

EDGAR DE OLIVEIRA CABRAL FILHO

**COBERTURA POR VÉRTICES MÍNIMA EM GRAFOS LEI
DE POTÊNCIA**

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Renato José da Silva Carmo

Co-Orientador: Prof. Dr. André Luís Vignatti

CURITIBA

2016

EDGAR DE OLIVEIRA CABRAL FILHO

**COBERTURA POR VÉRTICES MÍNIMA EM GRAFOS LEI
DE POTÊNCIA**

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Renato José da Silva Carmo

Co-Orientador: Prof. Dr. André Luís Vignatti

CURITIBA

2016

EDGAR DE OLIVEIRA CABRAL FILHO

**COBERTURA POR VÉRTICES MÍNIMA EM GRAFOS LEI
DE POTÊNCIA**

Dissertação aprovada como requisito parcial à obtenção do grau de Mestre no Programa de Pós-Graduação em Informática da Universidade Federal do Paraná, pela Comissão formada pelos professores:

Orientador: Prof. Dr. Renato José da Silva Carmo
Departamento de Informática, UFPR

Prof. Dr. André Luís Vignatti
Departamento de Informática, UFPR

Prof. Dr. Elias Procópio Duarte Júnior
Departamento de Informática, UFPR

Prof. Dr. Murilo V. G. da Silva
Universidade Tecnológica do Paraná, UTFPR

Curitiba, 29/02/2016

AGRADECIMENTOS

Agradeço primeiramente minha mãe, Suely de O. Cabral, por dar-me o dom da vida e sempre estar ao meu lado em todos os momentos.

Agradeço também a minha família através do mais belo e novo integrante, meu primo Dimitri.

Ao meu orientador, prof^o. Dr. André Vignatti, pelos conselhos, orientações e por ter fé em mim.

Meus amigos do laboratório de algoritmos Alane Marie, Santiago Viertel e Renato Melo.

Agradeço também meus amigos do Mato Grosso que estão vivendo em Curitiba, em nome da amiga Elaine Rocha e Ivan Pires.

E aos Botecos, que aliviaram os momentos de angústias, trazendo a motivação e esperança para a realização desse trabalho.

SUMÁRIO

LISTA DE FIGURAS	vi
LISTA DE TABELAS	vii
RESUMO	viii
ABSTRACT	ix
1 INTRODUÇÃO	1
2 CONCEITOS PRELIMINARES	4
2.1 Cobertura por vértices	4
2.2 Grafos lei de potência	7
2.3 Tratabilidade por parâmetro-fixo	10
2.3.1 Regras de redução ao núcleo do problema	15
3 ALGORITMOS PROPOSTOS	17
3.1 Regras de redução ao núcleo para cobertura por vértices	17
3.2 Algoritmos Propostos	20
3.2.1 Formato das instâncias	23
4 RESULTADOS EXPERIMENTAIS	25
4.1 Cenário e grafos utilizados nos experimentos	26
4.1.1 ca-citeseer	26
4.1.2 ca-dblp-2012	26
4.1.3 ca-MathSciNet	26
4.1.4 ia-wiki-Talk	26
4.1.5 sc-nasasrb	27
4.1.6 soc-brightkite	27

4.1.7	soc-douban	27
4.1.8	soc-gowalla	27
4.1.9	socfb-Berkeley13	27
4.1.10	socfb-OR	27
4.1.11	socfb-Stanford3	28
4.1.12	socfb-UCLA	28
4.1.13	socfb-UCSB37	28
4.1.14	tech-RL-caida	28
4.1.15	web-sk-2005	28
4.2	Análise e discussão dos resultados	29
4.3	Redução do tamanho do grafo	31
4.4	Tamanho da cobertura por vértices encontrada	34
4.5	Tempo computacional	36
4.6	Conclusão da análise dos experimentos	39
5	CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS	41
	ANEXO 1	43
	ANEXO 2	49
	BIBLIOGRAFIA	55

LISTA DE FIGURAS

2.1	Possível cobertura por vértice num grafo qualquer.	5
2.2	Cobertura por vértice ótima num dado grafo	6
2.3	Classes P, NP, NP-Completo e NP-Difícil (supondo $P \neq NP$) [12].	6
2.4	Processo de surgimento de uma rede lei de potência, onde um novo vértice escolhe dois vértice já existente na rede. Fonte: Adaptado de [6]	8
2.5	Distribuição de grau do grafo <i>ia-wiki-Talk</i>	9
2.6	Distribuição de grau do grafo <i>ia-wiki-Talk</i> na escala log-log.	10
2.7	(a) Tempo de execução exponencial atinge toda a entrada do problema. (b) A parte principal contribui polinomialmente, enquanto a exponencialidade fica confinada ao parâmetro. Fonte: [15].	13
3.1	Um dado grafo com seus vértices v_{12} e v_{13} de grau zero selecionados.	18
3.2	Um dado grafo com seus vértices v_1 e v_{11} de grau um selecionados.	18
3.3	Grafo resultante após a aplicação da regra 2.	19
3.4	Um dado grafo com seus vértices de grau dois v_5 e seus vizinhos, v_7 e v_{11}	20
3.5	Grafo exemplo do arquivo no formato DIMACS 3.1	24
4.1	Grafo complexo (a) e grafo lei de potência (b).	29
4.2	Grafo <i>sc-nasasrb</i> . Fonte: [24]	33
4.3	Comparação entre as duas estratégias aplicadas nos grafos lei de potência - Redução de vértices	34
4.4	Comparação entre as duas estratégias aplicadas nos grafos lei de potência - Cobertura por vertices mínima	36
4.5	Tempo de execução, em grafos lei de potência, das estratégias discutidas neste trabalho.	39
5.1	Distribuição de grau do grafo <i>Brightkite_dges</i>	43
5.2	Distribuição de grau do grafo <i>CA-GrQc</i>	43

5.3	Distribuição de grau do grafo <i>CA-AstroPh</i>	44
5.4	Distribuição de grau do grafo <i>CA-CondMat</i>	44
5.5	Distribuição de grau do grafo <i>CA-HepTh</i>	44
5.6	Distribuição de grau do grafo <i>Email-Enron</i>	45
5.7	Distribuição de grau do grafo <i>soc-Epinions1</i>	45
5.8	Distribuição de grau do grafo <i>YeastS</i>	45
5.9	Distribuição de grau do grafo <i>sc-nasasrb</i>	46
5.10	Distribuição de grau do grafo <i>soc-brightkite</i>	46
5.11	Distribuição de grau do grafo <i>socfb-OR</i>	46
5.12	Distribuição de grau do grafo <i>ia-wiki-Talk</i>	47
5.13	Distribuição de grau do grafo <i>tech-RL-caida</i>	47
5.14	Distribuição de grau do grafo <i>soc-gowalla</i>	47
5.15	Distribuição de grau do grafo <i>ca-citeseer</i>	48
5.16	Distribuição de grau do grafo <i>ca-dblp-2012</i>	48
5.17	Distribuição de grau do grafo <i>ca-MathSciNet</i>	48
5.18	Distribuição de grau do grafo <i>Cit-HepTh</i>	49
5.19	Distribuição de grau do grafo <i>p2p-Gnutella04</i>	49
5.20	Distribuição de grau do grafo <i>p2p-Gnutella05</i>	50
5.21	Distribuição de grau do grafo <i>p2p-Gnutella06</i>	50
5.22	Distribuição de grau do grafo <i>p2p-Gnutella08</i>	50
5.23	Distribuição de grau do grafo <i>p2p-Gnutella09</i>	51
5.24	Distribuição de grau do grafo <i>p2p-Gnutella24</i>	51
5.25	Distribuição de grau do grafo <i>p2p-Gnutella25</i>	51
5.26	Distribuição de grau do grafo <i>p2p-Gnutella30</i>	52
5.27	Distribuição de grau do grafo <i>p2p-Gnutella31</i>	52
5.28	Distribuição de grau do grafo <i>socfb-Stanford3</i>	52
5.29	Distribuição de grau do grafo <i>socfb-UCSB37</i>	53
5.30	Distribuição de grau do grafo <i>socfb-UCLA</i>	53
5.31	Distribuição de grau do grafo <i>socfb-Berkeley13</i>	53

5.32	Distribuição de grau do grafo <i>web-sk-2005</i>	54
5.33	Distribuição de grau do grafo <i>soc-douban</i>	54

LISTA DE TABELAS

2.1	Relação dos algoritmos FPT para o problema da k -cobertura por vértices [20].	14
3.1	Exemplo de arquivo no format DIMACS	24
4.1	Instâncias utilizadas no trabalho de [27]	25
4.2	Classificação dos grafos do mundo real utilizados nos experimentos	30
4.3	Redução do tamanho dos grafos após aplicação das regras de redução - N.L.P	32
4.4	Redução do tamanho dos grafos após aplicação das regras de redução - L.P	33
4.5	Grafos do mundo real que não apresentam uma distribuição de grau lei de potência	35
4.6	Grafos do mundo real com distribuição de grau lei de potência	36
4.7	Tempo de execução dos grafos sem distribuição de lei de potência	38
4.8	Tempo de execução dos grafos sem distribuição de lei de potência	38

RESUMO

A teoria dos grafos é um ramo da matemática utilizada para modelar e representar um conjunto de elementos e suas relações, além de ser muito utilizada na resolução de problemas computacionais. Um grafo pode representar diversos sistemas naturais e digitais como ligações protéicas, redes sociais, conexões digitais, entre outras. Essas redes contêm diversas características, uma dessas é a distribuição de grau dos vértices. Muitos grafos do mundo real apresentam em sua estrutura uma distribuição de grau que segue uma lei de potência (*Power Law*), o que informalmente significa que existem poucos vértices de grau elevado, enquanto muitos vértices apresentam grau baixo. Dentro dos problemas clássicos algorítmicos, estamos interessados em problemas computacionalmente difíceis de serem resolvidos e que pertencem à classe NP-Difícil, especificamente o problema da cobertura por vértices mínima em grafos que apresentam uma distribuição de grau lei de potência. Assim, é apresentado neste trabalho um método inspirado em regras de redução ao núcleo do problema. Os resultados obtidos sugerem ser uma boa heurística de aproximação da solução ótima, além de reduzir significativamente o tempo computacional na resolução do problema da cobertura por vértices.

Palavras-chave: Grafos, Cobertura por Vértices, Regras de redução, Lei de Potência.

ABSTRACT

Graph theory is a branch of mathematics used to model and represent a set of elements and their relationships, which is also often used to solve computational problems. A graph can represent various natural and digital systems, such as: protein binding, social networks, digital connections, among others. Those networks contain diverse characteristics, which one of these is the degree of the distribution of the vertices. Many graphs of the real world have in their structure a degree of distribution following a power-law, where informally means that there are few high degree vertices, while many other vertices have a low degree. Within the algorithmic classic problems, we are interested in computationally difficult problems to be solved and which belong to the NP-Hard class, specifically the problem of minimum vertex cover in graphs, that have a power-law degree distribution. Thus, this work presents a method based on reduction rules to the core of the problem. The achieved results indicate that it is a good approximation heuristic from the optimal solution, in addition to be a technique that significantly reduces on the computational time to solve the problem of minimum vertex cover.

Keywords: Graph, Vertex Cover, reduction rules, Power Law.

CAPÍTULO 1

INTRODUÇÃO

A teoria dos grafos é ramo da matemática que estuda um conjunto de elementos e a relação entre esses elementos, podendo ser modelados com o objetivo de representar alguma estrutura de um problema do mundo real [12].

Grafos do mundo real, também chamados de redes complexas, foram foco de estudos na última década devido algumas propriedades comuns existentes em grafos de diferentes domínios. Trabalhos experimentais foram realizados nessa área por pesquisadores [6, 29, 11, 7, 2] a fim de entender a estrutura e propriedades dessas redes.

Algumas das propriedades encontradas em redes complexas são a distribuição de grau lei de potência, fechamento triádico, fenômeno do mundo pequeno, entre outros [6]. No caso da distribuição de grau lei de potência, nem todas as redes complexas contêm tal distribuição de grau em sua estrutura [29].

Grafos que contêm uma distribuição de grau com lei de potência podem ser facilmente observados em redes, como a WWW, redes sociais, citações em artigos, entre outros [6]. Com relação à distribuição dos graus, podemos informalmente dizer que em tais grafos, poucos dos seus vértices contêm grau alto (*hubs*), enquanto em vários outros vértices o grau é extremamente baixo.

Dentre os problemas que podem ser resolvidos utilizando grafos, estão os problemas de otimização. Problemas de otimização são amplamente estudados devido à sua importância em aplicações práticas. Alguns destes problemas pertencem à classe de complexidade NP-Difícil e o grau de dificuldade em conseguir resolvê-los de forma ótima é alto (supondo $P \neq NP$). Como exemplos clássicos de problemas de otimização podemos citar o problema do caixeiro viajante, o problema da mochila, o problema da floresta de *Steiner* e o problema da satisfatibilidade máxima [12].

Os problemas pertencentes a classe NP-Difícil tornaram-se objeto de estudo dentro

da teoria da complexidade parametrizada. A teoria da complexidade parametrizada [15] estuda problemas considerados intratáveis, no que se refere a complexidade clássica computacional, remodelando a abordagem no tratamento deste problemas difíceis. A remodelagem que esta teoria traz remove parcialmente a exponencialidade da complexidade do problema original, confinando a exponencialidade a um parâmetro contido na entrada do problema, ao invés da exponencialidade ser em relação à todo o tamanho da entrada, permitindo assim soluções exatas com baixo custo computacional.

Neste trabalho, estudamos o problema da *cobertura por vértices mínima*, que busca encontrar a menor cobertura por vértices, dentro de um grafo com distribuição de grau lei de potência (*Power Law*). Tal problema faz parte da classe NP-Difícil, apresentado por *Karp* [22] em 1972. Na literatura encontram-se algoritmos heurísticos e aproximados para resolver este problema da cobertura por vértices mínima [19, 21, 4, 18], abordando estratégias diferenciadas, tais como algoritmos gulosos, regras de redução ao núcleo do problema, programação linear [1], entres outros.

Em [1], os autores apresentam alguns algoritmos que resolvem o problema da cobertura por vértices, além de comparar a eficiência do tempo de execução de cada estratégia apresentada. Os autores [15, 14, 16, 1] propõem ainda a utilização de regras de redução ao núcleo antes da utilização de qualquer estratégia mencionada, uma vez que o uso de tais regras reduzem significativamente o tamanho do grafo original.

Neste trabalho experimental temos como objetivo resolver o problema da cobertura por vértices mínima utilizando regras de redução ao núcleo em grafos lei de potência antes da utilização da estratégia gulosa apresentada por da Silva et. al. [13], a qual mostrou experimentalmente um resultado médio de 2% acima da solução ótima.

Os experimentos foram executados em 33 grafos do mundo real, sendo que 18 foram retirados da mesma base de dados do trabalho de da Silva et. al [13]. Do total de instâncias utilizadas, 16 grafos não apresentaram distribuição lei de potência. E desta forma foi possível observar os resultados para grafos com e sem distribuição lei de potência, algo que não havia sido feito por da Silva et. al. [13].

Obtivemos uma diminuição, utilizando as regras de redução antes da aplicação do

algoritmo guloso, de 4,16% no tamanho da cobertura por vértices e uma melhora de aproximadamente 50% no tempo computacional.

Este trabalho está organizado em 5 Capítulos: o Capítulo 2 apresenta a teoria e os conceitos necessários que fundamentam o trabalho; o Capítulo 3 contém a descrição das metodologias aplicadas nesta pesquisa; o Capítulo 4 apresenta a análise dos resultados obtidos; e por fim, o Capítulo 5 contém as considerações finais desta pesquisa e trabalhos futuros.

CAPÍTULO 2

CONCEITOS PRELIMINARES

Neste Capítulo são apresentados conceitos necessários para o entendimento do problema proposto neste trabalho, além de apresentar as técnicas que nos baseamos para o desenvolvimento da nossa heurística. Estas técnicas são geralmente estudadas dentro do contexto de tratabilidade por parâmetro-fixado mas, em alguns problemas de otimização podem ser utilizadas.

2.1 Cobertura por vértices

A cobertura por vértices é um importante problema com muitas aplicações no mundo real, tais como: segurança de rede, exemplos didáticos de programação e criação de circuito integrado [9]. Podemos definir uma *cobertura por vértices* como na Definição 1.

Definição 1 *Dado um grafo não direcionado $G = (V, E)$, uma cobertura por vértices é um subconjunto de vértices S tal que, para cada aresta $(u, v) \in E$, $u \in S$, ou $v \in S$ (ou ambos). O tamanho da cobertura por vértices é o número de vértices em S .*

A Figura 2.1 apresenta uma possível cobertura por vértices de um grafo dado. Os vértices, v_2 , v_3 , v_4 e v_6 formam o conjunto da cobertura por vértices.

Como podemos observar na Figura 2.1, tal cobertura é um conjunto de vértices na qual cada aresta do grafo é incidente a pelo menos um vértice do conjunto. As duas versões mais conhecidas para o problema da cobertura por vértices são a de decisão e a de minimização (otimização), conforme definido pelos Problemas 1 e 2.

Problema 1 (Cobertura por vértices - decisão) *Dado um grafo G e um número k inteiro e positivo, existe uma cobertura por vértices de tamanho no máximo k em G ?*

Um problema que tem como solução uma resposta binária, no caso “Sim” ou “Não”, é chamado de *problema de decisão* [12]. O Problema 1 define o problema da cobertura por

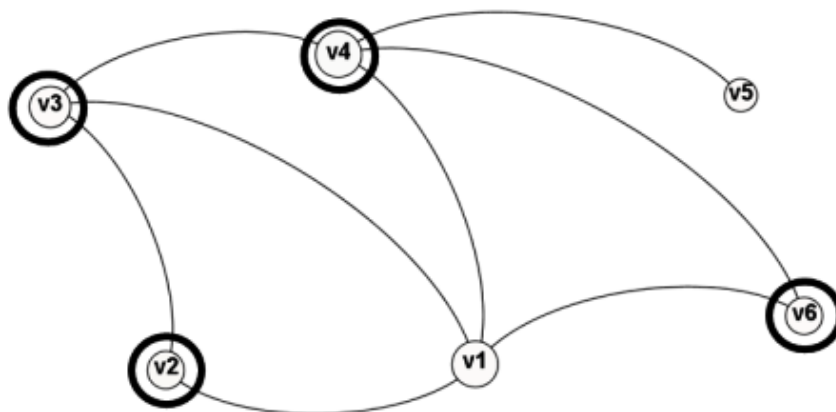


Figura 2.1: Possível cobertura por vértice num grafo qualquer.

vértices como um problema de decisão.

Na complexidade computacional um subconjunto da classe NP é o conjunto NP-Completo, que informalmente são problemas pelo menos tão difíceis quanto qualquer outro problema na classe NP [12], o Problema 1 é um problema NP-completo [22].

Este problema pode ser modelado utilizando como exemplo o monitoramento dos corredores de um museu. Neste caso, gostaríamos de saber se com no máximo k seguranças, todos os corredores estejam vigiados (pensando nos corredores como arestas e seus pontos de ligação como vértices e os seguranças podem ser alocados nos vértices, desta forma vigiando os corredores adjacentes).

Quando se procura uma solução no conjunto de soluções viáveis que seja o melhor possível em relação a algum critério, temos então um problema de otimização [17].

O Problema 2 a seguir, define o problema da cobertura por vértices na “versão” de problema de otimização.

Problema 2 (Cobertura por vértices - mínimo) *Dado um grafo G , qual é a menor cobertura por vértices deste grafo?*

Uma cobertura de vértices de tamanho mínimo, é chamada de *cobertura ótima*. Por exemplo, Na Figura 2.2, os vértices $v1$, $v2$ e $v5$ formam uma cobertura por vértices ótima do mesmo grafo apresentado pela Figura 2.1.

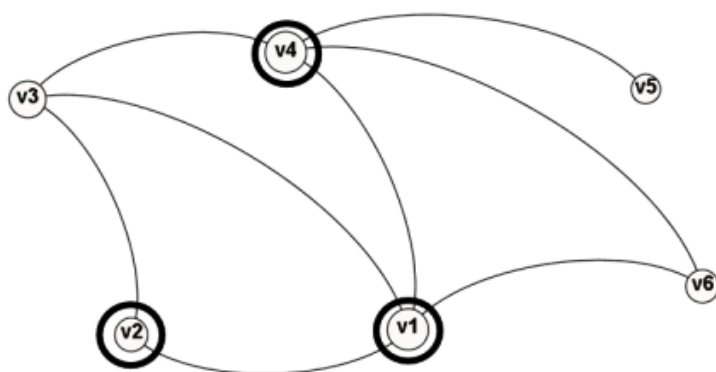


Figura 2.2: Cobertura por vértice ótima num dado grafo

O problema da cobertura por vértices mínima é NP-difícil [3, 22, 12].

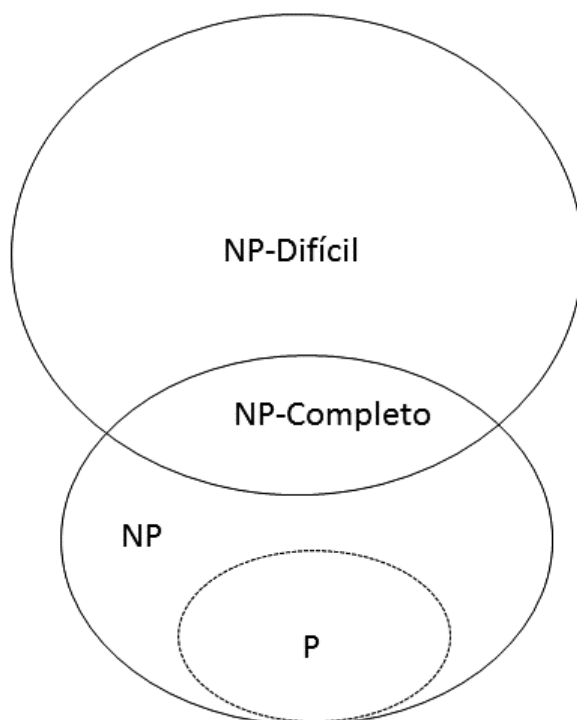


Figura 2.3: Classes P, NP, NP-Completo e NP-Difícil (supondo $P \neq NP$) [12].

Algoritmos para o problema de otimização se dividem em dois tipos: *algoritmos exatos* e *algoritmos heurísticos*.

- **Algoritmos exatos:** incluem algoritmos *branch-and-bound* [12], que garantem a otimalidade das soluções encontradas, mas nem sempre dão uma solução num prazo razoável de tempo para grandes instâncias, já que no pior caso executam em tempo

exponencial.

- **Algoritmos heurísticos:** incluem diversas estratégias, sendo que esta estratégia não garante a otimalidade de suas soluções, mas encontram soluções quase ótimas ou satisfatórias para instâncias grandes e difíceis, dentro de um tempo razoável [9].

O foco desta pesquisa é trabalhar com uma solução heurística da versão de otimização do problema da cobertura por vértices melhorando os resultados de da Silva et. al. [13].

2.2 Grafos lei de potência

O aumento no estudo de redes do mundo real traz pesquisas realizadas por cientistas da área da física, matemática, ciência da computação, entre outros, com o objetivo de entender a estrutura dessas redes complexas [2].

Muitos trabalhos experimentais foram realizados nas chamadas *redes complexas*, como também foram desenvolvidos modelos analíticos pela comunidade da matemática e ciência da computação teórica para explicar diferentes fenômenos que ocorrem nessas redes [13].

Redes de diferentes domínios, tais como sociais, econômicas, tecnológicas, biológicas, telecomunicações, entre outras, compartilham muitas vezes algumas propriedades comuns. Dentre essas, podemos citar o “fenômeno do mundo pequeno”, agrupamento, fechamento triádico e distribuição de graus de lei de potência [13].

Muitos dos pesquisadores de redes complexas vêm explorando as propriedades de grafos com distribuição lei de potência através de algoritmos, como por exemplo, a melhoria do fator de aproximação de problemas NP-Difíceis [13, 18, 4, 26].

O estudo de grafos com distribuição de grau de lei de potência começou na década de 90 com trabalhos de Kumar e Kleinberg [23] ao medirem a distribuição de grau da *World Wide Web*. Esses autores, observaram de forma independente que a sequência de grau é bem próxima a uma distribuição de lei de potência.

Uma propriedade respeita a lei de potência (*power law*) se a relação entre dois números y e i é tal que ela possa ser escrita como $y = ai^{-\beta}$, onde a e β são constantes reais. No

contexto da teoria dos grafos podemos definir grafos com distribuição de lei de potência como na Definição 2.

Definição 2 *Um grafo respeita a distribuição de lei de potência se a fração de vértices com grau i é igual a $ci^{-\beta}$ para algum α e β constante e maior que 1.*

Estudos apontam que o fator β varia entre 1 e 4 para a maioria das redes reais [26]. Esta distribuição foi confirmada mais tarde para um grande número de redes complexas do mundo real, tais como: ligações de proteínas, redes ponto-a-ponto, redes de chamadas telefônicas, entre outras [19].

O processo de surgimento de uma rede lei de potência, chamado de “*preferential attachment* (conexão preferencial)” [6], ocorre conforme a Figura 2.4, onde a rede inicia com um conjunto de 3 vértices e continuamente, novos vértices são adicionados e a rede começa a se expandir. Estes novos vértices se conectam com vértices com mais conexões, ou seja, quanto maior for o grau de um vértice, maior é sua chance dele receber uma nova conexão [6, 5].

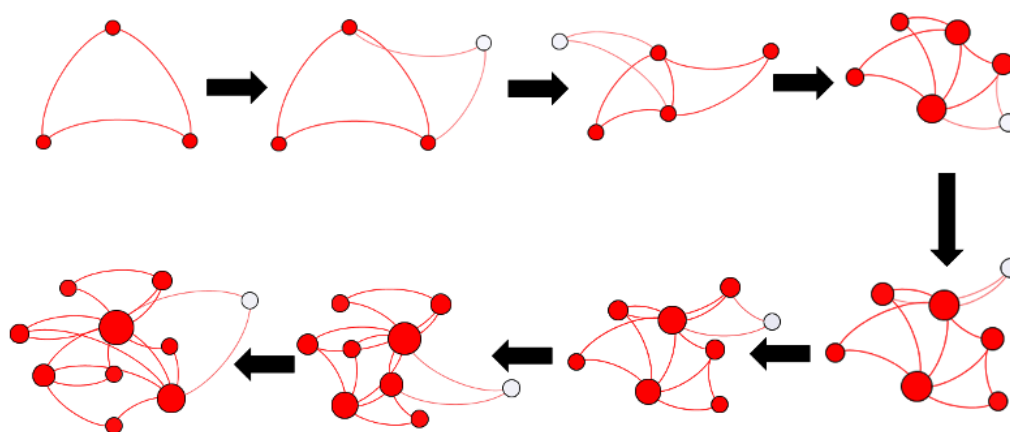


Figura 2.4: Processo de surgimento de uma rede lei de potência, onde um novo vértice escolhe dois vértice já existente na rede. Fonte: Adaptado de [6]

Assim, poucos vértices concentram uma grande quantidade de conexões enquanto vários outros vértices concentram poucas conexões. Mesmo ao considerarmos a diversidade das redes, em sua maior parte, essa propriedade está presente em sua estrutura.

Ao plotarmos a distribuição de grau de um grafo lei de potência, obtemos um gráfico com uma distribuição de cauda longa (*Long-tail distribution*), onde encontram-se os hubs

da rede.

A detecção de uma distribuição de grau lei de potência não é simples, pois existem variações que ocorrem na distribuição da cauda longa ¹ [11].

A Figura 2.5 apresenta a distribuição de grau do grafo *ia-wiki-Talk* retirado da base de dados [25].

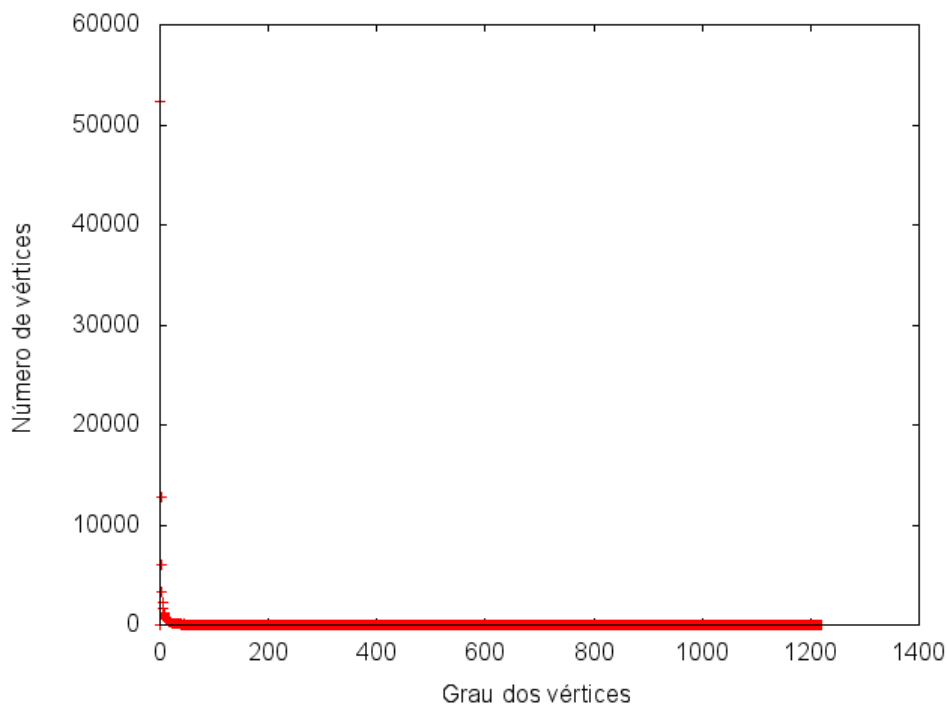


Figura 2.5: Distribuição de grau do grafo *ia-wiki-Talk*.

Podemos notar a presença da cauda longa no gráfico acima, porém, não é aparente a lei de potência, para isso temos que olhar sua distribuição na escala log-log e aplicar algum método que verifica a existência da lei de potência na rede. Um método possível é realizar uma regressão linear dos pontos no gráfico log-log, determinando assim a inclinação da reta e verificando se a regressão linear ocorreu com baixa taxa de erro. Caso tenha alta taxa de erro, podemos classificar a rede como não lei de potência. Autores como Cluset et. al. [11] e Virkar et. al [29] apresentam métodos mais elaborados que garantem maneiras mais eficientes de identificar se um grafo contém distribuição lei de potência.

A Figura 2.6 apresenta os dados da Figura 2.5 com sua distribuição de grau plotada

¹Informalmente, cauda longa é um termo utilizado para identificar distribuições de dados (curva de Pareto), no qual o volume de dados encontra-se de forma decrescente.

na escala log-log.

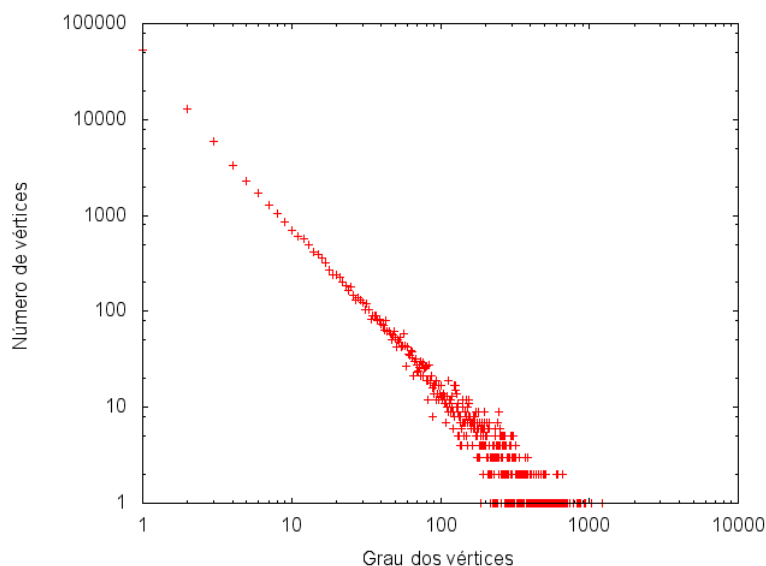


Figura 2.6: Distribuição de grau do grafo *ia-wiki-Talk* na escala log-log.

Na Figura 2.6 podemos notar uma linha reta na distribuição de grau do grafo *ia-wiki-Talk*. Os grafos utilizados nos experimentos deste trabalho cuja distribuição de grau aproximam-se da Figura 2.6 foram classificados como grafos lei de potência.

Na literatura [26, 19, 18, 2], muitos resultados experimentais sobre grafos aleatórios com lei de potência sugerem que os problemas podem ser mais facilmente resolvidos por causa da característica da distribuição de grau dos vértices, onde poucos vértices contêm graus extremamente altos, enquanto muitos outros contêm grau baixo.

Com base nessas evidências experimentais, o nosso foco está em utilizar grafos lei de potência e tratá-los a partir da aplicação de regras de redução ao núcleo do problema na resolução do problema da cobertura por vértices mínima.

2.3 Tratabilidade por parâmetro-fixo

Para iniciarmos a discussão sobre tratabilidade por parâmetro-fixo, vamos primeiramente apresentar a Definição 3, que explica o que é um problema intratável.

Definição 3 (Problema intratável) *um problema é intratável se não admite um algoritmo polinomial uma vez que existem instâncias que não podem ser resolvidas em tempo*

polinomial;

A teoria da complexidade parametrizada introduz a noção de tratabilidade por parâmetro fixo na remodelagem de alguns problemas difíceis, removendo a exponencialidade da complexidade do problema que era em relação ao tamanho da entrada e confinando a exponencialidade no tamanho do parâmetro. O parâmetro é uma parte da entrada. O uso desta teoria permite a obtenção de soluções exatas com baixo custo computacional.

Nesta teoria, o objetivo não está em estudar se um determinado problema é difícil, o objetivo está na questão do que torna o problema difícil computacionalmente [15, 10, 14]. Desta forma, parte-se do pressuposto que problemas são *intratáveis* quando considerados em seu contexto clássico da complexidade computacional, mas podem ser tratáveis se considerarmos algum certo parâmetro que não seja o tamanho da entrada [14].

Na complexidade clássica, o problema de decisão é especificado por dois itens de informação:

1. A entrada do problema.
2. A questão a ser respondida.

Já na complexidade parametrizada, existem três partes que especificam o problema [14]:

1. A entrada do problema.
2. Os aspectos da entrada que constituem o *parâmetro*.
3. A questão a ser respondida.

O parâmetro de um problema está relacionado à natureza do problema a ser resolvido, por exemplo:

- Problema de alocação de recursos numa rede de tamanho n , cuja a quantidade a ser alocada deve ser limitada. O parâmetro k representa então a quantidade de recursos a serem alocados, e escala com o crescimento da rede dado que a exponencialidade do problema está amarrada unicamente a quantidade de recursos e não ao tamanho da rede.

- Problema da cobertura por vértices de um grafo com tamanho n . O parâmetro k representa a quantidade de vértices pertencentes à cobertura.
- Em um problema envolvendo um conjunto de objetos geométricos, pode-se parametrizar pelo número máximo de vértices do objeto ou a dimensão do espaço onde o problema é definido.
- Um problema envolvendo um conjunto de strings pode ser parametrizado pelo tamanho máximo da string, pelo tamanho do alfabeto, pelo número máximo de símbolos distintos que apareçam na string, entre outros.

É importante notar que um parâmetro pode ser um agregado de informações. O parâmetro fornece uma maneira sistemática de especificar restrições da distribuição, na esperança de um “acordo bem sucedido com o diabo” [15]. Este acordo procura entender como as diferentes partes da entrada contribuem para a complexidade total do problema, gerando um “bom” e “mau” comportamento. Por “bom” comportamento, entende-se que a parte principal da entrada é resolvida em tempo polinomial, enquanto um “mau” comportamento é a parte mais difícil da instância, na qual não há como fugir do tempo de execução exponencial.

Desta forma, é possível desenvolver algoritmos que gastam tempo polinomial em relação à parte principal da entrada, restringindo a exponencialidade somente no parâmetro e não na instância toda. No entanto, para ser útil na prática, o parâmetro deve estar confinado num intervalo pequeno e útil.

A Figura 2.7 exemplifica o “acordo bem sucedido com o diabo”.

A Definição 4 apresenta a definição formal de um problema parametrizável, e fica evidente a separação entre parte principal da entrada e o parâmetro.

Definição 4 (Problema parametrizável) [20] *Um problema parametrizável é um conjunto $L \subseteq \Sigma^* \times \Sigma^*$, onde Σ é um alfabeto fixo. Se o par $(x, y) \in L$, então chamamos x de instância (ou parte principal) do problema parametrizável e y de parâmetro do problema.*

Para formalizar a noção de exponencialidade confinada ao parâmetro, temos a Definição 5:

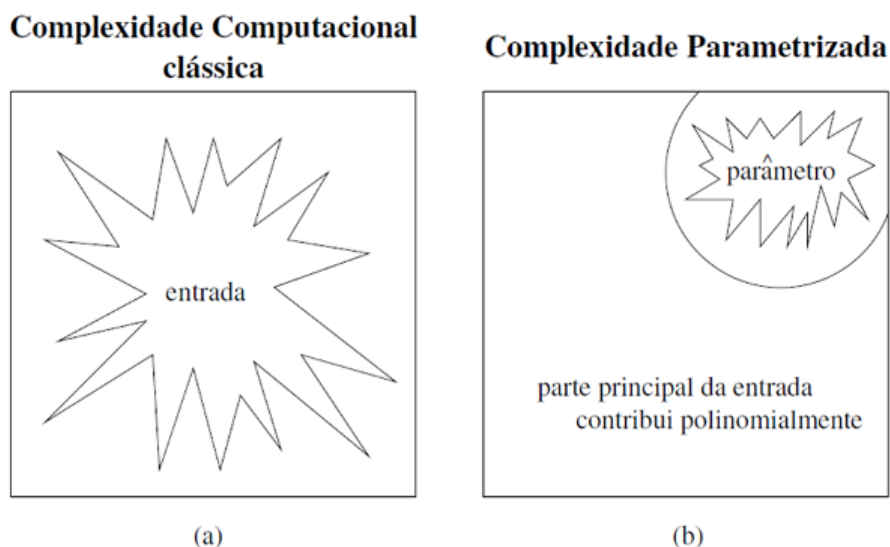


Figura 2.7: (a) Tempo de execução exponencial atinge toda a entrada do problema. (b) A parte principal contribui polinomialmente, enquanto a exponencialidade fica confinada ao parâmetro. Fonte: [15].

Definição 5 (Problema tratável por parâmetro-fixado [14]) *Um problema parametrizável $L \subseteq \Sigma^* \times \Sigma^*$ é tratável por parâmetro-fixado se existe um algoritmo que, dada uma entrada $(x, y) \in L$, resolve o problema em tempo $O(f(k)n^\alpha)$, onde n é o tamanho da parte principal da entrada x , ou seja, $|x| = n$; k é o tamanho do parâmetro y , ou seja, $|y| = k$, α é uma constante (independente de k), e f é uma função arbitrária.*

O que diz respeito a generalizar a noção clássica de tempo polinomial versus tempo exponencial, pela definição acima é: (1) a função arbitrária $f(k)$ pode contribuir de forma exponencial para a complexidade total do problema. (2) a parte principal da entrada do problema, contribui polinomialmente na complexidade total com n^α , conforme a complexidade clássica, onde o valor de α é constante. (3) A conexão entre estas duas contribuições é especificada pela multiplicação, gerando uma complexidade total de $f(k)n^\alpha$.

A propriedade (3) acima pode ser representada sem ser inalterada por, $f(k) + n^\alpha$, uma vez que executamos primeiramente as regras de redução ao núcleo e após a redução da instância, chamamos um algoritmo guloso para finalizar o processo. Esta é uma interessante e fundamental propriedade da tratabilidade por parâmetro-fixado, a qual permite mostrar que a conexão entre as duas contribuições seja aditivas [14].

Para ter uma parametrização bem-sucedida de um problema, duas propriedades devem ser satisfeitas:

1. Deve-se ter alguma razão para crer que o parâmetro selecionado seja “pequeno” em relação ao tamanho das instâncias de alguma aplicação;
2. Precisa-se de algoritmos eficientes, onde a o tempo de execução exponencial é restrito ao parâmetro, ou seja, o problema deve ser *Tratável por parâmetro-fixo (FPT)*, através dessa parametrização [20];

Os algoritmos que resolvem os problemas da classe FPT são chamados *algoritmos FPT*, ou seja, são algoritmos que resolvem os problemas tratáveis por parâmetro-fixo.

Em [15] encontra-se vários problemas parametrizáveis, além do problema da cobertura por vértices - decisão (conhecido como *k-cobertura por vértices*). O problema da cobertura por vértices - decisão foi um dos primeiros problemas provados a pertencerem a classe dos problemas tratáveis por parâmetro-fixo. Os algoritmos *FPT* tem sido utilizado em grandes instâncias de importância prática, uma vez que estas mesmas instâncias eram muito grandes para serem resolvidas pelos métodos existentes [14].

Na Tabela 2.1 constam os trabalhos que desenvolveram algoritmos FPT para o problema da k-cobertura por vértices.

Tabela 2.1: Relação dos algoritmos FPT para o problema da *k*-cobertura por vértices [20].

Autor(es)	Complexidade	Ano
Buss	$O(kn + 2^k k^{2k+2})$	1993
Downey e Fellows	$O(kn + 2^k k^2)$	1995
Balasubramanian et al	$O(kn + (\sqrt{3})^k k^2)$	1998
Balasubramanian et al	$O(kn + 1.324718^k k^2)$	1998
Downey, Fellows e Stege	$O(kn + 1.31951^k k^2)$	1999
Niedermeider e Rossmanith	$O(kn + 1.29175^k k^2)$	1999
Fellows e Stege	$O(kn + \max(1.25542^k k^2, 1.2906^k 2.5^k))$	1999
Chen, Jia e Kanj	$O(kn + 1.271^k k^2)$	2001

O parâmetro do problema da k-cobertura por vértices é o tamanho da cobertura. Logo a exponencialidade é permitida somente sobre o tamanho *k* da cobertura, conforme a Tabela 2.1. Existem dois métodos que são mais utilizados ao projetar algoritmos *FPT*,

a *árvore limitada de busca* (*bounded search tree*), e a *redução ao núcleo do problema* (*reduction to a problem kernel*). Em [15, 10] estes métodos são explicados, como se segue:

- **Redução ao núcleo do problema:** Este método tem como objetivo reduzir, em tempo polinomial uma instância I de um problema parametrizável, para uma instância I' , onde seu tamanho é limitado por uma função do parâmetro k . Se uma solução de I' é encontrada, então ela pode fazer parte da solução de I . O uso desse método contribui de forma aditiva para a complexidade total do problema ($f(k) + n^\alpha$).
- **Árvore limitada de busca:** Este método utiliza tentativas seguidas para resolver o problema por pesquisa exaustiva, requerendo tipicamente um tempo exponencial no parâmetro k [10]. Não focamos nesta técnica, pois não é o objetivo deste trabalho.

A questão da parametrização não está se ela pode ser resolvida por parâmetro-fixo, mas se ela pode ser resolvida por algoritmos melhores do que os de força bruta [15].

2.3.1 Regras de redução ao núcleo do problema

A *redução ao núcleo* (*kernelization*) é uma técnica utilizada para desenvolver algoritmos eficientes e, muitas vezes, é conseguida através da aplicação de um conjunto de regras de pré-processamento que eliminam partes da instância que são consideravelmente fáceis de manusear, deixando apenas um “*kernel*” (núcleo) a ser tratado [1]. O núcleo está relacionado de alguma forma ao parâmetro-fixo escolhido para o problema, de maneira que tratar o núcleo toma tempo exponencial.

Podemos definir uma redução ao núcleo por:

Definição 6 (Redução ao núcleo [15]) *Conjunto finito de regras de redução, que, ao serem aplicadas (em alguma ordem específica) em uma dada instância (x, y) de um problema parametrizado L , devolvem uma instância (x', y') de L tal que $|x'| \leq g(p)$ e $y' \leq g(p)$, para alguma função computável $g : \mathbb{Z} \rightarrow \mathbb{Z}$. Chamamos x' de núcleo da função e a função g de tamanho do núcleo.*

Em suma, as regras de redução trabalham utilizando parâmetros de entrada que um algoritmo que resolve o problema receberia, reduzindo a instância de tal forma que ao aplicar regras de redução, a obtenção da solução ótima não é comprometida, pois as regras são projetadas com esse objetivo (caso contrário, não podemos realizar reduções na instância).

Problemas que são tratáveis através de parâmetros-fixos, utilizam algoritmos para diminuir a dificuldade em encontrar soluções eficientes em problemas NP-difíceis, garantindo uma solução viável além da redução do tempo gasto na resolução do problema [1].

Um exemplo da utilização de regras de redução ao núcleo do problema pode ser encontrado no algoritmo de Buss [8] de 1993. O algoritmo elaborado pelo autor para o problema da k -cobertura por vértices utiliza dois métodos de redução ao núcleo do problema, o primeiro remove os vértices de grau maior do que k (problema de decisão) do grafo G , além das arestas incidentes a estes vértices; a segunda regra de redução é a eliminação dos vértices de grau zero.

Buss [8] baseia-se na ideia de que todos os vértices de tamanho maior que o parâmetro k , pertencem a alguma cobertura por vértice de tamanho menor ou igual que k . Logo, existindo mais de k vértices com grau maior do que k , não existe uma cobertura por vértices. Este algoritmo tem custo computacional de $O(k_n + 2^k k^{2k+2})$.

Neste trabalho está descrita a utilização de algumas regras de redução ao núcleo do problema, para o problema da cobertura por vértices mínima, a fim de reduzir o tamanho da entrada original do problema, deixando apenas um núcleo a ser tratado por um algoritmo exponencial sem escapar da solução ótima.

Estas regras de redução são pensadas para problemas de decisão, mas que em alguns casos (como as regras descritas no Capítulo 3) podem ser usadas em problemas de otimização, pois a redução é feita selecionando partes da solução que sabidamente fazem parte da solução ótima.

CAPÍTULO 3

ALGORITMOS PROPOSTOS

São apresentadas neste capítulo as regras de redução ao núcleo do problema propostas para o problema da cobertura por vértices mínima, e também os algoritmos propostos para resolver o problema.

3.1 Regras de redução ao núcleo para cobertura por vértices

As regras de redução ao núcleo mostradas a seguir foram apresentadas por Faisal et. al [1]. Estas regras exigem custo $O(n^2)$, aplicadas num grafo não direcionado com n vértices.

Regra 1: Existindo um *vértice isolado*, ou seja, de grau 0, ele não pode estar na cobertura de vértices ótima, uma vez que não existe nenhuma aresta incidente em cima de tal vértice, então não há nenhum benefício em incluí-lo na cobertura. Seja C o conjunto de todos os vértices isolados de G , o tamanho do grafo passa a ser $n = n - |C|$, ou seja, n é reduzido pela quantidade de vértices isolados do grafo. Aplica-se esta regra até que não exista nenhum vértice de grau 0. Por exemplo, os vértices selecionados, v_{12} e v_{13} na Figura 3.1, representam estes vértices de grau 0 que serão removidos.

Regra 2: Dado um *vértice de grau 1*, existe uma cobertura ótima que não contém este vértice, mas contém o seu vizinho. Seja o vértice u de grau 1 e seu vizinho v , a aresta uv estará coberta, caso o vértice u , ou v ou ambos estejam na cobertura. Mas uma cobertura ótima (neste caso) é constituída somente por um dos dois vértices, u ou v . Caso a cobertura ótima seja constituída pelo vértice u , não há problema em trocá-lo pelo seu vizinho (vértice v), uma vez que o tamanho da cobertura ótima continuará a mesma. Repete-se esta regra até que todos os vizinhos dos vértices de grau 1 sejam inseridos na cobertura e suas arestas incidentes removidas. Esta regra pode gerar novos vértices isolados que são eliminados sobre a regra 1, reduzindo assim n pelo número de vértices removidos pela regra 1 mais os vértices inseridos na cobertura.

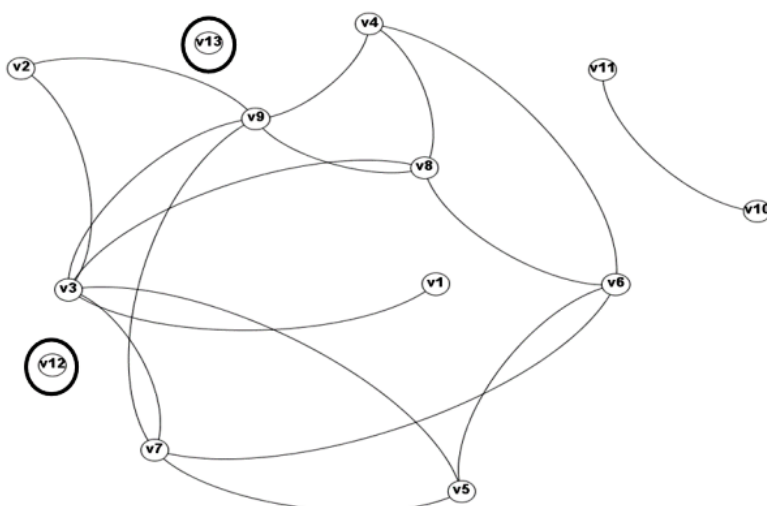


Figura 3.1: Um dado grafo com seus vértices v_{12} e v_{13} de grau zero selecionados.

Seja S o conjunto formado por todos os vértices inseridos na cobertura (na aplicação das regras de redução), e C o conjunto de todos os vértices isolados de G , após a aplicação da regra 2, o grafo resultante passa a ter o tamanho de $n = n - |C| - |S|$, isto após a primeira iteração. Na prática essa regra é executada até não existir vértices de grau 1.

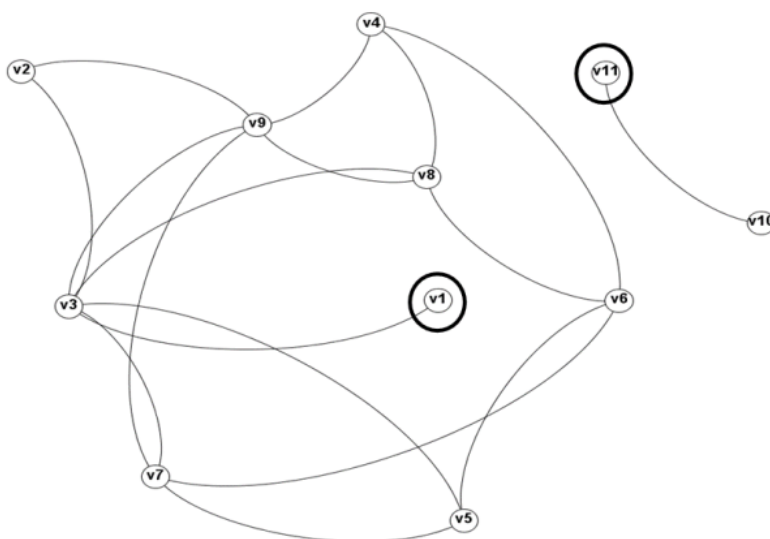


Figura 3.2: Um dado grafo com seus vértices v_1 e v_{11} de grau um selecionados.

Na Figura 3.2, os vértices selecionados, v_1 , v_{11} e v_2 (na 2ª execução da regra) foram selecionados pela regra 2, então, seus vizinhos v_3 , v_{10} e v_9 (primeira, segunda e terceira execução da regra) são inseridos na cobertura ótima. Na Figura 3.3, é simulado o resultado

do grafo após aplicadas a regra 1 e regra 2.

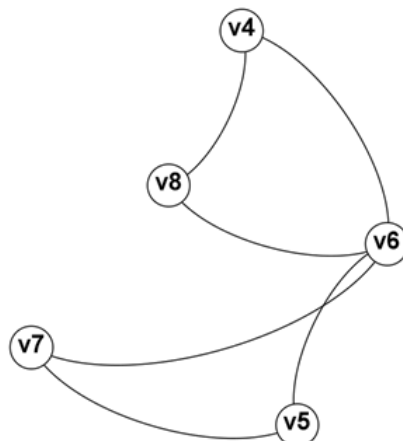


Figura 3.3: Grafo resultante após a aplicação da regra 2.

Note que na Figura 3.3 o tamanho do grafo original foi significativamente reduzido aplicando apenas duas regras de redução, e que os vértices inseridos na cobertura são vértices pertencentes a alguma cobertura ótima.

Regra 3: Se existe um vértice u de grau dois, com vizinhos v e w adjacentes, então há uma cobertura ótima que inclui ambos os vizinhos deste vértice. A Figura 3.4 apresenta a aplicação da regra 3. Se u (v_5) é um vértice de grau 2 e v e w (v_7 e v_{11} respectivamente) são os seus vizinhos, para cobrir as arestas, (u, v) , (u, w) e (v, w) de forma ótima, dois dos três vértices devem estar na cobertura (menos que dois vértices não seria uma cobertura válida, como também, mais do que dois vértices não seriam uma cobertura mínima). Ao escolher os vértices u e v , todas as arestas estarão cobertas, o mesmo se aplica ao escolher os vértices u e w . Mas, trocando o vértice u por um de seus vizinhos, v ou w (caso algum não tenha sido escolhido para a cobertura), o tamanho da cobertura ótima não modifica (continua sendo formada por dois vértices), porém caso v e/ou w contenham vizinhos além do vértice u , as arestas incidentes entre v e w e seus vizinhos também estarão cobertas. Logo existe uma cobertura por vértices ótima que inclui v e w , mas não u . Isto ainda reduz n ao número de vértices inseridos na cobertura e os removidos pela regra 1 e 2. Aplica-se esta regra até que todos os vértices de grau 2 com vizinhos adjacentes sejam inseridos na cobertura, após a inserção dos vértices de grau 2, aplica-se a regra 2 e por fim a regra 1.

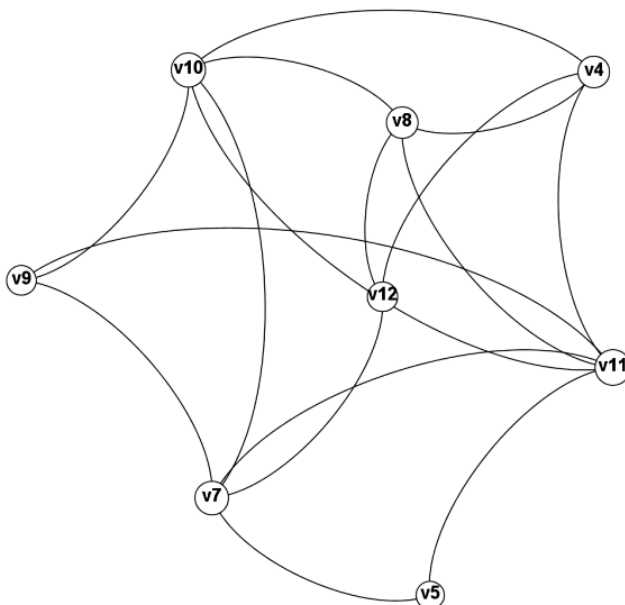


Figura 3.4: Um dado grafo com seus vértices de grau dois v_5 e seus vizinhos, v_7 e v_{11} .

Após aplicadas estas regras de redução, o grafo original G é reduzido. Estas regras facilitam na resolução do problema da cobertura por vértices mínima, pois além de diminuir o tamanho do instância original [1], garantem não “desviar” da cobertura ótima. Assim, é ideal a utilização destas regras de redução, na cobertura por vértices mínima em grafos com características de lei de potência, uma vez que este tipo de rede contém em sua estrutura muitos vértices de grau 1 e 2.

3.2 Algoritmos Propostos

Dois algoritmos são descritos nesta seção, o algoritmo ED-Regras que baseia-se nas regras de redução ao núcleo do problema apresentados na Seção 3.1 (usado como sub-rotina) e o algoritmo guloso descrito por [27].

O algoritmo *ED-Regras* recebe como entrada um grafo $G = (V, E)$ não direcionado e retorna um conjunto S contendo uma cobertura por vértices, além de retornar o grafo reduzido (este algoritmo é uma sub-rotina de um algoritmo maior).

Conforme discutido no Capítulo 2, as regras de redução devem seguir uma ordem pré-estabelecida. Como os grafos complexos apresentam diversos vértices de grau 1, no algoritmo ED-Regras a regras 2 torna-se a primeira regra a ser aplicada na execução do

algoritmo, seguido da regra 1 (remoção dos vértices isolados) uma vez que surgem vértices de grau 0.

Ao final do primeiro processo, parte-se para a execução da regra 3 busca por triângulos no grafo. A cada iteração da regra 3 as regras 2 e regra1 são executadas novamente.

Algorithm 3.1: ALGORITMO ED-REGRAS

```

1 Entrada:  $G = (V, E)$  não direcionado, onde  $V$  representa o conjunto de vértices e
    $E$  o conjunto de arestas
2 Saída: Cobertura por vértices de  $S$ , grafo  $G$  reduzido
3 início
4    $S \leftarrow \emptyset$ 
5   enquanto  $\exists u \in V | \delta(u) = 1$  faça
6     Seja  $\{u, v\} \in E$ 
7      $S \leftarrow S \cup \{v\}$ 
8      $E \leftarrow E - S_v$ 
9      $V \leftarrow V - \{w \in V | \delta(w) = 0\}$ 
10  fim
11  enquanto  $\exists u \in V | \delta(u) = 2$  faça
12    Seja  $\{\{u, v\}, \{u, w\}\} \subseteq E$  e  $v \neq w$ 
13    se  $\{v, w\} \in E$  então
14       $S \leftarrow S \cup \{v, w\}$ 
15       $E \leftarrow E - S_v - S_w$ 
16    fim
17    enquanto  $\exists u \in V | \delta(u) = 1$  faça
18      Seja  $\{u, v\} \in E$ 
19       $S \leftarrow S \cup \{v\}$ 
20       $E \leftarrow E - S_v$ 
21       $V \leftarrow V - \{w \in V | \delta(w) = 0\}$ 
22    fim
23  fim
24  retorna  $(S, G)$ 
25 fim

```

Os conjuntos S_v e S_w correspondem aos conjuntos estrelas de v e w . O conjunto estrela é formado por toda aresta em G incidente em v e w .

No Algoritmo 3.1, da linha 5 até a linha 10 é executada a regra 2 e na linha 9 é realizada a eliminação dos vértices de grau 0, após inserir o vizinho do vértice de grau 1 na cobertura e remover suas arestas incidentes. Este laço é executado enquanto existir vértices de grau 1.

O segundo laço da linha 11 até a linha 23 é responsável em realizar a regra 3 descrita

na Seção 3.1, ou seja, busca-se vértices de grau 2 com vizinhos adjacentes, existindo tal estrutura, são inseridos na cobertura os vizinhos do vértice de grau 2. Após a inserção desses vértices na cobertura, torna-se necessário executar a regra 2 novamente, pois podem surgir novos vértices com grau 0 ou grau 1. Esta rotina é executada enquanto existirem triângulos no grafo.

Com o final da execução do segundo laço, o algoritmo ED-Regras retorna o conjunto S contendo a cobertura encontrada além do grafo reduzido para ser utilizado pelo algoritmo Guloso abaixo.

Algorithm 3.2: ALGORITMO GULOSO [27]

```

1 Entrada:  $G = (V, E)$ , onde  $V$  representa o conjunto de vértices e  $E$  o conjunto de
   arestas
2 Saída: Cobertura por vértices de  $S$ 
3 início
4    $S \leftarrow \emptyset$ 
5   repita
6     Ordene  $V$  por ordem decrescente de grau;
7     para todo  $v \in V$  faça
8       para todo  $\{u, w\} \in E$  faça
9         se  $v = u$  ou  $v = w$  então
10           $S \leftarrow S \cup \{v\}$ 
11           $E \leftarrow E - \{\{u, w\}\}$ 
12        fim
13      fim
14    fim
15  até  $|E| = 0$ ;
16  retorna  $S$ 
17 fim

```

O Algoritmo 3.2 recebe um grafo e devolve o conjunto S contendo a cobertura encontrada no grafo da entrada. Para encontrar tal cobertura, o algoritmo primeiramente ordena os vértices em ordem decrescente e insere no conjunto S os vértices de maior grau, remove as arestas incidentes de V , atualiza a lista de grau e reordena novamente até que não existam mais arestas no grafo.

A nossa heurística utiliza os dois Algoritmos (Algoritmo 3.1 e Algoritmo 3.2) para resolver o problema da cobertura por vértices mínima. Primeiro é executado o Algoritmo 3.1, que retorna os vértices do conjunto solução S e o grafo reduzido da entrada original.

Nisso, o Algoritmo 3.2 recebe o grafo reduzido pela execução do algoritmo ED-Regras e retorna os vértices pertencentes ao conjunto solução. Terminados os dois algoritmos é realizada a união das coberturas encontradas gerando assim a solução do problema.

O Algoritmo 3.3 apresenta este processo descrito.

Algorithm 3.3: ALGORITMO ED

```

1 Entrada:  $G = (V, E)$ , onde  $V$  representa o conjunto de vértices e  $E$  o conjunto de
   arestas
2 Saída: Cobertura por vértices de  $S$ 
3 início
4    $(S', G') \leftarrow \text{ED-Regras}(G)$ 
5    $S'' \leftarrow \text{Guloso}(G')$ 
6   retorna  $S' \cup S''$ 
7 fim

```

3.2.1 Formato das instâncias

O Centro de Matemática Discreta e Ciência da Computação Teórica (*DIMACS*) [28] definiu um formato para grafos não direcionados, esse formato tem sido usado como padrão para problemas em grafos não direcionados.

Os algoritmos utilizados nos experimentos foram implementados na linguagem Java. O formato das instâncias encontram-se no formato *DIMACS* [28] discutidos na subseção 3.2.1 abaixo.

O arquivo no formato *DIMACS* contém todas as informações de um grafo não direcionado, onde os vértices estão numerados de 1 até “ N ”. A instância é composta de alguns caracteres especiais (que são ignorados pelo programa), mas dão informações legíveis sobre o arquivo. Existe uma linha problema por arquivo de entrada. A linha do problema deve aparecer antes de quaisquer outras linhas, descrevendo os vértices ou arestas. A linha do problema tem o seguinte formato. “ p ” “FORMATO” “NODES” “EDGES”.

A letra minúscula “ p ” significa que esta é a linha do problema. O campo FORMATO deve conter a palavra “edge”. O campo NODES contém um valor inteiro especificando “ n ”, o número de vértices no grafo. O campo EDGES contém um valor inteiro especificando “ m ”, o número de arestas no grafo.

Há uma linha de descrição para cada aresta do grafo, a letra minúscula “e” informa no arquivo que esta é uma linha de descrição da aresta. Cada aresta (u, v) aparece apenas uma vez no arquivo de entrada. A Tabela 3.1 apresenta um exemplo do formato de arquivo em *DIMACS*.

Tabela 3.1: Exemplo de arquivo no format DIMACS

p	edge	8	9
e	1	2	
e	1	3	
e	1	4	
e	1	5	
e	1	7	
e	2	5	
e	4	8	
e	5	6	
e	5	7	

A Figura 3.5 representa o grafo da entrada no formata DIMACS da Tabela 3.1.

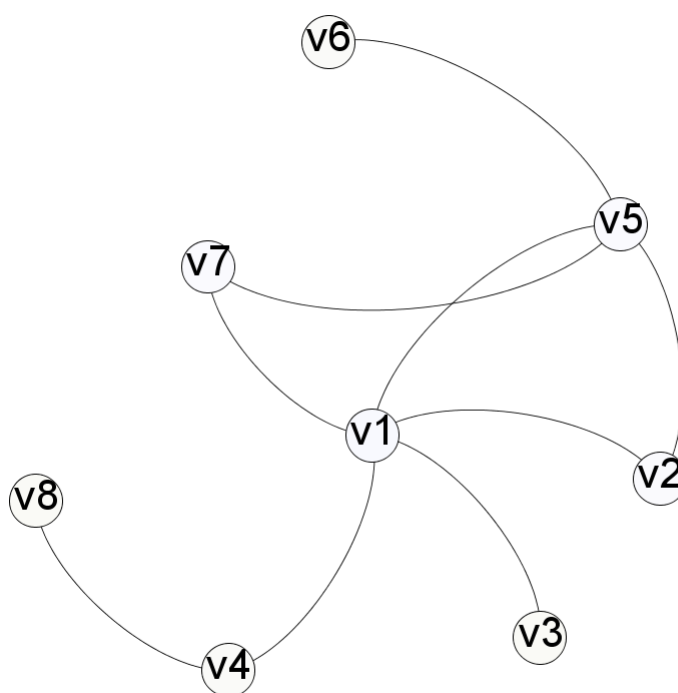


Figura 3.5: Grafo exemplo do arquivo no formato DIMACS 3.1

CAPÍTULO 4

RESULTADOS EXPERIMENTAIS

Neste capítulo são apresentados experimentos realizados em 33 grafos do mundo real, nos quais 18 foram retirados da mesma base de dados apresentados por [27] em seu trabalho.

Esta quantidade de instâncias foi escolhida uma vez que nem todos os grafos utilizados por [27] apresentavam uma distribuição de grau lei de potência. Além disso, como a proposta desta pesquisa é trabalhar com grafos lei de potência houve a necessidade de selecionar mais grafos do mundo real.

Os 18 grafos retirados da mesma base de dados apresentados por [27] são apresentados na Tabela 4.1.

Tabela 4.1: Instâncias utilizadas no trabalho de [27]

Grafo	# Vértices	Tipo do grafo
YeastS	2284	Biológico
CA-GrQc	5241	Citação
p2p-Gnutella08	6301	Informação
p2p-Gnutella09	8114	Informação
p2p-Gnutella06	8717	Informação
p2p-Gnutella05	8846	Informação
CA-HepTh	9875	Citação
p2p-Gnutella04	10876	Informação
CA-AstroPh	18771	Colaboração
p2p-Gnutella25	22687	Informação
CA-CondMat	23133	Colaboração
p2p-Gnutella24	26518	Informação
Cit-HepTh	27769	Citação
p2p-Gnutella30	36682	Informação
Email-Enron	36692	Comunicação
Brightkite-edges	58228	Rede social
p2p-Gnutella31	62586	Informação
soc-Epinions1	75879	Rede social

Os demais grafos foram extraídos da base de dados [25]. Esta é uma base interativa que apresenta importantes informações dos grafos, como o número de vértices, arestas, grau mínimo, grau máximo, número de triangulos, densidade entre outras informações.

A base ainda fornece uma análise em tempo real de alguns grafos temporais e contém a disposição para pesquisas mais de 500 grafos separados em seus tipos, biológico, ciência social, rede social entre outros.

Na seção abaixo temos a descrição de cada grafo utilizado nos experimentos deste trabalho (exceto os grafos listados na Tabela 4.1).

4.1 Cenário e grafos utilizados nos experimentos

São apresentados nesta seção a descrição dos grafos retirados da base de dados [25] e utilizados em nossos experimentos.

4.1.1 ca-citeseer

Rede de colaboração extraída da biblioteca digital *CiteSeer*. Seus 227320 vértices representam publicações, enquanto suas arestas, representam citações entre as publicações.

4.1.2 ca-dblp-2012

Rede de colaboração com 317090 vértices. A DBLP é uma lista abrangente de trabalhos de pesquisa em ciência da computação, no qual os vértices representam autores, enquanto suas arestas representam trabalhos realizados em conjunto.

4.1.3 ca-MathSciNet

Rede de colaboração com 332689 vértices. A *MathSciNet* é uma biblioteca que lista as publicações na área da matemática, os vértices representam autores e suas arestas, trabalhos publicados em conjunto.

4.1.4 ia-wiki-Talk

Rede de iteração do Wikipédia, onde os vértices são representados por usuários cadastrados e suas arestas, por edições de material publicado. Esta rede contém 92117 vértices coletados desde 3 de janeiro de 2008.

4.1.5 sc-nasasrb

Rede de computação científica que contém 54870 vértices. Seu grau mínimo inicia-se de 3, enquanto o grau máximo chega a 275. Não é descrito no banco de dados o significado dos vértices e suas arestas.

4.1.6 soc-brightkite

Esta rede social contém o total de 196591 vértices. É baseada em localização, onde seus usuários compartilham suas localizações. O conjunto de dados contém todas as ligações entre os usuários.

4.1.7 soc-douban

Rede social online (com 154908 vértices), que fornece serviços de revisão ao usuário e recomendação de filmes, livros e música. O conjunto de dados contém todas as ligações entre seus usuários.

4.1.8 soc-gowalla

Rede social baseada em localização, onde seus usuários compartilham suas localizações pelo check-in. O conjunto de dados contém todas as ligações entre os usuários. Esta rede contém o total de 196591 vértices.

4.1.9 socfb-Berkeley13

Rede social do Facebook da universidade de Berkeley, contendo 22900 vértices extraídos em Setembro de 2005. Nesta rede, os vértices são usuários do facebook e as arestas relacionamento de amizade entre os usuários.

4.1.10 socfb-OR

Rede social do Facebook da universidade de Oregon, contendo 63392 vértices extraídos em Setembro de 2005. Nesta rede, os vértices são usuários do facebook e as arestas

relacionamento de amizade entre os usuários.

4.1.11 socfb-Stanford3

Rede social do Facebook da universidade de Stanford, contendo 11586 vértices extraídos em Setembro de 2005. Nesta rede, os vértices são usuários do facebook e as arestas relacionamento de amizade entre os usuários.

4.1.12 socfb-UCLA

Rede social do Facebook da universidade de UCLA, contendo 20453 vértices extraídos em Setembro de 2005. Nesta rede, os vértices são usuários do facebook e as arestas relacionamento de amizade entre os usuários.

4.1.13 socfb-UCSB37

Rede social do Facebook da universidade da Califórnia, contendo 14917 vértices extraídos em Setembro de 2005. Nesta rede, os vértices são usuários do facebook e as arestas relacionamento de amizade entre os usuários.

4.1.14 tech-RL-caida

Rede tecnológica com 190914 vértices. O conjunto de dados é formado através da topologia de rede nível roteador via Rocketfuel.

4.1.15 web-sk-2005

Rede de grafo Web contendo 121422. Este grafo foi obtido a partir de um rastreamento em 2005 do domínio .sk, realizados por *UbiCrawler*, grupo de pesquisadores da Eslováquia.

4.2 Análise e discussão dos resultados

Esta seção apresenta uma discussão a partir dos dados coletados através da aplicação da estratégia gulosa de [27] e das regras de redução ao núcleo discutidas no Capítulo 2 deste trabalho.

Os testes foram realizados em grafos com algumas centenas de vértices até grafos com 332689 vértices, numa máquina com sistema operacional Linux Mint 17 Cinnamon 64-bit, com Kernel do Linux:4.0.1-Kernel+ e processador: AMD FX(tm)-6300 Six-Core Processor x3.

Na Tabela 4.2, apresentamos todos os grafos utilizados, além do seu tipo e classificação referente em ser ou não um grafo lei de potência. Na primeira coluna temos o nome dos grafos utilizados nos experimentos, em seguida, o número de vértices de cada grafo. Na terceira coluna encontram-se o tipo do grafo (Biológico, Colaboração, rede social, entre outros). Na quarta e última coluna temos a classificação dos grafos, se são grafos lei de potência (L.P) ou não (N.L.P).

Conforme a Definição 2 apresentada no Capítulo 2 deste trabalho, ao plotar a distribuição de grau de um grafo na escala log-log, podemos observar uma linha reta no gráfico. Esta linha reta representa que a distribuição de grau do grafo obedece uma lei de potência.

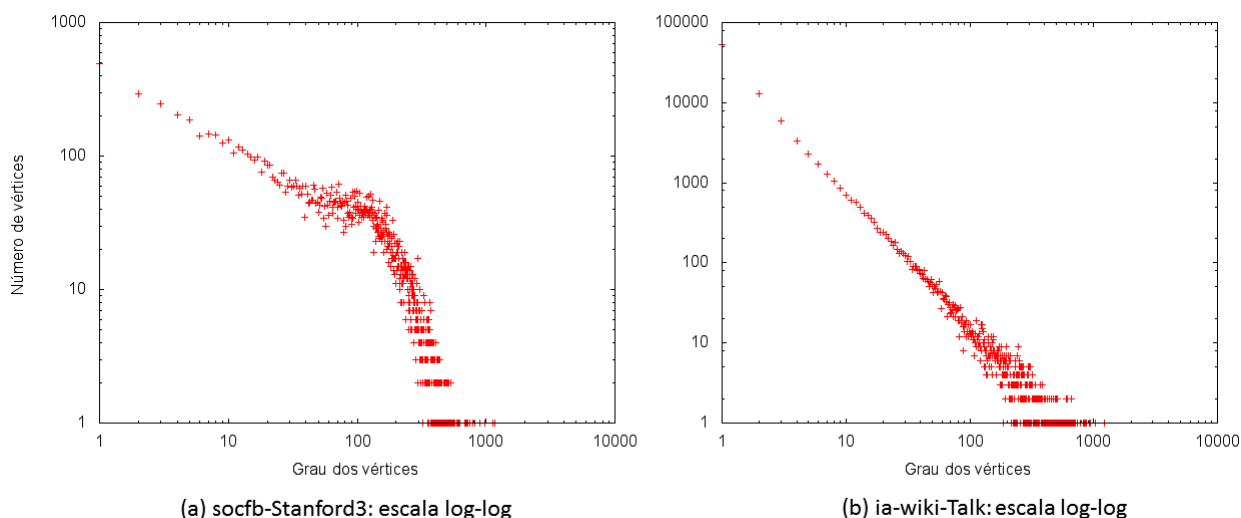


Figura 4.1: Grafo complexo (a) e grafo lei de potência (b).

Tabela 4.2: Classificação dos grafos do mundo real utilizados nos experimentos

Grafo	# Vértices	Tipo do grafo	Classificação
YeastS	2284	Biológico	L.P
CA-GrQc	5241	Citação	L.P
p2p-Gnutella08	6301	Informação	N.L.P
p2p-Gnutella09	8114	Informação	N.L.P
p2p-Gnutella06	8717	Informação	N.L.P
p2p-Gnutella05	8846	Informação	N.L.P
CA-HepTh	9875	Citação	L.P
p2p-Gnutella04	10876	Informação	N.L.P
socfb-Stanford3	11586	Rede social - Facebook	N.L.P
socfb-UCSB37	14917	Rede social - Facebook	N.L.P
CA-AstroPh	18771	Colaboração	L.P
socfb-UCLA	20453	Rede social - Facebook	N,L.P
p2p-Gnutella25	22687	Informação	N.L.P
socfb-Berkeley13	22900	Rede social - Facebook	N.L.P
CA-CondMat	23133	Colaboração	L.P
p2p-Gnutella24	26518	Informação	N.L.P
Cit-HepTh	27769	Citação	N.L.P
p2p-Gnutella30	36682	Informação	N.L.P
Email-Enron	36692	Comunicação	L.P
sc-nasasrb	54870	Computação Científica	L.P
soc-brightkite	56739	Rede social	L.P
Brightkite-edges	58228	Rede social	L.P
p2p-Gnutella31	62586	Informação	N.L.P
socfb-OR	63392	Rede social - Facebook	L.P
soc-Epinions1	75879	Rede social	L.P
ia-wiki-Talk	92117	Rede de iteração	L.P
web-sk-2005	121422	Grafo Web	N.L.P
soc-douban	154908	Rede social	N. L.P
tech-RL-caida	190914	Rede Tecnológica	L.P
soc-gowalla	190914	Rede Social	L.P
ca-citeseer	227320	Colaboração	L.P
ca-dblp-2012	317090	Colaboração	L.P
ca-MathSciNet	332689	Colaboração	L.P

Conforme a Figura 4.1, os grafos que delineiam uma reta, como o da Figura 4.1b foram classificadas como L.P (lei de potência), os demais foram classificados como N.L.P. (como, por exemplo, o da Figura 4.1a)

Nas seções seguintes, são apresentados os resultados da porcentagem de redução dos grafos ao núcleo do problema, tamanho da cobertura por vértices encontrada e o tempo de execução gasto em cada experimento. Todos estes dados foram obtidos através da

utilização do Algoritmo 3.3 apresentado na Seção 3.2.

4.3 Redução do tamanho do grafo

Nesta seção é apresentada a taxa de diminuição de cada um dos 33 grafos utilizados no experimentos, após a execução **somente** das regras de redução ao núcleo, mencionadas no Capítulo 3.

Os dados para a sua apresentação foram separados em duas tabelas, Tabela 4.3 para os grafos que não são lei de potência e Tabela 4.4 para os grafos lei de potência.

Na primeira coluna, temos os nomes dos grafos, seguido (segunda coluna), do seu número de vértices originais. A terceira coluna apresenta a quantidade de vértices restantes após a aplicação das três regras de redução ao núcleo. Por fim, na quarta coluna, temos a porcentagem de diminuição do grafo original.

A aplicação das regras de redução ao núcleo garantem uma diminuição significativa quando temos grafos lei de potência, uma vez que em sua distribuição de grau existem muitos vértices de grau baixo e poucos vértices com grau elevando. No entanto, estas regras também podem, por vezes, reduzir significativamente grafos que não sejam lei de potência.

Na Tabela 4.3, vários grafos tiveram uma redução de 100% do seu tamanho original. Outros grafos, por exemplo *socfb-Stanford3*, *socfb-UCSB37*, *socfb-UCLA* e *socfb-Berkeley13* tiveram uma redução abaixo de 20%, em média tiveram 72,82% de redução.

É perfeitamente possível ocorrer reduções de 100%, principalmente em grafos esparsos. Este foi um caso particular nos experimentos realizados e ocorreram em grafos do tipo informação. Sabemos que os grafos *p2p-Gnutella* não são árvores e nem contém uma distribuição de grau lei de potência, mas não foram investigados as características topológicas do tamanho desses grafos que ocasionaram tal redução, uma vez que o objetivo é saber a taxa de diminuição pela aplicação das regras de redução em grafos lei de potência.

Como era de se esperar, em grafos lei de potência, obtivemos reduções significativas, conforme apresenta a Tabela 4.4. Descartando o grafo *sc-nasasrb* a menor redução encontrada foi de 60,9%, do grafo *CA-AstroPh*. A instância que não obteve redução deve-se

Tabela 4.3: Redução do tamanho dos grafos após aplicação das regras de redução - N.L.P

Grafo	# Vértices	# Vértices Restantes	Diminuição (%)
p2p-Gnutella08	6301	0	100%
p2p-Gnutella09	8114	0	100%
p2p-Gnutella06	8717	0	100%
p2p-Gnutella05	8846	0	100%
p2p-Gnutella04	10876	7	99,9%
p2p-Gnutella25	22687	0	100%
p2p-Gnutella24	26518	0	100%
Cit-HepTh	27769	17632	36,5%
p2p-Gnutella30	36682	0	100%
p2p-Gnutella31	62586	0	100%
socfb-Stanford3	11586	9314	19,6%
socfb-UCSB37	14917	13225	11,3%
socfb-UCLA	20453	17543	14,2%
socfb-Berkeley13	22900	19937	12,94
web-sk-2005	121422	35775	70,54
soc-douban	154908	0	100%

ao grafo em questão ter como grau mínimo 3 e como as regras de redução apresentadas nesta pesquisa processam vértices de grau zero, um e dois, o grafo *sc-nasasrb* não sofre nenhuma alteração em sua estrutura. A Figura 4.2 apresenta o grafo *sc-nasasrb*.

Em média, obteve-se 87,93% de redução nos grafos lei de potência (excluindo o grafo *sc-nasasrb*). Isto demonstra que a utilização das regras de redução ao núcleo do problema são efetivas em grafos lei de potência.

Podemos dizer que as regras de redução ao núcleo são mais eficientes quando tratamos de grafos lei de potência. Nos experimentos, realizados neste trabalho, obtivemos uma diferença de 15% a mais de redução, nos grafos L.P. em relação aos grafos N.L.P. A figura 4.3 um gráfico que apresenta a diminuição nos grafos lei de potência da Tabela 4.4.

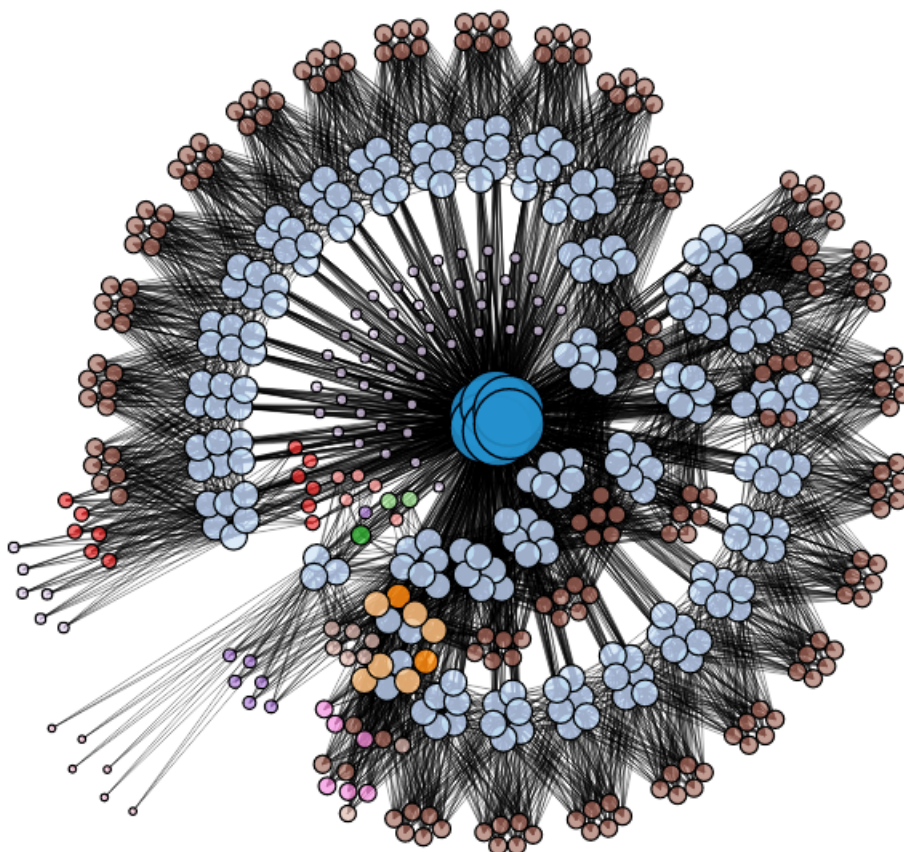


Figura 4.2: Grafo *sc-nasasrb*. Fonte: [24]

Tabela 4.4: Redução do tamanho dos grafos após aplicação das regras de redução - L.P

Grafo	# Vértices	# Vértices Restantes	Diminuição (%)
YeastS	2284	0	100%
CA-GrQc	5241	631	88%
CA-HepTh	9875	570	94,2%
CA-AstroPh	17903	7341	60,9%
CA-CondMat	23133	6233	73,1%
Email-Enron	36692	4185	88,6%
sc-nasasrb	54870	54870	0%
soc-brightkite	56739	882	98,4%
Brightkite-edges	58228	409	99,3%
socfb-OR	63392	10850	82,9%
soc-Epinions1	75879	246	99,7%
ia-wiki-Talk	92117	0	100%
tech-RL-caida	190914	35775	81,3%
soc-gowalla	196591	2568	98,7%
ca-citeseer	227320	73235	67,8%
ca-dblp-2012	317090	66054	79,2%
ca-MathSciNet	332689	16600	95%

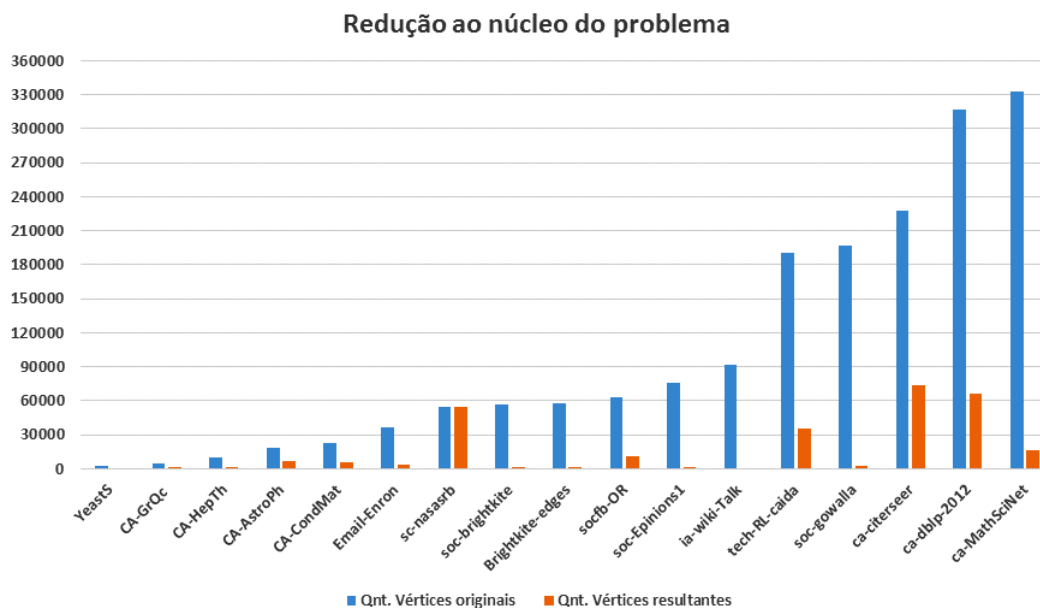


Figura 4.3: Comparação entre as duas estratégias aplicadas nos grafos lei de potência - Redução de vértices

4.4 Tamanho da cobertura por vértices encontrada

Nesta seção discute-se sobre o tamanho da solução encontrada pelo Algoritmo 3.2 apreciado no trabalho de [27], em comparação com o Algoritmo 3.1.

Para uma melhor análise dos dados obtidos, os grafos foram separados pelo seu tipo, ou seja, separados em conjuntos (grafos com e sem distribuição de grau que segue uma lei de potência).

As Tabelas 4.5 e 4.6 apresentam 4 colunas. A primeira coluna relaciona o nome do grafo utilizado no experimento. Na segunda coluna, tem-se a cobertura por vértices mínima obtida utilizando o mesmo algoritmo guloso apresentado em [27] e descrito no capítulo 3. Os dados apresentados na terceira coluna referem-se aos resultados obtidos pela execução do Algoritmo 3.3. Na última coluna tem-se a relação da diminuição do tamanho das coberturas encontradas (Algoritmo 3.2 em relação ao Algoritmo 3.1).

Pode-se observar que independente de ser ou não grafos lei de potência, a utilização das regras de redução antes da execução do algoritmo guloso diminui o tamanho da solução.

Entre essas diminuições, a maior foi de 2,28%, do grafo *p2p-Gnutella06*, uma diminuição de quase 80 vértices. Já a menor diminuição da cobertura por vértices encontrada

Tabela 4.5: Grafos do mundo real que não apresentam uma distribuição de grau lei de potência

Grafo	# Vértices	Guloso	Regras de redução	Diminuição da Cobertura (%)
p2p-Gnutella08	6301	2070	2054	0,77
p2p-Gnutella09	8114	2589	2577	0,46
p2p-Gnutella06	8717	3484	3405	2,28
p2p-Gnutella05	8846	3475	3428	1,35
p2p-Gnutella04	10876	4428	4349	1,78
p2p-Gnutella25	22687	6055	6017	0,63
p2p-Gnutella24	26518	7250	7206	0,61
Cit-HepTh	27769	18225	17913	1,71
p2p-Gnutella30	36682	9321	9272	0,53
p2p-Gnutella31	62586	15864	15696	1,06
socfb-Stanford3	11586	8623	8603	0,27
socfb-UCSB37	14917	11480	11466	0,12
socfb-UCLA	20453	15497	15465	0,21
socfb-Berkeley13	22900	17513	17474	0,22
web-sk-2005	121422	58490	57995	0,85
soc-douban	154908	8719	8685	0,39

neste conjunto de grafos listados na Tabela 4.5 foi de 0,12%. Em média obteve-se uma diminuição no tamanho da cobertura por vértices de 0,83%.

Na Tabela 4.6 (com lei de potência), pode-se observar uma diminuição média ainda maior do tamanho das coberturas na Tabela 4.5.

Na Tabela 4.6, a diminuição no tamanho da cobertura por vértices encontrada pelo algoritmo guloso em relação as regras de redução teve em média uma redução de 4,16% relativo ao tamanho da solução.

A maior diminuição da cobertura por vértices utilizando as regras de redução ocorreram no grafo *tech-RL-caida*, 23,44% menor em relação a solução encontrada pela estratégia gulosa. Como o grafo *sc-nasasrb* contém vértices de grau 3 para cima, ele não apresenta uma diferença na solução aplicando a estratégia das regras de redução antes do algoritmo guloso.

A Figura 4.4, mostra que a solução obtida ao utilizar o Algoritmo 3.3, apesar de serem melhores, não são tão diferentes da solução encontrada pelo Algoritmo 3.2.

Tabela 4.6: Grafos do mundo real com distribuição de grau lei de potência

Grafo	# Vértices	Guloso	Regras de redução	Diminuição da Cobertura (%)
YeastS	2284	773	763	1,29
CA-GrQc	5241	2795	2783	0,43
CA-HepTh	9875	5003	4713	5,80
CA-AstroPh	18771	12044	11725	2,65
CA-CondMat	23133	13561	12920	4,73
Email-Enron	36692	14477	13501	6,74
sc-nasasrb	54870	51597	51597	0,00
soc-brightkite	56739	21494	20669	3,84
Brightkite-edges	58228	22177	21564	2,76
socfb-OR	63392	37251	36586	1,79
soc-Epinions1	75879	22418	22071	1,55
ia-wiki-Talk	92117	17427	17268	0,91
tech-RL-caida	190914	75755	57995	23,44
soc-gowalla	196591	85456	82515	3,44
ca-citeseer	227320	129350	125104	3,28
ca-dblp-2012	317090	165249	158471	4,10
ca-MathSciNet	332689	140674	135103	3,96

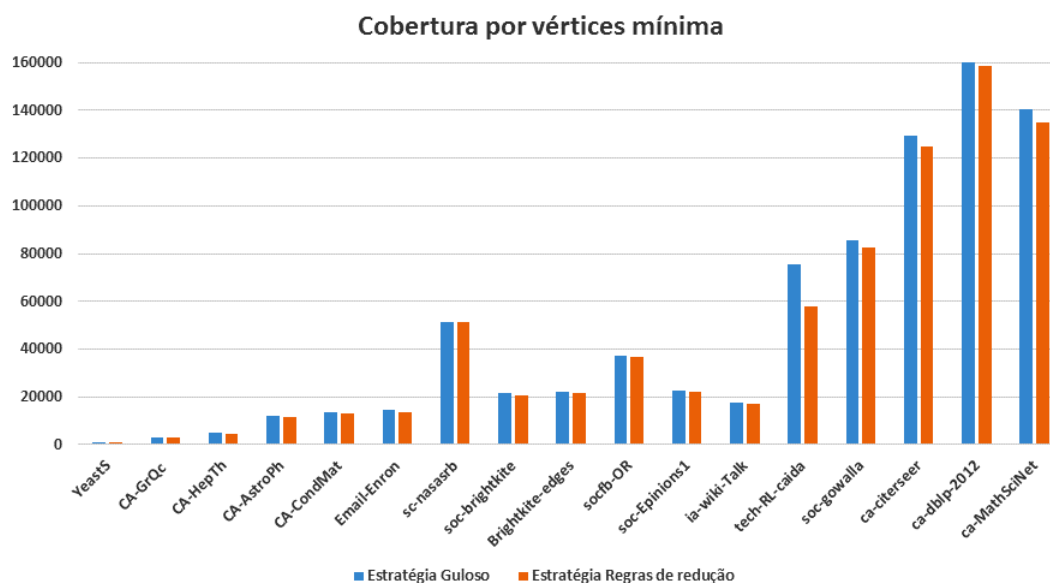


Figura 4.4: Comparação entre as duas estratégias aplicadas nos grafos lei de potência - Cobertura por vértices mínima

4.5 Tempo computacional

Nesta seção é apresentado os resultados obtidos sobre o tempo de execução gasto para encontrar a solução do problema da cobertura por vértices mínima nas 33 instâncias utilizadas neste trabalho.

A apresentação dos dados encontra-se separados em duas tabelas, na Tabela 4.7, encontram-se o tempo de execução dos grafos que não são lei de potência. Já na Tabela 4.8 temos o tempo gasto dos grafos lei de potência.

Cada tabela contém 4 colunas: a primeira coluna é responsável em identificar a instância utilizada; a segunda coluna apresenta o tempo de execução em segundos gastos pela estratégia gulosa; o tempo de execução da estratégia que utiliza as regras de redução encontram-se na terceira coluna; a quarta coluna apresenta a relação (em porcentagem) da diminuição do tempo de execução entre as duas estratégias ditas acima (estratégia gulosa pelas regras de redução).

Conforme apresentado no Algoritmo 3.1, o custo computacional das regras de redução é $O(n^2)$. Sendo assim, o uso das regras além de reduzir o tamanho do grafo e aproximar a solução “ótima”, também diminui o tempo de execução na resolução do problema.

Nos grafos que não são lei de potência, a utilização das regras de redução levou, em alguns casos, um tempo de execução maior ao ser aplicadas, como foi o caso do grafo *socfb-Berkeley13*. Acredita-se que este fato esteja interligado com o quanto o grafo pode ser reduzido ao núcleo como também ao quanto ele obteve de diminuição da cobertura entre as estratégias (gulosa e regras de redução), apresentada na seção 4.3.

A média da diminuição do tempo entre as estratégias aplicadas foi de 51,2%, nos grafos N.L.P., excluindo o tempo gasto pela instância *socfb-Berkeley13*.

Os grafos lei de potência (Tabela 4.8), apresentam uma diminuição do tempo de execução em média 49% mais rápido, entre a estratégia gulosa e as regras de redução. Mesmo algumas instâncias como *CA-GrQc*, *CA-CondMat* e *socfb-OR* em que o tempo aumentou, este aumento não foi maior do que 20%.

Acredita-se que esta melhora no tempo de execução acontece quando a distribuição do grafo é a mais fiel possível de uma distribuição lei de potência. Ou seja, quando mais fácil exergar uma reta na distribuição de graus de um grafo na escala logarítmica, maior será a diminuição da instância, menor será o tempo de execução e menor será a cobertura por vértices mínima encontrada (isto aplicando as regras de redução de baixo grau).

Comparando as médias do tempo gasto nos grafos N.L.P. e L.P obtivemos aproxima-

Tabela 4.7: Tempo de execução dos grafos sem distribuição de lei de potência

Grafo	Tempo Guloso (s)	Tempo Regras (s)	Diminuição do tempo em relação às regras (%)
p2p-Gnutella08	0,547	0,162	70,38
p2p-Gnutella09	1,024	0,268	73,83
p2p-Gnutella06	1,238	0,386	68,82
p2p-Gnutella05	1,187	0,378	68,16
p2p-Gnutella04	1,757	0,623	64,54
p2p-Gnutella25	4,569	1,335	70,78
p2p-Gnutella24	7,273	2,361	67,54
Cit-HepTh	43,917	50,026	-13,91
p2p-Gnutella30	14,826	2,876	80,60
p2p-Gnutella31	49,081	8,772	82,13
socfb-Stanford3	49,05	52,4	-6,83
socfb-UCSB37	47,51	48,575	-2,24
socfb-UCLA	116,8	118,593	-1,54
socfb-Berkeley13	17,51	154,301	-781,22
web-sk-2005	343,97	110,224	67,96
soc-douban	94,62	21,037	77,77

Tabela 4.8: Tempo de execução dos grafos sem distribuição de lei de potência

Grafo	Tempo Guloso (s)	Tempo Regras (s)	Diminuição do tempo em relação às regras (%)
YeastS	0,175	0,05	71,43
CA-GrQc	0,443	0,462	-4,29
CA-HepTh	1,484	0,513	65,43
CA-AstroPh	14,536	14,377	1,09
CA-CondMat	10,19	11,887	-16,65
Email-Enron	18,007	10,883	39,56
sc-nasasrb	638,67	659,942	-3,33
soc-brightkite	49,2	12,187	75,23
Brightkite-edges	52,598	12,862	75,55
socfb-OR	301,255	333,396	-10,67
soc-Epinions1	84,734	28,345	66,55
ia-wiki-Talk	73,385	20,789	71,67
tech-RL-caida	1178,89	109,477	90,71
soc-gowalla	1511,08	374,187	75,24
ca-citeseer	2556,06	708,687	72,27
ca-dblp-2012	4822,66	1088,048	77,44
ca-MathSciNet	4093,38	574,235	85,97

damente 50% de melhoria utilizando o Algoritmo 3.3. Ou seja, a utilização das regras de redução contribuem no desempenho do algoritmo.

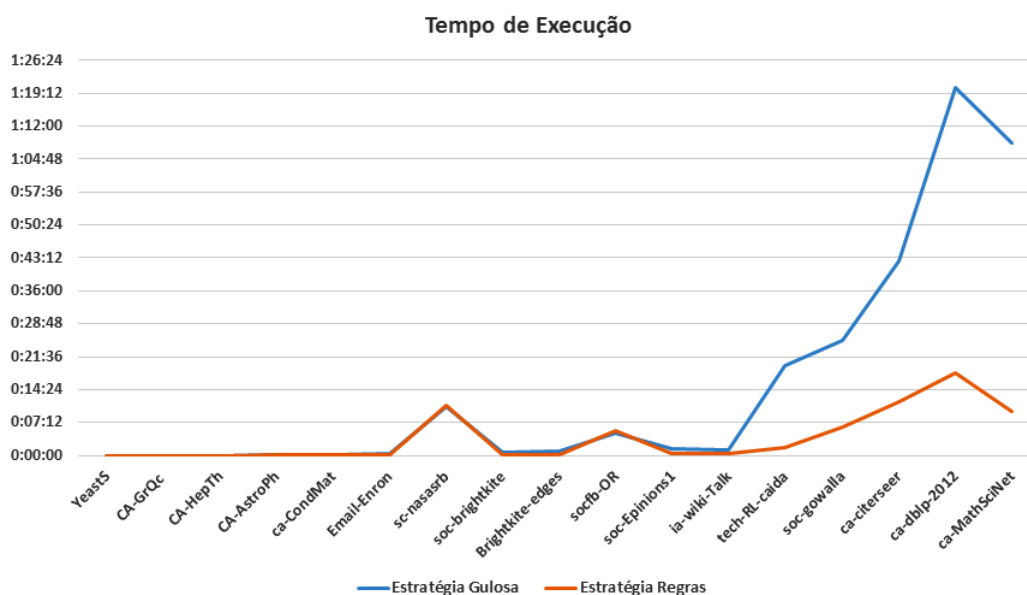


Figura 4.5: Tempo de execução, em grafos lei de potência, das estratégias discutidas neste trabalho.

A Figura 4.5 apresenta a escala de tempo entre as duas estratégias utilizadas na resolução do problema da cobertura por vértices mínima, dos grafos lei de potência. Nota-se que a estratégia Gulosa obteve um gasto de tempo cada vez maior ao executar instâncias grandes, chegando a ultrapassar 1 hora de execução em instâncias com mais de 300 mil vértices. Já com o uso das regras de redução, o tempo gasto nestas instâncias maiores, não ultrapassou 20 minutos.

4.6 Conclusão da análise dos experimentos

A utilização das regras de redução apresentadas neste trabalho, antes da execução do algoritmo guloso discutido por [27], em grafos lei de potência reduzem significamente o tamanho da instância original, (72, 82% N.L.P. e 87, 93% L.P.) podendo em alguns casos reduzir toda a instância.

Esta redução pode não melhorar muito o tamanho da solução encontrada pela estratégia gulosa (redução de 0, 83% N.L.P. e 4, 16% L.P.), porém proporciona uma economia de tempo computacional relevante (diminuição de 51, 2% N.L.P. e 49, 01% L.P.). Logo, a utilização destas regras de redução apresentadas neste trabalho melhoram a solução do

problema da cobertura por vértices mínima e principalmente contribui para a diminuição do tempo computacional para resolver tal problema. Além disso, em todos os casos, aplicar as regras de redução fornecem mais vantagens em grafos L.P do que em grafos N.L.P.

CAPÍTULO 5

CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Foi apresentada neste trabalho uma pesquisa experimental, observando as melhorias na utilização de mecanismos de pré-processamento na resolução do problema da cobertura por vértices mínima em grafos lei de potência.

Nos testes realizados observou-se que estratégia gulosa apresentada por da Silva et. al. [27] ainda pode ser melhorada ao trabalhamos com grafos lei de potência, obtendo uma melhor aproximação da cobertura ótima, dentro de um tempo computacional mais curto.

Um grafo lei de potência é uma rede complexa cujo grau de distribuição segue uma power law (lei de potência). Estas redes complexas apresentam alguns poucos hubs, que são vértices de grau elevados, em sua estrutura. O algoritmo guloso proposto por [27] explora esta propriedade, ou seja, aproveita a existência dos hubs e os adicionam numa cobertura. Esta abordagem gulosa apresenta até então uma boa performance, mas ao considerar o uso das regras de redução ao núcleo do problema, a abordagem gulosa melhora ainda mais sua performance, principalmente no tempo de execução para encontrar uma possível solução.

Para a obtenção dos dados, foram realizados testes com 33 grafos reais de diferentes domínios. Destas redes, 17 apresentaram uma distribuição de grau lei de potência. A quantidade de vértices varia entre milhares até centenas de milhares e conforme seu tamanho, quanto maior o grafo lei de potência, mais a quantidade de vértices de grau baixo.

Os resultados confirmaram a efetividade das regras de redução ao núcleo do problema nos grafos lei de potência, com redução de 88% em média. Algumas redes tiveram reduções de 100% o que significam que encontraram uma solução ótima para o problema da cobertura por vértices mínima.

Já na Tabela 4.6, as regras de redução aplicadas antes da estratégia gulosa, conseguiu obter 23% de diminuição da cobertura por vértices mínima em relação ao uso somente da estratégia gulosa. Porém, a melhoria não foi tão significativa em relação aos demais grafos, obtendo uma média de 0,83%.

Em relação ao tempo gasto na execução dos experimentos, as regras de redução, ao tratarem de grafos lei de potência, aplicadas antes do algoritmo guloso de [27], obteve um custo computacional significativamente melhor em relação ao uso somente do algoritmo guloso.

Resumidamente, o uso das regras de redução ao núcleo do problema em grafos lei de potência, para o problema da cobertura por vértices mínima é uma abordagem mais interessante do que somente utilizar uma estratégia gulosa. Principalmente ao trabalhar com problemas práticos que necessitam de rapidez e otimização dos resultados.

Por fim, algumas questões que ficaram em aberto neste trabalho merecem um estudo mais aprofundado, tais como:

- A implementação da quarta regra de redução apresentada em [1] consegue melhorar ainda mais os dados apresentados neste trabalho?
- O que leva algumas redes complexas terem 100% de redução mesmo não se tratando especificamente de grafos lei de potência?
- Ao se trabalhar com outras classes de grafos complexos (como por exemplo grafos de mundo pequeno) estas regras de redução, funcionariam bem?

ANEXO 1

Distribuição de grau dos grafos lei de potência, na escala log-log.

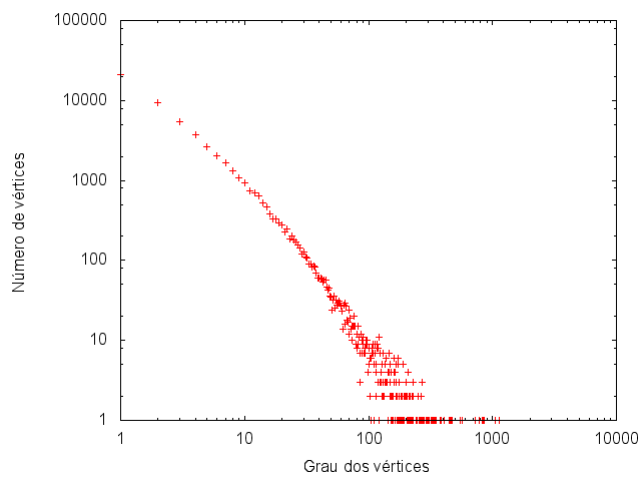


Figura 5.1: Distribuição de grau do grafo *Brightkite_{edges}*

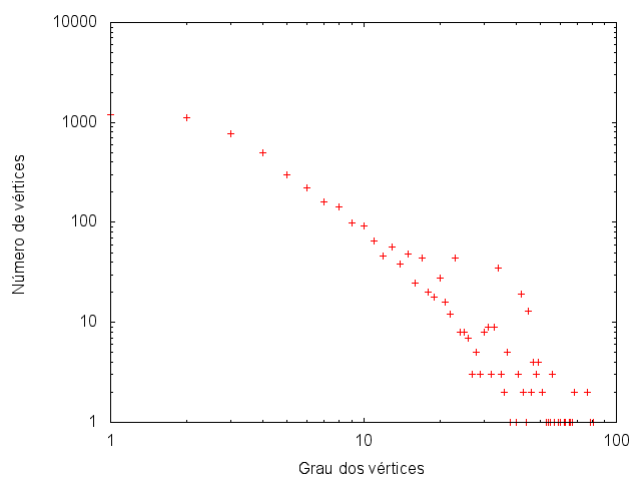


Figura 5.2: Distribuição de grau do grafo *CA-GrQc*

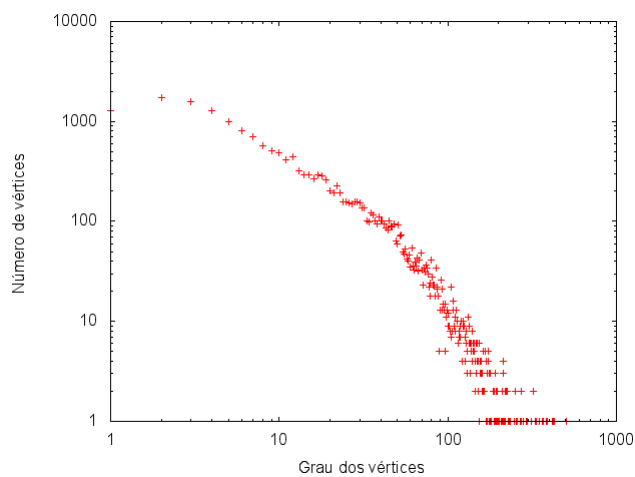


Figura 5.3: Distribuição de grau do grafo *CA-AstroPh*

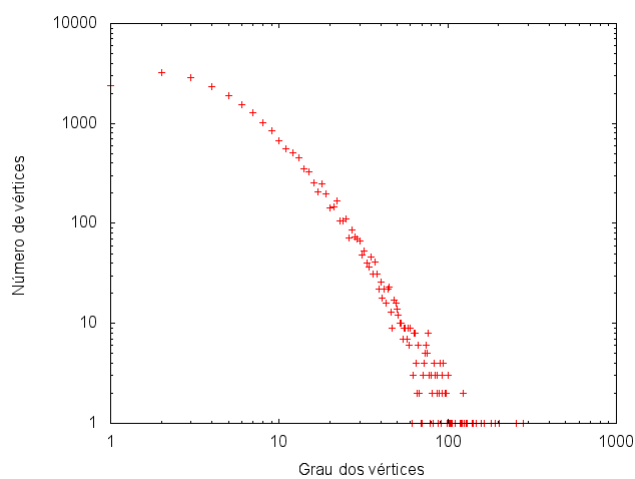


Figura 5.4: Distribuição de grau do grafo *CA-CondMat*

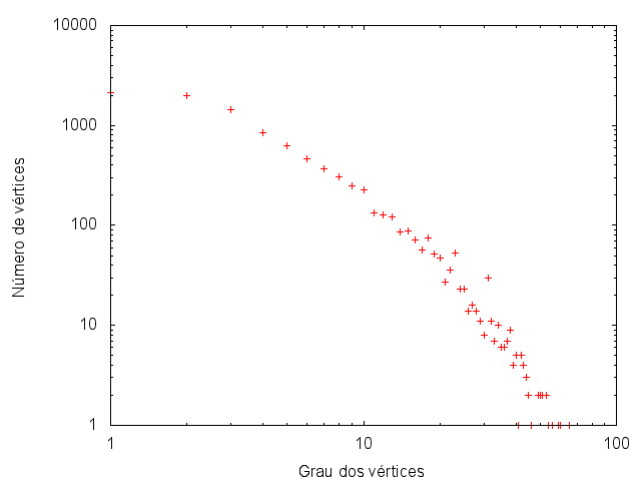


Figura 5.5: Distribuição de grau do grafo *CA-HepTh*

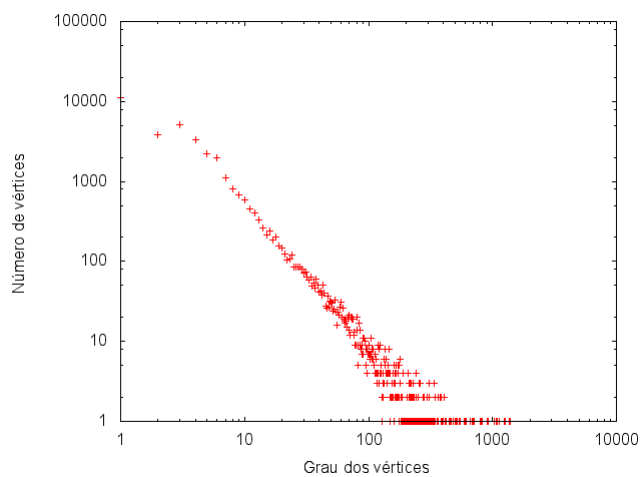


Figura 5.6: Distribuição de grau do grafo *Email-Enron*

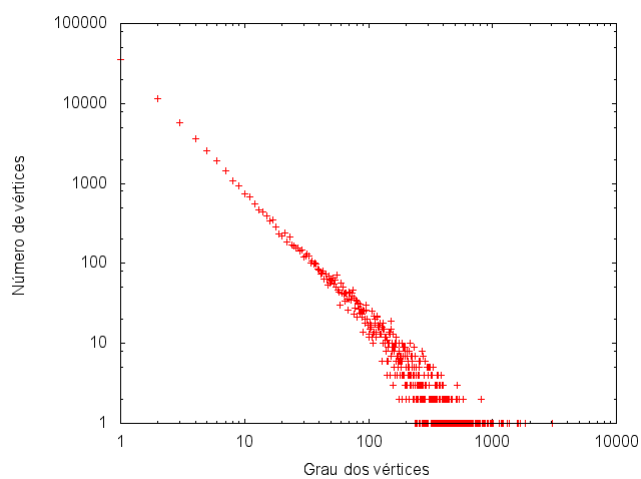


Figura 5.7: Distribuição de grau do grafo *soc-Epinions1*

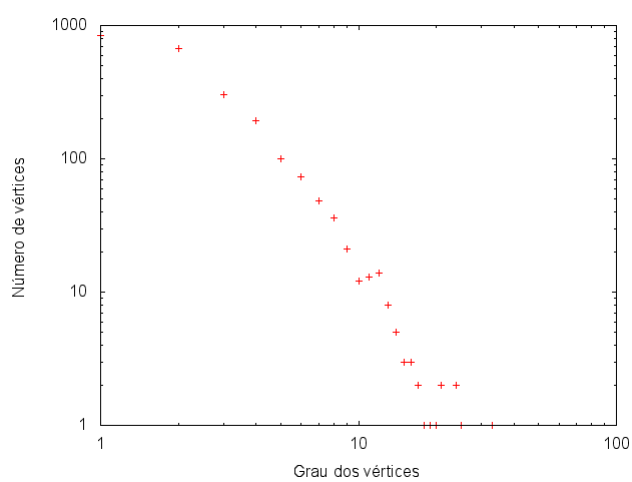


Figura 5.8: Distribuição de grau do grafo *YeastS*

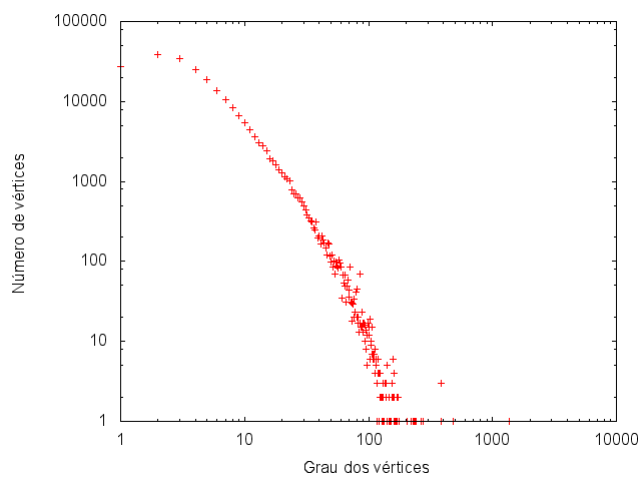


Figura 5.9: Distribuição de grau do grafo *sc-nasasrb*

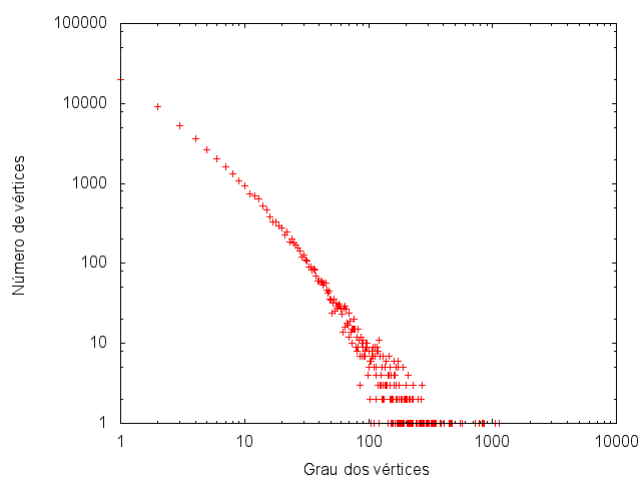


Figura 5.10: Distribuição de grau do grafo *soc-brightkite*

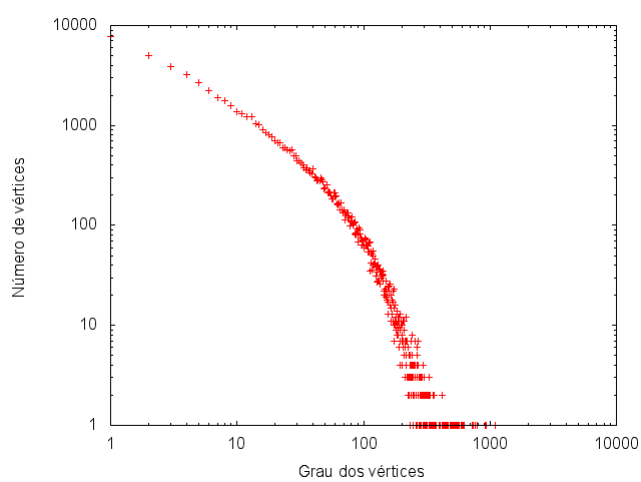


Figura 5.11: Distribuição de grau do grafo *socfb-OR*

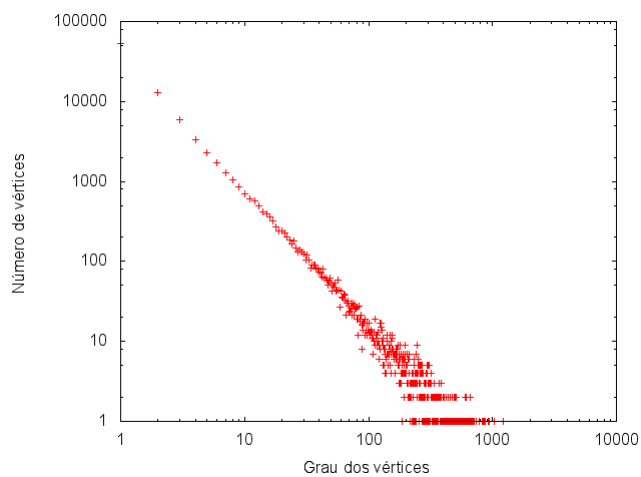


Figura 5.12: Distribuição de grau do grafo *ia-wiki-Talk*

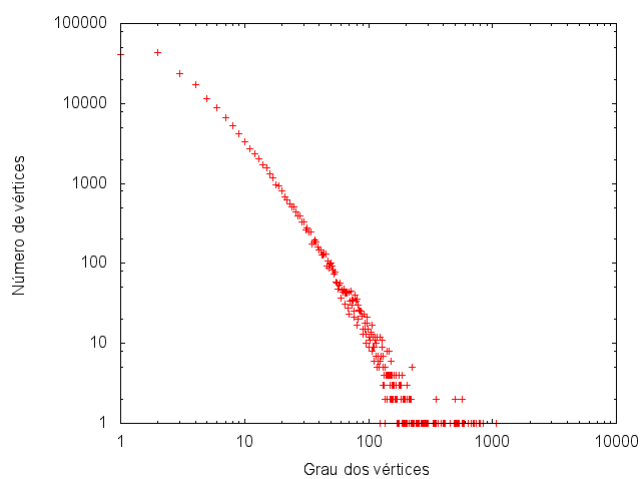


Figura 5.13: Distribuição de grau do grafo *tech-RL-caida*

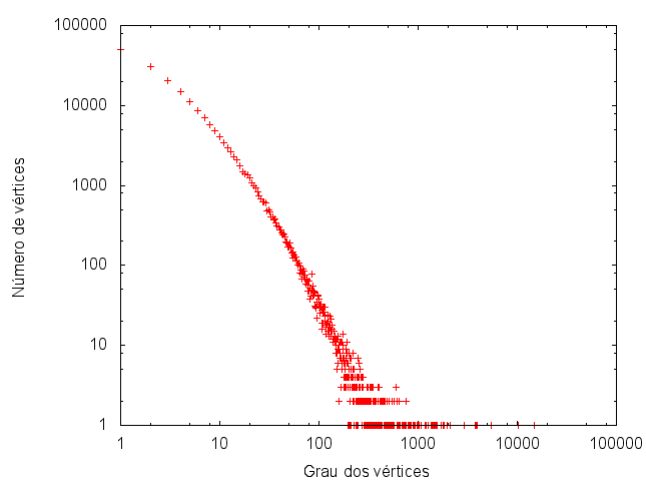


Figura 5.14: Distribuição de grau do grafo *soc-gowalla*

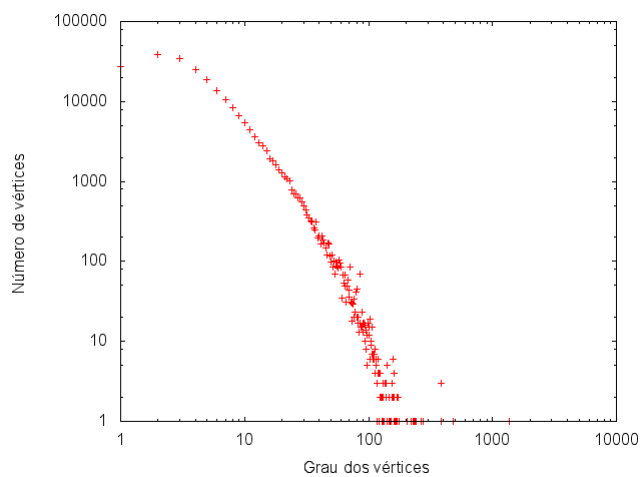


Figura 5.15: Distribuição de grau do grafo *ca-citepeer*

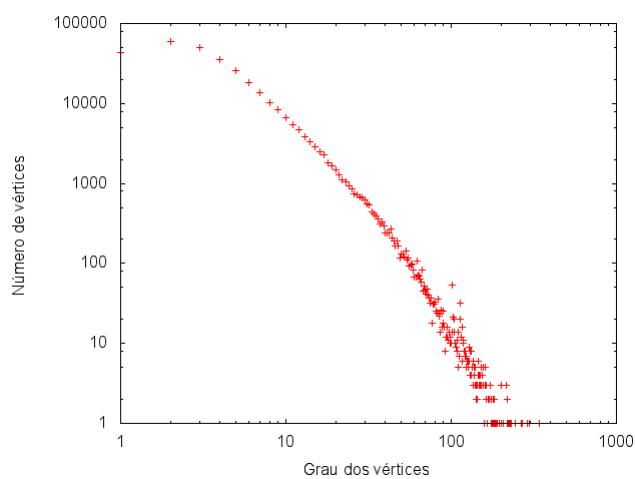


Figura 5.16: Distribuição de grau do grafo *ca-dblp-2012*

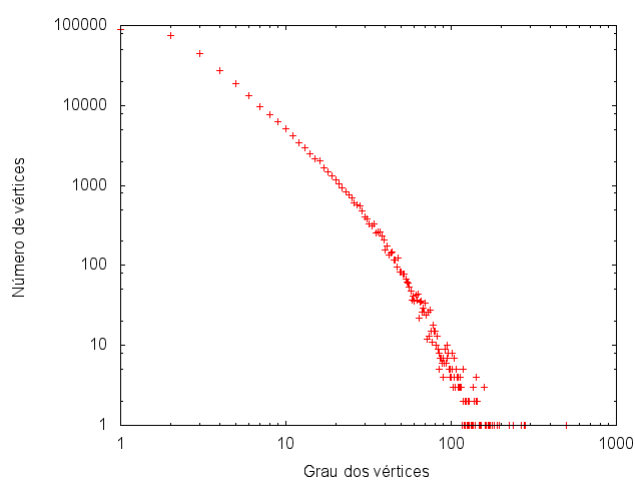


Figura 5.17: Distribuição de grau do grafo *ca-MathSciNet*

ANEXO 2

Distribuição de grau dos grafos não lei de potência, na escala log-log.

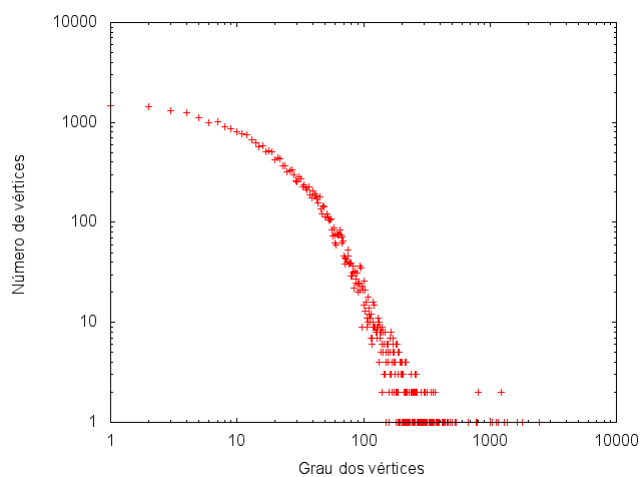


Figura 5.18: Distribuição de grau do grafo *Cit-HepTh*

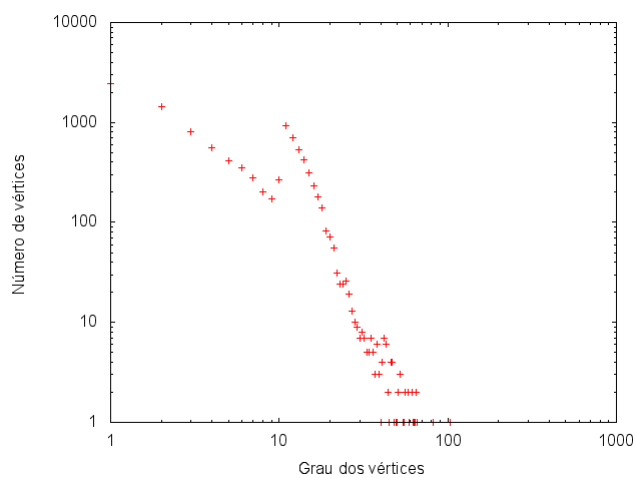


Figura 5.19: Distribuição de grau do grafo *p2p-Gnutella04*

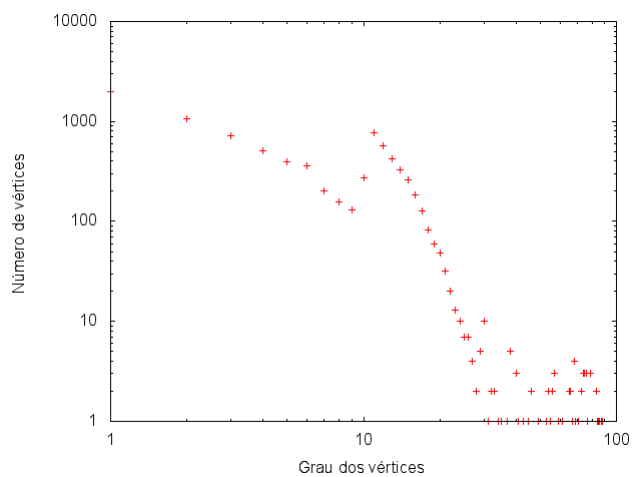


Figura 5.20: Distribuição de grau do grafo $p2p$ -Gnutella05

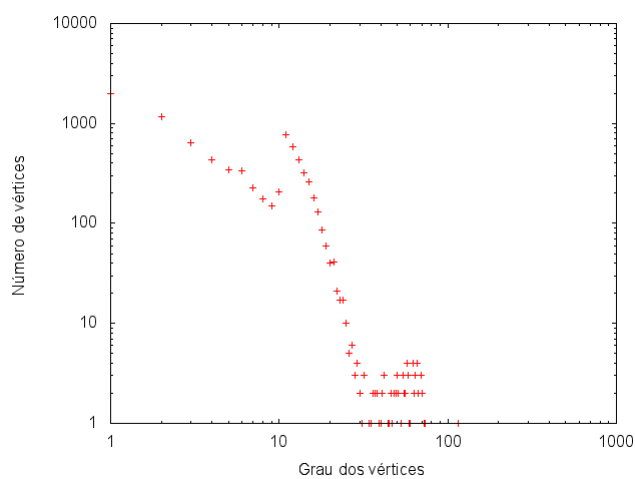


Figura 5.21: Distribuição de grau do grafo $p2p$ -Gnutella06

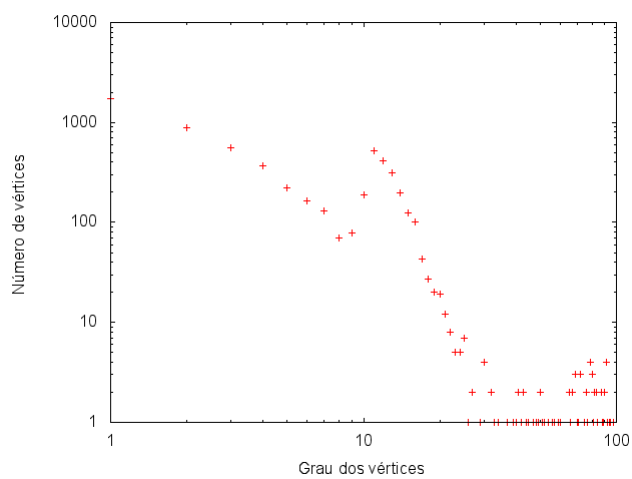


Figura 5.22: Distribuição de grau do grafo $p2p$ -Gnutella08

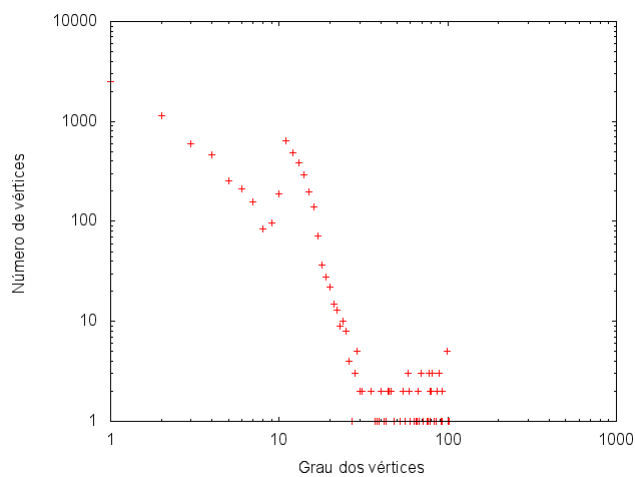


Figura 5.23: Distribuição de grau do grafo $p2p$ -Gnutella09

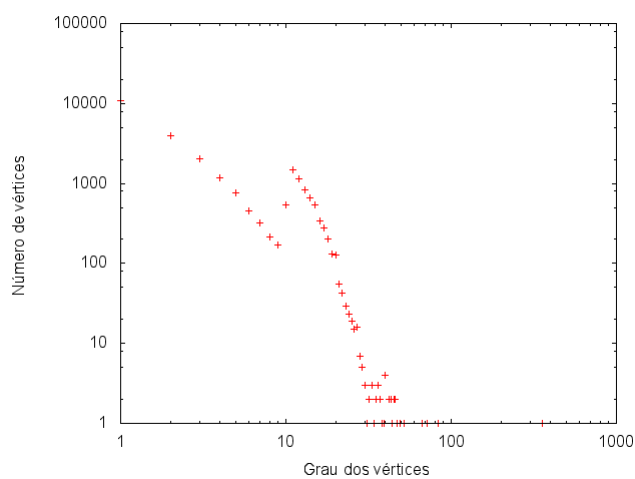


Figura 5.24: Distribuição de grau do grafo $p2p$ -Gnutella24

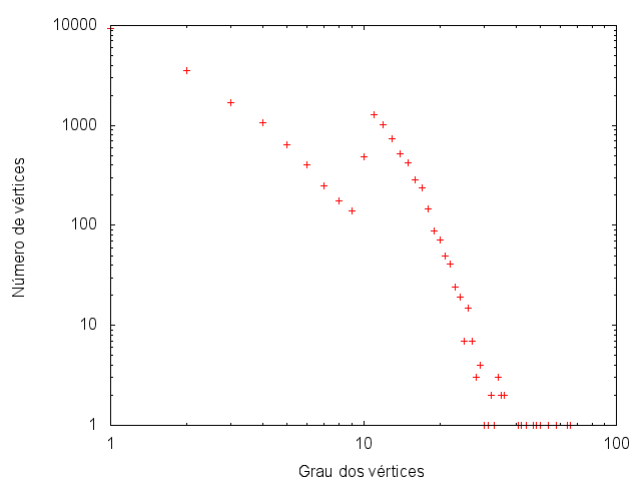


Figura 5.25: Distribuição de grau do grafo $p2p$ -Gnutella25

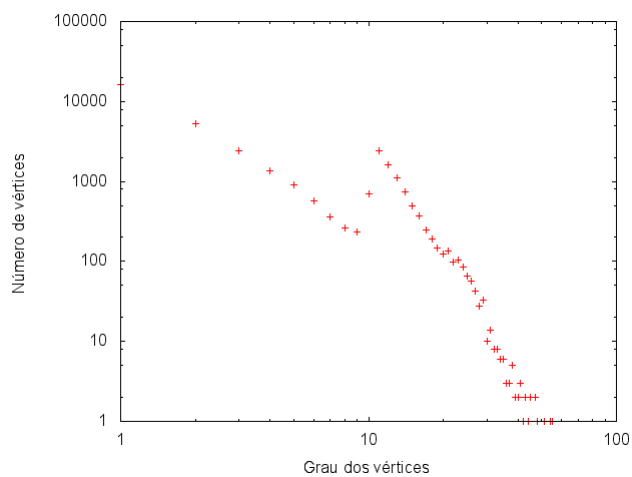


Figura 5.26: Distribuição de grau do grafo $p2p$ -Gnutella30

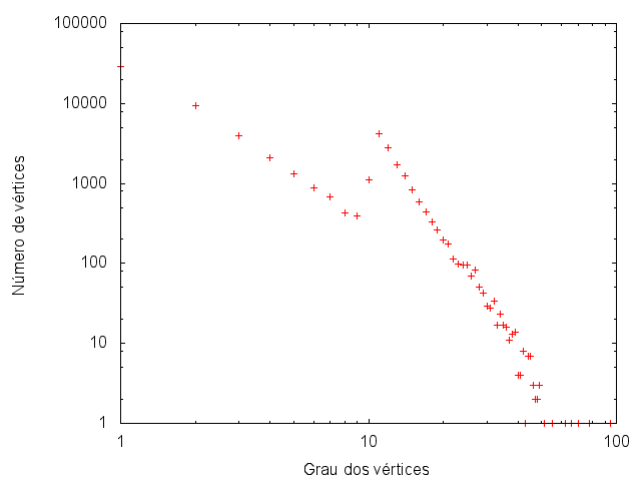


Figura 5.27: Distribuição de grau do grafo $p2p$ -Gnutella31

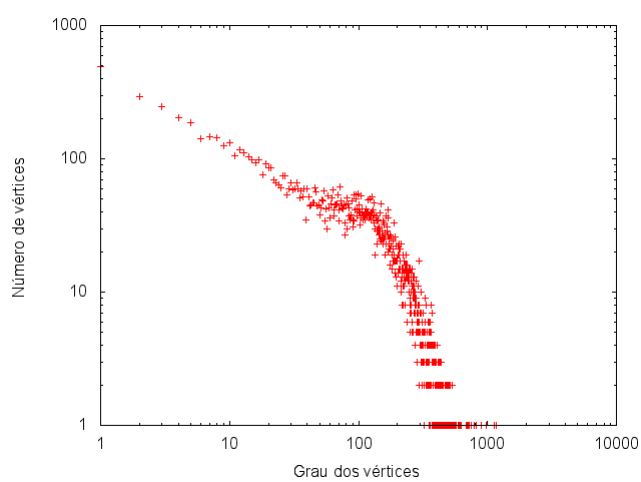


Figura 5.28: Distribuição de grau do grafo $socfb$ -Stanford3

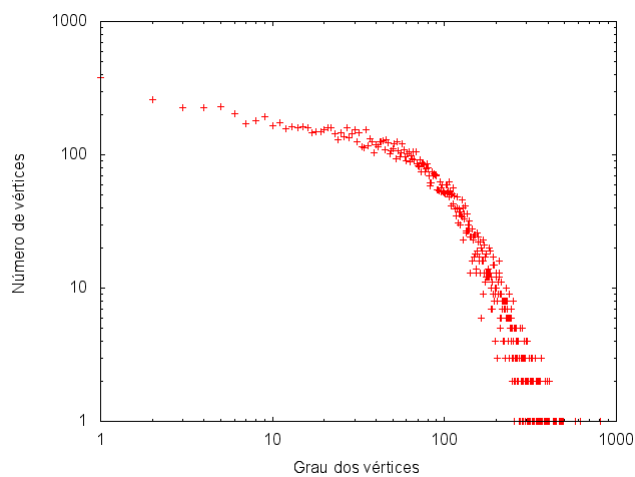


Figura 5.29: Distribuição de grau do grafo *socfb-UCSB37*

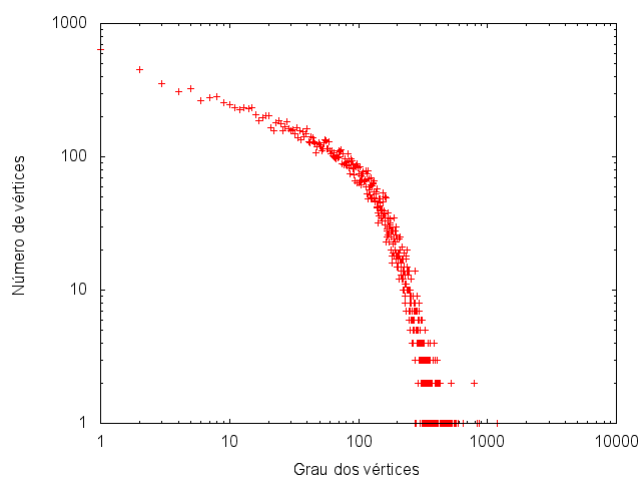


Figura 5.30: Distribuição de grau do grafo *socfb-UCLA*

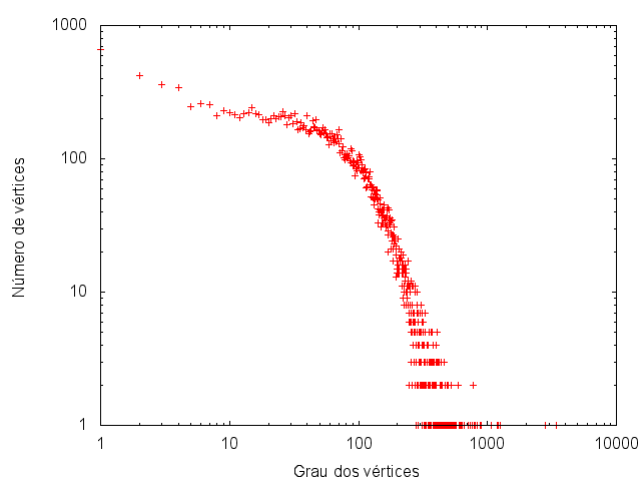


Figura 5.31: Distribuição de grau do grafo *socfb-Berkeley13*

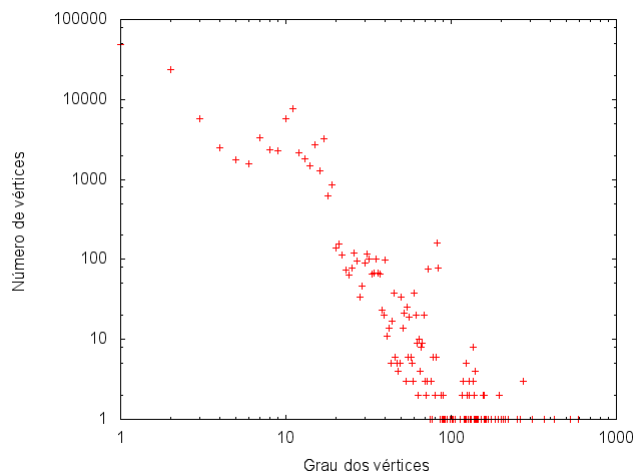


Figura 5.32: Distribuição de grau do grafo *web-sk-2005*

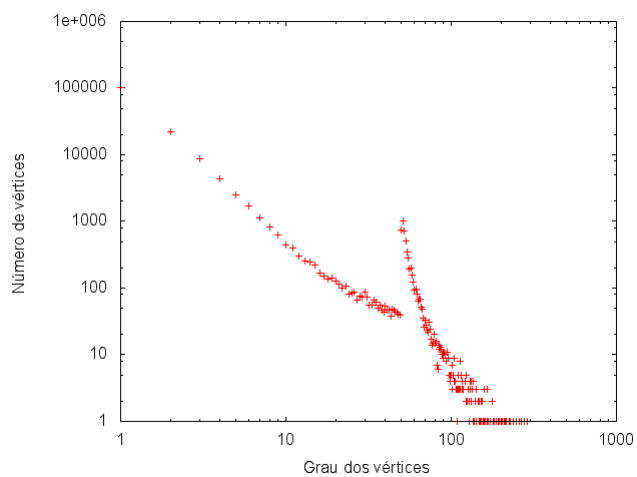


Figura 5.33: Distribuição de grau do grafo *soc-douban*

BIBLIOGRAFIA

- [1] Faisal N Abu-Khzam, Rebecca L Collins, Michael R Fellows, Michael A Langston, W Henry Suters, e Christopher T Symons. Kernelization algorithms for the vertex cover problem: Theory and experiments. *ALLENEX/ANALC*, 69, 2004.
- [2] William Aiello, Fan Chung, e Linyuan Lu. A random graph model for power law graphs. *Experimental Mathematics*, 10(1):53–66, 2001.
- [3] Sanjeev Arora e Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- [4] Matti Åstrand, Patrik Floréen, Valentin Polishchuk, Joel Rybicki, Jukka Suomela, e Jara Uitto. A local 2-approximation algorithm for the vertex cover problem. *Distributed Computing*, páginas 191–205. Springer, 2009.
- [5] Albert-László Barabási, Réka Albert, e Hawoong Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1):69–77, 2000.
- [6] Albert-László Barabási et al. Scale-free networks: a decade and beyond. *science*, 325(5939):412, 2009.
- [7] Paolo Boldi, Bruno Codenotti, Massimo Santini, e Sebastiano Vigna. UbiCrawler: A scalable fully distributed web crawler. *Software: Practice & Experience*, 34(8):711–726, 2004.
- [8] Jonathan F. Buss e Judy Goldsmith. Nondeterminism within p. *SIAM J. Comput.*, 22(3):560–572, junho de 1993.
- [9] Shaowei Cai, Kaile Su, Chuan Luo, e Abdul Sattar. Numvc: An efficient local search algorithm for minimum vertex cover. *Journal of Artificial Intelligence Research*, páginas 687–716, 2013.

- [10] James Cheetham, Frank Dehne, Andrew Rau-Chaplin, Ulrike Stege, e Peter J Taillon. Solving large fpt problems on coarse-grained parallel machines. *Journal of Computer and System Sciences*, 67(4):691–706, 2003.
- [11] Aaron Clauset, Cosma Rohilla Shalizi, e Mark EJ Newman. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703, 2009.
- [12] Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.
- [13] Mariana O Da Silva, Gustavo A Gimenez-Lugo, e Murilo VG Da Silva. Vertex cover in complex networks. *International Journal of Modern Physics C*, 24(11), 2013.
- [14] Rodney G Downey e Michael R Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013.
- [15] Rodney G Downey e Michael Ralph Fellows. *Parameterized complexity*. Springer Science & Business Media, 2012.
- [16] Michael R Fellows, Danny Hermelin, e Frances Rosamond. On the fixed-parameter intractability and tractability of multiple-interval graph problems. *Unpublished Result*, 2007.
- [17] Carlos Eduardo Ferreira. *Uma introdução sucinta a algoritmos de aproximação*. IMPA, 2001.
- [18] Mikael Gast e Mathias Hauptmann. Approximability of the vertex cover problem in power-law graphs. *Theoretical Computer Science*, 516:60–70, 2014.
- [19] Mikael Gast, Mathias Hauptmann, e Marek Karpinski. Improved approximation lower bounds for vertex cover on power law graphs and some generalizations. *arXiv preprint arXiv:1210.2698*, 2012.
- [20] Erik Joey Hanashiro. O problema da k-cobertura por vértices: uma implementação fpt no modelo bsp/cgm. 2004.

- [21] George Karakostas. A better approximation ratio for the vertex cover problem. *ACM Transactions on Algorithms (TALG)*, 5(4):41, 2009.
- [22] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.
- [23] Jon M Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, e Andrew S Tomkins. The web as a graph: measurements, models, and methods. *Computing and combinatorics*, páginas 1–17. Springer, 1999.
- [24] Ryan A. Rossi e Nesreen K. Ahmed. sc-nasasrb - scientific computing, 2013.
- [25] Ryan A. Rossi e Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [26] Yilin Shen, Dung T Nguyen, Ying Xuan, e My T Thai. New techniques for approximating optimal substructure problems in power-law graphs. *Theoretical Computer Science*, 447:107–119, 2012.
- [27] Mariana Oliveira da Silva. Problema de cobertura por vértices em redes complexas. 2014.
- [28] Rutgers University. Center for discrete mathematics and theoretical computer science, junho de 2015.
- [29] Yogesh Virkar, Aaron Clauset, et al. Power-law distributions in binned empirical data. *The Annals of Applied Statistics*, 8(1):89–119, 2014.

EDGAR DE OLIVEIRA CABRAL FILHO

**COBERTURA POR VÉRTICES MÍNIMA EM GRAFOS LEI
DE POTÊNCIA**

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Renato José da Silva Carmo

Co-Orientador: Prof. Dr. André Luís Vignatti

CURITIBA

2016