MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CURITIBA
ENGENHARIA DE COMPUTAÇÃO

LEONARDO PRESOTO DE OLIVEIRA

CONTROLABILIDADE EM REDES COMPLEXAS

CURITIBA
2014

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CURITIBA
ENGENHARIA DE COMPUTAÇÃO

LEONARDO PRESOTO DE OLIVEIRA

CONTROLABILIDADE EM REDES COMPLEXAS

Dissertação apresentada à disciplina Trabalho de Conclusão de Curso 2 do Curso Superior de Engenharia de Computação, dos Departamentos Acadêmicos de Informática e Eletrônica da Universidade Tecnológica Federal do Paraná, como requisito parcial para obtenção do título de Engenheiro de Computação

Orientador: Gustavo Alberto Giménez Lugo

CURITIBA
2014

Licenciamento

AGRADECIMENTOS

- Ao meu pai Ivar e a minha mãe Gislene, que trabalharam muito duro para dar a a mim e a minha irmã Cecília o estudo que não tiveram. Espero daqui para frente poder retribuir tudo que fizeram para mim.

- A minha irmã Cecília que mesmo apesar das (constantes) brigas sempre me apoiou. Acho que a cada briga eu gosto mais dela (então imagine só???).

- Aos meus Avós Geraldo (Peba) e Mafalda aos quais eu também devo muito dos meus estudos e da educação que tive. Só eu sei o quanto vocês são importantes para mim.

- Ao meu Avo João Eduardo falecido durante o curso. Pessoa que eu também admirava e respeitava muito.

- À minha família Angélica, Letícia, Bruno e Bia por fazerem parte da minha vida, e estarem presentes nos momentos mais importantes dela.

- À minha segunda família Corrêa Borsato, a qual só tenho a agradecer pelo carinho.

- Aos amigos que fiz morando na cidade; Ronaldo e família que sempre me apoiaram; André Saliba grande pessoa; Neymar, que foi meu melhor amigo aqui nos maiores perrengues que passamos; e Wedson, grande cara que tive o prazer de conhecer.

- Aos meus Droogs da Universidade: Grande Ari, Murilo, Alessi, Eduardo, André, Leandro, Ivan, Ricardo, Domanski.

- Ao professor Murilo pela importante ajuda durante o trabalho.

- Ao professor Rafael, por disponibilizar o servidor do Departamento de Física, e por se interessar em auxiliar o projeto.

- Ao pessoal do DAFIS, serei um engenheiro formado dentro do departamento de Física (com muito orgulho). Professores Lenz, Nestor em especial ao professor Arandi, que foram orientadores, mestres, amigos, conselheiros. Só tenho a agradecer tudo que passei com vocês; e dizer que se no futuro eu me tornar um terço dos profissionais e homens que vocês são, estarei mais do que realizado, muito obrigado sempre.

- Ao meu orientador Gustavo, grande pessoa com quem também aprendi muito; e que

teve muita paciência comigo durante a orientação. Muito obrigado pelas conversas e opiniões durante este período.

- Um agradecimento ao meu amor, Juliana, pelo amor incondicional, pela força. There are places I remember ...

Esta página com certeza estará sempre em construção ...

Não sou nada.
Nunca serei nada.
Não posso querer ser nada.
À parte isso, tenho em mim todos os sonhos do mundo.

Álvaro de Campos

RESUMO

OLIVERA, L. P.. CONTROLABILIDADE EM REDES COMPLEXAS. 91 f. Dissertação – Trabalho de Conclusão de Curso - Engenharia de Computação (Monografia), Universidade Tecnológica Federal do Paraná. Curitiba, 2014.

Durante os últimos 25 anos, pesquisas relacionadas a sistemas complexos trouxeram novas perspectivas e metodologias ao estudo de fenômenos sociais e naturais. Da rede econômica formada por grandes corporações, até a dinâmica de processos celulares em biologia, inúmeras são as aplicações e benefícios gerados por esses avanços. Entretanto, o não determinismo intrínseco a esses sistemas tem sido um grande empecilho na busca por sua controlabilidade (capacidade de ser controlar a rede). O desenvolvimento de um método de controle capaz de guiar uma rede complexa até uma desejada configuração, através da manipulação de poucas variáveis, traria grande contribuição na compreensão científica de alguns fenômenos emergentes da natureza e da sociedade. Sendo assim, esse trabalho tem como objetivo avaliar um algoritmo capaz de, em tempo finito, identificar um subconjunto de nós controladores(nós que podem interferir no controle da rede) em um grafo de sistema complexo. O estudo foi fundamentado no artigo Controllability of Complex Networks, de LIU (2011), e motivado pelo artigo The Network of Global Corporate Control, de Battiston et al (2007). O desenvolvimento foi feito em linguagem Java, e os testes conduzidos com o auxilio de ferramentas de simulação de redes.

Foram desenvolvidos dois algoritmos gulosos, um guloso com a heurística de escolher os nós com menor grau e outro guloso de aproximação. O resultado obtido com estes algoritmos foram comparados ao algoritmo ótimo desenvolvido no artigo Controllability of Complex Networks (LIU, 2011). Obteve-se um erro médio de 6,25% para o caso do algoritmo com a heurística de escolha do menor nó e 73,41% para o algoritmo guloso de aproximação.

A procedência das escolhas que levaram ao algoritmo proposto e os bons resultados apresentados nos testes podem justificar a continuidade da pesquisa à nível de um mestrado científico.

Palavras-chave: Controlabilidade, Sistemas Complexos, Teoria dos Grafos, Emparelhamento

ABSTRACT

OLIVERA, L. P.. CONTROLLABILITY ON COMPLEX SYSTEMS. 91 f. Dissertação – Trabalho de Conclusão de Curso - Engenharia de Computação (Monografia), Universidade Tecnológica Federal do Paraná. Curitiba, 2014.

During the last 25 years, research related to complex systems brought new perspectives and methodologies to the study of social and natural phenomena. From the economic network formed by large corporations, to the dynamics of cellular processes in biology, there are countless applications and benefits of these advances. However, the non-determinism inherent to these systems has been a major impediment in the search for its controllability. The development of a control method capable of guiding a complex network to a desired configuration, through the manipulation of a few variables, would bring great contribution to the scientific understanding of some nature and society phenomena. Therefore, this study aims to evaluate an algorithm that, in a finite time, identify a subset of driver nodes in a graph of complex system. The study was based on the paper Controllability of Complex Network, of Liu et al. (2011), and motivated by the paper The Network of Global Corporate Control, of Battiston et al. The development was done in Java language, and the tests conducted with the aid of network simulation tools.

Two greedy algorithms were developed, one with the heuristic of choosing the driver nodes with lesser degree, and another approximation one. The results of these algorithms were compared to the optimal algorithm as developed in paper Controllability of Complex Networks (LIU, 2011). There was obtained an average error of 6.25% in the case of the algorithm with heuristics choice to the smaller node and 73.41% for the greedy approximation algorithm.

The origin of the choices that led to the proposed algorithm and the good results in tests justify continuing research to a MSc level.

Keywords: Controllability, Complex Systems, Graph Theory, Matching

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

NAFTA	Tratado Norte-Americano de Livre Comércio (inglês: *North American Free Trade Agreement*)
BOVESPA	Bolsa de Valores de São Paulo
NASDAQ	Associação Nacional Corretora de Valores e Cotações Automatizadas (inglês: *National Association of Securities Dealers Automated Quotations*)

LISTA DE SÍMBOLOS

G = (V,E)	- Grafo = (Vértices, Arestas)
V	- Conjunto de Vértices
E	- Conjunto de Arestas
G=(V, E , p)	- Definição alternativa de grafos. Grafo = (Vértices, Arestas, pesos)
V(G)	- Conjunto de Vértices do Grafo G
E(G)	- Conjunto de Arestas do Grafo G
p(G)	- Função peso do Grafo G
$\|V\|$	- Ordem de um Grafo
$\|E\|$	- Dimensão de um grafo
$g_G(v)$	- Grau de um vértice pertencente ao grafo G
A(G)	- Representação de Grafo Completo
$\overline{G}$	- Complemento de um Grafo G
Z(E)	- Soma dos graus de todos os vértices de um grafo
$\|V\|$	- Número de vértices de um grafo

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo serão apresentados o contexto em que se insere o tema escolhido, o problema, seus objetivos, a motivação que levaram ao desenvolvimento do projeto, bem como a forma de estruturação deste documento.

A "Hipótese de Gaia", apresentada por Lovelock e Margulis em seu artigo *"Atmospheric homeostasis by and for the biosphere: the gaya hypotesis"* (LOVELOCK; MARGULIS, 1974) sugere que todos os organismo orgânicos ou inorgânicos da Terra estão intimamente ligados e interagem entre si para manter as condições de vida no planeta. Esses organismos formam por sua vez um sistema complexo e estão ligados por uma rede de inter-relacionamentos.

Paralelamente ao ramo da Biologia, também aconteciam experimentos no ramo da Sociologia que visavam determinar a interligação entre os seres humanos. O mais conhecido deles foi o experimento do sociólogo americano Stanley Milgram (1967),conhecido como Teoria dos Seis Graus de Separação. O pesquisador desejava mensurar a distância social entre as pessoas nos Estados Unidos. Seu experimento baseou-se em escolher dois destinatários situados na região de Boston; e solicitou então que voluntários que moravam em outras regiões do pais fizessem que as cartas chegassem aos destinatários, por meio de pessoas intermediárias que poderiam, ou não, conhecer diretamente os destinatários. O resultado obtido foi que das 160 cartas enviadas inicialmente, 42 chegaram aos destinatários e, em média, precisaram passar por 5,8 (aproximadamente 6) intermediários.

No ano de 2012 foi lançado o artigo *"Four Degrees of Separation"*(BACKSTROM et al., 2012), em que os autores utilizaram as conexões do Facebook (vista como ferramenta de medição) e propuseram que as pessoas estão ligadas na Terra, por em média, 4 graus de separação, ou seja, é possível que qualquer pessoa contate outra utilizando-se em média de 4 conhecidos intermediários.

Tanto a pesquisa de Milgram, quanto a mais recente que se utiliza do Facebook, sugerem que as pessoas podem estar conectadas em uma rede. O estudo das redes é feito com base

na Teoria de Grafos[1], ramo da matemática responsável por estudar as redes e suas propriedades. De fato forma mais comum de se representar uma rede é através de grafos. Um grafo é dado por um par G= (V,E), onde V representa um conjunto arbitrário de vértices, e E são subconjuntos de pares de V conhecidos como arestas (BONDY; MURTY, 2008).

Grafos são representados visualmente por pontos (vértices) que se interligam com linhas (arestas). Essas arestas, por sua vez, podem ser valoradas, bilaterais, ou unilaterais, dependendo do caso em estudo.

A representação de diversos fenômenos ou estruturas do mundo real sob a perspectiva de Teoria dos Grafos, recai sobre um determinado tipo de sistema, conhecido como Sistema Complexo. Definido como estruturas topológicas não triviais, com vértices e arestas altamente dependentes, de forma que qualquer mudança nesses elementos leve a uma nova configuração do sistema (BULLMORE; SPORNS, 2009). O conceito de Sistemas Complexos será definido formalmente na Seção 2.

O pesquisador Lazso Barabási, ao estudar sistemas complexos, procurou definir padrões que pudessem levá-lo a predizer o comportamento dessas redes. Baraábsi defende que nenhum evento ocorre independentemente, e sim interagindo com os ouros componentes da rede.

A evolução de sua pesquisa o levou a seu artigo *"Controlability of complex networks"* (LIU et al., 2011), no qual o autor busca definir vértices que seriam capazes de influenciar o comportamento da rede. Esses vértices foram denominados *"driver nodes"*, e seriam, segundo o autor, a chave para se controlar sistemas complexos.

Barabasi se utiliza do algoritmo de emparelhamento de Hopfcroft-Karp (HOPCROFT; KARP, 1973) como ferramenta auxiliar para determinar o número de *driver nodes*. Trata-se de um algoritmo ótimo com complexidade O($\sqrt{mn}$) (no pior caso), em que m representa o número de vértices e n o número de arestas.

Verificar a controlabilidade de um grafo pode trazer uma grande gama de aplicações para o trabalho; qualquer relação, seja ela econômica, social, biológica ou qualquer outra pode ser mensurada e entendida segundo um conjunto de atributos significantes ao fenômeno observado. No caso de uma rede social, atributos como interação e distância entre os nós influem fortemente na configuração do grafo. Já em uma rede econômica, o valor de interação (força da relação) pode significar mais do que quantidade .

A pesquisa tem foco acadêmico mas, pelo grande número de situações que podem ser simuladas, o interesse pode não se restringir apenas ao acadêmico. Empresas de marketing,

[1] a palavra grafo é um neologismo para a palavra da língua inglesa *graph*

bancos, sistemas de saúde, transporte, entre outros, são exemplos de grupos fora da academia que podem se interessar por este tipo de trabalho.

Em seu artigo *"The network of global corporate control"*, Vitalli, Glattefelder e Battiston, 2009, discutem a possibilidade de poucas empresas concentrarem grande parte do controle econômico do mercado.

No caso do marketing, a empresa poderia usar os dados de uma rede social, como o Twitter, para definir qual a pessoa ideal para estrelar uma campanha e atingir em maior número o público alvo. Outro exemplo importante da utilização das redes sociais foi a campanha presidencial estadunidense de 2008, na qual Barack Obama utilizou fortemente ferramentas como Twitter e Youtube entre outros, como é retratado nos artigos (GOMES et al., 2009) e (CÂMARA; PORTO, 2011).

A ascensão das compras *on-line*, faz com que as empresas que vendem na internet procurem cada vez mais conhecer o seu cliente com a finalidade de oferecer a ele o produto que mais lhe interesse. O Google e Facebook (MATEUS, 2010), por exemplo, são especialistas em captar os dados dos usuários em seus sistemas e direcionar a propaganda ao cliente "que mais provavelmente" comprará o produto.

Esse comércio "direcionado", baseado em dados do cliente, é uma área delicada (do ponto de vista ético), pois o cliente pode se sentir lesado e com a sua privacidade invadida. Dada a importância do tema, o Marco Civil da internet, proposto em abril de 2014 (LEI Nº 12.965, DE 23 ABRIL DE 2014), tem, entre seus artigos, diretrizes que visam restringir esse tipo de prática a fim de proteger o interesse dos usuários. Deve-se esclarecer que a função deste trabalho não é defender ou condenar tal técnica mas, como se trata de uma grande área de interesse, é necessário cita-lá.

1.1 PROBLEMA

O trabalho busca o cálculo e uso de algoritmos de controlabilidade (capacidade de controlar a rede), que tentem melhorar o desempenho de algoritmos de controlabilidade de grafo. A principal maneira de se encontrar os *"driver nodes"* é pelo método do emparelhamento, porém os algoritmos existentes têm complexidade relativamente alta, dificultando a aplicação do método para redes maiores (com milhares ou até milhões de vértices).

No artigo *"Controllability of complex networks"*, (Yang, Slotine e Barabási, 2011), apresentam ferramentas analíticas para o estudo de controlabilidade em uma rede complexa direcionada qualquer. São usadas redes reais como a da cadeia alimentar e a da internet, por

exemplo. Ao final do artigo, os autores concluem que muitos aspectos de controlabilidade ainda podem ser explorados, exatamente ou analiticamente, para redes arbitrárias, se forem combinados aspectos de teoria de controle e estudos de redes. Isso poderia abrir caminho para aprofundar o conhecimento sobre sistemas complexos.

Liu et al. (2011) chama os nós que têm maior controle sobre outros de *"driver nodes"*. Controlar uma rede significa levá-la de um estado inicial conhecido para um determinado estado final; o papel dos *"driver nodes"* é imprescindível pois são eles que têm a capacidade de alcançar todos os nós e assim movê-los de estado para outro. Neste contexto controlar uma rede significa levar a rede de um estado inicial para um estado final em um determinado tempo finito.

O controle de redes é um problema que envolve não só o mercado econômico, como citado anteriormente, mas também problemas envolvendo redes biológicas, redes sociais, redes de transporte (aéreo, rodoviário, ferroviário), redes elétricas entre outros. Na rede elétrica por exemplo, conhecendo-se os *"driver nodes"*, os projetistas podem inferir quais são os postes ou terminais que se apresentarem falhas prejudicarão o funcionamento da rede.

Portanto, com base nos resultados obtidos e relatados nos artigos de Liu et al. (2011) Glattfelder e Batiston (2009), este trabalho procura aprofundar o estudo sobre sistemas complexos e abri novos caminhos para melhor compreender sistemas complexos, abrangendo seus mecanismos e buscando aplicar ferramentas que possam ser aplicadas em redes o mais generalizadas possível.

1.2 MOTIVAÇÃO

"O HOMEM É FEITO DE TAL MANEIRA QUE QUANDO ALGO INCENDEIA SUA ALMA, AS IMPOSSIBILIDADES DESAPARECEM." (Jean de la Fontaine)

A motivação surgiu da curiosidade em relação a mercados financeiros, como se dá a queda ou valorização de uma ação? Quais fatores determinam esse fenômeno? É possível prognosticar o resultado?

Dessa curiosidade surgiu o desejo de se estudar mais a fundo o tema, e o assunto tomou proporções maiores, pois os fatores que determinam a oscilação da bolsa de valores não estão ligados apenas ao campo da economia. Questões políticas e sociais também influenciam no valor de uma determinada ação.

A crise mundial, iniciada em 2007 (BRESSER-PEREIRA, 2010), é um claro exem-

plo de como problemas em um país, no caso, Estados Unidos desencadearam problemas em vários outros países que mantinham relação com os estadunidenses. Países emergentes como China, Brasil e Índia não sofreram tanto as consequências dessa crise como os países da União Europeia e NAFTA.

Assim, qual seria (se é que existe) a lógica desse processo? Quais países têm mais "poder" (controle) sobre outros?

1.3 JUSTIFICATIVA

O trabalho justifica-se pela sua aplicação em grafos de sistemas reais, mas não só por isso, visa também contribuir com a avaliação de um algoritmo que possa ser computável(possível ser calculado), além de tratável(possível ser processado por computadores), ou seja, que tenha complexidade algorítmica próxima da linear e possibilite o estudo para redes maiores, com centenas de milhares ou milhões de nós envolvidos.

Haverá uma comparação entre os algoritmos ótimos e os gulosos, de maneira a analisar a relação entre desempenho e qualidade dos resultados (os algoritmos gulosos costumam ter melhor desempenho em processamento, porém não garantem encontrar sempre o resultado ótimo).

Redes complexas representam fenômenos que podem ser sociais, biológicos, econômicos, entre outros. No exemplo de um fenômeno econômico, como a relação entre a tendência de queda ou subida de ações em Bolsas de Valores, entender que fatores levam a esses acontecimentos, "quais nós"(neste contexto representam empresas) (Bolsas de Valores - BOVESPA ou NASDAQ, por exemplo) influenciam mais (controlam) o fenômeno, é de extrema importância.

Definir vértices controladores pode, entre outras muitas aplicações, ajudar os cientistas a entenderem fenômenos muito complexos como as crises, ou a interação entre países.

1.4 OBJETIVOS

1.4.1 OBJETIVO GERAL

Explorar um algoritmo capaz de, em tempo próximo ao linear, identificar quais nós teriam a capacidade de controlar a rede, ou seja, levá-la de um estado inicial para um estado final desejado.

1.4.2 OBJETIVOS ESPECÍFICOS

- Discutir a aplicação dos conceitos de Teoria dos Grafos a diferentes domínios (e.g. Economia, Biologia, redes sociais).
- Comparar algoritmos de emparelhamento, considerando cenários representados por distintos tipos de redes complexas.
- Avaliar o desempenho temporal e o grau de aproximação dos resultados obtidos em relação a valores ótimos.

1.5 ESTRUTURA DO DOCUMENTO

A dissertação está dividida em três partes. A primeira, composta pelos capítulos 2,3,4 compreende a fundamentação teórica e os conceitos necessários para o desenvolvimento do trabalho. Na segunda parte, composta pelos capítulos 5, 6, 7 e 8, são apresentados os métodos, resultados obtidos e conclusões. A terceira e última parte compreende os apêndices que englobam informações sobre a gerência de projeto (plano de projeto, procedimentos de teste e validação, planejamento e análise de riscos.)

No capítulo 2 é abordado o tema Teoria dos Grafos, são apresentadas uma breve introdução histórica, e as formas de classificação e representação dos grafos.

No capítulo 3 são comentados alguns tópicos importantes sobre redes complexas, suas propriedades e modelos.

No capítulo 4 é apresentado o conceito de controlabilidade e como se dá a sua aplicação em redes complexas. Há neste capítulo o motivo pelo qual o método de cálculo tradicional(utilizados na Teoria Clássica de controle) foi substituído pelo método do emparelhamento de grafos. Também é discutido neste capítulo o conceito de emparelhamento e os principais algoritmos de emparelhamento conhecidos na literatura.

A segunda parte é iniciada com o capítulo 5, que apresenta o método utilizado; como o trabalho foi desenvolvido. O capítulo 6 apresenta os algoritmos utilizados e como foram realizados os testes. No capítulo 7 são explicitados os resultados obtidos e a análise dos mesmos. O capítulo 8 engloba as conclusões finais e trabalhos futuros.

Os apêndices foram desenvolvidos no decorrer das disciplinas de TCC 1 e TCC 2 e apresentam os conceitos de gerencia de projeto.

2 TEORIA DOS GRAFOS

A fundamentação teórica deste trabalho toma como base algumas características de grafos, as quais apresentarei na revisão bibliográfica.

A teoria dos grafos dedica-se a estudar as características dos diferentes tipos de grafos, que podem representar os mais diferentes sistemas, desde rotas de voos aéreos até indivíduos e suas interações, sistemas biológicos entres outros.

2.1 GRAFOS

Definição 2.1 *Um **grafo** é dado por um par G = (V,E), em que V [1] representa um conjunto arbitrário de vértices, e E são subconjuntos de V conhecidos como arestas.*

As arestas a,b, por exemplo, serão denotadas como ab, ou ba. Isso significa que a aresta incide em a e b, e que sendo assim, a e b são as extremidades da aresta. Contudo, se ab é uma aresta, então os vértices a e b são ditos **vizinhos** ou **adjacentes**.

Grafos são representados visualmente por pontos (vértices) que se interligam com linhas (arestas). Essas arestas, por sua vez, podem ser valoradas, bilaterais ou unilaterais dependendo do caso em estudo. A utilização de grafos abrange as mais diferentes áreas na Sociologia, por exemplo, os grafos representam redes sociais que refletem a interação entre indivíduos (COLNAGO, 2012) (FEOFILOFF et al., 2011).

As denominações diferem de acordo com a área do estudo, para Matemática são arestas e vértices, para sociologia são ator e relações; e para Computação são nó e ligação (ou links) (ADAMIC, 2008b). Como se trata de um projeto que envolve conceitos de Matemática e Computação serão utilizadas as denominações de vértices e arestas; ou nós e links (ligações) no decorrer do texto.

[1] No decorrer do texto os conjunto serão representados por letras maiúsculas, enquanto os elementos do conjuntos serão representado com letra minúscula e itálico. Exemplo: V = conjunto de vértices; *v* - um vértice contido no conjunto V

Definição 2.2 *Um **grafo ponderado** G é uma relação tripla G=(V, E , p), em que V e E seguem a Definição 2.1. adicionando-se o atributo de peso p para uma ligação.*

Esse peso é usado para definir a importância da aresta no grafo. Por exemplo, em um grafo que representa a relação entre duas pessoas, as triplas (A, B, 20) e (B, A, 10), poderiam representar um relacionamento direcionado no qual A gosta mais de B, do que B gosta de A. (BESSA et al., 2010)

Para facilitar o entendimento, será convencionado que V(G) representa o conjunto de vértices do grafo G, E(G) representa o conjunto de arestas do grafo G e p(G), a função peso do mesmo grafo G.

Seguem outras definições importantes:

Definição 2.3 ***Ordem*** *representada por $|V|$, é o número de vértices de G;*

Definição 2.4 ***Dimensão*** *representada por $|E|$, é o número de arestas de G*

Definição 2.5 *O **grau** de um vértice v representa o número de arestas que incidem em v, será denotado por $g_G(v)$*

2.1.1 REPRESENTAÇÕES

As três formas mais usuais de se representar um grafo são: graficamente, matriz de adjacências e lista de adjacências.

A representação gráfica é dada da seguinte forma:

- Para cada nó é desenhado um ponto. As arestas são representadas por um segmento de curva que liga dois pontos. Caso o grafo seja ponderado, o peso é colocado próximo à aresta correspondente (Figura 1).

A representação de matriz de adjacência é feita da seguinte forma:

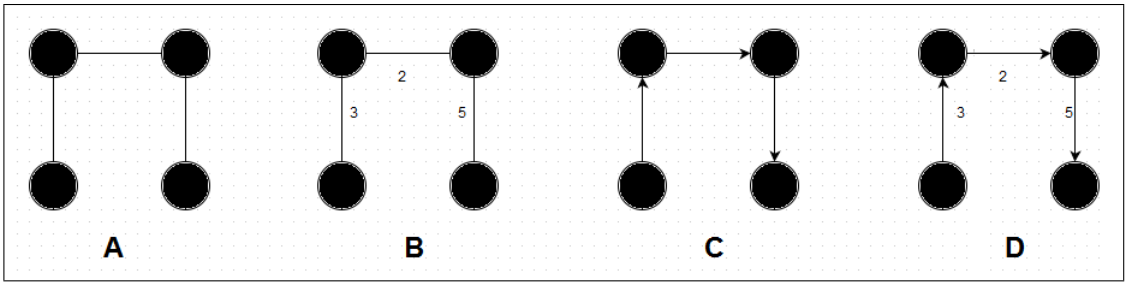

Figura 1: *Representação gráfica de um grafo A) não direcionado. B) não direcionado e ponderado. C) direcionado. D) não direcionado e ponderado*

Dada uma matriz:

$$A(i,j) = \begin{cases} p_G & \text{se } i,j \in E(G) \\ 0 & \text{caso contrário.} \end{cases} \quad (1)$$

Caso a matriz não seja ponderada, coloca-se 1 ao invés do peso p_G.

$$A = \begin{matrix} 0 & 1 & 0 \\ 8 & 0 & 3 \\ 5 & 0 & 0 \end{matrix}$$

A matriz acima indica as seguintes ligações entre os nós:

- Nó 1 ligado ao nó 2 com peso 1;
- Nó 2 ligado ao nó 1 com peso 8;
- Nó 2 ligado ao nó 3 com peso 3;
- Nó 3 ligado ao nó 1 com peso 5.

A Figura 2 retrata como seria a representação gráfica do grafo representado na matriz A.

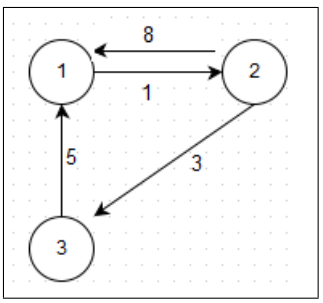

Figura 2: *Representação do grafo A apresentado na forma de matriz de adjacência pela equação 2*

2.1.2 CLASSIFICAÇÃO

Existem várias alternativas para classificar grafos, abaixo seguem as mais relevantes para este trabalho.

Definição 2.6 *Um grafo é dito **vazio** se o E(G) = 0*

Isso significa que ele não possui nenhuma ligação e portanto, existem apenas pontos representados sem nenhuma relação entre eles. (COLNAGO, 2012)

Definição 2.7 *Grafo **completo**, representado por A(G), é aquele onde* $|E(G)| = |V(G)|^2$

Isso representa que todos os vértices estão ligados e, não há mais a possibilidade de se criar nenhuma ligação sem a inclusão de um novo vértice.

Definição 2.8 *O **complemento** de um grafo G é dado por* $\overline{G}$*. Representa que todas as arestas que não foram utilizadas em G serão utilizados em* $\overline{G}$ *e vice e versa. (FEOFILOFF et al., 2011)*

Definição 2.9 *Um grafo é conexo se, e somente se, para quaisquer dois vértices distintos, sempre há um caminho que os conecte. (BESSA et al., 2010)*

Exemplo:

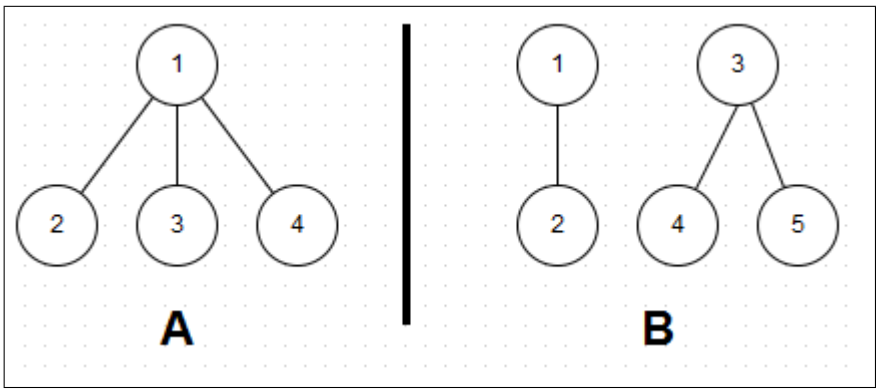

Figura 3: *A) Grafo Conexo. B) Grafo Desconexo*

Na Figura 5 no caso do grafo B, não há um caminho que conecte o vértice 1 ao 5 por exemplo, logo esse grafo é dito desconexo.

Definição 2.10 *Seja G =(V,E) um grafo. Esse grafo é dito bipartido se o conjunto de vértices V de G puder ser divididos em dois grupos V1 e V2 (não vazios), tal que toda a aresta de G tenha uma extremidade em V1 e outra em V2.*

É importante observar que a definição acima esclarece que, em um grafo bipartido, uma dada aresta *e* tem de ter um vértice em V1 e outro em V2. A definição não estabelece que entre dois vértices de grupos diferentes precise necessariamente haver uma aresta ligando-os.

Um grafo bipartido completo é um simples grafo bipartido em que todos os vértices V1 possuem ligações com algum vértice em V2, e vice-versa. Ou seja, não pode sobrar vértices sem conexões em nenhum dos dois grupos.

As propriedades ajudam no estudo e caracterização do grafo. Algumas das propriedades mais importantes de grafos estão listadas abaixo.

A medida do grau auxilia na detecção do **vértice central**, que é o vértice que possui mais ligações (ADAMIC, 2008b).

Definição 2.11 *seja Z(E) a soma dos graus de todos os vértices de um grafo, e sendo* $|V|$ *o número de vértices do grafo, logo, o grau médio do grafo é* $Z(E)/|V|$.

Há também uma diferenciação importante para grafos direcionados entre grau de saída (número ligações que se originam de um determinado vértice) e grau de chegada (ligações que chegam a um determinado vértice).

No decorrer do texto quando não for especificado se é grau de saída ou chegada subentende-se que se esteja falando do grau (grau de saída + grau de chegada) do vértice

Definição 2.12 *A medida de* ***densidade*** $\Delta(G)$ *está relacionada ao grau, um grafo denso é um grafo que possui densidade próxima de 1, este parâmetro é calculado dividindo-se as ligações que existem pelo o número de ligações possíveis em um grafo.*

Por exemplo:

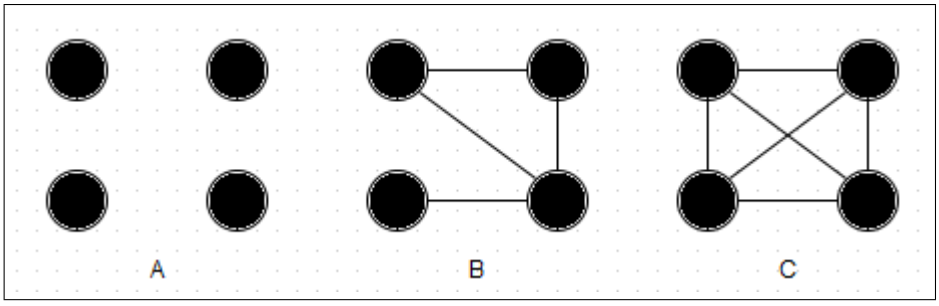

Figura 4: *A) Grafo A . B) Grafo B. C) Grafo C*

A) Número de arestas = 0

Número máximo de arestas = 6

Densidade $\Delta(A)$ **= 0**

B) Número de Arestas = 4

Número máximo de Arestas = 6

Densidade $\Delta(B)$ **= 4/6 = 0,67**

C) Número de Arestas = 6

Número máximo de Arestas = 6

Densidade $\Delta(C)$ **= 6/6 = 1**

Definição 2.13 *O **diâmetro** é a medida do maior caminho no grafo*

Em um grafo ponderado, o valor do peso da aresta influi no diâmetro. Já em um grafo não ponderado, o diâmetro é medido em relação aos saltos (hops) que um vértice precisa dar para alcançar o outro vértice mais distante no grafo.

Definição 2.14 *Dois grafos, G e H, serão **isomorfos** se eles possuírem o E(G) e E(H) iguais, ou seja, se for possível alterar o nome dos vértices de um deles de maneira que os dois fiquem exatamente iguais. (BESSA et al., 2010)*

3 REDES COMPLEXAS

Segundo Barabási (2003), o termo redes complexas refere-se a um grafo que apresenta uma estrutura topográfica não trivial composto por um conjunto de vértices que são interligados por meio de arestas.

Trata-se uma área recente de estudo, que envolve o formalismo matemático da Teoria dos Grafos com uma análise da estatística baseada em conceitos, tais como invariância de escala e estudo de modelos. Juntamente a isto, a evolução dos computadores (poder de processamento) auxiliou muito no crescimento da complexidade dos modelos de sistemas complexos, que foram criados pela ciência (NEWMAN, 2003).

O início do uso de grafos para representação de problemas é creditado ao matemático suíço, Leonhard Euler, que, em 1736, resolveu o Problema das Pontes de Konigsberg. O problema tratava da perspectiva de atravessar as sete pontes da cidade sem passar duas vezes por nenhuma delas (ALVES-JR, 2008).

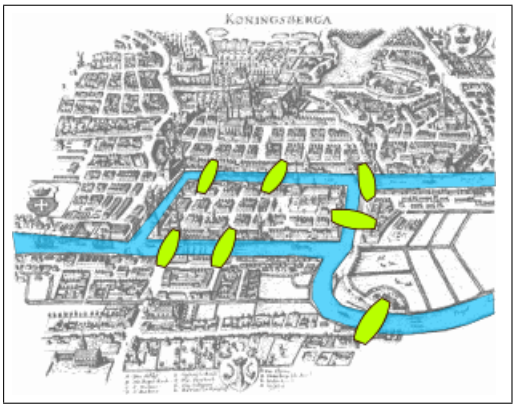

Figura 5: *Representação Gráfica das Pontes de Konigsberg*

Para a resolução do problema, Euler construiu um modelo simplificado da cidade (Figura 5) , em que os nós representavam os bairros e as ligações representavam as pontes. Com isto, o matemático concluiu que só seria possível fazer o caminho completo atravessando apenas uma vez cada ponte, se todos os bairros tivessem números pares de pontes, ou se apenas dois bairros tivessem números ímpares. Dessa forma, Euler provou que o problema era impossível

de ser resolvido com a configuração similar à apresentada na Figura 6 (BESSA et al., 2010) (NEWMAN, 2003).

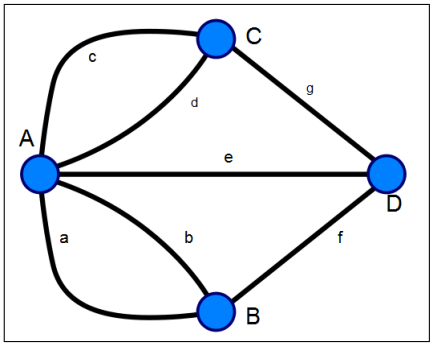

Figura 6: *Exemplo de grafo para as pontes de Konigsberg*

Euler se preocupou então em definir a que tipos de grafos esse conceito de caminho fechado, passando apenas uma vez por cada aresta poderia ser aplicado. Esse tipo de caminho ficou conhecido como caminho euleriano e os grafos que permitem esse percusso são conhecidos com grafos Eulerianos.

O resultado de sua pesquisa mostrou que para um grafo ser euleriano, ou ele deve possuir todos os vértices com grau (número de arestas que um vértice possui) par, ou possuir dois (nem mais, nem menos) vértices com grau ímpar (ARAúJO, 2001).

Desse problema se originaram vários outros, e estudiosos ajudaram a desenvolver o estudo da Teoria dos Grafos; entre eles, cabe ressaltar alguns de maior destaque - Kirchhoff (1847) com a Teoria das Árvores; Guthrie, em 1852, com a conjectura das quatro cores , que mais tarde permitiu o surgimento da coloração em grafos que se conhece atualmente (a resolução desse problema é a de que são necessárias quatro cores para se pintar os nós de um grafo sem que nenhum nó vizinho tenha a mesma cor); e Rowan Hamilton, em 1859, inventou um jogo: um desafio em que em um dodecaedro regular, fossem percorridos todos os vértices, passando uma vez por cada um. Inspirados em seu trabalho surgiram os conceitos de ciclo hamiltoniano. Entende-se por ciclo hamiltoniano um caminho em um grafo em que cada vértice é visitado apenas uma vez e o percurso começa e termina no mesmo ponto (NEWMAN, 2003).

A Teoria de Redes complexas é amplamente utilizada em modelagem e caracterização de Sistemas Complexos. A modelagem é a redução de uma realidade física, limitada pela heurística do modelador, que focará em características mais determinantes para o fenômeno que deseja estudar. A modelagem deverá garantir confiabilidade e operacionalidade ao modelo apresentado. (BESSA et al., 2010)

3.1 PROPRIEDADES DE REDES COMPLEXAS

As Redes possuem características que são importantes para a análise dos aspectos do estudo em questão. Mensurar estas características, portanto, é crucial em um sistema formal de estudo.

3.1.1 ORDEM E TAMANHO

Definição 3.1 *Dado um grado G(V,E) o tamanho é dado pela cardinalidade do conjunto de ligações E. A ordem é dada pelo número de vértices V presentes do grafo.*

3.1.2 COEFICIENTE DE CLUSTERIZAÇÃO DE UM VÉRTICE, COEFICIENTE DE MÉDIO DE CLUSTERIZAÇÃO

Definição 3.2 *Dado um vértice v, o seu coeficiente de clusterização é a probabilidade que os vértices conectados a v, sejam conectados entre si.*

O cálculo do coeficiente de clusterização é dado por:

$$C_i = \frac{2n_i}{K_i(K_i - 1)} \tag{2}$$

Sendo n_i o número de arestas ligadas ao vértice e K_i, o grau do vértice

Já coeficiente de clusterização médio do grafo é dado pela média aritmética dos coeficientes de clusterização de cada nó.

$$C_i médio = \frac{\sum_{k=1}^{|} V|C_i}{|V|} \tag{3}$$

$|V|$ = Número de vértices no Grafo

3.1.3 ROBUSTEZ

Define a resistência da rede em relação aos vértices que podem ser retirados sem que haja perda da funcionalidade da rede. Esta propriedade está fortemente relacionada com o grau médio da rede, pois a remoção de um nó pode tornar um grafo desconexo, ou aumentar significativamente o caminho de um nó a outro.

3.2 MODELOS DE GRAFOS PARA REDES COMPLEXAS

Redes complexas utilizam-se do formalismo dos grafos, acrescentando métodos e medidas em sistemas reais (NEWMAN, 2003). As relações entre os componentes (vértices e arestas) do grafo não seguem nenhum padrão específico, podendo "gerar" tanto grafos totalmente aleatórios, como grafos que seguem uma estrutura bem regular (todos os nós com mesmo grau, por exemplo). A análise de apenas um componente não levaria a nenhuma conclusão sobre o todo, analisar um ser humano, por exemplo; não permite que se conheça toda a sociedade em que ele vive. A partir desse argumento vem a necessidade de se desenvolverem modelos de redes para estudar as relações, graus e outras métricas do grafo, e não apenas de um indivíduo. (RODRIGUES, 2007)

Os modelos definidos apresentam características bem determinadas e topologias bem distintas. O primeiro modelo de redes complexas foi o modelo de Paul Erdos e Alfréd Reyni publicado no artigo *"On Random Graphs"*, em 1959, que ficou conhecido como modelo Erdos-Renyi, ou modelo de grafos aleatórios.

Mais tarde, novos modelos foram desenvolvidos e, com maior destaque, surgiram os modelos de Watts e Strogatz *"Collective dynamics of small-world networks"*, em 1998, conhecido como modelo de mundo pequeno e o modelo de livre escala de Barabási e Albert, publicado no artigo *"Emergence of scaling in random network"*, em 1999.

3.2.1 GRAFOS ALEATORIOS

Foi um modelo proposto por Erdos e Renyi, em 1959. São grafos construídos com n vértices e probabilidade p de que os vértices se liguem, isto é, para quaisquer dois vértices a probabilidade de eles possuírem, ou não, ligação é p (SANTANA, 2007). O número máximo de arestas, *maxE*, possíveis é dado por:

$$maxE = \frac{|V|(|V|-1)}{2} \tag{4}$$

, em que $|V|$ representa o número de vértices presentes no grafo.

A probabilidade p de uma aresta aparecer é dada por :

$$m = p\frac{|V|(|V|-1)}{2} \tag{5}$$

E a probabilidade de um grafo $G_{n,p}$ ser formado é dada por:

$$P(G_{n,p}) = p^m(1-p)^{M-m} \tag{6}$$

O grau médio de um vértice é dado por $p(n-1)$, e segue a distribuição de Poisson; esses grafos apresentam grau de clusterização dependente da probabilidade p; dessa forma qualquer que seja o número de $|V|$, este não terá influência sobre o grau de aglomeração do grafo (LOPES, 2011) (NEWMAN, 2003).

3.2.2 REDES DE MUNDO PEQUENO

As redes de mundo pequeno ou *Small worlds* foram proposta por Watts e Strogatz em 1998. O modelo surgiu como opção para o modelo aleatório. O diferencial da abordagem de mundo pequeno é a suposição de que redes biológicas, sociais e tecnológicas não têm comportamento totalmente aleatório.

O nome Mundo Pequeno deve-se ao experimento feito por Stanley Milgram, em 1960, no qual cerca de 160 cartas deveriam ser entregues por famílias de Nebraska e Kansas às pessoas em Boston, utilizando apenas a intermediação de amigos.

Foram definidas regras para o experimento:

- Os envelopes tinham nome, endereço e alguns dados pessoais do destinatário. Caso o remetente não conhecesse o destinatário, deveria passar o envelope para um amigo seu (do remetente) que possivelmente poderia conhecer a pessoa alvo.

- Cada pessoa que recebia o envelope tinha de colocar seu nome nele, para evitar que a carta passasse duas vezes pela mão da mesma pessoa.

Os pesquisadores inicialmente esperavam que as cartas fosse entregues a seus destinos com cem passos aproximadamente. Mas para surpresa dos cientistas, ao fim do experimento, cerca de 20% dos envelopes chegaram a seus destinos e com um caminho médio de tamanho 6.5 (ADAMIC, 2008a) (METZ J.; CALVO et al., 2007).

Com os resultados Milgram e seu grupo puderam deduzir o conceito conhecido como **seis graus de separação**; o que define que quaisquer duas pessoas podem se "comunicar" por intermédio de em média seis amigos, ou seja, mesmo que duas pessoas no mundo não se conheçam, é muito provável que tenham um conhecido em comum (ALVES-JR, 2008).

O resultado dos estudos de Watts e Strogatz foi o algoritmo de redes de mundo pequeno, que pode gerar tanto grafos aleatórios como grafos regulares. As redes de mundo pequeno têm o grau de clusterização (coeficiente de clusterização) maior e o caminho médio

menos se comparados às redes aleatórias com o mesmo número de vértices e arestas (BESSA et al., 2010).

Redes de mundo pequeno, como mostrado na figura abaixo, podem ser geradas a partir de redes regulares (redes nas quais todos os vértices têm mesmo grau), retirando-se as conexões e colocando-as entre outros nós do grafo. De maneira similar, se for admitido que a probabilidade p de se retirar, ou recolocar uma aresta entre os vértices do grafo seja p =0 para grafos regulares, essa probabilidade valerá p = 1 para grafos aleatórios. Com valores intermediários de p podem-se obter as redes de mundo pequeno (NEWMAN, 2003).

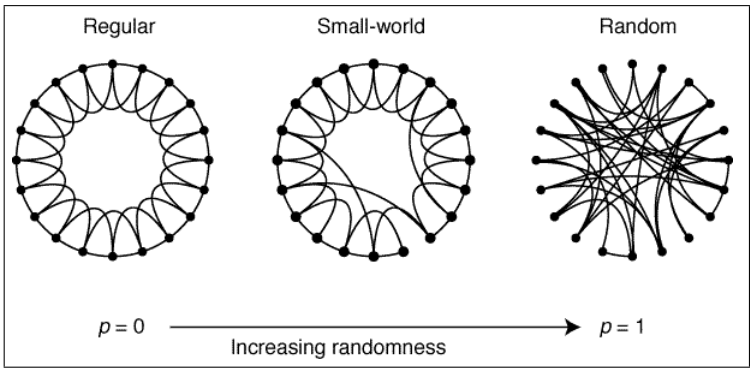

Figura 7: *Grafos de Mundo Pequeno*

Fonte: *Collective dynamics of 'small-world' networks*, WATTS e STROGRATZ

As implicações de mundo pequeno na dinâmica das relações em redes facilmente entendidas, por exemplo, uma informação, ou "doença"podem se espalhar mais rapidamente (até seis hops), do que em outras topologias de rede.

3.2.3 GRAFOS DE ESCALA LIVRE

Watts e Strogatz em seus estudos sobre redes de mundo pequeno, adotaram a distribuição de probabilidade conhecida como *distribuição normal*, pois questionavam a criação de redes de mundo pequeno sem que fossem considerados *hubs*(nós que possuem grau bem acima do grau médio do grafo). (GIMÉNEZ-LUGO, 2007)

Barabási e Albert no artigo *"Emergence of scaling in random network"*, publicado em 1999, mostraram que a lei de potência [1] (power law) rege o grau de conectividade dos nós em redes reais como a internet, por exemplo. (GIMÉNEZ-LUGO, 2007)

[1] a equação que rege a lei de potência é dada por $P(k)\ K^{-y}$, onde K é o grau de conectividade ou número de conexões e o y é uma constante.

A Figura 8 mostra a diferença entre distribuição normal e lei de potência; fica nítido que há um valor na distribuição normal que representa um corte na função. Esse valor define uma escala para distribuição, ao contrário do que acontece com a lei de potência, que decai lentamente.

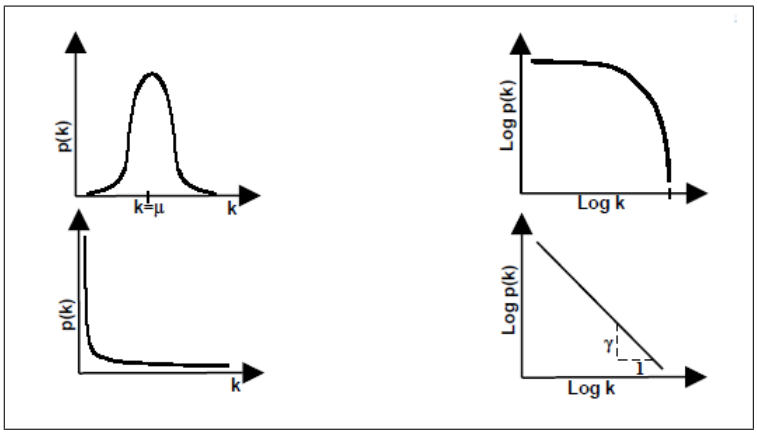

Figura 8: *Comparação entre Lei de Potência e Distribuição Normal, em escala normal(esq) e logaritmica (dir)*

Fonte: (GIMÉNEZ-LUGO, 2007)

A rede de livre escala é construída com a adição progressiva de elementos (vértices) à rede já existente, as conexões entre novos elementos como elementos pré existentes é dada por:

$$P(K_i) = frack_i \sum_{j=1}^{N} k_j \tag{7}$$

em que K_i é o número de conexões do iésimo nó e $|V|$ é o total de vértices da rede. (ALVES-JR, 2008)

O modelo visa à criação de redes, a partir da conexão com nós preferenciais (hubs) que terão um grau muito alto de conectividade, ou seja, nessa rede, quanto mais conexões um nó produz mais provável é que ele receba mais nós (nós ricos tendem a ficar mais ricos). (ADAMIC, 2008a)

A remoção de um nó de alto grau na rede caracteriza um processo de ataque. Já a falha é a remoção aleatória de nós em uma rede. Um assunto largamente discutido em redes de livre escala é a tolerância a falhas. Isso implica que a remoção aleatória de nós tende a remover nós com baixa conectividade (pois eles são maioria), não afetando grande parte da rede. Por outro lado, esse tipo de rede é extremamente sensível à ataques. (BESSA et al., 2010)

4 CONTROLABILIDADE E USO DE ALGORITMOS DE EMPARELHAMENTO

São inúmeros os exemplos de redes que podem ser encontrados em torno de todos nós, como exemplos podem ser citadas redes sociais, cadeias alimentares, citações de trabalhos acadêmicos, circuitos elétricos entre outros. Os indivíduos são representados pelos vértices, enquanto as arestas entres eles podem representar a informação que flui de um vértice para o outro.

Tendo esta situação, os cientistas e estudiosos da área começaram a questionar se é possível "controlar"o comportamento desta rede, e uma vez que o seja, como fazê-lo. (LIU et al., 2011), discutem em seu artigo que o fluxo de informação em uma rede é o que permite aos vértices atualizarem seus estados internos. O ponto central da discussão seria de quais fatores dependem o comportamento da rede, ou seja, como a informação é compartilhada, e como os vértices recebem esta informação e atualizam seus "estados".

Para ilustrar melhor o que é a controlabilidade é preciso imaginar que um indivíduo queira influenciar o comportamento de outros indivíduos em uma rede social. Ou seja, uma ideia será propagada na rede com o intuito que o maior número possível de indivíduos adiram a esta nova ideia(comportamento).

Quais vértices devem ser escolhidos? Qual é o "poder"destes vértices para alcançar objetivo de disseminar a ideia? Liu et al, (2011) combinaram a teoria clássia de controle com as teorias de Ciência de Redes, e chegaram ao resultado de que nem sempre os vértices centrais (com mais ligações) são os que mais influenciam sobre o controle da rede, isto significa que em uma rede social as pessoas com mais conhecidos não são necessariamente os principais responsáveis por controlar o comportamento daquele grupo.

Em seu artigo "Networks dominated by rule of the few", Rachel Ehrenberg, retrata esta situação como um filme de suspense de Hollywood, no qual algumas pessoas "sombrias"podem controlar milhões de mentes. A principal contribuição de Lui et al;(LIU 2011) está em definir um algoritmo que consegue, a partir da arquitetura da rede, determinar quantos nós são necessários para controlar todo o sistema. (EHRENBERG; MARTINO,)

O resultado mostra, por exemplo, que redes que representam genes necessitam de cerca de 80% dos nós para atingir o controle geral, enquanto redes sociais (característica de rede mais densas) não necessitam de mais que 20% (em média) para atingir este controle “global”.

A teoria sobre controle de redes, é uma ciência em formação e alguns cientistas ainda tem desconfiança se este algoritmo e estes cálculos realmente se aplicam às redes. Mas de qualquer forma o avanço na área pode melhorar o entendimento sobre a dinâmica da rede e permitir que os pesquisadores possam determinar por exemplo, a vulnerabilidade ou robustez das redes, podendo assim no caso de redes elétricas, ou de fluxo de informação eliminar os pontos mais vulneráveis a ataques e tornar a rede menos suscetível a falhas.

De acordo com os estudos desenvolvidos por Kalman (1963) e Luenberger (1979) sobre teoria clássica de controle, um sistema pode ser dito controlável se com escolhas adequadas de entrada, for possível alcançar qualquer estado final desejado em um determinado tempo finito. A teoria de controle é desenvolvida por engenheiros, com aplicações em circuitos elétricos, processo de manufatura, controle de robôs entre outros.

A dificuldade de se controlar um sistema está ligada a dois fatores independentes que contribuem para a controlabilidade, o primeiro deles é a arquitetura do sistema, representada pela rede encapsula as interações entre os componentes, e a segunda são as regras dinâmicas que determinam as interações entre os componentes. Assim em redes complexas o controle se torna bastante complicado, e a ciência ainda não possui todas as respostas quando se trata de redes grandes, dirigidas e com peso entres as arestas. (LIU et al., 2011)

Segundo (PEREIRA; HAFFNER, 2013):

> A descrição de um determinado sistema de controle na forma de espaço de estados, pressupõe uma fase preliminar de modelagem, que nada mais é do que a descrição em linguagem matemática do conjunto de fenômenos físicos que estabelecem o comportamento do processo como um todo.

Um sistema linear invariante no tempo é controlável se, dado um conjunto de estados iniciais (representados por $x(t_0)$, estes estados puderem ser transferidos para qualquer conjunto final de estado $x(t_f)$, em um intervalo finito de tempo.

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = Ax(t) + Bu(t) \tag{8}$$

A é uma matriz de adjacência N x N (N neste contexto representa o número de vértices no grafo)e representa um sistema de interações fortes entre os componentes (representa as

ligações presentes no grafo), como uma comunicação individual entres dois nós. B é uma matriz N x M (com N ¡ M) que representa a matriz de sinais de entrada, x(t) representa um estado inicial e u(t) representa o vetor de controle. Esta abordagem é proposta por YANG-YU et al. (2011) porém, a partir das matrizes A e B é é definida a matriz de controlabilidade $C = (B,AB,A^2B, \ldots ,A^{(n-1)B})$ (LIU et al., 2011). Então é calculado o Rank (posto) de C, se o valor do posto for igual a N o sistema é controlável, caso seja menor N então o sistema é incontrolável (LUENBERGER, 1979).

Abaixo segue um exemplo para ilustrar o método desenvolvido na Teoria de Controle Estrutural.

Os exemplos foram adaptados do material de textos suplementares do artigo (LIU et al., 2011).

Dado o grafo apresentado na Figura Abaixo:

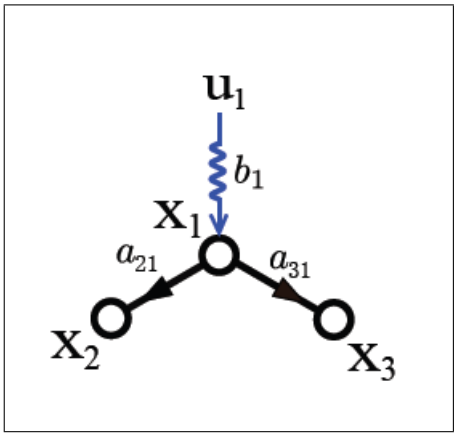

Figura 9: Grafo em formato Estrela

Este sistema pode ser escrito como:

$$\begin{bmatrix} \acute{x_1}(t) \\ \acute{x_2}(t) \\ \acute{x_3}(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ a_{21} & 0 & 0 \\ 0 & a_{32} & 0 \end{bmatrix} * \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} b_1 \\ 0 \\ 0 \end{bmatrix} u(t) \tag{9}$$

A matriz de controlabilidade é dada por:

$$C = [B,A*B,A^2*B] = b_1 \begin{bmatrix} 1 & 0 & 0 \\ 0 & a_{21} & 0 \\ 0 & 0 & a_{32}a_{21} \end{bmatrix} \tag{10}$$

O posto desta matriz é 3 (número de linhas linearmente independentes), logo o posto é

igual a N (número de vértices), então o sistema é dito controlável.

Anteriormente ao trabalho apresentado por (LIU et al., 2011) os teoremas de controlabilidade só eram aplicados a redes não direcionadas, o que era limitado já que a maioria das redes complexas tem a característica de serem direcionadas. Outra limitação anterior ao trabalho em questão é que a complexidade dos algoritmos existentes até então, não permitiam que que fossem calculados os "drivers nodes"para redes muito grandes.

Uma técnica utilizada como opção à teoria clássica (proibitiva), é encontrar matematicamente o número mínimo de nós,porém esta técnica possui complexidade computacional de $O(2^n)$ tornando assim esta técnica proibitiva para redes com algumas centenas de nós. Surgem assim como opção os algoritmos de Emparelhamento de grafos (serão estudados mais afundo no Capítulo 5), que possuem complexidade próxima a $O(e\sqrt{v})$, onde v representa o número de vértices e e o número de arestas.

Cabe ressaltar que para adaptar o emparelhamento à sua solução Liu et al. (2011) faz uma pequena mudança no conceito, para Liu et al. (2011) quando uma aresta é selecionada no emparelhamento apenas o nó incidente (nó final da aresta) é marcado como emparelhado. Na definição formal os dois nós seriam marcados.

O emparelhamento de grafos é um método com complexidade menor que os anteriores, contudo o intuito desta dissertação é desenvolver um algoritmo guloso que possa gerar resultados satisfatórios, com uma complexidade computacional menor que os algoritmos de emparelhamento de grafos. Para melhor entendimento do problema o próximo capítulo descreverá os principais algoritmos de emparelhamento existentes, e suas respectivas complexidades.

4.1 EMPARELHAMENTOS EM GRAFOS

4.1.1 CONCEITOS

Emparelhamento (ou *matching*) é um subconjunto de arestas pertencentes a um grafo,no qual cada aresta não tem nenhum vértice em comum com nenhuma outra aresta do subconjunto (WILSON; WATKINS,) (BONDY; MURTY, 2008) . Logo, todo grafo pode ser decomposto em emparelhamentos, sendo que um, por exemplo, um grafo G composto de **v** arestas pode claramente ter m emparelhamentos diferentes se cada um deles for constituído por apenas uma aresta.

Este problema é facilmente resolvido, porém o problema de se encontrar um emparelhamento máximo (emparelhamento que cubra todas as aresta) não é tão simples, e esta não é

uma questão apenas acadêmica, já que a aplicação de conceitos de grafo também é utilizada em larga escala por outros setores, como Economia e Ciências Biológicas, entre outras. (NICOLETTI, 2007)

Definição 4.1 *Seja G =(V,E) um grafo. O subconjunto $E \subset V$ é chamado emparelhamento em G se duas arestas quaisquer de E não forem adjacentes.*

Ou seja, segundo a Definição 4.1, um subconjunto E é emparelhamento de um grafo G quando quaisquer duas arestas de E não possuírem vértice em comum. A Figura 10 ilustra diferentes emparelhamentos para o mesmo grado. O conjunto de arestas pertencentes ao emparelhamento é representado pela cor amarela (NICOLETTI, 2007)

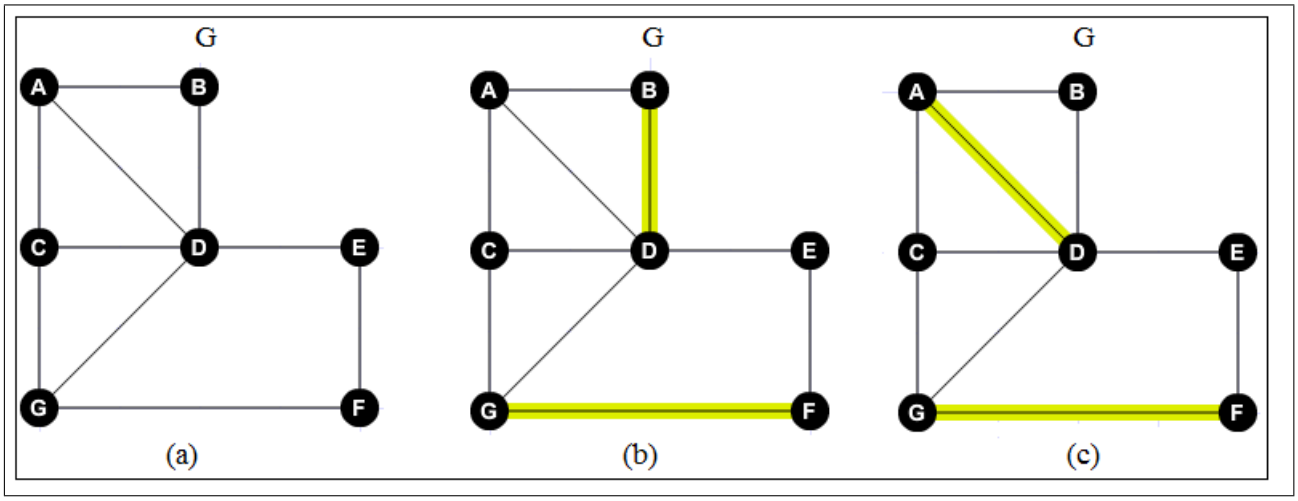

Figura 10: *(a) Grafo G; (b) Emparelhamento M_1; (c) Emparelhamento M_2*

Um emparelhamento é dito perfeito se ele possui todos os nós do grafo, sendo formalmente definido como:

Definição 4.2 *Seja G =(V,E) um grafo. M é um emparelhamento perfeito em G se, $\forall v \subset V$, v está contido em qualquer aresta pertencente ao conjunto M. (NICOLETTI, 2007)*

Emparelhamento máximo é o emparelhamento que utiliza o maior número de vértices possível é definido formalmente como:

Definição 4.3 *Seja G =(V,E) um grafo. M_{max} é um emparelhamento máximo se, não existir nenhum outro emparelhamento M* em G, tal que $|M| \subset |M_{max}|$.*

Emparelhamento maximal é um exemplo de emparelhamento que não pode ser aumentado com a adição de um vértice. (JUNGNICKEL; SCHADE, 2005)

Definição 4.4 *Seja G =(V,E) um grafo. M_{mal} é um emparelhamento maximal se não existir nenhum outro emparelhamento M*, tal que $|M*| > |M_{mal}|$.*

Um emparelhamento então, pode ser maximal se todo o vértice que não está em M é incidente em uma aresta pertencente a M. É possível concluir também que todo emparelhamento máximo é maximal, todavia nem todo maximal é máximo. (NICOLETTI, 2007)

A figura abaixo que exemplifica a diferença entre o conceito de emparelhamento máximo e emparelhamento maximal.

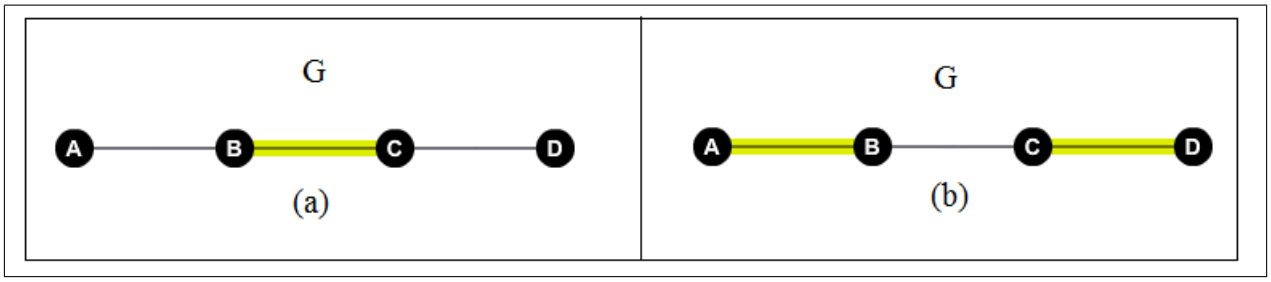

Figura 11: *(a) Emparelhamento maximal (b) Emparelhamento máximo*

Definição 4.5 *Seja G =(V,E) um grafo. M é um emparelhamento de G, então o caminho M-alternado é um caminho no qual as arestas alternadamente pertencem a M, e não pertencem a M (E - M).*

A Figura 11, que apresenta Emparelhamento maximal e máximo exemplifica bem o conceito de caminho alternado. (NICOLETTI, 2007)

Definição 4.6 *Seja G =(V,E) um grafo. M é um emparelhamento de G, então, o caminho M-aumentado é um caminho M-alternado no qual os nós de origem e fim não são nós saturados, ou seja, não são nós que fazem parte de nenhuma aresta presente em M.*

A figura abaixo que auxilia na descrição de caminho M-aumentado

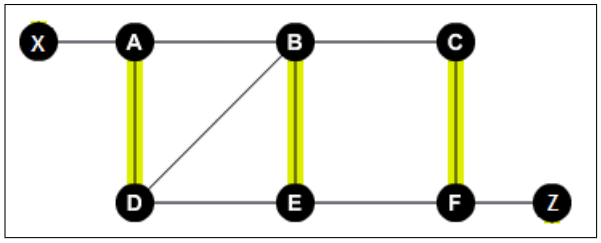

Figura 12: *Emparelhamento Aumentado - O emparelhamento e representado pelo caminho A-D-B-E-F*

4.1.2 APLICAÇÕES

Abaixo seguem exemplos de problemas que são resolvidos com emparelhamento em grafos.

Problema do Casamento

Trata-se de um problema em que se tem um conjunto finito de moças e rapazes divididos em dois grupos (um grupo só de moças e outro só de rapazes). Considerando que as moças conhecem vários rapazes, quais seriam as condições mínimas para que todas as moças se casem com rapazes que elas conheçam? (FIGUEIREDO; SZWARCFITER,)

Esse problema pode ser escrito na forma de grafos bipartidos, nos quais o conjunto de vértices V é tal que, $V = X \cup Y$, em que X representa o conjunto de moças e Y o conjunto de rapazes. O problema, desta forma, torna-se equivalente a encontrar um emparelhamento no grafo que sature todos os vértices de V.

A resposta deste problema foi obtida por meio do trabalho de Hall publicado em 1935. O Teorema de Hall define que:

- Dado um grafo bipartido, com $V = X \cup Y$. G terá um emparelhamento que satura todos os vértices em X, se e somente se: $(S)| \geq |S|$ para todo subconjunto S de V, onde N(S) representa o conjunto vizinhança de S em G. (BONDY; MURTY, 2008)

Ou seja, todas as k moças devem conhecer ao todo pelo menos k rapazes, satisfazendo assim $1 \leq K \leq n$, em que n representa o número total de moças.

Problema da Alocação de Funcionários

Uma empresa tem n vagas de empregos e tem n candidatos concorrendo a elas. É possível que cadas candidato ocupe apenas uma função na empresa e todas as vagas sejam preenchidas?

Posteriormente a isso a empresa definiu numericamente a eficiência de cada candidato para cada função. O objetivo agora não é apenas preencher as vagas, mas também, maximizar

o aproveitamento geral da empresa maximizando a eficiência dos funcionários nas funções a serem exercidas. (BONDY; MURTY, 2008)

O primeiro problema é similar ao problema do casamento, correspondendo apenas ao emparelhamento perfeito em grafos bipartidos. Já o segundo problema é o problema do emparelhamento máximo em grafos bipartidos com pesos.

Máximo Fluxo em Rede

Seja G=(V,E) um grafo direcionado e com peso, ou seja, a cada aresta (e) é associado um valor positivo que representa a capacidade da aresta (c(e)). O somatório dos fluxos das arestas divergentes a partir de uma aresta s, é chamado de fluxo em s. (BONDY; MURTY, 2008)

4.1.3 ALGORITMOS DE EMPARELHAMENTO

Abaixo seguem alguns dos principais algoritmos para emparelhamento máximo. Além da demonstração e do entendimento dos algoritmos, também serão discutidos aspectos relacionados à complexidade algorítmica.

Tabela 1: Os algoritmos de Emparelhamento mais eficientes.v = vértices, e = arestas, W é peso máximo e $SP+(v;e;W)$ é o tempo necessário para percorrer o menor caminho de um grafo direcionado. Esta tabela foi retirada do trabalho de (HUANG; KAVITHA, 2012)

Cardinalidade Máx em Grafos Bipartidos		**Cardinalidade Máx em Grafos Gerais**	
$O(\sqrt{v}e)$	(HOPCROFT; KARP, 1973) Karzanov (1973)	$O(\sqrt{v}e)$	(BLUM, 1990), (MICALI; VAZIRANI, 1980) (GABOW; TARJAN, 1991))
$O(\sqrt{v}e\log_v(\frac{v^2}{e}))$	(FEDER; MOTWANI, 1995) (GOLDBERG; KARZANOV, 2004)	$O(\sqrt{v}e\log_v(\frac{v^2}{e}))$	(GOLDBERG; KARZANOV, 2004)
$O(v^\omega)$	(MUCHA; SANKOWSKI, 2004) (HARVEY, 2009)	$O(v^\omega)$	(MUCHA; SANKOWSKI, 2004) (HARVEY, 2009)
Peso Máx em Grafos Bipartidos		**Peso Máx em Grafos Gerais**	
$O(W\sqrt{v}e)$	(GARBOW, 1985) (KAO et al., 2001)	$O(W\sqrt{v}e)$	(GARBOW, 1985)
$O(W\sqrt{v}e\log_v(\frac{v^2}{e}))$	(KAO et al., 2001)	$O(W\sqrt{v}e\log_v(\frac{v^2}{e}))$	(HUANG; KAVITHA, 2012)
$O(\sqrt{v}e\log(vW)))$	(GABOW; TARJAN, 1991) (GOLDBERG, 1993)	$O(e\log(vW)$ $\sqrt{v\log(v)*\alpha(e,v))}$	(GABOW; TARJAN, 1991)
$O(\sqrt{v}e\log(W))$	(DUAN; SU, 2012)		
$O(Wv^\omega)$	(SANKOWSKI, 2009)	$O(Wv^\omega)$	(HUANG; KAVITHA, 2012)
$O(vSP_+(v,e,W))$	(EDMONDS; KARP, 1972) (TOMIZAWA, 1971)	$O(v(e+v\log v))$	(GABOW; TARJAN, 1983)
$O(v^{2.5}\log(vW)$ $(\frac{\log}{\log v})^{\frac{1}{4}})$	(CHERIYAN; MEHLHORN, 1996)		

A tabela é dividida em quatro grupos que representam os tipos de algoritmos e as técnicas utilizadas para se definir o emparelhamento.

Cardinalidade Máxima ou Peso máximo se refere à heurística utilizada pelo algoritmo para determinar quais vértices entrarão ou não no emparelhamento, já as propriedades bipartido ou gerais referem-se ao tipo de grafo que será analisado, sendo que gerais podem ser quaisquer tipos de grafos incluindo os bipartidos.

Os algoritmos que merecem destaque são os de Hopcroft-Karp e o de Micali,Vazirani que possuem a menor complexidade em suas respectivas categorias. Ambos serão tratados posteriormente nas seções 6.1 e 5.3.2. respectivamente.

4.1.3.1 EMPARELHAMENTO DE GRAFOS BIPARTIDOS

Abaixo estão alguns exemplos de algoritmos de emparelhamento máximo para grafos bipartidos.

Algoritmo para Grafos Bipartidos

Algoritmo 1: Algoritmo para Grafos Bipartidos. Fonte: Bondy2008

Passo 1. Considere um emparelhamento E qualquer (pode até ser uma única aresta).
Passo 2. Busque um vértice x_0 que seja E-não saturado em X.

- **2.1** Se não existir nenhum, então E é o emparelhamento procurado;
- **2.2** Se existir tal vértice x_0, busque um caminho aumentado P, com origem x_0. Se tal caminho for encontrado crie um emparelhamento maior M' pela transferência ao longo do caminho. Uma vez que M' é um emparelhamento maior que M, ele satura mais vértices em X que M. Continue, então, a partir do Passo 3;
- **2.3** Se tal caminho não for encontrado, produza o subconjunto S de X, com $|N(S)| < |S|$, e, assim, G não tem emparelhamento que satura X;

Passo 3. Se todos os vértices de X forem saturados por M', então pare, dado que M' é o emparelhamento do tipo desejado. Caso contrário, repita o Passo 2 com M substituído por M'.

Este algoritmo inicia-se com um emparelhamento qualquer emparelhamento inicial e então, o algoritmo procura pelo caminho aumentado (Definição 5.6). A cada iteração tenta aumentar o número de arestas envolvidas no caminho aumentado, quando o algoritmo não consegue gerar um caminho maior, então para a execução e as arestas presentes no caminho aumentado são as arestas presentes no emparelhamento

Algoritmo Húngaro

Algoritmo 2: Algoritmo Húngaro. Fonte: (BONDY; MURTY, 2008)

Passo 1. Considere um emparelhamento E qualquer (pode até ser uma única aresta).
Passo 2. Se E satura todo vértice de X, pare. E é o emparelhamento do tipo desejado. Caso contrário, seja x_0 um vértice E-não-saturado de X, faça $S = \{x_0\}$ e $T = \emptyset$.
Passo 3. Se, em G, N(S) = T, então $|N(S)| < |S|$, uma vez que $|T| < |S| - 1$. Nesse caso, pare, uma vez que o Teorema do Casamento garante que G não tem emparelhamento que sature todo vértice em X. Caso contrário, escolha um elemento y em N(S) que não esteja em T.
Passo 4. Se y é E-saturado, seja $yz \in M$, ou seja, z é o vértice emparelhado a y sob M. Nesse caso, substitua S por $S \cup \{z\}$ e T por $T \cup \{y\}$ e retorne ao Passo 3 (note que os novos S e T ainda satisfazem $|T| < |S| - 1$). Caso contrário, uma vez que y é E-não-saturado, P seja um caminho E-aumentado a partir de x_0 a y e substitua E pela transferência ao longo de P de M, detonando esse novo emparelhamento também como E, retorne ao Passo 2.

4.1.3.2 EMPARELHAMENTO DE GRAFOS GERAIS

Abaixo seguem exemplos de algoritmos de emparelhamento máximo para grafos generalizados.

Algoritmo de Rabin e Vazirani

Algoritmo 3: Algoritmo de Rabin e Vazirani. Fonte (RABIN; VAZIRANI, 1989)

Entrada: Grafo
Saída: Emparelhamento Máximo
M := ∅ **início**
  for $i := 1$ ***to*** n **do**
    if v_i *não foi marcado como emparelhado* **then**
      calcule $A(G)^{-1}$;
      encontre j, tal que v_j pertence E(G) e $(A(G)^{-1}) \neq 0$
      $M := M \cup$ vj
      $G := G -$ vj
    retorna M
  end
fim

Neste algoritmo A(G) representa a matriz de adjacência de G e n é o número de vértices em G. Este algoritmo busca o caminho aumentado baseado no teorema de Rabin-Vazirani (RABIN; VAZIRANI, 1989).

Definição 4.7 *Dado G =(V,E) seja um grafo tendo o emparelhamento perfeito, e dado* $\tilde{A} = \tilde{A}(G)$ *seja a matriz de adjacência. Então* $(\tilde{A}^{-1})_{ij} \neq 0$*se e somente se o grafo* $G - v_i, v_i$ *é um emparelhamento perfeito.*

5 METODO

Busca-se analisar algoritmos gulosos, os mais genéricos possível para mensurar a quantidade de vértice necessários para se controlar um grafo. Para isto, serão utilizados softwares que poderão auxiliar na criação e bateria de testes deste algoritmo.

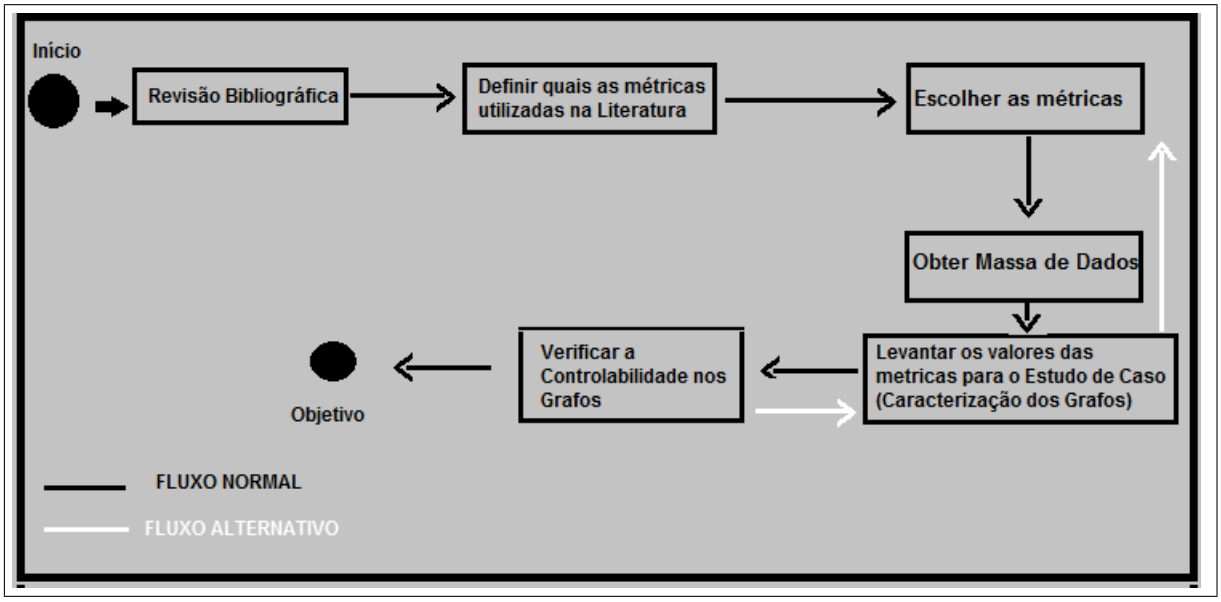

Figura 13: *Metodo - passos de desenvolvimento do projeto*

O projeto foi desenvolvido como mostrado na Figura 13. De início foi feita uma revisão bibliográfica para a interação com conceitos já difundidos e consolidados. Existem trabalhos de autores que são essenciais para qualquer projeto na área de redes complexas, entre eles podem ser citados Newman, Barabási, Strogatz. Conceitos como: Rede de Mundo pequeno, Rede de livre escala e Métricas já desenvolvidos, principalmente por estes autores, foram ponto de partida para o trabalho. Somando-se a estes, os conceitos de controlabilidade em grafo, difundidos por autores como Battiston, Newman e Barabási também foram empregados.

Após a etapa de revisão bibliográfica ocorreu a definição das métricas mais utilizadas na literatura, e o motivo de sua utilização. Com isso, pode-se obter conhecimento de quais métricas serão determinantes para o trabalho e qual o seu papel (o uso da métrica levará a qual informação). Dentre as métricas escolhidas pode-se citar, ordem, grau, grau de clusterização,

intermediação entre outras.

Definidas então as métricas, a próxima etapa foi obter as massas de dados a serem analisadas. Uma vez que existam os dados, transformados em grafos, a serem analisados, foi preciso usar as métricas selecionadas para caracterizar os grafos. Ou seja, aplicar as métricas aos grafos e obter os resultados de cada uma, para que assim, seja possível fazer um estudo de caso. Neste ponto (representado na Figura 13), existe um fluxo alternativo, a possibilidade que os dados coletados não tenham informações relevantes a serem estudadas.

No estágio final foi feita a verificação da controlabilidade dos grafos e a comparação dos resultados com algoritmos ótimos já conhecidos na literatura; as métricas calculadas na etapa anterior permitiram a mensuração de quais os nós que possuem maior controle sobre a rede, isto é, aqueles que se forem atacados podem influir no comportamento de grande parte da rede, ou toda ela. Nesta etapa há um fluxo alternativo que pode retornar à etapa anterior de caracterização de grafos, esta volta poderia ocorrer casos os dados levantados não fossem suficiente para a aplicação do algorítimo de controlabilidade.

Detalhes sobre o Projeto de Software, teste de validação e Planejamento são apresentados nos Apêndices A, B e C respectivamente.

Abaixo serão apresentados os algoritmos utilizados no projeto, sendo eles algoritmo ótimo (Seção 6.1, será utilizado como base para a comparação) e algoritmo 2-aproximação (Seção 6.2, para comparação com o algoritmo ótimo) e algoritmo guloso (Seção 6.3, desenvolvido neste projeto para determinar a controlabilidade).

6 ABORDAGEM EXPERIMENTAL

6.1 ALGORITMO DE CONTROLABILIDADE ÓTIMO

O algoritmo considerado ótimo está disponível na biblioteca Igraph (CSARDI; NEPUSZ, 2006), que foi desenvolvida por Tamas Nepusz e Gabor Csardi. O manual de referências (CSÁRDI; NEPUSZ, 2010) relata que o algoritmo desenvolvido é uma adaptação do algoritmo de Hopcroft-Karp (HOPCROFT; KARP, 1973).

Basicamente o que o algoritmo faz é definir um conjunto inicial de emparelhamentos, em então ele faz uma busca em largura para tentar melhorar estes emparelhamentos iniciais. (SAIP, 1993). Após esta etapa é executada uma etapa que busca aumentar o emparelhamento (caminho aumentado), o algoritmo executa esta etapa por meio de busca em profundidade e ao final de cada iteração verifica se houve uma melhora no resultado parcial. Caso o software não encontre uma melhore o algoritmo para e aquele é o emparelhamento máximo(SAIP, 1993).

Para a realização dos teste foi utilizado o software Netctrl (NEPUSZ; VICSEK, 2012). Este software implementa tanto o modelo de controlabilidade de Lui (LIU et al., 2011), como o método de controle desenvolvido por Nepusz e Vicsek (NEPUSZ, 2012). Foi utilizada uma versão pré compilada compatível com Linux 64-bits.

A complexidade do Algoritmo de Hopcroft-Karp é de $O(E\sqrt{V})$, onde V representa o grupo de vértices e E representa o grupo de arestas. (HOPCROFT; KARP, 1973)

6.2 ALGORITMO DE CONTROLABILIDADE 2-APROXIMAÇÃO

Algoritmos de aproximação [1] são geralmente utilizados em problemas de otimização visando encontrar soluções mais simples, porém não ótimas (SILVA, 2014). Segue abaixo o algoritmo de 2-aproximação desenvolvido para este trabalhado.

[1] O algoritmo de aproximação também se utiliza de uma abordagem gulosa, entretanto, para diferenciar os dois algoritmos e tornar o texto mais "didático" utilizarei a denominação de Algoritmo de Aproximação para o algoritmo guloso de aproximação que utiliza a heurística de escolher randomicamente as arestas. E utilizarei a denominação guloso para o algoritmo que utiliza a heurística de escolher o nó com menor grau

A complexidade do algoritmo tende a ser próxima de $\theta(\sqrt{|E|})$, onde E representa o grupo de arestas.No caso médio o algoritmo percorre a lista de arestas e apaga um número de arestas. A cada iteração a lista diminui seu tamanho e próxima passagem é menor que a anterior.

Algoritmo 4: Algoritmo de 2-Aproximação

```
Entrada: Grafo
Saída: Emparelhamento Maximal
M := ∅ início
    while Houver arestas em G do
        Selecione Aleatoriamente uma aresta (a,b) em G
        Armazene o vértice b em M
        for i = 0 to tamanho do Grafo do
            Percorra todo a Lista de arestas e apague uma aresta caso ela se
            origine do vértice a, ou incida no vértice b.
        end
    end
    retorna M
fim
```

Dado um Grafo G, escolha aleatoriamente uma aresta `ab` presente em G. Insira o vértice b no array M que representa o matching, e então remova todas as outras arestas que partem de a e as que chegam (incidem) em b. Repita a estratégia até que não haja mais arestas em G. Os nós que representam a controlabilidade, C, deste grafo são os nós que não estão presentes em M, ou seja, C = E - M.

6.2.1 EXECUÇÃO PASSO A PASSO DO ALGORITMO 2-APROXIMAÇÃO

Esta seção é destinada à explicar passo a passo como ocorre a execução do algoritmo de aproximação definido na seção anterior. O grafo utilizado será o da figura abaixo.

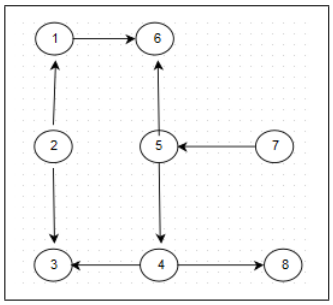

Figura 14: *Grafo G*

Para resolução do problema definiu-se que E representa o grupo de arestas do grafo, M o grupo de vértices que estão no matching.

- **Primeira Iteração:** Nesta etapa começa a iteração propriamente dita, as iterações ocorrerão até que o G seja vazio (as arestas são retiradas a cada iteração).

- Primeiro Passo: Escolhe-se uma aresta aleatoriamente. Aresta Escolhida foi a 5 4 (origina-se em 5 e incide em 4).

- Segundo Passo: Insere-se o vértice 4 em M. Apaga-se a aresta 5 4 de G.

- Terceiro Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 5 ou incida em 4. Nesta iteração a aresta 5 6 é apagada.

Resultado da Primeira Iteração:

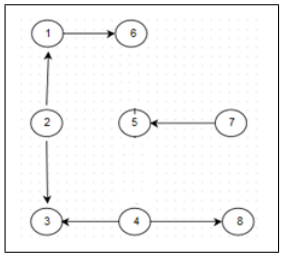

Figura 15: *Grafo G após a primeira iteração*

$M = \{4\}$.

- **Segunda Iteração:**

- Primeiro Passo: Escolhe-se uma aresta aleatoriamente. Aresta Escolhida foi a 2 1 (origina-se em 2 e incide em 1).

- Segundo Passo: Insere-se o vértice 1 em M. Apaga-se a aresta 2 1 de G.

- Terceiro Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 2 ou incida em 1. Nesta iteração a aresta 2 3 é apagada.

Resultado da Segunda Iteração:

$M = \{1,4\}$.

- **Terceira Iteração:**

- Primeiro Passo: Escolhe-se uma aresta aleatoriamente. Aresta Escolhida foi a 4 8 (origina-se em 4 e incide em 8).

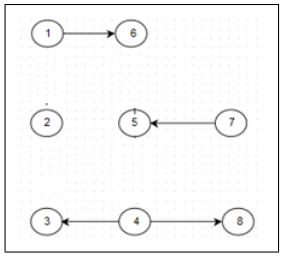

Figura 16: *Grafo G após a segunda iteração*

- Segundo Passo: Insere-se o vértice 8 em M. Apaga-se a aresta 4 8 de G.

- Terceiro Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 4 ou incida em 8. Nesta iteração a aresta 4 3 é apagada.

Resultado da Terceira Iteração:

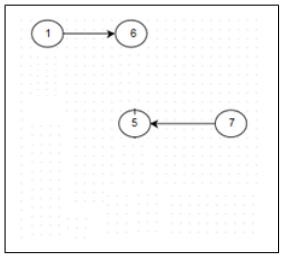

Figura 17: *Grafo G após a terceira iteração*

$$M = \{1, 4, 8\}$$

- **Quarta Iteração:**

- Primeiro Passo: Escolhe-se uma aresta aleatoriamente. Aresta Escolhida foi a 1 6 (origina-se em 1 e incide em 6).

- Segundo Passo: Insere-se o vértice 6 em M. Apaga-se a aresta 1 6 de G.

- Terceiro Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 1 ou incida em 6. Nesta iteração nenhuma aresta é apagada.

Resultado da Quarta Iteração:

$$M = \{1, 4, 6, 8\}$$

- **Quinta Iteração:**

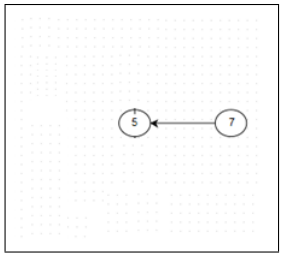

Figura 18: *Grafo G após a quarta iteração*

- Primeiro Passo: Escolhe-se uma aresta aleatoriamente. Aresta Escolhida foi a 7 5 (origina-se em 7 e incide em 5).

- Segundo Passo: Insere-se o vértice 5 em M. Apaga-se a aresta 7 5 de G.

- Terceiro Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 7 ou incida em 5. Nesta iteração nenhuma aresta é apagada.

- Quarto Passo: O grafo G está vazio neste momento.

Resultado da Quinta Iteração:

$$M = \{1,4,5,6,8\}$$

- **Resultado do algoritmo:** Os vértices que fazem parte do matching são 1,4,5,6,8.

Os vértices controladores são aqueles que estavam presentes em G e não estevam presentes no matching. Logo, os vértices controladores neste exemplo são: 2, 3, 7. Abaixo segue uma figura para ilustrar o resultado final.

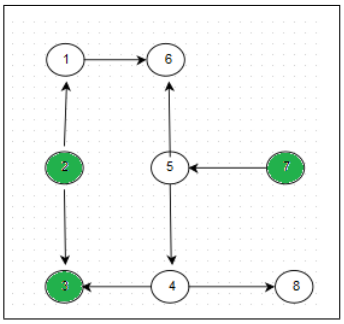

Figura 19: *Resultado do Algoritmo 2-Aproximação*

6.3 ALGORITMO DE CONTROLABILIDADE GULOSO

Dado um Grafo G, determine qual é o nó com menor grau. Então escolha, em G, uma aresta ab que contenha este nó (em caso de dois nós que tenham a mesmo grau a heurística

Algoritmo 5: Algoritmo Guloso

```
Entrada: Grafo
Saída: Emparelhamento Maximal
M := ∅ início
    Enumere o grau de cada nó (Passe pela lista de aresta e conte quantas
    vezes cada vértice aparece)
    while Houver arestas em G do
        Selecione O vértice de menor Grau
        Selecione uma aresta (a,b) em G que contenha este vértice (de menor
        peso)
        Armazene o vértice b em M
        for Cada Aresta do Grafo do
            apague a aresta caso ela se origine a, ou incida no vértice b.
        end
    end
    retorna M
fim
```

utilizada foi escolher o nó que tenha o menor módulo). Insira o vértice b no array M que representa o emparelhamento, e então remova todas as outras arestas que partem de a e as que chegam (incidem) em b. Repita a estratégia até que não haja mais arestas em G. Os nós que representam a controlabilidade, C, deste grafo são os nós que não estão presentes em M, ou seja, C = E - M.

6.3.1 EXECUÇÃO PASSO A PASSO DO ALGORITMO GULOSO

Esta seção é destinada à explicar passo a passo como ocorre a execução do algoritmo guloso definido na seção anterior. O grafo utilizado será o mesmo do seção do algoritmo 2-Aproximação.

Para resolução do problema definiu-se que G representa o grupo de arestas do grafo, M o grupo de vértices que estão no matching.

- **Pré Iteração:** É determinado o grau de cada vértice do Grafo G. O resultado final desta iteração é um vetor C que guarda a relação do vértice e seu grau (ver tabela abaixo)

Tabela 2: Pré Iteração

vértice	1	2	3	4	5	6	7	8
grau	2	2	2	3	3	2	1	1

- **Primeira Iteração:** Nesta Etapa começa a iteração propriamente dita, as iterações ocorrerão até que o G seja vazio (as arestas são retiradas a cada iteração).

- Primeiro Passo: Escolhe-se o vértice com o menor grau (em caso de empate eu opto pelo que tem menor módulo). O vértice escolhido foi o 7

- Segundo Passo: Escolhe-se uma aresta que contenha o vértice 7. Aresta Escolhida foi a 7 5 (origina-se em 7 e incide em 5).

- Terceiro Passo: Insere-se o vértice 5 em M. Apaga-se a aresta 7 5 de G (Diminui-se 1 para o vértice 5 e para o vértice 7 em C)

- Quarto Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 7 ou incida em 5. Nesta iteração nenhuma aresta é apagada.

Resultado da Primeira Iteração:

Tabela 3: Vetor C após a primeira Iteração

vértice	1	2	3	4	5	6	7	8
grau	2	2	2	3	2	2	0	1

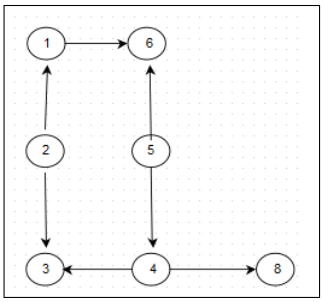

Figura 20: *Grafo G após a primeira iteração*

M = {5}

- **Segunda Iteração:**

- Primeiro Passo: Escolhe-se o vértice com o menor grau. O vértice escolhido foi o 8.

- Segundo Passo: Escolhe-se uma aresta que contenha o vértice 8. Aresta Escolhida foi a 4 8 (origina-se em 4 e incide em 8).

- Terceiro Passo: Insere-se o vértice 8 em M. Apaga-se a aresta 4 8 de G (Diminui-se 1 para o vértice 4 e para o vértice 8 em C)

- Quarto Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 4 ou incida em 8. Nesta iteração a aresta 4 3 é apagada.

Resultado da Segunda Iteração:

Tabela 4: Vetor C após a segunda Iteração

vértice	1	2	3	4	5	6	7	8
grau	2	2	1	1	2	2	0	0

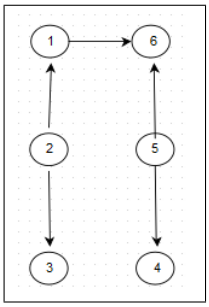

Figura 21: *Grafo G após a segunda iteração*

$M = \{5, 8\}$

- # Terceira Iteração:

- Primeiro Passo: Escolhe-se o vértice com o menor grau. O vértice escolhido foi o 3.

- Segundo Passo: Escolhe-se uma aresta que contenha o vértice 3. Aresta Escolhida foi a 2 3 (origina-se em 2 e incide em 3).

- Terceiro Passo: Insere-se o vértice 3 em M. Apaga-se a aresta 2 3 de G (Diminui-se 1 para o vértice 2 e para o vértice 3 em C)

- Quarto Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 2 ou incida em 3. Nesta iteração a aresta 2 1 é apagada.

Resultado da Terceira Iteração:

$M = \{3, 5, 8\}$

- # Quarta Iteração:

Tabela 5: Vetor C após a terceira Iteração

vértice	1	2	3	4	5	6	7	8
grau	1	0	0	1	2	2	0	0

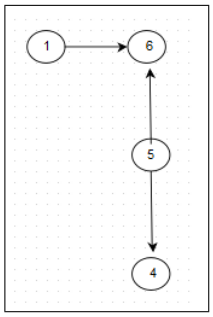

Figura 22: *Grafo G após a terceira iteração*

- Primeiro Passo: Escolhe-se o vértice com o menor grau. O vértice escolhido foi o 1.

- Segundo Passo: Escolhe-se uma aresta que contenha o vértice 1. Aresta Escolhida foi a 1 6 (origina-se em 1 e incide em 6).

- Terceiro Passo: Insere-se o vértice 6 em M. Apaga-se a aresta 1 6 de G (Diminui-se 1 para o vértice 1 e para o vértice 6 em C)

- Quarto Passo: Percorre-se G, e elimina-se uma aresta caso ela se inicie em 1 ou incida em 6. Nesta iteração a aresta 5 6 é apagada.

Resultado da Quarta Iteração:

Tabela 6: Vetor C após a quarta Iteração

vértice	1	2	3	4	5	6	7	8
grau	0	0	0	1	1	0	0	0

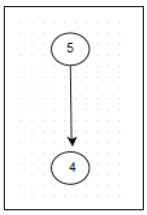

Figura 23: *Grafo G após a quarta iteração*

$M = \{3, 5, 6, 8\}$

- **Quinta Iteração:**

- Primeiro Passo: Escolhe-se o vértice com o menor grau. O vértice escolhido foi o 4.

- Segundo Passo: Escolhe-se uma aresta que contenha o vértice 4. Aresta Escolhida foi a 5 4 (origina-se em 5 e incide em 4).

- Terceiro Passo: Insere-se o vértice 4 em M. Apaga-se a aresta 5 4 de G (Diminui-se 1 para o vértice 5 e para o vértice 4 em C)

- Quarto Passo: O grafo G está vazio neste momento.

Resultado da Quarta Iteração:

Tabela 7: Vetor C após a quarta Iteração

vértice	1	2	3	4	5	6	7	8
grau	0	0	0	0	0	0	0	0

$M = \{3,4,5,6,8\}$

- **Resultado do algoritmo:** Os vértices que fazem parte do matching são 3,4,5,6,8.

Os vértices controladores são aqueles que estavam presentes em G e não estevam presentes no matching. Logo, os vértices controladores neste exemplo são: 1, 2, 7. Abaixo segue uma figura para ilustrar o resultado final.

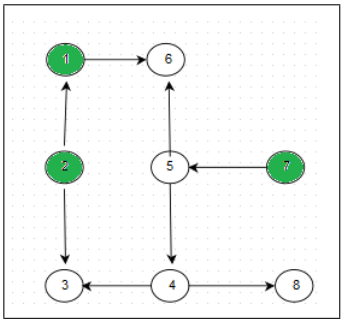

Figura 24: *Resultado do Algoritmo Guloso*

6.4 CENÁRIOS UTILIZADOS

Os experimentos foram executados em 34 grafos, extraídos do artigo *"Controllability of complex networks with node and edge dynamics"* (NEPUSZ, 2012). Os grafos utilizados nos experimentos variam de algumas dezenas até cerca de 75.000 vértices.

Na descrição de cada grafo, quando se descreve A regula B, por exemplo, isto representa que em uma aresta direcionada que sai de A e vai para B, o vértice A regula o vértice B. O vértice A sempre representará a origem e o B o destino. Caso seja representado ao contrário, A ainda será origem, apenas a semântica fica mais clara, por exemplo no grafo de *Prison inmate*, A é confiado por B; o mais claro seria B confia em A.

Abaixo, estão listados todos os grafos utilizados nos experimentos, com uma descrição sucinta de cada um deles:

TRN-Yeast

Rede Biológica, representa relações entre proteínas. Possui 688 vértices e 10079 arestas; as suas arestas são direcionadas e representam que A regula B.

TRN-EC-1

Rede Biológica, representa relações entre proteínas. Possui 418 vértices e 519 arestas; as suas arestas são direcionadas e representam que A regula B.

Prison inmate

Rede Social ue retrata a confiança entre presidiários de um determinado presídio, possui 67 vértices e 182 arestas. As suas arestas são direcionadas e representam que B confia em A.

Ythan

Rede Biológica que representa cadeia alimentar, possui 135 vértices e 601 arestas. As suas arestas são direcionadas e representam que A se alimenta de B.

Little Rock

Rede Biológica que representa cadeia alimentar, possui 183 vértices e 2494 arestas. As suas arestas são direcionadas e representam que A se alimenta de B.

GrassLand

Rede Biológica que representa cadeia alimentar, possui 88 vértices e 137 arestas. As suas arestas são direcionadas e representam que A se alimenta de B.

Seagrass

Rede Biológica que representa cadeia alimentar, possui 49 vértices e 226 arestas. As suas arestas são direcionadas e representam que A se alimenta de B.

s838

Rede Tecnológica que representa circuitos eletrônicos, possui 512 vértices e 819 arestas. As suas arestas são direcionadas e representam que B é uma função de A.

s420

Rede Tecnológica que representa circuitos eletrônicos, possui 252 vértices e 399 arestas. As suas arestas são direcionadas e representam que B é uma função de A.

s208

Rede Tecnológica que representa circuitos eletrônicos, possui 122 vértices e 189 arestas. As suas arestas são direcionadas e representam que B é uma função de A.

Neural

Rede Social, possui 297 vértices e 2345 arestas; as suas arestas são direcionadas e representam que A regula B.

ArXiv-HepTh

Rede Tecnológica que representa as citações de artigos, possui 27770 vértices e 352807 arestas. As suas arestas são direcionadas e representam que A foi citado por B.

ArXiv-HepPh

Rede Tecnológica que representa as citações de artigos, possui 34546 vértices e 421578 arestas. As suas arestas são direcionadas e representam que A foi citado por B.

nd.edu

Rede Tecnológica que representa as conexões entre sites e blogs, possui 325729 vértices e 1497134 arestas. As suas arestas são direcionadas e representam que A faz conexão para B.

stanford.edu

Rede Tecnológica que representa as conexões entre sites e blogs, possui 281903 vértices e 2312497 arestas. As suas arestas são direcionadas e representam que A faz conexão para B.

Political blogs

Rede Tecnológica que representa as conexões entre sites e blogs, possui 1224 vértices e 19025 arestas. As suas arestas são direcionadas e representam que A faz conexão para B.

p2p-1

Rede Tecnológica que representa troca de mensagens Peer-to-peer, possui 10876 vértices e 39994 arestas. As suas arestas são direcionadas e representam que A enviou mensagem para

B.

p2p-2

Rede Tecnológica que representa troca de mensagens Peer-to-peer, possui 8846 vértices e 31839 arestas. As suas arestas são direcionadas e representam que A enviou mensagem para B.

p2p-3

Rede Tecnológica que representa troca de mensagens Peer-to-peer, possui 8717 vértices e 91826 arestas. As suas arestas são direcionadas e representam que A enviou mensagem para B.

Freemans-2

Rede Social, possui 34 vértices e 830 arestas; as suas arestas são direcionadas e representam que A regula B.

Freemans-1

Rede Social, possui 34 vértices e 695 arestas; as suas arestas são direcionadas e representam que A regula B.

Manufacturing

Rede Social, possui 77 vértices e 2228 arestas; as suas arestas são direcionadas e representam que A regula B.

Consulting

Rede Social, possui 46 vértices e 879 arestas; as suas arestas são direcionadas e representam que A regula B.

7 RESULTADOS E ANÁLISE

Nesta seção serão apresentados os resultados experimentais para o problema de controlabilidade em grafos. Os algoritmos de 2-Aproximação e Guloso foram implementados em linguagem Java. O resultado denominado Controlabilidade Ótima foi obtido utilizando o software desenvolvido pelo pesquisador Tamas Nepusz e apresentado no Artigo *"Controllability of complex networks with node and edge dynamics"* (NEPUSZ, 2012).

Além do Algoritmo ótimo e o guloso, foi criada também uma implementação do algoritmo de 2-Aproximação, descrito no capítulo anterior. Os resultado obtidos com o algoritmo de aproximação são piores que o algoritmo guloso. Na média o número de nós controladores encontrados com o método de aproximação é 1,33 maior do que o número encontrado com o algoritmo guloso. Os resultados baseiam-se nos 24 grafos analisados, a serão apresentados a seguir.

A Tabela 8 descreve detalhes sobre o tamanho e também a natureza dos grafos, o tamanho dos grafos analisados variam de 34 até 325729. As redes variam entre redes sociais, biológicas, e tecnológicas, esta variação é importante para ser possível determinar que o algoritmo é eficiente para grafos dos mais variados tipos (tamanho, densidade, entre outros).

Tabela 8: Detalhes sobre a Base de dados

ID	Nome do Grafo	Vértices	Arestas	Tipo do Grafo
1	TRN-Yeast-2	688	1079	Regulatory
2	TRN-RC-2	418	519	Regulatory
3	Prison inmate	67	182	Trust
4	WikiVote	7115	103689	Trust
5	Epinions	75888	508837	Trust
6	Ythan	135	601	Food web
7	Little Rock	183	2494	Food web
8	GrassLand	88	137	Food web
9	Seagrass	49	226	Food web
10	s838	512	819	Eletronic
11	s420	252	399	Eletronic
12	s208	122	189	Eletronic
13	ArXiv-HepTh	27770	352807	Citação
14	ArXiv-HepPh	34546	421578	Citação
15	nd.edu	325729	1497134	www
16	stanford.edu	281903	2312497	www
17	p2p-1	10876	39994	Internet
18	p2p-2	8846	31839	Internet
19	p2p-3	8717	91826	Internet
20	Uclonline	1899	39994	Social
21	Freemans-2	34	830	intraOrg
22	Freemans-1	34	695	intraOrg
23	Manufacturing	77	2228	intraOrg
24	Consulting	46	879	intraOrg

Para a obtenção dos resultados foi utilizado um computador ASUS, com processador Core i7 3,4 Ghz Quad Box, memória RAM de 16 Gb DDR3.

O algoritmo ótimo foi executado pelo framework netcrl (NEPUSZ; VICSEK, 2012), para que fosse comprovada a veracidade dos dados, e de fato os dados eram condizentes que o que foi apresentado no artigo (LIU et al., 2011).

Os algoritmos guloso e de aproximação foram executados posteriormente e os resultados obtidos são mostradas na Tabela 9. Nesta tabela o número representa a densidade de *driver nodes* em relação ao total de vértices do grafo. Por exemplo, se para um grafo de 100 vértices, foram encontrados 20 *driver nodes*, o resultado seria 0,2 (20/100).

Tabela 9: Resultados Obtidos Com os experimentos

ID	Nome do Grafo	Ótimo	Guloso	2-Aproximação
1	TRN-Yeast-2	0,821	0,811	0,817
2	TRN-RC-2	0,751	0,746	0,749
3	Prison inmate	0,134	0,164	0,239
4	WikiVote	0,666	0,665	0,666
5	Epinions	0,549	0,514	0,216
6	Ythan	0,511	0,519	0,607
7	Little Rock	0,541	0,612	0,661
8	GrassLand	0,523	0,523	0,568
9	Seagrass	0,265	0,286	0,387
10	s838	0,232	0,219	0,256
11	s420	0,234	0,25	0,313
12	s208	0,238	0,262	0,279
13	ArXiv-HepTh	0,216	0,165	0,093
14	ArXiv-HepPh	0,232	0,183	0,154
15	nd.edu	0,677	-	-
16	stanford.edu	0,317	-	-
17	p2p-1	0,552	0,546	0,547
18	p2p-2	0,578	0,565	0,566
19	p2p-3	0,577	0,571	0,572
20	Uclonline	0,323	0,296	0,288
21	Freemans-2	0,029	0,029	0,029
22	Freemans-1	0,029	0,029	0,118
23	Manufacturing	0,013	0,013	0,091
24	Consulting	0,022	0,022	0,087

Na Tabela 10 são apresentados os erros percentuais obtidos em cada um dos experimentos. O erro percentual foi calculado com base nos resultados obtidos com o algoritmo ótimo.

Ao final da tabela são descritas as informações de erro médio e também desvio padrão. O erro médio para resultado guloso ficou em 6,25% com desvio padrão de 7,54. É importante ressaltar que em 12 casos o erro ficou abaixo de 1,5%, e em outros 6 o erro ficou entre 1,5% e 6,5%. Os erro mais altos encontrados foram para os grafos ArXiv-HepTh e ArXiv-HepPh, sendo 23,48% e 21,19% respectivamente. Estes grafos tem a características de serem pouco densas. Redes menos densas obtiveram, de maneira geral, resultado um pouco piores em detrimento à redes mais densos. Porém o resultado médio de 6,25% é um bom resultado pois o algorítmo guloso tem uma complexidade menor que o algorimo ótimo.

Para o algoritmo de Aproximação o erro médio encontrado 73,41% é um fator proibitivo, é um erro alto que inviabiliza seu uso geral, apesar de que para alguns casos específicos o algoritmo teve um resultado bom (e as vezes até melhor o guloso), o algoritmo de aproximação

funciona bem para grafos menores e mais densos.

Tabela 10: Erros Percentuais obtidos com os experimentos

ID	Nome do Grafo	Guloso (Erro Percentual)	2-Aproximação (Erro Percentual)
1	TRN-Yeast-2	1,21	0,50
2	TRN-RC-2	0,61	0,29
3	Prison inmate	22,52	89,35
4	WikiVote	0,08	0,03
5	Epinions	6,36	60,65
6	Ythan	1,47	18,87
7	Little Rock	13,13	22,22
8	GrassLand	0,05	8,64
9	Seagrass	7,82	46,32
10	s838	5,71	10,28
11	s420	6,84	33,97
12	s208	10,21	17,10
13	ArXiv-HepTh	23,48	56,82
14	ArXiv-HepPh	21,19	33,67
17	p2p-1	1,01	0,83
18	p2p-2	2,25	2,13
19	p2p-3	0,99	0,79
20	Uclonline	8,38	10,82
21	Freemans-2	1,42	1,42
22	Freemans-1	1,42	305,68
23	Manufacturing	0,10	599,30
24	Consulting	1,19	295,26
	Média	6,25	73,41
	Desvio Padrão	7,54	145,08

Para finalizar a análise dos dados é apresentado na Figura 25, um gráfico que ilustra a diferença entre o número de vértices obtidos para cada uma das abordagens (ótimo, guloso e aproximação).

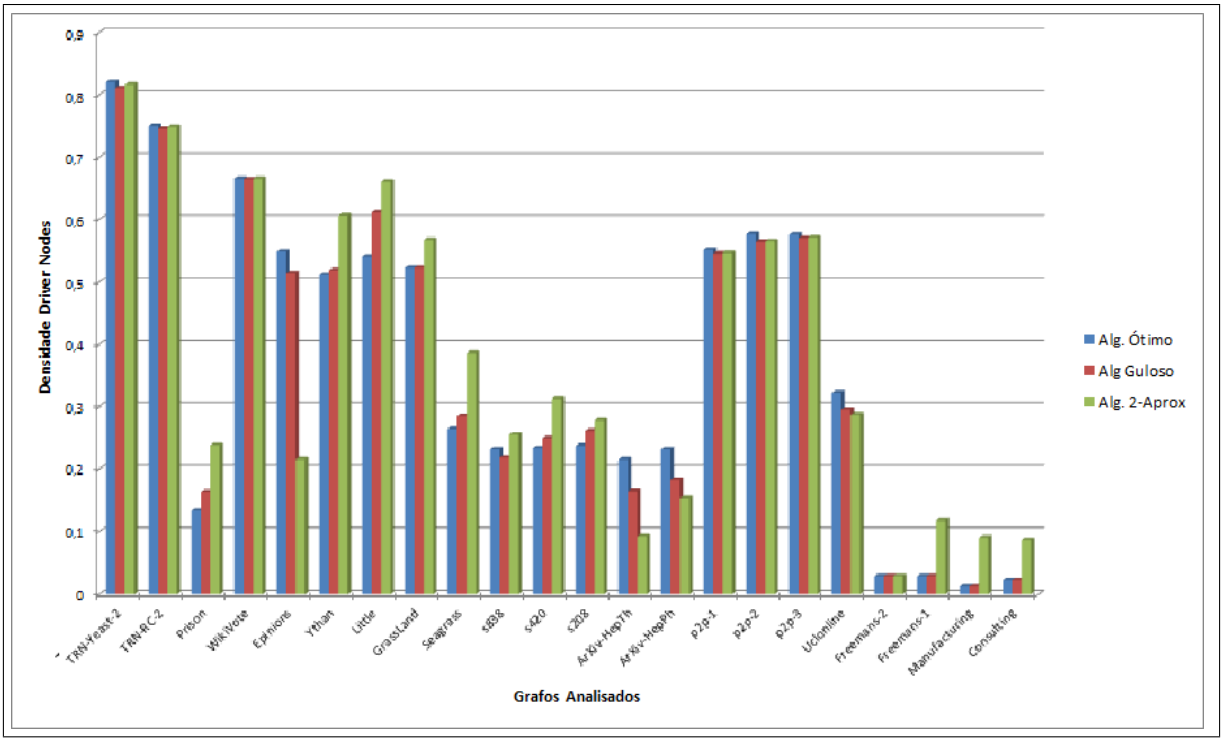

Figura 25: *Gráfico dos Resultados Obtidos*

A Figura 25 é importante para ilustrar que em todos os casos a abordagem gulosa sempre se aproximou da abordagem ótima, esta informação é relevante por que para redes com um baixo número de *driver nodes* uma porcentagem de erro de 6% ou 7% pode representar a diferença de 1 ou 2 vértices encontrados, é o que acontece no caso dos grafos s838 e s420, por exemplo.

8 CONCLUSÕES

Neste trabalho foi executado uma avaliação experimental relacionando o desempenho de dois algoritmos gulosos com o algoritmo ótimo.

O problema associado à avaliação do desempenho de algoritmos gulosos para calcular a controlabilidade de grafos reais, foi introduzido no primeiro capítulo. O problema foi abordado, por meio de um estudo experimental que relacionou o comportamento do algoritmo guloso em relação a outros algoritmos (aproximação e ótimo). Os experimentos realizados denotam que embora a estratégia gulosa não garanta sempre o melhor resultado o erro percentual ficou em média a 6,25% com uma complexidade algoritímica menor que a abordagem ótima. E o seu uso pode ser reafirmado quando é comparado com a abordagem de aproximação na qual o erro médio ficou em torno de 73%.

A controlabilidade é a capacidade que um grupo de vértices tem de levar o grafo de um estado inicial à outra estado final em um determinado tempo finito. Neste projeto buscou-se um grupo mínimo de vértices que pudessem levar o grafo em sua totalidade. O experimento foi realizado com 24 grafos que representam redes reais. É possível concluir que o para o caso de algoritmo de aproximação quanto mais denso fosse o grafo, menor foi o erro percentual encontrado. Isto se deve ao fato de que para uma rede densa, na qual os nós tem mais ou menos os mesmo graus, os *driver nodes* podem ser diferentes nós. já para redes menos densa os erros foram maiores.

Uma análise dos resultados também mostrou que, assim como demonstrado por LIU (2012), os *driver nodes* tendem a evitar os nós de maior grau (*hubs*). Este é um resultado importante, por que em uma primeira análise sobre o assunto é normal pensar que os nós de maior grau poderiam ser os mais importantes para a controlabilidade.

As principais contribuições foram tanto a comparação entre algoritmos, como o desenvolvimento de um algoritmo guloso de controlabilidade. São escassas as referências que tratam deste problema devido ao pouco tempo que o problema vem sendo abordado, o artigo de (LIU et al., 2011) foi talvez o primeiro a introduzir o problema relacionando-o à emparelhamento.

8.1 PROBLEMAS ENCONTRADOS

- **Problema da escassez de referências ligadas à controlabilidade em grafos.**

Isto foi um problema para a fundamentação do Capítulo 5 que abordava o tema controlabilidade, os trabalhos derivativos da pesquisa apresentada no artigo (LIU et al., 2011), isto dificulta pois as informações são basicamente as mesma mudando apenas a maneira que é apresentada.

- **Falta de Informação sobre alguns pontos importantes no artigo base.**

Durante o desenvolvimento e teste do algoritmo guloso, o resultado nunca era satisfatório em comparação com o que é apresentado no artigo (LIU et al., 2011).

Para o desenvolvimento do método Liu e Barabási (2011) fizeram uma pequena mudança no emparelhamento, como eles desejavam que o controle "fluísse" de um vértice para outro ele alteraram quais nós seriam marcados como emparelhados. No caso normal presente na maior parte da literatura tanto o nó de origem como o nó de destino da aresta são adicionados no emparelhamento; já no caso específico do trabalho deles apenas o nó destino é inserido.

Essa informação não estava presente no texto e só foi resolvido, eu encontrei uma apresentação de power-point do próprio Barabasi que apresentava esta adaptação.

Outro ponto importante é que no artigo (LIU et al., 2011), os autores não explicam que alguns grafos foram invertidos antes de serem analisados. Só encontrei esta informação quando entrei em contato com outro pesquisador (Tamás Nepusz) que implementou o algoritmo de Lui e Barabasi e me alertou sobre o fato de que alguns grafos eram analisados ao contrário (Inverte-se a orientação das arestas).

8.2 TRABALHOS FUTUROS

- Aplicar o problema em grafos que representem redes nacionais e avaliar se o resultado realmente se aplica. Um exemplo poderia ser um grafo de transito ou de rede elétrica.
- - Desenvolver outros algoritmos para controlabilidade com base em outros aspectos. Um exemplo para isto seria o trabalho desenvolvido por (BANERJEE; ROY, 2012), neste exemplo os autores determinam a controlabilidade analisando grau de intermediação (betweeness) e grau de proximidade (closeness).
- Estabelecer condições de dinâmica para a simulação do grafo.
- Simular a Dinâmica do grafo nos Tempo.

REFERÊNCIAS

ADAMIC, L. **Network basics & some tools**. 2008.

ADAMIC, L. **Scale Free Networks**. 2008.

ALVES-JR, N. **Introdução às Redes Complexas**. Ola: Centro Brasileiro de Pesquisa Física, 2008.

ARAúJO, A. **As pontes de Königsberg**. 2001. Disponível em: <http://www.mat.uc.pt/ alma/escolas/pontes/>.

BACKSTROM, L. et al. Four degrees of separation. In: ACM. **Proceedings of the 3rd Annual ACM Web Science Conference**. [S.l.], 2012. p. 33–42.

BANERJEE, S. J.; ROY, S. Key to network controllability. **arXiv preprint arXiv:1209.3737**, 2012.

BESSA, A. D. et al. **Introdução às Redes Complexas**. Dissertação (Mestrado) — Universidade Federal da Bahia, 2010.

BLUM, N. **A new approach to maximum matching in general graphs**. [S.l.]: Springer, 1990.

BONDY, J. A.; MURTY, U. **Graph theory, volume 244 of Graduate Texts in Mathematics**. [S.l.]: Springer, New York, 2008.

BRESSER-PEREIRA, L. C. A crise financeira global e depois: um novo capitalismo? **Novos Estudos-CEBRAP**, SciELO Brasil, n. 86, p. 51–72, 2010.

BULLMORE, E.; SPORNS, O. Complex brain networks: graph theoretical analysis of structural and functional systems. **Nature Reviews Neuroscience**, Nature Publishing Group, v. 10, n. 3, p. 186–198, 2009.

CÂMARA, A.; PORTO, E. Os seguidores da democracia: um estudo sobre o papel das redes sociais na campanha de barack obama. **Culturas Midiáticas**, v. 2, n. 2, 2011.

CHERIYAN, J.; MEHLHORN, K. Algorithms for dense graphs and networks on the random access computer. **Algorithmica**, Springer, v. 15, n. 6, p. 521–549, 1996.

COLNAGO, B. V. **Uma proposta para a formalização do problema de clusterização em grafos**. Dissertação (Mestrado), Curitiba, 2012.

CSARDI, G.; NEPUSZ, T. The igraph software package for complex network research. **InterJournal, Complex Systems**, v. 1695, n. 5, 2006.

CSÁRDI, G.; NEPUSZ, T. igraph reference manual. **URL: http://igraph. sourceforge. net/documentation. html (acesso em Agosto de 2014)**, v. 20, 2010.

DUAN, R.; SU, H.-H. A scaling algorithm for maximum weight matching in bipartite graphs. In: SIAM. **Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms**. [S.l.], 2012. p. 1413–1424.

EDMONDS, J.; KARP, R. M. Theoretical improvements in algorithmic efficiency for network flow problems. **Journal of the ACM (JACM)**, ACM, v. 19, n. 2, p. 248–264, 1972.

EHRENBERG, R.; MARTINO, M. Networks dominated by rule of the few.

FEDER, T.; MOTWANI, R. Clique partitions, graph compression and speeding-up algorithms. **Journal of Computer and System Sciences**, Elsevier, v. 51, n. 2, p. 261–272, 1995.

FEOFILOFF, P.; KOHAYAKAWA, Y.; Y., W. **Uma Introdução Sucinta à Teoria dos Grafos**. [S.l.], 2011. Disponível em: <http://www.ime.usp.br/ pf/teoriadosgrafos/texto/TeoriaDosGrafos.pdf.>.

FIGUEIREDO, C. M. H. de; SZWARCFITER, J. L. Emparelhamento em grafos.

GABOW, H. N.; TARJAN, R. E. A linear-time algorithm for a special case of disjoint set union. In: ACM. **Proceedings of the fifteenth annual ACM symposium on Theory of computing**. [S.l.], 1983. p. 246–251.

GABOW, H. N.; TARJAN, R. E. Faster scaling algorithms for general graph matching problems. **Journal of the ACM (JACM)**, ACM, v. 38, n. 4, p. 815–853, 1991.

GARBOW, H. N. Scaling algorithms for network problems. **Journal of Computer and System Sciences**, 1985.

GIMÉNEZ-LUGO, G. A. Redes sociais aplicadas a sistemas colaborativos. **WESAAC 2007 Workshop - Escola de Sistemas de Agentes para Ambientes Colaborativos**, 2007.

GOLDBERG, A. V. Scaling algorithms for the shortest paths problem. In: SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS. **Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms**. [S.l.], 1993. p. 222–231.

GOLDBERG, A. V.; KARZANOV, A. V. Maximum skew-symmetric flows and matchings. **Mathematical Programming**, Springer, v. 100, n. 3, p. 537–568, 2004.

GOMES, W. et al. A campanha on-line de barack obama em 20081. **Revista de Sociologia e Política**, SciELO Brasil, v. 17, n. 34, p. 29–43, 2009.

HARVEY, N. J. Algebraic algorithms for matching and matroid problems. **SIAM Journal on Computing**, SIAM, v. 39, n. 2, p. 679–702, 2009.

HOPCROFT, J. E.; KARP, R. M. An nˆ5/2 algorithm for maximum matchings in bipartite graphs. **SIAM Journal on computing**, SIAM, v. 2, n. 4, p. 225–231, 1973.

HUANG, C.-C.; KAVITHA, T. Efficient algorithms for maximum weight matchings in general graphs with small edge weights. In: SIAM. **Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms**. [S.l.], 2012. p. 1400–1412.

JUNGNICKEL, D.; SCHADE, T. **Graphs, networks and algorithms**. [S.l.]: Springer, 2005.

KAO, M.-Y. et al. A decomposition theorem for maximum weight bipartite matchings. **SIAM Journal on Computing**, SIAM, v. 31, n. 1, p. 18–26, 2001.

KARNER, G. Resource estimation for objectory projects objective systems. **Objective Systems SF AB (copyright owned by Rational/IBM)**, 1993.

LIU, Y.-Y.; SLOTINE, J. J.; BARABÁSI, A. L. Controllability of complex networks. **Nature**, v. 473, 2011.

LOPES, F. M. **Redes Complexas de expressão gênica: síntese, identificação, análise e aplicações**. Dissertação (Mestrado) — Universidade de São Paulo, 2011.

LOVELOCK, J. E.; MARGULIS, L. Atmospheric homeostasis by and for the biosphere: the gaia hypothesis. **Tellus**, Wiley Online Library, v. 26, n. 1-2, p. 2–10, 1974.

LUENBERGER, D. G. Introduction to dynamic systems: Theory, models & applications. **Wiley**, 1979.

MATEUS, I. d. A. T. A relação entre marcas e consumidores no facebook. 2010.

METZ J.; CALVO, R. et al. **Redes Complexas: Conceitos e Aplicações**. Dissertação (Mestrado) — Universidade de São Paulo, 2007.

MICALI, S.; VAZIRANI, V. V. An o(v, v, c , e,) algoithm for finding maximum matching in general graphs. In: **Proceedings of the 21st Annual Symposium on Foundations of Computer Science**. Washington, DC, USA: IEEE Computer Society, 1980. (SFCS '80), p. 17–27. Disponível em: <http://dx.doi.org/10.1109/SFCS.1980.12>.

MUCHA, M.; SANKOWSKI, P. Maximum matchings via gaussian elimination. In: IEEE. **Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on**. [S.l.], 2004. p. 248–255.

NEPUSZ, T. **Netctrl**. 2012. Disponível em: <http://ntamas.github.io/netctrl/>.

NEPUSZ, T.; VICSEK, T. Controlling edge dynamics in complex networks. **Nature Physics**, Nature Publishing Group, v. 8, n. 7, p. 568–573, 2012.

NEWMAN, M. E. J. **The structure and function of complex networks**. [S.l.: s.n.], 2003.

NICOLETTI, M. C. H. J. E. R. **Fundamentos da Teoria dos Grafos para Computação**. [S.l.]: Editora da UFSCar,, 2007.

PEREIRA, L. F. A.; HAFFNER, J. F. **Observabilidade e Controlabilidade**. Março 2013. Disponível em: <http://www.feng.pucrs.br/ gacs/new/disciplinas/psc/apostilas/Aula11$_n$ew.pdf¿.

PFLEEGER, S. L. **Engenharia de software: teoria e prática.** [S.l.: s.n.], 2004.

PRESSMAN, R. S. **Engenharia de Software**. [S.l.: s.n.], 1995.

RABIN, M. O.; VAZIRANI, V. V. Maximum matchings in general graphs through randomization. **Journal of Algorithms**, Elsevier, v. 10, n. 4, p. 557–567, 1989.

REAES, P. A. **Curso de Gestão da Produção, Notas de Aula - Redes Pert-CPM**. 2013.

RODRIGUES, F. A. **Caracterização, classificação e análise de redes complexas**. Tese (Doutorado) — Universidade de São Paulo, São Carlos, 2007.

SAIP, H. A. B. Algoritmos para emparelhamentos em grafos bipartidos. Biblioteca Digital da Unicamp, 1993.

SANKOWSKI, P. Maximum weight bipartite matching in matrix multiplication time. **Theoretical Computer Science**, Elsevier, v. 410, n. 44, p. 4480–4488, 2009.

SANTANA, D. S. **Modelando Redes Complexas em Grades Computacionais**. 2007. Universidade Federal da Bahia.

SILVA, M. O. d. Problema de cobertura por vértices em redes complexas. Curitiba, 2014.

TOMIZAWA, N. On some techniques useful for solution of transportation network problems. **Networks**, Wiley Online Library, v. 1, n. 2, p. 173–194, 1971.

WILSON, R. J.; WATKINS, J. J. **Graphs: An Introductory Approach. 1990**. [S.l.]: John Wiley & Sons, Inc.

APÊNDICE A – PROJETO DE SOFTWARE

Segue neste capítulo o desenvolvimento do projeto de software. Na primeira seção é detalhando o levantamento de requisitos (funcionais e não funcionais), diagramas de casos de uso.

Na seção posterior são descritos os procedimentos para teste e validação do software em questão.

A.1 LEVANTAMENTO DE REQUISITOS

A.1.1 REQUISITOS FUNCIONAIS

- O Usuário carregará uma rede no software para ser analisada
- O usuário decidirá se a rede será, ou não direcionada.
- O usuário executará o algoritmo
- O algoritmo responderá com os nós mais “controladores” da rede.

A.1.2 REQUISITOS NÃO FUNCIONAIS

- A rede terá de seguir os padrões presentes no Gephi
- O sistema aceitará apenas uma rede por vez
- A resposta será dada em uma tela com os nós e seu respectivo “poder” (controle sobre a rede)

A.2 DIAGRAMAS DE CASO DE USO

Os diagramas de caso de uso do sistema e suas respectivas especificações constam a seguir. Após todos os casos de uso individuais, consta o caso de uso geral do sistema.

A.2.0.1 USUARIO CARREGA A REDE A SER ANALISADA

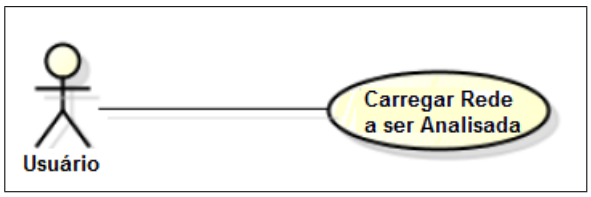

Figura 26: *Caso de Uso - Usuario Carrega a Rede a ser Analisada*

Especificação do Caso de Uso

Caso de Uso: Carregar Rede que será analisada

Ator Principal: Usuário

Ator de Suporte: Algoritmo

Descrição:O Administrador carrega a rede a ser analisada, a rede deve possuir uma configuração aceita no software Gephi

Pré-condições: Possuir a rede com as configurações corretas

Pós-condições:O rede está carregada e pronta para ser analisada

Fluxo Básico:O usuário escolhe a opção carregar rede e então carrega.

Fluxo Alternativo: •O software não consegue carregar a rede, devido à má configuração da mesma

•O usuário pode fechar a aplicação a qualquer momento.

•O usuário terá de corrigir os problemas de configuração da rede.

Regras de Negócio: Apenas uma rede por vez deve ser carregada.

A.2.0.2 USUARIO DECIDE SE A REDE E DIRECIONADA OU NÃO

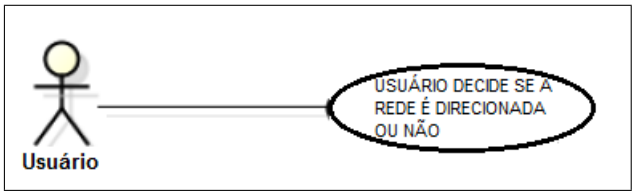

Figura 27: *Caso de Uso - Usuario decide se a Rede e Direcionada ou não*

Especificação do Caso de Uso

Caso de Uso: Usuário decide se a Rede é Direcionada ou não

Ator Principal: Usuário

Ator de Suporte: Software

Descrição: Com a Rede carregada o usuário decide se a análise será feita com uma rede direcionada ou não

Pré-condições:A rede já deve estar carregada no software

Pós-condições: Dependendo da Escolha do usuário a rede será configurada para direcionada, ou não direcionada.

Fluxo Básico: •O Administrador seleciona a opção “Rede Direcionada”.

•O Administrador seleciona a opção “Rede Não Direcionada”.

Fluxo Alternativo:

Regras de Negócio: Obviamente a rede deve ser direcionada ou não. Não podendo existir nós com características misturadas (um relação é direcionada e outras não).

A.2.0.3 USUARIO EXECUTA O ALGORITMO

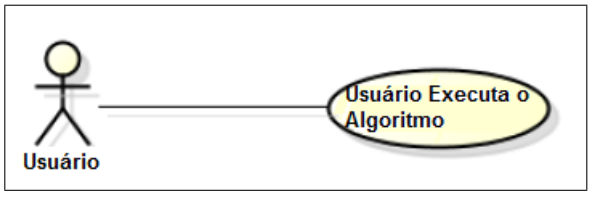

Figura 28: *Caso de Uso - Usuario Executa o Algoritmo*

Especificação do Caso de Uso

Caso de Uso: Execução do Algoritmo

Ator Principal: Usuário

Ator de Suporte: Software

Descrição:O usuário, com a rede já carregada , executa o software.

Pré-condições: •Rede já carregada

•Ou rede totalmente direcionada, ou rede totalmente não direcionada

Pós-condições: Outra tela se abrirá com o resultado dos nós mais controladores.

Fluxo Básico:O usuário seleciona a opção “Executar”.

Fluxo Alternativo:O software não executa devido à problemas com a rede

Regras de Negócio: Uma rede por vez deve ser analisada. Configuração de Direcionada, ou não deve ser respeitada

APÊNDICE B – PROCEDIMENTOS DE TESTE E VALIDAÇÃO

Testar um software é uma fase crítica para que se possa garantir a qualidade do software, esta etapa é a última revisão desde a especificação até a codificação (PFLEEGER, 2004).

B.1 DESCRIÇÃO DOS PROCEDIMENTOS DE TESTE E VALIDAÇÃO

Como as empresas que produzem software dependem do sucesso do programa, não é incomum encontrar projetos nos quais 40% de todo o trabalho é dedicado a testes e validações. Até porque se forem softwares que controlam aviões, ou algum aparelho médico por exemplo, a vida dos usuários depende do bom funcionamento do sistema, e este só é garantido com uma extensiva e programada bateria de testes (PRESSMAN, 1995).

Para o desenvolvimento do algoritmo será utilizado o TDD (Desenvolvimento Dirigido a Testes - métodos ágeis) , os testes serão executados durante a codificação.

Após a codificação serão utilizadas duas ferramentas presentes no framework Netbeans, são elas:

- JUnits: que é um framework que auxilia na definição e realização de testes unitários em classes Java.

- EMMA: ferramenta de código livre, capaz de fazer a análise da cobertura de testes em um processo Java.

Ao final do processo é gerado um relatório de quantas áreas foram cobertas pelos testes.

B.2 CRITERIOS DE ACEITAÇÃO PARA OS TESTES E VALIDAÇÕES

A validação de um software só é realizada depois de uma série de testes de caixa preta que prova que o sistema realmente está de acordo com os requisitos. O plano de teste é responsável por esboçar as classes de testes que serão realizadas e o procedimento de teste caracteriza os casos de teste específicos que poderão comprovar a conformidade com o levantamento

de requisitos (PFLEEGER, 2004).

As falhas existentes em um sistema podem ser (PRESSMAN, 1995):

- Stress: quando a estrutura de dado recebe mais informação do que foi projetada para receber.

- Defeitos de sincronia: código que coordena os eventos é inadequado.

- Defeito de desempenho: o sistema não funciona com a velocidade mínima descrita no projeto.

- Defeitos de recuperação: Ocorrem quando há uma falha e o sistema não se porta da maneira que deveria.

- Defeitos de padrões e procedimentos: não necessariamente afetam a execução do programa, mas criam um ambiente em que os erros aparecem na medida que o software é utilizado.

Uma vez que o software não apresente nenhuma destas falhas ele pode ser considerado eficiente para o uso a que se propõe.

B.3 DESCRIÇÃO DOS TESTES DE CAIXA PRETA PARA CADA CASO DE USO DO SISTEMA

Os métodos de teste de caixa preta se concentram nos requisitos funcionais do software. Com este teste é possível que o desenvolvedor simule todas as condições de entrada para cada requisito do programa. Os testes de caixa preta visam detectar falhas de funções incorretas ou ausentes, erros de interface, erros nas estruturas de dados ou no acesso a bancos de dados externos, erros no desempenho e erros de inicialização ou finalização (PFLEEGER, 2004) (PRESSMAN, 1995).

A execução destes testes tem de se concentrar nos “limites” de certos casos, por exemplo:

- o algoritmo possui um “if” que será verdadeiro se a condição for maior ou igual a 2 ou menor que 10. Neste caso as entradas deverão ser 1,99 , 2, 9,99, 10 , 10.01.

Seguem abaixo os testes de caixa preta desenvolvidos para cada caso de Uso.

B.3.1 USUARIO CARREGA A REDE A SER ANALISADA

- Testar com redes fora do padrão;

- Redes que tenham apenas alguns caracteres fora do padrão;

- Redes excessivamente grandes, (milhares de nós, com milhões de conexões)

B.3.2 USUARIO DECIDE SE A REDE E DIRECIONADA OU NÃO

-Transformar uma rede direcionada em Não direcionada.

-Transformar uma rede não direcionada em direcionada.

- Testar, após uma troca, voltar a rede inicial.

B.3.3 USUARIO EXECUTA O ALGORITMO

- Redes excessivamente grandes, (milhares de nós, com milhões de conexões)

- Inserir erros, que teoricamente inviabilizariam a execução.

B.3.4 O ALGORITMO RESPONDERA COM OS NOS MAIS “CONTROLADORES” DA REDE

- Eliminar os nós considerados “controladores”;

- Inserir redes com configurações já apresentadas em outros trabalhos (“comparação de resultados”, resultados diferentes não significam necessariamente que o algoritmo esteja errado);

- Verificar o tamanho da entrada e o tempo de processamento;

APÊNDICE C – PLANEJAMENTO

O objetivo deste capítulo é relatar questões técnicas como a seção contexto que explicitará quem são os interessados diretamente pelo trabalho, e quem são os responsáveis por financiá-lo.

Na seção de Levantamento de Recursos de Hardware e Software é discutida a necessidade de se usar ou não softwares e hardwares auxiliares.

Na seção de Use Case Points são levantados dados baseados nos casos de uso. estes dados mensuram o tempo e as condições de execução do projeto.

As seções que se seguem (Gerência de Tempo e Cronograma Preliminar), demonstram as etapas do projeto e o cronograma a ser seguido, respectivamente.

Ao final do capítulo, a viabilidade técnica e financeira do projeto, buscando demonstrar a real possibilidade de se prosseguir com o trabalho. Será demonstrado um cronograma preliminar, que nada mais é do que um prognóstico de como teoricamente o projeto será produzido.

C.1 LEVANTAMENTO DE RECURSOS DE HARDWARE E SOFTWARE

Pela natureza do trabalho, é desnecessário o uso de Hardware específico, portanto o documento não possuirá levantamento de Hardware.

A utilização de software é necessária, mas não é imprescindível, já que o produto final é o algoritmo. Os softwares, citado no capítulo Método, foram utilizados por atenderem algumas necessidades básicas, mas não são de maneira alguma insubstituíveis, podendo servir como opção a eles qualquer linguagem de programação que permita a codificação da simulação. Contudo, ao invés do Levantamento de Software criterioso, foi escrito um Apêndice com noções básicas dos programas citados no Capítulo Método.

A massa de dados de testes a ser utilizada é vasta, a princípio serão usados os dados utilizados no artigo de Yang, Slotine e Barabási, no artigo *"Controllability of complex networks"*.

É vantajoso se utilizar desta massa de dados pois é possível comparar os resultados obtidos com os apresentados no artigo.

C.2 USE CASE POINTS

O método de Use Case Point foi desenvolvido por Gustav Karner em 1993 (KARNER, 1993). O propósito do autor foi criar um sistema que permitisse predizer o total de recursos necessário no início do processo de desenvolvimento.

Karner, dividiu o processo em 5 passos, que se complementam para a obtenção do resultado de quantas horas seriam necessárias para o trabalho.

As etapas e a forma com que são definidos cada passo estão disponíveis no artigo (KARNER, 1993). Abaixo são demonstrados apenas os resultados obtidos em cada etapa.

C.2.1 USE CASE POINT : 1° PASSO

Neste primeiro passo são definidos os tipos de atores, presentes no projeto com seus respectivos pesos.

Tabela 11: Primeiro Passo Use Case

Tipo do Autor	Peso	Número de Atores	Resultado
Ator Simples	1	0	0
Ator Médio	2	0	0
Ator Complexo	3	3	9
Total de UAW			9

C.2.2 USE CASE POINT : 2° PASSO

Depois de calculado o peso total dos atores, então calcula-se o peso dos casos de uso definidos no capítulo Projeto de Software.

Tabela 12: Segundo Passo Use Case

Complexidade	Peso	Número de Caso de Uso	Resultado
Caso de Uso Simples	5	1	15
Caso de Uso Médio	10	0	0
Caso de Uso Complexo	10	0	0
Total de UUCW			15

C.2.3 USE CASE POINT : 3° PASSO

Neste terceiro passo é calculado os pontos de casos de uso não-ajustado, que nada mais são que a soma dos resultados obtidos 1º e 2º passo.

$$UUCP = 12 + 15 = 27 \tag{11}$$

C.2.4 USE CASE POINT : 4° PASSO

Nesta etapa são levantados os valores técnicos (funcionais) e fatores de ambiente (não funcionais do projeto)

Tabela 13: Fator de Complexidade Técnica

Fator	Fatores que contribuem para a complexidade	Peso	Valor	Total
F1	Sistemas Distribuídos	2	0	0
F2	Tempo de Resposta	1	3	3
F3	Eficiência para o usuário final (online)	1	0	0
F4	Processamento interno Complexo	1	2	2
F5	Código Reusável	1	3	3
F6	Facilidade de Instalação	0,5	0	0
F7	Facilidade de uso (facilidade operacional)	0,5	5	2,5
F8	Portabilidade (facilidade operacional)	2	4	8
F9	Facilidade de mudança (facilidade operacional)	1	4	4
F10	Concorrência (acesso simultâneo à aplicação)	1	1	1
F11	Recursos de Segurança	1	0	0
F12	Fornece acesso direto para terceiros	1	0	0
F13	Requer treinamento especial para o usuário	1	0	0
TFactor				23,5

$$TCF = 0,6 + (0,01 + TFactor) \tag{12}$$

TCF = 0,6 + (0,01 * 23,5)

TCF = 0,6 + (0,235)

TCF = 0,835

$$EF = 1,4 + (-0,03xEFactor) \tag{13}$$

EF = 1,4 + (-0,03 x 14,5)

Tabela 14: Fator de Complexidade de Ambiente

Fator	Fatores que contribuem para a complexidade	Peso	Valor	Total
F1	Familiariedade com o processo formal de desenvolvimento	1,75	5	7,5
F2	Colaboradores de Meio Período	-1	2	-2
F3	Capacidade do Lider em analisar os requisitos e modelagem	0,5	5	2,5
F4	Experiência em desenvolvimento de aplicações do gênero	0,5	3	1,5
F5	Experiência em Orientação a Objetos	1	3	3
F6	Motivação da Equipe	1	2	2
F7	Dificuldades com a linguagem de programação	-1	4	-4
F8	Requisitos estáveis	2	2	4
EFactor				14,5

EF = 1,4 + (- 0,435)

EF = 0,965

C.2.5 USE CASE POINT : 5° PASSO

No último passo são determinadas quantas horas o projeto necessita para ser realizado.

$$UCP = UUCPxTCFxEF \tag{14}$$

UCP = 24 x 0,835 x 0,965

UCP = 19,34

Tempo de Trabalho Estimado:

Tempo Estimado : 19,34 x 20

Tempo Estimado: 386,77 Horas de Trabalho

O tempo estimado para o trabalho foi de 387,77 horas, como o cronograma conta com 38 semanas, o tempo necessário é de aproximadamente 10 horas por semana, entretanto, obviamente podem haver problemas e este tempo é apenas uma estimativa.

C.3 GERÊNCIA DE TEMPO (REDES PERT-CPM)

Os termos PERT (Program Evaluation and Review Technique) e CPM (Critical Path Method), são técnicas desenvolvidas por volta de 1950, apesar de serem métodos independentes são usados juntos devido à semelhança entre eles (REAES, 2013).

Utilizam-se principalmente dos conceitos grafos para o planejamento, visualização e

coordenação das atividades do projeto (REAES, 2013).

Cronograma

Tabela 15: Gerência de Tempo (Redes Pert-CPM)

Número da Tarefa	Nome da Tarefa	Duração (Semanas)
1	Levantamento Bibliográfico	8
2	Definição de Métricas	3
3	Obter Base de Dados	4
4	Levantamento de Valores das Métricas	4
5	Verificar Controlabilidade nos Grafos	4
6	Buscar Motifs nos Grafos	3
7	Estabelecer Condições de Dinâmica	4
8	Implementar Algoritmo de Controlabilidade	2
9	Simular Dinâmica das Redes no Tempo	8

Rede Pert-CPM

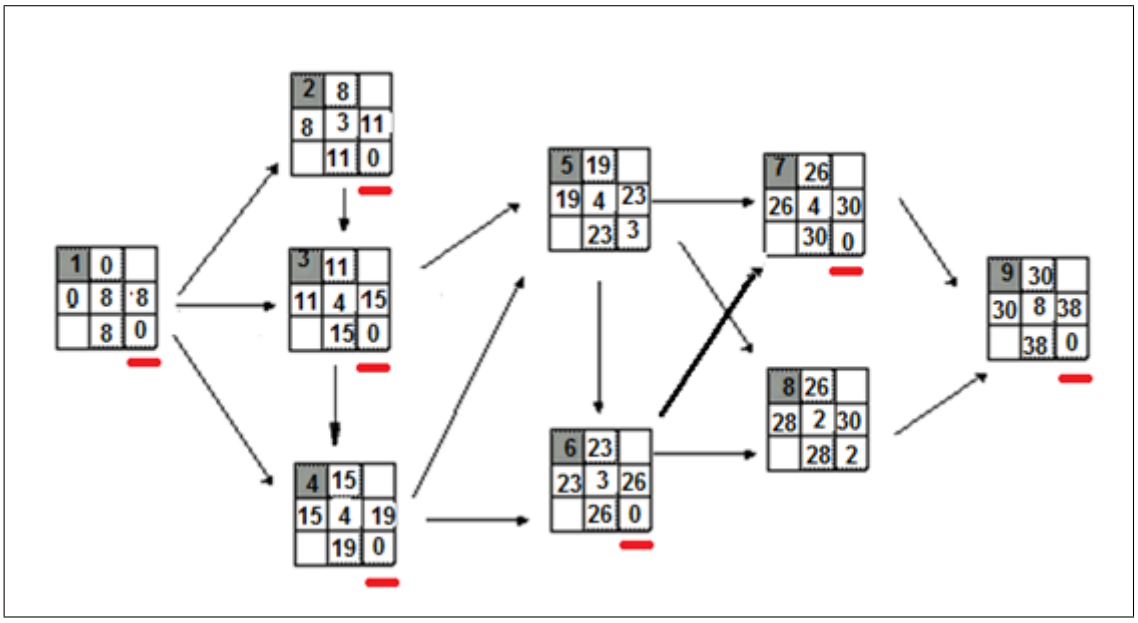

Figura 29: *Rede de Gerência (PERT-CPM)*

O Caminho Crítico é dado por :

1 -¿ 2 -¿ 3 -¿ 4 -¿ 6 -¿ 7 -¿ 9 O projeto todo foi dimensionado para 38 semanas de trabalho

C.4 CRONOGRAMA PRELIMINAR

O tempo calculado para a duração do trabalho é de 13 meses iniciado em Fevereiro de 2013, com final em Março de 2014.

Nome da tarefa	Duração	Início	Término
Trabalho de Conclusão de Curso	**342 dias**	**Qua 05/12/12**	**Qui 27/03/14**
Trabalho de conclusão de Curso I	**111 dias**	**Qua 05/12/12**	**Qua 08/05/13**
Pré-Proposta	**34 dias**	**Qua 05/12/12**	**Seg 21/01/13**
Entrevista com o Orientador	1 dia	Qua 12/12/12	Qua 12/12/12
Retorno das Respostas (Prof. Gustavo	3 dias	Qui 13/12/12	Seg 17/12/12
Discussão e Entrega	1 dia	Ter 18/12/12	Ter 18/12/12
Levantamento Bibliográfico	13 dias	Qua 05/12/12	Sex 21/12/12
Recesso	11 dias	Sáb 22/12/12	Sex 04/01/13
Aprofundamento Teórico	21 dias	Seg 24/12/12	Seg 21/01/13
Proposta TCC	**40 dias**	**Ter 12/02/13**	**Seg 08/04/13**
Introdução	7 dias	Ter 12/02/13	Qua 20/02/13
Referencial Teórico	8 dias	Qui 21/02/13	Seg 04/03/13
Método	6 dias	Ter 05/03/13	Ter 12/03/13
Recursos de Hardware e Software, Viabilidade, Cronograma, Contexto, Conclusões e Referências Bibliográficas	5 dias	Qua 13/03/13	Ter 19/03/13
Correção	4 dias	Qua 03/04/13	Seg 08/04/13
Implementação	**21 dias**	**Qua 10/04/13**	**Qua 08/05/13**
Definição de Métricas	12 dias	Ter 05/03/13	Qua 20/03/13
Obter Base de Dados	13 dias	Qui 21/03/13	Seg 08/04/13
Levantamento de Valores das Métricas	9 dias	Ter 09/04/13	Sex 19/04/13
Plano de Projeto	**31 dias**	**Qua 27/03/13**	**Qua 08/05/13**
Levantamento de Requisitos e Diagramas de Caso de Uso	5 dias	Qua 27/03/13	Ter 02/04/13
Cálculo dos Use Case Points	5 dias	Qua 03/04/13	Ter 09/04/13
Lista de tarefas, Rede PERT-CPM e Caminho Crítico	5 dias	Qua 17/04/13	Ter 23/04/13
Projeto de Hardware e Procedimentos de Teste e Validação.	6 dias	Ter 23/04/13	Ter 30/04/13
Análise de Risco.	6 dias	Ter 30/04/13	Ter 07/05/13
Entrega do Plano de Projeto	1 dia	Ter 07/05/13	Ter 07/05/13
Apresentação	2 dias	Ter 07/05/13	Qua 08/05/13
Recesso	13 dias	Qui 09/05/13	Seg 27/05/13
Período Intermediário	**94 dias**	**Ter 28/05/13**	**Sex 04/10/13**
Trabalho de Conclisão de Curso II	**113 dias**	**Seg 07/10/13**	**Qua 12/03/14**

Figura 30: *Cronograma TCC I*

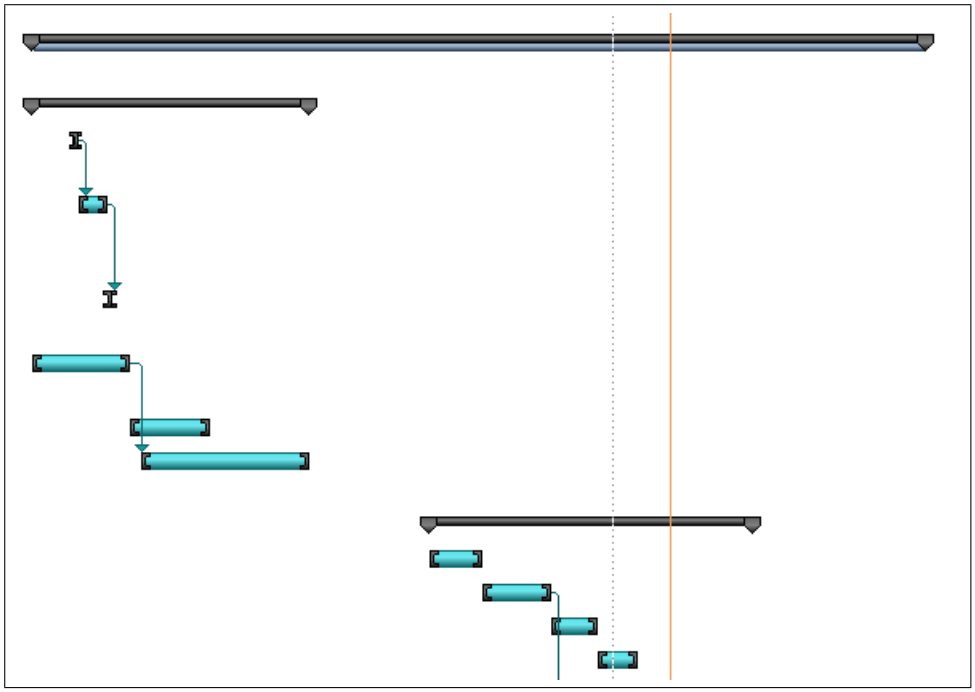

Figura 31: *Grafico de Gantt TCC I*

C.5 VIABILIDADE

O projeto é totalmente viável sob a visão financeira, pois, inicialmente não envolve Hardware ou Softwares Pagos; entretanto para a execução será necessário o trabalho semanal

Nome da tarefa	Duração	Início	Término
Trabalho de Conclusão de Curso	**342 dias**	**Qua 05/12/12**	**Qui 27/03/14**
Trabalho de conclusão de Curso I	**111 dias**	**Qua 05/12/12**	**Qua 08/05/13**
Recesso	13 dias	Qui 09/05/13	Seg 27/05/13
Período Intermediário	**94 dias**	**Ter 28/05/13**	**Sex 04/10/13**
Verificar a Controlabilidade nos Grafos	25 dias	Ter 28/05/13	Dom 30/06/13
Buscar Motifs Nos Grafos	21 dias	Seg 01/07/13	Seg 29/07/13
Possibilidade de VOLTA NO PROJETO	24 dias	Ter 30/07/13	Sex 30/08/13
Estabelecer Condições de Dinâmca para a Simulação do Comportamento do	11 dias	Seg 02/09/13	Seg 16/09/13
Implementar o Algoritmo de Controlabilidade Para o Gephi	14 dias	Ter 17/09/13	Sex 04/10/13
Trabalho de Conclisão de Curso II	**113 dias**	**Seg 07/10/13**	**Qua 12/03/14**
Possibilidade de VOLTA NO PROJETO	24 dias	Seg 07/10/13	Qui 07/11/13
Simular a Dinâmica das Redes no Tempo	89 dias	Sex 08/11/13	Qua 12/03/14

Figura 32: *Cronograma TCC II*

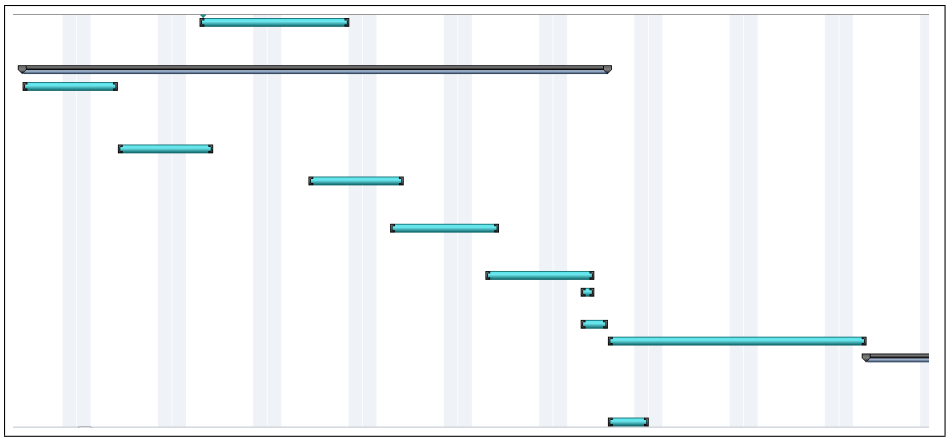

Figura 33: *Grafico de Gantt TCC II*

e futuramente diário de profissionais (e candidatos a profissional) da área. De início será o estudante e o orientador, mas o estudo pode avançar e abranger mais pessoas.

O único custo existente seria calculado com base no preço, por hora, de cada pessoa envolvida no projeto.

Em anexo é mostrada a Análise de Riscos, e do ponto de vista técnico é um trabalho factível, uma vez que existem inúmeras referências disponíveis e os risco não impossibilitam o andamento do projeto.

APÊNDICE D – ANALISE DE RISCOS

Projeto: Controlabilidade em Redes Complexas	
1ª Etapa: **Identificação do Risco**	
Denominação do risco: Não cumprimento de Metas estabelecidas no cronograma	Nº Identificação 1
Descrição do risco: Não conseguir os resultados intermediários ou finais no prazo estipulado.	
2ª Etapa: **Avaliação do Risco**	
Impacto: 5(alto) Explique: Pode acarretar desde uma carga horária maior para o cumprimento do projeto ou até mesmo o seu fracasso.	
Probabilidade: 3(média) Explique: Apesar da vontade de desenvolver o projeto pode ser que aconteça algum problema que inviabilize o projeto.	
3ª Etapa: **Desenvolvimento da Resposta ao Risco**	
Ações, Responsáveis e Datas de Conclusão.	
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): Prevenir: Deverão ser feitas reuniões para o acompanhamento do andamento do projeto, remanejando atividades que não estiverem sendo eficientes. Transferir:- Mitigar: Remanejar o cronograma e/ou os requisitos do projeto para que possa se adequar às possibilidades de trabalho. Aceitar: -	
Impacto reavaliado: 4(médio/alto)	**Probabilidade reavaliada**: 2(média/**baixa)**

Elaborado por: Leonardo	Data: 23/04/2013	Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou Anexos

Formulário sugerido por Gasnier, 2000 Editora IMAN e alterado por Wille.

Planejamento de Riscos

Projeto: Controlabilidade em Redes Complexas

1ª Etapa: **Identificação do Risco**

Denominação do risco: Desistência de um membro da equipe	Nº Identificação 2
Descrição do risco: Desistência do projeto e/ou da disciplina.	

2ª Etapa: **Avaliação do Risco**

Impacto: 2(alto) Explique: A desistência inviabilizaria totalmente o projeto. O orientador também pode desistir do projeto.
Probabilidade: 2(média) Explique: Desentendimento das partes; Projeto inviável para uma das partes.

3ª Etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão.	
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): Prevenir: Conversar periodicamente com o orientador, definir metas alcançáveis em curto prazo. Transferir: Mitigar: Aceitar: E preciso saber se há possibilidade de se desenvolver um projeto na mesma área, com o escopo um pouco diferente.	
Impacto reavaliado: 3(médio)	**Probabilidade reavaliada:** 3(média)

Elaborado por: Leonardo	Data: 24/04/2013	Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou Anexos

Formulário sugerido por Gasnier, 2000 Editora IMAN e alterado por Wille

Planejamento de Riscos

Projeto: Controlabilidade em Redes Complexas

1ª Etapa: **Identificação do Risco**

Denominação do risco: Falha de planejamento	N Identificação 3
Descrição do risco: Planejamento é o fator crucial no sucesso de um projeto e suas falhas podem vir de diferentes maneiras, como erro na previsão da duração das tarefas, erro na distribuição das tarefas, entre outros.	

2ª Etapa: **Avaliação do Risco**

Impacto: 4(médio/alto) Explique: Gera problemas que podem prejudicar e inviabilizar o resultado final esperado
Probabilidade: 2(média/baixa) Explique: É pouco provável que aconteça devido à experiência do orientador

3ª Etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): Prevenir: Dedicar muito tempo ao planejamento, pensar no maior número de aspectos possível relacionados ao projeto. Transferir:- Mitigar: Possuir alternativas de planejamento, caso a primeira dê errado. Aceitar: Quando percebido o problema, repensar e refazer o planejamento para diminuir as perdas.	
Impacto reavaliado: 3(médio)	**Probabilidade reavaliada:** 2(média/baixa)

Elaborado por: Leonardo	Data: 24/04/2013	Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou Anexos

Formulário sugerido por Gasnier, 2000 Editora IMAN e alterado por Wille

Planejamento de Riscos

Projeto: Controlabilidade em Redes Complexas	

1ª Etapa: **Identificação do Risco**

Denominação do risco: Escolha de tecnologias inadequadas	N Identificação 4
Descrição do risco: Escolher uma abordagem e linguagem que não responda adequadamente as suas expectativas.	

2ª Etapa: **Avaliação do Risco**

Impacto: 5(alto) Explique: Escolher uma forma de resolver o problema que não atenda aos requisitos, exemplo: o programa demora de mais para processar as informações, o programa não suporta o volume de dados.
Probabilidade: 1 (baixa) Explique: Como as tecnologias vêm sendo pesquisadas desde o início da disciplina, a chance de escolher algo inadequado é muito baixa.

3ª Etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): Prevenir: Buscar pesquisar ao máximo as tecnologias que serão utilizadas para a verificação de que as mesmas são adequadas às necessidades do projeto. Transferir:- Mitigar: Procurar possuir uma segunda opção para que caso ocorrido, o processo de escolha de outra tecnologia não seja demorado. Aceitar: Uma vez ocorrido, procurar encontrar a outra tecnologia o mais rápido possível, realocando ainda as horas de trabalho.	
Impacto reavaliado: 3(médio)	**Probabilidade reavaliada:** 1(baixa)

Elaborado por: Leonardo	Data: 24/04/2013	Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou Anexos

Formulário sugerido por Gasnier, 2000 Editora IMAN e alterado por Wille

Planejamento de Riscos

Projeto: Controlabilidade em Redes Complexas	
1ª Etapa: **Identificação do Risco**	
Denominação do risco: Problemas de Saúde	N Identificação 5
Descrição do risco: Sofrer de algum dano à saúde.	
2ª Etapa: **Avaliação do Risco**	
Impacto: 3(médio) Explique: A indisponibilidade devido a problemas de saúde pode levar ao atraso do projeto.	
Probabilidade: 2(média/baixa) Explique: O risco de ocorrer um problema de saúde grave é pequeno.	
3ª Etapa: **Desenvolvimento da Resposta ao Risco**	
Ações, Responsáveis e Datas de Conclusão	
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): Prevenir: Levar uma vida com hábitos saudáveis evitando o estresse ajuda a evitar problemas de saúde. Transferir:- Mitigar: Redistribuir as tarefas atrasadas e tentar cumpri-las da melhor forma possível. Aceitar: Contornar este problema, além disso, o trabalho tem que ser realocado.	
Impacto reavaliado: 2(médio/baixo)	**Probabilidade reavaliada**: 2(média/baixa)

Elaborado por: Leonardo	Data: 24/04/2013	Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou Anexos

Formulário sugerido por Gasnier, 2000 Editora IMAN e alterado por Wille

Planejamento de Riscos

Projeto: Controlabilidade em Redes Complexas

1ª Etapa: **Identificação do Risco**

Denominação do risco: Subestimação dos Problemas	N Identificação 6
Descrição do risco: Um (ou mais) dos integrantes da equipe sofrem de algum dano à saúde.	

2ª Etapa: **Avaliação do Risco**

Impacto: 2(médio/baixo) Explique: A indisponibilidade devido a problemas de saúde pode levar ao atraso do projeto.
Probabilidade: 2(média/baixa) Explique: O risco de ocorrer um problema de saúde grave é pequeno.

3ª Etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): Prevenir: Levar uma vida com hábitos saudáveis evitando o estresse ajuda a evitar problemas de saúde. Transferir:- Mitigar: Redistribuir as tarefas atrasadas e tentar cumpri-las da melhor forma possível. Aceitar: Contornar este problema, além disso, o trabalho tem que ser realocado.	
Impacto reavaliado: 2(médio/baixo)	**Probabilidade reavaliada:** 2(média/baixa)

Elaborado por: Leonardo	Data: 24/04/2013	Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou Anexos

Formulário sugerido por Gasnier, 2000 Editora IMAN e alterado por Wille