > 0, the order in which the bytes appear in the I-Object. The position of bytes will vary from 0 to 255, representing from the 1st to the 256th byte present in the data block. The vector will use the decimal value of the byte to represent the values of the bytes positions of the I-Object. Table 1 shows an example in which byte-1 occurs 3 times and byte-3 occurs 1 time in the I-Object and there is no occurrence of bytes 0, 2 and 255. The Measure[256] is recorded in the file measure.dec.

Table 1. Example of Filling In of Quality-Quantity and Measure Vectors

Vector Items Quantity_Quality[256]	Measure[256] Vector Items
Quantity_Quality[0]=0	
Quantity_Quality[1]=3	Measure[1] = 5_{10} = 0000 0101_2 Measure[2] = 27_{10}=0001 1011_2 Measure[3] = 43_{10} =0010 1011_2
Quantity_Quality[2]=0	
Quantity_Quality[3]=1	Measure[4] = 54_{10} =0011 0110_2
...	
Quantity_Quality[255]=0	

The files measure.dec and quantity-quality.dec are stored in different cloud providers. In this case, each one of the files is insufficient to rebuild the original file. For example, assuming that the providers which have the quantity-quality.dec file try to rebuild a block of 256 bytes from the original file. Using a brute-force method to try to rebuild the sequence of 256 bytes of the I-Object, the probability of the provider to discover the correct sequence of the bytes, knowing the quantity and the quality is a repeated permutation of 256 elements: Probability of reassembling the I-Object (P) = $1/P_{256}{}^{n1,n2,n3...}$, where n_1,n_2,n_3... are known quantity and quality items. For an I-Object with only 1 diverse byte, the probability is of P = 1/256. For an I-Object with 2 diverse bytes, the probability is of approximately P = $1/10^{76}$. As the diversity of bytes increases, the probability of discovering the order of bytes tends to be zero. For example, with 10 diverse bytes, the probability reaches P = $1/10^{256}$. For the cloud provider which stores the measure.dec, in other

words, the order in which the bytes are arranged in the block, the probability of recomposition of the I-Object, using brute-force is P = 1/256!, in other words, approximately $1/10^{506}$. A file is formed by a set of *n* I-Objects. The higher the value of n, the greater the difficulty of an attacker to rebuild it, from the measure file or from the quantity-quality file. Since these files are dispersed in sites of distinct providers in the cloud, no provider can obtain any information on the content of the fragment it stores.

The advantages of this technique over the conventional techniques that use cryptography to guarantee confidentiality of data stored in the cloud are the following:

a) There is no use of cryptographic keys. Confidentiality is obtained through fragmentation and dispersion of data.
b) the solution supports purge of cloud data as the files available in distinct providers do not reveal information on original data. If the user leaves the cloud, data can be considered purged automatically
c) secret splitting provides attack resilience because, unlike pure encryption, it is requires the adversary to collect multiple pieces of data to reconstitute any portion of the original block
d) for long term survivability, data must be recoverable without any information from outside the set of archives; fulfilling requests in a reasonable time cannot require any outside data, such as external indexes or encryption keys

The recomposition of the original file is done by the recovering of the quantity-quality.dec and measure.dec which are stored in 2 distinct providers in the cloud which are used as an entrance to the reassembling algorithm of the original file. The computational complexity of the decomposing algorithm is of O(2n) and of the recomposition is O(n).

3. Current Work Status

To date, the algorithms for decomposition and recomposing of data files have already been developed. The next steps of the work are the following:

a) There will an elaboration of the formal description of the data confidentiality model.
b) Development of algorithms of data dispersion and the algorithms to generate the *k* quantity, quality and measure files.
c) Development of interface for the cloud data storage, including the management of the access control list of users.
d) There will execute performance test of algorithms.

4. Related Work

In the last years, several proposals to ensure data confidentiality in cloud services have been proposed [Stefanov and Shi 2013, Li et al. 2013, Yang et al. 2013, Jung et al. 2013, Yeh 2013, Nimgaonkar et al. 2012]. These solutions can de classified in two groups or approches: Statistical Disclosure Control (SDC) and Fragmentation-Redundancy-Scattering (FRS).

In the first group, the main techniques are: k-anonymity, l-diversity, t-closeness, b-likeness e δ-Presence. However, these techiques are not directly related to the solution proposed in this paper.

In the second group are the techniques based on data decomposition and dispersion. These techniques consist on decomposing a file in n parts and store these parts at different sites. The retrieving of k from the n parts, where $k < n$ are sufficient to remake an original file. One premise of this technique is that a set smaller than $k - 1$ sites will not cooperate to remake a file or will not be compromised by an invader, what guarantee the confidentiality of the file. Besides, these techniques are based on criptografy.

The main dispersion techniques of information presented by [Storer et al. 2008, Rhea et al. 2001, Storer et al. 2009, Goodson et al. 2004, Subbiah and Blough 2005] are described as follows.

Table 2. Capability Overview of the Storage Systems

archival storage system	technical protocol	data retrievability	secrecy	consistency
I-OBJECT	decomposition	replication	secret splitting	hash() function
Pergamum	Rabin's Approach	erasure-code	secret splitting	algebraic signatures
OceanStore	Rabin's Approach	erasure-code	encryption	hash() function
POSTHARDS	Shamir's Secret Sharing	RAID	secret splitting	algebraic signatures
PASIS	Rabin's Approach	erasure-code	secret splitting	cross checksums
GridSharing	Shamir's Secret Sharing	replication	secret splitting	replication

a) Shamir's Secret Sharing: this technique proposes to divide data in n parts, so that they can be retrieved from any k parts, where $k < n$, but that does not reveal any information from $k - 1$ parts of the data. This technique is used to create key management schemes for cryptographic systems. One assumes that the data can be represented by a number (D). The objective is to divide the number in n parts ($d_1, \ldots, d_n$), in such way that the knowledge of any k or more d_i parts, makes D computable. For this, one uses a polynomial of degree $k - 1$, $q(x) = a_0 + a_1x + ... + a_{k-1}x^{k-1}$ on some F finite field , in which $a_0 = D$. Knowing any S set of cardinality to minus k of elements $q(i)$ and knowing the set of indexes correspondent to the S elements as being I_s. One uses the Interpolation of Lagrange to calculate D as it follows: $J = \sum_{j \in I_s} \gamma_j p(j)$ where $\gamma_j = \prod_{j \in I_s} j(j-1)^{-1}$ The disadvantage of this technique is the increasing of storage space requirement due to a n factor and lack of capacity to detecting and correcting errors.
b) Rabin's Approach: uses the concept of information dispersal algorithms (information dispersal algorithms - IDAs). It proposes the use of non-systematic

erasure code (erasure code) instead of polynomial secret sharing. The D data is seen as a vector ($vector(D)$) with k elements and one can choose a matrix of invertible dispersion $M(n\ x\ k)$. The keyword is obtained as $array(c) = M.array(D)$. The original data is obtained as $array(D) = M^{'-1}.array(c)^{'}$, where $array(c)^{'}$ contains k of n elements of the keyword and $M^{'}$ is the correspondent Sub-matrix of M. Although efficient in the use of space and processing time, this technique does not guarantee confidentiality of data, so one recommends encrypting of data before using the technique. This technique reduces the increasing of storage space for a factor of n/k.

c) Secret Sharing Made Short: this technique uses a safe symmetric encryption scheme (IND-CPA), that provides identicalness – the adversary cannot learn any information from the x clear text, from a y cipher - and that is resistant to a chosen plaintext attack - CPA, situation in which the adversary can select ciphers from clear texts as it chooses. In this technique, a sk random symmetric secret key is used to encrypt the D data. To the result of encryption, one applies an information dispersion algorithm (for example: Rabin's Approach) to create a $vector(C)$ containing n keywords as elements, after one applies a sharing technique of polynomial secret to compute n sharing $s_1, \cdots, s_n$ of the sk key. Finally, the $(array(c_i), s_i)$ set is distributed to I sites, where i is a factor of n.

d) AONT-RS: It is a new approach to the Rabin algorithm, adding confidentiality. This technique avoids using sharing of polynomial secret and, instead, uses a variant of Rivest AONT Algorithm (All-or-nothing transform) and, as a step of data preprocessing before using an information dispersion algorithm.
The technique works as follows: the D data is seen as a array ($array(D)$) with $k+1$ elements, where element $k+1$ is a value of integrity check (checksum). An sk random key is chosen from a symmetric encryption scheme to encrypt the (D) array elements, creating the (E) array as follows: $e_i = d_i\ xor\ E(i+1, sk)$. After encrypting all the $k+1$ elements of (D) array, a new element is inserted in array $d_{k+2} = sk\ xor\ H(e_1 \parallel \cdots \parallel e_{k+1})$, where H is a hash cryptographic function. The sk key will be distributed in the $k+1$ elements of (D) array. The following step is the application of a Reed-Solomon(RS) erasing code to produce M where the first $k+2$ columns compose an identity matrix. The keyword is obtained as $array(c) = M.array(E)$. Ultimately, the n elements of $array(c)$ are distributed to the different sites.

5. Conclusions and Future Works

This article introduced I-OBJECT, a system designed to provide privacy longterm archival storage to address the new challenges and privacy threats posed by archives that must securely storing data on the cloud. The proposal's approach showed that it is possible to perform confidentiality in cloud storage services without use encryption techniques.

Several open issues still remain to be addressed such as: the integrity and availability of fragmented data. Simulations and benchmarks with actual cloud-storage providers will be performed to test the performance of the algorithms in the coming months.

References

Camenisch, J., Crispo, B., Fischer-Hübner, S., Leenes, R., and Russello, G. (2011). Privacy and identity management for life: 7th ifip wg 9.2, 9.6/11.7, 11.4, 11.6 international summer school, trento, italy, september 5-9, 2011. revised selected papers. In *The 7th IFIP WG 9.2, 9.6/11.7, 11.4, 11.6 International Summer School on Privacy and Identity Management for Life*.

Domingo-Ferrer, J. (2008). A survey of inference control methods for privacy-preserving data mining. In Aggarwal, C. C. and Yu, P. S., editors, *Privacy-Preserving Data Mining*, volume 34 of *Advances in Database Systems*, pages 53–80. Springer.

Goodson, G. R., Wylie, J. J., Ganger, G. R., and Reiter, M. K. (2004). Efficient byzantine-tolerant erasure-coded storage. In *Dependable Systems and Networks, 2004 International Conference on*, pages 135–144.

Hegel, G. W. F. (1995). In *Enciclopédia das ciências filosóficas II-Filosofia da natureza*. Edicoes Loyola.

Jung, T., Li, X.-Y., Wan, Z., and Wan, M. (2013). Privacy preserving cloud data access with multi-authorities. In *INFOCOM*, pages 2625–2633. IEEE.

Li, M., Yu, S., Ren, K., Lou, W., and Hou, Y. T. (2013). Toward privacy-assured and searchable cloud data storage services. *IEEE Network*, 27(4):1–0.

Nimgaonkar, S., Kotikela, S., and Gomathisankaran, M. (2012). Ctrust: A framework for secure and trustworthy application execution in cloud computing. In *CyberSecurity*, pages 24–31.

Rhea, S., Wells, C., Eaton, P., Geels, D., Zhao, B., Weatherspoon, H., and Kubiatowicz, J. (2001). Maintenance-free global data storage. *Internet Computing, IEEE*, 5(5):40–49. 1089-7801.

Stefanov, E. and Shi, E. (2013). Oblivistore: High performance oblivious cloud storage. In *IEEE Symposium on Security and Privacy*, pages 253–267.

Storer, M. W., Greenan, K. M., Miller, E. L., and Voruganti, K. (2008). Pergamum: Replacing tape with energy efficient, reliable, disk-based archival storage. In *Proceedings of the 6th USENIX Conference on File and Storage Technologies*, page 1. USENIX Association.

Storer, M. W., Greenan, K. M., Miller, E. L., and Voruganti, K. (2009). Potshards-a secure, recoverable, long-term archival storage system. *Trans. Storage*, 5(2):1–35.

Subbiah, A. and Blough, D. M. (2005). An approach for fault tolerant and secure data storage in collaborative work environments.

Yang, K., Jia, X., Ren, K., Zhang, B., and Xie, R. (2013). Dac-macs: Effective data access control for multiauthority cloud storage systems. *IEEE Transactions on Information Forensics and Security*, 8(11):1790–1801.

Yeh, C.-H. (2013). A secure shared group model of cloud storage. In *Proceedings of the 2013 27th International Conference on Advanced Information Networking and Applications Workshops*, WAINA '13, pages 663–667, Washington, DC, USA. IEEE Computer Society.

Zhifeng, X. and Yang, X. (2013). Security and privacy in cloud computing. *Communications Surveys Tutorials, IEEE*, 15(2):843–859.

paper:24

Master's Thesis: A Tuning Approach Based on Evolutionary Algorithm and Data Sampling for Boosting Performance of MapReduce Programs

Tiago R. Kepe[1], **Eduardo Cunha de Almeida**[1]

[1]Programa de Pós-Graduação em Informática (PPGINF)
Universidade Federal do Paraná (UFPR)
Curitiba – PR – Brazil

{trkepe,eduardo}@inf.ufpr.br

- **Master's admission year**: 07/2012
- **Proposal's defense date**: 11/2013
- **Expected finish date**: 12/2014
- **Concluded steps**:
  - Implementation;
  - Initial results.
- **Future steps**:
  - Generating relevant results;
  - Analysing the new results;
  - Writing the thesis.

Abstract. *The Apache Hadoop data processing software is immersed in a complex environment composed of huge machine clusters, large data sets, and several processing jobs. Managing a Hadoop environment is time consuming, toilsome and requires expert users. Thus, lack of knowledge may entail misconfigurations degrading the cluster performance. Indeed, users spend a lot of time tuning the system instead of focusing on data analysis. To address misconfiguration issues we propose a solution implemented on top of Hadoop. The goal is presenting a self-tuning mechanism for Hadoop jobs on Big Data environments. To achieve this, our self-tuning mechanism is inspired by two key ideas: (1) an evolutionary algorithm to generate and test new job configurations, and (2) data sampling to reduce the cost of the self-tuning process. From these ideas we created a framework for testing usual job configurations and get a new configuration suitable to the current state of the environment. Experimental results show gains in job performance against the Hadoop's default configuration and the rules of thumb. Besides, the experiments prove the accuracy of our solution which is the relation between the cost to obtain a better configuration and the quality of the configuration reached.*

Keywords: Big Data, MapReduce, Hadoop, Self-Tuning

1. Introduction

Nowadays, companies and research scientific institutes are awash in a flood of data where "Big Data" has emerged. A standard solution and vastly used for Big Data purposes is the programming framework MapReduce (MR). It was an initiative of the Google company in 2004 when they disclosed the MapReduce framework [Dean and Ghemawat 2004] for storing and processing data on large clusters.

A popular open source implementation of the MR framework is the Apache Hadoop which has an intuitive interface to create MR jobs. Hadoop orchestrates parallel processing, handles node and task failures, stores data in a distributed way, recovers data and so on. In addition, users ought to properly setup a MR job because misconfiguration may degrade the job performance [Kambatla et al. 2009]. However, the Hadoop's environment is complex and there is a huge number of parameters to be configured for each job, such as: settings for memory allocation, I/O controllers, network timeouts etc. Performance and resource usage of jobs are directly bound to these configuration parameters. Normally, a job is configured manually by developers or testers, which is obviously time consuming as the process has to be iterative: configuration of job, execution of the job, measurement of performance, re-configuration of the job and so on. Accomplishing tuning on real Hadoop environments becomes impracticable due to the large amount of jobs, the high number of parameters involved larger than 250 and the cluster "dynamism".

2. Contribution

To perform self-tuning of MR jobs we were inspired by two key ideas: the Bacteriological Algorithm (BA) [Baudry et al. 2005] using the job execution time as fitness value, and data sampling which led us to create a dynamic reservoir sampling algorithm called KSample. Whilst the BA generates new job configurations along the execution, KSample reduces the cost to test these configurations.

2.1. Algorithm for testing

The Bacteriological Algorithm (BA) belongs to the family of evolutionary algorithms. We adapted the BA to work in the genetic context. Our BA implementation has the gene as atomic unit, a group of genes forms an individual (bacterium), and a group of individuals forms a population (bacteria). As Hadoop is involved in a peculiar environment, we performed a context transformation to adapt the BA to work on the Hadoop context. We called a specific configuration parameter as **knob**, an entire setup as **set of knobs** and the combination of setups as **sets of knobs**. In the context transformation, each component of the genetic context was translated to a component of the Hadoop context. A gene was transformed to a knob, an individual was transformed to a set of knobs, and individuals population was transformed into sets of knobs.

The BA in high-level of abstraction is described in the Algorithm 1. It receives as input an initial sets of knobs, the number of generations to be reached and the desired fitness value. (1) The following process computes the fitness value for each set of knobs: initially, a job is configured with the current set of knobs and sent to be executed over a data sample. Then the job execution time is assigned as the fitness value for this set of knobs. (2) The best set of knobs reached is memorized based on the fitness value. (3) The reproduction operator creates a new generation composed of clones of the best set of

knobs. (4) The mutation operator iterates on the new population, and for each set of knobs one or more knobs are randomly selected and their values are changed randomly. The process continues until the number of generations or the desired fitness value is reached.

Algorithm 1: Bacteriological Algorithm

Input : Pop Initial population
Input : Gen Number of generations
Input : $Fitness$ Desired fitness value
Output: $BestKnobs$ The best set of knobs reached
repeat
  for *each* $knobs \in Pop$ **do**
    CalcFitness($knobs$)
  $BestKnobs \leftarrow Memorization$(Pop, BestKnobs)
  $Pop \leftarrow Reproduction$(BestKnobs)
  $Pop \leftarrow Mutation$(Pop)
until $Gen \vee Fitness$;
return $BestIndiv$

2.2. Sampling on MR

The BA's process allows to create new configurations and test them in order to select which one is more suitable to the MR job at the system's current state. However, running the algorithm in the whole data set may be unfeasible, especially for big data. Thus, to achieve automation of testing, it is crucial to apply techniques that efficiently reduce the amount of data to be processed. A common approach used on relational DBMSs is data sampling. However, on Big Data context, data sampling is challenging due the large amount of data, not only the sampling must be done in a distributed fashion, but also the storage.

Thus, according to [Vitter 1985, Cloudera, Inc 2013, Grothaus 2013] unstructured data stream, which is common for MR applications, can be addressed by Reservoir Sampling. However, the size of the reservoir can be challenging to find due to the size of data. Then, we created a new algorithm called (*KSample*) which is inspired by the Reservoir Sampling algorithm. KSample receives a percentage (as parameter) of the population to be sampled. For instance, 10% is received as input, then KSample will create a reservoir sample that represents 10% of the input data records. The KSample works on unstructured data, the atomic unit for the algorithm freely can be defined such as: record, row, byte, file, etc. The KSample works with an undefined reservoir size and increases the reservoir sample on demand.

We proved mathematically, see Appendix A, that the KSample algorithm holds at least $\rho\%$ of the input data. We built the KSample as a MR application in order to execute distributively. Basically, the map function receives the key/value pair $\langle lineNumber, lineContent \rangle$ and generates the intermediate pair $\langle randomNumber, lineContent \rangle$. MR sorts the intermediate pairs and aggregates the values that share the same key, then the reduce receives the pair $\langle key, list < lineContent > \rangle$ and runs the KSample locally.

Algorithm 2: KSample Algorithm

```
Input  : percentage percentage for sampling
Input  : stream data stream with undefined length
Output: reservoir[ ]
sLength ← 0
slotRound ← 0
while stream != EOF do
    sLength++
    if reservoir.size() < (percentage × sLength) then
        reservoir.newSlot()
        slotRound ← 0
    slotRound++
    probability ← 1 / slotRound
    rand ← Random(0, 1)
    if rand ≤ probability then
        reservoir.currentSlot ← stream[sLength]
return reservoir
```

2.3. Framework for testing

We created a framework for testing the configuration setup composed of three main modules: the front-end for users interacting with the system using a Domain-Specific Language (DSL), the engine to test job configurations and the back-end that saves the best configuration found by the engine.

The engine is composed of three sub-modules:

- Sampler: generates data sampling by invoking the KSample;
- Core: generates job configurations using the BA;
- AutoConf: enforces Hadoop applying the configurations generated by the Core.

An overview of the framework is presented in the Figure 1. First, (0) an user creates a file following our DSL, (1) this file is submitted to the Front-end. (2) The Analyser is invoked to validate the DSL code. (3) The parsed file is sent to the Engine component, it actives the Sampler module (4). After the core starts the BA process that will generate new configurations and send them to the AutoConf (5). The Core submits the job to the Hadoop (6), when Hadoop queues the job for execution, it sends a request to the AutoConf, which returns the current configuration for the job. The steps 5, 6 and 7 occur until the BA process is finished. In the end the resulting configuration is sent to the Back-end. So, users will have a new configuration suitable for the current state of the environment or at least a good trade-off about the job settings.

3. Related tuning techniques for MapReduce

Some rules of thumbs have been created to adjust Hadoop environments based on administrators and developers' knowledge [White 2009, Cloudera 2014, Intel 2010]. Applying the rules is straightforward, but not individually accurate. The rules are generic, aiming to be applied in all jobs without regard to specific job behaviors.

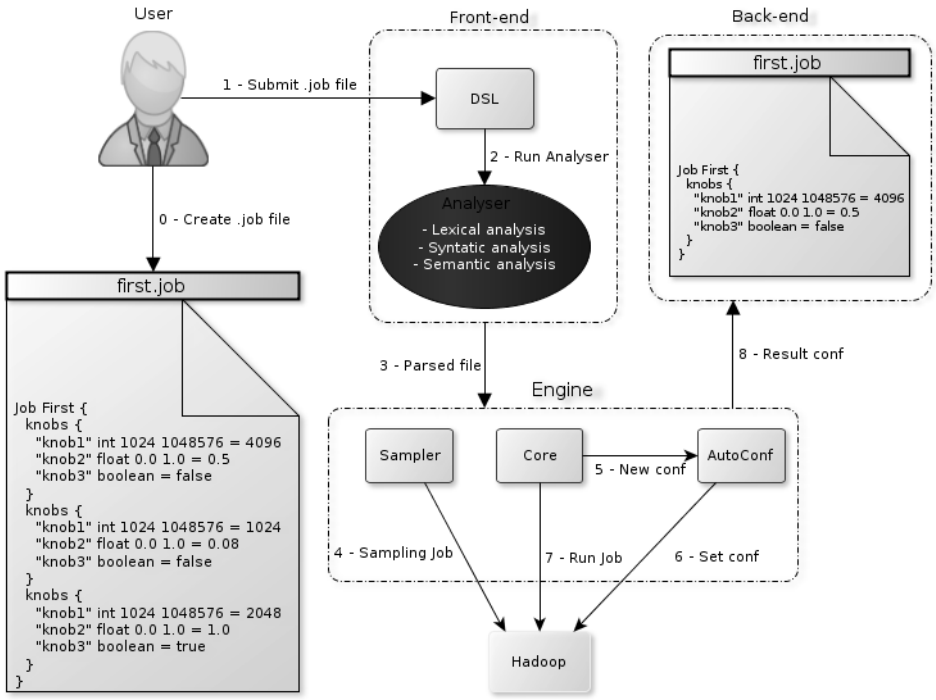


Figure 1. Framework overview.

Another solution is profile-driven tuning which is based on job profiles using performance models, such as Starfish [Herodotou et al. 2011, Herodotou and Babu 2011]. Profile-driven tuning requires a previous job execution, when a new job is submitted, Starfish runs it and uses dynamic instrumentation to collect statistical data. The drawback is performance overload caused by this instrumentation.

Another alternative is to use simulators such as MRPerf [Wang et al. 2009], WaxElephant [Ren et al. 2012] and SimMapReduce [Teng et al. 2011]. The simulators emulate real big data environments, but they require extra information such as: existing workloads, log files and other relevant sources. However, they might not simulate events that only happen in the real world, such as: cluster volatility, i.e. nodes entering and leaving of the cluster, data dynamism that can be delivered in new formats, etc.

4. Related data sampling approaches for MapReduce

BlinkDB is a query engine for running interactive SQL queries on large volumes of data. It focuses on running short exploratory queries to provide trade-off query accuracy for response time [Agarwal et al. 2013]. BlinkDB creates a set of stratified samples based on the past queries behaviors. EARL [Laptev et al. 2012] is another framework for Hadoop to run queries on samples. It uses the bootstrapping sample technique, but the sampling algorithm requires some assumptions set by the users, such as: the size and the number of samples.

Those data sampling approaches requires prior knowledge about data or the query history. However, we are interested in using a naive input for sampling in order to abstract the complexity from the users, like a percentage of the data population.

5. Initial Results

We performed initial experiments on Hadoop with 1 master machine and 2 slave machines. These 3 machines ran Linux SO with 3GB of memory. Each slave machine was populated with 10GB of random texts using the job *randomtextwriter* embedded in the Hadoop example jar file. We ran 10 generations (rounds) of the BA using the WordCount job on a sample of 20% percent of the data (4GB). We saved the best configuration reached in each round and applied them in the whole data set (20GB) to measure the quality of the configurations. We also ran the BA without sample to compare with our tuning approach.

Figure 2 shows the quality of our approach against the default Hadoop configuration and the rules of thumb applied on the WordCount job. The configuration reached in the round #10 had a gain in performance of 10% compared with the default configuration and almost 25% against the rules of thumb. The same graphic shows that the best configuration reached with and without sampling had a similar performance, but the cost using data sampling is significantly lower, as can see in Figure 3, where the BA without sample was 6 up to 7 times less costly.

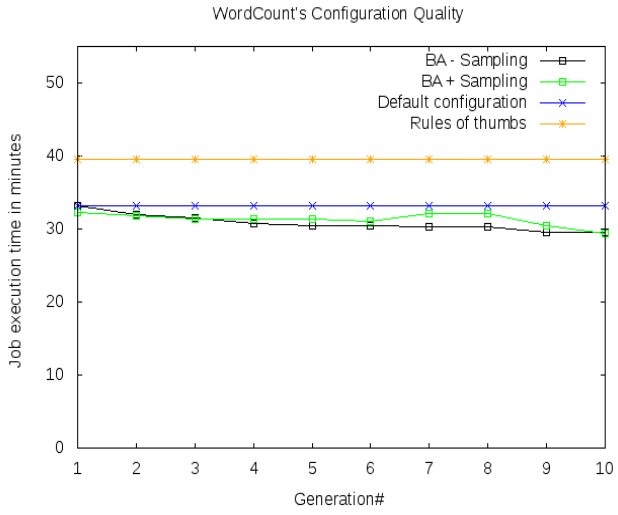


Figure 2. Configuration quality.

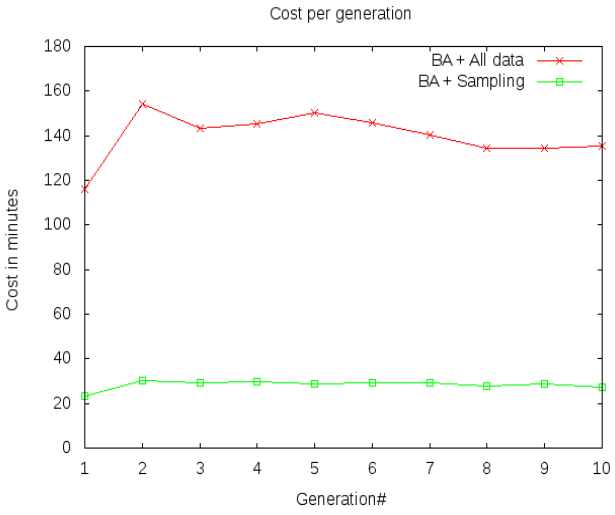


Figure 3. Self-tuning cost.

6. Future work

For future work we plan to perform more experiments on a large cluster to corroborate the accuracy of our tuning approach. We have considered just the job execution time as fitness value, we should be considering other metrics like: IO operation, CPU, Memory and Network usage. In cloud computing premises, users might be more concerned about resource usage than in response time, as cloud service providers offer plans based on "pay as you go", such as Amazon EC2. We are also studying to implement other algorithms to perform MR self-tuning, such as: hill climbing and simulated annealing.

References

Agarwal, S., Mozafari, B., Panda, A., Milner, H., Madden, S., and Stoica, I. (2013). Blinkdb: Queries with bounded errors and bounded response times on very large data. EuroSys '13, New York, NY, USA. ACM.

Baudry, B., Fleurey, F., Jézéquel, J.-M., and Le Traon, Y. (2005). From genetic to bacteriological algorithms for mutation-based testing: Research articles. *Software, Testing, Verification & Reliability (STVR)*, 15:73–96.

Cloudera (2014). 7 tips for improving mapreduce performance. http://blog.cloudera.com/blog/2009/12/7-tips-for-improving-mapreduce-performance.

Cloudera, Inc (2013). Algorithms every data scientist should know: Reservoir sampling. Site: http://blog.cloudera.com/blog/2013/04/hadoop-stratified-randosampling-algorithm.

Dean, J. and Ghemawat, S. (2004). Mapreduce: Simplified data processing on large clusters. In *OSDI*. USENIX Association.

Grothaus, G. (2013). Reservoir sampling - sampling from a stream of elements. Site: http://gregable.com/2007/10/reservoir-sampling.html.

Herodotou, H. and Babu, S. (2011). Profiling, what-if analysis, and cost-based optimization of mapreduce programs. *PVLDB*, 4(11):1111–1122.

Herodotou, H., Lim, H., Luo, G., Borisov, N., Dong, L., Cetin, F. B., and Babu, S. (2011). Starfish: A self-tuning system for big data analytics. *CIDR '11*.

Intel (2010). Optimizing hadoop* deployments. Technical report.

Kambatla, K., Pathak, A., and Pucha, H. (2009). Towards optimizing hadoop provisioning in the cloud. HotCloud'09, Berkeley, CA, USA. USENIX Association.

Laptev, N., Zeng, K., and Zaniolo, C. (2012). Early accurate results for advanced analytics on mapreduce. *Proc. VLDB Endow.*, 5(10):1028–1039.

Ren, Z., Liu, Z., Xu, X., Wan, J., Shi, W., and Zhou, M. (2012). Waxelephant: A realistic hadoop simulator for parameters tuning and scalability analysis. CHINAGRID '12, Washington, DC, USA. IEEE Computer Society.

Teng, F., Yu, L., and Magoulès, F. (2011). Simmapreduce: A simulator for modeling mapreduce framework. In *MUE*, pages 277–282. IEEE.

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57.

Wang, G., Butt, A. R., Pandey, P., and Gupta, K. (2009). A simulation approach to evaluating design decisions in mapreduce setups. In *MASCOTS*, pages 1–11. IEEE.

White, T. (2009). *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition.

A. KSample's proof

KSample is based on the fact that the reservoir always holds at least a gave $percentage$ of elements. That is the invariant property of the KSample, i.e. independent of the current stream length, it ensures that in any step the reservoir always will hold at least a gave $percentage$ of elements that arrived. Also, the KSample ensures that every element has the same probability to be inserted into the reservoir which is inherited from the reservoir algorithm, as long as it treats any slot as a mini-reservoir of size 1. The invariant property is ensured by proofing that KSample creates a new slot at right moment, i.e. if the KSample didn't create a new slot, the reservoir wouldn't contain at least the percentage of elements from the stream. Let's prove that property by induction:

- *Notations*:
  R: reservoir size.
  P: percentage of the stream.
  L: stream length.
- *Base Case*: When E_1 (first element) cames.
  R = 0, the algorithm has to decide in creating or not a new slot, for this it cheks the condition $(R < (P \times L))$.
  As $P \in]0,1]$ and $L = 1$, then $(P \times L) \in]0,1]$.
  Consequently, $R < (P \times L)$ is true and a new slot is created, thus the reservoir will hold E_1 and it will have at least P percentage of elements from the stream.
- *Induction Hypothesis (I.H.)*:
  Suppose in the step $\#N$ after the E_n element arriving at the reservoir holds at least P percentage of elements from the stream.
- *Induction Step*: step $\#(N+1)$.
  We have to prove two cases:
  1. *Create a new slot*:
     For this case the condition: $R < (P \times L)$ has to be true. By the I.H. in the last step (step $\#N$) the reservoir holds $P\%$ of elements from the stream, as the KSample will create a new slot and will hold the element E_{n+1} (following the reservoir sample algorithm), then, certainly, adding a new slot the reservoir size will increase, thus it will continue holding at least $P\%$ of elements from the stream.
  2. *Don't create a new slot*:
     The KSample decided not to create a new slot, then $R \geq (P \times L)$, means that the reservoir is holding $P\%$ or more elements from the stream, otherwise would be the case 1.

  Therefore, after the step $\#(N+1)$ the reservoir size is invariant in any round of the KSample, because it will contain at least $P\%$ of elements from the stream.

paper:30

Arquitetura Híbrida de Integração entre Banco de Dados Relacional e de Grafos: Uma Abordagem Aplicada à Biodiversidade

Patrícia Raia Nogueira Cavoto[1], Orientador: André Santanchè[1]

[1]Instituto de Computação – Universidade Estadual de Campinas (UNICAMP)
Nível: Mestrado – Pós-graduação em Ciência da Computação
patricia.cavoto@gmail.com, santanche@ic.unicamp.br

Ingresso no programa: Junho de 2013
Defesa da Proposta: 25 de Abril de 2014
Previsão de conclusão: Junho de 2015

Etapas já concluídas:
- Entrevista com o usuário e levantamento de requisitos
- Revisão da literatura
- Proposta da arquitetura
- Qualificação

Etapas futuras:
- Implementação da arquitetura de integração
- Testes com o usuário
- Escrita da dissertação
- Escrita de artigos

Resumo. *A complexidade e o volume dos relacionamentos entre as informações, bem como a necessidade de manter e integrar dados de estruturas heterogêneas aumentam exponencialmente a cada dia. Isto é particularmente importante no contexto de eScience, especialmente biodiversidade, área de interesse deste projeto – em que as relações são fundamentais nas análises. Neste contexto, o modelo de banco de dados de grafos pode apresentar-se como uma abordagem mais apropriada e eficiente no gerenciamento e recuperação destas informações. Em contrapartida, há um grande legado de sistemas que utilizam bancos de dados relacionais, que cumprem um papel fundamental em diversas tarefas. Apresentamos então neste trabalho uma proposta de arquitetura híbrida de integração que permite a convivência dos modelos relacional e de grafos em sua forma nativa, reduzindo o impacto de adaptações em bases relacionais preexistentes e explorando as vantagens de cada modelo nativo nas operações de gerenciamento e recuperação.*

Palavras-chave: *integração de bases, banco de dados híbrido, modelo relacional, modelo de grafos.*

1. Introdução

Nos últimos anos, com o advento da Web Semântica e a popularização do movimento NoSQL (*Not Only Structured Query Language*), a adoção de modelos de bancos de dados de grafos está se tornando cada vez mais comum. Estes se mostram particularmente apropriados quando o modelo de dados não possui uma estrutura rígida ou apresenta grandes volumes de relações, pois têm uma infraestrutura otimizada para responder a operações com ênfase nas relações [8], por exemplo, percorrer um caminho, realizar uma busca em largura ou em profundidade. Mesmo com esse crescimento na utilização de bancos de dados de grafos, temos um grande legado em todas as áreas de sistemas que utilizam o modelo relacional para armazenamento dos dados, e este modelo é a melhor opção para alguns tipos de problema. Para aproveitar as vantagens que cada modelo de dados proporciona, pode-se integra-los em uma arquitetura híbrida unificada.

O problema tratado nesta pesquisa se apresentou a partir de um projeto de colaboração com o MNHN – *Muséum national d'Histoire naturelle*[1] – envolvendo o Xper, um *software* de criação, gestão, armazenamento, análise, edição e distribuição de dados descritivos de seres vivos [13]. Nesse *software*, sempre que um novo projeto é registrado, uma nova base de dados relacional é criada para armazenar as informações. Essas bases não são compartilhadas e esta abordagem dificulta o progresso das pesquisas, uma vez que a análise dos dados em biologia é altamente interdependente.

Os dados disponibilizados nas bases estanques do Xper possuem um grande volume de relações latentes que, se pudessem ser exploradas, possibilitariam aos biólogos a descoberta de novas informações a partir dos dados já existentes. Uma abordagem relacional para este problema torna certas consultas SQL complexas e/ou ineficientes, uma vez que este modelo apresenta, entre outras coisas, restrições na realização de consultas com relacionamentos transitivos [9], por exemplo, uma consulta para identificar o impacto ambiental sofrido com a extinção de uma determinada espécie. Bancos de dados de grafos são muito eficientes neste tipo de consulta, possibilitando a utilização de algoritmos conceituados de grafos, além de permitir a criação de regras de inferência, que facilitarão ainda mais as pesquisas dos biólogos.

Grande parte dos trabalhos analisados integrando modelos distintos de dados, [5], [14] e [15], não trata a integração específica dos modelos relacional e de grafos, e outros, mais recentes, [2], [3], [6] e [7], produzem uma *view* de grafo sobre os dados relacionais, sem materializá-la ou materializando-a somente para leitura. Dessa forma, o desafio desta pesquisa se apresenta na definição de uma arquitetura híbrida que permita a integração de uma ou mais bases de dados relacionais com um banco de dados de grafos, mantendo suas representações nativas, de forma a combinar a otimização de consultas com relacionamentos complexos e a possibilidade de gravação na base de grafos.

A principal contribuição da arquitetura proposta é a combinação das seguintes estratégias: mapeamento automático ou manual de uma ou mais bases relacionais para

[1] http://www.mnhn.fr/fr/

Agradecimentos: Trabalho parcialmente financiado pelos projetos NavScales (FAPESP-MSR Virtual), MuZOO (CNPq), Complex Descriptions on the Web (CNPq), Computational Engineering and Sciences (FAPESP/Cepid), eScience (FAPESP-PRONEX) e Web Science (INCT).

grafos; materialização deste mapeamento com replicação total ou parcial dos dados; criação de novos nós e arestas no modelo de grafos; e garantia de consistência de dados, nas operações de inserção, atualização e exclusão. Uma generalização da arquitetura proposta está ilustrada na Figura 1.

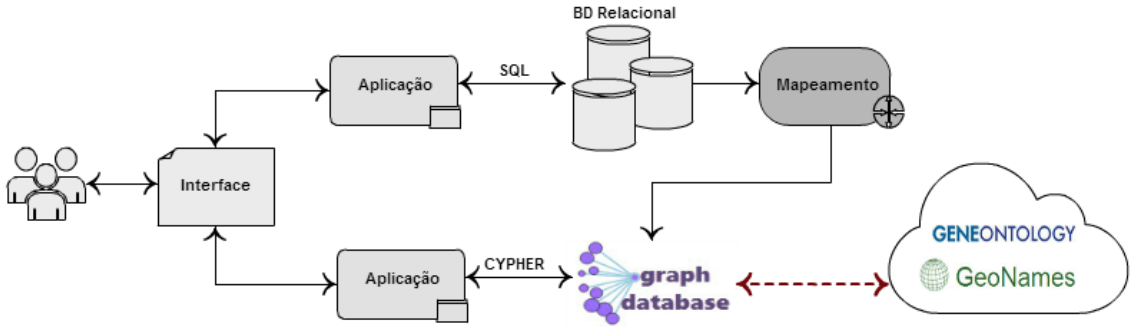


Figura 1 – Arquitetura Híbrida de Integração entre BD Relacional e de Grafos

Este trabalho está organizado da seguinte maneira: a Seção 2 apresenta a revisão da literatura e uma análise comparativa dos trabalhos relacionados; a Seção 3 detalha os desafios e contribuições; a Seção 4, trata da validação da arquitetura proposta; e, por fim, a Seção 5 apresenta as considerações finais.

2. Revisão da Literatura: Integrando Diferentes Modelos de Dados

Apresentamos a seguir uma classificação das arquiteturas propostas por trabalhos relacionados, cujo foco é a integração de dados relacionais e não estruturados, com destaque para o modelo de grafos. Essa classificação é uma contribuição deste trabalho baseada em [11] e aplicada ao cenário de integração com grafos.

2.1. Grafos em Bancos Relacionais

A base desta arquitetura é o modelo relacional, em sua forma nativa ou através da extensão de suas funcionalidades para suportar outros modelos de dados. Classificamos essa arquitetura em três principais abordagens:

(i) Mapeamento dos dados não estruturados no banco relacional: Nesta abordagem, a representação nativa não sofre modificações, pois os dados são representados no próprio modelo relacional pela criação de novas tabelas, colunas ou relações. Operações sobre os dados são realizadas utilizando operadores nativos da linguagem SQL.

(ii) Extensão do modelo relacional: O foco desta abordagem está na criação de novas estruturas na representação nativa relacional, extensões para a linguagem SQL e na aplicação de estratégias de otimização na execução das consultas. Os autores da patente descrita em [6] propõem um modelo relacional melhorado, que permite armazenar, recuperar e manipular grafos dirigidos, através da implementação de novos tipos de dados e de duas novas diretivas ao comando *SELECT*: *expand* e *depth*, utilizadas para melhorar a performance em consultas com relacionamentos transitivos. O *Database Graph Views* [7] propõe uma camada de abstração dos dados como um mecanismo para criação, visualização e manipulação de grafos, independente da organização física onde os dados originais estão armazenados. O *Virtuoso RDF View* [3] propõe o mapeamento de dados relacionais para grafos RDF sem materialização. O acesso aos dados é

realizado de forma independente, utilizando SQL ou SPARQL, ou unificada, através de um mecanismo que combina os modelos.

(iii) Incorporação de outros modelos ao relacional: Os principais fabricantes de sistemas gerenciadores de banco de dados comerciais também estão investindo esforços para contemplar a integração dos modelos, introduzindo estruturas nativas de grafos dentro de esquemas relacionais: o *Oracle Spatial and Graph* inclui um conjunto avançado de funcionalidades para tratar dados espaciais e analíticos e aplicações sociais e semânticas com grafos [10]. O DB2 RDF [4] é uma proposta para armazenamento e recuperação de informações em grafos RDF no banco relacional DB2 da IBM.

2.2. Arquitetura com Módulo de Integração

Esta arquitetura integra dois ou mais modelos de bancos de dados, mantidos em sua forma nativa, através de um módulo integrador independente, que provê uma interface de acesso unificado aos modelos, utilizando uma terceira estrutura de dados e de consulta como suporte. Cada banco de dados de origem possui um *wrapper*, com regras de mapeamento e de tradução da linguagem nativa para a unificada. Embora bastante custosa para ser implementada, essa arquitetura possibilita a interação com as diversas fontes de dados através de interface e linguagem únicas.

Seguindo esta arquitetura, o projeto *Garlic* [5] define a *Garlic's object query language*, uma extensão orientada a objetos da linguagem SQL. As consultas são distribuídas nas diversas bases e os resultados são posteriormente recuperados e agrupados. O *Garlic* possui um repositório de metadados com as informações do esquema, regras de conversão e semântica entre as diversas fontes, permitindo que novas fontes de dados sejam integradas dinamicamente. O projeto D2RQ [2] propõe a integração de ontologias RDF com bases relacionais, através de um módulo de integração que cria um grafo RDF virtual somente para consulta. A linguagem de mapeamento do D2RQ realiza as traduções necessárias e gera as sentenças SQL ou SPARQL correspondentes.

2.3. Arquitetura em Camadas

Nesse tipo de arquitetura, um modelo de banco de dados é implementado como uma aplicação que opera sobre outro modelo, tornando o modelo da camada inferior dependente do modelo implementado na camada superior. O acesso aos dados de ambos os modelos passa a ser exclusivamente realizado pelo modelo da camada superior. Em [14], um *Information Retrieval System* (IRS) é integrado com um banco de dados orientado a objetos, o VODAR. Os documentos são armazenados e indexados no IRS e o VODAR é implementado no topo da arquitetura, encapsulando as funcionalidades do IRS e mantendo uma representação abstrata da estrutura interna dos documentos. Todo o acesso ao IRS é realizado indiretamente a partir de consultas realizadas no VODAR.

Em [15], encontramos uma arquitetura que apresenta um IRS integrado a um banco de dados orientado a colunas, o MonetDB. A arquitetura é organizada da seguinte forma: o IRS como interface do usuário com o sistema, representando documentos, formulando consultas e classificando processos; a álgebra de objetos, denominada MOA, que provê a funcionalidade de gerenciar a estrutura lógica dos documentos; e, por fim, na base da arquitetura, o MonetDB, acessado pelo MOA.

2.4. Análise Comparativa das Arquiteturas

Inspirados no *framework* criado pelo W3C RDB2RDF Incubator Group [12], classificamos as arquiteturas e trabalhos apresentados anteriormente na Tabela 1, considerando os seguintes critérios: (a) Tipo de mapeamento; (b) Materializa o modelo que se integra com o relacional?; (c) Linguagem de acesso aos dados; (d) Volume de modificações nos sistemas, esquemas e dados atuais ao adotar a abordagem; (e) Esforço de implementação; (f) Coexistência de modelos nativos; e (g) Possibilidade de integrar mais de uma base de dados relacional.

Tabela 1: Análise Comparativa das Arquiteturas e Trabalhos

Arquitetura: Trabalhos	(a)	(b)	(c)	(d)	(e)	(f)	(g)
Mapeamento: -	Manual	N	SQL	Baixo	Baixo	N	N
Extensão: [6], [7]	Manual	N	SQL	Baixo	Baixo	N	N
Extensão: [3]	Manual e Automático	N	SQL / SPARQL	Baixo	Médio	S	S
Incorporação: [10], [4]	Semiautomático	S	SQL / SPARQL	Alto	Médio	S	N
Mód. Integração: [5]	Manual	S	Própria	Baixo	Alto	S	S
Mód. Integração: [2]	Manual e Automático	N	SQL / SPARQL	Baixo	Médio	S	N
Camadas: [14], [15]	Manual	S	Camada Superior	Alto	Alto	S	S
Nossa proposta	Manual e Automático	S	SQL / Cypher	Baixo	Médio	S	S

3. Arquitetura Híbrida de Integração entre Banco Relacional e de Grafos

O objetivo da proposta apresentada neste trabalho é definir uma arquitetura híbrida de mapeamento e integração entre bases de dados relacionais e de grafos, mantendo a representação nativa dos modelos. Será adotado o modelo de banco de dados de grafos baseado em propriedades. A principal contribuição desta proposta é uma arquitetura híbrida capaz de combinar as estratégias a seguir:

3.1. Mapeamento: Nossa arquitetura irá trabalhar com replicação total ou parcial dos dados do modelo relacional, materializando-os fisicamente na base de dados de grafos. O mapeamento dos dados poderá ser realizado de duas formas: (i) Automática: como estratégia inicial, aplicaremos as regras definidas em [1] no contexto RDF; ou (ii) Manual: permitirá a configuração de quais tabelas e/ou campos deverão ser migrados para o grafo, e qual o mapeamento específico de cada um: nós, propriedades ou arestas (relacionamentos). A configuração do mapeamento manual permite também definir regras para relacionar dados de fontes distintas.

3.2. Implementação do Mapeamento: A implementação do mapeamento será realizada por um serviço que poderá ser agendado em intervalos definidos pelo usuário. O serviço utilizará dados registrados em campos e tabelas específicos, produzidos por gatilhos nas bases relacionais, que terão a função de notificar quaisquer mudanças nos dados. Neste cenário, teremos a convivência integrada de ambos os modelos, mantendo sua representação nativa: os dados dos bancos de dados relacionais continuam a ser acessados por seus sistemas via SQL e os dados do banco de grafos serão acessados através da linguagem *Cypher*, específica do Neo4J, que será adotado nesta arquitetura.

3.3. Consistência dos Dados: Modificações na base de grafos: será possível criar novos nós, propriedades e arestas no banco de dados de grafos, cuja existência não será replicada para a base relacional; Modificações na base relacional: (i) inclusão: dados inseridos no modelo relacional serão migrados para o banco de grafos com acesso somente para consulta, limitado pela atribuição de um tipo específico "*read-only*" dentro

do banco de grafos; (ii) alteração: primeiramente realizada no modelo relacional e, posteriormente, refletida no grafo, seguindo a estratégia definida em 3.2; (iii) exclusão: o usuário poderá configurar qual política de exclusão será adotada no banco de grafos após a exclusão de um registro no banco relacional: "excluir" em que todas as propriedades, nós e arestas referentes ao registro excluído também serão excluídas, ou "manter" em que os dados serão mantidos no grafo desvinculados do modelo relacional, removendo o tipo "*read-only*" dos dados no grafo.

4. Validação da Arquitetura Proposta

Um protótipo desta arquitetura será implementado e validado na integração das diversas bases relacionais do Xper [13] (apresentado na introdução) a um banco de grafos. O protótipo será utilizado e testado pelo departamento de biologia do MNHN. O projeto pretende também resolver os problemas de compartilhamento das informações vivenciados pelos biólogos, além da exploração dos novos relacionamentos e possibilidade de realização de consultas e inferências, atualmente inviáveis ou ineficientes na arquitetura utilizada.

Etapas a serem validadas: (i) mapeamento: definição de mapeamentos automáticos e manuais entre bases de dados existentes do Xper; (ii) implementação do mapeamento: validação da base de dados de grafos mapeada de acordo com as definições de mapeamento; (iii) consistência dos dados: projeto e execução de lotes de inserção, alteração e exclusão nas bases relacionais e de grafos e verificação de consistência de acordo com as políticas implementadas; e, por fim, (iv) implementação e testes de consultas com grandes volumes de relacionamentos transitivos, executadas nas bases de dados relacionais e na base de dados de grafos, com o objetivo de avaliar os ganhos na utilização dos grafos.

5. Considerações Finais e Resultados Esperados

Existem diversas arquiteturas e abordagens disponíveis para realização da integração de dados entre modelos distintos e, a partir da análise dessas abordagens e do problema apresentado pelo MNHN, definimos o desafio desta pesquisa: criar uma arquitetura híbrida de integração entre bancos de dados relacionais e de grafos, com baixo impacto na estrutura atual, explorando as vantagens de cada modelo nativo nas operações de gerenciamento e recuperação da informação. Em referência aos trabalhos relacionados, nossa arquitetura se difere por ser capaz de combinar as seguintes estratégias: mapeamento manual/automático; materialização de mapeamentos; possibilidade de atualização e consistência nas operações executadas em ambos os modelos.

Dentre os trabalhos futuros desta pesquisa, será investigada a conexão das bases com fontes externas na forma de grafos, como o sistema GeoNames[2], que identifica unicamente todas as localidades do planeta, e a ontologia Gene Ontology[3], uma base de bioinformática que possui informações sobre genes e proteínas (vide Figura 1). Outro aspecto bastante relevante para pesquisa será a criação de uma interface para o usuário, que facilite a realização de consultas complexas envolvendo algoritmos de grafos.

[2] http://www.geonames.org/
[3] http://www.geneontology.org/

Referências

[1] Berners-Lee, T. (1998). Relational Databases on the Semantic Web. Retrieved June 10, 2014, from http://www.w3.org/DesignIssues/RDB-RDF.html

[2] Bizer, C., & Cyganiak, R. (2007). D2RQ - Lessons Learned. *W3C Workshop on RDF Access to Relational Databases*, 1–5.

[3] Blakeley, C. (2007). Virtuoso RDF Views Getting Started Guide. Retrieved June 10, 2014, from http://www.openlinksw.co.uk/virtuoso/Whitepapers/pdf/ Virtuoso_SQL_ to_RDF_Mapping.pdf

[4] Briggs, M. (2012). DB2 NoSQL Graph Store What, Why & Overview. Retrieved February 27, 2014, from https://www.ibm.com/developerworks/community/blogs/ nlp/resource/ DB2_NoSQLGraphStore.pdf?lang=en

[5] Carey, M. J., Haas, L. M., Schwarz, P. M., Arya, M., Cody, W. F., Fagin, R., Flickner, M., Luniewski, A.W., Niblack, W., Petkovic, D., Thomas, J., Williams, J.H., Wimmers, E.L. (1995). Towards heterogeneous multimedia information systems: the Garlic approach. *Proceedings RIDE-DOM'95*, 124-131.

[6] Goldberg, R. N., & Jirak, G. A. (1993). Relational database management system and method for storing, retrieving and modifying directed graph data structures. *United States Patents*. Retrieved February, 25, 2014, from http://www.google.com /patents/US5201046

[7] Gutiérrez, A., Pucheral, P., Steffen, H., & Thévenin, J. (1994). Database Graph Views: A Practical Model to Manage Persistent Graphs. *VLDB*, *33*(1), 1–20.

[8] Hecht, R., & Jablonski, S. (2011). NoSQL evaluation: A use case oriented survey. *2011 International Conference on Cloud and Service Computing*, 336–341.

[9] Kleppmann, M. (2009). Should you go Beyond Relational Databases? Retrieved March 01, 2014, from http://blog.teamtreehouse.com/should-you-go-beyond-relational-databases

[10] Oracle Corportation. (2013). Oracle Spatial and Graph Developer's Guide. Retrieved February 26, 2014, from http://docs.oracle.com/cd/E16655_01/ appdev.121/e17896.pdf

[11] Raghavan, S., & Garcia-Molina, H. (2001). Integrating Diverse Information Management Systems: A Brief Survey. *Technical Report. Stanford.*

[12] Sahoo, S. S., Halb, W., Hellmann, S., Idehen, K., Jr, T. T., Auer, S., Sequeda, J. & Ezzat, A. (2009). A survey of current approaches for mapping of relational databases to RDF. *W3C RDB2RDF Incubator Group.*

[13] Ung, V., Dubus, G., Zaragüeta-Bagils, R., & Vignes-Lebbe, R. (2010). Xper2: introducing e-taxonomy. *Bioinformatics (Oxford, England)*, *26*(5), 703–4.

[14] Volz, M., Aberer, K., Böhm, K., & GMD-IPSI, D. (1996). An oodbms-irs coupling for structured documents. *IEEE Data Eng. Bull.*, 34–42.

[15] Vries, A. De, & Wilschut, A. (1999). On the integration of IR and databases. *Proceedings of the 8th IFIP 2.6Working Conferene on Database Semantics*.

paper:27

Auto Escalonamento Proativo para Banco de Dados em Nuvem

Victor A. E. Farias
Orientadores: Flávio R. C. Sousa, Javam C. Machado

[1] Mestrado e Doutorado em Ciência da Computação (MDCC)
Universidade Federal do Ceará (UFC) – Fortaleza, CE – Brasil

victorfarias@lia.ufc.br {sousa,javam}@ufc.br

Nível: Mestrado
Ingresso: Agosto 2013
Defesa de proposta: Agosto 2014
Conclusão prevista: Agosto 2015
Etapas Concluídas: Revisão da literatura, definição do problema e solução
Próximas Etapas: Implementação e avaliação

Resumo. *A computação em nuvem é uma tendência recente de tecnologia cujo objetivo é proporcionar serviços sob demanda com pagamento baseado no uso. Neste ambiente, a elasticidade, implementada por meio do auto escalonamento, permite adicionar e remover recursos, sem interrupção e em tempo de execução para lidar com a variação da carga, melhorando a qualidade de serviço e reduzindo os custos associados. Como a maioria das aplicações em nuvem é orientada a dados, sistemas de gerenciamento de banco de dados (SGBDs) são componentes fundamentais na pilha de software da nuvem. Entretanto, implementar o auto escalonamento para SGBDs em nuvem é complexo, visto que é necessário realizar predições com um intervalo de tempo aceitável para a adição ou remoção de recursos. Este artigo propõe uma estratégia de auto escalonamento proativo para SGBDs em nuvem baseada na análise de séries temporais. Esta estratégia permite implementar a elasticidade para SGBDs, melhorando a qualidade de serviço e reduzindo os custos.*

Palavras-chaves*: Computação em nuvem, gerenciamento de dados em nuvem, auto escalonamento, predição, qualidade de serviço.*

1. Introducão

A computação em nuvem é um paradigma de computação orientado a serviços extremamente bem sucedido que revolucionou a forma como a infraestrutura de computação é disponibilizada e utilizada. Escalabilidade, elasticidade, pagamento baseado no uso são as principais razões para o sucesso na adoção generalizada de infraestruturas em nuvem. Como a maioria das aplicações em nuvem é orientada a dados, sistemas de gerenciamento de dados (SGBDs) são componentes fundamentais na pilha de software da nuvem [Sousa et al. 2010].

Neste ambiente, a elasticidade permite adicionar e remover recursos, sem interrupção do serviço e em tempo de execução para lidar com a variação da carga. Dessa forma, um sistema elástico deve adicionar ou remover a quantidade certa de recursos, já que escassez de recursos implica em comprometimento da qualidade de serviço (QoS) e recursos em excesso ocasionam custos desnecessários. Em geral, a elasticidade é implementada por meio de estratégias de *auto scaling* ou auto escalonamento para melhorar a qualidade e reduzir os custos. Contudo, implementar elasticidade para SGBDs é um desafio, visto que estes sistemas precisam migrar o estado atual para um novo recurso quando este é adicionado ao ambiente e assim, faz-se necessário identificar alterações no sistema com um intervalo de tempo aceitável para a adição ou remoção de recursos [Cecchet et al. 2011].

Como o ambiente em nuvem é um ambiente altamente dinâmico, pode ocorrer picos ou depressões no volume da carga de trabalho, é fundamental dispor de mecanismos automáticos de escalonamento dos recursos da nuvem. Os métodos de auto escalonamento podem ser divididos em duas categorias: *Reativo*, quando o sistema reage ao estado atual da carga de trabalho usando técnicas baseadas em regras estáticas; *Proativo ou preditivo*, quando o sistema antecipa o comportamento da carga de trabalho e toma a ações *a priori* baseadas em diferentes técnicas, tais como teoria de filas ou análise de séries temporais [Lorido-Botrán et al. 2012].

Para tratar estas questões, este trabalho apresenta uma estratégia de auto escalonamento proativo para SGBDs em nuvem. Esta estratégia baseia-se na análise de série temporal sobre as métricas de monitoramento para prever o estado futuro da carga de trabalho e em métodos de regressão para calcular a quantidade de recursos necessários com o objetivo de melhorar a qualidade de serviço e reduzir os custos.

2. Contribuições

A principal contribuição deste trabalho é uma estratégia de auto escalonamento proativo para um serviço de banco de dados em nuvens computacionais. Esta estratégia é baseada em um processo de predição de carga de trabalho sobre este serviço por meio de uma técnica de análise de séries temporais com suavização do sinal. Com isso, implementa-se a elasticidade para serviços de banco de dados em nuvem, melhorando a qualidade de serviço e reduzindo os custos.

3. Definição do Problema

Em seu sentido mais amplo, este trabalho está relacionado com o problema de auto escalonamento de banco de dados no ambiente de computação em nuvem. Tal domínio de problema, claramente extenso, é restringido a partir de um conjunto de características.

Considere um serviço de banco de dados no ambiente de computação em nuvem fornecido por um provedor e acessado por clientes. Este serviço possui uma QoS especificada por meio de um acordo de nível de serviço (SLA) e uma carga de trabalho associada, definida por uma taxa de transações variável ao longo do tempo enviada pelos clientes do serviço. Para fornecer este serviço, o provedor utiliza uma infraestrutura com máquinas virtuais e réplicas de banco de dados de acordo com a carga de trabalho. As máquinas são caracterizadas por recursos de CPU, memória, disco e um custo associado por hora de utilização.

O problema deste artigo consiste em: *Como melhorar a qualidade para o serviço de banco de dados de acordo com a carga de trabalho corrente enquanto utiliza os recursos eficientemente?*

4. Fundamentação Teórica

O presente trabalho se fundamenta, principalmente, em conhecimentos existentes em dois pontos: predição e escalonamento. Estes pontos são descritos sucintamente abaixo.

4.1. Predição

Abordagens de auto escalonamento podem ser divididas em duas categorias: reativa e proativa. A abordagem reativa é baseada em regras estáticas, como, por exemplo, um recurso é adicionado quando a média de utilização da CPU excede 80% por um período contínuo de 2 minutos ou remove um recurso quando a utilização da CPU é menor do que 20% pelo mesmo período.

Já as abordagens proativas propõem soluções que executam ações antes que ocorram situações desfavoráveis ao sistema, como uma degradação da QoS do sistema, por exemplo. Estas abordagens são baseadas em teoria do controle, aprendizado por reforço, teoria das filas e análise e predição de séries temporais, sendo que somente esta última pode ser classificada como puramente proativa. Os métodos de predição para séries temporais tentam determinar as características do sinal para tentar predizer os valores futuros da variável de estudo. Uma série temporal, ou sinal, é uma sequência ordenada de valores de uma variável em espaços de tempos igualmente espaçados.

Existem técnicas amplamente utilizadas para auxiliar no processamento de séries temporais, tais como ARMA e *AutoRegressive Integrated Moving Average* (ARIMA), que permite descrever de maneira satisfatória séries estacionárias e não-estacionárias. Com isso, pode-se modelar as diversas variações na carga de trabalho decorrentes do ambiente de computação em nuvem. É importante ressaltar que os outros modelos *random-walk*, *random-trend*, auto regressivos e *exponential weighted moving averages* são casos especiais da técnica ARIMA.

Para situações nas quais é necessário aumentar a acurácia da predição ou remover características indesejadas do sinal, pode-se aplicar técnicas que permitem suavizar o sinal, por exemplo, o algoritmo de *Scale-space* [Santos et al. 2013]. Este algoritmo permite remover situações de picos ou depressões pontuais e não relevantes da carga de trabalho, melhorando a predição.

4.2. Auto Escalonamento

As abordagens de predição descritas anteriormente disparam ações para adicionar novos recursos quando o sistema não consegui lidar com a carga de trabalho atual ou remover

recursos desnecessários. Essas ações têm como objetivo melhorar a QoS ou reduzir os custos quando recursos subutilizados são removidos. Para computar a quantidade aproximada de recursos a serem utilizados para uma dada carga de trabalho, é necessário relacionar o estado do sistema com a QoS definida para o mesmo.

Uma forma de relacionar os recursos e a QoS é modelando-os como um problema de regressão. Métodos de regressão pertencem à classe de algoritmos supervisionados de aprendizado de máquina. Eles têm a capacidade de determinar o efeito casual quantitativo de um conjunto de variáveis, chamadas de variáveis independentes, sobre uma variável alvo, chamada de dependente. Para tanto, utiliza-se um conjunto de treinamento, que consiste de um conjunto de amostras sobre as variáveis independentes etiquetadas com o valor correspondente da variável dependente. A partir do conjunto de treinamento, um método de regressão gera uma função, ou modelo, que associa as variáveis independentes à variável dependente. Entre os métodos de regressão, destaca-se os modelos *Support Vector Regression* (ϵ-SVR) e regressão linear múltipla (RLM). Vale mencionar que o ϵ-SVR também pode ser utilizado em predição de análises temporais.

5. Solução Proposta

Este trabalho propõe uma estratégia de auto escalonamento proativo para implementar elasticidade para serviço de banco de dados em nuvem. Esta estratégia consiste de dois módulos: (1) *Módulo de Predição*, com o objetivo de mensurar oscilações futuras na carga de trabalho (2) *Módulo de Auto Escalonamento*, para estimar e escalar o volume de recurso necessário para atender as oscilações na carga de trabalho. Cada um destes módulos é descrito a seguir.

5.1. Módulo de Predição

O módulo de predição baseia-se na análise da série temporal ou sinal. Para tanto, será usada o número de transações que o sistema recebe por unidade de tempo (hora, minuto ou segundo) como métrica a ser predita. O ϵ-SVR e o ARIMA serão utilizados para realizar a predição a partir de uma janela de observação ou conjunto de treinamento. Esta janela permite definir o intervalo que deve ser observado para realizar a predição, por exemplo, observar a série temporal por trinta minutos para realizar uma de cinco minutos.

É importante observar que diminuindo a janela de observação, pode-se perder características do sinal, comprometendo a qualidade da predição. Por outro lado, se a janela de observação for muito larga, pode-se captar características muito antigas do sinal que já não correspondem ao atual estado do sistema. Adicionalmente, quanto maior a largura da janela de predição, maior será o erro estimado para a predição. Assim sendo, deve-se ajustar a janela de predição para que seja possível predizer em tempo hábil se um dado recurso deve ser alocado, ou, pelo menos, deve ser rápido o suficiente para amenizar os efeitos negativos das mudanças na carga de trabalho. Essas questões são relavantes para SGBDs em nuvem, visto que estes sistema são *stateful* e precisam migrar o seu estado quando um novo recursos é adicionado. Isso implica em uma predição capaz de fornecer ações com um intervalo de tempo aceitável para realizar a migração dos dados [Cecchet et al. 2011].

Dado que o ambiente de nuvem é altamente dinâmico, podem ocorrer oscilações no sinal da carga de trabalho que não sejam significativas. Por exemplo, se ocorrer um

pico no sinal com duração de dois segundos, não se faz necessário tomar nenhuma ação, visto que o tempo de alocar um novo recurso tem duração maior e este recurso não seria utilizado caso fosse alocado. Para tanto, pode-se utilizar a técnica de suavização de sinal denominada *Scale-space*, que permite tratar picos e depressões que não seja significativo como descrito em [Santos et al. 2013].

Uma vez que um sinal for tratado pelo *Scale-space*, será executado o ϵ-SVR e o ARIMA para mensurar o comportamento da carga de trabalho. Como serão utilizadas as predições obtidas pelos dois métodos, será escolhida qual delas deve-se levar em consideração para tomar a ação de escalonamento. Pretende-se adotar a seguinte política: o método que fornecer o menor erro na última iteração será levado em consideração na iteração atual. Para medir o erro, será utilizado o erro quadrático médio comparando a predição de uma janela de predição em relação ao seu valor atual.

5.2. Módulo de Auto Escalonamento

O módulo de auto escalonamento tem como objetivo calcular e escalar a quantidade de recursos que devem ser adicionados ou removidos de acordo com as variações na carga de trabalho. Para realizar esta tarefa, este recebe informações do módulo de predição, assim como informações e métricas sobre o estado do sistema fornecidas por um serviço de monitoramento.

Com base nestas informações, o módulo de auto escalonamento deverá definir a quantidade de recursos necessária para melhorar o QoS e reduzir os custos. Para isso, este deverá estimar o número de violações de SLA a partir um dado estado do sistema, o que pode ser resolvido como um problema de regressão, visto que o valor a ser estimado é um valor numérico. Em seguida, será utilizada uma abordagem baseada em nosso trabalho anterior [Farias et al. 2013]. Serão testados dois métodos de regressão: ϵ-SVR e RLM.

Visto que cada método de regressão tem parâmetros específicos, será adotado um processo de parametrização para maximizar a acurácia do modelo final gerado. Para esse fim, o uso da técnica de *Grid Search* auxiliará a encontrar a melhor combinação de parâmetros. Esta técnica precisa de uma métrica para avaliar a qualidade de um dado modelo. Para medir a qualidade de um dado modelo, será utilizada a técnica *K-fold Cross Validation* (K-fold CV) com o erro quadrático médio como função de erro.

Com este processo será possível estimar a quantidade de máquinas necessárias para uma determinada carga de trabalho, melhorando a qualidade de serviço. Com o objetivo de decidir a quantidade mínima recursos necessários para melhora a QoS, pode-se usar o algoritmo *Hill Climbing*. Com isso, a estratégia proposta permitirá encontrar uma relação satisfatória entre a qualidade e a quantidade de recursos utilizados, visto que cada recurso tem um custo associados e este é um ponto importante no ambiente de nuvem, principalmente considerando SGBDs com grandes volumes de dados.

6. Estado Atual do Trabalho

Algumas etapas definidas no início do trabalho já foram concluídas. O trabalho foi inicialmente dividido em quatro etapas. A primeira, já concluída, constituiu-se de uma revisão bibliográfica para obtenção do embasamento teórico necessário: gerenciamento de dados em nuvem, elasticidade, auto escalonamento e predição. Na segunda etapa foi realizado um estudo comparativo entre técnicas de predição com o propósito de detectar qual destas

seriam mais apropriados para auxiliar no auto escalonamento. Baseado neste estudo, foi escolhido um conjunto de técnicas descritas em [Farias et al. 2013].

Na terceira etapa, ainda a ser realizada, serão implementadas as técnicas para a construção da estratégia proposta, que será testada em uma nuvem computacional. E por fim, na quarta e última etapa, será realizada a avalição desta estratégia por meio de um estudo de caso.

7. Avaliação Experimental

Para a avaliação a estratégia proposta será utilizada a nuvem da *Amazon Web Services* (AWS) e o *framework OLTPBenhmark* [Difallah et al. 2013] para gerar diferentes cargas de trabalho de banco de dados, tais como *TPC-C*, *Wikipedia*, *Twitter* e *YCSB*. Serão executados experimentos variando a quantidade de requisições dos usuários ao longo de um intervalo de tempo de tal sorte a termos cenários onde cada faz-se necessário a adição ou remoção de recursos. Pode-se verificar diferentes métricas, tais como a melhoria da qualidade por meio da redução no número de violações de SLA, erro aceitável de predição, provisionamento de recursos eficiente e aspectos de custos.

8. Trabalhos Relacionados

Em [Cecchet et al. 2011] é apresentado Dolly, um sistema para provisionamento dinâmico de réplicas de banco de dados em nuvem. Este sistema foca na virtualização e tem como base utilizar clonagem e *snapshots* de máquinas virtuais. Dolly utiliza funções de custo definidas pelo usuário para otimizar a sobrecarga de adicionar ou remover réplicas, o que melhora o desempenho do sistema. Contudo, o sistema Dolly não possui estratégias de predição ou auto escalonamento.

Em [Minhas et al. 2012], os autores propõem uma estratégia para minimizar o custo de reconfiguração de um serviço de banco de dados. Esta estratégia trabalha com dados particionados e replicados para implementar a elasticidade. Eles também destacam, de forma manual, os efeitos proporcionados na vazão do serviço de banco de dados em virtude do escalonamento. Contudo, não apresentam técnicas de predição e auto escalonamento.

O trabalho [Heinze et al. 2014] trata o problema de auto escalonamento usando técnicas reativas baseadas em regras estáticas e técnicas proativas baseadas em métodos de aprendizado por reforço. Este trabalho mostrou que essas técnicas são adaptativas em relação a mudança da carga de trabalho, são configuráveis pelo usuário final, além de serem independentes da natureza da carga de trabalho. A estratégia de escalonamento desse trabalho considera apenas a adição ou remoção de uma máquina virtual.

O artigo [Guabtni et al. 2013] apresenta uma estratégia para otimizar o tempo de processamento de consultas para banco de dados em nuvem usando a estratégia de replicação total. Para tanto, a abordagem se baseia principalmente no particionamento da consulta de forma balanceada considerando a distribuição dos dados. Para a elasticidade e o balanceamento de carga, a estratégia utiliza o serviço do provedor de infraestrutura *Amazon Elastic Load Balancer (ELB)*, que não trata questões de predição, essenciais ao auto escalonamento de serviços de banco de dados.

Em [Fernandez et al. 2014], os autores apresentam uma estratégia de auto escalonamento proativo de aplicações web. Essa estratégia engloba diferentes tipos de recursos e

clientes com requisitos diferentes de QoS e tratamento de picos na carga de trabalho. Eles usam técnicas de análise temporal para tratar a carga de trabalho e árvores binárias para computar a ação de escalonamento a ser tomada. Por outro lado, nossa estratégia utiliza algoritmos de aprendizado de máquina para tomar as ações de auto escalonamento, assim como técnicas de suavização para remover características que não relevantes na carga de trabalho [Santos et al. 2013], o que tente a melhorar resultados da nossa estratégia.

9. Conclusão

Este trabalho apresentou uma estratégia de auto escalonamento proativo para serviços de banco de dados em nuvem. Esta estratégia implementa a elasticidade combinando diferentes técnicas, tais como análise de série temporal com suavização do sinal para a predição e regressão para calcular a quantidade de recursos necessários. Com isso, espera-se implementar uma estratégia do auto escalonamento que permita melhorar a qualidade de serviço de serviços de banco de dados em nuvem, assim como reduzir os custos na utilização deste serviço.

Referências

Cecchet, E., Singh, R., Sharma, U., and Shenoy, P. (2011). Dolly: virtualization-driven database provisioning for the cloud. In *ACM VEE*, pages 51–62. ACM.

Difallah, D. E., Pavlo, A., Curino, C., and Cudré-Mauroux, P. (2013). Oltp-bench: An extensible testbed for benchmarking relational databases. *PVLDB*, 7(4):277–288.

Farias, V. A., Moreira, L. O., Sousa, F. R., Santos, G. A., Maia, J. G., and Machado, J. C. (2013). A machine learning approach for sql queries response time estimation in the cloud. In *Proceedings of the Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 1–6.

Fernandez, H., Pierre, G., and Kielmann, T. (2014). Autoscaling web applications in heterogeneous cloud infrastructures. In *Proceedings of the IEEE International Conference on Cloud Engineering (IC2E)*.

Guabtni, A., Ranjan, R., and Rabhi, F. (2013). A workload-driven approach to database query processing in the cloud. *The Journal of Supercomputing*, 63(3):722–736.

Heinze, T., Pappalardo, V., Jerzak, Z., and Fetzer, C. (2014). Auto-scaling techniques for elastic data stream processing. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, DEBS '14, pages 318–321, New York, NY, USA. ACM.

Lorido-Botrán, T., Miguel-Alonso, J., and Lozano, J. A. (2012). Auto-scaling Techniques for Elastic Applications in Cloud Environments. Research EHU-KAT-IK, Department of Computer Architecture and Technology, UPV/EHU.

Minhas, U. F., Liu, R., Aboulnaga, A., Salem, K., Ng, J., and Robertson, S. (2012). Elastic scale-out for partition-based database systems. In *Proceedings of the 7th International Workshop on Self Managing Database Systems (SMDB)*, pages 281–288. IEEE Computer Society.

Santos, G. A. C., Maia, J. G. R., Moreira, L. O., Sousa, F. R. C., and Machado, J. C. (2013). Scale-space filtering for workload analysis and forecast. In *Proceedings of the IEEE Sixth International Conference on Cloud Computing*, CLOUD '13, pages 677–684. IEEE Computer Society.

Sousa, F. R. C., Moreira, L. O., Macêdo, J. A. F., and Machado, J. C. (2010). Gerenciamento de dados em nuvem: Conceitos, sistemas e desafios. In *Proceedings of the Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 101–130. SBC.

paper:28

Complex Pattern Detection and Specification to Support Biodiversity Applications

Jacqueline Midlej do Espírito Santo[1], **Claudia Bauzer Medeiros**[1] **(advisor)**

[1]Programa de Pós-Graduação do Instituto de Computação
Universidade Estadual de Campinas (UNICAMP)
13.083-852 – Campinas – SP – Brazil

jacqueline.santo@students.ic.unicamp.br, cmbm@ic.unicamp.br

Level: Master's degree
Admission: March 2013
Qualifying Exam: 29^{th} November 2013
Expected Conclusion: March 2015
Completed Steps: Bibliographic research; Event and pattern specification for biodiversity context.
Future Steps: Extend Event Processing Language (EPL) from the Esper engine; Adapt a framework to detect complex patterns using the language extended; Evaluate the framework.

Abstract. *Biodiversity scientists often need to define and detect scenarios of interest from data streams delivered from meteorological sensors. For example, scenarios such deforestation or forest fire need to be detected in order to reduce impacts over the environment. Such data streams are characterized by their heterogeneity across spatial and temporal scales, which hampers detection of events and construction of scenarios. To help scientists in this task, this work proposes the use of the theory of Complex Event Processing (CEP) to define and detect complex event patterns in this context. The two main contributions focus on the specification of events and patterns for the biodiversity context and on the mechanism to detect these patterns. The first one requires to extend an Event Processing Language (EPL) to include spatial relationships in the pattern. The second one will extend Koga's framework [Koga 2013], which integrates heterogeneous data sources, with the detection of complex patterns. This paper extends the short paper accepted for the Brazilian Workshop e-Science (BreSci) 2014 with the specification for events and patterns.*

Keywords: *Complex Pattern Detection, Complex Pattern Specification, Multi-scale Data, Environmental Variable, Spatial and Temporal Relationship.*

1. Introduction

Biodiversity broadly means the abundance, distributions and interactions across genotypes, species, communities, ecosystems and biomes. Countless problems in biodiversity studies require data collected and analyzed at multiple space and time scales, correlating environmental variables, living beings and their habitats [Hardisty and Roberts 2013]. For example, environmental monitoring requires data from environmental variables mainly generated by meteorological sensors. When this monitoring involves animals, there are other sensors for motion and sound. An open problem in this context is how to specify and detect scenarios of interest (as climate change, deforestation or water pollution) from environmental variables, in multiple scales, to help scientists analyze phenomena and correlate results with data collected on the field.

To help solve the problem, this work proposes to use Complex Event Processing (CEP) to process data streams of meteorological sensors. The main goal of CEP is to detect event patterns in near real-time, in order to signal situations of interest [Sen et al. 2010]. Our idea is to allow researchers specify and combine events that characterize such situations and detect their occurrence in the context of biodiversity applications.

The paper presents two main contributions: (i) The specification of events and complex patterns for biodiversity context. This context requires patterns composed by combining events with temporal and spatial relationships (the latter unavailable in CEP). Therefore, this paper proposes to extend an Event Processing Language (EPL) to support these particulars. (ii) The development of a mechanism to detect complex patterns. For this purpose, the paper extends the framework used to integrate heterogeneous data sources proposed by [Koga 2013], which deal with simple events. This paper extends the short paper accepted for the Brazilian Workshop e-Science (BreSci) 2014 with the specification for events and patterns in the biodiversity context.

2. Theoretical Foundations

In CEP, the word *event* means the programming entity that records an occurrence of something in a domain [Etzion and Niblett 2010]. Events are classified into primitive and complex. Primitive events represent an occurrence at a given place and time. Complex events are formed by combinations of primitive or complex events. Primitive events represent observations outside from the event system, while complex events represent events defined inside the system [Pietzuch et al. 2004]. Examples of primitive events from environmental variables are measurements about temperature, barometric pressure and wind speed. Examples of complex events are cold front, fire, or poor water quality.

The main task of CEP is to detect complex events, in order to identify within a set of events those that are significant to an application domain. Such detection occurs through matching events with event patterns (patterns for short). Patterns represent models of scenario of interest composed by specification of events and their relationships [Etzion and Niblett 2010]. Patterns can be defined on a hierarchy of events in which the patterns specify how the highest level events are formed by inferences from lower level events. Their composition are defined by Event Processing Languages (EPL). Sometimes, the literature refers to composition of patterns using the term complex event, and other times, the term complex pattern. This paper discusses the subject regardless of the term

used in the literature. One of the contributions of the dissertation is the uniformization of related work, which misses term such as event and pattern, with respect to a wide variety of similar operations and constructs.

3. Related Work

Depending on the context, the structure and components of events can change. [Koga 2013] defines 4 attributes to specify events in environmental applications: measured-value, nature, spatial-variable, and timestamp. However, this representation only describes primitive events. Complex events must define relationships between events. For example, [Sen et al. 2010] represents complex events in business applications by a model based on semantics which, besides the basic attributes, references to operators that connect events.

Event Processing Languages, used to specify patterns, are mainly defined using approaches based on logics (logic-based) or automata (automata-based). Many research efforts are concerned with defining more powerful languages. For instance, [Barga and Caituiro-Monge 2006] describe the language *Complex Event Detection and Response* (CEDR) for expressing patterns that filter, generate and correlate complex events in business applications.

Logic-based patterns are defined as combinations of logic predicates on events. Examples using this approach are [Motakis and Zaniolo 1995] and [Obweger et al. 2010]. The first define a model for active databases in which the pattern composition is described by $Datalog_{1S}$ rules. For biodiversity applications, our target, this model is limited because $Datalog_{1S}$ only supports one temporal operator. Scenarios that have more complex temporal relationships and/or have spatial relationships cannot be represented. On the other hand, [Obweger et al. 2010] do not limit the predicate to the use of specific operators. In addition, their model allows users to compose hierarchical patterns using an interface that abstracts the definition of sub-patterns.

In automata-based approaches, regular expression operators are used to compose patterns. This approach limits the temporal relationships to the notion of precedence and does not support spatial operators. Examples of papers in this line are [Pietzuch et al. 2004] and [Agrawal et al. 2008]. The first one performs event detection in distributed systems. The latter focuses on improving the runtime performance of pattern queries over event streams, for business applications.

4. Contributions

This work has two main parts. The first one aims at formalizing specification of events and patterns on the biodiversity context, inspired by proposals applied to different domains (e.g., [Etzion and Niblett 2010, Barga and Caituiro-Monge 2006, Sen et al. 2010]). It must: allow hierarchical events composition, such as [Sen et al. 2010]; combine heterogeneous data sources, such as [Koga 2013]; and consider the place where the event occurs, such as [Koga 2013]; It must also extend the semantics of operators to support spatial and temporal multiscale data. This specification can express biodiversity scenarios of different complexity, from excessive rain to situations combining hydrographic data with vegetation and relief data.

The second contribution of this work is the development of a mechanism that allows patterns composition and detection to assist biodiversity applications. This step extends the work of [Koga 2013], which allows integrating data from heterogeneous sources, but it is limited to the detection of primitive event patterns. Figure 1 illustrates the architecture, horizontally drawn, of the extended framework. Kogas's proposal has two main aspects: the use of Enterprise Service Bus (ESB) to process data streams uniformly and the use of CEP to detect patterns. Environmental data are pre-processed and translated into events, which pass through the ESB and are processed by CEP. We extend this work by providing continuous event feedback into the bus, to allow event composition and detection of complex events.

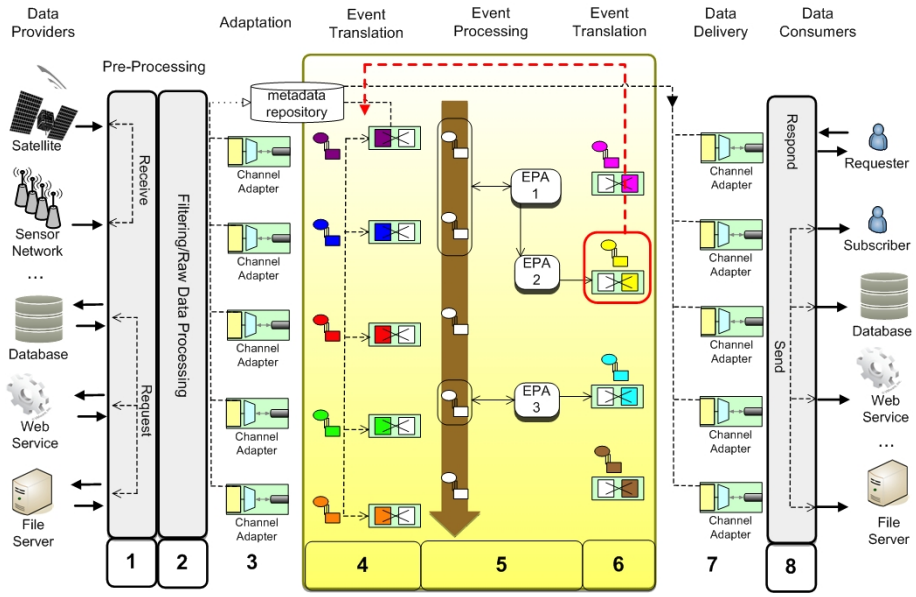


Figure 1. Adapted architecture from [Koga 2013]

From left to right, 1 through 3 filter data according to the goal of the application and encapsulate events into messages. Steps 4 and 5 correspond to the translation of messages into events and their processing by CEP. If a pattern is detected, step 6 encapsulates the matched event into a new message. At steps 7 and 8, the message is sent to the interested user.

Our work complements the architecture adding complex pattern composition and detection, illustrated by the red dotted arrow from step 6 to step 4 in Figure 1. This adaptation provides more representative patterns. The detected composition of events is sent back to the ESB bus, and forwarded back into the pipeline, creating a hierarchical structure. The output of a complex event may become part of more complex compositions, generating composite events at a higher level. Though represented by one small arrow, this extension will require considerable design and implementation efforts.

This framework will be validated over meteorological sensor data combined with hydrographic, topographic and relief data. Meteorological data are provided by Cooxupé, the largest coffee cooperative in the world, from 14 weather stations in cities located in Minas Gerais and São Paulo. The stations continuously collect at least 26 types of measurements, e.g., temperature, humidity and barometric pressure. Each station records all

measurements once a minute. However, the other data sources correspond to records collected at distinct temporal granularities, and distinct geospatial coverage, thereby instantiating the multiscale aspect in time and space. We will create patterns that specify scenarios of interest exploring spatial and temporal relationships among these data. The framework will hierarchically detect data that match a pattern (starting from atomic events upwards). We point out that, as far as we know, event detection approaches using CEP do not take spatial relationships into consideration, nor do they consider handling multiscale data.

5. Ongoing Work

5.1. Biodiversity Event Definition

The definition of events in the biodiversity context requires using data from different sources and scales on time and space. In the following list, we define the set of attributes required to represent primitive events, obtained as part of our research:

- *Id* refers to the attribute to identify a single event.
- *Source* refers to device, database, sensor or application that produces the event. This attribute filters the set of data in which the user is interested. Different sources can measure different variables with varied quality. For example, an expert may know which sensors are in a better location or provide more reliable data.
- *Type* refers to which type of variables were observed or measured. In other words, what this event is about.
- *Value* refers to the value measured from the variable specified by *Type*.
- *UnitOfMeasure* refers to the unit of measure from the *Value* attribute. Events with the same *Type* can be stored with different units of measure due to, for example, an user's cultural habits.
- *Space* refers to the location where the event occurs. It can handle data generated at multiscales on space: a point at space, a line or a polygon.
- *Time*, composed by *StartValidTime* and *EndValidTime*, refers to the time when the event is considered valid for the event system. The meaning of this attribute is adapted from the CEP concept. Originally, CEP uses an interval to represent the time in which the event is happening. Our adaptation provides a way to handle different types of events generated at multiscales on time. Events concerning soil data, for example, do not change quickly, so they may be generated just once a year. Because of that, these events are considered valid for the entire year. On the other side, temperature events have a short valid time and are generated frequently.

These attributes support multiscale data; we believe they are also sufficient to represent events from heterogeneous sources in biodiversity. We point out that we can pre-process data before transforming them into events (steps 1 and 2 from Figure 1). For example, satellite images or animal sounds can be represented by descriptors. These descriptors are formed by a set of measured variables and values, which can become a set of events.

Complex events, however, do not represent a single observation. Thus, they do not keep the *Type* and *Value* attributes from the primitive event schema. They have an *Id*, a *Source* (the event system) and they keep aggregated information about space and time.

The values of *Space* and *Time* of complex event E are computed from the values of the events that compose E (immediate lower level events). *Space* considers the smallest area and *Time* considers the smallest time interval containing all the immediate lower level events. In addition to these events attributes, the complex event has a list of operators and a list of references to events. The first represents the relationships between the events. It is the operator used to aggregate the lower level events into complex event. The later consists of references to the immediate lower level (primitive or complex) events.

In our solution, complex events are defined by hierarchical composition of less complex events. Events are aggregated into complex events using different kinds of relationships. This approach allows the lower level events be traced back from the complex event. In CEP, this backtracking task is called *drill down*. On our specification, each complex event keep information about the events that directly creates it, stopping the drill down at primitive events.

5.2. Biodiversity Pattern Definition

Using the logic-based approach, we define the aspects that a pattern language should consider for the biodiversity context: connectors, quantifiers and operators over event attributes. The connectors are logical operators used on the combination of two or more predicates. The main connectors are $\wedge$ (conjunction), $\vee$ (disjunction) and $\neg$(negation). The main quantifiers are $\forall$ (universal quantifier) and $\exists$ (existential quantifier). Operators should consider the nature of each attribute. Examples of operators over numerical attributes are =, $<$ and $>$. Examples of operator over temporal attributes are [Allen 1983]'s relationships: *before/after, meets/is-met-by, overlaps/is-overlapped-by, finishes/is-finished-by, contains/during, starts/is-started-by* and *equals*. Examples of spatial operators are distance and topological relationships between two objects (*contained-in/contains, overlaps/disjoint, equals* e *touches*).

It is important to notice that biodiversity scenarios need to express relationships between events considering temporal and spatial aspects. We have not found EPL that provide spatial operators in the CEP literature. We aim to extend the EPL from the Esper Engine[1], an open source software, to support spatial operators. This SQL-like EPL has good documentation and is the most complete pattern language found. It supports numerical and temporal operator, besides the connectors and quantifiers required by our definition.

5.3. Running Example

Using the specifications and framework proposed, biodiversity scientists can represent scenarios (as deforestations and forest fires) by complex patterns and detect them. For instance, detecting climate changes as the arrival of a cold front in Campinas involves the monitoring of several environmental variables. A short logic-based pattern for this scenario can be:

$$\exists Et1 | Et1.type = temp \wedge value < 31 \wedge dist(Et1.space, Campinas) < 200km \wedge$$
$$\exists Et2 | Et2.type = temp \wedge value > 54 \wedge touch(Et1.space, Et2.space) \wedge$$
$$\exists Ew | Ew.type = DirWind \wedge value = southwest \wedge overlap(Ew.space, Et1.space) \wedge$$
$$overlap(Et1.time, Et2.time, Ew.time)$$

[1]http://esper.codehaus.org/

This pattern contains composition of event $Et1$ signaling low temperature (cold air mass), "meeting" with $Et2$ signaling high temperature in Campinas (hot air mass), and Ew, which shows the presence of wind carrying the cold front to Campinas. At the framework, the detection process finds events $Et1$ and $Et2$, generating complex event $CE1$ with the operators *touch* and *overlap*. This event is fed back to the bus. Next, $CE1$ and Ew are detected, generating the complex event $CE2$ with the operator *overlap* that confirms the cold front. When $CE1$ and $CE2$ are generated, they form a higher hierarchical level.

6. Conclusions and Future Work

This paper proposes a software framework to help biodiversity scientists to specify and detect scenarios of interest. These scenarios are specified by event patterns. The expressiveness of patterns and events and the handling of multiscale data are considered in their specification. The detection is made by a hierarchical and logic-based approach. Future directions include extending Esper's EPL to support pattern with spatial relationships and extending the [Koga 2013]'s framework to detect the biodiversity scenarios described by this language.

Acknowledgments Work partially financed by FAPESP (2013/02269-7), FAPESP/Cepid in Computational Engineering and Sciences (2013/08293-7), the MSR-FAPESP Virtual Institute (NavScales project), CNPq (MuZOO Project), FAPESP-PRONEX (eScience project), INCT in Web Science, and grants from CNPq.

References

Agrawal, J., Diao, Y., Gyllstrom, D., and Immerman, N. (2008). Efficient pattern matching over event streams. In *ACM SIGMOD*, pages 147–160.

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843.

Barga, R. S. and Caituiro-Monge, H. (2006). Event correlation and pattern detection in cedr. In *EBDT*, pages 919–930.

Etzion, O. and Niblett, P. (2010). *Event Processing in Action*. Manning Publications Co.

Hardisty, A. and Roberts, D. (2013). A decadal view of biodiversity informatics: challenges and priorities. *BMC Ecology*, 13(1).

Koga, I. K. (2013). *An Event-Based Approach to Process Environmental Data*. PhD thesis, Instituto de Computação - Unicamp. Supervisor Claudia Bauzer Medeiros.

Motakis, I. and Zaniolo, C. (1995). Composite temporal events in active database rules: A logic-oriented approach. In *DOOD*, volume 1013 of *LNCS*, pages 19–37.

Obweger, H., Schiefer, J., Kepplinger, P., and Suntinger, M. (2010). Discovering hierarchical patterns in event-based systems. In *SCC*, pages 329–336.

Pietzuch, P., Shand, B., and Bacon, J. (2004). Composite event detection as a generic middleware extension. *IEEE Network*, 18(1):44–55.

Sen, S., Stojanovic, N., and Stojanovic, L. (2010). An approach for iterative event pattern recommendation. In *DEBS*, pages 196–205.

paper:16

Data quality assessment of very large database through visualization system

João Marcelo Borovina Josko[1], **João Eduardo Ferreira**[1] **(Advisor)**

[1]Department of Computer Science
Institute of Mathematics and Statistics (IME)
University of São Paulo (USP)
São Paulo - SP - Brazil

{jmbj,jef}@ime.usp.br

Level: PhD
Enrollment in the program: February, 2011
Qualification Evaluation: September, 2013
Expected Completition: June, 2015
Steps completed: Taxonomy submitted as a paper and preliminary framework structure
Future steps: Full framework and respective case study

Abstract. *Data Quality Assessment outcomes are essential to improve data quality and are required condition to support analytical processes. There are several successful approaches to automate this support to syntactic data defects. In contrast, the dependence of semantic data defects on data context knowledge implies on human supervision. The visualization systems belong to a class of supervised tools that can turn data defects into visual items. However, there is no design support for this purpose. Hence, this paper presents a framework to assist the design of these systems fitting the visual quality assessment of semantic data defects. Such an approach is based on data defects structure, data characteristics and user-centered tasks.*

Quality in Big Data, Visual Data Quality Assessment, Semantic Data Defects, Visualization Design, Systematic Design, Information Visualization

1. Introduction

New technologies enable industry and scientific organizations to collect, store and distribute large databases to address their analytical processes. More than data processing capacity, such a knowledge-intensive processes depend on reliable data to produce useful outcomes. Improving and keeping data quality at desired level requires to reach out an alternative based on numerous methods, techniques, procedures, processes and technological approaches. However, determining which the more effective resources are and how to apply them implies knowing the current data quality state of databases; this is the aim of the Data Quality Assessment process.

Relevant computational models support this process, specially for syntactic data defects which have precise rules, like *Functional Dependency Violation* [Borovina Josko et al. 2014]. These models are based on quantitative

[Alpar and Winkelsträter 2014] or constraint [Liu et al. 2012] functionalities and share a non-interactive approach through data quality evaluation. In others words, they restrict the human role to the interpretation of their outcomes [Dasu 2013].

However, the Data Quality Assessment process strongly depends on data context knowledge since it is impossible to confirm or refute a defect based only on data [Lee et al. 2009], [Dasu 2013]. The context specifies the structure of meaning between data and an environment (e.g., a department of an organization). Hence, human supervision through this process is essential, even more to semantic data defects due to their difficult rule specification (e.g. False Tuple) [Borovina Josko et al. 2014].

Visualization systems belong to a class of supervised approach that combines computational capacities to pattern-finding and semantic distinctions innate to the human beings. There has been a huge literature in regard to design of these systems, such as the relevant works by [Bertin 1983] and [Ware 2004]. However, such literature does not provide adequate support to the design of visualization systems for the tasks of data quality assessment.

Having set the problem, this work presents an approach to assist the design of visualization system for visual data quality assessment of large databases. Such an approach is based on a systematic framework that encodes visualization system properties based on data defects structure and data characteristics. Our hypothesis is: each data defect may be connected to certain visualization system properties which enable the tasks of data quality assessment.

The work reported here is organized as follows: Section 2 introduces some basic foundations related to this work, while Section 3 reviews all related works. Section 4 describes this work's main contributions. Section 5 draws the first results, while Section 6 presents the current work and future directions.

2. Foundations

2.1. Data Quality

Inadequate data quality directly affects the outcomes and costs of different working processes, especially the knowledge-intensive ones [Redman and Blanton 1997]. Moreover, such an inadequacy also leads to enormous effort towards the required needs of relevant projects (e.g., Data Warehouse, Re-engineering) and prevents organization to provide reliable data to customers and partners.

However, the consequences aforementioned correspond to impurities that arise at any point of the data life cycle [Redman and Blanton 1997]. The life cycle is a model that exposes the various activities (data acquisition, maintenance, use, disposal) on data as well as the elements (software, hardware, working processes, people) in charge of these activities. Gathered, they determine multiple ways to affect data quality.

2.2. Data Quality Assessment

The Data Quality Assessment process involves inspecting and understanding prioritized data regions, according to the Data Quality Dimensions (e.g., completeness and accuracy). It is a collaborative process that provides practical inputs for choosing the most suitable alternative to solve inadequate data [Lee et al. 2009].

Besides the data context (Section 1), performing a data quality assessment requires to consider additional key issues (organizational, cultural and technical) to ensure its final cost lower than the benefits [Lee et al. 2009]. Particularly, data defects are relevant technical issues because they occur in different granularities (e.g. attribute, tuple, relationship) and share complex relationships chains that requires assessing all grains to ensure their absence. For all these reasons, automation of the Data Quality Assessment process is essential to its efficiency and efficacy [Rahm and Do 2000], [Dasu 2013].

2.3. Visualization and Visual Analytics

The visualization research area investigates the use of computational resources to synthesize visual interactive metaphors to enable interpretations of special facts within large amounts of data [Ware 2004]. Thus, its purpose is to make the corresponding behavior of data perceptible to the human mind. Historically, visualization has two interrelated disciplines: scientific visualization deals with physical references inherited from data; information visualization is concerned with visual mapping of abstract and non spatial data. This work is naturally related to the latter.

The increasing data volume prevents data analysis to be addressed only by visual or computational models. Visual Analytics represents an interdisciplinary effort (involving human-computer interaction, cognitive science, among others) to combine human, computational and visual communication capacities to enable reasoning in large data volumes [Thomas and Cook 2005]. Gathering the best of all worlds, Visual Analytics is an important support to detect inadequate quality of data.

3. Related Works

Knowledge concerning the design of visualization systems is encoded in different forms that offer distinct levels of depth and primary foci. Due to the huge literature and space restrictions, only certain papers are introduced.

Taxonomies organizes core concepts about visualization system, such as [Shneiderman 1996]. *Guidelines* describes recommendations to design visualizations systems in given conditions. While certain guidelines provide directions for particular issues [Baldonado et al. 2000], others assume a broader picture of visualization systems [Tang et al. 2004]. In contrast, *implementations* offers design examples through the description of visualization systems for data quality assessment. Such implementations reveals certain visual or systemic appeal to address syntactic data defects [Rundensteiner et al. 2007], [Kang et al. 2008], [Kandel et al. 2012].

Reference Models is the most robust support to visualization system design, which is driven by different theoretical perspectives and purpose. Certain models are based on theories as psychophysics [Csinger 1992], visual perception [Ware 2004] or cognitive psychology [Patterson et al. 2014]. In terms of purpose, certain models are concerned with automatic visualization generation [Casner 1991], [Zhu et al. 2009], rules for graphic construction [Bertin 1983], [Wilkinson et al. 2006] or activities involved with visualization design [Card et al. 1999].

Close to this study, the Spatial Data Quality (SDQ) and the Uncertainty Visualization (UV) explore the relationship between the data quality and the decision-making processes [Devillers et al. 2006]. Both research areas systematically describe models and

classifications combining data characteristics, space and time to determine the visual symbols [Potter et al. 2012] or visual variables [Thomson et al. 2005] to expose uncertainties.

Knowledge Encodings	*Main Characteristics*
Taxonomy	- Describes visualization properties. Little support on designing
Guideline	- Does not address the Data Quality Assessment process - Does not provide a systematic approach for designers - Proposals are unrelated, unstructured and susceptible to contradiction
Implementation	- Does not describe how visual data quality assessment was considered - Does not describe how data defects structures was considered - Does not provide a systematic approach for designers - Visualization is a Quality-aware media - Addresses some syntactic defects, mostly
Reference Model (All the others)	- Does not address the Data Quality Assessment process - Driven to a limited repertoire of visual attributes and techniques (Automation)
Reference Model (SDQ and UV)	- Shares a mathematical and statistical basis for data quality assessment - Visualization is a Quality-aware media (expose data uncertainties)
This study's Approach	- Driven to the Data Quality Assessment process - Visualization is media for extracting and correlating relevant information, until gathering a set of evidences to confirm or refute a data defect - Provide a systematic approach for designers - Strongly based on semantic data defects structure

Table 1. Brief Comparison between Approaches (Source: The authors)

Although such literature offers a seminal knowledge, it can not assist the design of visualization systems for assessment of the semantic data defects. Among the reasons (see Table 1), such literature does not address a key element of the Data Quality Assessment process: the *structures of the data defects*. These structures disclose the data behavior and relationships toward which reasoning and actions are directed. In other words, these are the information to be extracted from the visual stimuli for subsequent cognitive processing. Hence, the design must use this knowledge to align visualization systems properties to the demands of the tasks of visual data quality assessment [Patterson et al. 2014].

There has been much literature describing data defects. Certain relevant literature has used the hierarchical model [Rahm and Do 2000] or mathematical formalization [Oliveira et al. 2005] to explain defects in a broad sense. More recently, such defects have also been analyzed using data warehouse [de Almeida et al. 2013] and temporal [Gschwandtner et al. 2012] perspectives. However, an analysis of such literature shows remarkable discrepancy in terminology, nomenclature, description, defects coverage and granularity of defects nomination.

4. Contributions

Due to the related works limits, the contributions of this work are as follow: a more representative defects taxonomy of atemporal and structure data; a framework that systematically encodes data quality assessment characteristics to core visualization systems properties. They represent the first systematic support to the design of visualization systems for visual data quality assessment.

4.1. A Data Defects Taxonomy based on a Formal Framework

The limitations of the data defect literature (see Section 3) make difficult to answer questions about data quality assessment process: *What is the problem structure behind each defect? What is the representative set of defects related to the quality dimensions of accuracy, completeness and consistency? Which defect subset requires human supervision?*

The taxonomy proposed to answer these questions resulted from applying three steps in sequence. The *review step* used a theoretical approach to the identification of data defects through re-examining topics, such as conceptual data modeling, transformation decisions between conceptual and logical models, and relational theory based on [Maier 1983], [Elsmari and Navathe 2010]. This review also determined the terminology and mathematical formalism.

		Granularity				
Nature	*Data Defect Name*	Value	Attribute	Tuple	Relation	Inter-relation
Semantic	Atypical Tuple	-	-	●	-	-
	Contradictory Attribute	-	-	-	-	●
	Contradictory Circular Reference	-	-	-	●	●
	Duplicate Tuples	-	-	-	●	●
	False Tuple	-	-	●	-	-
	Heterogeneous Granularity	-	●	-	-	-
	Heterogeneous Measurement Unit	-	●	-	-	-
	Homonymous Values	-	●	-	-	-
	Incorrect Reference	-	-	-	●	●
	Incorrect Value	●	-	-	-	-
	Missing Reference	-	-	-	●	●
	Missing Tuple	-	-	●	-	-
	Overloaded Tuple	-	-	●	-	-
	Synonymous Values	-	●	-	-	-

Table 2. Classified Data Defects (partial list) [Borovina Josko et al. 2014]

The subsequent step *classified each data defect according to the nature of the efforts involved in its detection*. Semantic nature indicates that a defect diagnosis requires data domain knowledge due to the impossibility of rule delimitation, which eliminates the use of fully computational solutions. Syntactic nature suggests that a defect has precise rules, allowing purely computational models to contribute.

The final step *classified each data defect based on its place or granularity of occurrence*, including attribute value, single attribute, single tuple, single relation or inter-relation (binary relationship), which involves one or more database instances. The resulting classes from the last two procedures are the basis to classify the data defects listed in Table 2.

4.2. A Framework based on visualization system properties for Data Quality Assessment

Only the integration of certain visualization properties can provide appropriate support to the tasks of visual data quality assessment [Ware 2004], [Casner 1991], [Bertin 1983]. However, the related works can not guide the visualization designer to such goal (see Section 3). The systematic framework proposed to address this situation resulted from applying three steps in sequence. In all of them, the interactive visualization prototypes used are based on R environment and its graphical libraries [Chambers 2008].

The first step represents the *framework preliminary formulation*, illustrated at Figure 1. An user-centered task set establishes the visualization system requirements for data assessment purpose. These requirements are mapped to visualization properties (design domain), such as the analytical visual perception [Rodrigues et al. 2007]. The characteristics of these tasks are determined by the problem domain elements. The manipulated

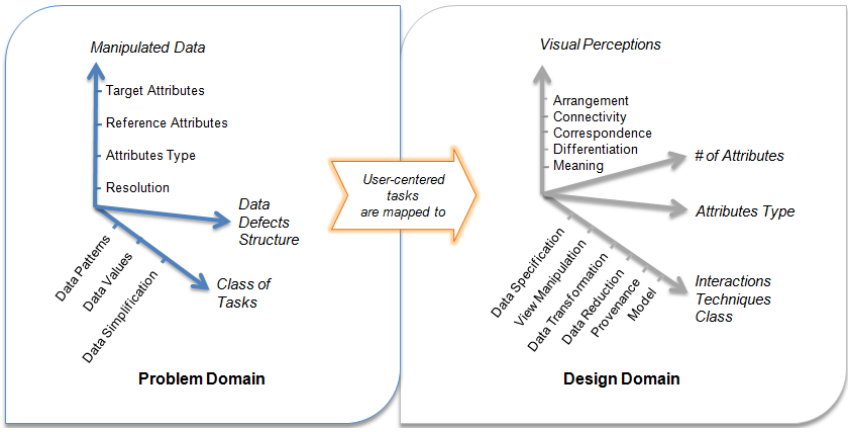


Figure 1. Framework Structure (Source: The authors)

data specify the attributes and data resolution. Data defects are essential to determine kinds of data patterns, relationships among values or granularity to look for through a task, while task class is related to a three-level analytical process: overall, characterization and assessment.

The subsequent step intends to *enhance the framework's structure* through a exploratory case study. Such method is used to comprehend which and how the properties of visualization system may favor (or not) the detection of each semantic defect, according to its structure. The analysis units of this case comprise of visualization techniques with different characteristics (planar or visual) and certain interaction techniques, such as sorting, attributes arrangement, filtering and zooming. The synthetic databases to be used will range from 1000 to 50000 tuples, which about 1% are defective.

The third step is a thoughtful exploration of the *framework in practice*. A case study will show the viability of the framework to assist the design of a interactive visualization prototype capable to support the tasks regarding to the assessment of certain semantic data defect. Such a prototype has visual resolution reduction capacities through filters because it is intended to use a synthetic database with near 50 millions tuples, which about 5% are defective.

Although the framework provides the core guiding for visualization systems design for visual data quality assessment, it does not represent exhaustive rules. To leverage the interpretation of data defects, the framework must treat the cognitive theories and the social collaborative interactions. These issues represent one of the future work directions.

5. First Results

The taxonomy (see Section 4.1) has been submitted to JDIQ (*Journal of Data Quality and Information*). It provides a greater coverage of data defects mathematically formalized and satisfies different data quality assessment needs, as guiding the basic design decisions of supervised systems. The example below is an fragment of such a taxonomy.

> *Definition (Incorrect Value):* Let Σ be the work tape alphabet of an OTM, such that $\Sigma = R(A)$. Let Γ be the oracle tape alphabet of an OTM, such that $\Gamma = \{0, 1\}$. Let A be the OTM oracle that contains all of the attributes of relation R that faithfully represent facts about objects of the interest domain. An attribute has an incorrect value *iff* $\exists a \in R(A) : O^A(a) = 0$.

An incorrect value is an unfaithful or contradictory representation of facts about objects of the interest domain.

Example 1: A customer's birth date is 03/10/1970, but it was represented as 01/01/1980.
Example 2: A customer's name is "Ridley Scott", but it was represented as "Joan Ridley".

In turn, the framework's preliminary result shows the systematic encoding of a core set of suitable visualization system properties, as illustrates the visual assessment of *atypical value defect* in Figure 2. At upper left, the *Task Set* section represents the tasks sequence in regard to such a defect. These tasks are mapped (one by one) to visualization system properties, as seen in the *Mappings* section. The *Final List* section gathers all these properties which are use to design a R interactive prototype using synthetic data (shown at right).

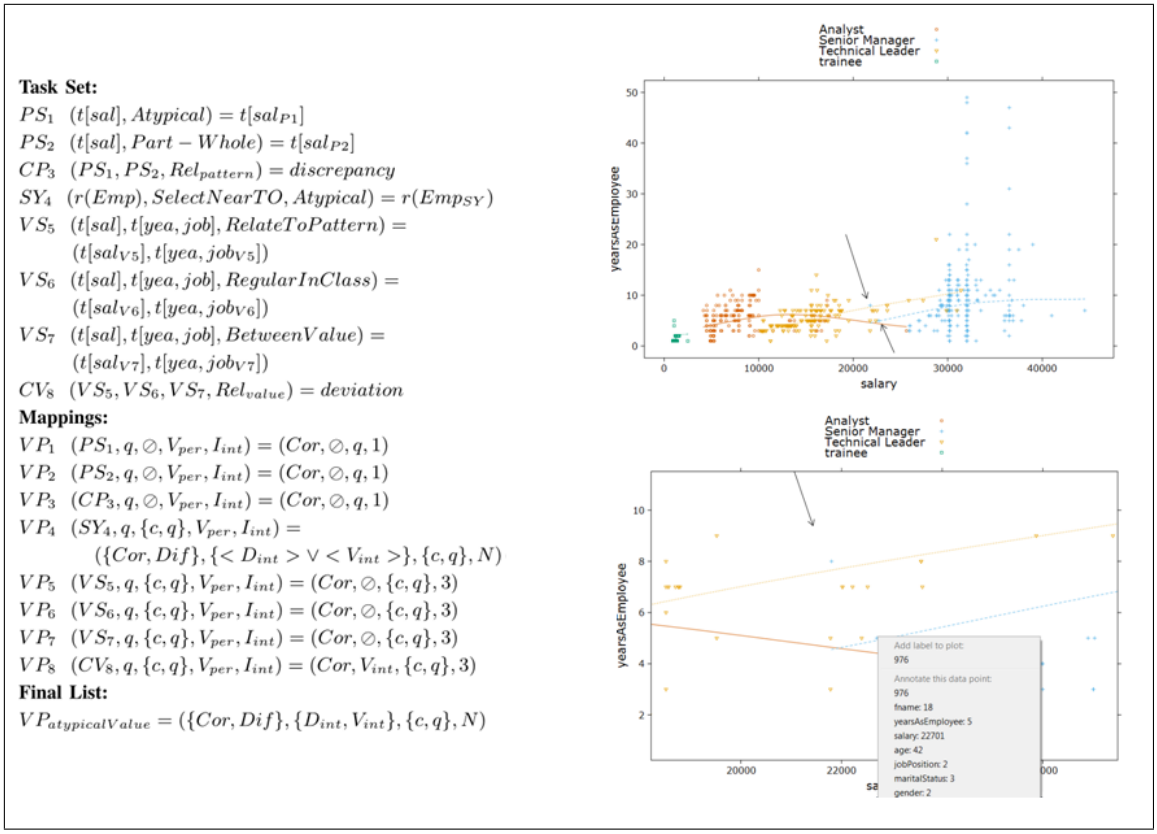


Figure 2. Framework in practice - preliminary study (Source: The authors)

It is worth to mention that this work has explored the probability sampling as a data volume reduction solution. The general idea is use the inference statistic to increase the probability to sample suspicious data in regard to defect structure. Due to the complexity and the impact on this work's scope, such a subject was left as future work.

6. Current Work and Future Directions

The current effort executes the exploratory case study to collect qualitative information about the relationship between each semantic data defect and visualization system properties. Subsequently, such an information will be basis for enhancing the framework preliminary structure. Lastly, a case study will explore the framework in practice. Additionally, a second paper will be produced to gather all these findings.

References

Alpar, P. and Winkelsträter, S. (2014). Assessment of data quality in accounting data with association rules. *Expert Systems with Applications*, 41(5):2259–2268.

Baldonado, M. Q. W., Woodruff, A., and Kuchinsky, A. (2000). Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '00, pages 110–119, New York, NY, USA. ACM.

Bertin, J. (1983). *Semiology of graphics: diagrams, networks, maps*. University of Wisconsin press.

Borovina Josko, J. M., Oikawa, M. K., and Ferreira, J. E. (2014). A taxonomy of data defects based on a formal framework. (Submitted to ACM Journal of Data and Information Quality).

Card, S. K., Mackinlay, J. D., and Shneiderman, B. (1999). *Readings in information visualization: using vision to think*. Morgan Kaufmann.

Casner, S. M. (1991). Task-analytic approach to the automated design of graphic presentations. *ACM Transactions on Graphics (TOG)*, 10(2):111–151.

Chambers, J. M. (2008). *Software for data analysis: programming with R*. Springer.

Csinger, A. (1992). The psychology of visualization. Technical Report TR-92-28.

Dasu, T. (2013). Data glitches: Monsters in your data. In *Handbook of Data Quality*, pages 163–178. Springer.

de Almeida, W. G., de Sousa, R. T., de Deus, F. E., Daniel Amvame Nze, G., and de Mendonca, F. L. L. (2013). Taxonomy of data quality problems in multidimensional data warehouse models. In *Information Systems and Technologies (CISTI), 2013 8th Iberian Conference on*, pages 1–7. IEEE.

Devillers, R., Jeansoulin, R., et al. (2006). *Fundamentals of spatial data quality*. ISTE London.

Elsmari, R. and Navathe, S. (2010). *Fundamentals of database systems*. Addison-Wesley, 6th. edition.

Gschwandtner, T., Gärtner, J., Aigner, W., and Miksch, S. (2012). A taxonomy of dirty time-oriented data. In *Multidisciplinary Research and Practice for Information Systems*, pages 58–72. Springer.

Kandel, S., Parikh, R., Paepcke, A., Hellerstein, J. M., and Heer, J. (2012). Profiler: integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 547–554, New York, NY, USA. ACM.

Kang, H., Getoor, L., Shneiderman, B., Bilgic, M., and Licamele, L. (2008). Interactive entity resolution in relational data: A visual analytic tool and its evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 14(5):999–1014.

Lee, Y. W., Pipino, L. L., Funk, J. D., and Wang, R. Y. (2009). *Journey to Data Quality*. The MIT Press.

Liu, J., Li, J., Liu, C., and Chen, Y. (2012). Discover dependencies from data-a review. *IEEE Transactions on Knowledge and Data Engineering*, 24(2):251–264.

Maier, D. (1983). *The theory of relational databases*, volume 11. Computer science press Rockville.

Oliveira, P., Rodrigues, F., and Henriques, P. (2005). A formal definition of data quality problems. In *International Conference on Information Quality*.

Patterson, R. E., Blaha, L. M., Grinstein, G. G., Liggett, K. K., Kaveney, D. E., Sheldon, K. C., Havig, P. R., and Moore, J. A. (2014). A human cognition framework for information visualization. *Computers & Graphics*, 42:42–58.

Potter, K., Rosen, P., and Johnson, C. (2012). From quantification to visualization: A taxonomy of uncertainty visualization approaches. In Dienstfrey, A. and Boisvert, R., editors, *Uncertainty Quantification in Scientific Computing*, volume 377 of *IFIP Advances in Information and Communication Technology*, pages 226–249. Springer Berlin Heidelberg.

Rahm, E. and Do, H. (2000). Data cleaning: Problems and current approaches. *IEEE Bulletin of the Technical Committee on Data Engineering*, 24:11.

Redman, T. and Blanton, A. (1997). *Data quality for the information age*. Artech House.

Rodrigues, J. F., Traina, A. J., de Oliveira, M. C. F., and Traina, C. (2007). The spatial-perceptual design space: A new comprehension for data visualization. *Information Visualization*, 6(4):261–279.

Rundensteiner, E., Ward, M., Xie, Z., Cui, Q., Wad, C., Yang, D., and Huang, S. (2007). Xmdvtool q: quality-aware interactive data exploration. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 1109–1112. ACM.

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, VL '96, pages 336–, Washington, DC, USA. IEEE Computer Society.

Tang, D., Stolte, C., and Bosch, R. (2004). Design choices when architecting visualizations. *Information Visualization*, 3(2):65–79.

Thomas, J. J. and Cook, K. A. (2005). *Illuminating the path: The research and development agenda for visual analytics*. IEEE Computer Society Press.

Thomson, J., Hetzler, E., MacEachren, A., Gahegan, M., and Pavel, M. (2005). A typology for visualizing uncertainty. In *Electronic Imaging 2005*, pages 146–157. International Society for Optics and Photonics.

Ware, C. (2004). *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Wilkinson, L., Wills, D., Rope, D., Norton, A., and Dubbs, R. (2006). *The grammar of graphics*. Springer.

Zhu, Y., Suo, X., and Owen, G. S. (2009). A visual data exploration framework for complex problem solving based on extended cognitive fit theory. In *Advances in Visual Computing*, pages 869–878. Springer.

paper:23

Declarative Model Discovery in Provenance Data for Aiding in Scientific Experiment Planning

Student: Mateus Ferreira Silva

mateus.silva@uniriotec.br

Advisors: Fernanda Baião, Kate Revoredo

{fernanda.baiao, katerevoredo}@uniriotec.br

Programa de Pós-Graduação em Informática
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Av. Pasteur, 458 – Rio de Janeiro – RJ – Brazil

Level: Master's degree

Proposal defense date: May/2014

Expected timing of completion: March/2015

Completed steps: Research project planning and activity schedule has defined. Research proposal was presented in a seminar at UNIRIO. The literature review is underway. Part of the solution has been implemented to conduct the case study, in which process mining techniques were applied to discover declarative models.

Future steps: Implementation and evaluation.

Abstract. *Data provenance manages a collection of metadata cataloging origin and history of data. In scientific workflows, this metadata supports scientific experiment planning. However, the amount of provenance data generated from scientific workflow executions can grow through time, becoming infeasible evaluate them manually. Thus, mechanisms for automatically extracting and presenting knowledge from provenance data are demanding. Due to the diversity and flexibility inherent to scientific experimentation scenarios, declarative models are potentially adequate. In this work, we propose an approach for automatically learning a declarative model from provenance data for aiding in scientific experiment planning.*

Keywords: provenance data, scientific workflow, declarative model, process mining.

1. Introduction

A scientific experiment has phases such as composition, execution and analysis [Matosso et al. 2010]. The focus of our work is in the scientific experiment planning, which generally happens in the composition phase. The contribution of this work is to present a method for planning future scenarios of scientific experiments using data management techniques in environments where large-scale provenance data is available.

Scientific workflows are suitable for supporting simulations during scientific experiments in a computational environment, in several scenarios such as climate model simulations, bioinformatics [Cuevas-Vicenttín et al. 2012], oil reservoir simulations or even in data management solutions. They assist scientists and other specialists in planning different workflow executions considering variations on input data, parameters or algorithms.

The historical data generated during the execution of scientific workflows is called provenance data and is important to manage, understand, reproduce and publish experiments [Cheah et al. 2013] [Deelman and Chervenak 2008]. However, for this to be possible, it is often necessary to capture a large amount of provenance data [Cheah et al. 2013]. In a large-scale environment, it is essential to identify the relevant information [Cuevas-Vicenttín et al. 2012] to assist planning execution scenarios of scientific workflows, among the huge amount of provenance data. However, the manual search for this information is costly and infeasible, thus demanding automatic mechanisms.

On the other hand, Knowledge Discovery in Databases (KDD) is a process that encompasses a mining step to learn models from data. In particular, declarative process mining can be applied to learn a declarative model from provenance data. Through declarative models it is possible to describe rules of workflows in a proper language, so domain specialists can better visualize and understand commonalities among successful instances of workflows in order to reuse, share and plan their experiments so as to obtain better results.

The objective of this work is to present a solution to assist scientists in planning scenarios for a scientific experiment, through the discovery of declarative models from provenance data.

The hypothesis of this research work is that IF techniques for declarative models discovery from provenance data are applied, THEN better execution scenarios of scientific workflows are found.

This article is further structured into six sections. Section 2 presents concepts of Scientific Workflows, Data Provenance and Declarative Process Mining. Section 3 presents the Solution Approach. Section 4 describes the Proposal Evaluation. Section 5 presents Current State of Research and Section 6 presents Related Works. Finally, Section 7 presents the concluding remarks.

2. Background

This section introduces the concepts of scientific workflow, provenance data and declarative process mining.

2.1 Scientific Workflow

A Scientific Workflow is an abstraction that models the chaining of activities and data of scientific experiments to be executed by a specialized engine. Many real-world scenarios can be expressed as workflows that can be automatically adapted to run in different environments. Complex workflows require a level of abstraction to help specialists express their experiments. This abstraction is offered by Scientific Workflow Management Systems (SWfMS), which model, execute and monitor scientific workflows [Oliveira et al. 2010]. Scientific workflows support specialists in planning different experiments considering variations on input data, parameters or algorithms. Kepler, VisTrails and Taverna are examples of SWfMS. A SWfMS can offer: run-time tasks management, resource capabilities, task scheduling, data provenance management, data transfer and monitoring tasks. There are other solutions such as SciCumulus, which is a middleware to orchestrate scientific workflows in distributed and parallel environments through different kinds of SWfMS [Oliveira et al 2010].

2.2 Data Provenance

Provenance is perceived as a crucial component of workflow systems, which helps scientists to ensure reproducibility of their scientific analysis and processes, publication and contribution between coworkers [Gil et al. 2007]. Provenance has been studied in various areas of data management, processing of scientific data and DBMS (Database Management Systems) [Cheney et al. 2009].

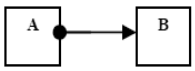
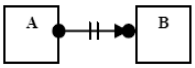
Different systems of data provenance have their own representation model. In order to provide interoperability among different systems, the PROV-DM (PROV Data Model) has been indicated as the standard for provenance representation. The PROV-DM is the conceptual data model that composes W3C provenance (PROV) family of specifications. PROV-DM represents the dependencies between the entities and their derivations, activities, agent and time at which they were created [Moreau and Missier 2013].

2.3 Declarative Process Mining

Process Mining is an advanced technique to discover process models through data logs. The learned models may be represented using several notations, such as BPMN, Petri nets, UML ADs, EPCs and BPEL, Declare and others [Maggi et al. 2011].

Declarative models focus on the logic that governs interactions between the actions of a process, describing what can be done, restricting only the undesired behavior [Zugal et al. 2013]. An example of declarative modeling language is Declare [Maggi et al. 2011]. Declare allows the discovery of non-sequential or coexisting events in the same process instance. The language is intended to be understandable for end-users and it has a formal semantics based on LTL (Linear Temporal Logic) semantics. ProM is a tool comprising algorithms for mining both imperative and declarative processes. It uses constraint templates with a graphical notation and implements semantics through operations such as *precedence*(A,B), *response*(A,B), *succession*(A,B), *not succession*(A,B), among others [Maggi et al. 2011]. Table 1 shows some examples of constraint templates, presenting LTL semantics and their graphical representation.

Table 1. Extract Of Declare Constraint Templates (Adapted [Bose et al. 2013])

Constraint	Meaning	LTL semantics	Graphical notation
precedence(A,B)	if B occurs then A occurs before B	$(\neg B \ U A) \vee \Box (\neg B)$	
response(A,B)	if A occurs then eventually B occurs after A	$\Box (A \Rightarrow \blacklozenge B)$	
succession(A,B)	for A and B both precedence and response hold	response(A,B) $\wedge$ precedence(A,B)	
not succession(A,B)	if A occurs then B cannot eventually occur after A	$\Box (A \Rightarrow \neg(\blacklozenge B))$	

3. Solution Approach

In this paper, we propose to automatically learn a declarative model from provenance data. The generic solution is designed to be applied to any PROV-compatible SWfMS. The learned declarative model can support specialists to select future scenarios of the scientific experiment to be executed, from among the provenance data generated by the workflows that achieved good results in the past. This approach will support scientists in better planning their scientific experiments. The quality measure to set the best results from each execution scenario should be previously defined by the specialist, and collected during the execution of workflow instances. The specialist must inform the quality metric of the workflow instances, e.g.: He/she can inform that workflow instances which obtain runtime less than two hours can be considered good results. Figure 1 depicts our solution architecture.

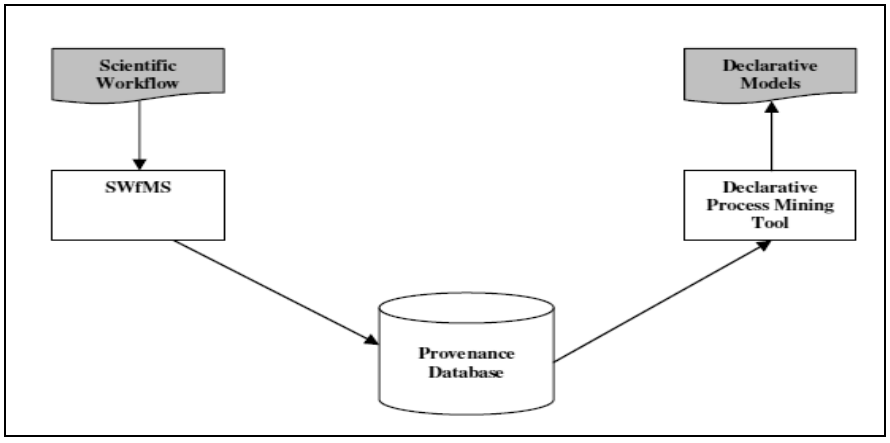


Figure 1– Solution Architecture

The ProM tool will be used to discover Declare models. Figure 2 presents a declare model extracted from a scientific workflow of an ontology matching process that executes four tasks: "*data loader*", and several combinations of the tasks, such as: *"apply terminological similarity metric"*, *"apply structural similarity metric"* and *"apply semantic metric"*. We can observe that similarity metric tasks only occur after

data loader. Thus, the "*data loader"* should precede "apply structural similarity metric. Another interesting extracted knowledge is *"apply terminological similarity metric"* is "*not succession*" *"(apply semantic metric")*, i.e., *"apply terminological similarity metric"* task should not be executed after *"apply semantic metric"* task. It is not possible visualize the last example in a scientific workflow model, because scientific workflow models are deterministics; and do not present constraints based in "*not*" semantics.

Provenance data is stored according to PROV metamodel. A step is automatically included in the workflow specification so as to calculate and collect quality metrics of workflow instances.

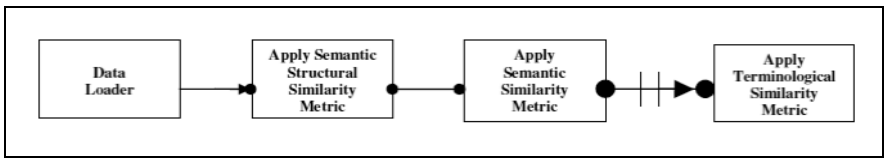


Figure 2 – Declarative model of scientific experiment of ontology matching

4. Proposal Evaluation

This solution is illustrated in a case study, using a scenario of ontology matching experiment. The dataset used is from OAEI (Ontology Alignment Evaluation Initiative) [OAEI 2014].

Ontology matching is a solution to the problem of semantic heterogeneity. It aims to find correspondences between semantically related entities in different ontologies given as input. These correspondences can be used for several tasks, such as ontology merging, answering a query or data translation. Similar entities are found using metrics such as linguistic, structural and semantic metrics. Each group evaluates similarities in different perspective; these perspectives are complementary [Euzenat and Shvaiko 2007].

The process of ontology matching was modeled as a scientific experiment. Different scenarios are simulated by combining several possible metrics and parameter values. Our workflow is implemented and executed in SciCumulus and provenance data is persisted in a relational DBMS.

Specialists in ontology matching will use the proposed solution. First, they will define the quality measure to set the best results of workflow instances. Next, they will model the scientific workflow. After this, we will measure the declare model accuracy through a fitness metric to evaluate the proposal.

5. Current State of Research

Currently, a controlled experiment was conducted to explore the solution, define future steps and evaluate the feasibility of our approach. A real workflow model was created and simulated, generating workflow instances. A ProM tool was executed in a provenance database. Besides, we reviewed the literature in order to search other process mining approaches to support and combine models representing extracted knowledge related to both data-flow and parameter-range perspectives.

Moreover, we are aware of the need to perform more tests and evaluate more case study scenarios. We will propose and implement a generic metamodel to define and represent the scientific workflow quality, and conduct a new case study in a real environment.

6. Related Works

Some SWfMS provide graphical interfaces to workflow design management, as VisTrail [Freire et al. 2006]; however, the inference to planning future workflows is visual. However, Vistrails does not provide any computational support for planning future scenarios.

Missier (2011) presents an approach of self-adaptive systems that use a feedback loop to incrementally analyze provenance data to control some of the workflow steps in subsequent executions. The author used a clustering algorithm to discover the possible subsequent workflow steps, which are then manually selected by experts. The goal is to offer trade-offs between execution cost, accuracy and completeness of the results.

Another work [Moore et al. 2013] proposes a declarative language for processing provenance data. While the approach proposed in [Bowers et al. 2012] infers data dependencies from workflow execution traces based on explicit user-defined rules, they propose a high-level language for expressing dependency rules that are converted in relational queries.

Although the above-mentioned approaches can be used to plan scientific workflows using data provenance, our proposed solution differs from them since we discover declarative models that represents the essence of a good workflow instance.

7. Concluding Remarks

Provenance data is an important part of scientific workflows, because it assists scientist and domain specialists to manage, understand, reproduce and publish their experiments [Cheah et al. 2013] [Gil et al. 2007] [Deelman and Chervenak 2008]. The use of data mining techniques is a way of providing the necessary information to domain specialists to reproduce and analyze their scientific experiment [Gil et al. 2007].

The solution proposed in this research applies techniques for discovering declarative models in large-scale provenance data generated by PROV-compatible SWfMS.

This proposal aims to provide information to domain specialists in planning future scenarios of scientific experiments. Preliminary results of the ontology matching study case, illustrated in this paper, points to the feasibility and good results in providing scientists with useful knowledge to better plan their future workflows.

References

Bose, R. J. C., Maggi, F. M., and van der Aalst, W. M. (2013) “Enhancing declare maps based on event correlations”. In Business Process Management, p. 97-112. Springer Berlin Heidelberg.

Bowers, S., McPhillips, T., and Ludäscher, B. (2012) “Declarative rules for inferring fine-grained data provenance from scientific workflow execution traces”, In

Provenance and Annotation of Data and Processes, p. 82-96. Springer Berlin Heidelberg.

Cheah, Y. W., Canon, R., Plale, B., and Ramakrishnan, L. (2013) "Milieu: Lightweight and Configurable Big Data Provenance for Science", In Big Data (BigData Congress), 2013 IEEE International Congress, p. 46-53.

Cheney, J., Chiticariu, L., and Tan, W. C. (2009) "Provenance in databases: Why, how, and where", In Now Publishers Inc, vol. 1, n. 4.

Cuevas-Vicenttín, V., Dey, S., Köhler, S., Riddle, S., and Ludäscher, B. (2012) "Scientific workflows and provenance: Introduction and research opportunities", In Datenbank-Spektrum, 12(3), p. 193-203.

Deelman, E., and Chervenak, A. (2008) "Data management challenges of data-intensive scientific workflows", In Cluster Computing and the Grid, 2008. CCGRID'08. 8th IEEE International Symposium, p. 687-692).

Euzenat, J., and Shvaiko, P. (2007) "Ontology matching". Heidelberg: Springer.

Freire, J., Silva, C. T., Callahan, S. P., Santos, E., Scheidegger, C. E., and Vo, H. T. (2006) "Managing rapidly-evolving scientific workflows", In Provenance and Annotation of Data, p. 10-18). Springer Berlin Heidelberg.

Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., and Myers, J. (2007) "Examining the challenges of scientific workflows", In IEEE Computer, 40(12), p. 26-34.

Maggi, F. M., Mooij, A. J., and van der Aalst, W. M. (2011) "User-guided discovery of declarative process models", In Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium, p. 192-199.

Mattoso, M., Werner, C., Travassos, G. H., Braganholo, V., Ogasawara, E., Oliveira, D., and Murta, L. (2010) "Towards supporting the life cycle of large scale scientific experiments", In. International Journal of Business Process Integration and Management, p. 79-92.

Missier, P. (2011) "Incremental workflow improvement through analysis of its data provenance", In TaPP.

Moore, S., Gehani, A., and Shankar, N. (2013) "Declaratively Processing Provenance Metadata", In TaPP.

Moreau , L. and Missier, P. (2013) "Prov-dm: The prov data model".

Oliveira, D., Ogasawara, E., Baião, F., and Mattoso, M. (2010) "Scicumulus: A lightweight cloud middleware to explore many task computing paradigm in scientific workflows", In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference, p. 378-385.

OAEI (2014) "Ontology Alignment Evaluation Initiative", http://oaei.ontologymatching.org, January.

Zugal, S., Soffer, P., Haisjackl, C., Pinggera, J., Reichert, M., and Weber, B. (2013) "Investigating expressiveness and understandability of hierarchy in declarative business process models", In Software & Systems Modeling, p. 1-23.

paper:25

Descoberta de Domínio Conceitual de Páginas Web

Aluno: Gleidson Antônio Cardoso da Silva
gleidson.silva@posgrad.ufsc.br
Orientadora: Carina Friedrich Dorneles
dorneles@inf.ufsc.br

Nível: Mestrado
Ingresso: Março de 2013
Conclusão prevista: Março de 2015
Etapas realizadas: Levantamento do estado da arte em extração de conceitos, classificação de páginas web e classificação de documentos. Definição do algoritmo proposto, implementação de protótipo, testes iniciais.
Etapas futuras: Avaliação e evolução do algoritmo proposto, comparação com outros algoritmos.

Programa de Pós-Graduação em Ciência da Computação
Universidade Federal de Santa Catarina

Resumo. *O aumento exponencial no número de documentos e dados na web demanda formas mais precisas para busca de informações e a classificação de páginas pode ser um grande aliado. Nesse contexto, o uso de informação semântica pode ser considerado essencial para a precisão da classificação. Sendo assim, este artigo propõe uma abordagem não supervisionada para a descoberta do domínio conceitual de páginas web. O algoritmo utiliza conceitos extraídos da página e de uma base de conhecimento afim de montar hierarquias de conceitos que são utilizadas para eleger um conceito mais representativo como domínio. Para avaliar o algoritmo foram realizados testes preliminares sobre conjuntos de páginas da Wikipedia. Os resultados e o potencial uso do algoritmo são apresentados e discutidos.*

Palavras-chave: classificação de páginas web, base de conhecimento, generalização de conceitos

1. Introdução

A classificação de documentos é comumente tratada como um problema de aprendizagem supervisionada, em que um conjunto predeterminado de dados é usado para treinar um classificador que atribui um ou mais rótulos de classe para cada documento [Tao et al. 2012]. Nesse contexto, as classes utilizadas são diretamente relacionadas com a capacidade de cobertura de termos pelo classificador, e a precisão de classificação é dada pela capacidade do classificador em encontrar uma ou mais classes corretas para os documentos.

Quando se trata de classificação no contexto de páginas web, a abordagem supervisionada se torna pouco eficiente, uma vez que os algoritmos devem ser capazes de trabalhar com o volume crescente de informações, e o uso de dados para treino pode se tornar inviável. Além disso, é necessário observar que o escopo semântico da web é irrestrito. Sendo assim, existe a necessidade de enriquecer o conhecimento utilizado pelo classificador com informações relevantes, afim de aumentar o poder de precisão e cobertura do processo de classificação.

Nesse sentido, trabalhos recentes exploram diferentes recursos com vistas à adição de semântica, considerada essencial para o avanço em algoritmos de classificação [Wang et al. 2008]. Em um deles, [Bleik et al. 2013] utiliza uma ontologia de domínio limitado para a classificação de documentos. Outros trabalhos utilizam recursos de domínio aberto como a Wikipedia [Yun et al. 2010] e a Wordnet [Luo et al. 2011] para obtenção de conceitos e relacionamentos. O grande problema de tais propostas é que necessitam de uma etapa de pré-processamento supervisionada sobre o corpus utilizado.

Desta forma, verifica-se a necessidade de uma proposta a fim de trabalhar com dados disponíveis online, tal como uma base de conhecimento de domínio aberto. Essa alternativa fornece uma fonte de informação semântica em tempo de execução, sem que haja necessidade de treinamento. Também possibilita a adição de conhecimento novo de forma independente. Sendo assim, este trabalho de mestrado apresenta um algoritmo para a descoberta de domínio conceitual de páginas web, tendo como entrada um conjunto de conceitos extraídos da página e uma base de conhecimento online de domínio aberto. Inicialmente, os conceitos são generalizados com a utilização de uma base de conhecimento. Por fim, o algoritmo elege um conceito como domínio.

Para apresentar o algoritmo proposto, o restante deste artigo está organizado da seguinte forma: a segunda seção aborda os fundamentos utilizados para a elaboração do algoritmo, apresentando também os trabalhos relacionados à proposta. Na terceira seção é apresentado em detalhes o algoritmo desenvolvido, em seguida são apresentados os testes realizados para avalização da proposta, discutindo os resultados obtidos. Por fim, são expostas considerações finais e trabalhos futuros.

2. Trabalhos relacionados

Recentemente, trabalhos vêm apresentando soluções alternativas à representação do conteúdo de documentos por palavras-chave com vistas à adição de semântica. Em um desses trabalhos [Barla et al. 2013] apresenta uma técnica para extração dos conceitos-chave a partir de texto livre utilizando a WordNet e um corpus disponibilizado pela Google. Por meio de estudos comparativos, o autor conclui que a representação utilizando conceitos-chave é mais eficiente que as técnicas convencionais de representação de conteúdo de documentos.

De forma similar, porém no âmbito da classificação de documentos, [Tao et al. 2012] apresenta um método para classificar livros sem a necessidade de treinar um classificador. Inicialmente o método minera o documento em busca de padrões de características, em um segundo momento, seleciona características a serem usadas na classificação. O método então utiliza uma ontologia de domínio aberto para, com o uso das entidades e relacionamentos presentes na ontologia, generalizar os termos adquiridos e utilizar os conceitos obtidos como classes para a classificação.

Essas abordagens trazem características muito importantes para um método de classificação de páginas web, como a independência de domínio e a ausência da etapa de treinamento. Entretanto, a classificação de páginas web difere em alguns aspectos da classificação de documentos [Qi et al. 2009]. Em primeiro lugar, a classificação de texto tradicional é normalmente realizada em documentos estruturados escritos com estilos consistentes (por exemplo, artigos de notícias), já páginas web normalmente não possuem essa característica. Em segundo, páginas web normalmente possuem conteúdo com assuntos diversificados (por exemplo, anúncios em blogs), enquanto documentos focam em assuntos específicos. A terceira e talvez mais importante característica, refere-se ao volume de texto, que varia muito em páginas web, e é abundante em documentos (se considerarmos livros por exemplo).

3. Abordagem Proposta

Neste trabalho, é definido um algoritmo não supervisionado, para a descoberta do domínio conceitual de páginas web. Para tal, assume-se que o domínio de um determinado conteúdo pode ser descrito por um único conceito de maior nível semântico que seja relacionado com conceitos que melhor representam o conteúdo obtido da página. Por exemplo, suponha que alguns dos principais termos candidatos a conceitos obtidos do processo de extração sejam: *presidente, estado e país*; o algoritmo proposto deve ser capaz de obter algum conceito mais generalista como *governo*.

O algoritmo se apóia na existência de conceitos mais abrangentes (que possam representar mais do que um conceito) entre aqueles que foram extraídos da página, dentre os quais alguns são mais relevantes dentro do escopo da página (que possuem maior semelhança entre si). Após encontrar o grupo dos conceitos mais relevantes e seus relacionamentos, é assumido que um conceito de maior nível hierárquico possa ser considerado o domínio. A proposta consiste em duas etapas principais expressas na Figura 1 e detalhadas a seguir.

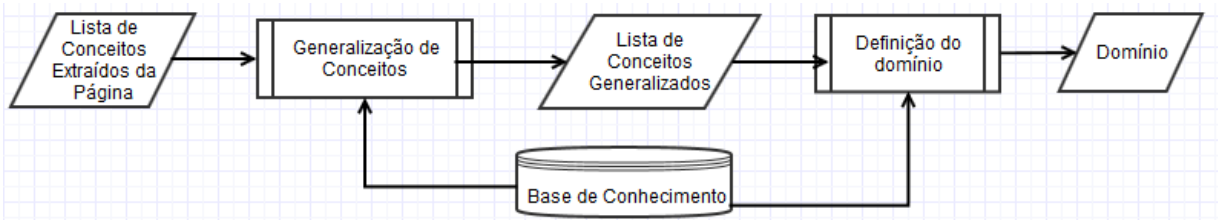


Figura 1. Etapas do algoritmo proposto.

3.1 Generalização de Conceitos

O módulo de Generalização de Conceitos tem como objetivo a construção de uma lista de conceitos de maior generalidade para cada conceito extraído da página. Conforme exemplificado na Figura 2, o módulo recebe como entrada uma lista de conceitos extraídos de uma página. Para cada conceito extraído, é obtida uma lista de conceitos de maior generalidade. A obtenção dessa lista é realizada com o uso de uma base de conhecimento. Primeiro, para cada conceito extraído, são consultados conceitos ligados por relacionamento IS-A, depois é realizada repetidamente a mesma consulta à base de conhecimento para cada conceito obtido na consulta anterior, até que a consulta não obtenha mais resultados. Os conceitos de maior nível na hierarquia construída compõem uma lista de conceitos candidatos a domínio para cada conceito extraído.

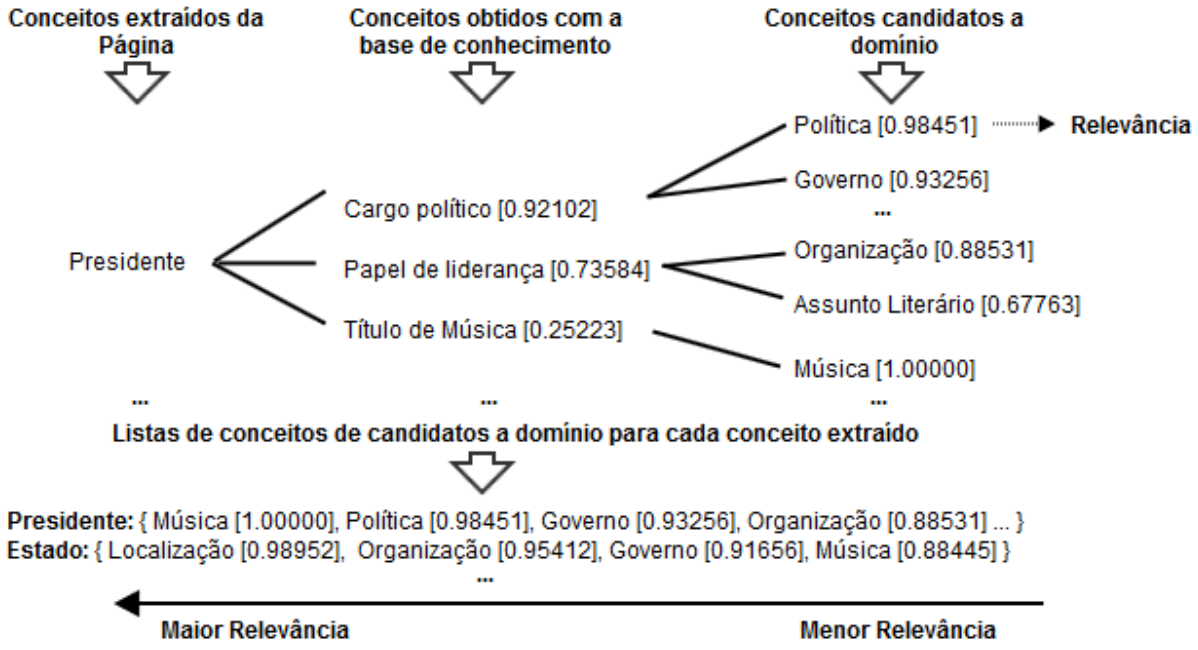


Figura 2. Exemplo de construção da lista de conceitos candidatos a domínio.

Após a geração das listas, estas devem ser ordenadas de acordo com a medida de relevância de cada conceito obtido em relação ao universo conhecido pela base consultada. No exemplo da Figura 2, a lista dos conceitos candidatos a domínio para o conceito presidente possui como conceito mais relevante Música, em seguida Política e assim em diante. Essa medida pode ser fornecida pela

própria base de conhecimento, ou calculada com uso de algoritmos como por exemplo PageRank, onde conceitos representam vértices e relacionamentos arestas.

3.2 Definição do Domínio

Para eleger um domínio, são considerados dois critérios: (i) o número de conceitos extraídos da página relacionados a cada conceito candidato a domínio; e (ii) a posição de conceitos candidatos nas listas de conceitos generalizados de cada conceito extraído. O primeiro critério expressa o quão comum pode ser o emprego de cada conceito candidato com relação ao escopo semântico da página. O segundo denota a probabilidade de conceito candidato ser mais importante que os outros dentro do contexto. Esses critérios são aplicados sobre um conjunto de listas de conceitos extraídos com suas respectivas listas de conceitos candidatos, como exemplificado na Figura 3. A escolha do domínio é feita através de um algoritmo que possui os seguintes passos:

Presidente: {Música, Política, Governo, Organização, Assunto Literário}
Estado: { Localização, Organização, Governo, Música}
País: {Governo, Organização, Localização, Música }

Figura 3. Exemplo de lista de conceitos extraídos com respectivas listas de conceitos elegíveis.

- **Passo 1:** Para cada conceito elegível, é verificado o número de vezes (frequência) que ele se faz presente nas listas de conceitos elegíveis de cada conceito extraído. Na Tabela 1, esses valores para os conceitos da Figura 3 estão discriminado na coluna Frequência, de acordo com essa figura o conceito candidato Música está presente nos três conceitos extraídos, desta forma sua frequência correspondente será 3.

Tabela 1. Exemplo de valores calculados de acordo com o algoritmo proposto

Conceitos	Frequência	Classificação Frequência	Soma Índices	Classificação Soma Índices	Classificação Final
Música	3	1	9	6	2.33
Governo	3	1	7	4	1.66
Organização	3	1	8	5	2
Localização	2	2	4	2	2
Assunto literário	1	3	5	3	6
Política	1	3	2	1	4

- **Passo 2:** Para cada conceito elegível, é realizado o somatório de suas respectivas posições nas lista de conceitos extraídos. Observando a Figura 3, é verificado que o conceito candidato Localização está presente nas listas dos conceitos extraídos Estado e País, nas posições 1 e 3 respectivamente. Desta forma, a soma de seus índices será igual a 4. Na Tabela 1, esses valores para os conceitos da Figura 3 estão discriminado na coluna Soma Índices.
- **Passo 3:** É construída uma lista com os conceitos candidatos, essa lista é então ordenada em ordem decrescente de acordo com a frequência desses conceitos. Em seguida para cada conceito é atribuído um valor de classificação de acordo com sua frequência. O valor de classificação da frequência é dado pela posição do valor do conceito na lista. Se observada a Tabela 1, pode-se verificar que a frequência de Música, Organização e Governo é 3, esse também é o maior valor de frequência do conjunto, então a classificação de frequência é 1.

- **Passo 4:** É construída uma lista com os conceitos candidatos respeitando a ordem com que esses conceitos aparecem nas listas dos conceitos extraídos. Em seguida essa lista é ordenada em ordem crescente de acordo com a soma dos índices dos conceitos. Em caso de empate durante essa ordenação, a ordem inicial é mantida. Posteriormente, para cada conceito candidato é atribuído um valor de classificação de acordo com sua posição na lista recém criada, esse valor está presente na Tabela 1, na coluna Classificação Soma Índices. Como exemplo tomamos o conceito Política na Tabela 1, como seu valor na soma dos índices é 2 e esse valor corresponde ao menor do conjunto, então sua classificação de soma de índices é 1.
- **Passo 3:** Para cada conceito candidato, é calculado o valor de classificação final para cada conceito candidato dado por:
    - **classificação final** = (**CF** + **CSI**) / **F** onde:
        - CF corresponde à classificação frequência;
        - CSI é classificação de soma de índices e
        - F equivale à frequência.
- **Passo 4:** O conceito candidato com menor valor de classificação final é eleito como domínio, caso o valor de classificação final seja zero ou igual ao valor do segundo conceito candidato, o conceito com maior classificação de frequência. Se ambos possuírem valores iguais para classificação de frequência então o conceito com menor valor de classificação de soma de índices entre eles é adotado como domínio.

4. Avaliação do Algoritmo

Para avaliar o desempenho do algoritmo foi implementado um protótipo em Java utilizando AlchemyAPI[1] para extração de conceitos das páginas e a API do Freebase[2] para consulta à base de conhecimento. Com a implementação do algoritmo foi possível realizar testes em páginas da Web. Inicialmente foram realizados testes com 150 páginas da Wikipedia[3], para avaliar diferentes domínios, foram selecionadas aleatoriamente 50 páginas de Filmes, 50 sobre Videogames e outras 50 se referindo a Localidades.

De 150 páginas, 103 tiveram o domínio eleito corretamente, o que representa 69% de precisão e 1 página não obteve resultados suficientes para execução do algoritmo. Apesar de preliminares, os resultados obtidos já demonstram como o comportamento do algoritmo depende da base de conhecimento e da qualidade dos conceitos fornecidos como entrada.

5. Considerações finais

Este trabalho apresenta uma proposta para a descoberta do domínio conceitual de páginas web. A informação sobre o domínio pode ser utilizada de forma a beneficiar o processo de classificação de páginas web. O algoritmo assume que para cada documento existe um conceito de maior generalizada.

[1] http://www.alchemyapi.com
[2] http://www.freebase.com
[3] http://www.wikipedia.org

Para encontrar esse conceito são extraídos os conceitos principais da página e, a partir destes conceitos, é montada uma lista de conceitos de maior generalidade com o auxílio de uma base de conhecimento de domínio aberto. A partir dos conceitos de nível mais alto na hierarquia, o algoritmo elege um conceito que represente semanticamente o maior número de conceitos extraídos da página e o assume como domínio.

Os próximos passos envolvem a avaliação de algoritmos de classificação com e sem a utilização do algoritmo proposto, afim de avaliar sua influência nos resultados em algoritmos já existentes. Além de uma avaliação mais profunda sobre algoritmo desenvolvido e possíveis ajustes e aperfeiçoamentos.

Referências

Barla, Michal, and Maria Bielikova. (2013). From Ambiguous Words to Key-Concept Extraction. Database and Expert Systems Applications (DEXA), 2013 24th International Workshop on. IEEE.

Bleik, S., Mishra, M., Huan, J., & Song, M. (2013). Text Categorization of Biomedical Data Sets Using Graph Kernels and a Controlled Vocabulary. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, *10*(5), 1211-1217.

Luo, Q., Chen, E., & Xiong, H. (2011). A semantic term weighting scheme for text categorization. *Expert Systems with Applications*, *38*(10), 12708-12716.

Qi, X., & Davison, B. D. (2009). Web page classification: Features and algorithms. *ACM Computing Surveys (CSUR)*, *41*(2), 12.

Tao, X., Li, Y., Lau, R. Y., & Wang, H. (2012). Unsupervised multi-label text classification using a world knowledge ontology. In *Advances in Knowledge Discovery and Data Mining* (pp. 480-492). Springer Berlin Heidelberg.

Wang, P., & Domeniconi, C. (2008). Building semantic kernels for text classification using wikipedia. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 713-721). ACM.

Yun, J., Jing, L., Yu, J., & Huang, H. (2010). Semantics-based representation model for multi-layer text classification. In *Knowledge-Based and Intelligent Information and Engineering Systems* (pp. 1-10). Springer Berlin Heidelberg.

paper:13

Desenvolvimento de um método de acesso métrico dinâmico eficiente baseado em pivôs adicionais*

Paulo Henrique de Oliveira[1],
Prof. Dr. Daniel dos Santos Kaster[1]

[1]Programa de Pós-Graduação em Ciência da Computação
Universidade Estadual de Londrina

oliveira.ph17@gmail.com, dskaster@uel.br

Nível	Mestrado
Ano de ingresso no programa	Agosto de 2013
Data da defesa da proposta	Dezembro de 2014
Época esperada de conclusão	Julho de 2015

Etapa	**Estado**
Revisão bibliográfica	Concluída
Definição do problema	Concluída
Definição da metodologia	Concluída
Definição da avaliação de resultados	Concluída
Implementação do método	Em desenvolvimento
Escrita do texto para o exame de qualificação	Em desenvolvimento
Realização de experimentos	Etapa futura
Análise de resultados	Etapa futura

Resumo. *Estruturas de indexação tradicionais não são adequadas para dados complexos, uma vez que a relação de ordem total não se aplica para tais dados. Existem estruturas adequadas para executar consultas sobre dados complexos, como os Métodos de Acesso Métricos (MAMs). A utilização de pivôs em MAMs permite particionar o espaço de busca de diferentes formas. Propostas recentes de estruturas baseadas em pivôs globais têm apresentado bom desempenho em consultas. Entretanto, o uso de pivôs globais pode comprometer a dinamicidade das estruturas. Assim, este trabalho propõe a construção de um MAM dinâmico baseado em múltiplos pivôs locais eficiente em termos de cálculos de distância e de acessos a disco.*

Palavras-chave: *consultas por similaridade, MAMs, pivôs adicionais.*

*Os autores agradecem à CAPES o apoio financeiro para o desenvolvimento deste trabalho.

1. Fundamentação teórica

Dados complexos, no contexto deste trabalho, são dados que não são representáveis por tipos de dados tradicionais, como números, datas e textos curtos. Dados multimídia, dados georreferenciados e séries temporais são alguns exemplos de dados complexos. O volume de dados complexos tem aumentado rapidamente devido à disseminação de dispositivos de aquisição de dados, tais como câmeras digitais e equipamentos para exames médicos e moleculares. O sucesso de serviços de compartilhamento como YouTube e Flickr é mais uma evidência do crescimento desses tipos de dados [Barrios and Bustos 2011].

Para permitir a execução de consultas sobre dados complexos, usualmente, extrai-se deles um conjunto de características. A recuperação de dados complexos através dessas características é conhecida como *recuperação baseada em conteúdo*. O conjunto de características extraído, chamado de *vetor de características*, é usado no lugar dos dados originais para avaliar as consultas. Dados complexos são comparados através de relações de (dis)similaridade entre pares de vetores de características. Isso ocorre aplicando-se uma função de distância cujo valor de retorno representa o quão distantes os dois vetores de características estão um do outro. Assim, consultas realizadas sobre dados complexos são chamadas de *consultas por similaridade*. Essas consultas recuperam os elementos mais similares ao elemento de consulta de acordo com a condição fornecida. As consultas por similaridade mais comuns são a consulta por abrangência (*Range query*), que retorna os elementos cuja dissimilaridade em relação ao elemento de consulta é menor ou igual a um dado limiar, e a consulta aos k-vizinhos mais próximos (k-*Nearest Neighbors query*), que retorna os k elementos mais similares ao elemento de consulta [Barioni et al. 2011].

Grande parte das estruturas de indexação implementadas por Sistemas Gerenciadores de Bancos de Dados (SGBDs) comerciais é capaz de executar consultas com muita eficiência sobre dados tradicionais fazendo uso da propriedade de *relação de ordem total* [Vieira et al. 2010]. O significado dessa propriedade é que, para cada par de elementos do domínio, é possível identificar qual deles precede o outro. No entanto, a propriedade de relação de ordem total não se aplica para a maioria dos domínios de dados complexos [Faloutsos 1996]. Portanto, as estruturas de indexação tradicionais não podem ser utilizadas para trabalhar em domínios complexos de forma apropriada. Por outro lado, existem estruturas de indexação adequadas para a execução de consultas por similaridade sobre dados complexos, dentre as quais destacam-se os Métodos de Acesso Métricos (MAMs).

1.1. Métodos de Acesso Métricos

Métodos de Acesso Métricos (MAMs) são estruturas de indexação que consideram que os dados estão imersos em um espaço métrico. Um *espaço métrico* é definido por um par $\langle \mathbb{S}, \delta \rangle$, onde $\mathbb{S}$ é o domínio de dados e δ é uma função de distância definida sobre esse domínio. A função $\delta : \mathbb{S} \times \mathbb{S} \mapsto \mathbb{R}_+$ é denominada *métrica* e deve satisfazer as propriedades a seguir, também conhecidas como *postulados do espaço métrico* [Zezula et al. 2006]:

- $\forall x, y \in \mathbb{S}, \delta(x, y) \geq 0$ — não-negatividade
- $\forall x, y \in \mathbb{S}, \delta(x, y) = \delta(y, x)$ — simetria
- $\forall x, y \in \mathbb{S}, x = y \iff \delta(x, y) = 0$ — identidade
- $\forall x, y, z \in \mathbb{S}, \delta(x, z) \leq \delta(x, y) + \delta(y, z)$ — desigualdade triangular

Com base nesses conceitos, os MAMs são capazes de indexar praticamente qualquer tipo de dado, sendo necessário apenas definir uma métrica para o domínio de dados

em questão, que, tipicamente, são vetores de características extraídos do conteúdo dos dados complexos. Existem inúmeros MAMs relatados na literatura, categorizados em geral com base em algumas características. Uma característica importante é se o MAM permite responder a consultas de forma exata (a resposta é exata) ou aproximada (a precisão da resposta é relaxada a fim de permitir uma execução mais eficiente). Outra característica é se a estrutura é estática (exige a existência a priori do conjunto de dados indexado e não permite atualizações posteriores) ou dinâmica (permite adicionar elementos sem degenerar a estrutura). Por fim, o tipo de particionamento do espaço e o tipo dos pivôs (globais ou locais) também são fatores importantes que diferenciam os MAMs. No contexto deste trabalho, pivôs são elementos referenciados pelo restante dos elementos da base de dados (ou por parte deles no caso de pivôs locais).

1.2. Escolha de pivôs em MAMs

O critério de escolha de pivôs no processo de construção dos MAMs influencia o desempenho das consultas, uma vez que eles ajudam a restringir a região de busca ao serem utilizados na poda de elementos não relevantes. Estudos realizados por [Bustos et al. 2003] apontam que bons pivôs estão distantes uns dos outros. Essa afirmação faz sentido, uma vez que pivôs muito próximos tendem a dar a mesma informação no processo de poda.

Tipicamente, os pivôs podem ser *pivôs globais* ou *pivôs locais*. Pivôs globais são referenciados por todos os elementos da base de dados, enquanto pivôs locais são referenciados por apenas alguns elementos, isto é, cada pivô local é associado a um número limitado de elementos do conjunto de dados. Para pivôs locais, existe também o critério de proximidade: um pivô é bom se estiver próximo a algum elemento da base de dados ou ao elemento de consulta. Em ambas as situações, o pivô fornece informações de distância mais precisas em relação ao elemento de quem está próximo, o que torna mais efetiva a poda de elementos não relevantes [Skopal and Hoksza 2007].

2. Caracterização da contribuição

2.1. Definição do problema e objetivo da pesquisa

MAMs que fazem uso de pivôs podem ser categorizados em: baseados em pivôs globais, baseados em pivôs locais e híbridos. MAMs baseados em pivôs globais particionam bem o espaço de busca se o número de pivôs e sua distribuição forem adequados. Porém, uma eventual necessidade de modificação no conjunto de pivôs, em geral, exige a reconstrução da estrutura. Por outro lado, a manutenção de pivôs locais requer a atualização de apenas uma parte dos elementos indexados. Essa característica é comum nos MAMs dinâmicos. Para aprimorar o desempenho das estruturas dinâmicas com pivôs locais, uma estratégia consiste em acrescentar pivôs globais ou locais. MAMs híbridos acrescentam pivôs globais, aumentando o desempenho, mas limitando a dinamicidade. Já no caso das propostas existentes que utilizam múltiplos pivôs locais, os ganhos obtidos dizem respeito apenas à redução no número de cálculos de distância.

Diante disso, este trabalho de mestrado tem como objetivo desenvolver um MAM dinâmico baseado em múltiplos pivôs locais que seja eficiente tanto em termos de cálculos de distância quanto de acessos a disco, explorando diferentes abordagens. O modo como pretende-se atingir esses objetivos é descrito na subseção 2.2.

2.2. Metodologia para resolução do problema

A metodologia desta proposta tem como foco estender um MAM dinâmico hierárquico. As estratégias fundamentais do trabalho são: (i) eleger elementos em cada nó como pivôs adicionais a fim de aprimorar a capacidade de poda da estrutura sem modificar a região de cobertura dos nós, (ii) modificar a estrutura do nó a fim de obter os valores das distâncias dos elementos aos pivôs e dos raios de cobertura sem acessar o nó em disco e (iii) definir formas de particionamento dos nós em regiões disjuntas, mantendo a estrutura dinâmica. As atividades a serem desenvolvidas são apresentadas a seguir.

1. Definir políticas de escolha de pivôs adicionais.
2. Modificar o MAM para acrescentar pivôs adicionais sem alterar o particionamento dos nós (i.e. as regiões de cobertura de cada nó são definidas pelo pivô principal).
3. Modificar a estrutura dos nós para antecipar o acesso às distâncias e aos raios.
4. Modificar o MAM definindo formas de particionamento dos nós em nós "gêmeos" de forma que cubram regiões disjuntas. Note-se que as regiões de cobertura de nós irmãos (não "gêmeos") podem ainda ter sobreposição.
5. Acomodar essas mudanças nos métodos de inserção e remoção e nas consultas.
6. Executar experimentos e analisar os resultados.
7. Disseminar os resultados da pesquisa.

3. Estado atual do trabalho

Primeiramente, as atividades deste trabalho de mestrado concentraram-se no aprendizado dos conceitos relacionados a dados complexos, buscas por similaridade, espaços métricos e MAMs. Em seguida, iniciou-se o desenvolvimento do novo MAM utilizando a primeira estratégia (utilizar pivôs adicionais sem modificar a região de cobertura dos nós), que usa como base a estrutura hierárquica da Slim-tree [Traina Jr. et al. 2002]. Esta seção dá uma visão geral da Slim-tree e do MAM em desenvolvimento, que recebeu o nome Slim*-tree.

3.1. Visão geral da Slim-tree

Na Slim-tree, os elementos são agrupados em páginas de tamanho fixo para serem gravados em disco, cada página correspondendo a um nó da árvore, e são guardados nas folhas. Os elementos são organizados em uma estrutura hierárquica em que um elemento denominado representativo é utilizado como centro da região de cobertura de uma subárvore, delimitada por um raio máximo. Cada elemento de um nó tem sua distância ao representativo calculada e armazenada no momento de construção da árvore. Isso é feito a fim de que sejam economizados cálculos de distância durante as consultas.

Existem dois tipos de nós na Slim-tree: os nós índice (*index nodes*) e os nós folha (*leaf nodes*). Um nó folha possui a seguinte estrutura:

$$\textit{leaf node}\ [\text{vetor de } \langle OID_i, s_i, \delta(s_i, s_{rep})\rangle]$$

Nessa estrutura, OID_i é o identificador do elemento em questão, s_i é o elemento propriamente dito, armazenado como um vetor de características, e $\delta(s_i, s_{rep})$ é a distância de s_i ao representativo do nó. Um nó índice possui a seguinte estrutura:

$$\textit{index node}\ [\text{vetor de } \langle s_i, r_i, \delta(s_i, s_{rep}), Ptr(T_{s_i}), \#Ent(T_{s_i})\rangle]$$

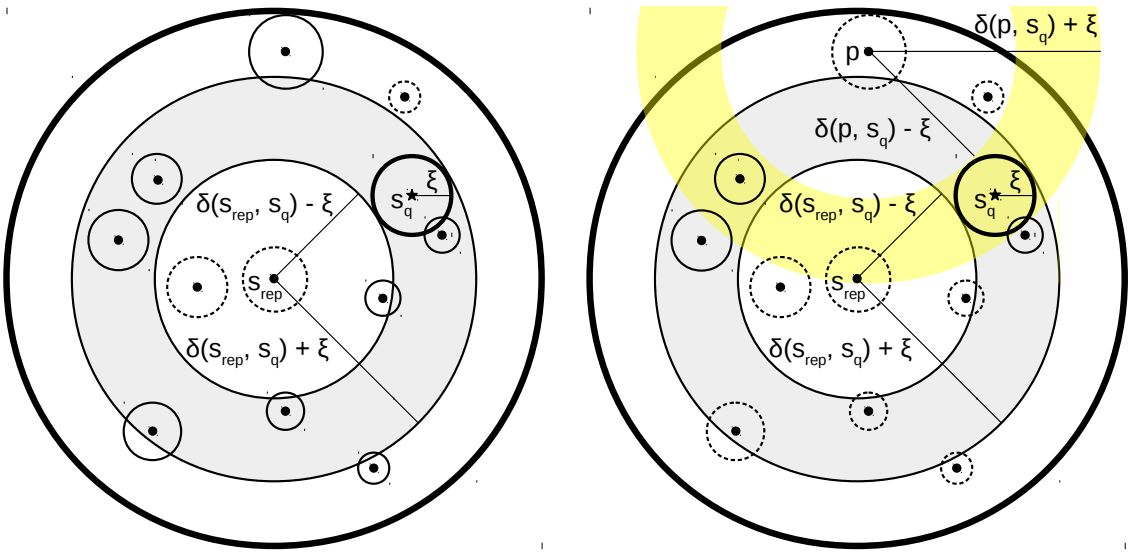
Nessa estrutura, s_i é o vetor de características do representativo da subárvore apontada por $Ptr(T_{s_i})$, r_i é o raio de cobertura do nó dessa subárvore (definido pela distância entre o representativo e o elemento mais distante nesse nó), $\delta(s_i, s_{rep})$ é a distância de s_i ao representativo dessa subárvore e $\#Ent(T_{s_i})$ é o número de entradas em T_{s_i}.

3.2. Visão geral da Slim*-tree

Na Slim*-tree, os elementos são organizados de forma hierárquica como na Slim-tree. A principal diferença em sua estrutura está no conteúdo dos nós, de forma que os valores de distância e o raio de cada elemento são deslocados um nível acima para que sejam obtidos antes de acessar o nó em disco, conforme descrito na subseção 2.2. Outra característica é que são escolhidos pivôs em cada nó para serem usados no processo de poda de elementos não relevantes. Esse processo de poda faz uso da propriedade de desigualdade triangular e ocorre da seguinte maneira:

1. Considerando o representativo do nó (Figura 1(a)), é podado todo elemento s_i (i.e. não é preciso calcular a distância de s_i a s_q) que satisfaz uma das inequações:
   - $\delta(s_{rep}, s_i) + r_i < \delta(s_{rep}, s_q) - \xi$
   - $\delta(s_{rep}, s_i) - r_i > \delta(s_{rep}, s_q) + \xi$
2. Para todo s_i não podado, considera-se cada um dos pivôs (Figura 1(b)) para tentar podar mais elementos. É podado todo elemento que satisfaz uma das inequações:
   - $\delta(p_j, s_i) + r_i < \delta(p_j, s_q) - \xi$
   - $\delta(p_j, s_i) - r_i > \delta(p_j, s_q) + \xi$

Por fim, outra melhoria que a Slim*-tree possui é que os elementos de cada nó são ordenados da maior para a menor soma da distância ao representativo mais o raio (apenas pela distância ao representativo em um nó folha). Com isso, durante a varredura do nó que utiliza a desigualdade triangular pelo representativo, ao encontrar o primeiro elemento que satisfaz $\delta(s_{rep}, s_i) + r_i < \delta(s_{rep}, s_q) - \xi$, pode-se ignorar o restante dos elementos do nó seguramente, pois sabe-se que eles não farão parte da resposta da consulta. Atualmente, já existe uma implementação inicial da Slim*-tree em fase de testes.



(a) Pelo representativo. (b) Pelo representativo mais um pivô adicional.

Figura 1. Processo de poda através da desigualdade triangular. Circunferências pontilhadas representam nós podados durante a consulta.

4. Trabalhos correlatos

Em [Traina Jr. et al. 2007], é proposta a técnica Omni, baseada em pivôs globais selecionados a partir do conjunto de dados, que pode ser aplicada em diversos MAMs dinâmicos. Tais MAMs podem ser considerados dinâmicos no que diz respeito a inserções e remoções de elementos. No entanto, no caso de mudança de pivôs, deve-se atualizar toda a estrutura. O trabalho de [Esuli 2012] apresenta um índice baseado em permutação, que realiza buscas aproximadas, em que cada elemento é representado por uma sequência, ordenada em relação à distância dos pivôs globais do conjunto, do pivô mais próximo ao pivô mais distante. A fim de economizar espaço em disco, são armazenados prefixos das permutações que representam os elementos (daí o nome *Permutation Prefix Index* — PP-Index). Nesse índice, também é necessário atualizar toda a estrutura no caso de mudança de pivôs. Como resultado, esses métodos permitem diminuir os acessos a disco e os cálculos de distância.

Diferentemente desses métodos baseados em pivôs globais, a Slim-tree (apresentada na subseção 3.1) usa pivôs locais, que são os representativos dos nós. Existe também a M*-tree, proposta por [Skopal and Hoksza 2007], que utiliza múltiplos pivôs locais em cada nó, além do representativo. Tais métodos têm a vantagem de serem dinâmicos, pois apenas parte da estrutura precisa ser reajustada quando há mudança de pivô. O mesmo vale para a Slim*-tree, que é a proposta deste trabalho. Porém, tanto a Slim-tree quanto a M*-tree são beneficiadas pelos pivôs apenas na redução de cálculos de distância. No caso da Slim*-tree, deseja-se reduzir também os acessos a disco modificando-se sua estrutura. Outro método baseado em pivôs locais é a DAHC-tree [Almeida et al. 2010], cuja maior diferença é o uso de regiões disjuntas e não disjuntas. Esse método é aproximado e sua estrutura pode ser desbalanceada. Ele permite diminuir acessos a disco e cálculos de distância. Porém, alguns de seus parâmetros dependem do conjunto de dados.

Existem também métodos híbridos. O trabalho feito por [Skopal et al. 2005] apresenta a PM-tree, uma estrutura que reduz a região de busca dos nós através de pivôs globais. Uma evolução dessa estrutura, a PM*-tree [Skopal and Hoksza 2007], acrescenta o uso de múltiplos pivôs locais assim como na M*-tree. Entretanto, ambas são estáticas. Os conceitos introduzidos pela PM-tree que diminuem a região de busca foram formalizados em [Lokoč et al. 2014] e podem ser aplicados em outros métodos de acesso.

5. Avaliação dos resultados

Os resultados serão avaliados através de experimentos envolvendo dados reais e sintéticos. No que diz respeito a dados reais, exemplos de bases de dados a serem usadas são: ALOI [Geusebroek et al. 2005], que tem 72 mil imagens, e CoPhIR [Bolettieri et al. 2009], que tem quase 106 milhões de imagens. Em relação a dados sintéticos, serão avaliados conjuntos de dados variando-se a dimensionalidade, o tamanho e a distribuição dos dados. Os parâmetros de avaliação incluem o desempenho de construção das estruturas, consumo de espaço e desempenho de execução de consultas por similaridade. Esses são os parâmetros normalmente considerados em trabalhos em que MAMs são desenvolvidos. Quanto ao desempenho de consultas por similaridade, espera-se que seu tempo de execução seja menor uma vez que os acessos a disco e os cálculos de distância possam ser reduzidos.

Referências

Almeida, J., Valle, E., da S. Torres, R., and Leite, N. J. (2010). DAHC-tree: An Effective Index for Approximate Search in High-Dimensional Metric Spaces. *Journal of*

Information and Data Management, 1(3):375–390.

Barioni, M. C. N., Kaster, D. S., Razente, H. L., Traina, A. J. M., and Traina Jr., C. (2011). Querying Multimedia Data by Similarity in Relational DBMS. In Yan, L. and Ma, Z., editors, *Advanced Database Query Systems: Techniques, Applications and Technologies*, pages 323–359. IGI Global, Hershey, PA, USA.

Barrios, J. M. and Bustos, B. (2011). Automatic Weight Selection for Multi-Metric Distances. In *Proceedings of the 4th International Conference on Similarity Search and Applications*, SISAP '11, pages 61–68, New York, NY, USA. ACM.

Bolettieri, P., Esuli, A., Falchi, F., Lucchese, C., Perego, R., Piccioli, T., and Rabitti, F. (2009). CoPhIR: a Test Collection for Content-Based Image Retrieval. *CoRR*, abs/0905.4627v2.

Bustos, B., Navarro, G., and Chávez, E. (2003). Pivot selection techniques for proximity searching in metric spaces. *Pattern Recognition Letters*, 24(14):2357–2366.

Esuli, A. (2012). Use of permutation prefixes for efficient and scalable approximate similarity search. *Information Processing & Management*, 48(5):889–902.

Faloutsos, C. (1996). *Searching Multimedia Databases by Content*, volume 3 of *Advances in Database Systems*. Kluwer Academic Publishers.

Geusebroek, J.-M., Burghouts, G. J., and Smeulders, A. W. M. (2005). The Amsterdam Library of Object Images. *International Journal of Computer Vision*, 61(1):103–112.

Lokoč, J., Moško, J., Čech, P., and Skopal, T. (2014). On indexing metric spaces using cut-regions. *Information Systems*, 43(0):1–19.

Skopal, T. and Hoksza, D. (2007). Improving the Performance of M-Tree Family by Nearest-Neighbor Graphs. In *Proceedings of the 11th East European Conference on Advances in Databases and Information Systems*, ADBIS '07, pages 172–188, Berlin, Heidelberg. Springer-Verlag.

Skopal, T., Pokorný, J., and Snášel, V. (2005). Nearest Neighbours Search Using the PM-Tree. In Zhou, L.-Z., Ooi, B. C., and Meng, X., editors, *Database Systems for Advanced Applications*, volume 3453 of *Lecture Notes in Computer Science*, pages 803–815. Springer Berlin Heidelberg.

Traina Jr., C., Filho, R. F. S., Traina, A. J. M., Vieira, M. R., and Faloutsos, C. (2007). The Omni-family of all-purpose access methods: a simple and effective way to make similarity search more efficient. *The VLDB Journal*, 16(4):483–505.

Traina Jr., C., Traina, A. J. M., Faloutsos, C., and Seeger, B. (2002). Fast Indexing and Visualization of Metric Data Sets Using Slim-Trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):244–260.

Vieira, M. R., Traina Jr., C., Chino, F. J. T., and Traina, A. J. M. (2010). DBM-Tree: A Dynamic Metric Access Method Sensitive to Local Density Data. *Journal of Information and Data Management*, 1(1):111–128.

Zezula, P., Amato, G., Dohnal, V., and Batko, M. (2006). *Similarity Search — The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Springer, 1st edition.

paper:15

Mineração de Opinião Multilíngue para Comentários sobre Produtos

Anderson U. Kauer[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

aukauer@inf.ufrgs.br

Resumo. *Avaliar produtos está se tornando uma prática comum entre consumidores na web. As avaliações fornecidas pelos usuários ajudam empresas a melhorarem seus produtos e fornecem aos novos consumidores informações adicionais sobre suas experiências. Entretanto, estas revisões estão espalhadas em diferentes sites de vendas e, frequentemente, estão disponíveis em outros idiomas. O objetivo deste trabalho é desenvolver métodos visando aumentar a densidade de opiniões para produtos brasileiros, utilizando avaliações de consumidores escritas em idiomas estrangeiros. Estão previstas tarefas de identificação, agrupamento, sumarização e classificação de opiniões. Este artigo descreve as tarefas já desenvolvidas e aponta as que ainda precisam ser realizadas para que o objetivo proposto seja alcançado.*

Palavras-chaves: mineração de opinião, mineração de dados multilíngues, extração de aspectos

Orientação: Viviane P. Moreira (viviane@inf.ufrgs.br)

Nível: Mestrado

Ano de ingresso no programa: 2013

Época esperada de conclusão: março de 2015

Etapas já concluídas: As etapas do método proposto são apresentadas na (Figura 1). As etapas concluídas são: proposta, pré-processamento e implementação de método para extração de aspectos, bem como a realização de experimentos preliminares utilizando datasets rotulados.

Etapas futuras: Ainda é necessário pesquisar métodos para identificação de um mesmo produto em diferentes idiomas e métodos para classificação da polaridade associada a um aspecto. Além disso, deve-se implementar os métodos para sumarização de aspectos para saída em português. Estas tarefas foram estudadas, mas ainda estão pendentes de implementação.

1. Introdução

Atualmente o comportamento dos usuários da web está mudando, pois além de consumir conteúdos ali disponíveis, eles também estão expondo suas opiniões e experiências, seja sobre um produto que adquiriram, um local que visitaram ou um serviço que utilizaram. Estes comentários estão disponíveis abertamente para consulta e, portanto, podem influenciar na decisão de outros usuários. Este processo de retroalimentação pode ser bastante útil, tanto para empresas, que podem utilizar destas informações para aprimorar seus produtos, quanto para os consumidores, que também podem tirar proveito de experiência de outros usuários. Os comentários servem como informação adicional que frequentemente não está disponível na descrição do produto ou serviço.

A mineração de opinião ou análise de sentimento (*opinion mining/sentiment analysis*) [Liu 2012] é a área voltada para a análise de opiniões, sentimentos, emoções e atitudes de pessoas e tem se tornado objeto de pesquisa em diversos cenários a partir da mídia social, tais como blogs, e-mails, fóruns de discussão, revisão de produtos, etc. Por outro lado, a mineração de textos multilíngues (*Multilingual Text Mining* – MTM) preocupa-se com o enriquecimento da informação a partir do casamento entre documentos em diferentes idiomas [Steinberger 2012]. A combinação entre mineração de opinião e mineração de textos multilíngue permitiria um maior número de avaliações, bem como uma maior diversidade e densidade de pontos de vistas de consumidores. Esta combinação possui aplicabilidade em diversos cenários, como por exemplo, fornecer a estrangeiros em viagem, sumários de opiniões sobre serviços, estabelecimentos, pontos turísticos, etc. feitos por consumidores locais ou permitir a comparação de comentários sobre produtos em diferentes idiomas [Zhou et al. 2012].

A prática de redigir comentários avaliando produtos e/ou serviços é bem mais frequente em algumas regiões do que em outras. Em um estudo preliminar, coletamos manualmente o número de avaliações de alguns produtos nos maiores sites de vendas online brasileiros comparados com a amazon.com. Esta análise mostrou que o número total de avaliações do site amazon.com, em média, é sete vezes maior do que a soma das avaliações dos sites brasileiros pesquisados. A motivação para este trabalho vem da possibilidade de utilizar revisões disponíveis em sites estrangeiros para gerar sumários contendo aspectos importantes do produto.

O objetivo deste trabalho é desenvolver métodos que utilizam avaliações de consumidores escritas em idiomas estrangeiros para fornecer sumários sobre produtos em sites brasileiros. Desta forma, o método proposto permitiria ao usuário ter mais informações sobre o produto em questão e, com isso, mais subsídios para embasar sua decisão de compra. Este artigo apresenta os conceitos fundamentais necessários para o entendimento do trabalho, uma síntese da literatura relacionada, as tarefas previstas e os experimentos preliminares realizados.

2. Trabalhos Relacionados

A mineração de opinião pode ser dividida em diversas subtarefas. Os trabalhos relacionados estão subdivididos principalmente em dois grupos: (i) extração e sumarização de aspectos e (ii) análise de sentimentos multilíngues. A identificação e extração de aspectos visa encontrar as palavras contidas nos comentários que são o alvo da opinião. Para

isso, existem quatro abordagens principais, das quais duas estão diretamente relacionadas a este trabalho.

A *extração baseada em substantivos frequentes* tem como objetivo encontrar expressões com aspectos explícitos, representados por substantivos a partir de um grande conjunto de revisões. [Hu and Liu 2004] utilizaram um algoritmo de mineração de dados combinado a um etiquetador gramatical. Esta abordagem é bastante interessante porque assume que usuários tendem a utilizar o mesmo vocabulário para definir um mesmo aspecto, enquanto que conteúdo irrelevante tende a ser pouco frequente no conjunto de revisões. A utilização de coocorrências de palavras foi estudada por [Moghaddam and Ester 2010]. Os autores buscaram identificar os aspectos mais frequentes e remover os substantivos irrelevantes baseado no número de ocorrência na coleção de revisões, utilizando remoção de afixos (*stemming*) e remoção de palavras comuns (*stopwords*).

A *extração por relação entre opinião e aspecto* foi inicialmente utilizada por [Qiu et al. 2011]. Os autores utilizam gramática de dependência para descrever as relações sintáticas, considerando que as relações entre aspectos e opiniões ocorrem somente de forma direta, que aspectos podem ser somente substantivos e opiniões somente como adjetivos. Da mesma forma, [Liu et al. 2013] utilizaram-se destes conceitos para aperfeiçoar o método ao remover palavras que não podem ser aspectos (chamadas de *general words*). Desta forma, melhorando a precisão do método sem perda significativa da revocação. A principal desvantagem desta abordagem deve-se principalmente ao fato de que alguns aspectos são expressões multi-palavras e muitas vezes estes aspectos estão implícitos. Entretanto, o método é bastante eficaz principalmente por identificar palavras de opinião independentemente do domínio.

A sumarização baseada em aspectos permite uma apresentação simples e direta a respeito de algum aspecto, juntamente com a avaliação quantitativa representada pelo número de pessoas que avaliaram de forma positiva e negativa este aspecto. A representação de aspectos através de sumários estruturados também permite uma forma de comparação entre produtos de uma mesma categoria, e.g. câmeras. A sumarização de opiniões [Liu 2012, Hu and Liu 2004, Liu et al. 2005] é uma técnica bastante útil para análise de opiniões de um grande número de usuários. Assim, um sumário poderia conter as principais características sobre determinados aspectos bem como uma escala de aceitação de um determinado produto [Liu 2012].

A análise de sentimentos multilíngues visa identificar e classificar opiniões em múltiplos idiomas. No contexto deste trabalho, a mineração de opinião multilíngue ocorre quando utilizamos revisões de um mesmo produto disponível em outros idiomas.

[Guo et al. 2010] utilizam um método orientado a aspectos que para encontrar as diferenças entre opiniões multilíngues. Os autores apresentam o modelo CLaSA (*Cross-lingual Latent Semantic Association*) que permite identificar as associações latentes das características dos produtos utilizando corpus de revisões multilíngues. O modelo permite classificar todas estas características e sumarizar as diferenças de opiniões sobre determinados aspectos. [Zhou et al. 2012] utilizam um método baseado em CRF (*conditional random fields*) que utiliza dados rotulados em inglês para extrair o alvo da opinião em diferentes idiomas. Esta proposta é bastante semelhante a de [Wan 2009], no entanto, utiliza

dois modelos diferentes de treinamento baseado em corpus não rotulado em chinês.

Visto que a maioria dos trabalhos utiliza corpus em inglês, [Balahur and Turchi 2014] utilizaram tradução automática para criação de corpora multilíngues e métodos supervisionados para a classificação de sentimentos. Assim foram avaliados a qualidade da tradução e o desempenho dos algoritmos de aprendizagem para identificação de polaridade em diferentes corpora. Os autores concluem que sistemas de tradução apresentam um bom nível de maturidade e podem ser utilizados para construir sistemas de classificação com resultados comparáveis ao obtido em inglês.

3. Caracterização da Contribuição

No melhor de nosso conhecimento, não existem abordagens que utilizam, revisões de um idioma de origem para fornecer ou complementar revisões de um idioma alvo. Os trabalhos relacionados à mineração de opinião multilíngue abordam principalmente problemas relacionados à classificação de sentimentos e comparação de revisões. Assim, a principal contribuição deste trabalho é desenvolver um sumário direcionado a produtos que não contêm revisões no idioma alvo, sendo necessário utilizar revisões que estão disponíveis em outros como origem. Nesta abordagem, a agregação e o resumo poderiam ser empregados em outros contextos, tais como, notícias, eleições, pontos turísticos, restaurantes, hotéis, etc.

A fim de atingir os objetivos propostos, será necessário realizar uma série de atividades. A Figura 1 ilustra tais atividades e como elas se relacionam. Como entrada (1), existe um produto em um determinado idioma alvo (no nosso caso, português). A seguir, é executada uma consulta a produtos que contêm grande quantidade de revisões em idiomas origem (idiomas estrangeiros). Uma lista de candidatos é gerada (2). Seleciona-se a partir desta lista, o produto correspondente. Isto é necessário pois os produtos podem ter nomes diferentes em idiomas diferentes. Na etapa 3 ocorre a coleta e seleção de revisões no idioma origem. Atualmente, existem grandes repositórios de revisões[1] e métodos par coleta de revisões automaticamente[2]. Devido à grande quantidade de ruído nos textos, torna-se necessário remover e corrigir parte do conteúdo para melhorar a qualidade do processamento.

Assim, o pré-processamento (etapa 4) também inclui tarefas essencialmente relacionadas ao processamento de linguagem natural, tais como, *tokenização* de palavras, etiquetagem gramatical (*part-of-speach* tagging) e árvore de dependência sintática. A identificação e a extração de aspectos e palavras de opinião incluem a combinação de diversas técnicas de extração, em sua maior parte estão disponíveis no estado da arte (conforme abordado nas Seções 2 e 4).

A geração do sumário (etapa 6) consiste em agrupar os aspectos extraídos e atribuir a pontuação média das palavras de opinião associadas a eles. Além disso, nesta etapa ocorre a tradução e organização do sumário. A saída é um sumário dos aspectos extraídos a partir das revisões nos idiomas origem traduzido para o idioma alvo.

[1]https://snap.stanford.edu/data/web-Amazon.html

[2]http://www.esuli.it/fossil/repo/amazonReviewsDownloader/index

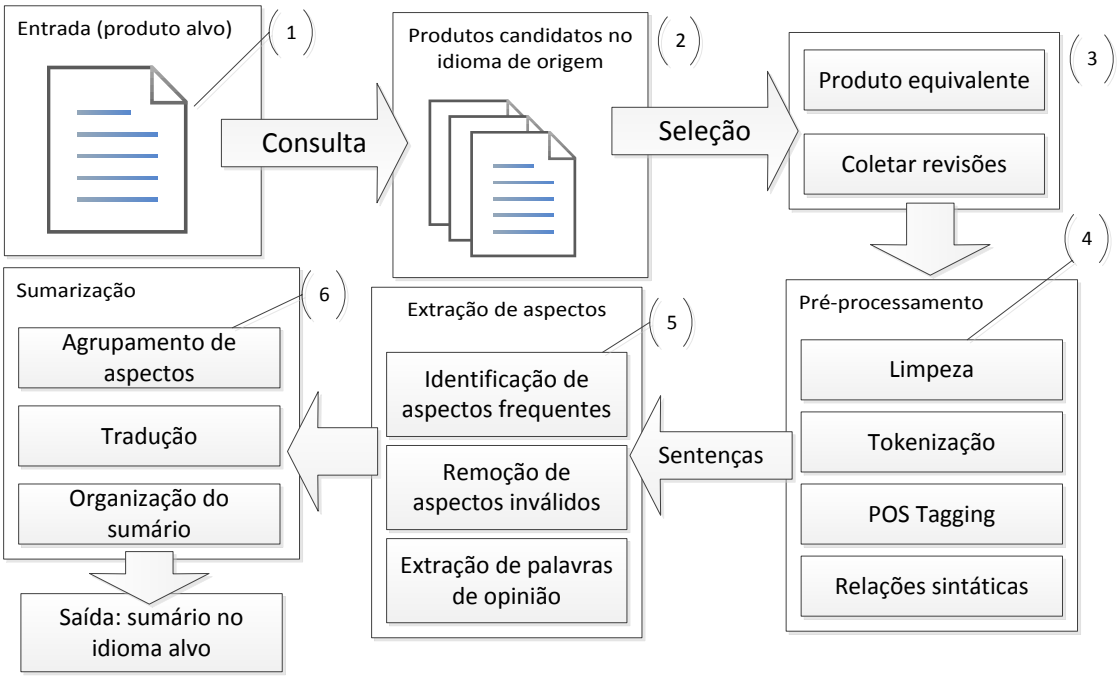


Figura 1. Fluxo de Atividades

4. Experimentos preliminares

A eficácia dos métodos para sumarização de aspectos depende da eficácia dos métodos de extração de aspectos. Assim, os primeiros experimentos realizados são direcionados à extração de aspectos em revisões. O objetivo destes experimentos é analisar a viabilidade dos métodos disponíveis no estado da arte.

4.1. Extração de aspectos utilizando relações sintáticas

Neste experimento, utilizou-se a metodologia descrita em [Qiu et al. 2011], que consiste na extração de aspectos através da relação sintática destes e suas respectivas palavras de opinião. A principal vantagem deste método em relação aos anteriores é a ausência de recursos externos, com exceção das palavras-sementes utilizadas na inicialização. Assim, trata-se de um método semi-supervisionado. Além disso, é baseado na observação de existe uma relação natural entre os palavras de opinião e aspectos, principalmente pelo fato de que opiniões são utilizadas para caracterizar aspectos.

As relações podem ser identificadas através de um analisador sintático (*parser*) baseado em dependências gramaticais (e.g. Figura 2). Entretanto, neste método são consideradas somente as dependências diretas, devido principalmente à grande quantidade de ruído presente nestas coleções e a dificuldade em processar textos informais. Além disso, são consideradas como palavras de opinião somente adjetivos (JJ), adjetivos comparativos (JJR) e adjetivos superlativos (JJS), assim chamados simplificadamente de $\{\text{JJ}\}$; e aspectos são substantivos (NN) e substantivos pluralizados (NNS), chamados de $\{\text{NN}\}$. As relações entre aspectos e opiniões incluem modificadores (mod), sujeito (subj), objeto (obj) e descrição (desc), denominadas como $\{\text{MR}\}$; e as relações entre aspectos inclui somente conjunções $\{\text{CONJ}\}$.

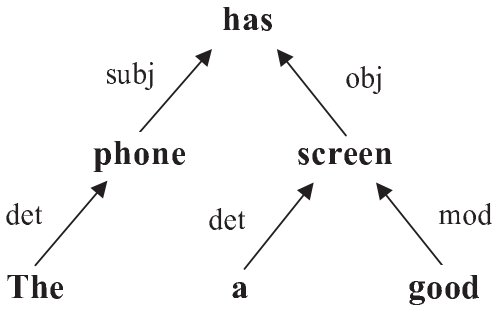


Figura 2. Exemplo de árvore de dependências [Liu et al. 2013]

A partir destas definições, o método é dividido em quatro sub-tarefas: (1) extrair aspectos usando palavras de opinião; (2) extrair aspectos usando aspectos extraídos; (3) extrair palavras de opinião usando aspectos extraídos; (4) extrair palavras de opinião utilizando palavras de opinião extraídas. Estas sub-tarefas são realizadas através de oito regras que extraem palavras de opinião e aspectos independentemente do domínio.

Tomando como exemplo a regra $R1_1$, dada uma palavra de opinião *O*, a palavra w, onde POS$(w) \in \{NN\}$ que satisfaça a relação *O-Dep* é extraída como aspecto. Por exemplo, a sentença *The phone has a good screen*, cuja árvore de dependência é apresentada na Figura 2, se a palavra *good* é uma palavra de opinião conhecida, e ela depende de *screen* através de uma relação *mod* que está contida em $\{MR\}$ e *screen* é marcada como *NN*, $R1_1$ será aplicada para extrair *screen* como um aspecto.

4.2. Identificação de aspectos incorretos

Neste experimento, utilizou-se a metodologia descrita em [Liu et al. 2013], que é uma extensão do método [Qiu et al. 2011] utilizado no experimento anterior com o propósito de remover aspectos não significativos e de alta frequência. Para isso, é introduzido o conceito de *general words* que são palavras extraídas de um dicionário. O método consiste de duas etapas: extração de aspectos candidatos e remoção de aspectos incorretos.

A extração de aspectos segue o método descrito na seção 4.1. A remoção de aspectos baseia-se em uma lista de palavras comuns que não são aspectos. Esta lista utiliza o dicionário SentWordNet no qual (*general words*) iniciais são "person", "place" e "thing". Assim, tendo uma *general word* T, todos os sinônimos de T são *general words* e todos os substantivos contidos na definição de T também são *general words*.

Nos experimentos realizados, foram utilizados os dados rotulados fornecidos pelos autores [Qiu et al. 2011, Liu et al. 2013][3] e uma segunda coleção[4], somando 14 produtos. Os dados pré-processados resultaram em 8204 sentenças e 3399 aspectos. A precisão média obtida com a extração de aspectos foi 69%, enquanto que ao remover aspectos incorretos obtivemos 80%. Este ganho também refletiu na F-Measure, onde obtivemos 74% e 79%. A revocação, por outro lado, apresenta uma pequena queda (em torno de 2%), principalmente porque alguns aspectos rotulados são *general words* e são eliminados. Foram considerados como aspectos válidos somente que estavam presente nas frases, ou seja, somente aspectos explícitos.

[3]http://www.cs.uic.edu/ liub/FBS/CustomerReviewData.zip
[4]http://www.cs.uic.edu/ liub/FBS/Reviews-9-products.rar

5. Conclusão

A extração de aspectos é uma tarefa fundamental para o processamento de revisões. Assim, a implementação das etapas de pré-processamento e identificação de aspectos está praticamente concluída, visto que são necessários testes em outras coleções e avaliação manual dos resultados.

As próximas etapas envolvem o casamento de produtos em diferentes sites e idiomas, enquanto que a segunda trata da polarização de aspectos e sumarização. Sobre o casamento de produtos ainda é necessário pesquisar métodos para identificar candidatos. Foram estudados diversos trabalhos que tratam da atribuição de polaridade a aspectos, inclusive os métodos apresentados nos experimentos.

Agradecimentos: Este trabalho foi parcialmente financiado pelo CNPq projeto 478979/2012-6. A. U. Kauer é aluno bolsista do CNPq.

Referências

Balahur, A. and Turchi, M. (2014). Comparative experiments using supervised learning and machine translation for multilingual sentiment analysis. *CSL*, 28(1):56–75.

Guo, H., Zhu, H., Guo, Z., Zhang, X., and Su, Z. (2010). Opinionit: A text mining system for cross-lingual opinion analysis. In *CIKM*, pages 1199–1208.

Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *SIGKDD*, pages 168–177.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.

Liu, B., Hu, M., and Cheng, J. (2005). Opinion observer: analyzing and comparing opinions on the web. In *WWW*, pages 342–351.

Liu, Q., Gao, Z., Liu, B., and Zhang, Y. (2013). A logic programming approach to aspect extraction in opinion mining. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, pages 276–283.

Moghaddam, S. and Ester, M. (2010). Opinion digger: an unsupervised opinion miner from unstructured product reviews. In *CIKM*, pages 1825–1828.

Qiu, G., Liu, B., Bu, J., and Chen, C. (2011). Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37(1):9–27.

Steinberger, R. (2012). A survey of methods to ease the development of highly multilingual text mining applications. *Language Resources and Evaluation*, 46(2):155–176.

Wan, X. (2009). Co-training for cross-lingual sentiment classification. In *ACL*, pages 235–243.

Zhou, X., Wan, X., and Xiao, J. (2012). Cross-language opinion target extraction in review texts. In *ICDM*, pages 1200–1205.

paper:66

PVFS-Store - Um repositório chave-valor com garantia de localidade

Ricardo M. Maeda[1]
Orientadora: Carmem Satie Hara[1]

[1]PPGInf - Programa de Pós-Graduação em Informática
Departamento de Informática – Universidade Federal do Paraná
Caixa Postal 19.081 – 81.531-990 – Curitiba – PR – Brasil

{rmmaeda, carmem}@inf.ufpr.br

Nível: Mestrado

Ingresso no programa: Março/2013

Previsão de conclusão: Março/2015

Etapas concluídas: Créditos em disciplinas

Resumo. *Há uma demanda crescente por soluções distribuídas de armazenamento, que preservem a localidade dos dados nos servidores espalhados geograficamente. Grande parte das soluções NoSQL distribuídas são baseadas em uma DHT e possuem pouco ou nenhum controle sobre a colocação dos dados. A garantia da localização é importante para permitir que as informações sejam posicionadas próximas às aplicações consumidoras e agrupadas semanticamente para acessos otimizados. Este trabalho propõe o PVFS-Store, um sistema de armazenamento distribuído baseado no modelo chave-valor, cujo objetivo é permitir o controle da localidade dos dados pela aplicação.*

Palavras-chaves. *Armazenamento distribuído em nuvem, sistemas de arquivos distribuídos, repositório chave-valor, NoSQL, proximidade dos dados, localidade dos dados*

1. Introdução

O crescimento vertiginoso no volume de dados manipulados atualmente vem trazendo desafios consideráveis no desenvolvimento de soluções escaláveis e globalmente distribuídas. Estima-se que até 2020 a quantidade de informação digital passará de 130 exabytes (cálculo estimado em 2005) para 40.000 exabytes gerados por ano. Aproximadamente 40% desta informação será armazenada na nuvem[1]. Manter a escalabilidade e a consistência dos dados distribuídos globalmente são desafios crescentes e acompanhados pela comunidade científica.

Implementações tradicionais de bases de dados distribuídas estudadas nas últimas décadas não apresentaram escalabilidade aceitável. Isso se deve em grande parte aos impactos em desempenho causados pelos custos da sincronização e tratamento de falhas. Tais implementações acabaram não sendo utilizadas amplamente na indústria [Agrawal et al. 2010], o que resultou no aparecimento de soluções projetadas para escalar horizontalmente e prover operações simples de leitura/escrita. Estas novas soluções são referenciadas como bases de dados NoSQL (*Not Only SQL*).

A maior parte destas soluções NoSQL optou pelo armazenamento baseado em DHT (*Distributed Hash Table)* [Wehrle et al. 2005]) como mecanismo de distribuição e localização dos dados pelo fato de proverem escalabilidade, tolerância a falhas e alta disponibilidade. A distribuição é fundamentada através de uma função de espalhamento, o que torna o controle da localidade dos dados pela aplicação muito baixo ou praticamente nulo. Em sistemas distribuídos geograficamente a proximidade das informações com as aplicações usuárias é fator essencial para evitar a alta latência de redes WAN (*Wide Area Network*) [Corbett et al. 2013].

Devido à DHT ser desenvolvida para dar suporte a consultas de correspondência exata da chave, ao efetuar buscas baseadas em intervalos ou conjunto de valores, múltiplos servidores acabam sendo acessados. Métodos de indexação sobre a DHT foram propostos [Tang et al. 2010], porém o intuito destas abordagens é possibilitar buscas por intervalo e não diminuir o acesso a múltiplos servidores. Outra alternativa é a adoção de funções de espalhamento que mantêm a ordem lexicográfica das chaves como encontrado no Scalaris[2].

Uma outra consequência da adoção de uma DHT é o fato da alocação dos dados ser realizada de maneira aleatória. Isto faz com que a aproximação de dados afins não seja considerada. O agrupamento de dados relacionados já é explorado nos SGBDs relacionais tradicionais, através de índices cluster e particionamento por intervalos para aproximação física dos blocos de dados. Em sistemas distribuídos globalmente, o mesmo conceito de agrupar informações semanticamente próximas em um mesmo servidor também é benéfica. Quando as informações acessadas em conjunto são agrupadas, a quantidade de servidores envolvidos na operação é minimizada, evitando transações distribuídas com protocolos de alto custo computacional [Shute et al. 2013].

Permitir o controle sobre a localidade dos dados é, portanto, essencial para a escalabilidade e desempenho da solução em ambientes distribuídos geograficamente. O objetivo deste trabalho é implementar um repositório distribuído baseado em um modelo

[1]http://idcdocserv.com/1414

[2]https://code.google.com/p/scalaris

chave-valor, cuja localidade seja garantida e controlada pela aplicação. A ideia primária é avaliar os benefícios do agrupamento de informações correlatas em um mesmo servidor e da proximidade dos dados com a aplicação consumidora.

Este trabalho está organizado da seguinte maneira: o capítulo 2 aborda os trabalhos relacionados, o capítulo 3 descreve a proposta de implementação deste trabalho e por fim o capítulo 4 apresenta as considerações finais.

2. Trabalhos Relacionados

Entre os sistemas de armazenamento distribuído presentes na literatura e relacionados com este trabalho, existem os sistemas de arquivos distribuídos, as bases de dados relacionais distribuídas e os sistemas NoSQL. Uma importante característica destas soluções é a forma como os dados são dispersos. Em geral, elas utilizam uma distribuição homogênea e uniforme dos dados.

Os sistemas de arquivos distribuídos apresentam uma estrutura não centralizada e os dados são dispersos em um conjunto de servidores, que compõem o sistema de armazenamento, aumentando consideravelmente a capacidade computacional da solução. Os clientes e aplicações usuárias não possuem acesso direto à estrutura de disco subjacente e a interação é realizada através de um protocolo pré-estabelecido. A dispersão dos arquivos é transparente e é encargo da solução distribuída localizar o arquivo e transportá-lo até a aplicação que solicitou a informação.

Soluções de código aberto como HDFS (*Hadoop Distributed File System*) [Shvachko et al. 2010], Ceph [Weil et al. 2006] e PVFS [Ross et al. 2000], ou proprietário como GFS (*Google File System*) [Ghemawat et al. 2003] não são em essência estruturados para dar ciência à aplicação sobre onde o arquivo ou um fragmento dele será armazenado. O HDFS e o GFS não implementam meios efetivos de alocação dos arquivos. Eles dividem os arquivos em fragmentos e estes são alocados em servidores com maior disponibilidade de recursos (menor carga no caso do HDFS ou menor uso de armazenamento no caso do GFS). O Ceph é um sistema de armazenamento distribuído baseado em objetos, que são espalhados por uma função de dispersão sobre um número de identificação associado a cada objeto. Este sistema provê uma flexibilidade maior, quando comparado às outras soluções, pois o administrador pode definir políticas de colocação dos objetos na estrutura do sistema distribuído (por exemplo, discos, servidores, *datacenters*, etc). O PVFS é proposto para fragmentação dos arquivos e distribuição uniforme dos fragmentos para desempenho. Ele possui uma forma efetiva de garantia da colocação do arquivo em determinado servidor. Esta colocação é definida através da interface do PVFS, na qual é possível especificar o nome do servidor no momento da criação do arquivo. Isto motivou a sua utilização como repositório de armazenamento, para estudo deste trabalho de mestrado.

Para os sistemas NoSQL, o modelo de armazenamento chave-valor é encontrado nas soluções MemcacheDB[3] e Amazon DynamoDB[4]. Elas utilizam DHT como mecanismo de dispersão e o fato da colocação das informações considerar uma função de espalhamento (*hash*), o posicionamento dos dados em determinado servidor não é garantido. Outros sistemas baseados em chave-valor,

[3]http://memcachedb.org

[4]http://aws.amazon.com/dynamodb

com noções de localidade para distribuição dos dados, são explorados nos trabalhos [Ribas et al. 2011, Arnaut et al. 2011, Schroeder et al. 2012] e estão presentes em soluções comerciais envolvendo servidores espalhados geograficamente, como Cassandra [Lakshman and Malik 2010] e Spanner [Corbett et al. 2013]. A localidade dos dados nestes repositórios distribuídos tem sido proposta com o intuito de permitir a proximidade das aplicações com seus dados, e minimizar as requisições e controle de acesso quando múltiplos servidores são necessários para atender uma consulta do usuário. Estas soluções, que adotaram noções de localidade na distribuição dos dados, são baseadas em um modelo, em que a aplicação utiliza a ordem lexicográfica das chaves para organização das informações. Ao utilizá-las para armazenamento sobre um sistema DHT com distribuição baseada em intervalo, a criação das chaves com prefixos comuns permite à aplicação o agrupamento dos dados em servidores próximos ou no mesmo servidor. Porém não há garantias de que chaves similares necessariamente serão alocadas em um único servidor.

As soluções existentes não proveem garantias de localidade na alocação das informações. As soluções que mais se aproximam em atender este requisito distribuem os dados agrupando pela ordem lexicográfica das chaves. Esta abordagem obriga a aplicação a modificar ou adequar as chaves, adicionando prefixos a elas para possibilitar seus agrupamentos e, apesar de as aproximarem, não há garantias na localidade delas no mesmo servidor. A ausência de uma solução distribuída com suporte a localidade dos dados motivou a proposta do PVFS-Store, apresentado a seguir.

3. PVFS-Store

PVFS-Store é uma proposta de implementação de repositório de dados distribuído, cujo armazenamento possui como base o modelo chave-valor. Este modelo é utilizado como estrutura de armazenamento físico para grande parte das soluções NoSQL citadas na seção 2 devido a sua flexibilidade, simplicidade e escalabilidade.

O PVFS-Store permite a alocação de um conjunto de pares chave-valor agrupados em uma única estrutura, cuja localidade é ministrada de maneira controlada e orientada pela aplicação usuária do sistema. Esta estrutura é denominada *bucket* e ela representa a unidade básica de armazenamento e transferência da solução.

3.1. Modelo

A arquitetura do sistema satisfaz a separação em níveis dos SGBDs tradicionais. A camada física descreve as estruturas físicas do banco de dados e é implementada sobre o repositório distribuído, PVFS-Store. A camada lógica abstrai os detalhes físicos de armazenamento. Ela se concentra em descrever como os dados são estruturados e como eles são apresentados para a aplicação. Para garantir a compatibilidade com outras soluções NoSQL, o modelo chave-valor é adotado nesta camada.

Uma aplicação sobre o PVFS-Store consegue acessar este repositório, através de uma interface de manipulação dos atributos chave-valor. Ela inclui além destes atributos, alterações e adições de métodos para prover suporte ao *bucket* e ao servidor:

- *create_bucket(bucket, servidor)*: criação e inicialização do *bucket* em um *servidor* do repositório.
- *drop_bucket(bucket)*: remoção do *bucket* juntamente com as chaves.

- *put_pair(chave, valor, bucket)*: inclusão de um par *chave-valor* ao *bucket*.
- *get_pair(chave)*: obtenção de um valor a partir de uma *chave*.
- *rem_pair(chave)*: exclusão de um par chave-valor a partir da *chave*.

Tais instruções são similares às encontradas nos sistemas DHT, com uma importante diferença de que a localidade dos registros é controlada pela aplicação, através dos métodos de criação de um *bucket* e inclusão de um par chave-valor. Além disso, uma interface e repositório adicionais serão desenvolvidos para armazenar as informações do metadado.

- *put_md(chave, bucket)*: inclusão no metadado da *chave* e do *bucket*, onde ela será inserida. Este método será chamado toda vez que a função *put_pair* for executada.
- *get_md(chave)*: obtenção do *bucket*, onde está localizada a *chave*. Ele será executado nas chamadas à função *get_pair*.
- *rem_md(chave)*: exclusão da *chave* do metadado. Sempre que um par chave-valor for removido através do *rem_pair* esta instrução será executada.

A disposição do PVFS-Store em relação à aplicação e ao sistema distribuído pode ser visualizado na figura 1. Entre a aplicação e a interface do PVFS-Store as informações sobre bucket, chave e valor são utilizadas na interação com o repositório de dados. Já a comunicação do PVFS-Store com o PVFS utiliza uma API do sistema de arquivos distribuído e possui como unidade o *bucket*.

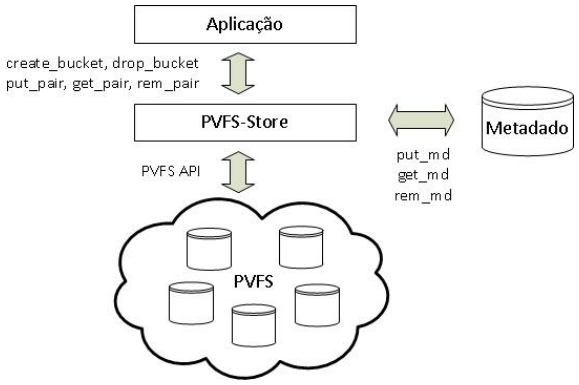


Figura 1. Arquitetura do PVFS-Store

Na implementação do PVFS-Store, um *bucket* é fisicamente um arquivo no sistema de arquivos PVFS. Todo *bucket* está associado a um servidor, necessário para a colocação deste arquivo na solução distribuída.

As informações sobre *buckets* e servidores são armazenadas em uma estrutura de dados compondo o metadado do PVFS-Store. O metadado é responsável pela associação de uma chave ao *bucket* e ao servidor, nos quais ela está armazenada. A dispersão e localização dos dados são controladas pela aplicação através deste metadado, ao invés de uma DHT.

Cada *bucket* possui no cabeçalho meta-informações, como número máximo de chaves, um mapa de bits para controle do espaço livre e uma estrutura de tamanho fixo contendo as chaves existentes no *bucket*. Os valores associados às chaves são salvos justapostos, no corpo do *bucket*, após o cabeçalho.

O PVFS-Store está sendo desenvolvido sobre o sistema de arquivos distribuídos PVFS.

3.2. Estudo de Caso

Para avaliação do PVFS-Store como um repositório distribuído será implementado um módulo de armazenamento customizado do MySQL[5] (*MySQL Custom Storage Engine*), que utiliza o PVFS-Store como armazenamento físico. A escolha do MySQL se deve à sua arquitetura modular, na qual é possível acoplar módulos de armazenamento de maneira transparente para a aplicação.

Como base deste estudo de caso serão utilizadas abordagens semelhantes às existentes em [Ribas et al. 2011] e [Chang et al. 2008] para mapeamento das relações em pares chave-valor. O módulo de armazenamento MySQL sobre o PVFS-Store será responsável por este mapeamento. Todas as operações existentes na base de dados relacional deverão ser convertidas para instruções do repositório chave-valor. Desta forma, tuplas deverão ser mapeadas para chave-valor no momento de uma operação de inserção e de forma inversa ao obtê-las.

Outras aplicações sobre o PVFS-Store podem ser desenvolvidas. O modelo proposto de chave-valor com garantia de localidade (por meio de *buckets*) permite utilizá-lo como repositório de uma aplicação NoSQL ou até implementações mais complexas de SGBD sobre ele, que adotem fragmentações horizontais (por linhas) ou verticais (por colunas).

Como validação da proposta deste trabalho e realização de experimentos, a solução será comparada a um armazenamento sobre uma DHT, cuja distribuição é uniforme e homogênea.

4. Considerações Finais

Este trabalho propõe o desenvolvimento de uma solução distribuída armazenada sobre um repositório chave-valor e um metadado para localização das chaves nos respectivos *buckets*. Esta solução pretende avaliar os impactos do controle da localidade dos dados em um ambiente distribuído e compará-la com uma abordagem que não leva em consideração a localidade. O PVFS-Store possibilita à aplicação a alocação exata em um determinado servidor das tuplas no repositório distribuído. As soluções atuais possuem pouco ou nenhum controle sobre a colocação das informações nos servidores e implementam mecanismos de distribuição uniforme dos dados ou baseados em ordenação lexicográfica.

Com a abordagem proposta é possível otimizar aplicações distribuídas globalmente, aproximando os dados das aplicações usuárias. O agrupamento das informações semanticamente relacionadas evita o acesso a múltiplos servidores e ajuda a diminuir a incidência de transações distribuídas.

Portanto, é esperado que a distribuição controlada pela aplicação traga benefícios em uma rede distribuída geograficamente. O repositório PVFS-Store e seu metadado estão sendo implementados sobre um ambiente em nuvem. Para a validação da proposta será desenvolvido um módulo de armazenamento MySQL sobre o PVFS-Store.

[5]http://www.mysql.com

O principal desafio na implementação deste trabalho é o desenvolvimento de um metadado descentralizado. Além disso, a especificação de uma interface chave-valor com suporte a localidade dos dados e o seu desenvolvimento sobre um repositório distribuído são desafios importantes na implementação desta solução.

Referências

Agrawal, D., El Abbadi, A., Antony, S., and Das, S. (2010). Data management challenges in cloud computing infrastructures. In *Databases in Networked Information Systems*.

Arnaut, D. E., Schroeder, R., and Hara, C. S. (2011). Phoenix: A relational storage component for the cloud. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*.

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R. E. (2008). Bigtable: A distributed storage system for structured data.

Corbett, J. C., Dean, J., Epstein, M., Fikes, A., Frost, C., Furman, J., Ghemawat, S., Gubarev, A., Heiser, C., Hochschild, P., et al. (2013). Spanner: Google's globally distributed database.

Ghemawat, S., Gobioff, H., and Leung, S.-T. (2003). The google file system. In *ACM SIGOPS Operating Systems Review*.

Lakshman, A. and Malik, P. (2010). Cassandra: a decentralized structured storage system.

Ribas, E. A., Uba, R., Reinaldo, A. P., et al. (2011). Layering a dbms on a dht-based storage engine.

Ross, R. B., Thakur, R., et al. (2000). Pvfs: A parallel file system for linux clusters. In *in Proceedings of the 4th Annual Linux Showcase and Conference*.

Schroeder, R., dos Santos Mello, R., and Hara, C. S. (2012). Affinitybased xml fragmentation. In *WebDB*.

Shute, J., Vingralek, R., Samwel, B., Handy, B., Whipkey, C., Rollins, E., Oancea, M., Littlefield, K., Menestrina, D., Ellner, S., et al. (2013). F1: A distributed sql database that scales.

Shvachko, K., Kuang, H., Radia, S., and Chansler, R. (2010). The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*.

Tang, Y., Zhou, S., and Xu, J. (2010). Light: a query-efficient yet low-maintenance indexing scheme over dhts. *Knowledge and Data Engineering, IEEE Transactions on*, 22(1):59–75.

Wehrle, K., Götz, S., and Rieche, S. (2005). 7. distributed hash tables. In *Peer-to-Peer systems and applications*.

Weil, S. A., Brandt, S. A., Miller, E. L., Long, D. D. E., and Maltzahn, C. (2006). Ceph: A scalable, high-performance distributed file system. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*.

paper:29

Searching Time Series via Semantic Annotations

Lucas Oliveira Batista[1], **Claudia Bauzer Medeiros**[1] **(advisor)**

[1] Programa de pós-graduação em Ciência da Computação do Instituto de Computação da UNICAMP – Universidade Estadual de Campinas
CEP 13083-852, Campinas - São Paulo - Brazil

lucas.batista@students.ic.unicamp.br, cmbm@ic.unicamp.br

Level: Master's Degree

Admission: March 2013

Qualifying exam: 09^{th} October 2013

Expected Conclusion: March 2015

Concluded stages: Credits; Qualifying exam; Definition of time series semantic annotation model; Implementation of first software prototype.

Future stages: Improve the software and evaluate the time series semantic annotation model; Develop algorithm to search time series via semantic annotation; Evaluate this algorithm; Write paper and dissertation; Master's degree defense;

Abstract. *Time series are used in several domains of knowledge. During their analysis, experts often create or analyze associations between time series and annotations. In order to study a problem, for example, patient behavior or crop patterns, experts need to search and correlate several time series. However, finding appropriate series related with a problem is a difficult task. Search is usually performed using a few parameters, such as series geographic location. Annotations may be used to help the search using string match. Given this scenario, this paper discusses a work in progress to design and partially develop a software framework to search time series via semantic annotations. It will support experts in the correlation of time series, foster collaboration among experts, and allow the use of Linked Data concepts to aggregate knowledge to content. This paper extends the short paper accepted for BRESCI - Brazilian Workshop e-Science 2014. The extensions include a time series semantic annotation model, implementation details, and a longer theoretical related work section.*

Keywords: *Search for time series, semantic annotation*

1. Introduction

Time series are used in several knowledge domains, such as economics (monthly unemployment rate), meteorology (daily temperature) or health (electrocardiogram). An efficient search for time series helps the task of analyzing these series, for instance, to perform forecasts, identify patterns, find the origins of certain phenomena, and so on. Experts often use series annotations to help this analysis.

An annotation establishes a relation between the annotated data and the annotation content [Oren et al. 2006]. Time series annotations are potentially made by different researchers or research groups and they are generally created in text format using annotation specific languages and stored in files. Due to the fact that there are many standards in a single domain, annotations have a large heterogeneity. It complicates the search, sharing and integration of data.

In several situations, scientists need to correlate many kinds of series to study a problem. The search for relevant series with respect to the problem is expensive and takes a long time. Conventional systems perform the search for time series using either time series similarity, parameters that are matched with time series values, or annotation textual match. However, these approaches may not be enough to find series related with a problem.

Suppose that a researcher needs to analyze the impact of corn production on local economy of a region. Several series are related to corn production, for instance, temperature, humidity, harvested amount, and so on. Searching for relevant series presents challenges, for instance, "corn" is cultivated in different parts of the world and, unless the series are georeferenced, the search by keyword "corn" will return too many series. Furthermore, each region has a different nomenclature for "corn" and many relevant series may not have this specific annotation.

Given this scenario, the goal of this work is designs and partially develops a framework to support series search via semantic annotations, thereby helping the correlation task. This paper extends the short paper accepted for BRESCI - Brazilian Workshop e-Science 2014. Extensions are: a time series semantic annotation model and implementation details (Section 5), and a longer related work section (Section 2).

2. Theoretical Basis

Annotations are data about data and they are used to provide extra information that may be relevant to data analysis [Silva 2013]. Annotations can have different information depending on the annotated data, and on who performs the annotation. Besides, annotations may be made in many formats, for example, video, audio or text. This work focuses in textual annotations of time series.

A *semantic annotation* is the description of a digital resource according to its semantics [Sousa 2010]. [Sousa 2010] defines semantic annotation as triples $< s, m, o >$, where s is the subject described, m is the label of a metadata field, and o is an ontology term that semantically describes this subject. We extend this notion to a more generic structure.

3. Related Work

The goal of this work is to provide the users with tools that help the search for time series, taking advantage of semantic annotations. Therefore, the literature review focuses on highlighting research on time series annotations (2.1) and time series search (2.2).

3.1. Time series annotation

There are many tools that support series annotations - for example, [Pressly 2008, Silva 2013]. Both enable associating multiple textual annotations to parts of a series. In the first, experts may just visualize annotations made by others. In the second, experts may collaborate modifying annotations made by others.

Such tools present a few disadvantages. First of all, they store annotations as free text, which hampers search, sharing, integration of data and their interpretation by machines. Second, they do not consider the ambiguity of annotation contents. These problems may be attacked using semantic annotations.

The Tatoo Framework [Pariente et al. 2011] enables the semantic annotation of environmental resources and has a time series semantic annotation model. However, this model is based on time series observations and has a different purpose. Tatoo performs semantic annotations to semantically represent the temporal data. Our work focus on annotations that are made by experts, attaching extra information to series.

3.2. Search for time series

According to [Gao and Wang 2009], the search for time series refers to finding, from a set of time series, the series that satisfy a given search criterion. This search may be done using a model or statistics of series, temporal dependencies, similarity among series or patterns and so on.

There are tools that perform the search for time series in a limited way, for instance, they just allow the user filter series selecting a category and a time period [Secretaria do Tesouro Nacional 2013]. Other perform search based on similarity among series using as input one time series - for instance, [Ding et al. 2008, Negi and Bansal 2005].

Still other approaches to search for time series are proposed by [Aßfalg et al. 2006, von Landesberger et al. 2009]. The former proposes a tool that searches time series using a range of values defined by the user as search parameter. The latter uses text in natural language, on financial databases. The disadvantage of these approaches is that they do not consider extra information that may be indirectly associated with series. Our work provides it by using semantics.

4. Characterization of the Contribution

This paper describes a software framework that helps users to find time series of interest by performing search via semantic annotations. The work of [Silva 2013] is used as the starting point to achieve this goal.

The proposed approach allows to solve the kinds of problems shown in Section 1. In order to return relevant results, besides text match, the search allows using extra information that may indirectly be associated to series. Figure 1 illustrates an overview

of the solution considered. In this framework, experts may add extra information to series semantically annotating parts of one or more time series. Series are stored in relational databases and may have many semantic annotations (step 1). Semantic annotations are stored in a RDF (Resource Description Framework) database (step 2) and are associated to external ontologies (step 3). It enables the use of Linked Data concepts to aggregate knowledge to annotations. Annotations are versioned through time.

Returning to the corn example presented in Section 1 and as shown in the figure 1, given a set of time series the researcher may use the string "milho" in the search for time series of interest (step 4). Navigating through ontologies that are connected to annotations, new information associated to "milho" is inferred, for example, "Zea Mays" (scientific name of corn), terms like "dentado" (that refers to texture of the corn grain), "Cercospora zeae-maydis" (fungus that affects corn), "mancha-branca" (corn disease), "espiga" (term related to corn), time period in which corn is harvested and places where corn is planted. Defining ontologies is very hard, so we intend to use known ontologies, like AGROVOC [1], which is an ontology that deals with several areas related with food and agriculture. As a consequence of using an ontology, the researcher will obtain as results (step 5) time series annotated with terms associated with "milho" (like "dentado"), temperature and humidity series for regions with corn, grain amount harvested on regions where "milho" is planted and dollar price series in the period in which "milho" is harvested. This will allow experts perform new types of correlations.

The framework supports collaboration among experts by allowing comments on the annotations (or meta-annotations). These meta-annotations are also structured; their values are a free-text values and they are associated to the versioned time series annotations. This new level of annotations records discussions over time, avoiding the repetition of time series annotations created just to register this collaboration (as made by [Silva 2013]). Therefore, new annotations will be created just when there is an agreement about the real change of content.

This framework is being validated with time series provided by EMBRAPA. These series contain Normalized Difference Vegetation Index (NDVI) values extracted from satellite images taken from Mato Grosso. Moreover, EMBRAPA series are textually annotated with information on the corresponding crop. We intend to expand our database with other kinds of time series – e.g., meteorological data. This will potentially allow retrieving heterogeneous series, given a set of terms, which are nevertheless indirectly (semantically) related. To evaluate this work, we will define a set of queries to be performed, to solve the problem described in Section 1, using semantic annotations in the search. Together with experts from EMBRAPA, we will compare this result to that provided by standard keyword based queries, to validate our proposal.

5. Ongoing Work

In order to achieve the main goal of this work, we first define a time series semantic annotation model (which we call TSSAM). Our model has the following components: (1) Subject: an URI (Uniform Resource Identifier) that represents one time series. This URI will be created by our software; (2) Annotation identifier: an URI that identifies one semantic annotation; (3) Annotation range: defines the granularity of the semantic

[1] http://aims.fao.org/standards/agrovoc/about

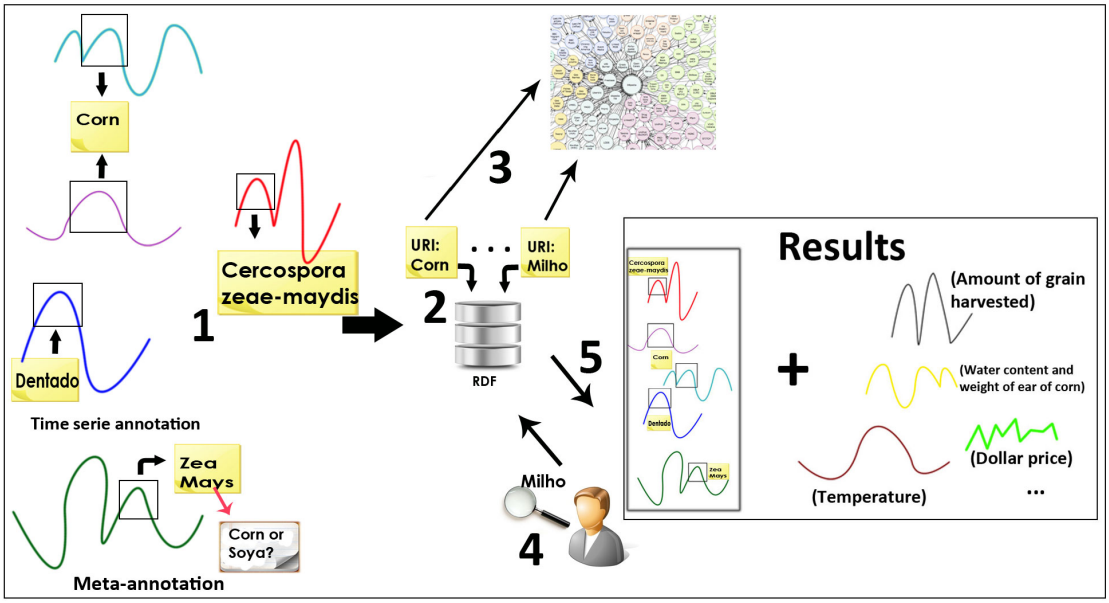


Figure 1. Overview of the model proposed

annotation, namely, a semantic annotation may be associated with all or part of one time series. This granularity will be specified by an time interval defined by the Time Ontology [2]; (4) Body: an URI representing the content of a semantic annotation. This URI is informed by the user; (5) Author: represents who creates the annotation; (6) Creation date: the date in which a semantic annotation was created; (7) Modification date: the date in which a semantic annotation was modified; (8) Version: version of a semantic annotation; (9) Time series property: represents a list of properties related with the annotated part of a time series, for instance, max point, amplitude, and so on.

Figure 2 illustrates our model in a RDF language where the ovals represent URIs and rectangles represent literals. The center of the figure contains the annotation identifier and around it are the other components of TSSAM. The "Time Series Property" is a rdf:Bag, which represents a group of literals. This model uses some properties already defined, for example, the property "annotates" relates an annotation to its resource and is defined in http://www.w3.org/2000/10/annotation-ns#annotates.

Once the TSSAM was defined, we developed an initial prototype. This first version enables the creation of one semantic annotation for parts of one time series, and one time series may have more than one annotation. As shown in figure 3, the user fills the form informing the URI that represents the annotated content (4^{th} TSSAM component), the annotation range (3^{rd} TSSAM component), and the author (5^{th} TSSAM component) in order to create a semantic annotation. The other components are automatically created by the software. On the right side it is possible to observe a graphical representation of semantic annotations for that time series and below more details about the annotations.

[2]http://www.w3.org/TR/owl-time/

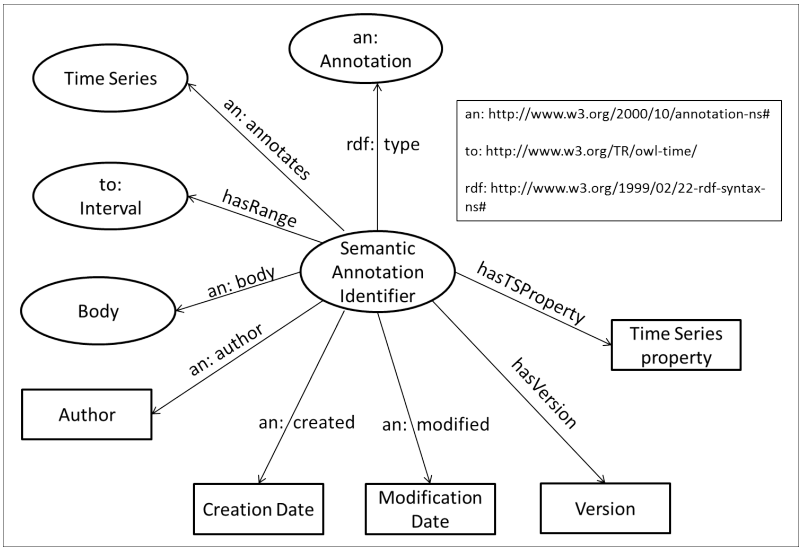


Figure 2. Time series semantic annotation model proposed

6. Conclusions and Future Work

This paper shows an ongoing research that helps experts share information, collaborate in production of data of common interest and search time series returning more relevant results. The framework proposed uses semantic annotations as a new possibility to search for time series, taking advantage of extra information attached to the series. As future work, we intend to improve our first prototype implementation and evaluate our semantic annotation model with experts from EMBRAPA. After that, the next stage is to develop algorithms to search time series via annotations, implement and evaluate them.

Acknowledgments Work partially financed by FAPESP (2014/07303-1), FAPESP/Cepid in Computational Engineering and Sciences (2013/08293-7), the MSR FAPESP Virtual Institute (NavScales project), CNPq (MuZOO Project), FAPESP-PRONEX (eScience project), INCT in Web Science, and grants from CNPq. We thank EMBRAPA for the data.

References

Aßfalg, J., Kriegel, H.-P., Kröger, P., Kunath, P., Pryakhin, A., and Renz, M. (2006). Tquest: threshold query execution for large sets of time series. In *Advances in Database Technology-EDBT 2006*, pages 1147–1150. Springer.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. (2008). Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proc. VLDB Endow.*, 1(2):1542–1552.

Gao, L. and Wang, X. S. (2009). Time series query. In Liu, L. and Özsu, M. T., editors, *Encyclopedia of Database Systems*, pages 3114–3119. Springer US.

Negi, T. and Bansal, V. (2005). Time series: Similarity search and its applications. In *Proceedings of the International Conference on Systemics, Cybernetics and Informatics: ICSCI-04, Hyderabad, India*, pages 528–533.

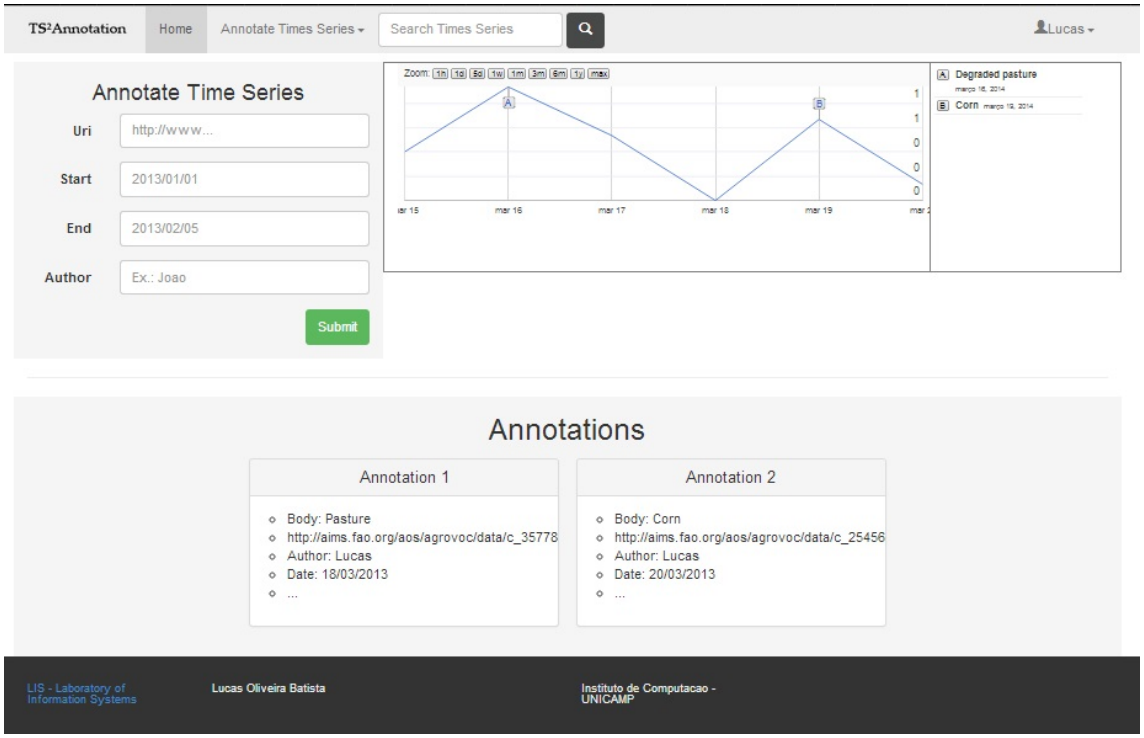


Figure 3. Time series semantic annotation prototype screen

Oren, E., Möller, K., Scerri, S., Handschuh, S., and Sintek, M. (2006). What are semantic annotations. *Relatório técnico. DERI Galway*.

Pariente, T., Fuentes, J. M., Sanguino, M. A., Yurtsever, S., Avellino, G., Rizzoli, A. E., and Nešić, S. (2011). A model for semantic annotation of environmental resources: the tatoo semantic framework. In *Environmental Software Systems. Frameworks of eEnvironment*, pages 419–427. Springer.

Pressly, Jr., W. B. S. (2008). Tspad: a tablet-pc based application for annotation and collaboration on time series data. In *Proc. of the 46th Annual Southeast Regional Conference*, ACM-SE 46, pages 166–171, New York, NY, USA. ACM.

Secretaria do Tesouro Nacional (2013). Séries temporais. `http://www3.tesouro.fazenda.gov.br/series_temporais/principal.aspx`. Acessado: 03-09-2013.

Silva, F. H. (2013). Serial annotator: Managing annotations of time series. Master's thesis, UNICAMP. Supervisor Medeiros.

Sousa, S. R. (2010). Gerenciamento de anotações semânticas de dados na web para aplicações agrícolas. Master's thesis, UNICAMP. Supervisor Medeiros.

von Landesberger, T., Voss, V., and Kohlhammer, J. (2009). Semantic search and visualization of time-series data. In *Networked Knowledge-Networked Media*, pages 205–216. Springer.

paper:11

Sintonia fina baseada em ontologia: o caso de visões materializadas

Aluno - Rafael Pereira de Oliveira[1]
Orientador - Sérgio Lifschitz[1]

[1] Departamento de Informática
Programa de Pós-graduação / Mestrado Acadêmico
Pontifícia Universidade Católica do Rio de Janeiro (PUC-RIO)
Rio de Janeiro, RJ - Brasil - Caixa Postal 38097 – 22451-900

{rpoliveira@inf.puc-rio.br, sergio@inf.puc-rio.br}

Defesa de proposta 08/2014
Data prevista de defesa 12/2014
Ingresso no curso 2013.1 – Conclusão prevista 2014.2
Etapas futuras: Defesa de proposta, defesa de dissertação.

Resumo. *Há uma crescente demanda por ferramentas que automatizem tarefas complexas relacionadas com sistemas computacionais. No caso particular de sistemas de informação e bancos de dados, especificamente no que diz respeito ao SGBD, podemos citar as tarefas relacionadas às atividades de sintonia fina (database tuning). Existe uma proposta de um framework, chamado Outer-tuning, que serve para apoiar a sintonia fina adicionando semântica ao processo. Trata-se de uma abordagem que oferece transparência e confiabilidade acerca das alternativas disponíveis para possíveis cenários no SGBD, possibilitando a combinação de estratégias independentes para solucionar um único problema e permitindo a extração de justificativas para possíveis ações de sintonia fina. Através do uso de uma ontologia específica para sintonia fina de bancos de dados relacionais, é possível entender os conceitos envolvidos e gerar, de maneira automática, novas práticas de sintonia fina, que podem ser obtidas a partir das práticas existentes (uso de inferências) ou de novas regras e conceitos que venham a surgir no futuro. Essa dissertação de mestrado propõe, inicialmente, a implementação do framework Outer-Tuning para apoiar o DBA (e possivelmente outros usuários) nas escolhas envolvidas na atividade de sintonia fina. Para isso é formalizada uma arquitetura de software adequada para atender os requisitos funcionais. Este trabalho também inclui a extensão da ontologia de domínio para contemplar soluções de sintonia fina com o uso de visões materializadas. Desta forma, é possível propor o uso de heurísticas na ontologia de tarefas para realizar a sintonia fina não somente por meio de índices mas também através do uso de visões materializadas. Essa ontologia pode ser usada no framework Outer-Tuning de forma a permitir uma avaliação experimental da qualidade dos resultados obtidos utilizando-se um benchmark.*

Palavras-chave: sintonia fina, visões, materialização, ontologia, framework.

1. Fundamentação teórica

Apresenta-se aqui, os principais conceitos envolvidos no trabalho. Descreve-se brevemente[1] os conceitos de sintonia fina, ontologia e visão materializada para um melhor entendimento da proposta de pesquisa apresentada.

As aplicações e os usuários que acessam bases de dados demandam cada vez mais que os Sistemas Gerenciadores de Bancos de Dados (SGBDs) respondam às suas requisições no menor tempo de resposta possível, e que tenham uma maior vazão (número de transações por unidade de tempo)[Shasha and Bonnet 2002]. Para atingir esses objetivos, administradores de bancos de dados (DBAs) utilizam estratégias de sintonia fina (*tuning*) sobre o projeto físico do banco de dados, ajustando de suas configurações, parâmetros, projeto físico, seleção de estruturas de acesso, duplicação de estruturas físicas, determinação dos objetos a serem particionados e seus respectivos tipos de particionamento, de acordo com a carga de trabalho executada no banco [Shasha and Bonnet 2002][Bruno 2012][Ramarkrishnan and Gehrke 2008][Salles 2004].

Uma das ações de sintonia fina possível é a criação de visões materializadas (VM). Visões são tabelas "virtuais" definidas através de uma consulta sobre tabelas persistidas pelo banco de dados. O SGBD armazena as definições da consulta envolvida, mas não os seus dados retornados. Visão materializada é uma visão cujo resultado da consulta é fisicamente armazenado na base de dados e persistido como se fossem tabelas [Silberschatz et al. 2006]. Hoje já são implementadas pelos SGBDs mais populares do mercado como PostgreSQL, SQL Server, Oracle, e outros [Chirkova and Yang 2011].

Outro conceito importante para esta pesquisa é a ontologia. Na área de computação define-se ontologia como sendo um conjunto de primitivas representacionais, que modelam um domínio de conhecimento ou discurso. As primitivas são tipicamente classes (ou conjuntos de objetos), atributos (ou propriedades) e relacionamentos (ou relações entre membros de classes) [Gruber 2009].

2. Caracterização da contribuição

Encontrar uma configuração ótima do projeto físico de banco de dados para uma carga de trabalho de forma manual não é uma tarefa trivial. A sintonia fina de banco de dados tem se tornado uma atividade extremamente complexa para as aplicações atuais. É necessário um profundo conhecimento acerca dos detalhes de implementação dos sistemas gerenciadores de banco de dados (SGBDs) e da carga de trabalho submetida [Monteiro et al. 2008].

Faz parte do trabalho de um DBA descrever todas ou, se inviável, a maioria das soluções disponíveis e avaliadas em uma atividade de sintonia fina. Além da complexidade na seleção das soluções de sintonia fina o DBA enfrenta o desafio de justificar as decisões tomadas para melhorar o desempenho da execução de uma determinada carga de trabalho. Não há uma ferramenta integrada que o auxilie com argumentos suficientes para justificar as soluções escolhidas pelas ferramentas de sintonia fina, sejam automáticas ou semiautomáticas. Entretanto, um software que realize a sintonia fina de banco de dados, que apresente as alternativas que foram pensadas e, ao mesmo tempo, fundamente

[1] Sugestão dos revisores para permitir detalhamento de outras seções.

a escolha de cada uma, trará uma maior confiança em relação ao uso de ferramentas automáticas, e facilitando o trabalho do DBA.

Existe uma proposta de um *framework*, chamado *Outer-tuning*, que serve para apoiar a sintonia fina adicionando semântica ao processo, através da execução de heurísticas para este fim. O *framework outer-tuning* procura capturar a carga de trabalho e, com o apoio de uma ontologia de aplicação e do uso de uma máquina de inferências sobre os conceitos definidos, realizar a tarefa de sintonia fina (automática ou semiautomática, de acordo com a decisão do DBA) para a criação, remoção ou reindexação de índices em bancos de dados relacionais [Almeida 2013].

Cabe observar que o *outer-tuning* não foi implementado e têm-se apenas o modelo conceitual desse *framework*, envolvendo a ontologia de aplicação de sintonia fina e evidenciando heurísticas de manutenção de índices [Almeida 2013]. Por ser um *framework* definido sobre a ontologia de aplicação, basta que a mesma seja estendida para contemplar outras opções de soluções de sintonia fina.

Diante disso, o presente trabalho propõe o projeto e a implementação do *framework outer-tuning*, o refinamento da sua ontologia de tarefa e a extensão da sua ontologia de domínio para contemplar os conceitos envolvidos no uso e execução de heurísticas que realizem a sintonia fina de banco de dados com visões materializadas.

Espera-se que a ferramenta possa recomendar, e executar se solicitada, as possíveis alternativas de soluções para sintonia fina (semi)automática ao mesmo tempo que tenha o poder semântico necessário para justificar as decisões tomadas. O DBA poderá ter transparência, semântica e confiabilidade acerca das alternativas possíveis, por meio de justificativas concretas para as decisões que foram tomadas.

Além disso, o *framework outer-tuning* será capaz de realizar mais de um tipo de sintonia fina, combinando e comparando soluções distintas como índices e visões materializadas e ter um maior espaço de possíveis soluções.

As principais contribuições esperadas dessa dissertação são: a) projeto arquitetural e implementação do *framework outer-tuning*; b) definição de um processo para extensão da ontologia de domínio, que seja capaz de ser reutilizado em extensões para outros tipos de sintonia fina; c) execução da extensão da ontologia de domínio para contemplar soluções de sintonia fina com o uso de visões materializadas; d) definição de uma heurística para realizar sintonia fina através de visões materializadas que pode ser usada no *framework*; e) avaliação experimental da qualidade dos resultados obtidos a partir da execução do *framework* para execução de sintonia fina utilizando um benchmark.

3. Estado atual do trabalho

Nesta sessão, são apresentados o estado de todas as tarefas da pesquisa, a situação do desenvolvimento do *framework* e a extensão da ontologia.

3.1. Tarefas

A presente pesquisa já concluiu as seguintes tarefas:

- ✓ Pesquisa de ferramentas de sintonia fina automáticas e semiautomáticas;
- ✓ Pesquisa e classificação dos principais métodos de seleção de visões materializadas utilizados em ferramentas de sintonia fina automática e semiautomática;

- ✓ Pesquisa de máquinas de inferência que se integrem à linguagem Java;
- ✓ Implementação da estrutura base e do agente de captura da carga de trabalho;
- ✓ Integração da máquina de inferência escolhida JESS[2];
- ✓ Controle de execução das regras SWRL[3] da ontologia através da API do JESS;
- ✓ Instanciação dos conceito de domínio através da captura da carga de trabalho;

Segundo o planejamento de pesquisa, ainda estão abertas as seguintes tarefas:

- Instanciar os demais conceitos da ontologia de domínio;
- Implementar método de instanciação dos conceitos pré-condição das heurísticas;
- Implementar um método para execução das ações de heurísticas de sintonia fina, que vão resultar em comandos DDL sobre o banco de dados [Almeida 2013];
- Realizar a extensão da ontologia de domínio para contemplar VM;
- Definir a heurística de seleção de visões materializadas baseadas na heurística de benefícios proposta por [Carvalho 2011] na ontologia de tarefa do *framework*;
- Realizar testes de validação do *framework* para a execução de sintonia fina com a seleção de visões materializadas, usando o benchmark.
- Escrita, revisão e defesa da dissertação.

Em resumo, a arquitetura base do *framework* já foi montada e as referências bibliográficas levantadas.

3.2. Arquitetura da implementação

O *framework Outer-Tuning* foi projetado com a arquitetura descrita na Figura 1:

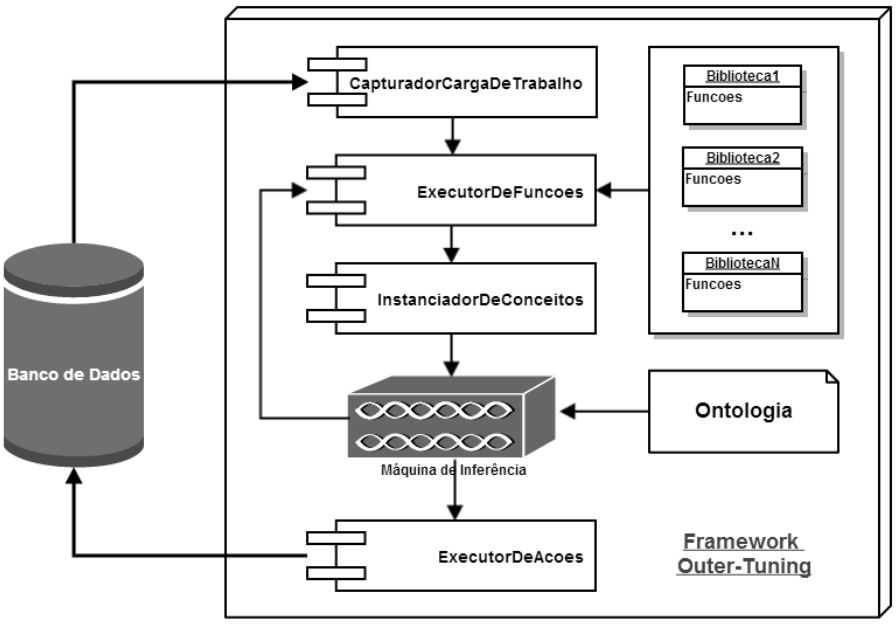


Figure 1. Arquitetura *Framework Outer-tuning*

O componente *CapturadorCargaDeTrabalho* é o responsável por adquirir a carga de trabalho em tempos pré-determinados e alimentar o *ExecutorDeFuncoes* com o resultado da captura.

[2]Jess, the Rule Engine for the JavaTM Platform. http://www.jessrules.com/jess/index.shtml
[3]SWRL Language. http://www.w3.org/Submission/SWRL

As *bibliotecas* foram projetadas para conter funções destinadas a extrair informações da carga de trabalho de acordo com as pré-condições das heurísticas utilizadas. Esse desmembramento foi realizado para que, ao se estender uma ontologia para um novo tipo de sintonia fina, o *framework* seja capaz de absorver novas funções sem a necessidade da alteração de seu código fonte. A ontologia torna-se independente da versão do *framework*, podendo ser estendida com a inclusão de biblioteca(s) de funções(s).

O componente *ExecutorDeFuncoes* recebe a) uma carga de trabalho do componente *CapturadorCargaDeTrabalho*; e b) uma lista de funções que são pré-condições de uma heurística, fornecida pela máquina de inferência. A partir daí, ele adquire as funções da lista através das *bibliotecas* de funções, executando-as utilizando a carga de trabalho como parâmetro de entrada para cada função. Feito isso, recebe o retorno das funções que serão sempre conceitos previstos na ontologia (ainda não instanciados na máquina de inferência) e os repassa para o componente *InstanciadorDeConceitos*.

O componente *InstanciadorDeConceitos* recebe os conceitos adquiridos e os instancia na máquina de inferência.

A máquina de inferência é responsável por ler a ontologia de um arquivo, fornecer a lista de funções que são pré-condição das heurísticas contidas na ontologia, e após ter todos os conceitos pré-condições instanciados, inferir as ações de sintonia fina previstas nas heurísticas, repassando-as para o componente ExecutorDeAcoes.

O componente ExecutorDeAcoes é responsável por receber da máquina de inferência as ações de sintonia fina inferidas, e executá-las no banco de dados.

3.3. Extensão da ontologia

Planeja-se implementar a heurística de seleção de visões materializadas como uma extensão da ontologia de [Almeida 2013]. Como o escopo dessa pesquisa não prevê a criação de uma nova heurística de seleção de visões materializadas, deverá utilizar-se uma heurística já descrita e validada na literatura.

A heurística escolhida para a extensão foi a heurística de benefícios, originalmente proposta para índices [Salles 2004] e posteriormente adaptada para visões materializadas [Carvalho 2011]. Os motivos que levaram a essa escolha foram: trata-se de uma heurística já utilizada para a tarefa de seleção de visões materializadas; é a base da heurística de índices que foi implementada na ontologia de [Almeida 2013]; e tem-se a ferramenta de seleção de visões materializadas [Carvalho 2011], implementada e pronta para ser utilizada como base de comparação com os resultados do *framework Outer-Tuning*.

A ideia básica da heurística de benefícios é atribuir um determinado valor a uma visão materializada, seja ela hipotética (não existe fisicamente no banco de dados) ou real. Esse valor é a diferença entre os custos obtidos através do plano de execução de uma determinada consulta da carga de trabalho com, e sem, a presença da visão materializada. Esse custo pode ser positivo, caso o uso da visão traga benefício na execução de uma consulta, ou negativo, em caso de malefício.

Esse custo é recalculado sempre que uma consulta da carga de trabalho é capturada. Caso o valor de uma visão materializada hipotética chegue a um valor maior que o custo de criação da mesma, a heurística determina que a visão materializada seja criada fisicamente no banco de dados, e tem-se então uma ação de sintonia fina.

4. Trabalhos relacionados na área

Existem muitas ferramentas para a seleção automática e semiautomática de visões materializadas que utilizam diferentes métodos de seleção do conjunto de consultas para a criação de visões materializadas. Entre elas, podemos citar: a) pesquisas que utilizam heurísticas gulosas como [Kumar and Kumar 2012], [Aouiche et al. 2006], [Chan et al. 2001]; b) métodos de seleção baseados em heurísticas randômicas como apresentados em [Kumar and Kumar 2013],[Yuhang et al. 2010], [Gao and Song 2010], [Sun and Wang 2009]; c) abordagens baseadas em heurísticas genéticas e evolucionárias como as apresentadas por [Kumar and Kumar 2012], [Talebian and Kareem 2010], [Lawrence 2006]; d) heurísticas híbridas como [Yuhang et al. 2010] que apresenta uma mistura de abordagens genética e randômica; e) heurísticas específicas como a de [Carvalho 2011].

Apesar de utilizarem a solução de visões materializadas para realizar sintonia fina, nenhuma das abordagens se propõe a fornecer ao DBA uma justificativa das decisões e tornar explícito o raciocínio utilizado na atividade de sintonia fina.

Outra diferença é que ao contrário das pesquisas citadas, o *framework outer-tuning* tem a capacidade de combinar e comparar heurísticas através de inferências. Com isso, pode-se avaliar não apenas as soluções específicas de um mesmo tipo de estrutura de acesso, mas comparar soluções de tipos diferentes de técnicas de sintonia fina como a criação de índices, visões, particionamento e qualquer outra técnica de que esteja prevista na ontologia.

5. Avaliação dos resultados

Para validar a proposta, planeja-se a execução do *framework Outer-Tuning* utilizando o benchmark TPC-H[4], Após obtidas as sugestões de visões materializadas do *framework*, elas deverão ser comparadas com soluções propostas por outras ferramentas, que sugerem ações de sintonia fina através de visões materializadas.

Espera-se que as soluções sugeridas pelo *framework Outer-Tuning* tragam um benefício para a execução da carga de trabalho, tão bom quanto às ferramentas disponíveis e que não utilizam semântica para a seleção de visões materializadas.

Os resultados obtidos até agora mostram a viabilidade no desenvolvimento do *framework outer-tuning*, visto que a máquina de inferência JESS já se é capaz de inferir todos os conceitos básicos da ontologia de domínio proposta, através das consultas da carga de trabalho capturada.

Como a implementação ainda não foi finalizada, não se pode realizar ainda experimentos que apoiem resultados práticos e concretos e executar o ciclo completo da heurística de criação de visões materializadas proposta. Porém a estrutura base já projetada e implementada.

Espera-se nos próximos passos então, a finalização do *framework outer-tuning*, a extensão da ontologia de domínio e a definição da heurística de seleção de visões materializadas e, por fim, realizar a avaliação experimental da qualidade dos resultados obtidos pelo *framework* e da ontologia proposta.

[4]TPC-H. http://www.tpc.org/

References

Almeida, A. C. B. d. (2013). *Framework para apoiar a sintonia fina de banco de dados*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro - PUC-RIO.

Aouiche, K., Jouve, P., and Darmont, J. (2006). Clustering-based materialized view selection in data warehouses. *10th East European conference on Advances in Databases and Information Systems*.

Bruno, N. (2012). *Automated Physical Database Design and Tuning*. CRC Press, New York, New York, USA.

Carvalho, A. W. (2011). Criação Automática de Visões Materializadas em SGBDs Relacionais. *MsC. thesis. Pontifícia Universidade Católica do Rio de Janeiro - PUC-RIO*.

Chan, G., Li, Q., and Feng, L. (2001). Optimized design of materialized views in a real-life data warehousing environment. *Intl. Journal of Information Technology*, 7(1):1–25.

Chirkova, R. and Yang, J. (2011). Materialized Views. *Foundations and Trends in Databases*, 4(4):296–405.

Gao, L. and Song, X. (2010). An Ant Colony based algorithm for optimal selection of Materialized view. *Conf. Intelligent Computing and Integrated Systems*.

Gruber, T. (2009). Ontology. *Encyclopedia of Database Systems*, pages 1963–1965.

Kumar, T. and Kumar, S. (2012). Materialized view selection using genetic algorithm. *Contemporary Computing*, pages 225–237.

Kumar, T. and Kumar, S. (2013). Materialized view selection using iterative improvement. *Advances in Computing and Information Technology*, pages 205–213.

Lawrence, M. (2006). Multiobjective genetic algorithms for materialized view selection in OLAP data warehouses. *GECCO '06*, page 699.

Monteiro, J. M., Brayner, A., and Lifschitz, S. (2008). Estado da Arte em Auto-Sintonia do Projeto Físico de BD. *Monografias em Ciência da Computação - PUC-RIO*.

Ramarkrishnan, R. and Gehrke, J. (2008). *Sistemas de gerênciamento de banco de dados*. McGraw Hill.

Salles, M. A. V. (2004). Criação autônoma de índices em bancos de dados.

Shasha, D. and Bonnet, P. (2002). *Database Tuning: Principles, Experiments, and Troubleshooting Techniques*. Elsevier Science.

Silberschatz, A., Korth, H. F., and Sudarshan, S. (2006). *Sistema de banco de dados*. CAMPUS - RJ.

Sun, X. and Wang, Z. (2009). An Efficient Materialized Views Selection Algorithm Based on PSO. *Intl. Workshop on Intelligent Systems and Applications*, pages 1–4.

Talebian, S. H. and Kareem, S. A. (2010). Lexicographic Ordering Genetic Algorithm for Solving Multi-objective View Selection Problem. *Intl. Conf. on Computer Research and Development*, pages 110–115.

Yuhang, Z., Qi, L., and Wei, Y. (2010). Materialized view selection algorithm—CSSA_VSP. *Intl. Conf. Comput. Intel. and Natural Computing (CINC)*, pages 68–71.

paper:37

Uma Abordagem de Classificação não Supervisionada de Carga de Trabalho MapReduce Utilizando Análise de Log

Ivan Luiz Picoli[1], Eduardo Cunha de Almeida[1]

Nível: Mestrado; Ano de Ingresso: 2013; Data da defesa de proposta: 19/05/2014; Época esperada de conclusão: Início de 2015

[1]Programa de Pós-Graduação em Informática – Departamento de Informática – Universidade Federal do Paraná (UFPR) – PR – Brasil

ilpicoli@inf.ufpr.br, eduardo@inf.ufpr.br

Etapas já concluídas: Estudo sobre o estado da arte, definição do tema, definição do problema, definição da solução, desenvolvimento da ferramenta, implementação do algoritmo de classificação K-Means, classificação de logs hadoop/hive, experimentos preliminares, defesa da proposta, implementação da ferramenta.

Etapas futuras: Término da implementação do mecanismo proposto; implementação de outros algoritmos de classificação; implementação de outras entradas de dados; classificação de logs de outros sistemas mapreduce; execução de experimentos em base de treinamento, teste e validação; defesa da dissertação.

Abstract. *Performance tuning in MapReduce query processing systems is a current hot topic in database research. Solutions such as Starfish and AutoConf provide mechanisms to tune MapReduce queries by changing the values of the configuration parameters for each job on the query plan. The goal of this paper is showing an approach for parameter tuning in MapReduce systems by using unsupervised machine learning. With this approach, we can trace the correct use of resources required by MapReduce jobs in order to improve performance tuning of similar upcoming jobs.*

Resumo. *Ajuste de desempenho no processamento de consultas em MapReduce é um tópico crescente na pesquisa em banco de dados. Soluções como Starfish e AutoConf possuem mecanismos que ajustam as consultas modificando os valores dos parâmetros de configuração das tarefas no plano de consulta, melhorando o desempenho. O objetivo do trabalho é apresentar uma abordagem de ajuste de parâmetros de configuração utilizando aprendizagem de máquina não supervisionado. A abordagem será capaz de encontrar o real uso de recursos exigidos pelas tarefas, possibilitando então, o ajuste de parâmetros em futuras execuções de tarefas semelhantes.*

Palavras-chave: MapReduce, query processing, parameter tuning, machine learning.

1. Introdução

O paradigma de programação MapReduce, desenvolvido em 2004 pela Google Inc., [Dean and Ghemawat 2004], foi projetado para simplificar as tarefas de processamento distribuído que necessitavam de escalabilidade linear. As principais características são relacionadas a tornar transparente a programação distribuída. Dentre

estas características, encontramos: o gerenciamento de falhas, redundância dos dados, e transferência de informações; balanceamento de carga; e a escalabilidade. O MapReduce é baseado na estrutura de dados em formato de chave/valor, estrutura que vem sendo cada vez mais utilizada entre os novos sistemas de banco de dados. Entre os sistemas baseados em MapReduce estão o Hadoop, Disco, Spark, Hadapt e Impala.

Processos de análise de dados e plataformas em nuvem podem ser reduzidas às aplicações MapReduce, juntamente com o seu ecossistema de aplicações. O desempenho das aplicações MapReduce está ligado diretamente aos parâmetros de configuração, e o correto ajuste desses parâmetros faz com que as aplicações aloquem recursos computacionais distribuídos de maneira mais eficiente.

Diversos fatores influenciam no desempenho de aplicações MapReduce, como por exemplo a leitura e escrita em disco que torna-se custoso devido ao sistema de arquivos estar fragmentado na rede. Tarefas com diferentes usos de recursos requerem configurações diferentes em seus parâmetros para que haja diminuição no tempo de resposta da aplicação. Um software autoajustável, o qual ajusta parâmetros de acordo com a carga de trabalho é uma solução para a diminuição do tempo de processamento.

2. Fundamentação Teórica

O MapReduce auxilia no processamento do tipo OLAP (On-line Analytical Processing), porém há um grande desafio em manter um bom desempenho neste tipo de sistema. Considere várias tarefas chegando a um cluster MapReduce onde cada uma delas possui características diferentes como o tamanho dos dados de entrada e o consumo de recursos de memória, disco, rede e processador. A mudança dos parâmetros ligados ao uso desses recursos afeta diretamente o desempenho da execução das tarefas, pois gerenciam o fluxo de informações entre os nodos do sistema distribuído. O ajuste de parâmetros pode ser aplicado para otimização, e pode ser feito de duas formas, pelo administrador do cluster MapReduce manualmente, ou automaticamente por uma ferramenta de autoajuste.

Dentre as abordagens de autoajuste de parâmetros de desempenho do Hadoop, se encontram o StarFish (otimização de parâmetros e geração de perfis de uso de recursos), [Herodotou 2011], Stubby (análise de fluxo de trabalho), [Lim, Herodotou e Babu 2012], e AutoConf (modificação de parâmetros em tempo de execução), [Lucas Filho 2013].

A geração de perfis consiste em coletar informações sobre a execução de uma tarefa e criar o que chamamos de perfil, contendo todas as informações coletadas. Essa abordagem é importante, pois através de buscas e análises nas informações coletadas é possível ajustar os parâmetros de sistemas MapReduce, [Herodotou 2011]. Starfish é um sistema que foi criado para determinar um perfil às tarefas do Hadoop e oferecer ao administrador do sistema melhores configurações de parâmetro, porém, monitora a tarefa durante sua execução, consumindo recursos que seriam utilizados pela tarefa. Sistemas como o Mochi e o Rumen analisam os logs do Hadoop com a finalidade de mostrar ao administrador informações relevantes sobre o ambiente, como: tempo total de execução, volume de dados processados e tarefas falhas. Porém, esses sistemas apenas mostram o uso de recursos, cabendo ao administrador tomar as decisões. Outra ferramenta abordada é o AutoConf, desenvolvido para otimização de consultas do

sistema de Data Warehouse Apache Hive, [Thusoo 2009]. O AutoConf realiza a análise dos operadores (TableScan, GroupBy, etc.) utilizados em cada estágio da consulta e extrai uma assinatura de código para esses estágios (i.e. ocorrência de operadores de consulta no código fonte). Cada assinatura possui uma configuração de parâmetros associada, a qual é aplicada em tempo de execução no Hadoop antes das tarefas serem executadas. Os grupos de tarefas classificados a partir das assinaturas de código recebem o nome de grupos de intenção.

Nossa abordagem amplia as funcionalidades dos sistemas citados, analisando a tarefa após a execução e possibilitando o ajuste de parâmetros em tarefas futuras, sem consumir recursos para análise durante a execução. Utilizando automaticamente as informações coletadas dos logs, a abordagem proposta visa encontrar o real uso de recursos computacionais de uma tarefa.

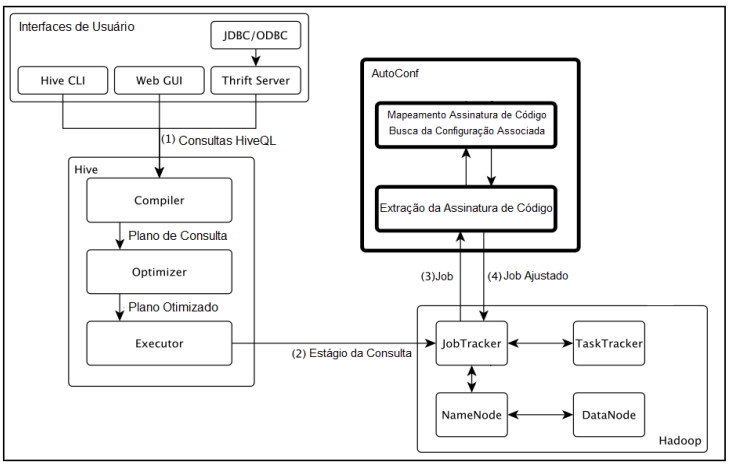


Figura 1. AutoConf em meio ao ecossistema do Hive e Hadoop [Autoria Própria]

A Figura 1 mostra a arquitetura do AutoConf. Em (1) as Interfaces de Usuário submetem uma consulta HiveQL; Após o Hive gerar e otimizar o plano de consulta, em (2) os estágios da consulta (tarefas) são enviados ao Hadoop; O JobTracker antes de ordenar a execução em (3) envia a consulta ao AutoConf; Após a extração da assinatura de código e a associação com a configuração correta, em (4) o AutoConf envia ao JobTracker os valores de parâmetros a serem ajustados; Por fim o JobTracker ajusta os parâmetros e ordena a execução da tarefa que agora possui as configurações ajustadas.

3. Caracterização da Contribuição

Classificar tarefas MapReduce antes da execução exige a extração de informações através de regras (abordagem AutoConf), resultando em dados menos precisos do que se a análise fosse realizada após a execução. Ao trabalhar com dados incertos, as tarefas MapReduce podem ser classificadas erroneamente, e os parâmetros de configuração receberão valores inapropriados.

Para que as tarefas MapReduce sejam ajustadas de forma mais precisa, propomos encontrar o uso real de recursos computacionais que as tarefas exigem do sistema, auxiliando em futuras execuções. Para tal, propomos uma abordagem de classificação de tarefas MapReduce que possuam informações de uso de recursos persistidos em arquivos de log de execução. Realizaremos a análise dos logs através do uso de algoritmos de classificação não supervisionada. Os algoritmos irão gerar como

resultado classes que determinam o uso de recursos que foram utilizados pelas tarefas. Nossa hipótese é que grupos de tarefas que possuam o mesmo uso de recursos podem receber a mesma configuração de parâmetros. Então, através de nossa abordagem qualquer tarefa MapReduce submetida para execução e que se assemelhe às tarefas já classificadas, poderão receber o mesmo ajuste de parâmetros.

4. Uma Abordagem de Classificação Não Supervisionada de Análise de Log

Propomos as seguintes contribuições nesta dissertação de mestrado:

- Objetivo: Aumentar a representatividade da classificação das tarefas utilizando algoritmos de classificação não supervisionados;

  Desafio: Validar o uso do algoritmo K-Means para classificação através da comparação dos valores de uso de recursos encontrados com o real uso encontrado por monitoramento;

- Objetivo: Viabilizar um mecanismo de troca de algoritmo de classificação e de entradas de log para comparações futuras e implementação em mais de um framework MapReduce;

  Desafio: Criar uma arquitetura que possibilite a troca do algoritmo e das entradas de log;

- Objetivo: Auxiliar com mais precisão no ajuste de parâmetros de sistemas MapReduce, como: Hive e PIG, [Olston 2008];

  Desafio: Avaliar a qualidade dos parâmetros.

4.1. Algoritmos de Classificação

A classificação é uma das técnicas de mineração de dados e sua função é agrupar os dados com características comuns. A classificação supervisionada pode ser utilizada quando se sabe a natureza dos dados a serem analisados e os mesmos encontram-se pré-classificados. Quando não se conhece a natureza dos dados utiliza-se classificação não supervisionada, onde não é preciso informar nenhuma informação prévia dos registros (no nosso caso, os logs).

4.2. Abordagem de Classificação de Logs MapReduce

Nosso objetivo é apresentar uma abordagem de classificação não supervisionada de uso de recursos de tarefas MapReduce através da análise de logs. A vantagem que propomos nesse trabalho é o possível uso de diversos algoritmos de classificação na análise dos logs, ou seja, uma abordagem extensível de classificação. Também propomos a possível implementação da abordagem em qualquer framework MapReduce que gere arquivos de log. A Figura 2 mostra os componentes de nossa abordagem, destacados pela linha pontilhada.

Na Figura 2, podemos visualizar a geração dos chamados grupos de comportamento, esses grupos são encontrados pelo algoritmo de classificação e os valores de seus centróides equivalem ao uso médio de recursos das tarefas pertencentes ao mesmo grupo. Cada grupo de comportamento recebe uma configuração de parâmetros que poderá ser aplicada em tarefas classificadas.

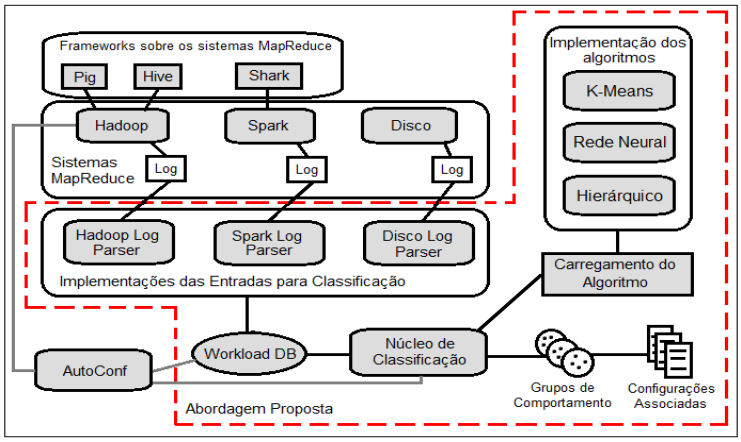


Figura 2. Nossa abordagem em meio ao ecossistema MapReduce [Autoria Própria]

Neste trabalho nos baseamos no algoritmo de classificação K-means [Aggarwal 2010], e pode ser descrito em cinco passos, descritos pelo Algoritmo 1 no estudo de caso ao analisar logs.

1: k = número de classes desejadas
2: $L = \{l_0, ..., l_n\} \rightarrow$ Conjunto de logs a serem classificados onde $l = \{dim_0, ..., dim_n\} \rightarrow$ Log contendo suas dimensões (e.g., atributos escolhidos para análise)
3: $C = \{c_1, ..., c_k\} \rightarrow$ Centróides (e.g., classes) onde $c = \{dim_0, ..., dim_n\} \rightarrow$ Centróide contendo suas dimensões (e.g, atributos escolhidos para análise)
4: $LC = [\{l_0, c\}, ..., \{l_n, c\}] \rightarrow$ Conjunto relacionando os logs a um centróide
5: $C \leftarrow$ todos os centróides iniciam suas dimensões com valores aleatórios entre o valor mínimo e máximo encontrados entre os logs;
6: **while** Houver alterações de classe entre os logs na última iteração **do**
7: **for** $l_i \leftarrow L$ **do** // Itera os logs
8: $D = \{d_0, ..., d_n\} \rightarrow$ Distâncias entre os centróides e o log
9: **for** $c_x \leftarrow C$ **do** // Itera os centróides
10: $D_x \leftarrow$ cálculo da distância euclidiana entre o log l_i e o centróide c_x;
11: **end for**
12: $LC_i \leftarrow \{l_i, D$ com menor valor (e.g., centróide mais próximo do log)$\}$
13: **end for**
14: **for** $c_i \leftarrow C$ **do** // Recalcula os valores dos centróides
15: **for** $dim_x \leftarrow c_i$ **do** // Itera as dimensões do centróide
16: $dim_x \leftarrow$ Dimensão dim_x do centróide recebe a média entre os valores da dimensão dim_x dos logs que foram classificados para a classe c_i
17: **end for**
18: **end for**
19: **end while**

Algoritmo 1. Algoritmo K-Means em um estudo ao analisar logs [Autoria Própria]

No Algoritmo 1, os vetores *C* e *LC* armazenam o resultado final da classificação. *C* contém as classes que representam o uso dos recursos computacionais, onde os recursos são representados pelas dimensões *dim*_x. *LC* contém a referência para os logs e a qual classe eles pertencem após a classificação.

4.3. Protótipo

Na implementação do protótipo, integramos nossa abordagem com a arquitetura do AutoConf e nomeamos o novo software de Chameleon. Essa proposta possui apenas a implementação do algoritmo K-Means. Essa estratégia irá complementar o AutoConf na otimização de consultas do Hive não somente pela assinatura de código, mas também através da comparação da tarefa com os grupos de comportamento gerados pela nossa abordagem. A Figura 3 mostra a arquitetura do Chameleon, e podemos visualizar os módulos de análise de log e um banco de dados (Workload Database) que armazena os resultados da classificação inseridos na arquitetura da Figura 1.

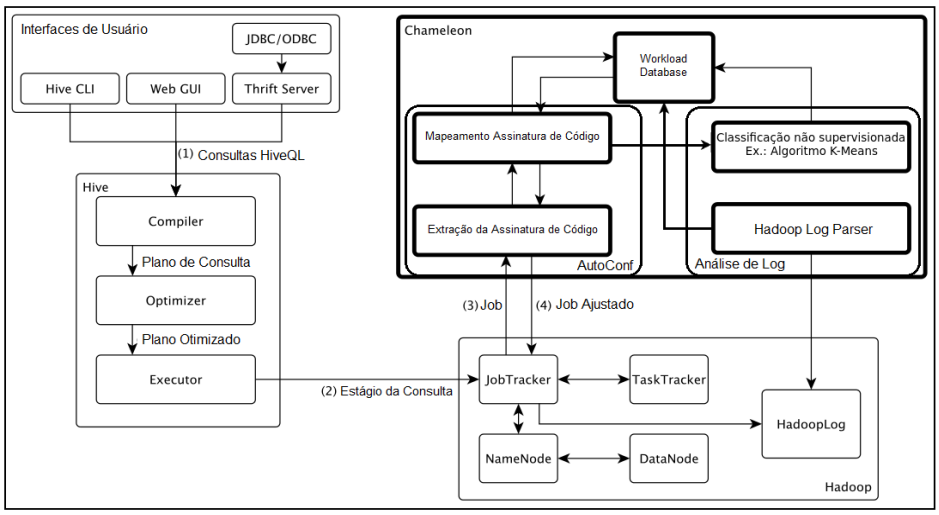


Figura 3. Chameleon e o ecossistema do Hadoop [Autoria Própria]

Tendo como base o funcionamento dos módulos e o objetivo do Chameleon na otimização das consultas do Hive, vê-se necessário uma forma de integração com o AutoConf. A conexão será feita após a identificação da assinatura de código e antes da aplicação da nova configuração. É realizada uma chamada remota ao módulo responsável por determinar a configuração, que será aplicada a partir dos grupos de comportamento e não mais a partir dos grupos de intenção do AutoConf. O método pode ser visualizado no Algoritmo 2, que mostra o processo de ajuste utilizando os grupos de comportamento.

```
   I = {j_0, ..., j_n} → Jobs com assinaturas de códigos idênticas
2: C = {c_0, ..., c_n} → Contém a quantidade de Jobs em cada cluster de comportamento,
   todos os valores se iniciam em 0
   procedure GETTUNING(I)
4:     for j_i ← I do
           x ← extrai o índice do cluster de comportamento de j_i
6:         c_x ← c_x + 1
       end for
8:     m ←busca o índice do cluster de comportamento com maior valor em C
       return m
10: end procedure
```

Algoritmo 2. Aplicação da configuração pelos grupos de comportamento [Autoria Própria]

No Algoritmo 2 ao chamar o método remoto, o AutoConf envia ao método um histórico de tarefas ($I=\{j_0,...,j_n\}$) que utilizaram a mesma configuração da tarefa a ser ajustada (j_n+1), ou seja, que possuam a mesma assinatura de código (grupo de intenção). Como essas tarefas de histórico já foram completadas, elas possuem logs que estarão armazenados no Workload Database, e então o Clustering (módulo responsável pela classificação) gera uma lista chave/valor (C) onde a chave é o identificador do grupo de comportamento e o valor é a quantidade de logs que foram enviados pelo AutoConf e que estão relacionados ao grupo. Esses dados são adquiridos ao analisar as duas tabelas citadas (intenção e comportamento) e podemos visualizar a geração da lista no laço entre as linhas 4 e 7 do Algoritmo 2.

O retorno do método remoto é o identificador do grupo de comportamento o qual a tarefa a ser ajustada se enquadra, ou seja, o grupo que mais obteve ocorrências de tarefas vindas do grupo de intenção (linha 8 e 9 do Algoritmo 2). Em seguida, o AutoConf aplica a configuração relacionada ao grupo de comportamento encontrado.

5. Resultados Preliminares e estado atual do Trabalho

O trabalho está na fase inicial dos experimentos. Neste seção apresentamos resultados preliminares da nossa abordagem de classificação não supervisionada. Em nossos experimentos foram extraídos 1505 logs de tarefas executadas em um cluster MapReduce com 10 máquinas idênticas. Os logs foram gerados através da execução do benchmark TPC-H adaptado para HiveQL. O TPC-H foi escolhido por possuir um grupo de consultas capaz de exercitar um data warehouse de formas distintas. Cada tarefa executada possui um arquivo de log contendo mais de 60 tipos de informações, e em nossos testes utilizamos cinco informações que estão descritas na Tabela 1, resultando em cinco entradas para cada um dos 1505 arquivos de log na execução do algoritmo. Foram realizados experimentos variando o número de classes do algoritmo e também as informações utilizadas. A Tabela 1 mostra os resultados de um dos testes.

Classe	Número de Maps	Número de Reduces	Tempo de Execução (seg)	Dados de Entrada nos Maps (MB)	Tamanho do Cluster	Porcentagem do Cluster
1	170	45	564,7	2047,0	10	0,66 %
2	156	37	398,8	2026,8	18	1,20 %
3	152	36	324,2	2036,0	33	2,20 %
4	139	36	316,3	2006,5	47	3,13 %
5	125	33	335,7	1971,7	60	4,00 %
6	45	11	160,9	1777,9	99	6,59 %
7	22	5	65,2	1431,0	226	15,04 %
8	3	1	16,7	90,5	1009	67,18 %

Tabela 1. Algoritmo K-Means agrupando logs Hadoop com 8 grupos [Autoria Própria]

Podemos descrever as próximas tarefas de testes e experimentos em: execução de testes com o Hadoop+AutoConf e comparação com Hadoop+Abordagem utilizando K-Means; validação dos resultados comparando o uso de recursos encontrados com dados de monitoramento de uso de hardware no momento da execução das tarefas; repetir o experimento utilizando outra carga de dados além do TPC-H para validação.

6. Referências

Aggarwal, Sonali e Phadke, Shashank e Bhandarkar, Milind A. Characterization of hadoop jobs using unsupervised learning. CloudCom, 2010, pages 748–753.

Dean, J e Ghemawat, S. MapReduce: Simplified data processing on large clusters. 6^{Th} OSDI, 2004, pages 137–150.

Herodotou, H. e Lim, H. e Luo, Gang e Borisov, N. e Dong, Liang e Cetin, F. B. e Babu, S. Starfish: A self-tuning system for big data analytics. CIDR, 2011.

Lim, H e Herodotou, H e Babu, S. A transformation-based optimizer for mapreduce workflows. VLDB, 2012.

Lucas Filho, Edson Ramiro. HiveQL self-tuning – Curitiba, 2013. 44f. : il. color. ; 30 cm. Dissertation (master) – UFPR, Pos-graduate Program in Informatics, 2013.

Olston, C. e Reed, B. e Srivastava, U. e Kumar, R. e Tomkins, A. Pig latin: A not-so-foreign language for data processing. SIGMOD, 2008.

Thusoo, A. e Sarma, J. S. e Jain, N. e Shao, Z. e Chakka, P. e Anthony, S. e Liu, Hao e Wycko, P. e Murthy, R. Hive - a warehousing solution over a mapreduce framework. VLDB, 2009.

paper:17

WTMatcher: Uma Abordagem para Determinação de Similaridade entre Tabelas na Web

Filipe Roberto Silva[1]
Orientador: Ronaldo dos Santos Mello[1]

[1]Programa de Pós-Graduação em Ciência da Computação
Departamento de Informática e Estatística– Universidade Federal de Santa Catarina
Florianópolis – SC – Brasil

{filipesilva.sc,ronaldo}@inf.ufsc.br

Nível : Mestrado
Admissão : Março de 2013
Exame de Qualificação : Junho de 2014
Conclusão : Março de 2015
Etapas Concluídas : Revisão bibliográfica, solução preliminar e avaliação
Etapas Futuras : solução completa e avaliação

Resumo. *A Web é uma vasta fonte de informação. Muitos dados são publicados diariamente e boa parte deles está disponível na forma de tabelas HTML. Alguns trabalhos têm estudado formas de extrair e integrar informações presentes em tabelas Web, visando torná-las mais acessíveis para consumo humano. Entretanto, esta é uma tarefa complexa e um tema ainda em aberto na literatura, uma vez que tabelas na Web não possuem um padrão único de representação. Aliado a isso, o uso de sinônimos e abreviações torna difícil a comparação entre os conteúdos das tabelas. Assim sendo, este trabalho se propõe a desenvolver um método para determinação de similaridade entre tabelas na Web capaz de lidar com suas estruturas heterogêneas e termos sinônimos. Trabalhos relacionados não lidam, ao mesmo tempo, com o tratamento dessas duas problemáticas. A intenção é definir uma estratégia simples e efetiva para comparar tabelas Web, tomando como base métricas de similaridade para dados textuais conhecidas na literatura e sua adaptação e/ou combinação para este propósito.*

Keywords. Tabelas Web, Comparação, Sinônimos, Similaridade, Semântica

1. Introdução

A Web é uma vasta fonte de informação. Muitos dados são publicados diariamente e boa parte deles está disponível na forma de tabelas HTML. Cafarella [Cafarella et al. 2008] extraiu 14.1 bilhões dessas tabelas, caracterizadas pela *tag* `<table>` e mostrou que 154 milhões delas continham dados relacionais de alta qualidade, ou seja, tabelas com dados relevantes em um domínio do conhecimento e que estão no formato relacional. Alguns trabalhos têm estudado formas de extrair e integrar informações presentes nessas tabelas, visando torná-las mais acessíveis para consumo humano [Sardi Mergen et al. 2010, Venetis et al. 2011]. Porém, a falta de esquema, padrões e metadados na Web torna difícil identificar quais tabelas podem ser integradas, ou obter outras fontes com dados semelhantes.

A identificação de tabelas similares pode ser de grande ajuda nesse processo. Esta identificação pode ser útil, por exemplo, para atividades de integração de dados, preenchimento de tabelas incompletas utilizando dados de tabelas similares, além de permitir a busca por tabelas Web. Entretanto, esta é uma tarefa complexa e um tema ainda em aberto na literatura.

Crestan e Pantel [Crestan and Pantel 2011] estimam que 88% das tabelas HTML são utilizadas para definir a estrutura das páginas Web e não possuem dados relacionais. Em um processo de recuperação de informação, tabelas desse tipo teriam que ser filtradas, visto que não possuem dados relevantes. Além dessas tabelas ditas de layout, tabelas de dados não possuem um padrão único de construção. Este é um dos principais problemas para a recuperação correta das suas informações, visto que é necessário detectar onde e como os dados são apresentados. Aliado a isso, o uso de sinônimos e abreviações torna difícil a comparação por similaridade entre os conteúdos das tabelas, visto que é preciso informar, de alguma maneira, ao software que aquelas palavras significam a mesma coisa.

Assim sendo, este trabalho se propõe a desenvolver um método para determinação de similaridade entre tabelas na Web capaz de lidar com suas estruturas heterogêneas e termos sinônimos. Trabalhos relacionados não lidam, ao mesmo tempo, com todas as representações existentes para tabelas na Web e com o tratamento de sinônimos. Essa é a principal contribuição deste trabalho. A intenção é definir uma estratégia simples e efetiva para comparar tabelas Web, tomando como base métricas de similaridade para dados textuais conhecidas na literatura e sua adaptação e/ou combinação para este propósito. O trabalho utiliza como definição de similaridade entre tabelas Web, duas tabelas que possuam dados pertencentes a um mesmo domínio, como por exemplo, duas tabelas com informações sobre filmes. Até o presente momento os experimentos preliminares consideram apenas aspectos sintáticos na comparação (strings), mas pretende-se, futuramente, considerar aspectos semânticos com o suporte de bases de conhecimento.

A Seção 2 deste trabalho, apresenta os trabalhos relacionados. As Seções 3 e 4 mostram respectivamente a descrição do método proposto e experimentos preliminares realizados. Por fim, a Seção 5 apresenta a conclusão e atividades futuras.

2. Trabalhos Relacionados

Esta seção apresenta uma breve descrição dos trabalhos mais recentes relacionados à problemática de comparação de tabelas na Web. Este levantamento do estado da arte foi alvo de um trabalho anterior [Silva and Mello 2014].

Lautert et al. [Lautert et al. 2013] sugerem uma taxonomia para tabelas na Web que divide as tabelas relacionais em:

- **Vertical**: tabelas cujas tuplas estão dispostas na direção vertical;
- **Horizontal**: tabelas cujas tuplas estão dispostas na direção horizontal;
- **Matriz**: tabelas que possuem cabeçalhos na vertical e horizontal e no cruzamento entre os dois encontra-se o valor;
- **Concisa**: tabelas que possuem células mescladas;
- **Aninhada**: tabelas que possuem tabelas internas;
- **Dividida**: tabelas que, por questão de espaço, são divididas horizontal ou verticalmente e suas partes são posicionados lado a lado ou uma sobre a outra;
- **Multivalorada Simples**: tabelas com múltiplos valores de um mesmo domínio em uma célula;
- **Multivalorada Composta**: tabelas com múltiplos valores de diversos domínios em uma célula.

Fan et al. [Fan et al. 2013] apresentam um sistema semi-supervisionado para recuperar o esquema das tabelas na Web e realizar o *schema matching*. O esquema das tabelas é obtido relacionando conceitos da base de conhecimento *Freebase* [Bollacker et al. 2008] com as colunas das tabelas. Descobertos os conceitos de cada tabela, o sistema determina quais tabelas são similares através de comparações entre seus conceitos.

Pawlik e Augsten [Pawlik and Augsten 2011] propõem o RTED, um algoritmo que calcula a distância de edição entre árvores, considerando as operações de edição, com menor custo, que transformam uma árvore em outra. Cada operação recebe um custo e a distância de edição é a soma desses custos para transformar uma árvore em outra. Esse algoritmo pode ser adaptado para comparar árvores HTML.

Sarma et al. [Das Sarma et al. 2012] propõem um *framework* de comparação de tabelas cujo principal objetivo é encontrar tabelas candidatas para *joins* e *unions*. O trabalho utiliza três bases de conhecimento para rotular as tabelas: *Freebase*, *WebIsA* [Venetis et al. 2011] e uma base própria. Esses rótulos, juntamente com os dados das tabelas são comparados para identificar quais tabelas são similares.

A Tabela 1 mostra um comparativo entre as abordagens, incluindo a abordagem proposta proposta neste trabalho (WTMatcher). Como é possível observar, grande parte das abordagens processa somente tabelas Horizontais. Também verifica-se que os trabalhos que tratam qualquer tipo de tabela não possuem tratamento de sinônimos. As limitações desses trabalhos motivaram o desenvolvimento desta dissertação. A intenção é propor uma abordagem para comparar tabelas Web que considere o tratamento de sinônimos e seja capaz de lidar com as estruturas heterogêneas das tabelas Web, conforme a classificação de Lautert et al. (2013), durante o processo de comparação.

3. *WTMatcher*

A solução proposta intitula-se *WTMatcher* (*Web Table Matcher*). *WTMatcher* é um processo de comparação de tabelas Web para determinação de similaridade entre elas. Uma visão geral deste processo é mostrada na Figura 1. O sistema possui como entrada duas tabelas HTML. Elas são processadas de forma a identificar suas principais estruturas,

Tabela 1. Comparativo entre as abordagens

Trabalho	Tipo de Tabelas Processadas	Processamento	Tratamento de Sinônimos	Estratégia
[Fan et al. 2013]	Horizontais	Semi-supervisionado	Sim	Base de Conhecimento e funções de similaridade
[Das Sarma et al. 2012]	Horizontais	Automático	Sim	Base de Conhecimento e funções de similaridade
[Pawlik and Augsten 2011]	Qualquer tipo	Automático	Não	TED
WTMatcher	Qualquer tipo	Automático	Sim	Base de Conhecimento e funções de similaridade

como cabeçalhos e dados. A seguir, as tabelas são comparadas utilizando um algoritmo de comparação de tabelas Web, que é o foco deste trabalho. Esse algoritmo de comparação atuará juntamente com uma base de conhecimento (BC). Pretende-se utilizar bases disponíveis e organizadas em domínios, como o Freebase [Bollacker et al. 2008]. Com isso é gerado um escore de similaridade entre elas. Esse escore é um valor entre 0 e 1, quanto mais próximo de 1 mais similares são as tabelas. O processo considera tabelas Web já identificadas, extraídas da Web e classificadas segundo a taxonomia proposta por Lautert et al. (2013). Para tanto, foi utilizado um extrator de tabelas Web [Scheidt and Dorneles 2013].

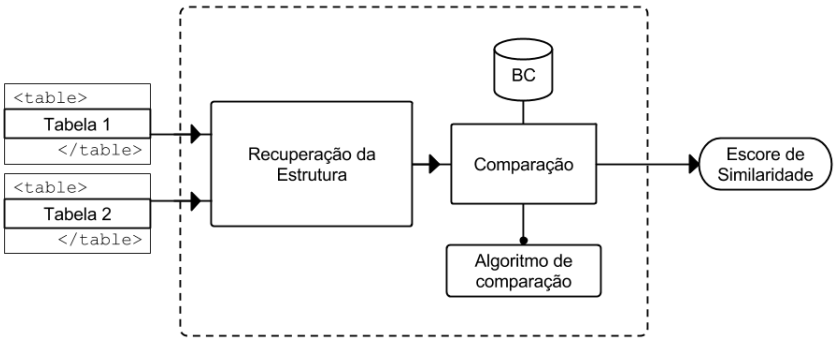


Figura 1. Arquitetura do *WTMatcher*

Com os dados já extraídos e classificados, este trabalho propõe, até o presente momento, dois métodos simples de comparação de tabelas: o *SimpleMatcher* e o *SimpleMatcherPlus*. A estratégia de ambos é comparar estruturas HTML puras de tabelas. Eles partem basicamente do mesmo princípio: selecionar dados e atributos a partir de heurísticas simples e realizar comparação de strings. Nesses métodos, ainda não é levada em conta a comparação de sinônimos nem a classificação (estrutura lógica) de cada uma das 2 tabelas. Ambos métodos são detalhados nas subseções 3.1 e 3.2.

3.1. *SimpleMatcher*

O método *SimpleMatcher* basicamente compara cabeçalhos e dados, utilizando as tags `<th>`, `<td>` e `<caption>` para detectar cabeçalhos, dados e títulos das tabelas respectivamente. O intuito dessa detecção a partir de tags é comparar tabelas de forma independente de suas estruturas lógicas (vertical, matricial e etc), além de verificar a viabilidade de comparação de cabeçalhos e dados dessa forma simples.

A implementação do método utiliza a biblioteca em Java JSoup [Hedley 2009], o que permite a seleção precisa das tags a partir de seletores CSS, além da manipulação do

HTML. Com isso, são extraídos conjuntos de dados a partir de tabelas HTML. Para fins de comparação desses conjuntos, foi utilizada a métrica *SubSetSim* [Dorneles et al. 2004] que visa comparar conjuntos de valores. O *SubSetSim* utiliza métricas de comparação de strings. Por isso foram estudadas e analisadas algumas métricas já conhecidas na literatura para essa finalidade, conforme descrito na Seção 3.3. A Equação 1 expressa o cálculo de similaridade do *SimpleMatcher*, onde CS, HS e VS denotam as similaridades entre os *Captions*, *Headers* e *Values* respectivamente.

$$\textit{SimpleMatcher} = (CS * 0.33) + (HS * 0.33) + (VS * 0.33) \quad (1)$$

3.2. *SimpleMatcherPlus*

Este método é uma extensão do *SimpleMatcher*, ele acrescenta as tags `<b>`, `<h1>` e `<h2>` à detecção de cabeçalhos. O *SimpleMatcherPlus* também acrescenta a comparação de atributos contidos nas tags HTML, por exemplo os atributos `class`, `id` e qualquer outro atributo presente. A Equação 2 expressa este método, onde AS é a similaridade entre os atributos. Nessa primeira versão ele compara somente os atributos da tag `<table>`.

$$\textit{SimpleMatcherPlus} = (CS * 0.25) + (HS * 0.25) + (VS * 0.25) + (AS * 0.25) \quad (2)$$

3.3. Métricas de Similaridade

SubSetSim é a função base de comparação utilizada pelo *WTMatcher*. Ela é representada pela Equação 3, sendo ϵ_p e ϵ_d conjuntos de valores, n e m tamanhos dos conjuntos ϵ_p e ϵ_d respectivamente. A função calcula as similaridades dos valores todos com todos e soma as máximas similaridades. A função sim pode ser alterada de acordo com a necessidade. Assim sendo, várias métricas de similaridade de string, contidas na biblioteca *SimMetrics* [Chapman 2006], foram avaliadas para determinar as mais eficazes para a comparação de tabelas Web. Observando a forma de comparação de cada uma, bem como seus pontos fortes e fracos, foram selecionadas 7 métricas: Cosine, Monge Elkan, Chapman Length, Euclidean Distance, Jaro Winkler, Jaccard e Soundex. Para essa avaliação, foram realizados experimentos de comparação de tabelas Web utilizando cada uma dessas métricas. Esses experimentos são apresentados na Seção 4.

$$SubSetSim(\epsilon_p, \epsilon_d) = \frac{\sum \epsilon_p^i.\eta = \epsilon_d^j.\eta^{(max(sim(\epsilon_p^i,[\epsilon_d^1,\ldots,\epsilon_d^m])))}}{max(m,n)} \quad (3)$$

4. Experimentos Preliminares

Esta seção apresenta alguns experimentos preliminares utilizando os métodos propostos, além de uma comparação das métricas de similaridade de strings para o propósito deste trabalho. Os experimentos foram executados sobre uma massa de 450 tabelas com informações do laboratório LISA (UFSC) e de projetos *Wikimedia* extraídas, na sua maioria, dos sites: *http://lisa.inf.ufsc.br/wiki/* e *http://meta.wikimedia.org/wiki/*. A massa de dados foi analisada manualmente a priori e foram anotadas quais tabelas eram similares

entre si. Os testes automáticos compararam as tabelas entre si de forma a obter os escores de similaridade. Um valor de corte de 0.7 foi empregado.

Primeiramente foram realizados testes utilizando o método *SimpleMatcherPlus* e variando a métrica de similaridade de string empregada. A Tabela 2 apresenta os resultados dessa comparação em termos de *Precision*, *Recall* e *F-Measure*.

Tabela 2. Comparativo das métricas de similaridade de strings

Métrica	Precision	Recall	F-Measure
MongeElkan	0.58	0.82	0.72
Cosine	0.95	0,76	0,91
ChapmanLength	0,91	0,68	0,84
EuclideanDistance	0,97	0,69	0,87
JaroWinkler	0,67	0,83	0,81
Jaccard	0,95	0,71	0,88
Soundex	0,58	0,81	0,75

Como é possível observar na Tabela 2, a métrica *Cosine* foi a que obteve melhores resultados em termos de *F-Measure* para essa massa de dados. Por isso, foram realizados experimentos comparando os 2 métodos propostos utilizando essa métrica. Também foram feitos experimentos utilizando a métrica *MongeElkan*, a qual obteve piores resultados na comparação entre as métricas. Isso foi feito para verificar se o resultado sofrerá muita alteração variando da melhor para a pior métrica. Devido ao acesso ao código fonte, o RTED [Pawlik and Augsten 2011] foi utilizado nos experimentos como *baseline*. A Tabela 3 mostra os resultados dos experimentos.

Tabela 3. Resultados da aplicação dos métodos propostos e um *baseline*

Método	Precision	Recall	F-Measure
SimpleMatcher (MongeElkan)	0.58	0.82	0.72
SimpleMatcher (Cosine)	0.95	0,76	0,91
RTED Matcher	0,85	0,61	0,74
SimpleMatcherPlus (MongeElkan)	0,56	0,86	0,71
SimpleMatcherPlus (Cosine)	0,85	0,78	0,85

Os métodos obtiveram melhores resultados com a métrica *Cosine* do que com a métrica *MongeElkan*, o que já era esperado. Dentre esses métodos, o *SimpleMatcher* com a métrica *Cosine*, foi o que obteve melhores resultados dentre todos. Pretende-se, na sequência, realizar esses experimentos comparando a métrica *Cosine* com *Jaccard* e *EuclideanDistance*, que também obtiveram bom desempenho.

5. Conclusão e Atividades Futuras

Os resultados dos experimentos foram promissores, porém vale lembrar que a massa de dados utilizada foi obtida de poucas fontes, muito semelhantes entre si e em pequena quantidade. Isso pode ter cooperado para os bons resultados. Futuramente, pretende-se utilizar uma massa de dados maior e mais diversificada para verificar se os bons resultados persistem. Também pretende-se criar melhores heurísticas para detecção de esquema das tabelas, possivelmente baseadas no trabalho de Sarma [Das Sarma et al. 2012] que utiliza uma BC para esse propósito. A intenção dessa extensão é tornar o método sensível a sinônimos e abreviações.

Quanto às estruturas heterogêneas de tabelas, num primeiro momento pretende-se observar o desempenho desses métodos mais genéricos propostos. Caso esse desempenho

não seja avaliado como bom, poderá ser considerado um tratamento mais específico levando em conta a comparação entre os diferentes tipos de tabelas classificados por Lautert [Lautert et al. 2013].

Referências

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Cafarella, M. J., Halevy, A., Wang, D. Z., Wu, E., and Zhang, Y. (2008). Webtables: Exploring the power of tables on the web. *Proc. VLDB Endow.*, 1(1):538–549.

Chapman, S. (2006). Simmetrics. *URL http://sourceforge. net/projects/simmetrics/. Work provided by UK Sheffield University funded by (AKT) an IRC sponsored by EPSRC, grant number GR N*, 15764.

Crestan, E. and Pantel, P. (2011). Web-scale table census and classification. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 545–554, New York, NY, USA. ACM.

Das Sarma, A., Fang, L., Gupta, N., Halevy, A., Lee, H., Wu, F., Xin, R., and Yu, C. (2012). Finding related tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 817–828, New York, NY, USA. ACM.

Dorneles, C. F., Heuser, C. A., Lima, A. E. N., da Silva, A. S., and de Moura, E. S. (2004). Measuring similarity between collection of values. WIDM '04, pages 56–63, New York, NY, USA. ACM.

Fan, J., Lu, M., Ooi, B. C., Tan, W.-C., and Zhang, M. (2013). A hybrid machine-crowdsourcing system for matching web tables. Technical report, Technical Report.

Hedley, J. (2009). jsoup: Java html parser.

Lautert, L. R., Scheidt, M. M., and Dorneles, C. F. (2013). Web table taxonomy and formalization. *SIGMOD Rec.*, 42(3):28–33.

Pawlik, M. and Augsten, N. (2011). Rted: A robust algorithm for the tree edit distance. *Proc. VLDB Endow.*, 5(4):334–345.

Sardi Mergen, S. L., Freire, J., and Heuser, C. A. (2010). Indexing relations on the web. In *Proceedings of the 13th International Conference on Extending Database Technology*, EDBT '10, pages 430–440, New York, NY, USA. ACM.

Scheidt, M. M. and Dorneles, C. F. (2013). Ferramenta para extração de webtables e criação de scripts sql. Technical report.

Silva, F. and Mello, R. (2014). Análise de abordagens para recuperação de informação em tabelas na web. In *X Escola Regional de Banco de Dados (ERBD 2014)*.

Venetis, P., Halevy, A., Madhavan, J., Paşca, M., Shen, W., Wu, F., Miao, G., and Wu, C. (2011). Recovering semantics of tables on the web. *Proc. VLDB Endow.*, 4(9):528–538.

tutorials

29th Brazilian Symposium on Databases

October 6th to 9th, 2014
Curitiba – PR – Brazil

TUTORIALS

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

Tutorials Chair

Vanessa Braganholo, UFF

Editorial

The tutorials at the Brazilian Symposium on Databases (SBBD) have been a large success over the years. SBBD Tutorials present introductory and advanced discussions on topics within the area of databases. Introductory tutorials target an audience consisting of advanced undergraduate and graduate students, as well as attendees from industry. Hence, they can be thought of as "recycling" courses, though they are not about traditional "textbook" topics. Advanced tutorials, on the other hand, cover a topic's state-of-the-art, motivating and exposing potential research issues.

Tutorial proposals are submitted by expert authors in their subject, and selected by the SBBD steering committee. This year, we had six high quality submissions: three from Brazil, one from Germany, another on from Egypt, and one from the USA. The steering committee then selected two of these submissions to be presented at SBBD in Curitiba.

The first tutorial is entitled "Automatic methods for Author Name Disambiguation in Bibliographic Data Repositories" and will be presented in Portuguese by Anderson Ferreira, Marcos André Gonçalves and Alberto Laender. It covers the problem of disambiguating author names in two situations: (i) when the same author appears with different names in his/her publications; (ii) when different authors have the same name in their publications. The main goal of the tutorial is to provide an overview of the main disambiguation methods in the literature, as well as to discuss the main research challenges of the area. Anderson Ferreira is a professor at Universidade Federal de Ouro Preto, Brazil. Marcos André Gonçalves and Alberto Laender are both professors at Universidade Federal de Minas Gerais, Brazil.

The second tutorial, "Human-Powered Database Operations", will be presented by Dongwon Lee. It explores the use of crowdsourcing in the Database area. The goal is to cover state-of-the-art human-powered algorithms in data management, such as crowdsourced counting, aggregation, sorting, join, top-k, skyline, search, and entity resolution. Dongwon Lee is an associate Professor at the Pennsylvania State University, USA.

I would like to invite you all to attend the tutorials this year in Curitiba. Both tutorials cover very hot topics, and we hope you will take advantage of the expertise of their presenters. I'd also like to take this opportunity to thank all the authors for contributing to SBBD 2014.

Vanessa Braganholo, UFF
SBBD 2014 Tutorials Chair

Table of Contents (Tutorials)

Human-Powered Database Operations 370
Dongwon Lee

Métodos Automáticos para Desambiguação de Nomes de Autores em Repositórios de Dados Bibliográficos 373
Anderson A. Ferreira, Marcos André Gonçalves, Alberto H. F. Laender

paper:95

Human-Powered Database Operations

Dongwon Lee

The Pennsylvania State University
University Park, 16802-6823 – PA, USA

dongwon@psu.edu

Abstract. *As a novel computation paradigm, crowdsourcing is being actively pursued in diverse academic disciplines. Within computer science, many sub-fields have embraced the concept of crowdsourcing with open arms and applied the concept to solve diverse challenging problems. Database community is no exception to this phenomenon and there have been many exciting new results using crowdsourcing appearing in recent database literature. In addition to recent tutorials by other scholars that covered broad issues of crowdsourcing in general and contributed to the increased interests in the community, this tutorial seeks to cover more narrow topics with the focus of database, covering state-of-the-art human-powered algorithms in data management. After gentle introduction on the concept of crowdsourcing, this tutorial will provide the overall landscape of crowdsourced database research, with the focus on some of latest human-powered database operations that extend conventional ones (e.g., crowdsourced counting, aggregation, sorting, join, top-k, skyline, search, entity resolution).*

1. Introduction

As a novel computation paradigm, crowdsourcing [Howe, 2008] or human computation [Law and von Ahn, 2011] is being actively pursued in diverse academic disciplines. Within computer science, many sub-fields have embraced the concept of crowdsourcing with open arms and applied the concept to solve diverse challenging problems. Database community is no exception to this phenomenon and there have been many exciting new results using crowdsourcing appearing in recent database literature. For instance, people have extended existing DBMS techniques to address data management questions that are difficult for machines but easier for humans and proposed novel systems such as: CrowdDB [Franklin et al., 2011] and Deco [Park et al., 2012].

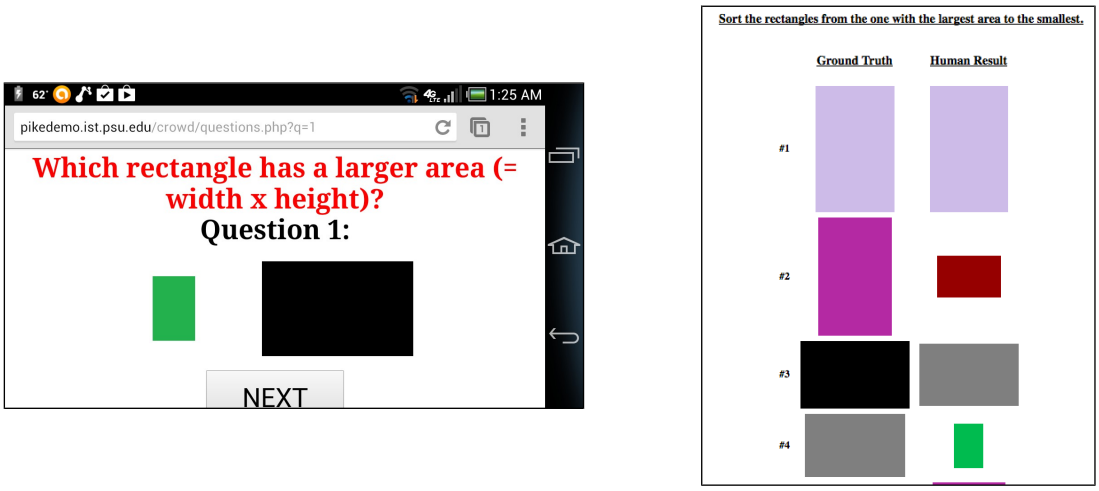
In addition to recent tutorials by other scholars that covered broad issues of crowdsourcing in general and contributed to the increased interests in the community, this tutorial seeks to cover more narrow topics with the focus of database, covering state-of-the-art human-powered algorithms in data management.

After gentle introduction on the concept of crowdsourcing, this tutorial will provide the overall landscape of crowdsourced database research, with the focus on some of latest *human-powered database operations* that extend conventional ones such as:

- Crowdsourced Counting and Sorting [Marcus et al., 2011; Marcus et al., 2012]

- Crowdsourced Top-1 and Top-k [Venetis et al., 2011; Guo et al., 2012; Davidson et al., 2013; Polychronopoulous et al., 2013]
- Crowdsourced Join [Wang et al., 2012; Wang et al., 2013; Whang et al., 2013; Gokhale et al., 2014]
- Crowdsourced Search [Parameswaran et al., 2011]

In addition, to help audience understand human-powered algorithms better and have an immersed learning experience, a mobile-device based hands-on demo will be used to demonstrate the power of human-powered sorting, as shown in Figure 1.



(a) Client side (b) Admin side

Figure 1. Screenshot of human-powered sort demo.

About the Author

Dongwon Lee is an associate professor of the Pennsylvania State University, College of Information Sciences and Technology, USA. I obtained my Ph.D. in Computer Science from UCLA in 2002. From 1995 to 1997, I have also worked at AT&T Bell Labs. Working mostly on the issues arising in the management and mining of data, I have (co-)authored over 110+ scholarly articles in selective publication outlets in Databases and Data Mining fields.

References

Davison, S. et al. Using the Crowd for Top-K and GrouP-By Queries, In *ICDT*, 2013.

Franklin, M. et al. CrowdDB: Answering Queries with Crowdsourcing, In *SIGMOD,* 2011.

Gokhale, C. et al. Corleone: Hands-Off Crowdsourcing for Entity Matching, In *SIGMOD*, 2014.

Guo, S. et al. So Who Won?: Dynamic Max Discovery with the Crowd, In *SIGMOD*, 2012.

Howe, J. Crowdsourcing: Why the Power of the Crowd is Driving the Future of Business, *Crown Business*, 2008.

Law, E. and von Ahn, L. *Human Computation*, Morgan & Claypool Publishers, 2011.

Marcus, E., Wu, D., Karger, S., Madden, S., and Miller, R. Human-Powered Sorts and Joins, In *VLDB*, 2011.

Marcus, E., Karger, S., Madden, S., Miller, R., and Oh, S. Counting with the Crowds, In *VLDB*, 2012.

Parameswaran, A., Das Sarma, A., Garcia-Molina, H., Polyzotis, N., and Widom, J. Human-Assisted Graph Search: It's Okay to Ask Questions, In *VLDB*, 2011.

Park, H., Garcia-Molina, H., Pang, R., Polyzotis, N., Parameswaran, A., and Widom, J. Deco: A System for Declarative Crowdsourcing, In *VLDB*, 2012.

Polychronopoulous, V. et al. Human-Powered Top-K Lists, In *WebDB*, 2013.

Venetis, P. et al. Max Algorithms in Crowdsourcing Environments, In *WWW*, 2012.

Wang, J., Kraska, T., Franklin, M., and Feng, J. CrowdER: Crowdsourcing Entity Resolution, In *VLDB*, 2012.

Wang, J., Li, G., Kraska, T., Franklin, M., and Feng, J. "Leveraging Transitive Relations for Crowdsourced Joins, In *SIGMOD*, 2013.

Whang, S., Lofgren, P., and Garcia-Molina, H. Question Selection for Crowd Entity Resolution, In *VLDB*, 2013.

paper:68

Métodos Automáticos para Desambiguação de Nomes de Autores em Repositórios de Dados Bibliográficos

Anderson A. Ferreira[1], Marcos André Gonçalves[2], Alberto H. F. Laender[2]

[1]Departamento de Computação – Universidade Federal de Ouro Preto
Campus Universitário Morro do Cruzeiro – CEP 35400-000 – Ouro Preto, MG, Brasil

[2]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627 – CEP 31270-010 – Belo Horizonte, MG, Brasil

ferreira@iceb.ufop.br, {laender,mgoncalv}@dcc.ufmg.br

Resumo. *Ambiguidade de nomes de autores é um problema que ocorre quando um conjunto de registros de citações bibliográficas contém nomes de autores ambíguos, ou seja, quando um mesmo autor aparece com nomes distintos em suas publicações ou quando diferentes autores possuem o mesmo nome (ou a mesma variação de nome) em suas publicações. O desafio envolvido na resolução deste problema tem levado ao desenvolvimento de inúmeros métodos de desambiguação. O objetivo deste tutorial é dar uma visão geral dos principais métodos de desambiguação de nomes de autores propostos na literatura e apresentar os principais desafios deste problema.*

1. Introdução

Bibliotecas digitais acadêmicas, tais como DBLP[1], CiteSeer[2], MEDLINE[3] e BDBComp[4], são repositórios bibliográficos que possuem serviços que facilitam a busca de trabalhos científicos e pesquisadores, bem como a descoberta de conhecimento no cenário científico. Esses sistemas podem conter milhões de registros de dados bibliográficas e são uma importante fonte para a descoberta de publicações relevantes e pesquisadores de interesse de maneira centralizada. Vários estudos interessantes podem ser conduzidos com base nos dados desses repositórios, tais como análise de tendências de pesquisa, avaliação da qualidade e impacto das publicações de um subcomunidade ou de um indivíduo, descoberta de padrões de colaboração, dentre outros. Os resultados desses estudos e análises podem ser utilizados, por exemplo, por agências de fomento ou instituições de pesquisa em decisões sobre financiamento de projetos ou promoção de indivíduos, e para tal pressupõem que os dados sejam de *alta qualidade* [Laender et al. 2008; Lee et al. 2007].

De acordo com Lee et al. [2007], os problemas relacionados à qualidade dos dados disponíveis nas bibliotecas digitais têm várias origens, tais como erros na entrada dos dados, falta (ou a exigência) de padrões, imperfeição na coleta de

1 http://dblp.uni-trier.de
2 http://citeseer.ist.psu.edu
3 http://medline.cos.com
4 http://www.lbd.dcc.ufmg.br/bdbcomp

metadados, ambiguidade de nomes de autores, abreviações de títulos dos veículos de publicação e geração de metadados em larga escala.

Dentre esses, o problema de ambiguidade de nomes de autores vem recebendo muita atenção da comunidade de pesquisa, devido à sua inerente dificuldade. Mais especificamente, este problema ocorre quando um conjunto de registros bibliográficos contém nomes de autores ambíguos, ou seja, quando um mesmo autor aparece com nomes distintos em suas publicações ou quando diferentes autores possuem o mesmo nome (ou a mesma variação de nome) em suas publicações. Este problema pode ser causado por diversas razões [McKay et al. 2010], incluindo mudanças de nomes devido a circunstâncias pessoais, variações em transliteração de nomes não romanos, erros de tipografia, falta de padrões e geração de conteúdo de forma descentralizada (*i.e.*, por meio de coleta automática de metadados [Lagoze and Sompel 2001]).

Os desafios envolvidos na resolução deste problema têm levado ao desenvolvimento de inúmeros métodos de desambiguação [Ferreira et al. 2012b]. O objetivo deste tutorial é dar uma visão geral dos principais métodos de desambiguação de nomes de autores propostos na literatura e apresentar os principais desafios de pesquisa relacionados ao problema, estando dividido em duas partes.

A primeira parte é baseada no artigo "A Brief Survey of Automatic Methods for Author Name Disambiguation" publicado no SIGMOD Record em junho de 2012 [Ferreira et al. 2012b]. Nessa primeira parte, apresentamos a motivação e a definição (formal) do problema, definimos uma taxonomia geral para caracterizar os métodos automáticos de desambiguação de nomes de autores propostos na literatura e descrevemos alguns métodos para ilustrar a taxonomia proposta. Concluímos essa parte inicial com alguns desafios de pesquisa em aberto na área.

Na segunda parte, apresentamos soluções de nossa autoria, algumas das quais consideradas o estado-da-arte atual da área. Inicialmente, apresentamos HHC – Heuristic-based Hierarchical Clustering [Cota et al. 2010], um método não supervisionado que utiliza algumas heurísticas específicas para lidar com o problema de ambiguidade de nomes de autores. Apesar de sua simplicidade, ele produz resultados muito competitivos quando comparado com outros métodos descritos na literatura. A seguir, apresentamos INDi – Incremental Name Disambiguation [Carvalho et al. 2011], um método desenvolvido para permitir a desambiguação *incremental* de um repositório, ou seja, desambiguar apenas os nomes de autores de registros que estão sendo inseridos no repositório em um dado momento no tempo, ao invés de desambiguar os nomes de autores de todos os registros já contidos nesse repositório, o que é feito pela maioria dos métodos. Por fim, apresentamos SAND – Self-training Author Name Disambiguator [Ferreira et al. 2014], um método que usa um classificador altamente eficaz adaptado para o problema de ambiguidade de nomes. Apesar de explorar classificadores, SAND tem como uma de suas principais vantagens o fato de selecionar os exemplos de treinamento de *forma automática* sem a intervenção do usuário. Além disso, SAND não requer nenhuma especificação de parâmetros e trabalha com informação escassa, ou seja, usa apenas as informações mais comumente encontradas nos registros bibliográficos, a saber, nomes dos autores e os títulos dos trabalhos e dos veículos de publicação. Para cada um dos métodos,

apresentamos resultados experimentais com coleções reais e sinteticamente geradas. Em particular, para a geração das coleções sintéticas, abordamos também o uso da ferramenta SyGAR [Ferreira et al. 2012a], que é um gerador de registros bibliográficos sintéticos que pode ser parametrizado para gerar registros para novos autores (autores sem nenhum registro de citação no repositório), alterar as áreas de interesse dos autores ao longo do tempo e adicionar erros aos registros, permitindo assim simular várias situações reais de maneira controlada, o que é difícil quando se utiliza apenas coleções reais.

Sobre os Autores

Alberto H. F. Laender é Engenheiro Eletricista (1974) e mestre em Ciência da Computação (1979) pela UFMG, e doutor em Computação (1984) pela University of East Anglia, Inglaterra. É Professor Titular do Departamento de Ciência da Computação da UFMG e bolsista de Produtividade em Pesquisa 1A do CNPq. Atua nas áreas de bancos de dados, gerência de dados da Web, bibliotecas digitais e sistemas de informação para a Web, tendo sido coordenador ou membro do comitê de programa de importantes conferências e workshops como SBBD, ER, CIKM, SPIRE, SIGMOD, WWW, VLDB, TPDL, JCDL, AMW e WIDM. Atualmente é membro da Comissão de Avaliação da Área de Ciência da Computação da CAPES e coordenador da linha de pesquisa em Gerência de Dados da Web do Instituto Nacional de Ciência e Tecnologia para a Web. Foi um dos fundadores da Akwan Information Technologies, empresa de tecnologia de busca na Web adquirida pela Google Inc. em julho de 2005. Em 2010 foi eleito Membro Titular da Academia Brasileira de Ciências e agraciado com a Ordem Nacional do Mérito Científico na classe Comendador.

Marcos André Gonçalves possui graduação em Ciência da Computação pela Universidade Federal do Ceara (1995), mestrado em Ciência da Computação pela Universidade Estadual de Campinas (1997), doutorado em Computer Science pela Virginia Polytechnic Institute and State University (Virginia Tech) e pós-doutorado pela Universidade Federal de Minas Gerais (2006). Atualmente é professor Adjunto da Universidade Federal de Minas Gerais. Recebeu diversos prêmios e homenagens ao longo de sua carreira. Atua na área de Ciência da Computação com ênfase em Recuperação de Informação, Bibliotecas Digitais e Banco de Dados. É atualmente Membro Afiliado da Academia Brasileira de Ciências, Bolsista de Produtividade do CNPq (nível 1-D) e Bolsista do Programa Pesquisador Mineiro da Fapemig.

Anderson Almeida Ferreira é bacharel em Informática pela Universidade Federal de Viçosa e mestre e doutor em Ciência da Computação pela Universidade Federal de Minas Gerais. Enquanto aluno de doutorado do Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, participou de vários projetos de pesquisa desenvolvidos no Laboratório de Bancos de Dados (LBD) e integrou a equipe do Instituto Nacional de Ciência e Tecnologia para a Web - InWeb. Tem diversos trabalhos publicados nas principais conferências e periódicos de Bibliotecas Digitais e Banco de Dados, tais como, JCDL, SBBD/JIDM, JASIST, IPM e SIGMOD Record. Atualmente, é professor do Departamento de Computação da Universidade Federal de Ouro Preto e integrante do grupo de pesquisa Gerência e Análise Inteligente de Dados - GAID.

Referências

Carvalho, A. P., Ferreira, A. A., Laender, A. H. F., and Gonçalves, M. A. Incremental unsupervised name disambiguation in cleaned digital libraries. *Journal of Information and Data Management*, 2(3):289–304, 2011.

Cota, R. G., Ferreira, A. A., Gonçalves, M. A., Laender, A. H. F., and Nascimento, C. An unsupervised heuristic-based hierarchical method for name disambiguation in bibliographic citations. *Journal of the American Society for Information Science and Technology*, 61(9):1853–1870, 2010.

Ferreira, A. A., Gonçalves, M. A., Almeida, J. M., Laender, A. H. F., and Veloso, A. A tool for generating synthetic authorship records for evaluating author name disambiguation methods. *Information Sciences*, 206:42–62, 2012a.

Ferreira, A. A., Gonçalves, M. A., and Laender, A. H. F. A brief survey of automatic methods for author name disambiguation. *SIGMOD Record*, 41(2):15–26, 2012b.

Ferreira, A. A., Veloso, A., Gonçalves, M. A., and Laender, A. H. F. Self-training author name disambiguation for information scarce scenarios. *Journal of the American Society for Information Science and Technology*. 65(6):1257-1278, 2014.

Laender, A.H.F., Gonçalves, M.A., Cota, R.G., Ferreira, A.A., Santos, R.L.T., and Silva, A.J.C. Keeping a Digital Library Clean: New Solutions to Old Problems. In *Proceedings of the 2008 ACM Symposium on Document Engineering*, São Paulo, Brazil, pages 257–262, 2008.

Lagoze, C. and Sompel, H. V. The openarchives initiative: building a low-barrier interoperability framework. In *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 54-62, Roanoke, Virginia, USA, 2001.

Lee, D., Kang, J., Mitra, P., Giles, C. L., and On, B. W. Are your citations clean? *Communications of the ACM*, 50(21): 33-38, 2007.

McKay, D., Sanchez, S. and Parker, R. What's my name again?: sociotechnical considerations for author name management in research databases. In *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*. Brisbane, Australia, pages 240-247, 2010.

talks

29th Brazilian Symposium on Databases

October 6th to 9th, 2014
Curitiba – PR – Brazil

INVITED TALKS

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Federal University of Paraná – UFPR
Pontifical University Catholic of Paraná – PUC-PR

Table of Contents (Invited Talks)

Big Text: from Language to Knowledge 379
Gerhard Weikum

Exploring Big Urban Data 381
Juliana Freire

Preference Mining for Personalization and Recommendation 383
Sandra de Amo

paper:67

Big Text: from Language to Knowledge

Gerhard Weikum

Max Planck Institute for Informatics
Campus E1.4 – D-66123 Saarbruecken, Germany

weikum@mpi-inf.mpg.de

Abstract. *News, social media, web sites, and enterprise sources produce huge amounts of valuable contents in the form of text and speech. To tap this wealth of unstructured Big Data and obtain insights, a decisive step is to identify the entities that are referred to and relationships between entities. This allows linking unstructured contents with structured data. However, this step faces the fundamental problem that names and phrases are often highly ambiguous; mapping them to entities and relations is a challenging task. The talk will discuss the state of the art, applications, and open problems on disambiguating named entities in text and heterogeneous tables. It will also put this line of research in perspective to the bigger picture of Big Data analytics.*

About the Author



Gerhard Weikum is a Scientific Director at the Max Planck Institute for Informatics in Saarbruecken, Germany, and also an Adjunct Professor at Saarland University. He graduated from the University of Darmstadt, Germany. Weikum's research spans transactional and distributed systems, self-tuning database systems, DB&IR integration, and the automatic construction of knowledge bases from Web and text sources. He co-authored a comprehensive textbook on transactional systems, received the VLDB 10-Year Award for his work on automatic database tuning, and is one of the creators of the YAGO knowledge base. Gerhard Weikum is an ACM Fellow, a Fellow of the German Computer Society, and a member of several European and German academies. He has served on various editorial boards, including Communications of the ACM, and as program committee chair of conferences like ACM SIGMOD, IEEE Data Engineering, and CIDR. From 2003 through 2009 he was president of the VLDB Endowment. He received the ACM SIGMOD Contributions Award in 2011 and has won an ERC Synergy Grant in 2013.

paper:96

Exploring Big Urban Data

Juliana Freire

New York University
2 Metrotech Pl, 10th floor – Brooklyn, NY 11201, USA
juliana.freire@nyu.edu

Abstract. *Today, 50% of the world's population lives in cities and the number will grow to 70% by 2050. Cities are thus the loci of economic activity, and will continue to be the source of many of the innovations and novel approaches to the challenges of the 21st century. At the same time, cities are also the cause of looming sustainability problems and face huge challenges around, for example, transportation, resource consumption, housing affordability, and inadequate or aging infrastructure. The large volumes of urban data currently available, along with vastly increased computing power, open up new opportunities for us to better understand cities. In fact, there are already successful stories that have resulted in better operations, informed planning, improved policies and a better quality of life for citizens. However, analyzing urban data often requires a staggering amount of work, from identifying relevant data sets, cleaning and integrating them, to performing exploratory analyses over complex, spatio-temporal data. An important long-term goal of our research is to enable interdisciplinary teams to crack the code of cities by freely exploring the vast amounts of data cities generate. In this talk, we will discuss some of the challenges involved which have led us to pursue new research on data management, data analysis and visualization techniques. We also will present methods and systems we have developed to increase the level of interactivity, scalability, and usability for spatio-temporal analyses. This work was supported in part by the National Science Foundation, a Google Faculty Research award, the Moore-Sloan Data Science Environment at NYU, IBM Faculty Awards, NYU School of Engineering and Center for Urban Science and Progress.*

About the Author



Juliana Freire is a Professor at the Department of Computer Science and Engineering at New York University. She holds an appointment at the Courant Institute for Mathematical Science, is a faculty member at the NYU Center for Urban Science and at the NYU Center of Data Science, where she is also the Director of Graduate Studies. Before, she was an Associate Professor at the School of Computing, University of Utah; an Assistant Professor at OGI/OHSU; and a member of technical staff at the Database Systems Research Department at Bell Laboratories (Lucent Technologies). An important theme is Professor Freire's work is the development of data management technology to address new problems introduced by emerging applications that manipulate different kinds of data, including urban, scientific and Web data. Her recent research has focused on big-data analysis and visualization, large-scale information integration, provenance management, and computational reproducibility. Professor Freire is an active member of the database and Web research communities, having co-authored over 140 technical papers, several open-source systems, and holding 10 U.S. patents. She is a recipient of an NSF CAREER, two IBM Faculty awards, and a Google Faculty Research award. She has chaired or co-chaired several workshops and conferences, and she has participated as a program committee member in over 70 events. Her research has been funded by grants from the National Science Foundation, DARPA, Department of Energy, National Institutes of Health, Sloan Foundation, Betty Moore Foundation, W. M. Keck Foundation, Google, Amazon, the University of Utah, New York University, Microsoft Research, Yahoo! and IBM.

paper:97

Preference Mining for Personalization and Recommendation

Sandra de Amo

Universidade Federal de Uberlândia
Av. João Naves de Avila, 2121 – 38400-902 – Uberlândia, MG, Brazil
deamo@ufu.br

Abstract. *The emerging of ubiquitous computing technologies in recent years has given rise to a new field of research consisting in incorporating context-aware preference querying facilities in database systems. One important step in this setting is the Preference Elicitation task which consists in designing automatic strategies for extracting preference information from the user without being invasive and bothersome. This talk is divided into three parts. The first part discusses formalisms for expressing preference data and presents different data mining techniques for extracting preference models from preference input data based in such formalisms. The second part focuses on the use of preference mining techniques for personalization. We present the* ***CPrefSQL*** *query language that extends the standard SQL with preference operators and incorporates preference models produced by preference mining algorithms in order to personalize query results. And finally, in the third part, we show how the preference mining techniques can be used for developing efficient recommendation systems. We present* ***PrefRec****, a general framework for developing recommendation systems using preference mining and preference aggregation techniques.*

About the Author



Sandra de Amo is Associate Professor in the Faculty of Computer Science at the Federal University of Uberlândia, Brazil. She received her PhD in Computer Science from the University of Paris 13, France, in 1995 and her MSc in Mathematics from the Institute or Pure and Applied Mathematics (IMPA-Rio de Janeiro, Brazil). Her research interests include Data Mining, Query Languages, Preference Modeling and Reasoning, Personalization and Recommendation, Temporal Databases and Inconsistent Information Management in Databases. During the last fifteen years most of her research work in Data Mining, Query Languages and Preference Modeling and Reasoning has been in collaboration with researchers in LABRI (Laboratoire Bordelais d'Informatique – Université de Bordeaux I), LIG (Laboratoire d'Informatique de Grenoble - Université de Grenoble) and LI – Tours (Laboratoire d'Informatique - Université de Tours. She coordinated two research cooperation networks between France-Brazil in the last years. She has a research productivity grant from CNPq since 2007. She is currently a member of the Advisory Board of Exact Sciences of FAPEMIG (Foundation for Research Support of Minas Gerais, Brazil). She has integrated the Database Special Commission of the SBC from 2007 to 2011 and has been PC chair of the 20th SBBD in 2008. She is an affiliate member of the ACM since 1999 and of the SBC since 1995.

Promoção



Apoio









