Universidade Federal do Paraná

Adolfo Welynton Sabino

Implementação de Framework de Modelos Epidemiológicos Estocásticos Compartimentais sobre Grafos

Curitiba, Paraná
2017

Adolfo Welynton Sabino

Implementação de Framework de Modelos Epidemiológicos Estocásticos Compartimentais sobre Grafos

Trabalho apresentado como requisito parcial à conclusão do Curso de Ciência da Computação - Bacharelado, setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: André Vignatti.

Curitiba, Paraná
2017

Este trabalho é dedicado à memória de Altair Sabino, meu pai, e do mestre Alexandre Ibrahim Direne, modelos de cidadão.

Agradecimentos

Agradeço o professor André Vignatti pelas orientações no decorrer do trabalho, pela paciência e compreensão.

Aos meus colegas Thiago Beraldin e Felipe Trzaskowski pelo suporte nas questões técnicas de implementação e de documentação.

Aos meus amigos Bruno Guerios, Stephanie Cristine Rend pelo auxílio com a revisão do texto e Luis Felipe Teixeira Marin e Willian Koarata, pelo apoio no decorrer do trabalho.

Agradeço Maria de Lourdes de Sousa Ferreira, mãe, que muito se empenhou durante toda sua vida, e atravéz de inúmeras dificuldades, para que eu pudesse ter condições de chegar onde cheguei.

E por fim, aos mestres Alexandre Ibrahim Direne, André Luiz Pires Guedes, Egon Hilgenstieler, Elias Procópio Duarte Júnior e Renato Carmo, por serem infinita fonte de admiração e inspiração.

Resumo

A Epidemiologia é a ciência que estuda quantitativamente os padrões que ocorrem em massa na saúde sobre populações definidas. Uma de suas funções é descrever a distribuição e evolução de doenças de forma sistêmica, observando seus o padrões de ocorrência e evolução ao longo do tempo, bem como inferir a efetividade de intervenções nesses cenários. Pesquisadores se valem de uma grande variedade modelos matemáticos, que visam cobrir os elementos dinâmicos e estruturais das doenças e populações, que sejam relevantes para elucidar a compreensão desses fenômenos. Neste trabalho buscamos compreender a relevância da consideração da topologia das redes sociais descritas pelas relações entre indivíduos das populações destes modelos. Para isso propomos um framework de simulação de modelos epidemiológicos, e realizamos nele alguns experimentos usando topologias de redes encontradas na literatura. **Palavras-chave:** Modelos

Compartimentais de Epidemiologia, Grafos, Processos Estocásticos, Modelo SIR, Modelo SIS, Modelo SIRS, Topologia de Redes.

Abstract

Epidemiology is a science that seeks to study quantitatively patterns, causes, and effects of health that occur over defined populations. One of its functions is to describe the spread and evolution of diseases in a systemic way, observing their patterns of occurrence and evolution over time, as well as to infer the effectiveness of interventions in these scenarios. To achieve this, researchers rely on a wide variety of mathematical models that cover the dynamic and structural elements of diseases and populations that are relevant to understand these phenomena. In this work we seek to verify the relevance of inclusion of the topology of the networks described by the relationships between drugs of these models. For this, we propose a framework for simulation of epidemiological models, and perform experimentations using network topologies found in the literature. **Keywords:** Compatmental Epidemiology Models, Graphs, Stochastic Process, SIR

Model, SIS Model, SIRS Model, Network Topology.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

SIS	*Suscetíveis, Infectados, Suscetíveis*
SIR	*Suscetíveis, Infectados, Recuperados*
SIRS	*Suscetíveis, Infectados, Recuperados, Suscetíveis*
SIRD	*Suscetíveis, Infectados, Recuperados, Mortos*
SIRDS	*Suscetíveis, Infectadoss, Recuperados, Morto, Suscetíveis*
PIP	*Python Package Index*
CSV	*Comma Separated Values*
LP	*Lei de Potência*
NLP	*Não-Lei de Potência*

Lista de Símbolos

kB	KiloBytes
MB	MegaBytes
GB	GigaBytes

Capítulo 1

Introdução

A Epidemiologia é a ciência que estuda quantitativamente os padrões que ocorrem em massa na saúde sobre populações definidas. Uma de suas funções é descrever a distribuição e evolução de doenças de forma sistêmica, observando seus o padrões de ocorrência e evolução ao longo do tempo, bem como inferir a efetividade de intervenções nesses cenários. Pesquisadores se valem de uma grande variedade modelos matemáticos, que visam cobrir os elementos dinâmicos e estruturais das doenças e populações, que sejam relevantes para elucidar a compreensão desses fenômenos.

Observamos na literatura, que a esmagadora maioria das abordagens em modelagem epidemiológica utilizam métodos e procedimentos computacionais determinísticos de equações diferenciais. Esta forma de atacar o problema é a mais antiga, consolidada e verificada que existe hoje. No entanto, os modelos construídos desta forma não levam em consideração a topologia das redes de contato, descrita pelas relações entre os indivíduos das populações.

O principal objetivo deste trabalho é demonstrar a relevância da consideração da topologia no modelo epidemiológico. Para isso construímos um software que implementa um framework de simulações de modelos epidemiológicos arbitrários. Desejamos também disponibilizar este framework como software livre, para que possa ser utilizado e aprimorado por qualquer pessoa interessada no tema.

Para validar nosso framework executamos uma bateria de simulações com base em topologias de redes encontradas na literatura.

No capitulo 2 discutiremos numa abordagem introdutória os conceitos que permeiam a epidemiologia e os diferentes tipos de modelos matemáticos conhecidos desta ciência. No capitulo 3 formalizaremos os objetivos e funcionalidades do framework proposto para a modelagem de fenômenos epidemiológicos levando em consideração as redes de contato. No capitulo 4 discutiremos as decisões de projeto e os desafios na implementação do framework. No capitulo 5 observaremos e discutiremos as métricas definidas e as medições obtidas nos experimentos realizados com nosso framweork. Por fim, no capitulo 6 avaliaremos os erros e acertos do trabalho, e levantaremos as melhorias possíveis de serem implementadas e as próximos passos para avançar este estudo.

Capítulo 2

Conceitos

2.1 Epidemiologia

A Epidemiologia é a ciência que estuda quantitativamente os padrões que ocorrem em massa na saúde sobre populações definidas, como sociedades, grupos, faixas etárias; analisa causas, fatores condicionantes e determinantes, e efeitos de doenças e suas formas de transmissão numa abordagem oposta à prática clínica, que observa e intervem em quadros de doença a nível de indivíduos. [Rocha, 2012]

Uma das funções da epidemiologia é descrever a distribuição e evolução da doença de forma sistêmica, observando a correlação da aquisição ou não da doença (Infecção) com fatores genéticos, organizações e práticas sociais, exposição a agentes contaminantes, seu ciclo de vida, definição de condições de risco entre outras características. [Wikipédia-PT-BR-Epidemiologia, 2017] A epidemiologia provê, portanto, ferramentas para a modelagem matemática e análise estatística de doenças, bem como a estimativa e medição da efetividade de intervenções sobre populações, sendo assim uma importante base de conhecimento para a discussão e implementação de programas e políticas de saúde pública, além da melhoria de práticas clínicas. [Wikipédia-EN-Epidemiology, 2017]

2.2 Modelos de Epidemiologia

A modelagem matemática e computacional de fenômenos epidemiológicos é uma prática de grande relevância na área de epidemiologia, pois possibilita uma melhor compreensão do desenvolvimento de doenças na população, e um planejamento de medidas de controle e erradicação mais embasado, além de uma estrutura para análise de impacto de tais medidas, provendo acima de tudo métricas.

O processo de modelagem consiste da análise de um contexto de epidemiologia específico, o levantamento de fatos sobre o comportamento típico conhecido da doença, a construção de considerações e hipóteses sobre elementos físicos, fisiológicos, biológicos e da estrutura e dinâmicas inatas da população afetada, com a a finalidade de mapear tais características para o modelo. A intenção é, como sempre em modelagem matemática de fenômenos reais, que o modelo seja uma abstração que provê uma aproximação da realidade, representando somente os fatores relevantes do cenário real.

A seguir veremos algumas categorias e premissas de modelos que são aplicados no contexto de epidemiologia.

2.2.1 Modelos Compartimentais

Em epidemiologia considera-se que um indivíduo, pertencente a uma população, está em um dado momento em um certo estágio do ciclo de vida do contágio e desenvolvimento da doença em questão no estudo, como por exemplo, suscetível ao contágio, imune, contaminado, ou recuperado. Divide-se a partir daí população em compartimentos, cada um sendo um subconjunto do todo contendo os indivíduos que se encontram naquele estágio. Esses compartimentos formam uma partição matemática do conjunto população, ou seja, todos os indivíduos pertencem a um e somente um dos compartimentos. Os compartimentos, ou classes, são comumente rotulados por letras maiúsculas referentes ao estágio do contágio em que aquela parcela da população se encontra no momento atual. Diz-se "a classe S"para tratar dos suscetíveis, por exemplo.

Para cada contexto de estudo da doença é interessante categorizar a população em diferentes conjunções de compartimentos. Tradicionalmente encontra-se na literatura as classes M, S, E, I, R, e suas combinações utilizadas para diferentes classes epidemiológicas.

A classe M refere-se aos indivíduos com imunidade passiva temporária, devida ao compartilhamento de anticorpos, como aqueles transmitidos pela mãe ao recém-nascido através da placenta. Após o desaparecimento destes anticorpos, o indivíduo passa a integrar a classe dos suscetíveis. Crianças que nascem sem a presença de anticorpos automaticamente pertencem à classe S. A classe S inclui todos os indivíduos que podem contrair a infecção. Um contato adequado entre um indivíduo suscetível e um infectado faz com que o primeiro passe para a classe E. A classe E, dos expostos, contém aqueles que estão em um estágio inicial, ou fase de latência. Ou seja, o indivíduo já foi infectado pela doença, porém ainda não é capaz de transmiti-la a outros. Passado o tempo de latência ou incubação estes indivíduos tornam-se membros da classe I. Os membros da classe I são capazes de transmitir a infecção e permanecem nesse compartimento até que a doença seja curada, seja de forma natural, passado um determinado tempo, ou através de alguma intervenção, o que varia de acordo com a doença e cenário estudados. É possível que o indivíduo permaneça nessa classe por tempo indeterminado, ou até mesmo permanentemente. Quando o período de infeccioso termina entra na classe R. A classe R inclui todos aqueles que se recuperaram da infecção e, por consequência, tornam-se imunes a novos contágios da mesma permanecendo assim indefinidamente.

A combinação dessas e outras possíveis classes determinam o tipo do modelo compartimental. Um modelo SI, por exemplo, não contempla a recuperação nem a latência da doença. Num modelo SIS, os indivíduos infectados se recuperam mas não adquirem imunidade. Num SIR, o indivíduo se recupera e torna-se imune. E assim por diante.

2.2.1.1 Modelos Compartimentais Elementares

Os modelos compartimentais mais amplamente discutidos e implementados na literatura de modelos de epidemiologia são os que propõem combinações dos três estágios, 'suscetível', 'infectado' e 'recuperado', devido à natureza do ciclo de vida da maior parte doenças transmissíveis, à simplicidade de análise desses modelos e à objetividade em se verificar que um indivíduo encontra-se em um destes estágios. Daí deriva-se os modelos mais comuns [Easley and Kleinberg, 2010]:

SIS (suscetíveis – infectados - suscetíveis). Este tipo de modelo é apropriado para várias doenças causadas por agentes bacterianos, e por vezes virais, descartando-se a modelagem da mutação e evolução do vírus por completo. Nestes cenários a recuperação não protege contra uma reinfecção. Um caso particular deste modelo acontece em ocasiões em que um indivíduo infectado nunca se recupera da doença, configurando um modelo SI.

SIR (suscetíveis – infectados - recuperado) Esse tipo de modelo, relaciona-se a doenças em que os indivíduos infecciosos podem recuperar-se da doença adquirindo imunidade permanente, como rubéola, varicela, sarampo e caxumba.

SIRS (suscetíveis - infectados - recuperados - suscetíveis) É como o anterior, porém compreendendo a possibilidade de uma imunidade parcial ou temporária após a infecção. O vírus da gripe é talvez o exemplo mais familiar. A gripe adquirida num inverno confere imunidade apenas parcial contra a gripe do inverno seguinte.

2.2.2 Modelos Baseados em Equações

Os primeiros modelos matemáticos de epidemiologia, propostos no começo do século 20 eram muito simples, mas já capazes de descrever as dinâmicas de doenças de forma aproximada, com uma representação gráfica da abundância dos indivíduos pertencentes a cada um dos compartimentos e sua variação ao longo do tempo. [Galante, 2008] Eles partiam das seguintes considerações:

Numa grande população, a variação do efetivo de cada um dos compartimentos ocorre a todo instante. Matematicamente, a variação de uma variável Y em função de outra X é medida pela derivada de Y em X. Se, por exemplo, o número de infecciosos (I) na população for representado em função do tempo, $I = f(t)$, então a variação de I à medida que t varia é medida pela derivada de I em t, $\frac{\partial I}{\partial t}$. E o mesmo se poderia dizer da abundância dos indivíduos em qualquer categoria da população. Desta forma, os modelos epidemiológicos podem ser representados matematicamente por sistemas de equações diferenciais.

O procedimento geral consiste em construir as equações apropriadas para cada tipo de modelo epidemiológico, justificando em detalhe todos os termos das mesmas. Em seguida, apresenta-se a solução das equações e a ênfase é colocada nas consequências epidemiológicas dessas soluções.[Hethcote, 2000]

2.2.3 Autômatos Celulares

Os Autômatos Celulares foram introduzidos nos anos 50 pelo matemático John von Neumann numa tentativa de modelar processos naturais de auto-reprodução. Os Autômatos Celulares consistem de simulações discretas no tempo, espaço e no estado do sistema. A ideia básica destes modelos consiste em considerar cada posição do domínio espacial como sendo uma célula, à qual é atribuído um estado. O estado de cada célula é modificado de acordo com o seu estado e o de suas vizinhas na etapa de tempo anterior, através de uma série de regras simples que tentam imitar as leis físicas ou biológicas que regem o sistema. [Galante, 2008]

Nesta abordagem, as variáveis de estado do sistema, assim como o tempo, são discretos. O sistema é representado espacialmente através de um reticulado de células que interagem obedecendo a algumas regras de mudança de estado. A dinâmica do sistema como um todo depende desta interação local entre as células. Cada célula representa um indivíduo, que pode estar em um dos compartimentos do modelo.

A chance de um indivíduo suscetível tornar-se infectado vai depender do número de contatos que ele estabelece com outros indivíduos no intervalo de tempo adotado e também da probabilidade de que cada contato resulte em transmissão. A chance com que cada indivíduo se recupere da doença é também levada em conta.

2.2.4 Modelos Baseados em Agentes

Modelos baseados em agentes podem ser utilizados para simular fenômenos reais através da construção de agentes computacionais individuais com propriedades específicas e simular interações entre eles. Um agente apresenta um comportamento que é consequência de suas percepções sobre o ambiente e de suas interações com outros agentes. Das interações entre muitos agentes podem emergir fenômenos globais, incluindo comportamentos coletivos. Assim, empregam-se agentes computacionais para compreender o impacto dos comportamentos individuais sobre os fenômenos coletivos nos casos em que não é possível compreendê-los de maneira dedutiva ou analítica.

Apesar de os modelos baseados em agentes não serem novidade, sua utilização na modelagem de sistemas complexos e simulações é recente. Nesses casos, a modelagem de um ambiente macro é feita definindo os níveis micro das relações entre os agentes com propriedades individuais específicas. Com o uso de agente computacionais pode-se buscar compreender as estruturas complexas das interações sociais.

2.3 Redes Complexas

Muitos modelos epidemiológicos assumem que os contatos entre os indivíduos das diversas classes são homogêneos, sem levar em consideração aspectos espaciais e topológicos e que os contatos são feitos através de interações individuais. Dessa forma, estes modelos podem obter resultados que não condizem tão fidedignamente com a realidade.

Uma das formas utilizadas na literatura para a modelagem dos contatos é o uso de redes complexas, que chamamos **Redes de Contato**. Tais redes podem ser representadas em grafos onde os vértices são as pessoas e as arestas representam os possíveis contatos pelos quais as doenças se propagam. A utilização de redes complexas na dinâmica de epidemias tem sido estudado recentemente de maneira geral e em tipos específicos de redes, tal como as redes de mundo pequeno e redes de escala livre, descritas sobre categorias conhecidas de grafos no estudo da Teoria de Grafos.

A seguir veremos alguns exemplos emblemáticos dessas redes complexas.

2.3.1 Reticulados

Uma rede reticulada é descrita sobre um grafo reticulado, ou de grade. Nesta rede todos os indivíduos estão posicionados numa grade regular de pontos, normalmente tridimensional, e indivíduos adjacentes estão conectados. As dinâmicas descritas sobre esse tipo de grafos são equivalentes àquelas dos autômatos celulares.

2.3.2 Redes Espaciais

Redes espaciais são a forma mais flexível de redes. Indivíduos são posicionados em uma área, ou espaço, e a conexão ou contato entre eles é modelada a partir da distância euclidiana entre os pontos. Com essa definição é possível modelar redes que se equivalem a praticamente todos os outros modelos de redes.

2.3.3 Redes de Mundo Pequeno

O conceito de redes de mundo pequeno foi introduzido no contexto de redes sociais pelo sociólogo Stanley Milgram e baseia-se na observação de que existe na natureza uma variedade de sistemas cujos elementos estão relativamente próximos uns dos outros, apesar do tamanho significativamente grande destes sistemas.

O objetivo do modelo de redes de Mundo Pequeno é descrever apropriadamente a pequena distância entre os elementos e ao mesmo tempo obter um alto índice de aglomeração.

Estas duas características, conhecidas como efeito Mundo Pequeno, são observadas em vários sistemas sociais e podem ser interpretadas como resultado dos padrões de interação entre os indivíduos destes sistemas, independentemente das limitações geográficas. Em termo epidemiológicos, redes de pequeno mundo implicam que o nível de infecciosidade requerido para a doença depende da alta sensibilidade da conexão topológica da população. A alta clusterização nestas redes implica que a doença propaga-se localmente de maneira muito rápida, enquanto que o curto caminho da rede permite que a doença atinja grandes distâncias no grafo.

2.3.4 Redes de Escala Livre

O modelo para redes de escala livre mais utilizado atualmente é o modelo de Barabási-Albert. Neste modelo as redes apresentam uma ordem na dinâmica de estruturação, com características bem específicas. Uma das características principais, denominada conexão preferencial, é a tendência de, ao construir uma nova conexão, um novo vértice se conectar a um vértice da rede que tem um grau elevado de conexões. Essa característica implica em redes com poucos vértices altamente conectados, denominados hubs, e muito vértices com poucas conexões. Desta maneira a distribuição dos graus dos nodos do grafo respeita uma Lei de Potência. [Keeling and Eames, 2017]

2.3.4.1 Lei de Potência

E estatística, uma lei de potência é uma relação de proporção direta entre duas grandezas, onde a variação de uma delas resulta na variação exponencial da outra. Por exemplo a variação da área do quadrado com relação ao comprimento de seu lado, onde se o lado é dobrado, a área cresce a um fator de quatro.

O interesse científico nas relações de leis de potência deriva, em parte, da facilidade com que certas classes gerais de mecanismos os geram. A demonstração de uma relação de lei de potência em grandezas pode ser um indicativo de tipos específicos de mecanismos subjacentes ao fenômeno natural em questão, e podem indicar uma conexão profunda com outros sistemas aparentemente não relacionados. A ubiquidade das relações de poder-lei na física é em parte devido a restrições dimensionais, enquanto que em sistemas complexos, as leis de poder são muitas vezes compreendidas como evidência de hierarquia ou de processos estocásticos específicos. Alguns exemplos notáveis de leis de potência são a lei de distribuição de renda de Pareto, auto-similaridade estrutural de fractais e leis de escala em sistemas biológicos. A pesquisa sobre as origens das relações de poder e os esforços para observá-las e validá-las no mundo real é um tema ativo de pesquisa em diversos campos da ciência, incluindo física, informática, linguística, geofísica, neurociência, sociologia, economia entre outras. [Wikipedia-EN-Power-Law, 2017]

2.4 Outras Características de Modelos e Conclusão

Como pudemos observado neste capitulo, são diversas as formas possíveis de molagem de fenômenos epidemiológicos levando em conta os mais variados aspectos dinâmicos e estruturais das populações atingidas, comportamento e ciclo de vida da doença em questão. Fica a cargo, então, do pesquisador levantar os atributos relevantes para o estudo em questão, de acordo com o cenário observado.

Outras características de cenários reais, não mencionadas ao longo deste capitulo, mas que podem ser consideradas, e incluídas na elaboração de modelos epidemiológicos são nascimento, morte, mutação, topologia dinâmica, genética, fatores ambientais, e quais mais relações puderem ser observadas no fenômeno epidemiológico que possa ter relevância para o estudo. A inclusão ou não destes fatores no modelo, são determinantes para a definição de certas características do mesmo, tais qual: modelagem do tempo como uma grandeza discreta ou contínua, geolocalização de componentes da rede de contato, pesos das relações, relacionamentos fuzzy, em que parte do cálculo do modelo são consideradas certas variáveis, como a probabilidade de contágio, de morte, etc.

A seguir elaboraremos uma definição de modelo com o intuito de verificar a relevância da consideração da topologia nos modelos epidemiológicos.

Capítulo 3

Proposta

3.1 Motivação

Durante o processo de revisão bibliográfica, procurando por referências de implementação de simuladores de modelos de epidemiologia, um resultado recorrente foi notado: não existem muitos softwares ou mesmo implementações simples de simulações disponíveis que usem grafos como base para o modelo, ou que se preocupem com a topologia da rede de qualquer forma. A esmagadora maioria dos métodos e procedimentos computacionais disponíveis na literatura hoje utilizam francamente a abordagem determinística das equações diferenciais, muitas vezes, inclusive, assumindo essa como se fosse a única possível. [Lloyd and Valeika, 2007] É claro que essa forma de entender o atacar o problema é já muito bem consolidada e extensivamente aplicada em cenários reais com resultados satisfatórios. No entanto, a favor da experimentação, e buscando criar aplicações para grafos, proporemos aqui mais uma opção.

Ao implementar pequenas rotinas computacionais para visualizar o comportamento de redes de contato, surgiu o interesse em experimentar com os modelos descritos sobre redes complexas, mais especificamente suas implementações que se relacionassem com grafos reais para, quem sabe, publicar um framework, em software livre, de simulação e análise de modelos epidemiológicos com enfoque na topologia. Para isso, seria necessário implementar um simulador capaz de operacionalizar e parametrizar modelos epidemiológicos em redes de contato, obter dados relevantes de redes sociais que pudesse servir de exemplo para as simulações, de preferência com tipos conhecidos de grafos, executá-las, gerar e consolidar métricas passíveis de análise.

3.2 Objetivos

O principal objetivo deste trabalho é demonstrar a relevância da consideração da topologia no modelo epidemiológico. Para isso construiremos um software que implementa um framework para simulações de modelos epidemiloógicos arbitrários.

Considerando as possibilidades de real utilização deste framework, espera-se que sua utilização seja fácil e prática, que seja viável a customização de parâmetros e funções de coleta e consolidação dos dados gerados pelas simulações, de forma a permitir que este software seja usado por pesquisadores de quaisquer áreas que tenham interesse em modelos epidemiológicos.

Nessa mesma direção, seria ideal que o software possa ser executado em um computador pessoal, sem que haja grandes limitações para seu uso que poderiam vir a ser empecilhos para a aplicação real. Em outras palavras, espera-se que esse programa possa ser amplamente utilizado

para francamente apoiar a pesquisa em epidemiologia, bem como suscitar o interesse do público geral nos modelos epidemiológicos.

No que se refere às métricas resultantes das simulações, espera-se que os dados obtidos forneçam material substancial para a consolidação de informações relevantes, testes de hipóteses e análise de cenários reais de aplicação.

3.3 O Framework

Para atingir os objetivos mencionados na seção anterior, foi elaborada a seguinte meta de um conjunto simples de ferramentas contendo:

- um software simulador que:
  - implemente pelo menos os modelos elementares:
    - SIS
    - SIR
    - SIRS
  - tenha a capacidade de usar como base para a simulação topologias estáticas arbitrárias, onde um indivíduo é representado por um nodo da rede, e a relação binária de contato entre indivíduos representada por uma aresta entre seus respectivos nodos
  - modele o tempo como uma grandeza discreta para que seja possível calcular e obter medições sobre a evolução do estado da rede iterativamente
  - tenha como parâmetros globais da simulação:
    - **p**: a probabilidade de contágio de um indivíduo suscetível por um infectado, dado que haja contato entre eles
    - **i**: a duração, em rodadas da simulação, do período de infecção, ou seja, o número de rodadas em que um indivíduo permanece no estágio infectado, de onde ele passa para a classe dos suscetíveis ou recuperados, dependendo do modelo instanciado
  - tenha a capacidade de gerar medições arbitrárias a cada rodada e ao termino da simulação, registrando-as em um formato fácil de se analisar por ferramentas computacionais genéricas
- um pacote de rotinas para preparar os dados topológicos obtidos de fontes arbitrárias para a simulação, bem como gerar volumes de dados de cenários de simulação sobre a topologia dada e posterior coleta, agrupamento e consolidação dos dados gerados pelas simulações em métricas definidas
- fazer tudo isso de forma que torne viável a posterior publicação deste software como um framework de epidemiologia computacional sobre grafos

3.4 Conclusão

No próximo capitulo, veremos como o framework proposto foi implementado, discorreremos sobre as decisões de projeto tomadas ao longo de sua implementação e as dificuldades e empencílhos encontrados no caminho.

Capítulo 4

Implementação

4.1 Tecnologias e Ferramentas

4.1.1 Linguagem de Programação

Para a implementação do simulador, bem como as ferramentas auxiliares optei de pelo uso de Python3, a versão mais atual do Python. Essa escolha foi tomada devido a esta ser uma linguagem de código aberto, multiplataforma, bem consolidada, com vasta base de usuários e comunidade ativa, uma das que mais tem crescido nos últimos anos, contribuindo para a o desenvolvimento de uma grande, e em constante crescimento, base de bibliotecas, frameworks e ferramentas para os mais variados contextos e finalidades. Notadamente, Numpy e SciPy são dois pacotes de, respectivamente, computação numérica e científica, dos mais usados no mundo, base para boa parte dos trabalhos no grande avanço recente na indústria de inteligência artificial, aprendizado de máquina e suas aplicações comerciais. Além da comunidade e da poderosa base compartilhada de código, Python também provê uma vasta base de conhecimento através das documentações de suas ferramentas, das PEPs [Barry Warsaw and Coghlan, 2017] que são as formalizações de suas filosofias, normas, diretrizes de arquitetura, de ferramentas e estilo de código, sendo assim um ecossistema completo de desenvolvimento de software, focado na simplicidade de código, arquitetura e documentação, agilidade no desenvolvimento e fácil manutenibilidade. Além disso, Python3 é a linguagem com a qual eu mais tenho experiência e proficiência de trabalho.

4.1.2 Bibliotecas

Para a representação e manipulação da rede de contato em memória, como a princípio não era certo exatamente que tipo de manipulações de dados seriam necessárias, optei pelo uso da biblioteca Networkx que já implementa uma abstração de grafos, a classe Graph, e todos os algoritmos clássicos em teoria dos grafos, como buscas em largura e profundidade, caminho mínimo, e vários algoritmos de diferentes aplicações de grafos. [Netwokx, 2017] Outro forte ponto a favor do uso desta biblioteca é o fato de ela já estar integrada com o ambiente de distribuição padrão da linguagem **(PIP)**, não requerendo download de fontes de terceiros, configuração e compilação. A NetworkX conta também com documentação padronizada segundo as diretrizes da comunidade Python e implementa interfaces condizentes com as filosofias da linguagem. Como alta performance não é um objetivo do projeto, pelo menos a princípio, já que a execução do trabalho se deu de forma bastante exploratória, esta escolha foi feita apesar de ter encontrado menções a outras bibliotecas que possuem benchmarks de eficiência de computação

superiores, porém apresentaram empecilhos na instalação e configuração por não se adequarem às normas de organização, distribuição e documentação Python, demandam longos períodos de compilação e apresentam alguns bugs.

Para o desenho dos grafos, que foi parte importante das primeiras etapas do trabalho de implementação, para rápida validação visual do comportamento da simulação de acordo com o esperado, utilizei o básico e já bem conhecido pacote Matplotlib [MatplotLib, 2017], com a qual a biblioteca Networkx possui também uma integração parcial.

Foi também levantada a possibilidade de usar a plataforma de visualização de grafos Gephi [Gephi, 2017], que consiste em uma aplicação desktop Java que provê um ambiente com extenso ferramental para a organização espacial e representação gráfica de redes estáticas e dinâmicas, características de nodos e vértices, e evolução ao longo do tempo. No entanto, a tentativa de integração com a ferramenta demandou um período longo de estudo de suas interfaces e formatos de descrição de dados, o que fez com que o seu uso fosse descartado, uma vez que foi constatado que a visualização de grafos é um problema muito grade em si, e que não faz parte da proposta principal do trabalho, consistindo apenas em uma ferramenta auxiliar.

4.2 Estrutura de Dados e Lógica da Simulação

Para criação do simulador, foi implementada uma extensão, ou especialização, da classe networkx.Graph. A classe original já resolvia o carregamento de um grafo a partir de estruturas de listas de adjacências. Internamente, ela armazena informações sobre o grafo em uma única cópia central de sua topologia, em formato de matriz, provendo rotinas, em forma de propriedades e métodos de classe, que calculam dinamicamente valores de atributos do grafo, de seus nodos, e vértices, e formatos alternativos de representação a partir desta estrutura central.

4.2.1 Vantagens e Desvantagens das Estruturas de Dados Utilizadas

A arquitetura da classe, tal como acabamos de ver, permite um alto grau de modularização dessas rotinas, com cada uma tendo a possibilidade de realizar chamadas às demais, calculando dinamicamente apenas o necessário para o uso em contextos específicos, já que a biblioteca se propõe a ser de uso geral. Esta abordagem tem como principais vantagens a qualidade de concisão dos códigos das rotinas de classe, já que implementam lógicas pontuais de cálculo de atributos parciais e/ou se valem de outras rotinas que realizam esses cálculos parciais e a facilidade na compreensão, manutenibilidade, e personalização da classe. Essas características se mostraram de grande valor durante a implementação das primeiras versões da classe estendida, provendo agilidade na codificação das lógicas do modelo. No entanto, toda decisão de projeto traz uma troca implícita associada. Favorece-se alguns aspectos sempre em detrimento de outros. Então, em contrapartida aos benefícios supracitados, a mais evidente desvantagem dessa arquitetura de classe é o alto custo computacional para a realização de qualquer operação, já que muitas chamadas de método são frequentemente encadeadas, por vezes tendo um mesmo trecho de código sendo executado várias vezes durante o cálculo de um atributo simples da classe. Este problema é agravado devido à arquitetura da linguagem. As documentações do Python sempre deixam claro que chamadas de métodos/funções são relativamente caras. [Python-Performance-Tips, 2017] Outro ponto negativo notado desta abordagem é a grande interdependência gerada entre as rotinas, que cresce como que exponencialmente ao passo que a complexidade da das dinâmicas da classe vai aumentando. Este entrelaçamento de chamadas de métodos dificulta a posterior alteração dos códigos, bem como a depuração de erros decorrentes destas modificações. Estas

duas desvantagens combinadas acabaram por se mostrar muito relevantes em estágios avançados do desenvolvimento, como veremos mais adiante na seção 4.6.

4.2.2 A Classe *Network*

Para a criação do principal componente do simulador, a classe Network, que estende a networkx.Graph, ou seja, para a especialização do grafo genérico em uma rede de epidemiologia compartimental, foram implementadas as seguintes funcionalidades:

- leitura e carregamento de topologias de redes a partir dos formatos de arquivos disponíveis na literatura como veremos na seção 4.3. Estes métodos, na prática, interpretam estes formatos como listas de adjacências;
- atribuição de estágios aos nodos da rede, a partir da interface disponibilizada pela classe original para a criação e edição de atributos arbitrários para cada nodo e vértice do grafo;
- o carregamento de um estado da rede, ou seja, um mapeamento de cada nó em um dos estágios implementados no modelo. Esta rotina lê de um arquivo **CSV** uma lista de pares nodo-estágio;
- um método que calcula dinamicamente, a partir dos dados dos nodos, o conjunto de indivíduos pertencentes a cada compartimento do modelo;
- um método de configuração geral do modelo, recebendo e armazenando na classe as variáveis globais **p** e **i**, além do modelo a ser simulado, dentre os propostos, **SIS**, **SIR** e **SIRS**;
- uma rotina de espalhamento de contágio pela rede, como um método que, para cada indivíduo da partição dos infectados, levanta sua vizinhança, e para cada vizinho gera um número aleatório que determina, de acordo com a variável global p, se este indivíduo irá torna-se ou não infectado;
- recuperação baseada em turnos dos indivíduos infectados. Toda vez que um indivíduo se torna infectado, ele recebe um atributo de contagem regressiva da duração da infecção. A cada rodada, todos os indivíduos infectados tem esse atributo decrementado em um. Quando o valor desse atributo se torna igual a zero, o indivíduo passa para o estágio seguinte, que será recuperado, ou suscetível, dependendo do modelo instanciado;
- rotina de execução da simulação, passando iterativamente pelas etapas de espalhamento e recuperação dos indivíduos até a que a condição de parada seja atingida, ou seja, até que haja a extinção da doença na rede. Esta condição é verificada observando o tamanho da partição dos infectados ao termino de cada rodada.
- mecanismos de registro de métricas da simulação. A classe permite que funções arbitrárias sejam configuradas como coletores de medições e agregadores de estatísticas ao longo da simulação. Três pontos de verificação foram definidos na execução da simulação: no início, assim que o método de simulação é chamado, ao fim de cada rodada e no fim da simulação. Um atributo global da classe é disponibilizado para que essas funções armazenem dados e cada uma deve implementar as verificações na estrutura da rede e agregações de dados que se julgarem adequadas. Caso essas funções não forem configuradas, a classe possui seus métodos padrão para a coleta de medidas, conforme discutido posteriormente na seção 5.

4.2.3 Funcionalidades Complementares

Outras funcionalidades acabaram sendo implementadas como auxiliares ao desenvolvimento do trabalho:

- funcionalidade para a determinação de um posicionamento dos nodos em um plano geométrico. O problema de desenho grafos, ou seja, representá-los de forma visual numa disposição que faça algum sentido para a sua análise, não tem uma solução exata. Este problema suscita uma área de estudo por si só [Terra, 2009]. Para ajudar a solucionar esta questão em diferentes contextos, a biblioteca NetworkX fornece a implementação de alguns algoritmos clássicos, métodos heurísticos, de organização espacial de grafos. Na classe Network, foi implementado um método que aceita qualquer função de escolha dessa organização (layout), para ser aplicada na rede, podendo ser uma das disponíveis na biblioteca, ou uma customizada. Esta funcionalidade também permite que várias execuções do método sejam executadas, até quem uma configuração satisfatória seja atingida;

- método que desenha a rede na tela, se um layout tiver sido atribuído para a rede. Para isto, plotamos o grafo segundo seu layout como um gráfico da biblioteca MatplotLib;

- uma versão da simulação que desenha o estado atual do grafo na tela, atualizando a cada rodada. Esse método foi vital para a validação do comportamento dos modelos. No entanto, os grafos reais usados na validação do framework têm um tamanho considerável. Para as redes geradas sobre esses grafos, esta funcionalidade não tem muita relevância, além de ser bastante ineficiente;

- outros métodos auxiliares menores para o cálculo de atributos relevantes à simulação, como contagem da população de um determinado compartimento, cópia da referência dos indivíduos de um compartimento, e outros. Estes métodos foram elaborados para serem bem pontuais, e manter os segmentos lógicos de código altamente modularizados, seguindo a lógica de organização de métodos da biblioteca Networkx.

4.3 Bases de Dados

Para observarmos o comportamento da simulação no contexto de um caso real de modelagem epidemiológica e obtermos métricas relevantes para essa finalidade de estudo, precisaríamos realizar o levantamento de uma rede real de contato, o que foge do escopo deste trabalho. A segunda opção seria usar os dados de um rede real utilizada numa pesquisa em epidemiologia. O site `www.networkrepository.com` provê uma base de dados de redes reais usadas em pesquisas acadêmicas de todas as áreas do conhecimento. Os dados dessas redes são doados ao repositório pelos pesquisadores que os levantaram. No site é possível obter os dados topológicos das redes em arquivos em formatos de matiz ou listas de vértices, bem como estatísticas básicas dos grafos, como grau máximo, mínimo e médio dos vértices, coeficientes de agrupamento, além de amostras de visualização. [Network-Repository, 2017]

Uma única rede de epidemiologia foi encontrada no repositório, a rede *mc2depi* [Rossi and Ahmed, 2015]. Suas dimensões, porém, foram um impeditivo para usa-la na validação do framework, devido a problemas de desempenho encontrados e que iremos abordar mais adiante na seção 4.5. Apesar disso, outras muitas redes de menores proporções, sociais, ou que apresentam alguma similaridade com redes sociais, como redes de computadores e redes de colaboração em pesquisa, encontram-se disponíveis no repositório. Estas provêm material

satisfatório para, ao menos, termoas alguma noção de como os modelos implementados pela classe Network se comportam em redes reais.

Experimentalmente verificamos que a proporção entre o tamanho do arquivo contendo a topologia da rede e o objeto da classe Network correspondente, carregado em memória é de aproximadamente $1 : 2,46$, uma vez que algumas redundâncias de dados foram implementadas, por exemplo as listas de referências de indivíduos por população, além dos atributos dos nodos. O espaço em memória ocupado por estes dados é linear linearmente proporcional ao número de nodos na rede.

Podemos calcular então, segundo as dimensões da rede *mc2depi*, que uma instância do modelo baseado nela requer cerca de 91 MB de memória. Um volume perfeitamente razoável para os padrões dos computadores pessoais contemporâneos. Contudo, um problema na simulação de sobre uma rede deste tamanho surge posteriormente, no tempo de execução como abordado na seção 4.5.

Uma vez observada a dificuldade em executar uma simulação completa sobre uma rede destas proporções, selecionamos dentre redes previamente classificadas como redes de Lei de Potência **(LP)** e não-Lei de Potência **(NLP)** [de Oliveira Cabral Filho, 2016] duas de cada uma dessas categorias que tivessem as menores dimensões.

Na figura 4.1 temos alguns dados sobre as quatro redes que foram selecionadas para a validação do framework, *p2p-Gnutella08* **(NLP)**, *p2p-Gnutella09* **(NLP)**, *ca-GrQc* **(LP)** e *ca-HepTh* **(LP)**, além de informações sobre a rede *mc2depi*.

	Nodos	Arestas	G min	G max	G med	Dens	Simulado
Completo1	7383	27250653	7382	7382	7382	1	Amostragem pequena
Completo2	170	14365	169	169	169	1	sim
Completo3	228	25878	227	227	227	1	sim
p2p-Gnutella08	6301	20777	1	97	6	0.0010468	sim
p2p-Gnutella09	8114	26013	1	102	6	0.000790322	sim
ca-GrQc	5242	14495	0	81	5	0.0010544	sim
ca-HepTh	9877	25998	0	65	5	0.000532532	sim
mc2depi	525825	~1600000	2	6	5	0.0000113884	não
ArvoreEstrela	2729409	2729408	1	2729408	~2	0.000000732759	não
ArvoreBinaria	2729409	2729408	1	3	~2	0.000000732759	não
ArvoreLista	2729409	2729408	1	2	~2	0.000000732759	não

Figura 4.1: Dados das redes inicialmente consideradas para avaliação. As redes arbitrárias descritas com grafos completos, árvores e redes reais, respectivamente em vermelho, verde e amarelo.

Adicionalmente definimos as escalas de algumas redes arbitrárias, sobre categorias notáveis de grafos, com o intuito de verificarmos um contraste entre as medidas coletadas das simulações sobre esses diferentes tipos de topologias. Separamos estas em duas categorias opostas.

A primeira classe de redes arbitrárias é definida por grafos completos, onde todos os nodos possuem uma relação com todos os demais. Esta topologia faz com que nosso modelo estocástico se aproxime do modelo determinístico baseado em equações, ou seja, não leva fortemente em consideração a topologia real da rede de contato, uma vez que um grafo completo não contém nenhuma complexidade topológica de rede. Nesta categoria definimos três redes, a *Completo1* feito para ter um número de nodos igual à média do número de nodos das quatro redes reais selecionadas. Como esta rede também esbarrou nos problemas de eficiência da simulação, definimos outras duas de menores proporções, a *Completo2* e *Completo3* criadas para

ter aproximadamente o número de arestas igual a média do número de arestas das quatro redes reais e o maior número de arestas entre elas, respectivamente.

Definimos a segunda classe de redes arbitrárias com o extremo oposto dos grafos completos, em termos de densidade, os grafos minimais conexos, mais conhecidos como árvores. Três configurações notáveis de árvores, com organizações topológicas amplamente diversas, foram consideradas, a Arvore Estrela, a Arvore Binária Balanceada e a Lista. Para cada uma destas definimos uma rede com as dimensões definidas para manter aproximadamente a mesma relação entre a densidade dos grafos completos e a média das densidade dos grafos reais, de 1168,20.

As medidas de todas as redes descritas até então podem ser observadas na figura 4.1. Visto o tamanho dos conjuntos de nodos e arestas das árvores, a regra de composição mantendo a relação de densidade entre as três categorias teve de ser abandonada, para que houvesse tempo hábil para a execução de simulações sobre as árvores. Um novo conjunto de árvores foi criado então, tendo como base o menor número de nodos dentre as redes reais. Suas dimensões podem ser observadas na figura 4.2.

	Nodos	Arestas	G min	G max	G med	Dens	Simulado
Completo1	7383	27250653	7382	7382	7382	1	Amostragem reduzida
Completo2	170	14365	169	169	169	1	sim
Completo3	228	25878	227	227	227	1	sim
p2p-Gnutella08	6301	20777	1	97	6	0.0010468	sim
p2p-Gnutella09	8114	26013	1	102	6	0.000790322	sim
ca-GrQc	5242	14495	0	81	5	0.0010544	sim
ca-HepTh	9877	25998	0	65	5	0.000532532	sim
mc2depi	525825	~1600000	2	6	5	0.0000113884	não
ArvoreEstrela	5242	5241	1	5241	~2	0.000381533765	sim
ArvoreBinaria	5242	5241	1	3	~2	0.000381533765	sim
ArvoreLista	5242	5241	1	2	~2	0.000381533765	sim

Figura 4.2: Dados das redes de fato utilizadas. As redes arbitrárias descritas com grafos completos, árvores e redes reais, respectivamente em vermelho, verde e amarelo.

4.4 Instanciação do Framework

Para realizar uma execução de simulação de um modelo, além da uma topologia de rede, é necessário que definamos também um estado inicial, o que consiste de um mapeamento dos nodos da rede nos estágios existentes no modelo. E por fim precisamos definir as variáveis **p** e **i**.

4.4.1 Estados Iniciais

Para que seja possível observar o comportamento de um modelo numa topologia específica, definir um único estado inicial arbitrário não é o suficiente. A execução pode gerar medições mais relacionadas com o estado inicial específico do que com a semântica da rede de forma geral. Por exemplo, é razoável esperar que o espalhamento da epidemia na rede se comporte de maneiras radicalmente diferentes nas seguintes circunstâncias hipotéticas:

1. No estado inicial, a partição dos infectados é composta por somente um nodo isolado, de grau baixo;

2. No estado inicial, 10% dos nodos, os de maior grau, integram o estágio dos infectados.

Tendo esta condição em mente optamos por criar os estados iniciais para nossas simulações de forma aleatória, segundo uma dada configuração de proporções. Podemos expressar uma configuração de proporção com uma tupla composta de n elementos no formato (C_i, p_i), onde todo $C_i \in C$, sendo C o conjunto de compartimentos do modelo, $n = |C|$, e para cada compartimento C_i, p_i é a proporção dos nodos da rede pertencentes a ele. Por exemplo, a configuração definida pela seguinte tupla $((S, 0, 9), (I, 0, 1), (R, 0))$. Nela o estado inicial é sorteado de forma tal que 10% dos nodos estejam nos estágio infectado, 90% sejam suscetíveis e 0% recuperados. Note que esta tupla é constitui uma configuração válida para um pacote estados iniciais para quaisquer modelos compostos exclusivamente dos compartimentos **S**, **I** e **R**, ou seja, tanto um modelo **SIR** quanto um **SIRS**. Além disso para evitar distorções nas medidas da simulação devidas à variância dessas distribuições, é desejável que haja uma amostragem contendo uma certa quantidade a de instanciações de estados iniciais da rede segundo essas proporções. Desta forma, adicionando o nome do modelo, para evitar ambiguidades, podemos definir uma configuração de um pacote de estados inciais para a simulação de um modelo específico com:

$$\begin{aligned} Rede\ -\ Modelo\,(a, ((C_1, p_1),\ \ldots,\ (C_i, p_i),\ \ldots,\ (C_n, p_n)) \\ p/\ C_i \in C,\ e\ n = |C|, \\ tal\ que\ \sum_{i=1}^{n} p_i = 1 \end{aligned}$$

Por exemplo:

$$p2p - Gnutella08\ -\ SIS\,(15, ((S, 0.99),\ (I, 0, 01))$$

4.4.2 Variáveis Globais

Uma vez estabelecidos um modelo compartimental, uma topologia, um pacote de estados iniciais seguindo uma determinada proporção de distribuições aleatorizadas dos nodos desta rede nos compartimentos do modelo, para termos todos os elementos necessários para executar as simulações, nos resta definir as variáveis globais **p** e **i**. No intuito de observar a relevância destas variáveis na instanciação de simulação desejada, é ideal que escolhamos não um par de valores fixos para essas variáveis, mas sim um par de intervalos. Por exemplo, $p \in]0 - 0, 2]$ variando de em incrementos de 0, 1 e $i \in]0 - 10]$ variando em incrementos de 1, atentando para que ambos intervalos sejam discretos, pois é necessário fazer uma combinação de todos os valores de **p** com todos de **i** para que possamos executar uma rodada de simulações sobre todos os estados iniciais para cada uma dessas combinações. Neste caso teríamos dois valores para **p**, (0,1 e 0,2), e dez valores de **i**, (1, 2, 3, 4, 5, 6, 7, 8, 9 e 10), totalizando vinte rodadas de simulações para a instanciação completa de um modelo, como:

$$\begin{aligned} p2p - Gnutella08\ -\ SIS\,(15, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\ para\ p\ variando\ em\ incrementos\ de\ 0, 1\ e\ i\ variando\ em\ incrementos\ de\ 1 \end{aligned}$$

Cada rodada de simulações para uma combinação (p, i) sendo executada, neste caso, para cada uma das quinze configurações de estado inicial, totalizando $20 * 15 = 300$ simulações desde um estado inicial até um estado terminal da rede. Logo torna-se evidente que a eficiência

em tempo de execução das simulações é determinante para a viabilidade da avaliação significativa de quaisquer métricas do modelo que desejemos realizar.

4.4.3 O Módulo *Simulations*

Implementamos um módulo no framework, chamado *Simulations* com rotinas que geram pacotes de estados iniciais segundo a parametrização definida acima, criando um diretório de arquivos contendo estados iniciais. Questões de desempenho na criação desses pacotes também foram uma preocupação, discutiremos esssas questões na seção 4.5.

Este mesmo módulo também completa as atividades de simulações, implementado rotinas que instanciam a classe Network adequadamente e executa as simulações do pacote para cada combinação configurada das variáveis globais. O produto dessa execução completa é um arquivo **CSV** contendo os registros das funções configuradas para serem executadas a cada começo e fim de simulação e a cada rodada de cada uma das simulações. Com as lógicas integradas destas funções as métricas de cada execução são consolidadas nestes arquivos. Adicionalmente métricas do próprio módulo são registradas. Estes dados ficam organizados em um arquivo para cada par (p, i).

Este produto é suficiente para que se analise as métricas resultantes de todo o processo por qualquer ferramenta de análise de dados. Mas para nosso estudo da relevância dos atributos de instanciação das simulações nos modelos, implementamos uma rotina que consolida esses dados em séries estatísticas, afim de que possamos as visualizar.

4.5 Dificuldades e Desafios

Ao término da revisão bibliográfica sobre o estado da arte da pesquisa em modelos epidemiológicos, etapa inicial desse trabalho, era possível vislumbrar um panorama geral da área, e, principalmente, uma lacuna notável onde caberia experimentação com grafos. Daí então surgiu a motivação para a criação. No entanto, não havia muito mais que um esboço de objetivos as serem alçados, tampouco um direcionamento ou metas que pudessem guiar decisões de projeto. O trabalho se deu de forma exploratória, levantado e validando hipóteses sobre o que poderia ser feito, ou seria interessante e relevante durante o processo. As dificuldades e problemas relatados a seguir se deram, principalmente, devido a essa falta de um projeto preliminar claro.

No começo da implementação, a perspectiva de poder visualizar o comportamento dos modelos, principalmente o espalhamento da doença pela rede, compunha grande parte do interesse em trabalhar com essas redes. No entanto, conforme a classe *Network* foi tomando forma, a representação da informação nas estruturas de dados foi se tornando cada vez mais difícil de mapear para as interfaces da plataforma Gephi que era a primeira escolha para solução do problema de representação visual. Muito tempo de trabalho foi investido em tentar manter uma integração com a plataforma. Esforço esse que frustrado ao descobrir mais adiante que esta funcionalidade não seria vital para o direcionamento que foi se consolidado para o trabalho. A integração nunca foi concluída.

A primeira abordagem para a geração de estados iniciais da rede segundo uma distribuição de proporção parametrizada foi a mais trivial possível:

- todos os nodos eram colocados em um conjunto;
- a partir no número total de nodos da rede, eram calculadas as quantidades esperadas de nodos em cada um dos compartimentos;

- para cada compartimento, criava-se um conjunto, e até que nele houvesse a quantidade esperada de nodos:
  - removia-se um nodo aleatório do conjuntos dos nodos;
  - atribuía-se este nodo ao compartimento

O sorteio do nodo a ser removido do conjunto dos ainda não atribuídos a nenhum compartimento era, no entanto, um procedimento ineficiente que levava tempo linearmente proporcional ao tamanho remanescente do conjunto. As operações de remoção e adição em conjuntos também acrescentavam um grande custo ao processo todo. Foi quando a geração do conjunto de estados inciais de uma das redes passou da de duas horas de duração que este processo foi revisado. Finalmente substituímos essa função por uma rotina de embaralhamento nativa da linguagem, que implementa o algoritmo de Fisher–Yates [Wikipedia-EN-Fisher–Yates-Shuffle, 2017]. Esta mera ordenação aleatorizada da lista de nodos, seguida da separação em segmentos mapeados para cada partição segundo a proporção dada, passou a resolver o mesmo problema na escala de segundos.

A princípio, a construção do modelo em memória era um procedimento monolítico, ou seja, todos os parâmetros eram requeridos no momento instanciação da classe: variáveis **p** e **i**, arquivo de topologia e arquivo de estado inicial. Ao executar repetidas simulações foi possível observar, atravéz das mensagens de progresso, que o carregamento da topologia era responsável pela maior parte do tempo de execução. Conforme ficou claro que a intenção do framwork era realizar diversas execuções como a mesma topologia, foi pertinente o desmembramento da procedimento de carga da topologia do restante da configuração da classe. Posteriormente, a mesma ideia foi aplicada para o carga do estado inicial, e configuração das variáveis do modelo.

Uma das primeiras versões da calasse *Network* implementava um funcionalidade de reinicialização da rede. Para isso, uma cópia adicional do estado inicial, incluindo os atributos de todos os nodos, era armazenada na estrutura da classe. Esta particularidade tornava o processo de carga do estado inicial consideravelmente mais lento, e a memória utilizada pelo processo cerca de 20% maior. Por questões de desempenho esta funcionalidade foi desabilitada.

Por fim, na tentativa de viabilizar o uso de redes de maiores dimensões no framework, realizamos um processo de auditoria na execução dos códigos. Verificamos que a maior parte do tempo de toda a execução é gasta com o processamento das listas de nodos, mais especificamente na checagem de seus estágios para compor os conjuntos dos compartimentos. Esse procedimento acontece a cada mudança de estágio de um nodo na rede. Em segundo lugar, a execução do método obtenção de vizinhança de um nodo, que é chamado em todas as rodadas para cada um do nodos infectados. Para a melhoria desses aspectos, uma refatoração profunda na estrutura da classe seria necessária. Idealmente cada método deveria realizar todos os cálculos de atualização de estado da rede e dos nodos armazenando em estruturas auxiliares informações necessárias ao longo do da simulação, para que estas não precisassem se calculadas dinamicamente, e redundantemente a cada rodada, e no processamento de cada nodo. Esta refatoração constituiria em uma troca, abrindo mão da alta modularização e dinâmica das propriedades da classe, e também da sua eficiência em uso de memória, para obter um ganho substancial em tempo de execução. Infelizmente, sem uma metodologia de testes automatizados aplicada à na base de código do framework, tal refatoração é inviável, pois a probabilidade de introdução de bugs neste processo é muito alta, em vista da alta complexidade da classe. Sem tempo hábil para a implementação de testes, esta refatoração não foi executada.

Na auditoria, também notamos que a função mais executada é de geração de números aleatórios. Esta função é notadamente ineficiente em termos de tempo de execução [Lemire, 2017]. A substituição das chamadas a esta função por algum outro método de obtenção de valores

aleatórios, que fosse mais eficiente, teria um grande impacto no desempenho da simulação. Devido a falta de espaço no cronograma do trabalho, esta melhoria também não foi feita.

4.6 Conclusão

Evidentemente o software que conseguimos implementar para o framework proposto possui falhas e vários pontos de melhoria. No entanto, este código serve como um protótipo suficiente para verificarmos a relevância da consideração de topologias nos modelos epidemiológicos, bem como observar a relação dos parâmetros dos modelos em seus comportamentos. No próximo capitulo discutiremos os experimentos executados, seus parâmetros e as métricas escolhidas, e por sua vez as medições e resultados obtidos.

Capítulo 5

Resultados

5.1 Escolha do Modelo e Parâmetros

Primeiramente, para os propósitos destes experimentos, e até mesmo a garantia da viabilidade de executar as simulações de forma programática, seria interessante garantir que as simulações eventualmente chegassem a um fim. Ou seja, garantir que em algum momento da simulação o compartimento dos indivíduos infectados tivesse tamanho zero, configurando a extinção da doença na rede. Para isso selecionamos o único modelo implementado que não contempla a ressucetibilidade dos indivíduos, o modelo **SIR**.

Para submeter as topologias selecionadas ao framework, escolhemos os seguintes parâmetros de instanciação.

5.1.1 Redes Reais

Para cada rede deste grupo geramos dois pacotes de estados iniciais, um com as proporções de distribuição $((S, 0, 99), (I, 0, 01), (R, 0))$, para observar o comportamento da epidemia começando com um contágio bastante pontual, e um segundo com $((S, 0, 9), (I, 0, 1), (R, 0))$, para representar um contágio inicial mais generalizado na rede.

Para cada pacote definimos que o tamanho da amostra deveria ser de 50 estados iniciais, esperando que ao consolidar as métricas obtidas como médias dos resultados dessa amostragem, fosse possível eliminar suficientemente os efeitos da variância.

Para observar a relevância das variáveis globais nestes cenários, determinamos que seus valores deveriam variar desde bastante pequenos até valores grandes o suficiente para emular o comportamento de uma doença altamente contagiosa. Desta forma, os intervalos escolhidos foram

$$((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10])$$
$$\textit{para p variando em incrementos de } 0, 01 \textit{ e i variando em incrementos de } 1$$

Resultando assim em 200 combinações de variáveis, para cada pacote de estados iniciais. Considerando que cada combinação (p, i) deve ser executada para cada exemplar da amostra de 50 estados iniciais, temos um total de 10000 execuções de simulação, desde um estado inicial até um terminal do modelo para cada pacote de estados iniciais, ou 20000 execuções para cada rede. No momento dessas execuções, a eficiência computacional da implementação se torna bastante evidente.

Em termos da definição das instanciações do framework, temos, portanto:

$$
\begin{aligned}
p2p - Gnutella08 \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
p2p - Gnutella08 \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
p2p - Gnutella09 \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
p2p - Gnutella09 \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ca - GrQc \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ca - HepTh \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ca - HepTh \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ca - GrQc \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
para\ p\ variando\ em\ incrementos\ de\ 0,01\ e\ i\ variando\ em\ incrementos\ de\ 1
\end{aligned}
$$

O que se traduz, na prática, a oito invocações das rotinas do framework com estas parametrizações.

5.1.2 Redes de Árvores

Exatamente o mesmo raciocínio foi aplicado para a escolha de parâmetros das execuções sobre as redes arbitrariamente construídas com topologias de árvore, instanciando portanto:

$$
\begin{aligned}
ArvoreEstrela \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ArvoreEstrela \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ArvoreBinaria \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ArvoreBinaria \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ArvoreLista \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
ArvoreLista \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
para\ p\ variando\ em\ incrementos\ de\ 0,01\ e\ i\ variando\ em\ incrementos\ de\ 1
\end{aligned}
$$

5.1.3 Redes de Grafos Completos

A mesma lógica para a definição de parâmetros foi utilizada para as redes de grafos completos. No entanto, no caso da rede *completo1*, devido a suas dimensões e as limitações do desempenho do framework abordadas em na seção 4.5, definimos para sua execução somente um pacote de estados iniciais, com o mesmo tamanho de amostragem de 50, porém, um intervalo reduzido para as variáveis globais. Desta forma, temos:

$$
\begin{aligned}
Completo1 \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 1],\ i \in]0 - 5]) \\
Completo2 \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
Completo2 \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
Completo3 \ - \ SIR\,(50, ((S, 0.99),\ (I, 0, 01),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
Completo3 \ - \ SIR\,(50, ((S, 0.9),\ (I, 0, 1),\ p \in]0 - 0, 2],\ i \in]0 - 10]) \\
para\ p\ variando\ em\ incrementos\ de\ 0,01\ e\ i\ variando\ em\ incrementos\ de\ 1
\end{aligned}
$$

5.2 Consolidação dos Dados

Aqui podemos observar os dados finais das execuções. Para cada cada pacote de cada rede, consolidamos as médias, para cada valor de **p** e para cada valor de **i** das seguintes métricas:

- tempo total de execução de uma simulação, em segundos, ou seja, desde o estado inicial, até a rodada final, descontando-se os tempos de carregamento de topologia e estado inicial;
- número de rodadas até o término da simulação, ou seja, a duração da vida da epidemia em termos de iterações de contágio/recuperação dos indivíduos na rede;
- o tamanho da maior infecção da rede ao longo da simulação, em termos de porcentagem da rede pertencente ao compartimento dos infectados, em uma determinada rodada da simulação;
- a rodada de ocorrência da infecção máxima da rede.

Arranjamos estas médias como séries relacionadas aos valores de **p** e **i**. Com estas métricas esperamos ter uma noção da complexidade da simulação para cada tipo de rede, a gravidade da epidemia, e a influência sobre estes dois aspectos das variáveis do modelo.

Podemos observar a seguir, essas métricas dispostas em 4 gráficos para cada rede, exceto a *completo1*, que teve apenas um cenário de simulação.

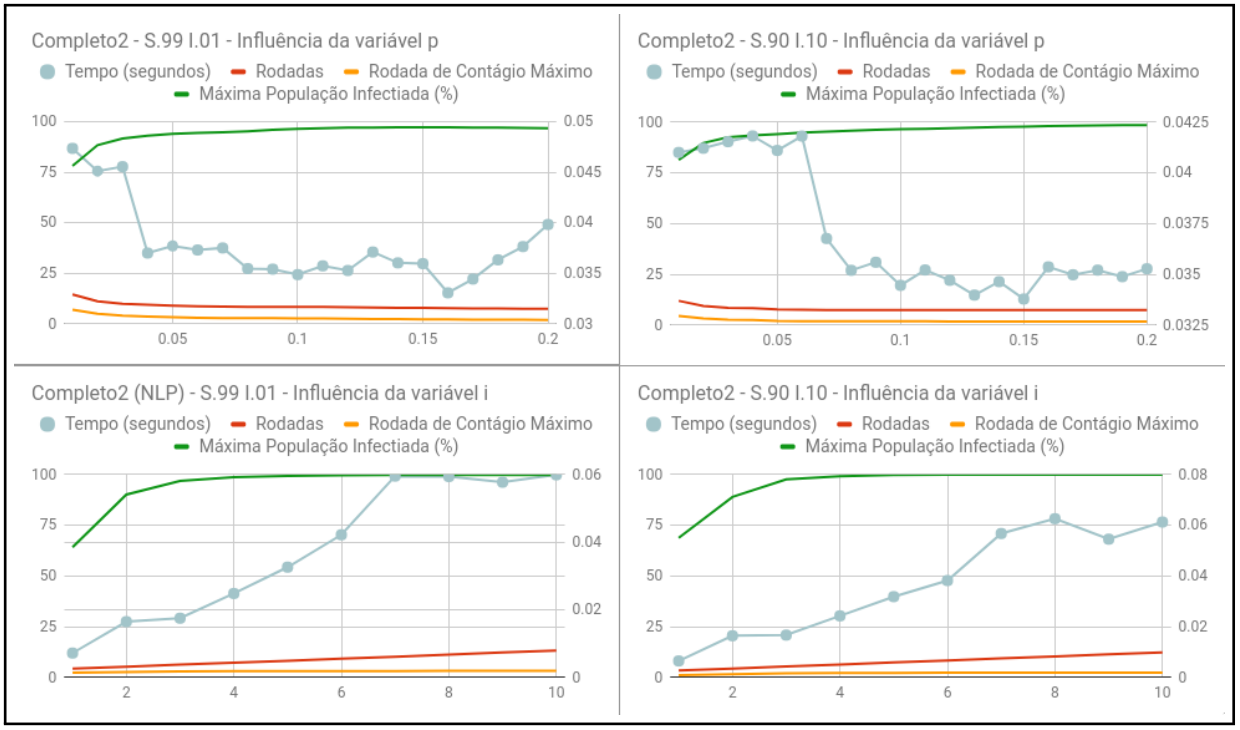

Figura 5.1: Influência das variáveis **p** e **i** nos cenários de simulação para a rede completo2

5.3 Análise

A primeira observação que cabe ser levantada, é como as redes de grafos completos tiveram resultados praticamente idênticos em todos os cenários. Independentemente de suas

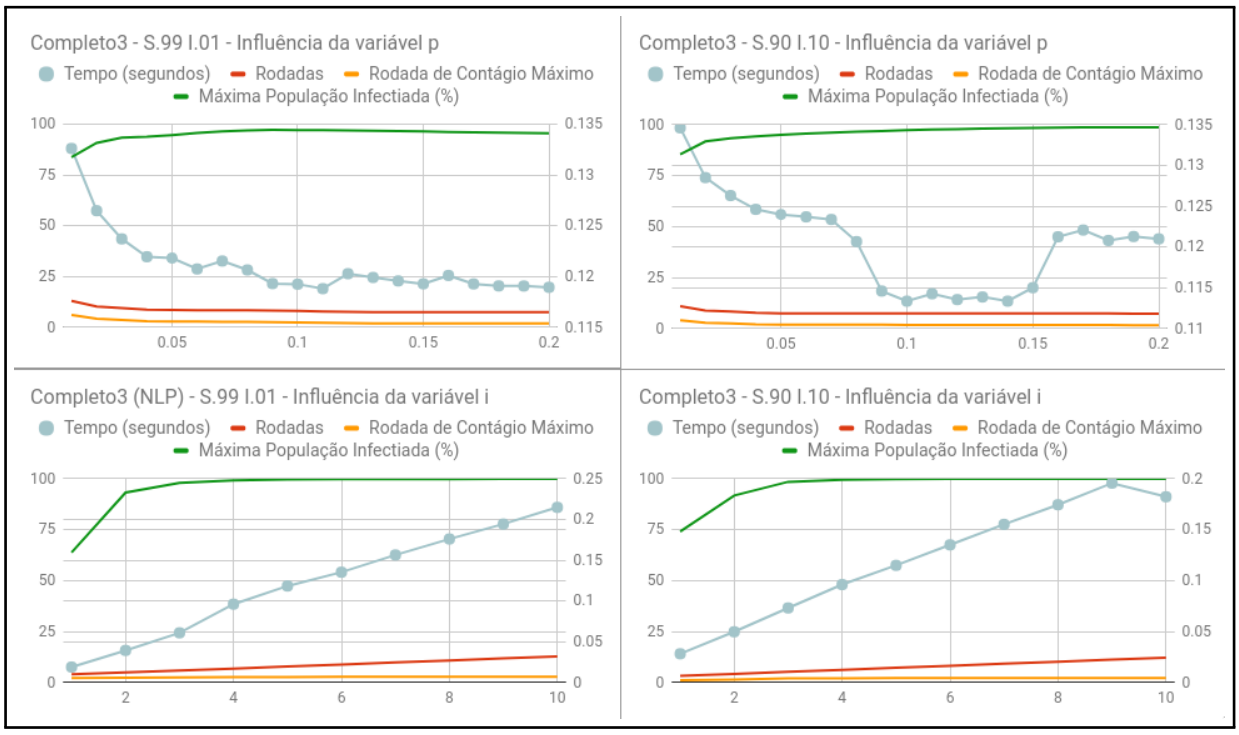

Figura 5.2: Influência das variáveis **p** e **i** nos cenários de simulação para a rede completo3

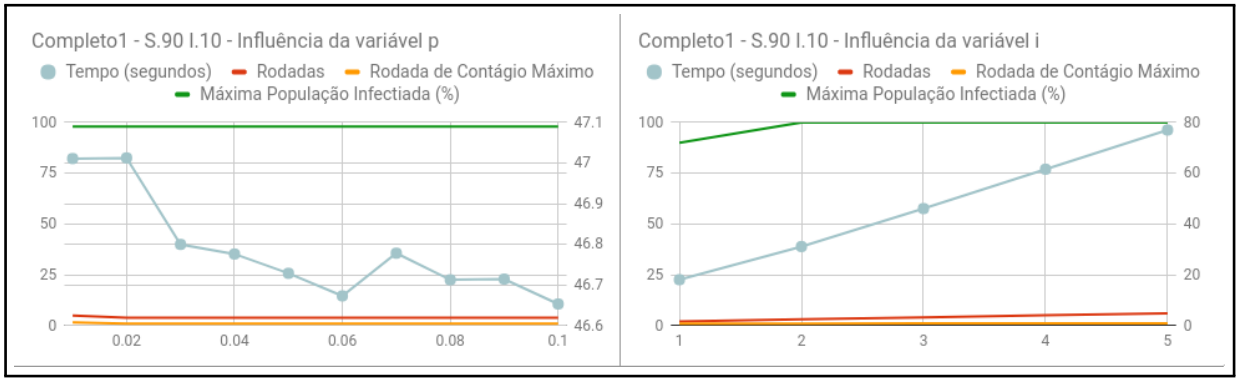

Figura 5.3: Influência das variáveis **p** e **i** nos cenários de simulação para a rede completo1

dimensões, e de todas as parametrizações, ou seja, proporções dos estados iniciais, e das variáveis **p** e **i**. Em todos os casos, aproximadamente 100% da população é infectada logo nas primeiras rodadas das simulações. Apenas algumas pequenas relações puderam ser notadas entre as variáveis. O crescimento assintoticamente linear da quantidade total de rodadas da simulação com relação a **i**, devido ao fato de que, uma vez que praticamente toda a população se encontra infectada, ainda devem decorrer aproximadamente mais **i** rodadas para o termino da simulação. E uma dramática redução do tempo de execução após um certo valor de **p**, provavelmente por que depois deste valor, a rede se torna totalmente infectada, muito rapidamente, reduzindo, por consequência, a quantidade de gerações de números aleatórios a partir deste ponto.

5.3.1 Redes de Árvores

Em todos os cenários, a curva de tempo de execução teve um comportamento muito similar à da população máxima infectada, evidenciando uma relação equivalência entre o máximo

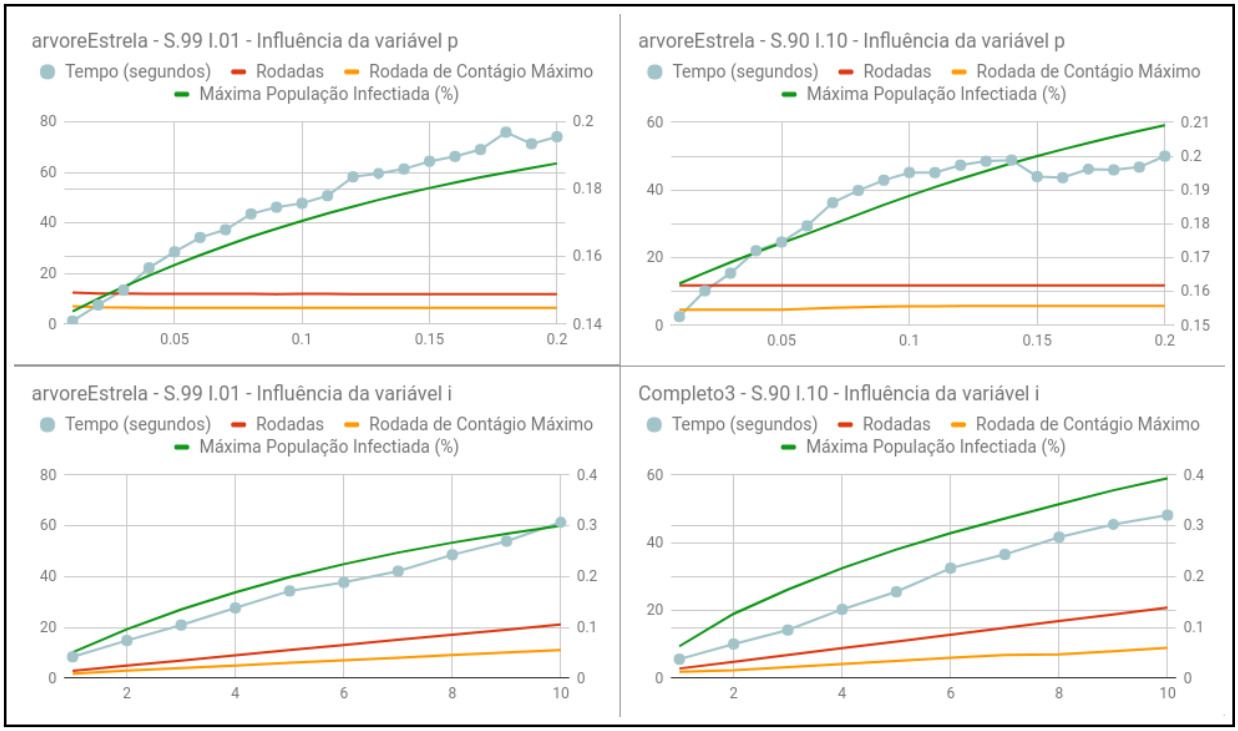

Figura 5.4: Influência das variáveis **p** e **i** nos cenários de simulação para a rede ArvoreEstrela

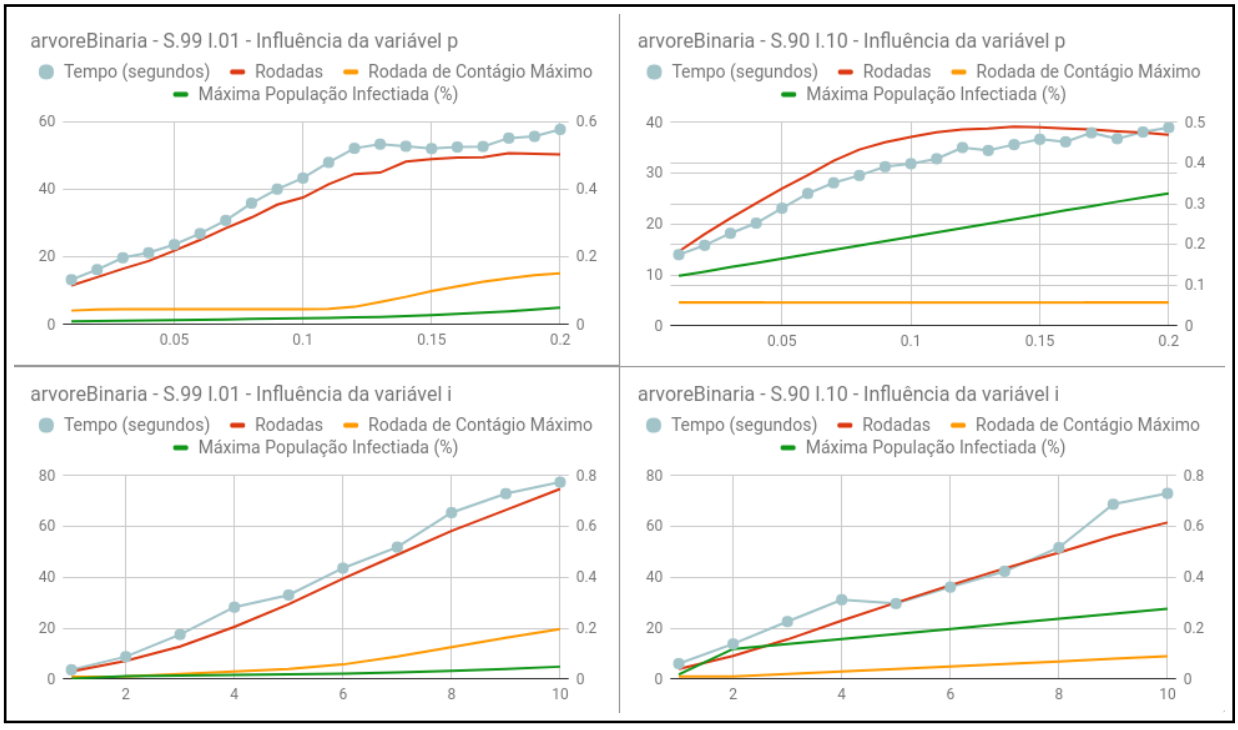

Figura 5.5: Influência das variáveis **p** e **i** nos cenários de simulação para a rede ArvoreBinaria

tamanho da população infectada numa simulação e as geração de números aleatórios, para redes de baixa densidade.

Tanto na rede arvoreLista, quanto na rede arvoreBinaria, no cenário em que apenas 1% da população começa a simulação no estágio infectado, a população máxima infectada não passou de 10%, independentemente das variáveis **p** e **i**, exatamente o contrário do que aconteceu

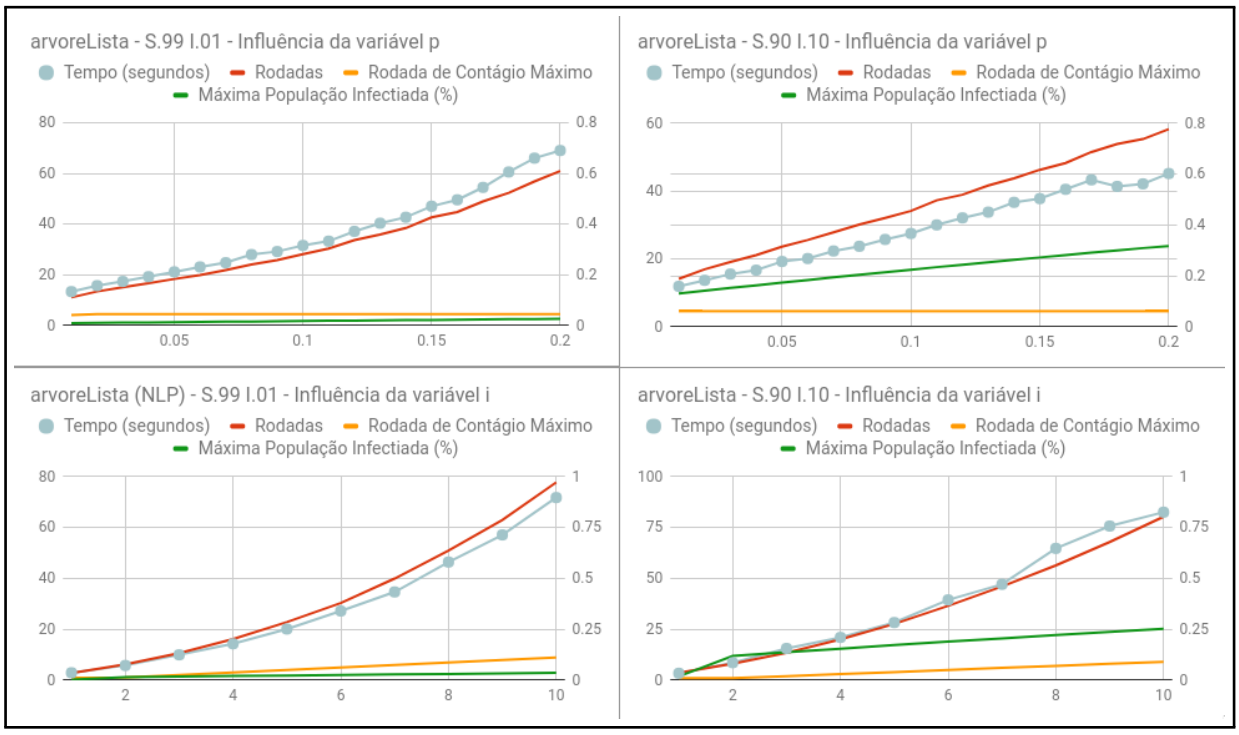

Figura 5.6: Influência das variáveis **p** e **i** nos cenários de simulação para a rede ArvoreLista

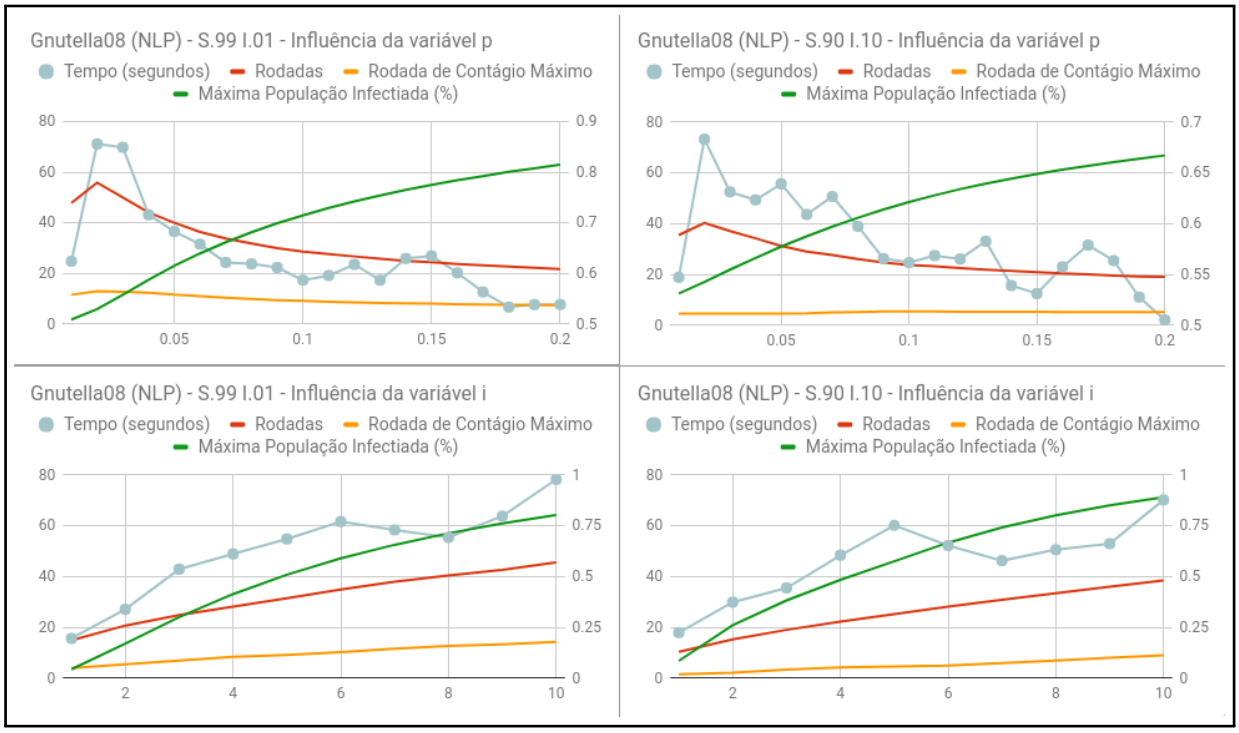

Figura 5.7: Influência das variáveis **p** e **i** nos cenários de simulação para a rede p2p-Gnutella08

nas redes de grafos completos. Este fenômeno sugere uma forte relação entre o grau máximo da rede, e o alcance da doença.

Na rede de arvoreEstrela, o número de rodadas da simulação e a rodada da ocorrência da infecção máxima da rede parecem depender exclusivamente do valor de **i**, mantendo-se constantes com relação a **p**.

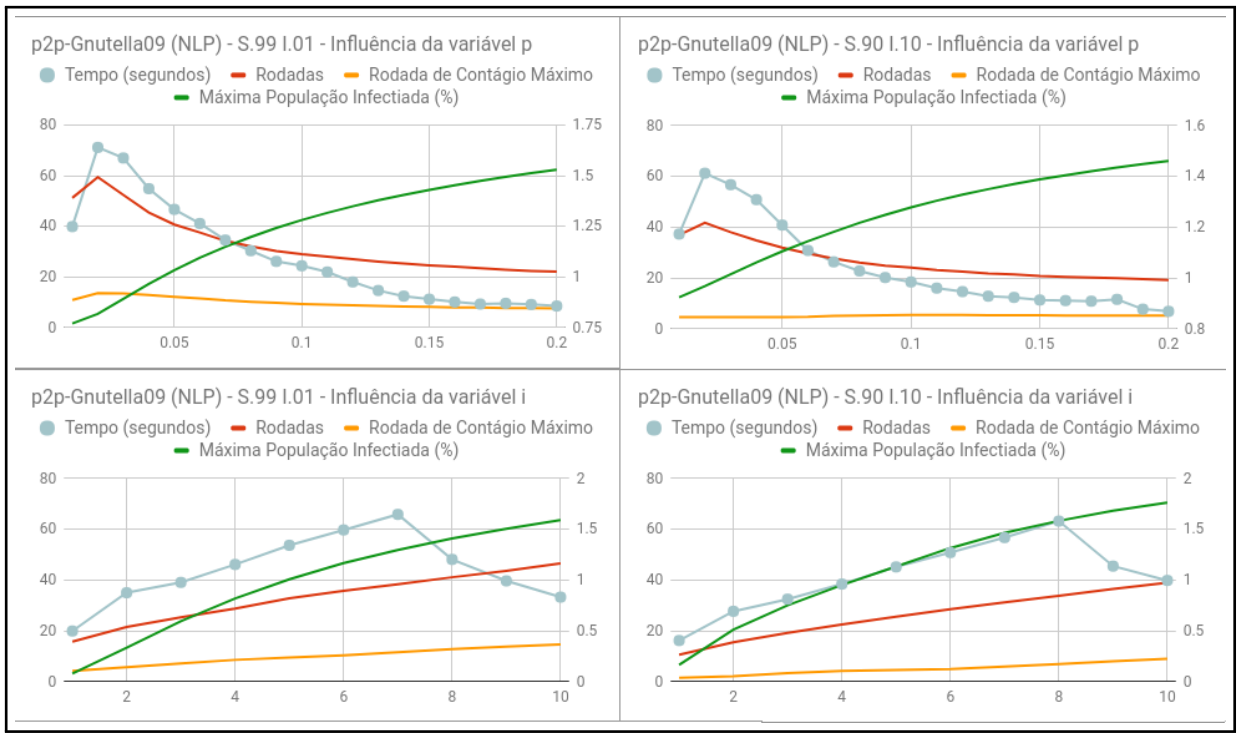

Figura 5.8: Influência das variáveis **p** e **i** nos cenários de simulação para a rede p2p-Gnutella09

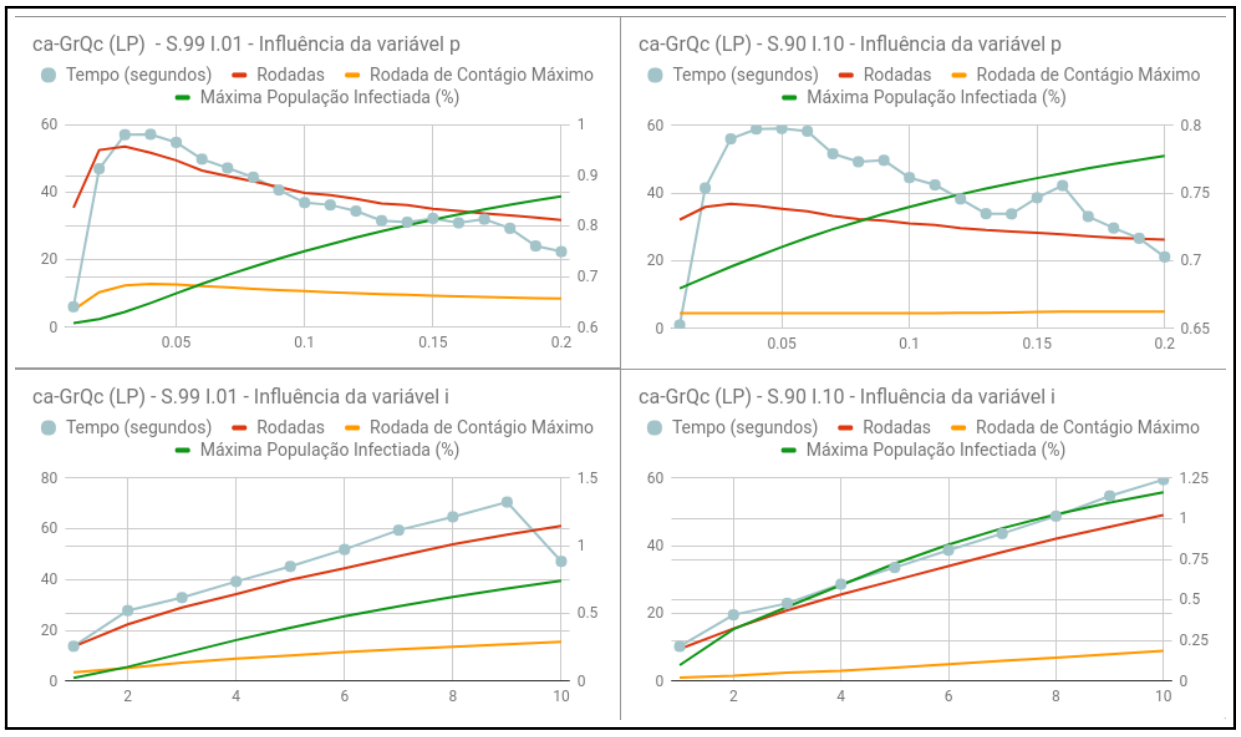

Figura 5.9: Influência das variáveis **p** e **i** nos cenários de simulação para a rede ca-GrQc

5.3.2 Redes Reais

Exceto as redes Gnutella08 e Gnutella09, que foram descritas a partir de fontes muito similares uma à outra, cada rede apresentou resultados diversos entre si. Apenas com este experimento, não é possível observar diferenças muito suficientemente significativas para levantar

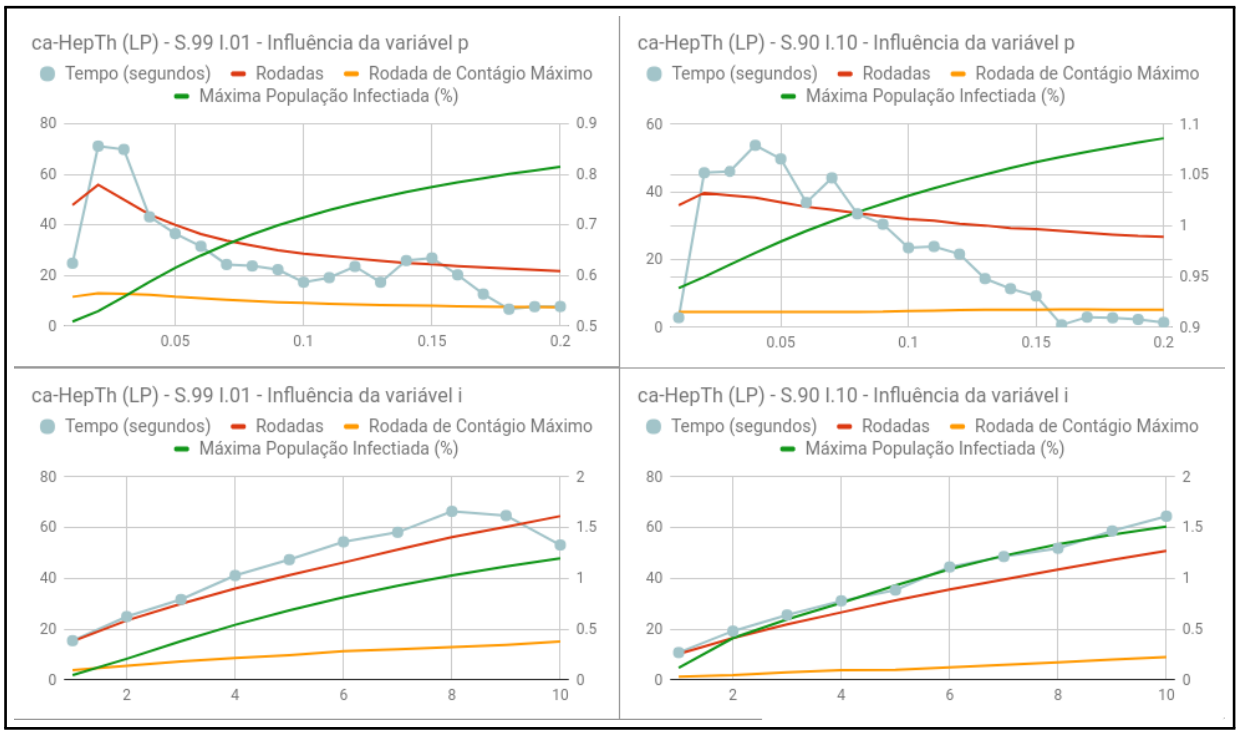

Figura 5.10: Influência das variáveis **p** e **i** nos cenários de simulação para a rede ca-HepTh

hipóteses sobre a natureza destas topologias. O que é possível, no entanto, é verificar que as curvas destas redes reais não seguem o padrão de comportamento das redes arbitrariamente definidas a partir de grafos notáveis. Para a finalidade deste trabalho, esta constatação é boa o suficiente.

Capítulo 6

Conclusão e Trabalhos Futuros

Ao termino deste trabalho, posso dizer que, a implementação de uma framework para lidar com a simulação, definição e obtenção e análise de métricas estatísticas de modelos compartimentais estocásticos de epidemiologia é um desafio grande. Certamente maior que o escopo de um trabalho de graduação. A sensação geral é de que, apesar de todo o esforço investido, ainda falta muito para que o software discutido neste trabalho atinja completamente os objetivos propostos. No entanto, já temos um bom começo.

Quanto ao principal objetivo deste trabalho, demonstrar a relevância da consideração da topologia no modelo epidemiológico, pudemos observar que as diferentes redes proporcionaram diferentes resultados, fazendo com que as variáveis do modelo impactassem de maneiras diversas as métricas dos modelos. Desta forma, torna-se evidente que a topologia da rede interfere dramaticamente no modelo. Justificando-se assim os o trabalho no sentido de elaborar e viabilizar uma plataforma de customização de modelos epidemiológicos que inclua a topologia no centro do modelo.

Vamos então, avaliar softawre produzido, segundo os objetivos pretendidos.

Gostaríamos que o uso do framework fosse prático. E de fato, uma vez que a rede de contato da doença a ser estudada está devidamente levantada, o software implementado fornec um ferramental bastante fácil de usar, e que permite a criação de modelos altamente customizáveis, e o total controle sobre as métricas extraídas do modelo. O código fonte está disponível a partir do GitHub[Sabino, 2017], o que garante a facilidade na distribuição, e a possibilidade de que quaisquer interesados possam usa-lo e contribuir para sua melhoria. No entanto, se quisermos, realisticamente, que ele seja usado e disperte interesse da comuindade, algumas melhorias devem ser implementadas, a saber:

- criação de uma documentação de alto nível e disponibilização desta no repositório central de documentações para pacotes Python [readthedocs.org, 2017];
- implementação de testes automatizados para garantir que novas adições e alterações no código não modifique a lógica dos modelos já implementada;
- integração com a base de pacotes padrão do Python, a PiPy [pypi.python.org, 2017], para a fácil instalação;
- refatoração da classe *Network* para implementar as melhorias de desempenho apontadas na seção 4.5;
- estudo e prospecção da redução de sub-problemas do simulador em outros problemas que possuam algoritmos eficientes conhecidos, como é o exemplo da geração de números aleatórios, conforme apontado na seção 4.5;

- reavaliar a utilidade final da biblioteca *Networkx* para o domínio específico de problema que estamos abordando, uma vez que esta biblioteca é de propósito geral em grafos, e já observamos que muitos problemas de desempenho, no nosso caso, derivam da organização desta ferramenta;
- revisar toda a implementação para entender em que pontos é possível substituir implementações próprias por funções matemáticas e estatísticas de alto desempenho implementadas pelas bibliotecas Numpy e Scipy. Estas bibliotecas provém, algumas das implementações (linguagem C) mais eficientes conhecidas de problemas de computação científica e numérica. [scipy.org, 2017].

Referências Bibliográficas

[Barry Warsaw and Coghlan, 2017] Barry Warsaw, Jeremy Hylton, D. G. and Coghlan, N. (2017). Pep 1 – pep purpose and guidelines. http://www.python.org/dev/peps/pep-0001/. Acessado em 02/07/2017.

[de Oliveira Cabral Filho, 2016] de Oliveira Cabral Filho, E. (2016). Cobertura por vértices mínima em grafos lei de potência. Master's thesis, Pós-Graduação em Informática - Universidade Federal do Paraná, Curitiba - PR.

[Easley and Kleinberg, 2010] Easley, D. and Kleinberg, J. (2010). *Networks, Crowds, and Markets*. Cambridge University Press.

[Galante, 2008] Galante, G. (2008). Epidemiologia matemática e computacional. http://www.inf.unioeste.br/~guilherme/pesq/report.pdf. Acessado em 01/06/2017.

[Gephi, 2017] Gephi (2017). Gephi. http://gephi.org/. Acessado em 02/07/2017.

[Hethcote, 2000] Hethcote, H. W. (2000). The mathematics of infectious diseases. *Society for Industrial and Applied Mathematics Review*, 42(4):599–653.

[Keeling and Eames, 2017] Keeling, M. J. and Eames, K. T. (2017). Networks and epidemic models. http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1578276/. Acessado em 02/07/2017.

[Lemire, 2017] Lemire, D. (2017). Ranged random-number generation is slow in python. http://lemire.me/blog/2016/03/21/ranged-random-number-generation-is-slow-in-python/. Acessado em 02/07/2017.

[Lloyd and Valeika, 2007] Lloyd, A. L. and Valeika, S. (2007). Network models in epidemiology: An overview. *World Scientific Lecture Notes in Complex Systems*, 7:189–214.

[MatplotLib, 2017] MatplotLib (2017). Matplotlib. http://matplotlib.org/. Acessado em 02/07/2017.

[Netwokx, 2017] Netwokx (2017). Netwokx. http://networkx.github.io/. Acessado em 02/07/2017.

[Network-Repository, 2017] Network-Repository (2017). Network repository. http://networkrepository.com/. Acessado em 02/07/2017.

[pypi.python.org, 2017] pypi.python.org (2017). Pypi. https://pypi.python.org/pypi. Acessado em 02/07/2017.

[Python-Performance-Tips, 2017] Python-Performance-Tips (2017). Python performance tips. http://wiki.python.org/moin/PythonSpeed/PerformanceTips. Acessado em 02/07/2017.

[readthedocs.org, 2017] readthedocs.org (2017). Read the docs. https://readthedocs.org/. Acessado em 02/07/2017.

[Rocha, 2012] Rocha, D. I. C. (2012). Modelos matemáticos aplicados à epidemiologia. Master's thesis, Faculdade de Economia da Universidade do Porto, Cidade do Porto, Portugal.

[Rossi and Ahmed, 2015] Rossi, R. A. and Ahmed, N. K. (2015). The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*.

[Sabino, 2017] Sabino, A. (2017). Scemf - stochastic compartmental epidemiology models framework. http://github.com/adolfo-sabino/SCEMF. Acessado em 02/07/2017.

[scipy.org, 2017] scipy.org (2017). Scipy. http://www.scipy.org/. Acessado em 02/07/2017.

[Terra, 2009] Terra, F. M. (2009). Desenho de grafos: Uma abordagem utilizando programação linear inteira. Master's thesis, Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, Belo horizonte - MG.

[Wikipedia-EN-Fisher–Yates-Shuffle, 2017] Wikipedia-EN-Fisher–Yates-Shuffle (2017). Fisher-yates shuffle. http://en.wikipedia.org/wiki/Fisher_Yates_shuffle. Acessado em 02/07/2017.

[Wikipedia-EN-Power-Law, 2017] Wikipedia-EN-Power-Law (2017). Power law. http://en.wikipedia.org/wiki/Power_law. Acessado em 02/07/2017.

[Wikipédia-EN-Epidemiology, 2017] Wikipédia-EN-Epidemiology (2017). Wikipédia en - epidemiology. https://en.wikipedia.org/wiki/Epidemiology. Acessado em 01/06/2017.

[Wikipédia-PT-BR-Epidemiologia, 2017] Wikipédia-PT-BR-Epidemiologia (2017). Wikipédia pt-br - epidemiologia. https://pt.wikipedia.org/wiki/Epidemiologia. Acessado em 01/06/2017.