www.hypergraphdb.org
Borislav Iordanov

HyperGraphDB – Graph Database

PPGInf - UFPR
CI829 - Oficina de Banco de Dados I
Profa. Carmem Satie Hara

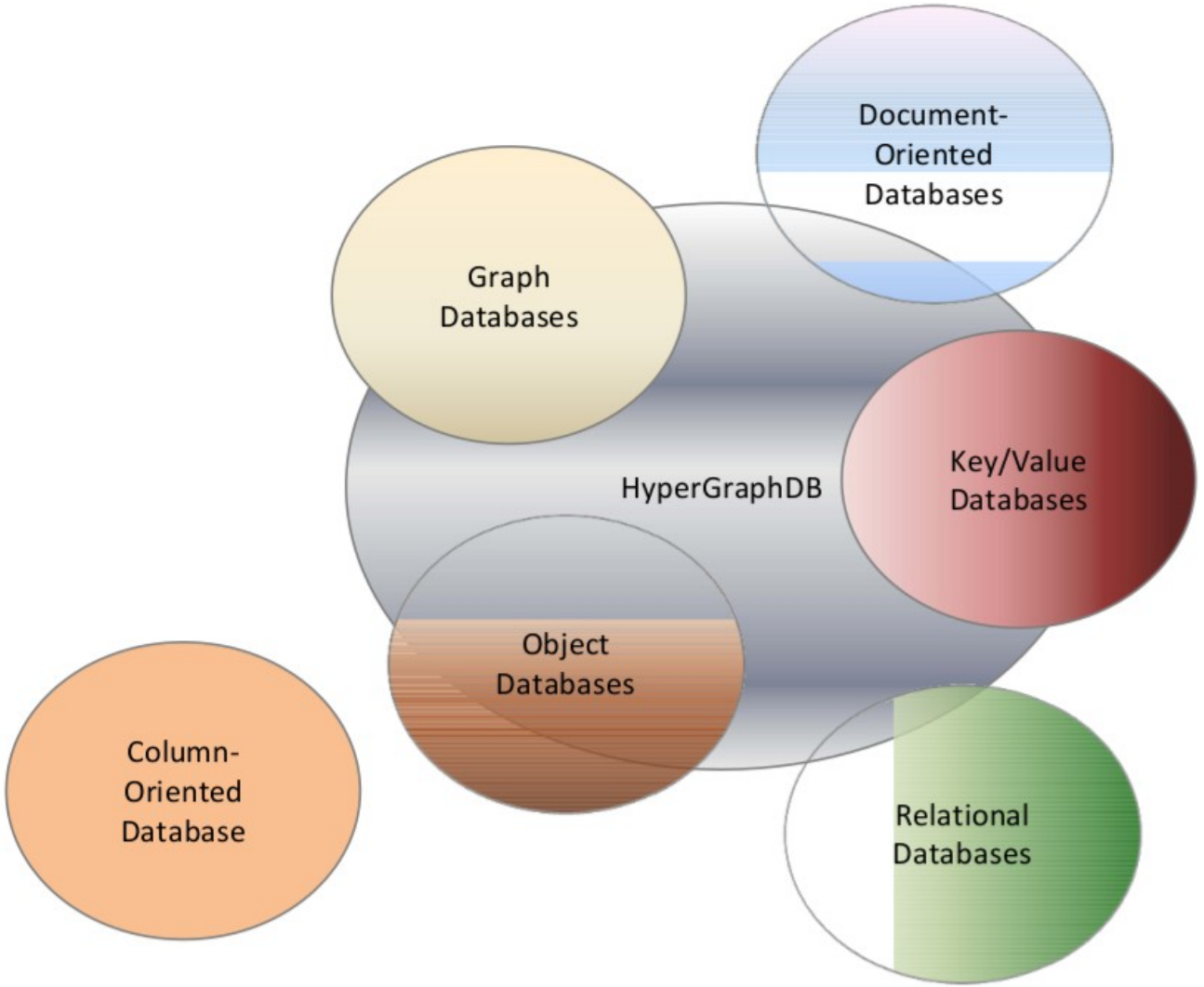
Ricardo Masashi Maeda

Introdução

- O que é HyperGraphDB?
  - Um banco de dados para armazenar **hipergrafos** (!)
  - **Não** é estritamente em **Grafo**
  - Pode ser relacional (NoSQL) ou em *key-value*
  - Mais próximo a **orientado a objetos** (**OO**)
  - Usos: IA, Web Semântica e *Embedded OO Databases*

HyperGraphDB

Introdução



HyperGraphDB

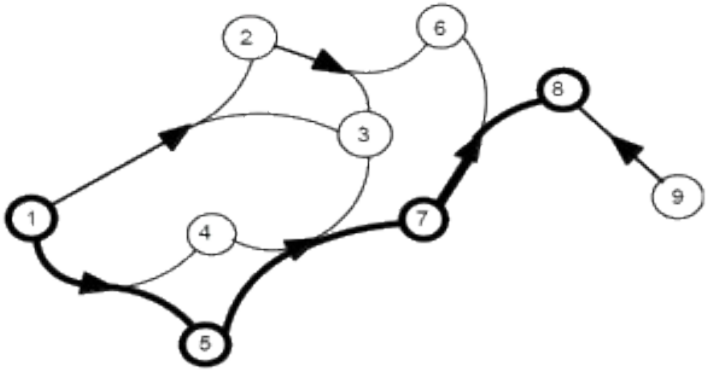
Introdução

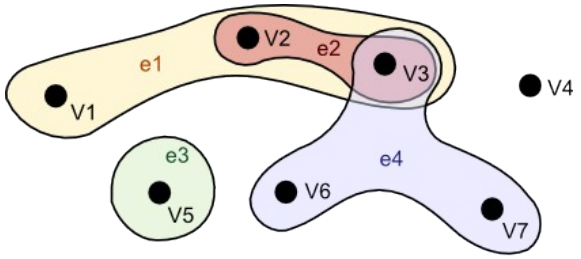
- O que é Hipergrafo?
  - É uma **generalização** do conceito de grafos
  - **Arestas** são **conjuntos** ao invés de pares
  - Permite uma **aresta** apontar para **mais de dois nós**
  - Uma **aresta** pode apontar para outra **aresta**
  - O conceito de hipergrafo direcionado não é tão trivial

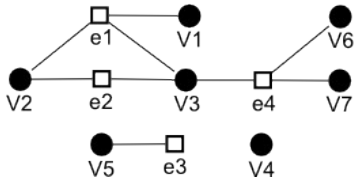
HyperGraphDB

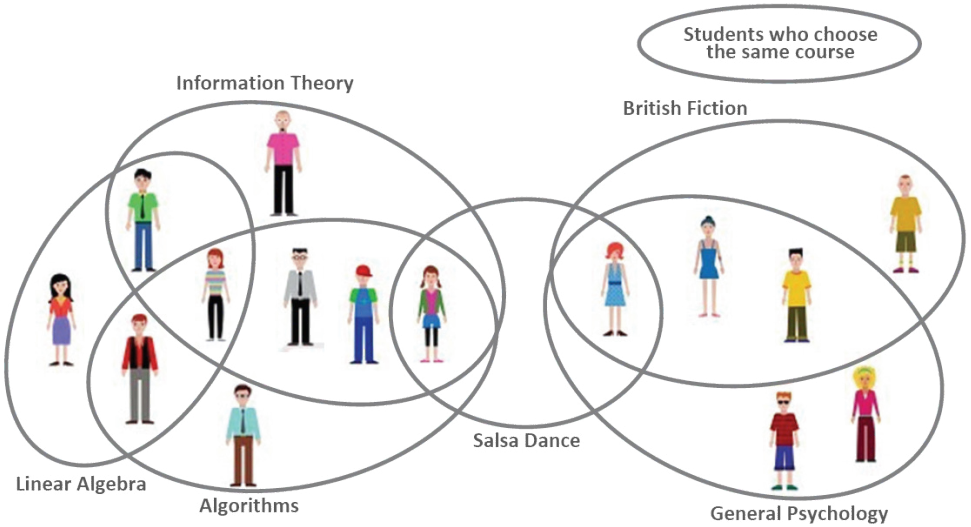
Introdução

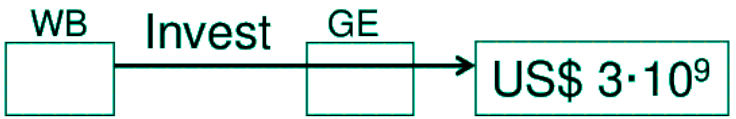
- Exemplos de Hipergrafos







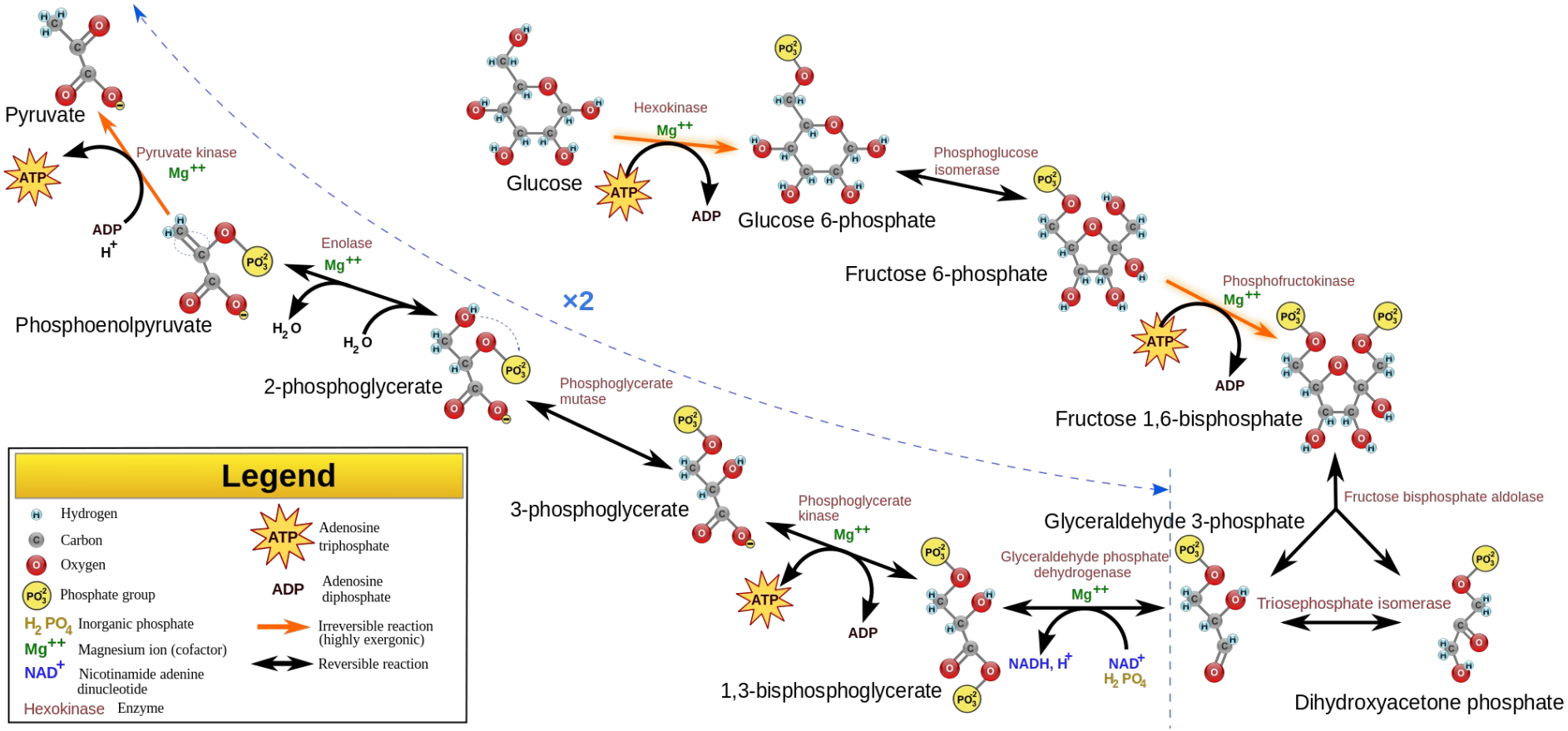




HyperGraphDB

Introdução

- Exemplos de Hipergrafos



HyperGraphDB

Características

- Armazenamento **chave-valor**: BerkeleyDB
- Open-source: LGPL
- Distribuído (P2P framework)
  - Particionamento e Replicação (-)
- *Open-architecture* (extensível e de propósito geral)
- ***Embedded***
- **Portável**

ORACLE
BERKELEY DB

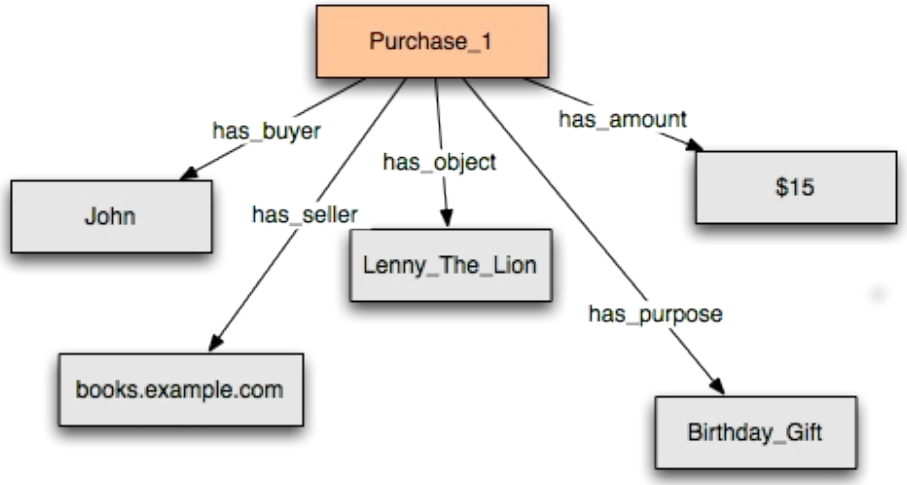
LGPL v3 Free Software

XMPP

HyperGraphDB

Características

- Relacionamento **N-ário**
- **Transacional**
- Multi-thread
- **MVCC**
  - Escritas e leituras não bloqueantes
- Customizável (índices, armazenamento, tipos, etc)
- Ausência de metadado e linguagem de manipulação / administração



HyperGraphDB

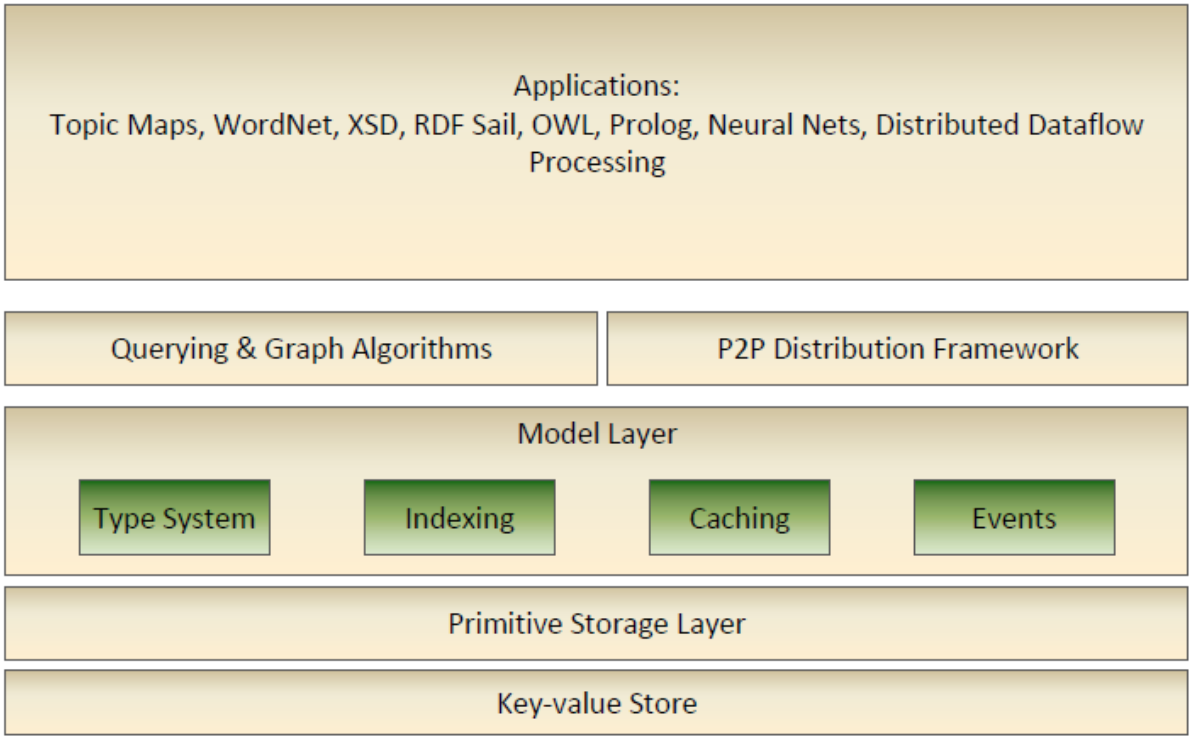
Nomenclatura

- ***Atom***: **unidade básica** de armazenamento
  - *Vertexes* e *Edges* = Atoms
- ***Target set(x)****:* conjunto de átomos, que x está relacionado
- *Arity(x)*: cardinalidade / tamanho do *target set*
- ***Incidence set(x)****:* conjunto de átomos, que possuem *x* em seus *Target set*.
- *Type(x)*: similares a classes em OO ou *datatypes* em databases (átomo)
- *Value(x)*: dado gerenciado no storage pelo *Type(x)*

HyperGraphDB

Arquitetura

- Duas camadas:
  - Primitiva
  - Modelo



HyperGraphDB

Arquitetura

- Camada primitiva
  - Array associativo (**chave-valor**)
  - Cada ID mapeia para uma tupla de IDs ou para um array de bytes (grafo)

$$LinkStore : ID \rightarrow List < ID >$$
$$DataStore : ID \rightarrow List < byte >$$

HyperGraphDB

Arquitetura

- Camada modelo
  - **Abstração do hipergrafo** é implementada nesta camada
  - Cada ID representa um atom ou um valor

$$AtomID \rightarrow [TypeID, ValueID, TargetID, ..., TargetID]$$
$$TypeID \rightarrow AtomID$$
$$TargetID \rightarrow AtomID$$
$$ValueID \rightarrow List < ID > | List < byte >$$

HyperGraphDB

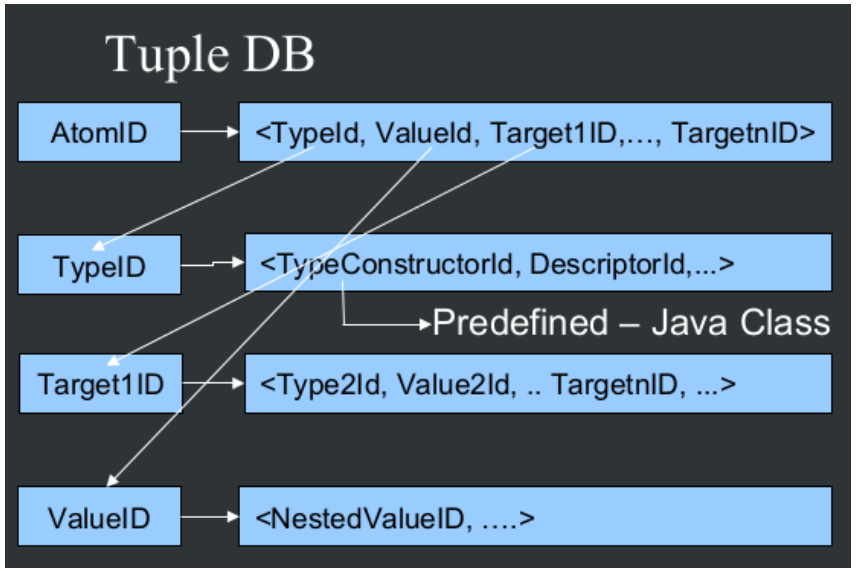
Arquitetura

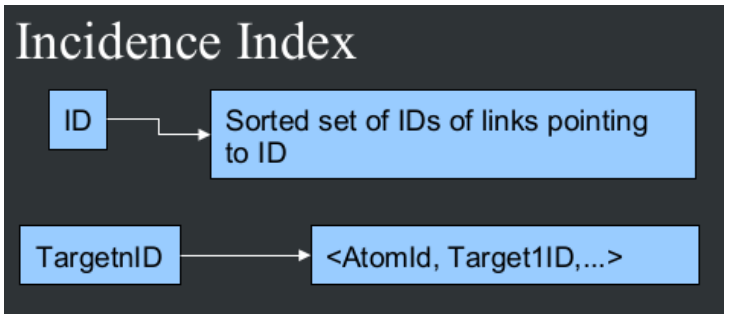
- Índices implícitos

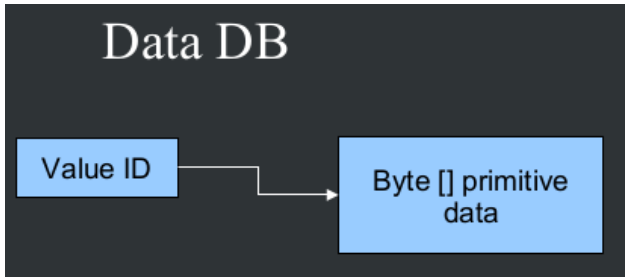
$$IncidenceIndex : UUID \rightarrow SortedSet < UUID >$$
$$TypeIndex : UUID \rightarrow SortedSet < UUID >$$
$$ValueIndex : UUID \rightarrow SortedSet < UUID >$$

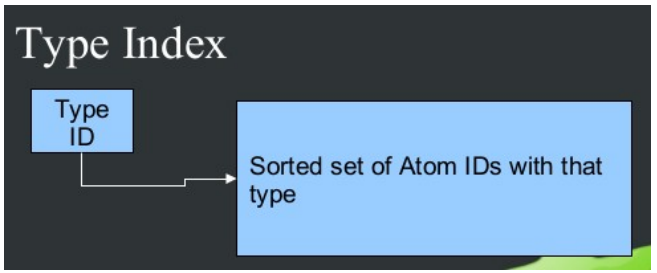
HyperGraphDB

Arquitetura



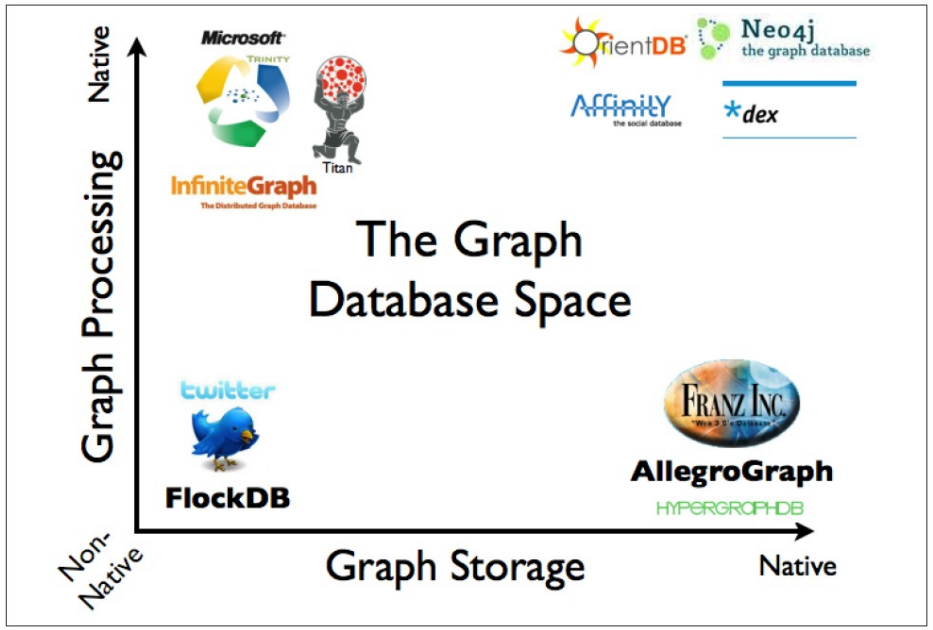






HyperGraphDB

Categorização



- Graph Processing
  - "it exhibits a property called index-free adjacency, meaning that connected nodes physically 'point' to each other in the database"
- Graph Storage
  - Armazenamento chave-valor
  - Orientado a grafo

HyperGraphDB

Instalação

- Teoricamente **não existem passos** para instalação do HyperGraphDB
- Basta incorporar as bibliotecas na aplicação
- Administração é **realizada pela aplicação (API)**
- Não há comunicação cliente/servidor

HyperGraphDB

Administração Bases de Dados

```
$ export CLASSPATH=./hgdb-1.2.jar:./hgbdbje-1.2.jar:./je-5.0.34.jar:.
$ javac HGDBCreateSample.java
$ java HGDBCreateSample
checkpoint kbytes:0
checkpoint minutes:0
hello world
$
```

- HGDB é incorporado na aplicação

```
import org.hypergraphdb.*;

public class HGDBCreateSample
{
    public static void main(String [] args)
    {
        HyperGraph graph = new HyperGraph("/hgdb/helloworld");
        String hello = graph.get(graph.add("Hello World"));
        System.out.println(hello.toLowerCase());
        graph.close();
    }
}
```

- Atividades de inicialização, criação, interrupção são gerenciadas pela aplicação

```
$ ls -tlr /hgdb/helloworld/
total 72
-rw-r--r-- 1 ricardomaeda users     0 Oct  6 03:13 je.info.0
-rw-r--r-- 1 ricardomaeda users     0 Oct  6 03:13 je.lck
-rw-r--r-- 1 ricardomaeda users 71989 Oct  6 03:13 00000000.jdb
$
```

HyperGraphDB

Configuração Bases de Dados

```
import org.hypergraphdb.*;

public class HGDBCreateSample
{
    public static void main(String [] args)
    {
        HGConfiguration config = new HGConfiguration();
        config.setTransactional(false);
        HyperGraph graph = new HyperGraph("/hgdb/helloworld");
        String hello = graph.get(graph.add("Hello World"));
        System.out.println(hello.toLowerCase());
        graph.close();
    }
}
```

- Alteração dos parâmetros só é válida na instanciação
- Poucos parâmetros

HyperGraphDB

Manipulação dos Átomos

Inserção

```
import org.hypergraphdb.*;

public class HGDBCreateSample
{
    public static void main(String [] args)
    {
        HyperGraph graph = new HyperGraph("/hgdb/helloworld");

        Book mybook = new Book("Critique of Pure Reason", "E. Kant");
        HGHandle bookHandle = graph.add(mybook);
```

Atualização

```
    public static void main(String [] args)
    {
        HyperGraph graph = new HyperGraph("/hgdb/helloworld");

        Book mybook = new Book("Critique of Pure Reason", "E. Kant");
        HGHandle bookHandle = graph.add(mybook);
        mybook.setYearPublished(1988);
        graph.update(mybook);
```

Remoção

```
    public static void main(String [] args)
    {
        HyperGraph graph = new HyperGraph("/hgdb/helloworld");

        Book mybook = new Book("Critique of Pure Reason", "E. Kant");
        HGHandle bookHandle = graph.add(mybook);
        graph.remove(bookHandle);

        graph.close();
    }
```

HyperGraphDB

Linking Átomos

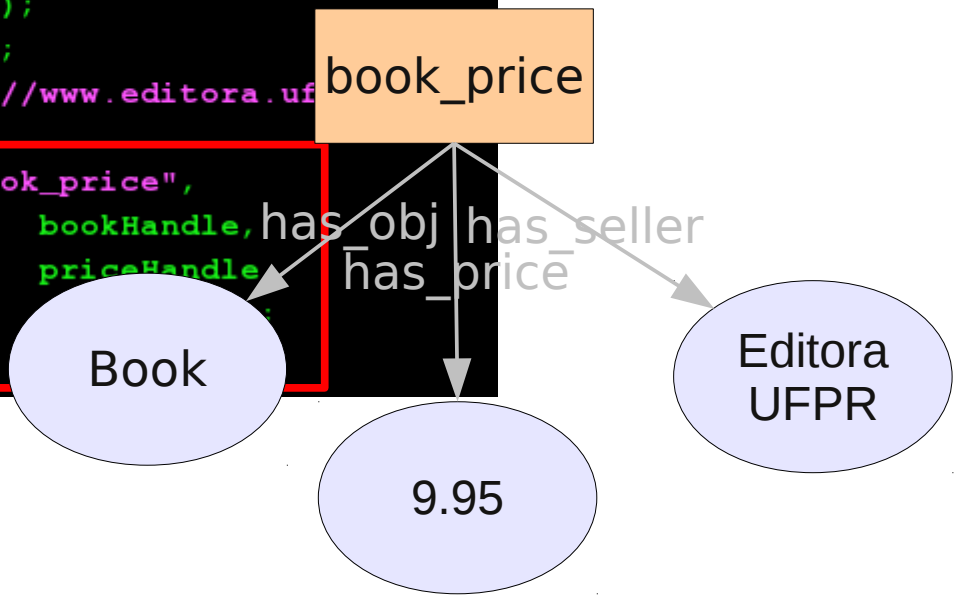
- Relembrando...
  - Arestas e vértices (átomos) apontam para um número arbitrário de elementos
  - Arestas podem apontar para outras arestas

```
public static void main(String [] args)
{
      HyperGraph graph = new HyperGraph("/hgdb/helloworld");

      Book mybook = new Book("Dom Casmurro", "Machado de Assis");
      HGHandle bookHandle = graph.add(mybook);
      HGHandle priceHandle = graph.add(9.95);
      HGHandle shopHandle = graph.add("http://www.editora.uf

      HGValueLink link = new HGValueLink("book_price",
                                         bookHandle,
                                         priceHandle
                                         );

      HGHandle linkHandle = graph.add(link);
```



Querying

```
HyperGraph graph = new HyperGraph("/hgdb/helloworld");

Book mybook = new Book("Dom Casmurro", "Machado de Assis");
HGHandle bookHandle = graph.add(mybook);
HGHandle priceHandle = graph.add(9.95);
HGHandle shopHandle = graph.add("http://www.editora.ufpr.br");

HGValueLink link = new HGValueLink("book_price", bookHandle, priceHandle, shopHandle);
```

```
List<HGHandle> bookAtoms = hg.findAll(graph,
                               hg.and(
                                  hg.type(Book.class),
                                  hg.eq("title", "Dom Casmurro")
                               )
                              );

for (HGHandle b : bookAtoms) {
        Book book = graph.get(b);
        System.out.println(book.getTitle());
}
```

HyperGraphDB

Querying

```
HyperGraph graph = new HyperGraph("/hgdb/helloworld");

Book mybook = new Book("Dom Casmurro", "Machado de Assis");
HGHandle bookHandle = graph.add(mybook);
HGHandle priceHandle = graph.add(9.95);
HGHandle shopHandle = graph.add("http://www.editora.ufpr.br");

HGValueLink link = new HGValueLink("book_price", bookHandle, priceHandle, shopHandle);
```

```
List<Book> books = hg.getAll(graph,
                             hg.and(
                                hg.type(Book.class),
                                hg.eq("title", "Dom Casmurro")
                             )
                            );

for (Book b : books)
        System.out.println(b.getTitle());
```

HyperGraphDB

Querying Graph

- hg.target(x)
- hg.incident(x)
- hg.arity(n)
- hg.disconnected()
- ...

```
HGValueLink link = new HGValueLink("book_price",
                                   bookHandle,
                                   priceHandle,
                                   shopHandle);
HGHandle linkHandle = graph.add(link);

HGQueryCondition cond = new And(
                        hg.type(Book.class),
                        hg.target(linkHandle));
HGSearchResult<HGHandle> rs = graph.find(cond);
while (rs.hasNext()){
        HGHandle current = rs.next();
        Book book = graph.get(current);
        System.out.println(book.getTitle());
}
rs.close();
```

HyperGraphDB

Graph Traversal

- Travessia ou percurso do grafo
- Atualmente dois algoritmos de travessia:
  - Busca em largura
  - Busca em profundidade

```
HGHandle linkHandle = graph.add(link);

HGDepthFirstTraversal traversal =
    new HGDepthFirstTraversal(bookHandle, new SimpleALGenerator(graph));

while (traversal.hasNext())
{
    Pair<HGHandle, HGHandle> current = traversal.next();
    HGLink l = (HGLink)graph.get(current.getFirst());
    Object atom = graph.get(current.getSecond());
    System.out.println("Visiting atom " + atom + " pointed to by " + l);
}
```

```
$ java HGDBCreateSample | cut -f1 -d'['
checkpoint kbytes:0
checkpoint minutes:0
Visiting atom Great book! pointed to by review
Visiting atom http://www.editora.ufpr.br pointed to by book_price
Visiting atom 9.95 pointed to by book_price
```

HyperGraphDB

Índices

- Além dos índices implícitos, é possível criar índices adicionais pelos atributos dos átomos
  - **Ex: author ou title ou price**
- Poucas opções existentes:
  - **ByPartIndexer**: criação de índice baseado no atributo de um átomo

HyperGraphDB

Transações

- ACI somente
  - Não são duráveis
  - Transações recentes e *committed* podem ser perdidas
- Toda modificação é encapsulada em uma transação (implícito)
- Detecção de deadlocks (-)
- Porém, um bloco transacional pode ser criado (explícito):

```
graph.getTransactionManager().beginTransaction();
try
{
   // conjunto de transações
   ...
   graph.getTransactionManager().commit();
}
catch (Throwable t)
{
   // Em caso de exceção, efetuar rollback
   graph.getTransactionManager().abort();
}
```

HyperGraphDB

Em estudo...

- Buscas *full-text*
- *Nested Graphs*
- *In-memory Graphs*
- C++
- Ferramentas / Utilitários de administração do HGDB
- Outras *Storage Engines*

HyperGraphDB

Referências

- Iordanov, Borislav. HyperGraphDB: A Generalized Graph Database. WAIM 2010.
- Iordanov, Borislav. HyperGraphDB - Data Management for Complex Systems. Strange Loop 2010.
- HypergraphDB homepage. http://www.hypergraphdb.org
- Robinson, I. et al. Graph Databases. O'Reilly. 2013
- Bretto, Allain et al. Hypergraph-based Image Representation. 2005
- Puente, Victor. HyperGraphDB: grafos e NoSQL para sistemas complexos. TDC 2012.