SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura:

Métodos para o pré-processamento e mineração de grandes volumes de dados multidimensionais e redes complexas

Ana Paula Appel

Orientador: ***Prof. Dr. Caetano Traina Junior***

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências - Ciências de Computação e Matemática Computacional.

USP – São Carlos
Abril de 2010

Aos meus pais Anna e José.

Ao meu amor Adriano.

Em memória do meu amigo e Prof. Mauro Biajiz.

Life is not tried, it is merely survived
If you're standing outside the fire
Garth Brooks

Agradecimentos

Em primeiro agradeço à Deus por estar sempre presente na minha vida, me dar forças na hora que fraquejei e tornar possível a realização de mais este trabalho. Aos meus pais, Anna e José, por todo amor, carinho e apoio incondicional, vocês são os melhores pais do mundo! Amo muito vocês! Ao meu noivo Adriano pelo apoio e força durante todo esse tempo. Como não poderia deixar de agradecer aos meus orientadores. Ao Prof. Dr. Caetano, obrigada pela orientação, confiança, oportunidade, broncas, paciência, ajuda nas horas de dúvidas e principalmente pela amizade. Ao Prof. Dr. Christos Faloutsos pela grande orientação e oportunidade de aprendizado. Também gostaria de agradecer à minha família: Ana Beatris, Antônio José, Solimar, Pedro, Matheus e Bruna por todo carinho e amor. Especialmente a Bruna por aguentar muitas vezes o meu stress. À minha família americana, Nancy e Miriam, pelos almoços de domingo, amizade, carinho e amor. Como não poderia esquecer aos meus amigos e colegas: sejam eles de longa data Regiane Laveli, Dany Laveli, Ariane do Amaral, Humberto Razente, Maria Camila Barioni, Enzo Seraphim, Thatiana Seraphim, Pedro Bugatti e Renato Bueno; ou recentes Estevam Hruschka, Simone Hruschka, Eduardo Hruschka, Daniel Kaster, Marcelo Ponciano, Willian Dener, Freddy, Robson Cordeiro, Eloise, Carolina, Sérgio, Mônica, Letricia, André Balan, Cristina Ciferri, Ricardo Ciferri, Felipe, David, Jenna, Polo, Leman, Mary, HangHang, Charlotte, U Kang e Kensuke. Aos meus futuros sogros Zélia e Marcos e aos meus cunhados(a) Matheus, William e Marcela. Aos meus tios postiços Ângela e Jeremias que torceram por mim. Aos professores e amigos da UFSCar, pelo incentivo e apoio, vocês também são responsáveis por isso. Especialmente à Profa. Dra. Rosângela Penteado que torceu muito por mim e ajudou nas orações. Aos professores e funcionários do ICMC pela atenção e apoio. Especialmente à Profa. Dra. Agma Traina pelos incentivos. À Angela no centro cultural pela ajuda com o inglês. Às instituições de fomento a pesquisa CNPq, Capes e FAPESP o meu agradecimento pelo apoio financeiro.

Resumo

A mineração de dados é um processo computacionalmente caro, que se apoia no pré-processamento dos dados para aumentar a sua eficiência. As técnicas de redução de elementos do conjunto de dados, principalmente a amostragem de dados se destacam no pré-processamento. Os dados reais são caracterizados pela não uniformidade da distribuição, grande quantidade de atributos e presença de elementos considerados ruídos. Para esse tipo de dado, a amostragem uniforme, na qual cada elemento tem a mesma probabilidade de ser escolhido, é ineficiente. Os dados nos últimos anos, vem passando por transformações. Assim, não só o seu volume tem aumentado significantemente, mas também a maneira de como eles são representados. Os dados usualmente são divididos apenas em dados tradicionais (número e pequenas cadeias de caracteres) e dados complexos (imagens, cadeias de DNA, vídeos, etc). Entretanto, uma representação mais rica, na qual não só os elementos do conjunto são representados mas também a suas ligações, vem sendo amplamente utilizada. Esse novo tipo de dado, chamado rede complexa, fez surgir uma nova área de pesquisa chamada mineração de redes complexas ou de grafos, já que estes são utilizados na representação das redes complexas. Para esta nova área é necessário o desenvolvimento de técnicas que permitam a mineração de grandes redes complexas, isto é, redes com centenas de milhares de elementos(nós) e ligações(arestas). Esta tese teve como objetivo explorar a redução de elementos em conjuntos de dados chamados desbalanceados, isto é, que possuem agrupamentos ou classes de tamanhos bastantes distintos, e que também possuam alta quantidade de atributos e presença de ruídos. Além disso, esta tese também explora a mineração de redes complexas com a extração de padrões e propriedades e o desenvolvimento de algoritmos eficientes para a classificação das redes em reais e sintéticas. Também é proposto a mineração de redes complexas utilizando gerenciadores de base de dados para a mineração de cliques de tamanho 4 e 5 e a apresentação da extensão do coeficiente de clusterização.

APPEL, A. P. *Métodos para o pré-processamento e mineração de grande volume de dados espaciais e redes complexas*. São Carlos, 2010. 153 p. Tese de Doutorado – Instituto de Ciências Matemáticas e de Computação – ICMC/USP.

Abstract

Data mining is an expensive computational process speeded up by data preprocessing. Data reduction techniques, as data sampling are useful during the data preprocessing. Real data are known for presenting non-uniform data distribution, a large amount of attributes and noise. For this type of data, uniform sampling, which selects elements with the same probability, is inefficient.

Over the past years, the data available to mining have been changed. Not only have their volume increased but also data format. Data are usually divided into traditional (number and small chains of character) and complex (images, DNA, videos, etc). However, a rich representation, in which not only elements but also the connections among the elements have been used, is necessary. This new data type, which is called complex network and is usually modeled as a graph, has created a new research area, called graph mining or complex network mining, which requires the development of new mining techniques to allow mining large networks, that is, networks with hundreds of thousands of nodes and edges.

The present thesis aims to explore the data reduction in unbalanced data, that is, data that have clusters with very different sizes, a large amount of attributes and noise. It also explores complex network mining with two basic findings: useful new patterns, which allow distinguishing real from synthetic networks and mining cliques of sizes 4 and 5 using database systems, discovering interesting power laws and presenting a new cluster coefficient formula.

Sumário

Lista de Algoritmos **xiii**

Símbolos **xv**

Siglas **xvii**

1 Introdução **1**
1.1 Motivação . . . 3
1.2 Objetivos . . . 5
1.3 Resultados Obtidos . . . 5
1.4 Organização do Trabalho . . . 6

I Conceitos, definições e trabalhos correlatos **9**

2 Descoberta de Conhecimento em Base de Dados **11**
2.1 Considerações iniciais . . . 11
2.2 O processo de descoberta de conhecimento em base de dados . . . 12
2.3 O pré-processamento de dados . . . 14
2.3.1 A redução de dados . . . 15
2.3.2 Amostragem de dados . . . 18
2.4 Mineração de dados . . . 25
2.4.1 Classificação . . . 26
2.4.2 Regras de Associação . . . 26
2.4.3 Detecção de agrupamentos . . . 27
2.5 Considerações finais . . . 32

3 Redes Complexas: Conceitos, história e aplicações **35**
3.1 Considerações Iniciais . . . 35
3.2 Conceitos Básicos e Definições . . . 38
3.2.1 Teoria dos Grafos: conceitos . . . 38
3.2.2 Leis de Potência . . . 41
3.3 Descoberta de Conhecimento em Grafos . . . 42

3.3.1 O pré-processamento de grafos . . . 43
3.3.2 A mineração de grafos . . . 45
3.3.3 Propriedades estatísticas das redes complexas . . . 46
3.3.4 Detecção de Comunidades . . . 52
3.3.5 Resistência a ataques . . . 54
3.3.6 Predição de Ligações . . . 57
3.4 Modelos de grafos . . . 58
3.5 Evolução das Redes Complexas . . . 63
3.6 Considerações finais . . . 64

II Trabalhos Desenvolvidos 66

4 Biased Box Sampling - BBS 68
4.1 Considerações iniciais . . . 68
4.2 Biased Box Sampling . . . 70
4.3 Resultados Obtidos . . . 75
4.3.1 Conjunto Uniforme . . . 77
4.3.2 Conjunto Pendigits . . . 78
4.3.3 Conjunto OneBig . . . 79
4.3.4 Escalabilidade . . . 81
4.4 Considerações Finais . . . 82

5 Mineração de Redes Complexa: algoritmos, leis e padrões 84
5.1 Considerações Iniciais . . . 84
5.2 As redes complexas utilizadas . . . 85
5.3 ShatterPlots . . . 88
5.3.1 Descrição do algoritmo `ShatterPlots` . . . 92
5.3.2 Experimentos . . . 94
5.3.3 Escalabilidade . . . 103
5.4 A técnica `FCR - Fast Clique Retrieval` . . . 105
5.4.1 Coeficiente de Clusterização Genérico . . . 107
5.4.2 Leis de potência dos Cliques . . . 108
5.4.3 Experimentos . . . 110
5.4.4 Método `FCR` . . . 119
5.4.5 Escalabilidade . . . 123
5.5 Considerações Finais . . . 126

III Conclusões 128

6 Conclusões 130
6.1 Considerações Iniciais . . . 130
6.2 Principais Contribuições . . . 131
6.3 Publicações . . . 132
6.4 Trabalhos Futuros . . . 133
6.4.1 Algoritmo *Biased Box Sampling - BBS* . . . 133
6.4.2 Algoritmo `ShatterPlots` . . . 134
6.4.3 Método `FCR - Fast Clique Retrieval` . . . 134

Lista de Figuras

1.1 Consulta por similaridade representada por um grafo feita na ferramenta Image swirl da Google. 2

2.1 Passos do processo de KDD segundo (Fayyad et al., 1996b,c) 13
2.2 Exemplo de um histograma representando a quantidade de pessoas em um faixa etária - *buckets* de tamanho 10 18
2.3 Exemplificação do algoritmo GBS (Palmer e Faloutsos, 2000) 23
2.4 Um exemplo de um conjunto de dados agrupado em três grupos. 27
2.5 Agrupamentos detectados pelos algoritmos CLARANS e DBSCan (Ester et al., 1996) 31
2.6 Exemplificação do processo realizado pelo algoritmo DBSCan 32

3.1 (a) Mapa de Königsberg (b) Grafo representando o problema das pontes de Königsberg (Newman et al., 2006). 36
3.2 Rede Biológica representada graficamente. 37
3.3 Gráfico Hop-Plot da rede Epinions. A linha tracejada em 5 representa o Diâmetro efetivo da rede. 47
3.4 Exemplo de um grafo com 3 triângulos (contados uma vez só) centrados no nó A: $P_1(A, A) = \{A, B, D, A\}$, $P_2(A, A) = \{A, D, C, A\}$, $P_3(A, A) = \{A, B, C, A\}$ 50
3.5 O gráfico apresentado em (a) representa a média dos triângulos vs. o grau dos nós e o gráfico (b) apresenta a distribuição do grau dos nós. Note que o expoente da lei de potência seguida por ambos são opostos e muito similares (Tsourakakis, 2008). 50
3.6 Grafo apresentando 3 comunidades do modo tradicional (Leskovec et al., 2008). 54
3.7 Grafo apresentando a topologia "Centro-Periferia"(Leskovec et al., 2008). . 55
3.8 Exemplo de uma rede complexa que tem 20% dos seus nós removidos aleatóriamente: (a) é a rede original e os nós marcados em vermelhos são os nós que serão removidos e (b) é a rede (a) já com os nós removidos. 55
3.9 Comparação do comportamento do tamanho da rede quanto a remoção de nós aleatória e nós de alto grau. (Albert et al., 2000) 60

3.10 Evolução da rede complexa Patentes U.S. segundo as leis de evolução. Em (a) o número de nós e arestas crescem proporcionalmente, (b) o diâmetro diminui com a evolução da rede, (c) o seu primeiro autovalor cresce com expoente menor que 0.5 e em (d) o número de nós da GCC cresce até todos os nós da rede formarem uma única componente conexa 65

4.1 Conjunto de dados 20-dimensional (projeção 2D) contendo um agrupamento com 50 mil elementos, oito com mil elementos e 15% de ruído. . . . 69
4.2 (a) Hiper-reticulado de 2 dimensões e 4 níveis (b) MG-Tree (*Multi Grid Tree* - Árvore Multi Resolução) 71
4.3 (a) Hiper-reticulado de 2 dimensões e 4 níveis concatenado (b) MGc-tree - *Multi Grid Compact Tree - Árvore Multi Resolução Compacta* 72
4.4 (a) Escolha dos elementos no Hiper-reticulado (b) Escolha dos elementos na MGc-Tree 74
4.5 Metodologia utilizada para avaliação do algoritmo BBS 75
4.6 Visualização do conjunto "UniformClusters". (a)Conjunto original; e amostras com 0.5% de taxa de amostragem obtidas dos algoritmos: (b) BBS, (c) DBS, (d) GBS; (e) US. 77
4.7 Quantidade de ruído para o conjunto "OneBig" com amostragem de 2%, 30% de ruído e variação da dimensionalidade. 80
4.8 Escalabilidade dos algoritmos de amostragem quanto à dimensionalidade (escala logarítmica) 81

5.1 Exemplificação gráfica do algoritmo `ShatterPlots`. As arestas em vermelho representam aquelas a serem removidas. 88
5.2 Medidas estruturais da rede feitas durante a remoção aleatória de arestas. As redes apresentam `ShatterPoint` em todas as medidas mas somente o diâmetro apresenta um pico. 90
5.3 Propriedades estruturais no (`ShatterPoint`). As redes sintéticas são representadas em triângulos; *Erdös-Rényi* em triângulos pretos. (a)número de nós não isolados ($\mathcal{V}_{sp}$) versus número atual de arestas $\mathcal{E}_{sp}$ no `ShatterPoint` (padrão *30-per-cent*); (b) número de arestas atuais $\mathcal{E}_{sp}$ sobre o número de arestas total $\mathcal{E}$ versus um sobre λ_1, primeiro autovalor da rede original (padrão *Eigenvalue* pattern); Amazon esta desviando da linha bem como as redes sintéticas BR, PA e 2D-grid. (c) número de nós sobreviventes, isto é, não isolados $\mathcal{V}_{sp}$ versus número total de nós da rede original $\mathcal{V}$ (padrão *NodeShatteringRatio*) 2D-grid estão a cima da linha do *Erdös-Rényi*; SW estão juntos com o *Erdös-Rényi*, BR e PA estão acima da linha "c" e abaixo da linha "b" 95
5.4 (d)a raiz quadrada do maior grau no `ShatterPoint` $d_{maxsp}(v_i)$ versus $\lambda_{1,sp}$ no `ShatterPoint` (padrão *Root-degree*); (e) número de nós N_{gccsp} versus número de arestas E_{gccsp} da GCC no `ShatterPoint` (padrão *TreeGCC*); (f) número de Triângulos $\kappa_{3_{sp}}$ versus número de arestas $\mathcal{E}_{sp}$ (padrão *Triangle-Ratio*). A rede Power Grid tem um número desproporcional de triângulos. Apenas as redes com um ou mais triângulos são apresentadas nesse gráfico. 96
5.5 Distribuição do grau do nós das redes: (a) Amazon, (b) Gnutella, (c) AS Oregon e (d) Web Google 100
5.6 Nó de maior grau da rede Power-Grid: (a) Rede original ($|\mathcal{V}| = 20$ e $|\mathcal{E}| = 29$) e (b) `ShatterPoint` ($|\mathcal{V}_{sp}| = 19$ e $|\mathcal{E}_{sp}| = 25$) 102

5.7 Escalabilidade 104
5.8 `ShatterPlots` da rede *Erdös-Rényi* com 500k arestas usando `Eigenvalue ShatterPlots` (triângulos azuis) e `Proportional ShatterPlots` (círculos vermelhos). 104
5.9 Duas representações de nós com o centro em preto com suas respectivas vizinhanças são apresentados em (a) e (b). Os nós centrais (em preto) como mesmo valor para o coeficiente de clusterização 0.2 mas topologias diferentes. 106
5.10 Três subgrafos na qual o nó central u, v, w tem o mesmo grau $d(u) = d(v) = d(w) = 12$ e $\kappa_3(u) = \kappa_3(v) = \kappa_3(w) = 6$: (a) o nó central u tem apenas triângulos, (b) um κ_4 centrado no v, (c) quatro κ_4 e um κ_5 centrado no nó w 107
5.11 Distribuição do grau das redes AS, Email Enron, Epinions e Amazon. . . 114
5.12 Média de $\kappa_l(v_i)$ ($l \in 3, 4, 5$) vs grau do nó das redes AS, Email Enron, Epinions e Amazon. 115
5.13 Média de número de κ_4 vs. κ_3, Média de número de κ_5 vs. κ_3 e Média de número de κ_5 vs. κ_4 das redes AS, Email Enron, Epinions e Amazon . . . 117
5.14 Distribuição do κ_l com $l = 3, 4, 5$ para as redes (a) As, (b) Email Enron, (c) Epinions e (d) Amazon em cada coluna. A distribuição dos cliques segue uma lei de potência com o grau similar a distribuição do grau e as mesmas "quebras" como no caso da distribuição do grau do nó da rede Amazon. . . 118
5.15 Um grafo G e o nó $v_i = 1$ que tem as arestas de seus subgrafos representadas por linhas grossas. 120
5.16 Consulta SQL para recuperar todos os κ_4 da rede Epinios 121
5.17 Grafo correspondente a Consulta SQL da Figura 5.16 121
5.18 Consulta SQL para recuperar todos os κ_4 da rede Epinios utilizando a tabela `Subgrafos` 122
5.19 Grafo correspondente a Consulta SQL da Figura 5.18 122
5.20 Consulta SQL para recuperar todos os κ_5 da rede Epinios utilizando a tabela `Subgrafos` 123
5.21 Consulta SQL para recuperar todos os κ_5 da rede Epinios 123
5.22 Tabela (a) é a lista de aresta tradicional (a única usada no método tradicional). Tabela (b) é a tabela `Subgrafos`, que armazena todas as arestas dos subgrafos $\mathcal{G}_{s_i}$. 124
5.23 (a) Informações sobre a quantidade de tuplas nas tabelas: `Subgrafos` e de arestas; (b) Informações sobre a quantidade de κ_4 e κ_5; (c) Tempo de consulta dos κ_4 para o método tradicional e o `FCR`; (d) Tempo de consulta dos κ_5 para o método tradicional e o `FCR`; 125
5.24 Média de mil consultas na rede Amazon 126

Lista de Tabelas

4.1 Conjunto de dados utilizado para os experimentos 76
4.2 Número de agrupamentos encontrados no conjunto "Pendigits". 78
4.3 Número de agrupamentos encontrados no conjunto "OneBig" com amostragem de 2%, 20 dimensões e variação da quantidade de ruído. 79
4.4 Número de agrupamentos encontrados no conjunto "OneBig" com amostragem de 2%, 30% de ruído e variação da dimensionalidade. 80

5.1 Conjuntos de dados considerado nesta tese. O símbolo e o número em cada linha são utilizados nas figuras das próximas seções. 86
5.2 Número de $\kappa_3(v_i)$, $\kappa_4(v_i)$, $\kappa_5(v_i)$, Coeficiente de Clusterização Genérico($C_5(v_i)$) e `eigenvalue drop` para 7 nós da rede AS todos os nós com grau igual $d(v_i) = 30$. O valor real do primeiro autovalor da rede é $66,1191$ 111
5.3 Número de $\kappa_3(v_i)$, $\kappa_4(v_i)$, $\kappa_5(v_i)$, grau do nó, `Coeficiente de Clusterização Generalizado` e `eigenvalue drop` para os 10 nós com maior grau da rede Email−Enron. O valor real do primeiro autovalor deste grafo é $118,4175$ 112
5.4 O valor de $\kappa_3(\mathcal{G})$, $\kappa_4(\mathcal{G})$ e $\kappa_5(\mathcal{G})$ na redes AS, Email Enron, Epinions e Amazon. 113

Lista de Algoritmos

4.2.1 Biased Box Sampling: Criação da árvore - BBSCT 71
4.2.2 Biased Box Sampling: Join List - BBSJL 72
4.2.3 Biased Box Sampling: Extrai amostra - BBSES 74
4.2.4 Biased Box Sampling 74
5.3.1 `ShatterPlots` Adaptativo 92
5.4.1 Criação da `Subgrafos` 120

Símbolos

E	número de atributos
$\mathcal{G}$	Grafo, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
$\mathcal{G}_s$	Subgrafo, $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$
H	Grafo temporário, $H = (\mathcal{V}, \mathcal{E})$
$\mathcal{E}$	Conjunto de arestas do grafo $\mathcal{G}$
$\mathcal{E}_s$	Conjunto de arestas do subgrafo $\mathcal{G}_s$
$\mathcal{E}_{sp}$	Conjunto de arestas do grafo $\mathcal{G}$ no `ShatterPoint`
$\mathcal{E}_{spgcc}$	Conjunto de arestas da GCC do grafo $\mathcal{G}$ no `ShatterPoint`
$\mathcal{E}_t$	Número de arestas total do grafo
$\|\mathcal{E}\|$	Número de arestas do grafo
$e = (u, v)$	aresta de um grafo
$\mathcal{V}$	Conjunto de nós do grafo $\mathcal{G}$
$\mathcal{V}_s$	Conjunto de nós do subgrafo $\mathcal{G}_s$
$\mathcal{V}_{sp}$	Conjunto de nós do grafo $\mathcal{G}$ no `ShatterPoint`
$\mathcal{V}_{spgcc}$	Conjunto de nós da GCC do grafo $\mathcal{G}$ no `ShatterPoint`
$\mathcal{V}_t$	Número de nós total do grafo
$\|\mathcal{V}\|$	Número de nós do grafo
GCC	maior componente conexa do grafo
$Npairs_{sp}$	número de pares alcançáveis do grafo $\mathcal{G}$ no `ShatterPoint`
A	Matriz de Adjacência
N	Número de elementos de um conjunto de dados ou quantidade de nós
$\mathcal{N}(v_i)$	Vizinhança do nó v_i
n	Número de elementos na amostra gerada
M	Quantidade de Arestas
$\mathcal{S}$	Amostra de um conjunto de dados
ϕ	um certo valor de grau
N_ϕ	quantidade de nós com grau ϕ

$d(v_i)$	grau do nó v_i
d_{max}	maior nó grafo
$d_{out}(v_i)$	grau de saída do nó v_i
d_{maxsp}	grau máximo do grafo no `ShatterPoint`
$d_{in}(v_i)$	grau de entrada do nó v_i
$P(v, u)$	caminho do nó v ao nó u
$\mathcal{D}$	Diâmetro do grafo
$\mathcal{D}_e$	Diâmetro do efetivo do grafo
$\mathcal{D}_{sp}$	Diâmetro efetivo do grafo no `ShatterPoint`
Q_t	quantidade de arestas inseridas no algoritmo `ShatterPlots`
t	unidade de tempo
y	expoente da lei de potência
α	expoente da lei de potência
u, v, w	nós do grafo
l	número de nós do clique (tamanho)
κ_l	Clique de tamanho l
λ_1	primeiro autovalor da matriz de adjacência do grafo $\mathcal{G}$
$\lambda_{1,sp}$	primeiro autovalor da matriz de adjacência do grafo $\mathcal{G}$ no `ShatterPoint`
R	número de níveis da árvore *MG-Tree*
ϵ	raio do algoritmo DBSCan
$Ratio$	taxa de amostragem

Siglas

ANF	Approximate Neighbourhood Function.
BBS	Biased Box Sampling.
BIRCH	Balanced Iterative Reducing and Clustering using Hierarchies.
DBS	Density Biased Sampling.
DBSCan	Density-Based Spatial Clustering of Applications with Noise.
CDF	Cumulative Distribution Function.
CURE	Clustering Using Representatives.
EM	Expectation Maximization.
ER	*Erdös-Rényi*
FCR	Fast Clique Retrieval.
GBS	Grid Biased Sampling.
GCC	Giant Connected Component.
ICMC	Instituto de Ciências Matemáticas e de Computação.
KDD	Descoberta de Conhecimento em Bases de Dados.
KNN	K nearst neighbor.
LiBOC	Linear Box Counting
MBR	*Minimum Bound Retangle.*
MG-Tree	*Multi Grid Tree* - Árvore Multi Resolução
MGc-tree	*Multi Grid Compact Tree* - Árvore Multi Resolução Compacta
PDF	Probabilistic Density Function
SGBD	Sistema Gerenciador de Bases de Dados
SQL	Structure Query Language.
US	Uniform Sampling.

Capítulo 1

Introdução

O volume de dados armazenados e disponibilizados para a mineração tem crescido consideravelmente e estão cada vez mais diversificados quanto a sua estrutura e distribuição. Os dados podem ser divididos em dados tradicionais (número e pequenas cadeias de caracteres) e dados complexos (imagens, cadeias de DNA, vídeos, etc). Para estes tipos de dados, a área de mineração de dados possuí algoritmos bem consolidados para tarefas como classificação, regras de associação e detecção de agrupamentos, etc. Entretanto, muitos desses algoritmos, que produzem resultados de qualidade, exigem um grande processamento computacional, como os algoritmos de detecção de agrupamentos que podem muitas vezes ter complexidade cubica. Este alto custo computacional, faz com que seja necessário o uso de técnicas de redução de elementos para a utilização eficiente desses algoritmos. Contudo, os dados reais são marcados pela não uniformidade da sua distribuição, presença de uma grande quantidade de atributos e de elementos considerados ruídos (elementos resultantes de medições incorretas) o que torna ineficiente o uso de técnicas simples, como a amostragem uniforme.

Nos últimos anos, grande parte dos dados disponíveis para a mineração são rede

complexa, que são melhores representadas por uma estrutura mais rica do que a estrutura tradicional utilizada. Esta estrutura é chamada de grafo, seus elementos são chamados de vértices ou nós e as ligações entre esses elementos são chamadas de arestas (Getoor e Diehl, 2005a). As redes complexas tem como característica principal representar o inter-relacionamento entre os elementos do conjunto.

A área de mineração de redes complexas vem crescendo rapidamente e é uma área multidisciplinar, já que, engloba a área de mineração de dados, banco de dados, aprendizado de máquina e outras áreas como ciências sociais, biologia e química. Os dados tradicionais e complexos podem ser transformados em redes complexas (Getoor e Diehl, 2005b). Por exemplo, um conjunto de pessoas de uma empresa em uma tabela pode, por meio de um relacionamento "trabalha com", ser representado como em uma rede complexa que representa o relacionamento das pessoas que trabalham juntas. Um outro exemplo interessante pode ser visto no aplicativo Image swirl (http://image-swirl.googlelabs.com/) da Google, na qual a busca por imagens é representada por um grafo que mapeia as imagens similares agrupadas . Um exemplo de consulta nesta ferramenta da Google é apresentado na Figura 1.1.

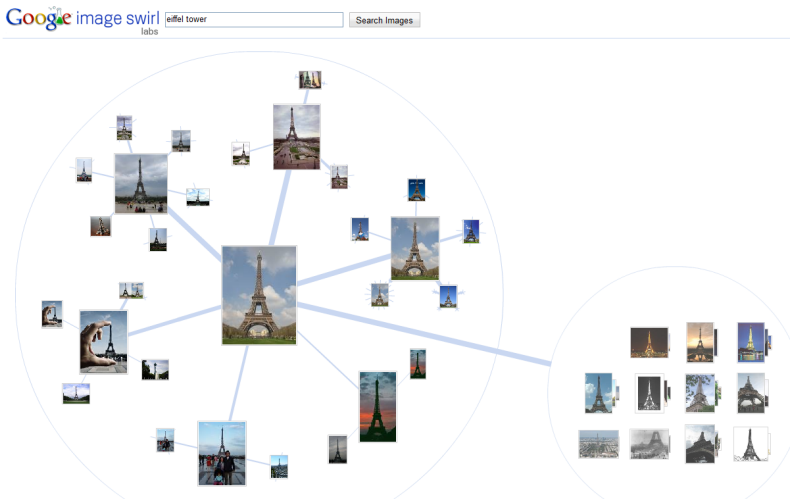


Figura 1.1: Consulta por similaridade representada por um grafo feita na ferramenta Image swirl da Google.

A área de mineração de redes complexas tem como grande desafio o desenvolvimento de novos algoritmos e novos métodos para a extração de conhecimento dessas redes, pois, não é possível neste tipo de dado a utilização de algoritmos tradicionais de mineração

de dados não é possível. A extração de padrões e propriedades desses dados permite que haja um melhor entendimento sobre esse novo tipo de dado permitindo o desenvolvimento de melhores geradores sintéticos de redes complexas.

1.1 Motivação

O processo de descoberta de conhecimento, conhecido como KDD (Knowledge Discovery in Database), é um processo interativo e iterativo com diversas fases, sendo a mais conhecida delas a mineração de dados. A mineração é composta por diversas tarefas, das quais podemos destacar a classificação, regras de associação e detecção de agrupamentos. Muitas dessas tarefas têm um custo computacional elevado, o que dificulta a sua utilização em grandes conjuntos de dados. A comunidade científica tem feito um grande esforço no desenvolvimento de algoritmos cada vez mais rápidos, entretanto o crescimento do volume dos dados supera a velocidade de desenvolvimento desses novos algoritmos.

Assim, a necessidade de extração de um subconjunto dos dados de tamanho significativamente menor e que seja suficientemente representativo do conjunto original tem se tornado uma tarefa de suma importância para o uso de técnicas de mineração e o sucesso do processo de KDD. As técnicas de extração de um subconjunto (amostra) devem garantir que o resultado obtido com a amostra gerada seja similar ao que será obtido com o conjunto original.

A escolha aleatória dos elementos é a maneira mais simples e intuitiva de geração de uma amostra. Neste processo, cada elemento do conjunto original tem a mesma probabilidade de ser escolhido. Entretanto, a aleatoriedade pode fazer com a amostra gerada não represente o conjunto de maneira adequada. Um exemplo é a realização de uma amostragem aleatória de um grupo de pessoas, se 10 pessoas forem escolhidas a probabilidade de serem 5 homens e 5 mulheres é grande. Contudo essa uniformidade na distribuição dos grupos é irreal (Cochran, 1977).

As redes complexas também estão em grande crescimento nos últimos anos. Um exemplo é o crescimento do número de usuários de redes sociais como Orkut (`www.orkut.com`) e Facebook (`www.facebook.com`). Este novo tipo de dado tem atraído

diversos pesquisadores da comunidade de mineração de dados e banco de dados, já que algoritmos tradicionais dessa área não podem ser utilizados de maneira direta para a mineração desse novo tipo de dado (Newman, 2003).

A pesquisa em rede complexas pode ser dividida em três grandes áreas. A primeira, tem como objetivo a extração de propriedades estatísticas, que são basicamente centradas no grau do vértice, que é número de arestas que cada vértice possui conectado à ele, e na distância entre os vértices. Dentre as propriedades estatísticas podemos destacar a distribuição do grau (Chakrabarti e Faloutsos, 2006), número triângulos (Tsourakakis, 2008), o diâmetro pequeno ("Small-World") (Milgram, 1967), entre outras. A segunda área de pesquisa, é o desenvolvimento de modelos de redes que permitam entender o significado das propriedades estatísticas extraídas e o como e porque as redes seguem tais propriedades. Por fim, a terceira área de pesquisa tem como objetivo a predição do comportamento das redes e sua evolução, isto é, como elas crescem e como esse crescimento pode ser predito.

Esta tese tem por objetivo a manipulação e mineração de grandes conjuntos de dados, que é o primeiro desafio da SBC (Sociedade Brasileira de Computação) dentre os cinco desafios propostos em (Carvalho et al., 2006). O primeiro tipo de dado tratado é o multidimensional, que são conjuntos de dados que possuam uma grande quantidade de atributos. Para esta tipo de dado foi estudado e desenvolvida uma técnica de amostragem de dados que permita a extração de uma amostra representativa do conjunto original, isto é, uma amostra que seja capaz de produzir resultados muito semelhantes aos obtidos o conjunto de dados original fosse utilizado. A técnica desenvolvida prioriza o pré-processamento de dados para a atividade de detecção de agrupamento e conjunto de dados com agrupamentos de tamanhos diferentes, que possuam ruídos e uma grande quantidade de atributos.

O segundo objeto de estudo foram as redes complexas. Sobre estes dados explorou-se o armazenamento e a mineração de grafos com a descoberta de novas propriedades estatísticas. O trabalho com redes complexas foi reforçado pela parceria que já ocorre há anos entre o Grupo de Base de Dados e Imagens (GBDI) do Instituto de Ciências

Matemáticas e de Computação (ICMC) - USP e a Carnegie Mellon University juntamente com o Prof. Dr. Christos Faloutsos, que é um pesquisador da área de mineração de redes complexas internacionalmente reconhecido.

1.2 Objetivos

O objetivo desta tese é definir e desenvolver métodos para o manuseio eficiente de grandes conjuntos de dados em tarefas de mineração e consultas a dados simples e redes complexas. O primeiro tipo de dado abordado foram os dados multidimensionais com características adversas, como ruídos e densidades distintas na sua distribuição. O segundo tipo foram dados chamados de redes complexas.

Para cada um dos tipos de dados tratados foram desenvolvidos métodos específicos. Para os dados multidimensionais explorou-se o pré-processamento de dados como uma ferramenta importante para o sucesso do processo de KDD. Assim, o objetivo do trabalho foi o desenvolvimento de uma técnica simples que permita a extração rápida e eficiente de uma amostra representativa que permita o uso de algoritmos tradicionais de descoberta de agrupamentos.

Em seguida foram desenvolvidos algoritmos para a extração de novas propriedades estatísticas das redes complexas de maneira que se permita por meio dessas propriedades estatísticas distinguirem redes complexas reais de redes geradas sinteticamente. Também se explorou o armazenamento de redes complexas em sistemas de gerenciamento de base de dados relacionais (SGBDR) para a posterior mineração dessas redes.

1.3 Resultados Obtidos

As contribuições trazidas por esta tese na área de banco de dados, mineração de dados e mineração de redes complexas são:

- Desenvolvimento do algoritmo *BBS - Biased Box Sampling* (Appel et al., 2007a, 2006, 2007b) que realiza amostragens em conjunto de dados multidimensional mesmo que possua agrupamentos desbalanceados e ruído.

- Desenvolvimento do algoritmo `ShatterPlots` (Appel et al., 2009) que permite a separação de redes complexas reais e sintéticas.

- Desenvolvimento de uma técnica, chamada `FCR - Fast Clique Retrieval`, que permite a contagem e listagem de cliques de tamanho 4 e 5 (κ_4 e κ_5) em redes complexas armazenando-as em sistemas gerenciadores de banco de dados relacional (SGBD).

- Extensão do coeficiente de clusterização, utilizado para determinar quão próximo uma rede complexa esta de ser um clique, para a identificação de diferentes topologias locais de um nó.

1.4 Organização do Trabalho

O restante deste trabalho esta organizado da seguinte maneira:

Parte I: apresenta o embasamento teórico e os trabalhos correlatos utilizados nesta tese:

- **Capítulo 2** apresenta os trabalhos estudados na literatura sobre descoberta de conhecimento em base de dados, sendo o foco a amostragem de dados e a detecção de agrupamentos.
- **Capítulo 3** apresenta os conceitos básicos da teoria de grafos e os trabalhos estudados na literatura sobre mineração de redes complexas.

Parte II: apresenta os trabalhos desenvolvidos nesta tese:

- **Capítulo 4** apresenta a técnica de amostragem desenvolvida e o algoritmo BBS - Biased Box Sampling que a implementa, e os experimentos realizados para a comprovação da eficiência do algoritmo.
- **Capítulo 5** apresenta os algoritmos e as técnicas desenvolvidas para a mineração das redes complexas e os resultados obtidos. Primeiramente é apresentado o algoritmo `ShatterPlots` juntamente com os padrões encontrados por meio

dele. Em seguida é apresentada a técnica `FCR - Fast Clique Retrieval`, a extensão feita ao coeficiente de clusterização e as leis de potência sobre cliques de tamanho 4 e 5 encontradas.

Parte III: Nesta parte são apresentadas as considerações finais deste trabalho.

- **Capítulo 6** apresenta as conclusões, contribuições, outras colaborações, publicações resultantes e trabalhos futuros desta tese.

Parte I

Conceitos, definições e trabalhos correlatos

Capítulo 2

Descoberta de Conhecimento em Base de Dados

2.1 Considerações iniciais

Os dados coletados nos últimos anos têm aumentado não só em volume, mas também em complexidade e dimensionalidade (número de atributos). O processo de Descoberta de Conhecimento em Base de Dados, mais tradicionalmente conhecido como *KDD - Knowledge Discovery in Database*, é um processo interativo, iterativo e composto de uma série de etapas sequenciais que podem ser retomadas dependendo das descobertas realizadas (ou a falta delas).

Por ser um processo exploratório, é desejável que o processo de KDD produza respostas rápidas. Entretanto, devido ao grande volume de dados e ao alto custo computacional dos algoritmos de extração de conhecimento, isso nem sempre é possível. Esse fato torna muitas vezes uma resposta aproximada e rápida mais interessante do que uma resposta exata e demorada.

Outra questão importante do processo de KDD é o tipo de dado analisado. A

grande maioria dos algoritmos de extração de conhecimento exige que os dados a serem analisados não possuam muitos atributos, estejam limpos e sem ruídos.

A distribuição dos dados também é muito importante, já que há conjuntos de dados que possuem regiões com quantidades (densidade) de elementos muito diferentes, chamados **desbalanceados** (Chawla et al., 2004), nas quais as tarefas de mineração de dados, como a classificação e a detecção de agrupamentos, podem produzir um resultado errado se não houver um cuidado maior na preparação destes dados.

Assim, um dos objetivos dessa tese foi desenvolver um método que permita a redução do conjunto de dados por meio da extração de uma amostra representativa, que permita aos algoritmos de detecção de agrupamentos tradicionais encontrarem grupos de maneira mais eficiente, em conjuntos de dados desbalanceados, com uma grande quantidade de atributos e presença de ruído.

Para atingir esse objetivo, neste Capítulo é feito o levantamento dos principais conceitos envolvidos no Processo de Descoberta de Conhecimento em Banco de Dados com especial atenção as técnicas de amostragem de dados.

2.2 O processo de descoberta de conhecimento em base de dados

O processo de Descoberta de Conhecimento em Base de Dados foi definido primeiramente por U. Fayyad (Fayyad et al., 1996a) como sendo " *o processo de identificação de padrões válidos, inéditos, potencialmente úteis e essencialmente compreensíveis embutidos nos dados*".

O processo de KDD consiste de múltiplos passos, que são executados em sequência. Cada passo subsequente é inicializado após a finalização com sucesso do passo anterior e requer o resultado deste como entrada. Outra característica importante do processo de KDD é a variedade de atividades abrangidas por ele, como o entendimento do domínio de aplicação e dos dados, a preparação e análise dos dados e a avaliação, entendimento e aplicação dos resultados gerados. O modelo do processo de KDD enfatiza a iterativi-

dade, já que qualquer uma das etapas pode ser revisitada se o resultado obtido não for satisfatório (Fayyad et al., 1996a,b). Um diagrama esquemático do processo de KDD é apresentado na Figura 2.1.

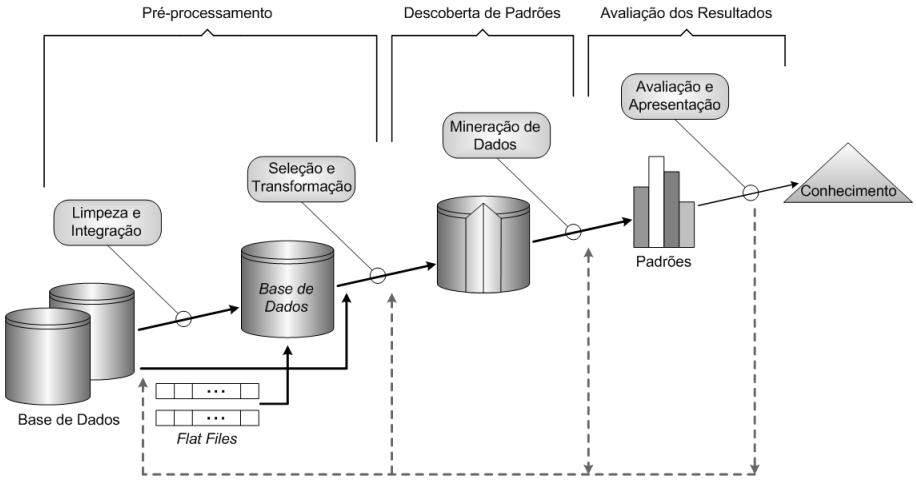


Figura 2.1: Passos do processo de KDD segundo (Fayyad et al., 1996b,c)

O processo de KDD proposto originalmente em (Fayyad et al., 1996a) é dividido em nove etapas. Contudo, este processo vem sendo modificado e melhorado por diversos pesquisadores da área. Nesta tese o modelo seguido é o apresentado em (Rezende, 2003), nas quais as etapas do processo de KDD são divididas em 3 grandes grupos descritos a seguir:

Pré-Processamento: Etapa na qual os dados disponíveis são preparados para análise, por meio da realização de operações como limpeza, integração, seleção e transformação dos dados.

Descoberta de Padrões: Principal etapa do processo de KDD. É nela que os dados preparados na etapa anterior são enviados a algoritmos de mineração de dados específicos, de acordo com o tipo de tarefa de mineração a ser realizada.

Avaliação: Etapa final em que os resultados obtidos pelos algoritmos de mineração de dados são avaliados quanto à sua utilidade e apresentados por meio de técnicas de representação de conhecimento e visualização (gráficos, tabelas, regras, etc), para que possam ser entendidos.

Um detalhe importante do processo de KDD é a possibilidade da sua aplicação em grande volume de dados. Quando foi concebido, o processo de KDD obteve inúmeros adeptos, e com o crescimento do volume de dados, este número se tornou ainda maior. Com o passar dos anos, novos algoritmos e técnicas têm sido propostos para a mineração de diversos tipos de dados, já que os dados também têm crescido quanto à variedade. Entretanto, o grande volume de dados fez com que o foco do processo de KDD não se restringisse apenas a algoritmos eficientes e rápidos para a descoberta de padrões. As tarefas de pré-processamento, como a redução do volume de dados se tornaram de suma importância, pois, apesar do crescente poder computacional, de processamento e armazenamento, ainda há restrições neste aspecto, já que, o processo de KDD possui um elevado custo computacional.

2.3 O pré-processamento de dados

Os dados reais tendem a ser incompletos, inconsistentes e com medidas ruidosas, resultantes de medições imperfeitas (Xiong et al., 2006). O pré-processamento de dados é responsável por melhorar a qualidade dos dados e com isso aumentar a acurácia e eficiência do processo de mineração de dados. Além disso, as decisões são baseadas em dados de qualidade. Outra tarefa importante no pré-processamento é a redução de dados que ajuda a diminuir o alto custo computacional do processo de KDD. Isso se deve ao fato que os algoritmos de mineração na sua maioria não escalam bem quando o volume de dados é muito grande. O pré-processamento de dados é responsável por 80% do esforço empregado durante o processo de KDD (Zhang et al., 2005) e segundo (Han e Kamber, 2000) compreende as seguintes tarefas:

Limpeza dos dados: Responsável pela remoção de dados ruidosos, valores faltantes e inconsistentes vindos do processo de coleta dos dados;

Integração: Responsável pela combinação de diversas fontes de dados;

Transformação: Responsável pela transformação dos dados em formatos apropriados;

Seleção e Redução: Responsável pela seleção de objetos ou atributos obtendo um conjunto de dados que possua uma representação menor.

Embora todas as etapas do pré-processamento sejam importantes, o foco do trabalho desenvolvido nesta tese está centrado na redução do número de elementos, que é detalhada nas próximas seções.

2.3.1 A redução de dados

A redução do conjunto de dados corresponde à escolha dos elementos ou de atributos que melhor caracterizam o conjunto de dados como um todo. Esta tarefa tem destaque dentro do pré-processamento, já que, o tamanho do conjunto de dados interfere drasticamente no desempenho dos algoritmos de mineração de dados. Diversas técnicas de redução de dados têm sido usadas para produzir uma resposta aproximada que apresente um alto custo benefício, isto é, rapidez na execução dos algoritmos, qualidade e a confiabilidade nos resultados apresentados.

A tarefa de redução de dados pode ser dividida em três grupos segundo (Rezende, 2003), que são apresentados a seguir.

Redução do número de atributos

A redução do número de atributos é a técnica que visa reduzir o número de atributos que descrevem o conjunto de dados a ser analisado na etapa de mineração de dados.

O grande número de atributos, também chamado de dimensão do conjunto de dados, aumenta a complexidade das técnicas de manipulação e degrada o desempenho dos algoritmos de mineração de dados (Fodor, 2002). Para diminuir esses efeitos, as técnicas de redução de atributos têm por objetivo representar um conjunto de dados com um número menor de atributos procurando manter as características do conjunto.

Os processos de redução de dimensionalidade podem ser divididos em processos de extração de atributos e processos de seleção de atributos. Um processo de extração de atributos busca alterar a representação de um conjunto de dados, de maneira que a nova representação apresente uma dimensão menor do que a representação original,

procurando manter as características inerentes da informação armazenada (Hair et al., 1998). Um processo de seleção de atributos escolhe uma ou mais dimensões do conjunto que se julga serem mais importantes.

Redução do número de valores de um atributo

A redução do número de valores de um atributo é feita, geralmente, por métodos de discretização dos valores de um atributo contínuo. A discretização de dados é definida como sendo o processo de dividir valores contínuos em intervalos (Kotsiantis e Kanellopoulos, 2006).

Os tipos mais comuns de atributos usados na mineração de dados são os atributos nominais (categóricos), os contínuos e os discretos. Os atributos nominais frequentemente assumem um número limitado de valores e não existe necessariamente uma relação de ordem entre os valores. Já, os atributos contínuos podem assumir um número infinito de valores onde existe necessariamente uma relação de ordem entre eles. Os atributos discretos são atributos que possuem um número reduzido de valores (em comparação com os atributos contínuos) e preservam a relação de ordem entre os valores. A discretização é usada para mapear valores contínuos em discretos quando é necessário reduzir o número de valores dos atributos e, ao mesmo tempo, manter a relação de ordem entre os valores. Muitas vezes, a discretização é necessária por exigência do algoritmo de mineração.

Redução do número de elementos

A redução do número de elementos do conjunto tem por objetivo reduzir a quantidade de dados mantendo as mesmas características do conjunto original. As técnicas que visam reduzir o número de elementos do conjunto podem ser classificadas em dois tipos: as que definem um modelo que representa os dados e com isso armazenar apenas o modelo e seus parâmetros; e as que selecionam um número pequeno de elementos e os armazenam, para na fase posterior, aplicar um processo de mineração.

Dentre as técnicas que armazenam um modelo para a representação dos dados destaca-se a técnica de regressão, que também é considerada uma tarefa de mineração de

dados e é usada para ajustar uma equação em um conjunto de dados. O método mais simples de regressão é a regressão linear, que ajusta a equação de uma reta (y = ax+b) e determina o valor mais apropriado para "a" e "b" para predizer os valores de "y" baseados nos valores de "x". Contudo, os dados reais são caracterizados pela não uniformidade na distribuição, presença de ruído e grande número de atributos. Assim, para tais conjuntos esses modelos se tornam inadequados.

Entre as técnicas que selecionam um conjunto pequeno de elementos do conjunto original destacam-se as técnicas de histograma e a amostragem de dados (*sampling*). Esta última é a mais usada por ser uma técnica simples e que permite a representação reduzida do conjunto de dados por meio da escolha de alguns elementos.

Os histogramas (Han e Kamber, 2000) são uma técnica bastante tradicional para a redução do volume de dados. A construção de um histograma consiste em se fazer a distribuição de frequência de um ou mais atributos, isto é, dividi-los em intervalos disjuntos, sem intersecção, chamados "*buckets*" e contar a frequência dos valores dos atributos escolhidos dentro desse intervalo. Os "*buckets*" são dispostos em um eixo horizontal, enquanto a altura (área) do "*bucket*" tipicamente representa a média da frequência dos valores. Como exemplo pode-se fazer a contagem de quantas pessoas vivem em cada cidade, ou uma contagem de quantas pessoas há em cada faixa de idade de 10 em 10, começando em 0 e terminando em 60, como mostra a Figura 2.2.

Os tipos mais comuns de histogramas são os chamados "*Equal-width*" e "*Equal-depth*". O histograma "*Equal-width*", divide os atributos em N *buckets* de tamanhos iguais, já o histograma "*Equal-depth*" divide os atributos em N "*buckets*" de maneira que a quantidade de elementos em cada "*buckets*" seja aproximadamente a mesma.

A amostragem de dados é uma técnica que seleciona um subconjunto do conjunto original para ser processado pela tarefa de mineração. A escolha do subconjunto pode ser feita de maneira aleatória ou segundo algum critério, como está detalhado na próxima seção. As técnicas de amostragem tendem a ser bastante eficientes e rápidas sendo por isso escolhida como o foco de estudo desta tese para o pré-processamento dos dados.

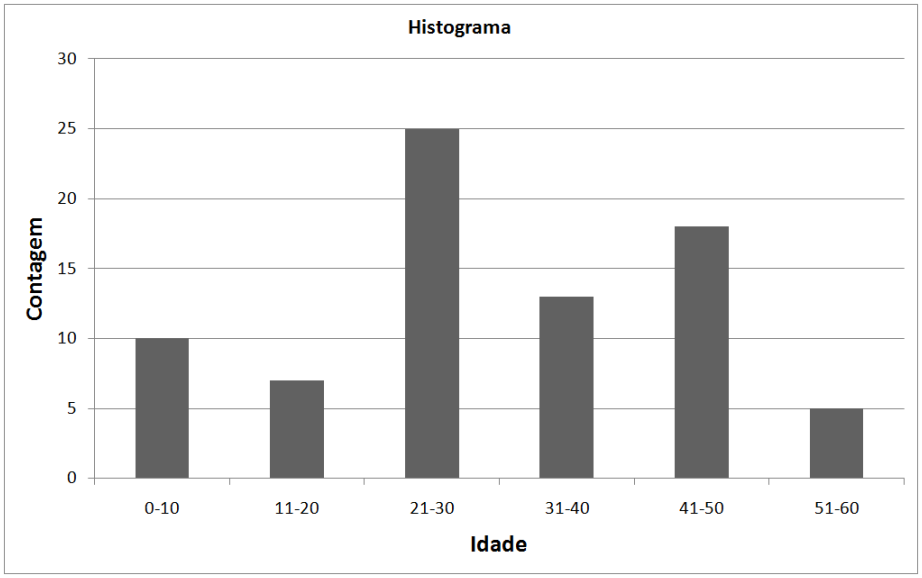


Figura 2.2: Exemplo de um histograma representando a quantidade de pessoas em um faixa etária - *buckets* de tamanho 10

2.3.2 Amostragem de dados

Amostragem de dados é um processo que seleciona alguns elementos do conjunto de dados para compor um subconjunto chamado amostra, tal que $n < N$, sendo que N representa o número total de elementos do conjunto original e n o tamanho da amostra $\mathcal{S}$ (Cochran, 1977).

A amostra $\mathcal{S}$ deve ser representativa, isto é, deve conservar características consideradas importantes pelo processo que utilizará a amostra, de maneira que os resultados obtidos usando apenas a amostra $\mathcal{S}$ sejam (quase) os mesmos obtidos quando usa-se todo o conjunto de dados. Um exemplo típico de aplicação prática da amostragem é a escolha de entrevistados para pesquisas de opinião pública, como eleições, avaliação de produtos, etc.

A amostragem de dados é uma técnica que tem sido aplicada com sucesso na preparação dos dados para a mineração dos mesmos de maneira mais rápida e eficaz. As tarefas de mineração de dados que mais fazem uso das técnicas de amostragem de dados são as técnicas de classificação e detecção de agrupamentos. No caso das técnicas de classificação a amostragem pode ser estratificada, usando a classe como atributo de estratificação. As técnicas de amostragem têm sido usadas com bastante eficácia para

conjuntos em que as classes possuem tamanhos muitos diferentes, como apresentado no Editorial (Chawla et al., 2004), chamados conjuntos desbalanceados (Batista et al., 2004; Chawla et al., 2005; Chen et al., 2009; Lu et al., 2008).

As técnicas de amostragem de dados podem ser usadas na preparação dos dados em diversas tarefas de mineração de dados, tais como: regras de associação (Chakaravarthy et al., 2009; Chen et al., 2002; Toivonen, 1996; Zaki et al., 1996), descoberta de agrupamento (Appel et al., 2007a, 2006, 2007b; Chehreghani et al., 2008; Fernandes et al., 2008; Kollios et al., 2003; Nanopoulos et al., 2006; Palmer e Faloutsos, 2000) e também na área de banco de dados ajudando na estimativa de custo de consultas e consultas aproximadas (Chaudhuri et al., 2007; Haas et al., 1995; Joshi e Jermaine, 2009; Ling, 1995; Olken e Rotem, 1995), *data streams* (Braverman et al., 2009; Chuang et al., 2009) e na mineração de grafos (Bar-Yossef e Gurevich, 2008; Krishnamurthy et al., 2007; Leskovec e Faloutsos, 2006; Tsourakakis et al., 2009). Esta última é abordada em mais detalhes no Capítulo 3.

A escolha da técnica de amostragem de dados mais adequada é dependente da tarefa de mineração a ser usada, bem como das características dos dados, como distribuição e presença de ruído. A amostragem de dados é caracterizada segundo a metodologia usada para escolher os elementos do subconjunto $\mathcal{S}$ (Cochran, 1977; Han e Kamber, 2000). Os principais tipos de amostragem são apresentados a seguir.

Amostragem Uniforme sem Repetição

Consiste em escolher uma amostra de um conjunto de dados, tal que todos os elementos do conjunto tenham a mesma probabilidade de serem selecionados. Neste tipo de amostragem, um elemento que foi selecionado não retorna para o conjunto para ser disponibilizado para a amostragem novamente. Assim, a probabilidade de um dado elemento estar em $\mathcal{S}$ na amostragem uniforme é $1/N$. Este tipo de amostragem é a mais simples e a mais usada. Contudo, ela é mais vulnerável a distorções devido ao processo aleatório, especialmente nos conjuntos desbalanceados. Suponha que no conjunto há dois grupos, que não são conhecidos pelo processo de amostragem, um grupo com 5000 elementos e o outro com 50 elementos, e se queira realizar uma amostragem de 1% do conjunto (50

elementos). Por ser uma quantidade pequena de elementos selecionados para a amostra, a chance é pequena de um elemento do grupo com 50 elementos ser escolhido.

Grande parte das técnicas de redução de elementos está baseada na amostragem de dados uniforme. No entanto a uniformidade na distribuição de valores e a independência entre atributos são propriedades bastante incomuns em dados do mundo real. Usualmente, dados reais são caracterizados pela ampla presença de correlações entre seus atributos e pela não uniformidade na sua distribuição. Assim, para esse tipo de dado, a amostragem balanceada é a mais aplicável.

Amostragem Uniforme com Repetição

Consiste em escolher uma amostra de um conjunto de dados, tal que todos os elementos do conjunto tenham a mesma probabilidade de serem selecionados. Neste tipo de amostragem, um elemento que foi selecionado retorna para o conjunto, para ser disponibilizado para a amostragem novamente.

Amostragem Sistemática

Apresenta características parecidas com a amostragem uniforme. Entretanto, nesta amostragem é necessário primeiro ordenar o conjunto em uma sequência, para em seguida, utilizar um intervalo fixo para escolher os elementos do conjunto. Por exemplo, em uma sequência retira-se um elemento de 5 em 5. Contudo é fácil notar que a amostragem sistemática é vulnerável à periodicidade da sequência criada.

Amostragem Estratificada

Este tipo de amostragem divide o conjunto de dados em subconjuntos, chamados *strata*, por meio de um ou mais atributos e depois faz uma amostragem uniforme em cada um dos subconjuntos gerados criando, no final, apenas uma única amostra $\mathcal{S}$. A divisão pode ser baseada em um atributo classificador do conjunto, por exemplo, masculino e feminino. A amostragem estratificada é um pouco mais eficiente que a amostragem uniforme, já que, por dividir o conjunto em subconjuntos, permite a inferência de características específicas

de cada subgrupo. Entretanto, muitas vezes é necessário criar diferentes subconjuntos usando atributos diferentes, para assim obter um melhor entendimento do conjunto de dados.

Amostragem por Agrupamento

Assim como a regressão, a detecção de agrupamentos pode ser vista como uma técnica para o redução do volume de dados. Como é visto em detalhes nas seções seguintes, a técnica de detecção de agrupamento tem por objetivo particionar o conjunto de dados em grupos. Com isso, para a redução de dados, basta escolher um ou mais elementos representantes de cada grupo para compor a amostra $\mathcal{S}$. Entretanto, esta é uma técnica custosa, já que, as técnicas de detecção de agrupamento exigem um alto custo computacional.

Amostragem Balanceada

A amostragem balanceada consiste em escolher uma amostra do conjunto de dados de maneira que alguns elementos tenham uma probabilidade maior de serem escolhidos do que outros elementos do conjunto. Esse desbalanço pode ser determinado por fatores como a variação na densidade na distribuição dos dados. Por exemplo, em um conjunto que possui mais homens do que mulheres, escolhe-se a mesma quantidade de homens e mulheres. Para a realização desse desbalanço duas técnicas são aplicadas, o "*oversample*" e o "*undersample*".

- O *oversample* consiste em recuperar mais elementos de uma região cuja densidade, isto é, a concentração de número de elementos, é menor do que em outras regiões do conjunto. Desta maneira, a representatividade desta região na amostra é aumentada, já que, dependendo do tamanho do conjunto amostra, a realização de uma amostragem uniforme nesta região não recuperaria uma quantidade suficiente de pontos que permitisse o algoritmo de detecção de agrupamentos detectá-la. Quando o número de elementos da região em que esta se realizando o *oversample* é menor do que a quantidade de elementos que se deseja recuperar, pode se adotar duas abordagens: inserir o mesmo elemento mais de uma vez no conjunto amostra ou

criar elementos fictícios para esta região. Contudo, esta inserção pode causar erro no pós-processamento dos dados na etapa de mineração dos dados.

- O *undersample* consiste em recuperar menos elementos de uma região cuja densidade é maior do que em outras regiões do conjunto. Assim, a representatividade desta região na amostra é reduzida, diminuindo a diferença entre as regiões com mais e menos elementos.

Essas técnicas são aplicadas quando deseja-se realizar uma amostragem de um conjunto de dados com regiões ou grupos de densidade muito diferentes. Dessa maneira regiões menos densas, isto é, com menos elementos, são representadas no conjunto amostra. Analogamente, regiões mais densas têm a sua representação reduzida sendo representadas no conjunto amostra de uma maneira menos discrepante em relação a outras regiões.

Grid Biased Sampling - GBS

O problema da amostragem balanceada para detecção agrupamentos foi introduzido por Palmer e Faloutsos em (Palmer e Faloutsos, 2000) com o algoritmo chamado GBS - Grid Biased Sampling. A proposta apresentada por eles é baseada em uma abordagem da construção de um reticulado sobre os dados. Como o número de agrupamentos não é conhecido a priori, um reticulado é construído dividindo os dados em grupos. Os elementos de cada uma das células do reticulado são mapeados em uma tabela *hash* que será usada pelo algoritmo para realizar a amostragem. A quantidade de elementos a ser amostrada em cada parte da tabela *hash* é definida pelo valor e que está entre 0 e 1. Se e é próximo de 0 o algoritmo se assemelha a uma amostragem uniforme e se está próximo de 1 o algoritmo seleciona uma quantidade igual de elementos em cada uma das partes da tabela *hash*. O valor recomendado para e segundo os autores é $0,5$. Um exemplo de como os elementos, de cada uma das células do reticulado, são mapeados para a tabela *hash* é apresentado na Figura 2.3.

O algoritmo GBS é afetado pela colisão e pela existência de ruído, como mostram os experimentos realizados nesta tese e em outros trabalhos (Kollios et al., 2003; Nanopoulos et al., 2006).

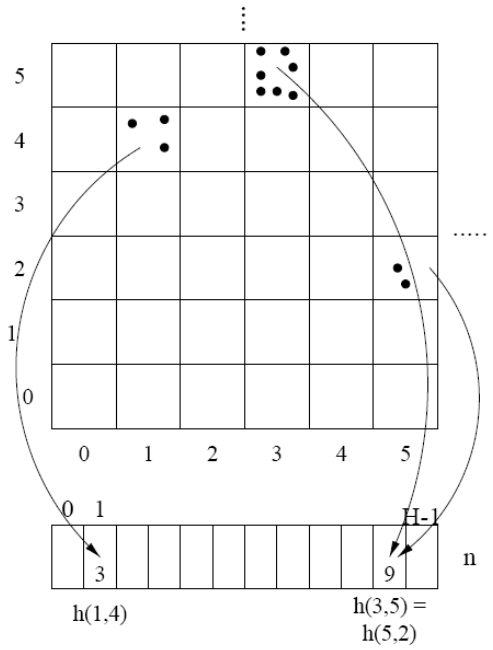


Figura 2.3: Exemplificação do algoritmo GBS (Palmer e Faloutsos, 2000)

Density Biased Sampling - DBS

Outro algoritmo proposto para o problema da amostragem balanceada é o DBS (*Density Biased Sampling*) apresentado em (Kollios et al., 2003). Neste algoritmo, os elementos são adicionados na amostra baseado na densidade local do conjunto de dados. Para a determinação da densidade local são utilizados métodos de estimação de densidade para contabilizar a relação de cada elemento com a sua vizinhança.

Particularmente, para um elemento x, $f(x)$ denota a função de estimação de densidade. Se $f(x) > 1$, então a densidade de x é maior que a densidade média do espaço inteiro e se $f(x) < 1$ então a densidade de x é menor que a densidade média do espaço inteiro. De acordo com isto, um elemento x é incluído na amostra com probabilidade:

$$P(incluindo\ elemento\ x) = \frac{n}{k} f^a(x) \tag{2.1}$$

Na Equação 2.1, n é o tamanho da amostra esperada, $k = \sum_x f^a(x)$ (usado para normalização do valor da probabilidade para ser menor ou igual a um) e a é o parâmetro de ajuste que muda conforme o tipo de conjunto utilizado.

Os resultados experimentais apresentados em (Kollios et al., 2003) mostram que

o método DBS melhora a qualidade da amostragem comparada com o GBS (Palmer e Faloutsos, 2000) e com a amostragem uniforme (US). Entretanto, a computação do estimador de densidade requer muito tempo computacional, já que, ele requer várias leituras do conjunto de dados e muito processamento, tornando-o impraticável para a tarefa de pré-processamento dos dados. Evidentemente, no caso de não ocorrer atualizações no conjunto de dados, o estimador de densidade (para cada elemento) pode ser computado apenas uma vez e, depois, esta informação pré-computada pode ser usada cada vez que o DBS é aplicado. Contudo, esta abordagem ainda requer (durante a execução DBS) o exame de cada elemento, o qual requer alto custo de processamento e de I/O.

R-Tree Sampling

Outra proposta de amostragem balanceada é apresentada em (Nanopoulos et al., 2006), na qual o autor propõe a criação de um índice espacial baseado na R-Tree para posterior execução da amostragem. A ideia desta abordagem é explorar o agrupamento dos dados inerente a organização dos dados em nós nas estrutura de indexação espacial. A organização dos elementos na R-Tree, para o propósito da amostragem balanceada, apresenta limitações, como a capacidade dos nós da R-tree, a divisão do nó em dois quando necessário e a dimensionalidade também influencia bastante o desempenho da estrutura.

Seja L o conjunto de nós folhas da R-Tree. Como é desejado que os elementos pertencentes ao mesmo nó folha tenham a mesma probabilidade de ser incluídos na amostra, a mesma densidade local é atribuída a todos os elementos de um nó folha. Seja f_i o a capacidade de um nó folha i. Consequentemente, uma aproximação da densidade local do elemento pode ser determinada utilizando o número f_i de cada nó folha i, dividido pelo volume do MBR_i (para o espaço 2D isto corresponde a área do MBR). Este valor da densidade é o mesmo de todos os elementos de i e é denotado por S_i como apresentado na equação 2.2.

$$S_i = \frac{f_i}{Volume(MBR_i)},\ i \in L \tag{2.2}$$

Dentre os métodos supracitados, o algoritmo DBS e a amostragem baseada na

R-Tree foram os métodos que apresentaram melhores resultados para os dados desbalanceados e com ruídos.

2.4 Mineração de dados

A mineração de dados é definida como sendo um conjunto de técnicas, procedimentos e algoritmos utilizados para a extração de padrões e conhecimento dos dados. Muitas vezes o termo mineração de dados é usado de maneira errônea como sinônimo de KDD. Contudo, a mineração de dados é simplesmente um dos passos do processo mais amplo de KDD.

Nos últimos anos a mineração de dados se estendeu para a mineração de estruturas mais complexas e com isso vem recebendo novas denominações como mineração relacional de dados, mineração de ligações e mineração de grafos.

Mineração de Dados: visa a descoberta de padrões em um conjunto de dados representados por uma única tabela;

Mineração Relacional de dados: visa a descoberta de padrões em um conjunto de dados representados por mais de uma tabela;

Mineração de Ligações: visa a descoberta de padrões em um conjunto de dados representados por mais de uma tabela sendo que a enfase da mineração esta na ligação entre as tabelas;

Mineração de Grafos: visa a descoberta de padrões em conjunto de dados representados como grafos.

Os trabalhos desenvolvidos nesta tese estão diretamente relacionados com a mineração de dados tradicionais e a mineração de grafos, assim, somente essas duas serão detalhas. Neste Capítulo será detalhada a mineração de dados tradicionais e no Capítulo 3 a mineração de grafos.

As tarefas de mineração de dados podem ser classificadas como de predição ou descrição. As tarefas de predição são responsáveis pela construção de modelos para predizer o

comportamento dos dados, por exemplo, a classificação e regressão, enquanto as tarefas de descrição são responsáveis por procurar padrões e propriedades presentes nos dados, como a detecção de agrupamento e as regras de associações. Dentre as tarefas de mineração destacam-se as tarefas de classificação, regras de associação e detecção de agrupamentos. Esta classificação se estende aos outros tipos de tarefa de mineração. No Capítulo 3 serão apresentadas as principais técnicas de mineração para dados representados como grafos

2.4.1 Classificação

A tarefa de classificação é definida como sendo o processo de encontrar modelos ou funções que descrevem e distinguem conceitos ou classes dos dados a fim de ser capaz de utilizar o modelo criado para predizer as classes, que são representadas por um atributo discreto, dos elementos que possuem este atributo desconhecido. Os modelos são derivados a partir de um conjunto de dados previamente classificado (Han e Kamber, 2000; Rezende, 2003).

A classificação também pode ser usada, em muitas aplicações, para predizer valores de atributos perdidos ou não disponíveis sem ser as classes. A regressão é de maneira geral similar quanto a sua definição em relação à classificação, sendo a grande diferença que o atributo a ser predito é contínuo e não discreto.

2.4.2 Regras de Associação

A análise de regras de associação (Agrawal e Srikant, 1994) é a descoberta de regras que descrevem padrões de relacionamento entre itens de uma base de dados que ocorrem frequentemente juntos. Estas regras representam combinações de itens que ocorrem com frequência em uma base de dados. Uma de suas aplicações típicas é a análise de transações de compra (*market basket analysis*). A partir de uma base de dados que armazena produtos comprados por clientes, por exemplo, de um supermercado ou de uma loja de departamentos, uma estratégia para a mineração de regras de associação poderia gerar o seguinte exemplo: *fralda* $\wedge$ *sábado* $\rightarrow$ *cerveja*. Esta regra é usada para indicar que os clientes que compram fraldas ao sábado, tendem também a comprar cerveja.

2.4.3 Detecção de agrupamentos

A atividade de detecção de agrupamentos pode ser definida como sendo o processo de organização de elementos de um conjunto em grupos, cujos membros são similares em alguma característica. Sendo assim, o problema de agrupamento de dados é determinar os grupos intrínsecos de um conjunto de dados sem rótulos. Um exemplo é mostrado na Figura 2.4, na qual alguns dados são agrupados com $k = 3$ utilizando a característica de proximidade (Han e Kamber, 2000; Jain et al., 1999; Rezende, 2003).

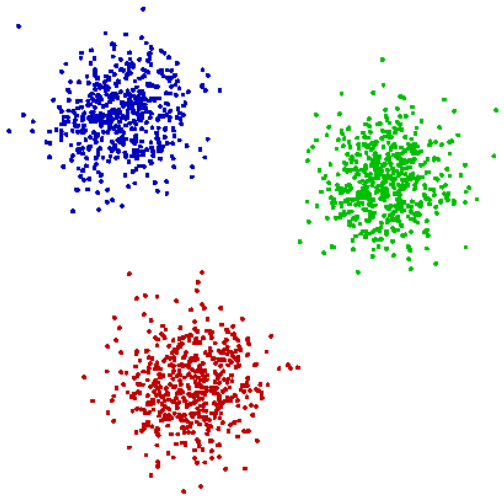


Figura 2.4: Um exemplo de um conjunto de dados agrupado em três grupos.

Usualmente, os algoritmos de agrupamentos de dados sofrem de problemas de tempo e complexidade, quando o volume dos dados aumenta em tamanho e dimensão. Com o aumento do volume de dados acumulados nas últimas décadas, a necessidade de que estes sejam analisados e o surgimento de conjuntos de dados de diversos tipos e distribuições fez com que diversos pesquisadores concentrassem seus esforços na tentativa de tornar o processo de detecção de agrupamentos possível também para grandes bases de dados (Berkhin, 2002; Carpineto et al., 2009; Han e Kamber, 2000; Kriegel et al., 2009; Lee et al., 2008).

A detecção de agrupamentos de dados é uma das principais técnicas empregadas no processo de KDD, podendo ser aplicada ou como ferramenta isolada, tendo como objetivo explorar a distribuição do conjunto de dados ou, como visto anteriormente, como uma tarefa de pré-processamento para outras técnicas. Os algoritmos de detecção de

agrupamentos podem ser classificados como: por particionamento, hierárquicos ou por densidade.

Particionamento

Dado um conjunto de N elementos, os métodos de particionamento constroem k partições de dados, em que cada partição representa um agrupamento, sendo $k \leq N$. Em suma, o método divide os dados em k grupos que satisfazem as seguintes condições:

1. cada grupo deve conter ao menos um elemento;

2. cada elemento deve pertencer a exatamente um grupo.

Algoritmos de detecção de agrupamento baseados em particionamento requerem o número de partições k usualmente fornecido pelo usuário. O processo se inicia criando-se uma partição inicial e depois melhorando-a, movendo-se os elementos de uma partição para outra. Há vários critérios para avaliar a qualidade de uma partição, sendo a mais comum a proximidade entre os elementos. Os dois métodos de detecção de agrupamentos baseados em particionamento mais conhecidos são o k-medoid (Ng e Han, 1994) e o k-means (MacQueen, 1967).

No algoritmo k-means, cada agrupamento é representado pelo seu elemento central, ou seja, pela média (ou média ponderada) dos elementos que o compõe (pelos centroides), enquanto no algoritmo de k-medoid, o representante de cada agrupamento é o elemento que esta o mais próximo possível do centro do agrupamento. Os três algoritmos baseados em k-medoids mais conhecidos são: PAM (*Partitioning Around Medoids*), CLARA (*Clustering LARge Applications*) e CLARANS (*Clustering Large Applications based upon RANdomized Search*) (Kaufman e Rousseeuw, 2005). O algoritmo PAM foi um dos primeiros algoritmos desenvolvidos tendo como base o algoritmo k-medoid, que baseia-se em um processo de otimização iterativo, que avalia o efeito da troca de um objeto medóide por um objeto não-medóide na qualidade do agrupamento resultante.

A complexidade computacional elevada do algoritmo PAM motivou o desenvolvimento do algoritmo CLARA, um algoritmo de agrupamento baseado em amostragem

(*sampling*) (Kaufman e Rousseeuw, 2005). A ideia fundamental desse algoritmo consiste em selecionar os k medóides de um conjunto de dados a partir da análise de apenas uma pequena amostragem desses dados. O CLARA seleciona várias amostras de um conjunto de dados, aplica o PAM em cada amostra e retorna como saída o melhor conjunto de medóides obtidos a partir dessas amostras. Nesse algoritmo, a qualidade do agrupamento obtido está relacionada ao tamanho das amostras, que é definido pelo usuário antes de sua execução.

O algoritmo CLARANS foi desenvolvido para a mineração de dados espaciais (Ng e Han, 1994). Ele utiliza uma estratégia baseada em busca aleatória com o objetivo de superar ambos PAM e CLARA em relação à eficiência computacional (complexidade computacional ou tempo de processamento) e à qualidade do agrupamento resultante (distorção média ou distância entre os objetos e seus medoides), respectivamente.

Os algoritmos baseados em particionamento usualmente são adequados para conjuntos pequenos, com agrupamentos em formato de bola e com pouco ruído. Para outros tipos de dados, como os desbalanceados ou com agrupamentos de diferentes formatos, os algoritmos de particionamento precisam ser estendidos.

Hierárquicos

A base dos algoritmos de detecção de agrupamentos hierárquico é a criação de uma hierarquia de agrupamentos, formada por vários níveis de partições aninhadas do conjunto de dados. De acordo com o modo como a hierarquia de agrupamentos é construída, os métodos empregados por esse tipo de algoritmo podem ser divididos em: aglomerativos (*bottom-up*) ou divisivos (*top-down*).

Os métodos aglomerativos começam com cada elemento sendo um agrupamento. Nos passos seguintes são feitas as junções desses agrupamentos em agrupamentos maiores. Os métodos divisivos começam com o conjunto todo sendo um único agrupamento que é dividido em agrupamentos menores sucessivamente pelo algoritmo, até chegar ao número de agrupamentos desejados. Os métodos divisivos raramente são aplicados. Como exemplo dos algoritmos aglomerativos tem-se o BIRCH (Zhang et al., 1996) e o CURE (Guha

et al., 1998).

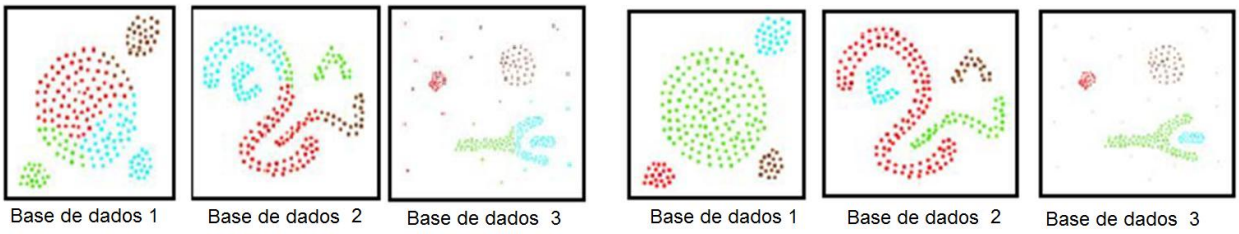
O algoritmo BIRCH (*Balanced Iterative Reducing and Clustering using Hierarchies*) utiliza uma estrutura em árvore especializada para realizar o agrupamento de grandes conjuntos de dados multidimensionais. Para tanto, ele calcula incrementalmente descrições compactas de subagrupamentos dos dados, chamadas de características do agrupamento (*Clustering Feature - CF*). As *CFs* são organizadas em uma árvore balanceada construída por meio da busca sequencial sobre todo o conjunto, de maneira similar à construção da *B+-tree*. Os valores das *CFs* são suficientes para obter informações sobre todo o conjunto de dados como centroides, raio e diâmetro. O algoritmo BIRCH foi o primeiro algoritmo a manusear dados que possuem ruído. Entretanto, ele mantém a característica dos algoritmos de particionamento ao encontrar somente agrupamentos com formato de bola.

O CURE é um algoritmo hierárquico aglomerativo que utiliza uma política mista para o cálculo da distância entre dois agrupamentos, a cada iteração. Esta política é uma mistura entre a política dos centroides (onde a distância entre dois agrupamentos é a distância entre seus centros de gravidade) e a chamada política *MST* (*Minimum Spanning Tree*) em que a distância entre dois agrupamentos é igual à distância mínima entre dois elementos, um em cada agrupamento. Entretanto, a sua complexidade é $O(N^2)$, usando a amostragem uniforme para ajustá-lo a grandes conjuntos de dados. O CURE, assim como os outros métodos, não consegue manusear agrupamentos desbalanceados.O CURE é capaz de encontrar agrupamentos com formas arbitrárias o que normalmente não acontece com os métodos por particionamento. Além disso, este algoritmo também é robusto quanto à presença de ruídos e uma maior quantidade de atributos (dimensões) (Ertoz et al., 2003).

Densidade

Algoritmos de detecção de agrupamentos baseados em densidade são utilizados para a descoberta de agrupamentos de formato arbitrário, especialmente em conjuntos que possuem dados ruidosos. Neste tipo de método, um agrupamento é considerado uma região

em que a densidade de elementos excede um limiar, isto é, para cada elemento de um agrupamento, a vizinhança em até um determinado raio tem que conter uma quantidade mínima de elementos. O algoritmo DBScan (*Density Based Spatial Clustering of Applications with Noise*) (Ester et al., 1996) é o principal representante desta categoria. As Figuras 2.5 (a) e (b) comparam os resultados da detecção de agrupamentos feita sobre três conjuntos de dados com o algoritmo CLARANS e com o DBSCAN, mostrando a capacidade de discriminação do DBSCAN.



(a) CLARANS (b) DBSCan

Figura 2.5: Agrupamentos detectados pelos algoritmos CLARANS e DBSCan (Ester et al., 1996)

Os algoritmos de detecção de agrupamentos apresentados requerem sempre que o número de agrupamentos k a serem encontrados seja fornecido como parâmetro. O DBScan requer apenas dois parâmetros: ϵ - a distancia máxima em que a vizinhança de um elemento deve ser encontrada; e o minPts - o número mínimo de elementos necessários para a formação de um agrupamento. O algoritmo começa com um elemento qualquer do conjunto que não foi visitado. A sua vizinhança ϵ é recuperada e se ela contém pelo menos *minPts* elementos, um agrupamento é formado. Caso contrário, o elemento é rotulado como ruído. Note que este elemento pode ser encontrado como vizinhança de um outro elemento futuramente e assim ser parte de um agrupamento.

Se um elemento é encontrado para ser parte de um agrupamento, sua vizinhança ϵ é também parte do agrupamento. Consequentemente, todos os elementos que forem encontrados na vizinhança ϵ são adicionados, bem como a vizinhança de cada elemento. Este processo continua até um agrupamento ser completamente encontrado. Então, um novo elemento ainda não visitado é processado, levando a descoberta de um novo agrupamento ou ruído.

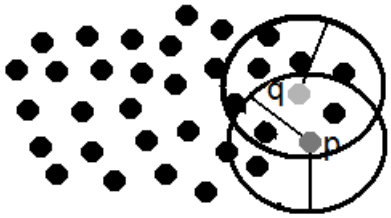


Figura 2.6: Exemplificação do processo realizado pelo algoritmo DBSCan

A Figura 2.6 apresenta o processo feito pelo DBSCan explicado acima. Primeiramente o elemento p é visitado e sua vizinhança encontrada. Em seguida, a vizinhança de cada elemento da vizinhança de p é recuperada e incluída no agrupamento. Na Figura 2.6 é apresentado somente a recuperação da vizinhança do elemento q pertencente a vizinhança do elemento p.

O algoritmo de detecção de agrupamentos usado para avaliar a amostragem balanceada desenvolvida nesta tese foi o DBScan. Dentre as razões da sua escolha estão: o número de agrupamentos a ser detectado não precisar ser definido pelo usuário, ele pode trabalhar com agrupamentos com formatos arbitrários, tamanhos e densidades distintas, ele não é suscetível a ruído e admite uma grande quantidade de atributos.

2.5 Considerações finais

Neste Capítulo foram apresentados alguns conceitos sobre o processo de descoberta de conhecimento com enfoque na mineração tradicional dos dados com especial atenção na redução de elementos no pré-processamento dos dados. No Capítulo 3 são apresentadas as técnicas para a descoberta de conhecimento em dados modelados como grafos, juntamente com a principais técnicas de mineração de redes complexas e suas propriedades estatísticas, além dos conceitos vindos da teoria dos grafos.

Como foi apresentado neste Capítulo, as técnicas de pré-processamento são importantes para o sucesso e eficiência na análise dos dados. A amostragem de dados tem destaque no pré processamento, por ser um método rápido, que agiliza o processo de mineração de dados de maneira eficiente. Uma estratégia comumente usada para melhorar

a eficiência dos algoritmos de detecção de agrupamentos é a redução do número de elementos processados por meio da realização de uma amostragem (*sampling*) do conjunto de dados.

A utilização de técnicas de amostragem mostrou-se especialmente útil para os métodos que realizam várias iterações considerando diferentes inicializações, como o algoritmo CLARANS que, dentre os algoritmos de agrupamento baseados no método k-medoid é o algoritmo que apresenta o melhor custo benefício entre tempo de processamento e qualidade de agrupamento, embora a sua complexidade computacional ainda seja de $O(N^2)$ no número de instâncias no conjunto de dados.

Os conjuntos de dados reais são tradicionalmente caracterizados pela não uniformidade, alta quantidade de atributos e presença de ruído. A amostragem balanceada permite a redução do conjunto de dados em número de elementos e a diminuição da quantidade de ruído, fazendo com que a mineração do conjunto seja mais rápida e com melhores resultados.

No Capitulo 4 serão apresentados os resultados alcançados pelo algoritmo BBS, desenvolvido neste trabalho para a realização da amostragem balanceada, juntamente com os algoritmos GBS, DBS e a amostragem uniforme. Como será visto, o algoritmo BBS tem um desempenho superior.

Como será visto no Capítulo 3 e 5 a redução também pode ser aplicada em grandes redes complexas, modeladas como grafos, para extração eficiente de propriedades estatísticas e assim agilizar o processo de mineração de redes complexas.

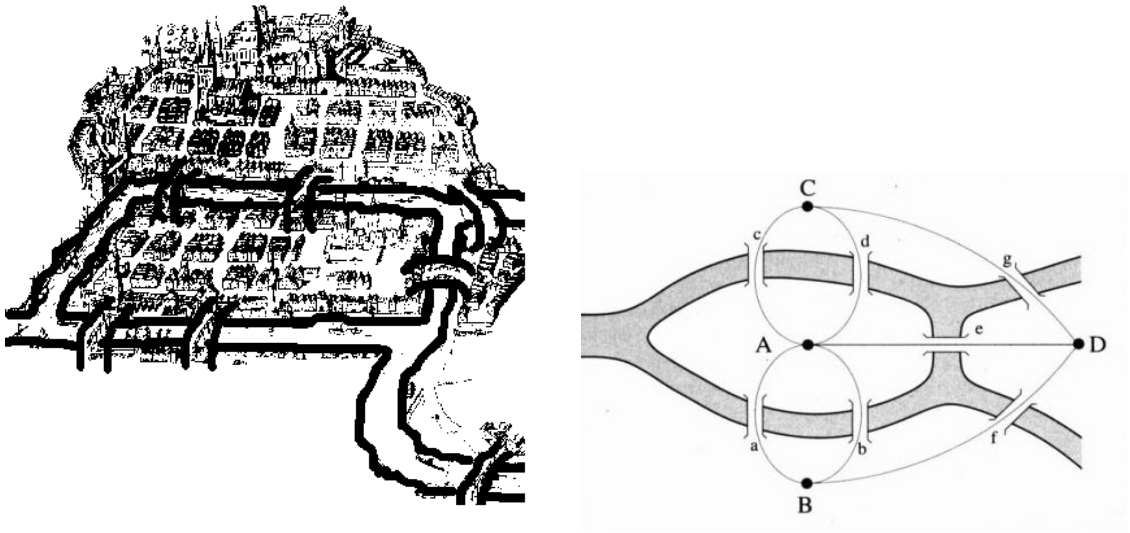
Capítulo 3

Redes Complexas: Conceitos, história e aplicações

3.1 Considerações Iniciais

O estudo de redes complexas tem uma longa história entre os matemáticos e os cientistas. Em 1736, o matemático Leonard Euler interessou-se pelo problema chamado "As pontes de Königsberg". A cidade de Königsberg fica nas margens do rio Pregel na Prússia, atualmente a cidade de Kaliningrado na Rússia, e é constituída de duas ilhas. A cidade é conectada às ilhas por sete pontes, como mostrado na Figura 3.1 (a). Um morador da cidade fez a seguinte proposição: "Há algum caminho que cruze as sete pontes apenas uma única vez?". A lenda diz que as pessoas da cidade de Königsberg gastaram muitas horas tentando encontrar uma solução antes que o matemático Euler provasse que não havia nenhum caminho possível (Newman et al., 2006).

A prova que hoje parece trivial, mas que não era tão óbvia em 1736, transforma o mapa em um grafo - um objeto matemático consistindo de pontos, que são chamados vértices ou nós, e as ligações entre esses pontos, chamadas de arestas. O objetivo de



(a) (b)

Figura 3.1: (a) Mapa de Königsberg (b) Grafo representando o problema das pontes de Königsberg (Newman et al., 2006).

um grafo é abstrair os detalhes do problema original exceto a conectividade. No grafo de Königsberg há quatro vértices representando os quatro pontos de terra e sete arestas representando as pontes que ligam as porções de terra, conforme mostrado na Figura 3.1 (b). Assim, o problema das pontes pode ser refraseado para uma linguagem matemática e a pergunta modificada para saber se há um caminho Euleriano no grafo em questão (Newman et al., 2006).

Um caminho Euleriano é definido como sendo um caminho que usa cada aresta apenas uma única vez. Euler provou que, para que o caminho exista, é necessário que ele contenha uma aresta para "entrar" e outra para "sair" de cada vértice, com exceção do primeiro e último vértice. Assim, para um grafo ser Euleriano, todos os nós devem ter número de arestas pares (entrando e saindo) ou exatamente dois nós com número de arestas ímpares no grafo. O grafo que representa o problema das pontes contém todos os vértices com grau ímpar, por isso não há solução para o problema apresentado. O problema da existência de um caminho Euleriano, bem como um caminho Hamiltoniano - na qual os vértices são visitados uma única vez - ainda são de grande interesse dos matemáticos (Newman et al., 2006).

Muitos consideram a prova de Euler como sendo o primeiro teorema no campo, hoje bem desenvolvido, da matemática discreta conhecido como teoria dos grafos. Passados

três séculos, a teoria dos grafos se tornou a principal linguagem matemática para descrever as propriedades das redes complexas.

As redes complexas, ou simplesmente redes, são um conjunto de elementos discretos que são representados pelos vértices e arestas, que são um conjunto de conexões entre os vértices. Os elementos e suas conexões podem representar, por exemplo, pessoas e ligações de amizade, computadores e linhas de comunicação (Faloutsos et al., 1999), componentes químicos e reações (Jeong et al., 2000), artigos e citações (Redner, 1998), entre outros. Um exemplo gráfico de uma rede biológica é apresentado na Figura 3.2.

Os grafos são capazes de abstrair os detalhes do problema ao descreverem características topológicas importantes com uma clareza que seria impossível se todos os detalhes fossem mantidos. Essa foi uma das razões por que a teoria dos grafos se espalhou, especialmente nos últimos anos, e tem sido utilizada por engenheiros, cientistas da computação e em especial por sociólogos.

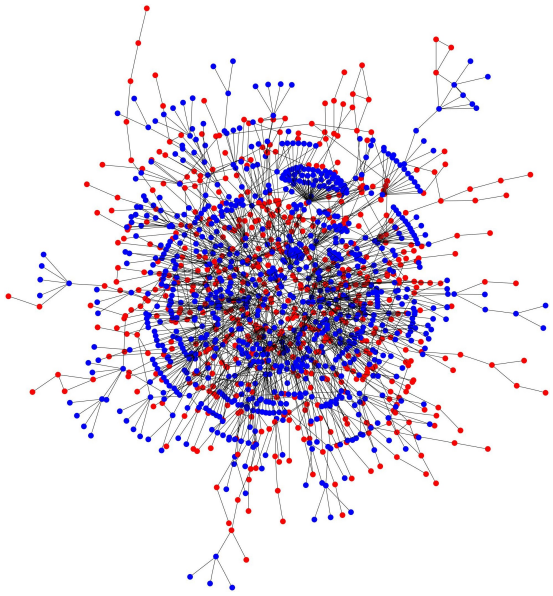


Figura 3.2: Rede Biológica representada graficamente.

Entretanto, o tamanho dos grafos analisados, principalmente por sociólogos e cientista da computação, cresce rapidamente, sendo a Web a principal fonte para este tipo de dado. As redes sociais se destacam pelo seu tamanho e aumento constante. Um exemplo é o número de usuários ao final de 2008 nas redes FaceBook (`www.facebook.com`) com cerca de 175 milhões de usuários, Orkut (`www.orkut.com`) com 47 milhões de usuários e

Flickr (`www.flickr.com`) com 7 milhões de usuários.

Com isso, os questionamentos feitos por pesquisadores mudaram também, principalmente na escala. Um exemplo é a tradicional pergunta "Qual nó deve ser removido para o grafo se partir em dois?" que foi adaptada para "Qual o conjunto mínimo de nós que deve ser removido para o grafo se partir em dois?".

Esta mudança de escala fez com que novos algoritmos e propriedades, específicas para redes complexas, mas que usam a teoria dos grafos como base, fossem desenvolvidas. Isso se deve ao fato de que muitos algoritmos tradicionais da teoria dos grafos são exaustivos, isto é, testam todas as possibilidades no grafo até encontrar uma resposta ou esgotar as possibilidades. Essa solução é, na prática, inexequível para grafos reais com centenas de milhares de nós e arestas ou mais.

Neste Capítulo são apresentados os conceitos básicos da teoria dos grafos, a terminologia e notação utilizadas nesta tese. Também será apresentado um resumo da área de mineração de grafos, com as principais tarefas e algoritmos existentes e que são base para o entendimento dos trabalhos desenvolvidos nesta tese.

3.2 Conceitos Básicos e Definições

A seguir são definidos brevemente os conceitos e a terminologia que são utilizados no decorrer dessa tese para a área de redes complexas. Primeiramente é apresentada a teoria básica de grafos e, em seguida, são apresentadas as principais propriedades e modelos que hoje são importantes para a análise das redes complexas.

3.2.1 Teoria dos Grafos: conceitos

Um grafo $\mathcal{G} = (\mathcal{V},\mathcal{E})$ é definido como um conjunto de vértices $\mathcal{V}$ e um conjunto de arestas $\mathcal{E}$, sendo que $|\mathcal{V}| = N$ denota o número de nós e $|\mathcal{E}| = M$ denota o número de arestas, sendo $e_k \in \mathcal{E}$ e $e_k = \{(v_i, v_j)|v_i, v_j \in \mathcal{V}\}$. Os termos nó ou vértice são considerados sinônimos. Nesta tese será usado o termo nó para referenciar os elementos do conjunto de vértices $\mathcal{V}$ e similarmente o termo aresta para referenciar os elementos do conjunto de arestas $\mathcal{E}$, que

também tem na literatura os seguintes sinônimos: *links*, *hops*, ligações ou conexões.

Uma maneira conveniente de representar um grafo $\mathcal{G}$ em um computador é usar uma matriz de adjacência, que é uma matriz $\mathbf{A}$ quadrada $N \times N$, sendo $N = |\mathcal{V}|$, em que $\mathbf{A}_{i,j} = 1$ se $(v_i, v_j) \in \mathcal{E}$ e 0 caso o contrário. Neste Capítulo são apresentadas algumas propriedades importantes dos grafos, baseadas diretamente na sua matriz de adjacência.

Apresentam-se agora alguns conceitos básicos que serão usados nesta tese:

- **Grafos Direcionados e Não Direcionados:** um grafo é *não direcionado* se $\{(v_i, v_j) \in \mathcal{E} \Leftrightarrow (v_j, v_i) \in \mathcal{E}\}$, isto é, as aresta são pares de nós sem ordem. Se um par de nós é ordenado, isto é, arestas tem direção, então o grafo é *direcionado*, também chamado de *dígrafo*.

- **Vizinhança de um Nó:** a vizinhança ou adjacência $\mathcal{N}(v_i)$ de um nó v_i é $\mathcal{N}(v_i) = \{v_j | (v_i, v_j) \in \mathcal{E}\}$.

- **Grau do Nó:** o nó v_i tem grau $d(v_i)$ se ele tem $|\mathcal{N}(v_i)|$ nós incidentes. Para grafos direcionados, o grau de um nó pode ser dividido em "grau de saída", $d_{out}(v_i)$ que é o número de arestas entram pelo nó v_i e "grau de entrada", $d_{in}(v_i)$ que é o número de arestas que saem para o nó v_i.

- **Triângulo:** em um grafo não direcionado um triângulo, também conhecido como fechamento transitivo, é uma tripla de nós conexos (u, v, w), tal que, $(u, v), (v, w), (w, u) \in \mathcal{E}$

- **Caminho:** é uma sequência de nós conectados entre si, $P(v_1, v_N) = (v_1, v_2, v_3, ..., v_n)$, tal que, entre cada par de vértice existe uma aresta $(v_1, v_2), (v_2, v_3), ..., (v_{n-1}, v_n) \in \mathcal{E}$. Um caminho é **simples** se nenhum vértice se repete. Dois caminhos são **independentes** se somente o primeiro e o último vértice são comuns à eles.

- **Comprimento de um caminho:** é o número de arestas que o caminho contém. O **menor caminho** entre dois nós $P(v_i, v_j)$ é o caminho de menor número de arestas que ligam os dois nós.

- **Ciclo ou Circuito:** é um caminho que começa e acaba no mesmo vértice - $P(v_i, v_i)$. Ciclos de comprimento 1 são chamados laços. Um ciclo simples é um ciclo que tem comprimento de, no mínimo, 3 e no qual o vértice inicial só aparece de novo como vértice final, e os outros nós aparecem somente uma vez.

- **Grafo Acíclico:** é um grafo que não possui ciclos.

- **Subgrafo:** um subgrafo $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$ de um grafo $G = (\mathcal{V}, \mathcal{E})$ é um subconjunto de arestas e todos os nós tal que $\mathcal{E}_s \subseteq \mathcal{E} \Rightarrow \mathcal{V}_s = \{v_i, v_j | (v_i, v_j) \in \mathcal{E}_s\}$.

- **Grafo Conexo:** é um grafo que possui pelo menos um caminho entre todos os pares de nós.

- **Componente Conexa:** é o maior subgrafo, na qual existe um caminho entre qualquer par de aresta.

- **Árvore:** é um grafo conexo e acíclico. Nesse tipo de grafo, os nós de grau 1 são chamado nós folhas. Uma **floresta** é um grafo em que todas as suas componentes conexas são árvores.

- **Caminho euleriano:** é o caminho que usa cada aresta exatamente uma vez passando por todas as arestas. Um caminho euleriano é chamado ciclo euleriano se o primeiro e o ultimo vértice são os mesmos.

- **Caminho hamiltoniano:** é o caminho que visita cada vértice uma só vez passando por todos os nós. Um caminho hamiltoniano é dito ciclo hamiltoniano se o primeiro e o último vértice são os mesmos.

- **Grafos Bipartidos:** um grafo $\mathcal{G}$ é bipartido se o conjunto de nós pode ser particionado em dois conjuntos distintos e sem intersecção, $\mathcal{V}_1 \cap \mathcal{V}_2 = \emptyset$, e tal que somente há arestas $\mathcal{E} = \{(v_i, v_j) | v_i \in \mathcal{V}_1, v_j \in \mathcal{V}_2\}$ entre nós do conjunto $\mathcal{V}_1$ e $\mathcal{V}_2$, isto é, não há arestas conectando nós do mesmo conjunto de nós.

- **Grafo Biconexo ou 2-conexo:** um grafo é biconexo ou 2-conexo se, removendo uma única aresta, o grafo ainda continua conexo. Isto significa que, entre quaisquer

pares de nós, há pelo menos dois caminhos independentes.

- **Grafo Induzido:** um subgrafo induzido $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$ de um grafo $G = (\mathcal{V}, \mathcal{E})$ é um subconjunto de nós e todas as arestas que ligam este subconjunto de nós no grafo original $\mathcal{G}$, tal que $\mathcal{V}_s \subseteq \mathcal{V}$ e $\mathcal{E}_s = \{(v_i, v_j) | (v_i, v_j) \in \mathcal{E}_s \, v_i, v_j \in \mathcal{V}_s\}$.

- **Grafo Completo:** é um grafo em que todo par de nó é ligado por uma aresta.

- **Clique (κ_{N_s}):** é um subgrafo completo que possui um subconjunto de nós $\mathcal{V}_s \subseteq \mathcal{V}$ e arestas conectando todos os pares de nós em $\mathcal{V}_s$. O tamanho N_s do clique é definido pelo número de nós, $|\mathcal{V}_s| = N_s$. Um triângulo é um clique de tamanho 3 - κ_3.

- **Diâmetro:** o diâmetro $\mathcal{D}$ de um grafo $\mathcal{G}$ é o maior caminho dentre todos os menores caminhos existentes entre todos os pares de nós do grafo $\mathcal{G}$.

3.2.2 Leis de Potência

Uma distribuição que segue uma lei de potência é uma distribuição na forma:

$$p(x) = a * x^{-\gamma} \tag{3.1}$$

na qual $p(x)$ é a probabilidade de x ocorrer, sendo a uma constante de proporcionalidade e γ é o expoente da lei de potência (Clauset et al., 2009; Newman, 2005).

Distribuições que seguem uma lei de potência ocorrem em muitas situações de interesse científico e são importantes para o entendimento de fenômenos naturais e humanos. A população das cidades e as intensidades dos terremotos são exemplos de fenômenos que têm a distribuição seguindo uma lei de potência. Em redes complexas, é comum que as seguintes distribuições tendam a seguir as leis de potência: Grau dos nós (Chakrabarti e Faloutsos, 2006); Número de triângulos em relação ao grau dos nós (Tsourakakis, 2008); Autovalores da matriz de adjacência (Faloutsos et al., 1999); Crescimento do número de nós e arestas na evolução das redes (Leskovec et al., 2007a) entre outras.

Leis de potência são muitas vezes chamadas de distribuições livre de escala, que intuitivamente significa que uma distribuição se parece com ela mesma independente da

escala em que se esta olhando. A noção de auto similaridade é implícita no nome "livre de escala". A auto similaridade de elementos consiste no fato do elemento manter as mesmas propriedades seja qual for a escala utilizada (Schroeder, 1991). Diversos pesquisadores tem argumentado que especificamente grafos Web (Crovella e Bestavros, 1997; Dill et al., 2002; Dorogovtsev et al., 2002) e redes biológicas (Barabási et al., 2002) tendem a ser auto-similares.

A propriedade de ser livre de escala significa que aumentando a escala ou unidade pela qual x é medido por um fator b, o formato da distribuição $p(x)$ não é mudada exceto pela multiplicação por uma constante. Isto significa que, não importa qual porção da distribuição é olhada, o ângulo da curva de qualquer parte do gráfico log-log é o mesmo. A família de distribuições exponenciais, tal como a distribuição Gaussiana, não são livres de escala. Na verdade, as leis de potência são as únicas distribuições livre de escala (Newman, 2005).

3.3 Descoberta de Conhecimento em Grafos

O processo de descoberta de conhecimento em grafos, também conhecido como descoberta de padrões em grafos, é semelhante ao processo de descoberta de conhecimento em bases de dados (KDD) que foi apresentado no Capítulo 2 quanto aos passos a ser seguido. Entretanto, as técnicas utilizadas em cada um desses passos são diferentes, já que os grafos são dados estruturados e, por isso, os algoritmos de KDD não podem ser aplicados diretamente.

Assim, o processo de descoberta de conhecimento em grafos também é composto pelas fases de pré-processamento, mineração e avaliação dos resultados. As etapas de pré-processamento e mineração são detalhadas a seguir e as principais técnicas de cada uma delas são apresentadas em conjunto. A avaliação dos resultados, que é a análise dos resultados obtidos da mineração, e a visualização dos resultados não são detalhados nesta tese por não serem o foco deste trabalho. A parte de avaliação é tratada apenas no Capítulo 5 com a apresentação dos resultados obtidos dos algoritmos de mineração de grafos desenvolvidos.

3.3.1 O pré-processamento de grafos

Como apresentado no Capítulo 2, as técnicas de pré-processamento são a chave para o sucesso do processo de descoberta de conhecimento, seja ele aplicado em grafos ou em dados tradicionais.

Na descoberta de conhecimento de grafos, as tarefas do pré-processamento possuem os mesmos objetivos que na descoberta de conhecimento tradicional. Contudo, as técnicas que implementam as tarefas são diferentes, já que é necessário levar em consideração as arestas existentes entre os nós. As técnicas de limpeza, integração e transformação são as mais utilizadas, já que os dados coletados podem vir de diversas fontes e, por isso, devem ser integrados e devem estar no formato adequado para as tarefas de mineração.

As técnicas de redução de dados, assim como no KDD, também são importantes, pois, com a utilização cada vez maior da Web para compras, sites de relacionamento, busca, entre outras atividades, o volume de dados representados como grafos tem crescido exponencialmente.

Grandes redes complexas reais, com milhões de nós e arestas, são atualmente uma importante fonte de dados. Contudo, esse volume pode se tornar inútil se não houver capacidade de processamento disponível. Como nas tarefas de mineração de dados tradicionais, a mineração de dados representados como grafos também exige uma iteratividade no processo de mineração. Dessa maneira, é necessário o uso de redes complexas com um tamanho que permita a execução de uma tarefa de mineração em um tempo razoável. Entretanto, a existência de um relacionamento (arestas) entre os elementos (nós) nesse tipo de dado faz com que as tarefas de redução se tornem complexas.

Amostragem de redes complexas

A amostragem da rede complexa pode ser definida como a seleção de um conjunto de nós e arestas do grafo original $\mathcal{G}$ para compor o subgrafo amostra $\mathcal{G}_s$ que será usado nas tarefas de mineração de dados. Este tipo de pré-processamento pode ser útil para ajudar a agilizar algoritmos de mineração de grafos, bem como para visualizar as redes, pois, para grandes redes torna-se impossível encontrar uma representação visual adequada.

Há poucos métodos disponíveis para a amostragem em redes complexas e, na sua grande maioria, a amostragem é feita focando apenas em garantir que uma ou mais propriedades da rede sejam mantidas.

Há três tipos de abordagem para a realização de uma amostragem em redes complexas: remoção, contração ou exploração. Cada uma delas é apresentada a seguir:

1. **Remoção:** os métodos baseados nesta abordagem removem nós ou arestas da rede complexa até o tamanho desejado ser alcançado. Exemplos da remoção de nós e arestas aleatoriamente são os métodos:

   - DOULIN (Tsourakakis et al., 2009), que tem por objetivo a remoção de uma pequena porção de arestas para aumentar a velocidade de algoritmos que fazem a contagem de triângulos. Neste método cada aresta é removida com probabilidade p, sendo $0 < p < 1$. Cada triângulo encontrado na rede resultante conta como $\frac{1}{p^3}$ triângulos.
   - `ShatterPlots` (Appel et al., 2009) é um dos trabalhos desta tese que será detalhado no Capítulo 5. Este algoritmo extrai propriedades das redes complexas, após a remoção de uma certa quantidade de arestas, que permite classificá-las em reais ou sintéticas.

2. **Contração:** os métodos contrativos agrupam nós adjacentes formando um único nó. Esse processo é aplicado de maneira recursiva até a rede alcançar o tamanho desejado. Exemplos dessa abordagem são os métodos "CRE" (*Contraction of random edge*) e "CRVE" (*Contraction of random vertex/edge*) (Krishnamurthy et al., 2007);

3. **Exploração:** os métodos exploratórios percorrem a rede complexa visitando alguns nós até formarem um caminho do tamanho desejado. As arestas e nós visitados compõe o subgrafo amostrado. Exemplos desta abordagem são as amostragens "*Forest Fire*" e "*Random Node Neighbor*" (Leskovec e Faloutsos, 2006), que têm por objetivo manter mais de uma propriedade estatística, como a média do grau do nó, a proporcionalidade do tamanho da componente conexa, etc. Em (Krishnamurthy

et al., 2007) são apresentados dois modelos baseados em exploração: o "EBFS" (*Exploration by breadth-first search*) e "EDFS"(*Exploration by depth-first search*), que criam uma rede menor percorrendo o grafo por meio de uma busca em largura ou em profundidade, respectivamente.

3.3.2 A mineração de grafos

Como apresentado no Capítulo 2, a mineração de dados pode ter variações segundo o tipo de dado que se quer minerar. À mineração de grafos é um tipo especial da mineração de dados, que tem por objetivo minerar dados que possuam elementos (nós) e ligações entre esses elementos (arestas), por isso, os algoritmos tradicionais, apresentados no Capítulo 2, não se aplicam à este novo tipo de dado.

Primeiramente, a mineração de grafos pode ser dividida em duas grandes categorias. Na primeira, a mineração é realizada em uma base composta por diversos grafos, sendo a principal tarefa a mineração de subgrafos frequentes. Nesta tarefa, um grafo é submetido como consulta à uma base composta por diversos grafos. A resposta é composta por todos os grafos que possuam o grafo consultado como subgrafo. Esta tarefa é bastante aplicada à grafos que representam componentes químicos (Chakravarthy e Pradhan, 2008; Kuramochi e Karypis, 2004; Yan e Han, 2002). A segunda tarefa é a descoberta de padrões, geralmente propriedades estatísticas e leis de potência, em um ou mais grafos, usualmente com centenas de milhares de nós. Esta tarefa é a que será abordada nos trabalhos desenvolvidos nessa tese. Assim, somente as técnicas desta tarefa são detalhadas nas próximas seções e o termo mineração de grafos será usado para descrever somente tarefas desta segunda categoria.

Dentre as tarefas mais exploradas na mineração de grafos destacam-se a descoberta de propriedades estatísticas das redes complexas, a detecção de comunidades ou grupos, a predição de ligações e a propagação de informação juntamente com a resistência a ataques. A seguir cada uma destas tarefas são detalhadas.

3.3.3 Propriedades estatísticas das redes complexas

Diversos domínio de aplicações têm seus dados modelados como redes complexas, por exemplo, a Internet, o World Wide Web (WWW), as redes sociais, de colaboração, biológicas, entre outras. Os pesquisadores nos últimos anos têm identificado classes de propriedades que podem ser encontradas em muitas das redes reais de vários domínios, sendo que muitas dessas distribuições seguem leis de potência, como a distribuição do grau dos nós, número de triângulos e os autovalores da matriz de adjacência da rede complexa.

Diâmetro e raio efetivo

O diâmetro $\mathcal{D}$, como definido anteriormente, é o maior caminho dentre todos os menores caminhos existentes entre todos os pares de nós do grafo $\mathcal{G}$. O diâmetro também é referenciado como *diâmetro completo*. Para grafos com mais de uma componente conexa, o diâmetro usualmente é definido como infinito. Além disso, o diâmetro é suscetível aos efeitos degenerativos da estrutura do grafo como, o surgimento de um caminho muito longo no grafo durante a sua evolução.

Calcular o diâmetro de um grafo grande é computacionalmente caro(complexidade de tempo $O(N^3)$). Uma maneira mais eficiente de realizar este cálculo é pela amostragem de nós, isto é, uma quantidade de nós é amostrada e então o diâmetro é calculado entre os pares de nós amostrados (Albert et al., 1999). Uma outra abordagem é usar o algoritmo de aproximação ANF (Palmer et al., 2002) que é baseado na cálculo aproximado chamado diâmetro efetivo. Define-se o diâmetro efetivo como sendo o menor número de "arestas" em que no mínimo 90% de todos os nós da maior componente conexa podem ser alcançados entre si.

O diâmetro efetivo é um valor mais robusto que o diâmetro, pois, nele somente os pares de nós conexos são considerados e a direção das arestas (no caso de grafos direcionados) são ignoradas. Além disso, muitos experimentos mostram que o diâmetro efetivo e o diâmetro exibem comportamento qualitativamente similar.

Em detalhes, seja $g_h(\mathcal{G})$ uma função que calcula para cada nó v_i o número de nós que tenham um caminho de máximo h arestas de distância partindo de cada nó v_i, sendo

$i = 1, ..., N$. O valor de h inicia-se em h=1 até que $g_{(h-1)}(\mathcal{G})$-$g_h(\mathcal{G})$<*threshold*. Para h=1 tem-se $g_h(\mathcal{G})$= $\sum_{i=1}^{N} |d(v_i)|$ =$|\mathcal{E}|$. O gráfico que representa a função $g_h(\mathcal{G})$ é chamado "Hop-Plot" e é apresentado na Figura 3.3 (Leskovec et al., 2005). A função $g_h(\mathcal{G})$ foi aplicada a rede Epinions (rede social que representa confiança interpessoal - quem confia em quem). O diâmetro efetivo da rede Epinions é igual a $\mathcal{D}_e = 5$ sendo representado pela linha tracejada no gráfico apresentado na Figura 3.3.

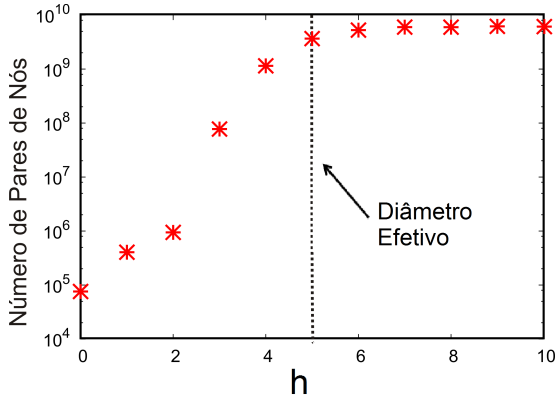


Figura 3.3: Gráfico Hop-Plot da rede Epinions. A linha tracejada em 5 representa o Diâmetro efetivo da rede.

Definindo mais formalmente o diâmetro efetivo tem-se: o diâmetro efetivo de $\mathcal{G}$ é definido como sendo $\mathcal{D}_e = h$ sendo que ghg representa 90% do número de pares de nós alcançáveis.

Muitos dos grafos reais exibem um diâmetro relativamente pequeno, conhecido como o fenômeno "Small-World". Por exemplo, o diâmetro efetivo é pequeno para grandes redes reais, tais como Internet, Web, redes sociais (Barabasi e Albert, 1999; Bollobás e Riordan, 2004; Broder et al., 2000; Chung et al., 2002; Milgram, 1967; Reka e Barabási, 2002; Watts e Strogatz, 1998).

Distribuição do grau

A distribuição do grau dos nós de uma rede é uma lei de potência se o número de nós N_ϕ que possui um grau ϕ é dado por $N_\phi \propto \phi^{-y}$ $(y > 1)$, sendo $\mathcal{V}_\phi = \{v_i \in \mathcal{V} | d(v_i) = \phi\}$, $|\mathcal{V}_\phi| = N_\phi$ e y é chamado de expoente da distribuição do grau. A grande maioria das redes reais apresentam uma distribuição do grau dos nós que segue uma lei de potência, por

isso são chamadas redes livres de escala. Um exemplo pode ser visto na Figura 3.5 (b), que apresenta a distribuição do grau da rede complexa Epinions com expoente $y = 1,59$.

Este tipo de distribuição tem sido encontrada em grafos de ligações telefônicas (Abello et al., 1998), Internet (Faloutsos et al., 1999), na Web (Broder et al., 2000; Flaxman et al., 2005; Huberman e Adamic, 1999; Kleinberg et al., 1999; Kumar et al., 1999), grafos de citações (Redner, 1998), *click-stream* (Bi et al., 2001), redes sociais online (Chakrabarti et al., 2004) e muitas outras.

Tipicamente, para muitos conjuntos de dados, o expoente da distribuição do grau tem valor $2 < y < 3$. Por exemplo, a distribuição do grau de entrada da Web é $y_{in} = 2,1$ e de saída $y_{out} = 2,4$ (Reka e Barabási, 2002), enquanto para as redes de computadores chamadas de Sistemas Autônomos $y = 2,4$ (Faloutsos et al., 1999). Contudo, alguns desvios do padrão da lei de potência foram notados em (Pennock et al., 2002).

As redes reais possuem muitos nós com grau que excedem exageradamente a média do grau dos nós, isto é um indício de uma distribuição que segue uma lei de potência. Esta descoberta foi importante para o estudo das redes reais (Faloutsos et al., 1999), pois permitiu fazer a distinção entre as redes reais e as redes *Erdös-Rényi* que são apresentadas em detalhes na Seção 3.4. Os nós com altíssimo grau são frequentemente chamados "*hubs*", e usualmente têm um propósito específico nas redes às quais pertencem, embora isso dependa muito do domínio de aplicação da rede.

Triângulos e coeficiente de clusterização

Em muitas redes é notado que se um nó u é conectado com um nó v que é conectado com w, então há um grande probabilidade de u ser conectado com w. Esta relação é chamada de transitividade e é medida pelo coeficiente de clusterização (Watts e Strogatz, 1998), aplicado especialmente em redes sociais. A transitividade significa a presença de um alto número de triângulos (κ_3) na rede. Exemplos de um grafo com triângulos é apresentado na Figura 3.4. A contagem de triângulos é a principal parte do coeficiente de clusterização, que pode ser calculado para cada nó do grafo (Equação 3.2) ou para o grafo como um todo (Equação 3.3). Este coeficiente tem o objetivo de indicar quão próximo o grafo esta

de ser um grafo completo.

O coeficiente de clusterização $C(v_i)$ de um vértice v_i de grau $d(v_i)$ é definido pela Equação 3.2 a seguir.

$$C(v_i) = \frac{2 * \kappa_3(v_i)}{d(v_i) * (d(v_i) - 1)} \tag{3.2}$$

Seja v_i um nó com $|d(v_i)|$ vizinhos; então no máximo $d(v_i) * (d(v_i) - 1)/2$ arestas podem existir entre eles, sendo $\kappa_3(v_i)$ a fração de arestas que realmente existe. Isto significa que, o coeficiente de clusterização $C(v_i)$ de um vértice v_i é a proporção de arestas entre os nós da sua vizinhança dividido pelo número de arestas que podem existir entre eles. Equivalentemente, $C(v_i)$ é a fração de triângulos centrados no nó v_i entre $(d(v_i) * (d(v_i) - 1))/2$ triângulos que possam existir.

O coeficiente de clusterização global $C(\mathcal{G})$ é a média da soma de todos os $C(v_i)$ dos nós do grafo $\mathcal{G}$, dividido pelo número total de nós N. A equação do coeficiente de clusterização global é apresentada na Equação 3.3 a seguir.

$$C(\mathcal{G}) = \frac{1}{N} * \sum_{i=1}^{N} C(v_i) \tag{3.3}$$

Encontrar a quantidade de triângulos que cada nó possui, bem como a quantidade total de triângulos no grafo é um processo computacionalmente caro. A sua complexidade é de $O(N^2)$, sendo N o número total de nós do grafo.

Para reduzir essa complexidade, alguns trabalhos como em (Latapy, 2008) os autores propõem algumas otimizações para a contagem e listagem dos triângulos em um grafo. Outro trabalhos como em (Bar-Yossef et al., 2002; Becchetti et al., 2008; Tsourakakis, 2008; Tsourakakis et al., 2009) contam triângulos sem identificá-los. Isto é feito utilizando-se aproximações como amostragem do grafo, multiplicação de matrizes ou multiplicação dos autovalores da matriz de adjacência do grafo. Entretanto, com este tipo de algoritmo não é possível identificar quais são os nós que compõe cada um dos triângulos.

Uma característica interessante é quanto a distribuição dos triângulos. A média da quantidade de triângulos dos nós com um determinado grau, em relação ao grau dos nós,

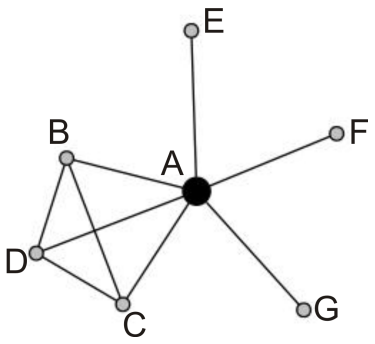


Figura 3.4: Exemplo de um grafo com 3 triângulos (contados uma vez só) centrados no nó A: $P_1(A,A) = \{A,B,D,A\}$, $P_2(A,A) = \{A,D,C,A\}$, $P_3(A,A) = \{A,B,C,A\}$

segue uma lei de potência com o expoente oposto ao da distribuição do grau (Tsourakakis, 2008). A Figura 3.5 (a) apresenta a média da quantidade de triângulos vs. a distribuição do grau da rede Epinions e (b) com a distribuição do grau da rede Epinions. Como pode-se notar os expoente são opostos e muito parecidos.

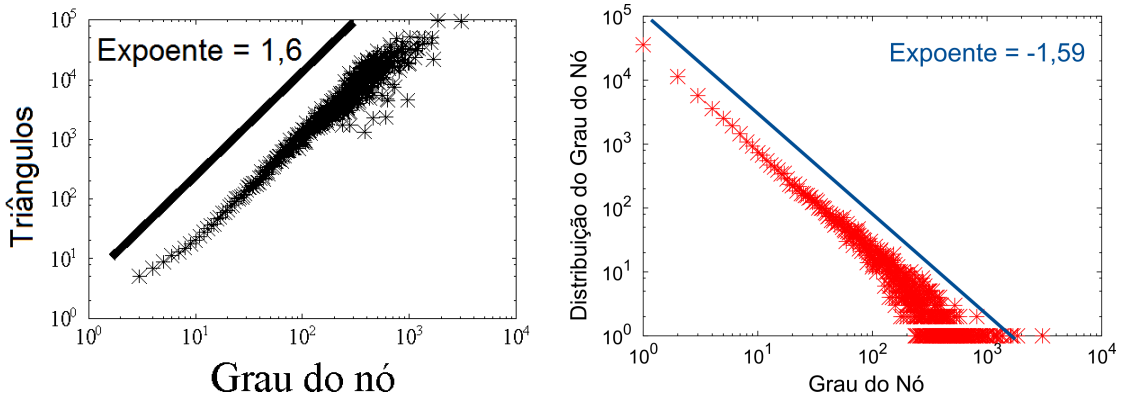


Figura 3.5: O gráfico apresentado em (a) representa a média dos triângulos vs. o grau dos nós e o gráfico (b) apresenta a distribuição do grau dos nós. Note que o expoente da lei de potência seguida por ambos são opostos e muito similares (Tsourakakis, 2008).

Na literatura, também há trabalhos que propõem extensões para o coeficiente de clusterização, como em (Caldarelli et al., 2004; Fronczak et al., 2002), em que é usada a contagem de ciclos de tamanhos maiores na equação do coeficiente de clusterização, já que um triângulo, além de ser um clique, é também um ciclo.

Autovalores para grafos

Dentre os elementos estudados na álgebra linear, há os tipos de dado escalares, que são apenas números, e vetores, nos quais podem ser pensados como arcos que tem magnitude

e direção. Uma transformação $T : X \rightarrow W$ é

$$T(x) = \lambda * x, \tag{3.4}$$

sendo λ o escalar e x o vetor.

Uma das mais importantes ferramentas da álgebra linear são as chamadas "transformações lineares" que são expressas tradicionalmente por uma matriz e um vetor, como apresentado na Equação 3.5.

$$T(x) = A * x, \tag{3.5}$$

sendo A uma matriz e x o vetor.

Igualando-se a Equação 3.4 com a Equação 3.5 tem-se $A * x = \lambda * x$. Se a multiplicação do vetor por um escalar, alterar sua magnitude mas não sua direção, então o vetor é chamado de autovetor e o escalar de autovalor associado.

Autovalores e autovetores são conceitos importantes de matemática, com aplicações práticas em áreas diversificadas como mecânica quântica, processamento de imagens, análise de vibrações, mecânica dos sólidos, estatística, etc.

Definição 1. *Dada uma transformação linear e um vetor não nulo x. x é definido como sendo um autovetor de uma transformação se ele satisfaz a equação $Ax = \lambda x$ para algum escalar λ chamado autovalor.*

Na teoria dos grafos, os autovalores λ_i de um grafo $\mathcal{G}$ são definidos como sendo os autovalores da matriz de adjacência A do grafo $\mathcal{G}$.

Para todas as definições e teoremas que serão apresentados a seguir tem-se que o grafo $\mathcal{G}$ é não direcionado. As provas dos teoremas e definições podem ser encontradas em (Mihail e Papadimitriou, 2002). Nos teoremas apresentados, estão sendo considerados os autovalores e autovetores da matriz de adjacência do grafo (Chung, 1994).

Teorema 1. *Para um grafo $\mathcal{G}$ com $|V| = N$ nós, tem-se que $\lambda_1(\mathcal{G}) \geq \lambda_2(\mathcal{G}) \geq \lambda_3(\mathcal{G}) \geq$ ... $\lambda_N(\mathcal{G})$ são os autovalores da matriz de adjacência do grafo G em ordem decrescente.*

Assim, $\lambda_1(\mathcal{G})$ é o maior autovalor de $\mathcal{G}$, também chamado de principal autovalor.

Teorema 2. *Seja $\mathcal{G}$ uma árvore com $N-1$ nós folhas. Então: $\lambda_1(\mathcal{G}) = \sqrt{N-1}$ e $\lambda_N(\mathcal{G}) = -\sqrt{N-1}$ e $\lambda_i(G) = 0$ sendo $i = 2, 3, ..., N-2$*

O Teorema 2 mostra que a conectividade de um grafo pode ser medida pelo autovalor da sua matriz de adjacência pois, quanto mais ciclos o grafo tiver, menor será o seu autovalor. Se o grafo for semelhante a uma árvore, o seu autovalor tende a ser dominado pelo nó de maior grau sendo $\lambda_1 = \sqrt{d_{max}(v_i)}$.

Os autovetores da matriz de adjacência de um grafo podem ser utilizados para medir a centralidade dos nós. Um exemplo é o algoritmo PageRank do Google (Page et al., 1998), que utiliza o primeiro autovetor da matriz de adjacência modificada de um grafo para fazer o *rank* de cada um dos nós. Sendo que o principal objetivo do PageRank é a ordenação de páginas Web para consultas, os nós do grafo são páginas e as aresta os *links* entre as páginas. Neste algoritmo também há um fator chamado *damping factor* - *df* - que é a probabilidade de um usuário encerrar a consulta e este valor é geralmente $0,85$. A Equação 3.6 representa o cálculo original do PageRank para todos os N nós de uma rede.

$$PR = \frac{1 - df}{N} + df * A * PR \tag{3.6}$$

Neste algoritmo, a matriz de adjacência A é primeiro normalizada, isto é, cada elemento de uma linha da matriz é dividido pelo quantidade de uns existente na linha, sendo o somatório de cada linha igual a um. Na iteração inicial tem-se: $A(i,j) = \frac{1}{d_{out}(v_i)}$ se $\exists(v_i, v_j)$ senão $A(i,j) = 0$. O vetor PR é inicializado todo com 1.

3.3.4 Detecção de Comunidades

Grupos de nós que tendem a ser mais conectados entre si que com o restante da rede são chamados de grupo ou comunidade. Pessoas tendem a formar comunidades, isto é, grupos pequenos no qual todo mundo conhece praticamente todo mundo. Com isto, grupos de nós pertencentes a uma mesma comunidade tendem a ter um número elevado de triângulos, e

consequentemente, tendem a formarem cliques de tamanho maiores que 3. Além disso, os membros das comunidades têm pouco relacionamento com membros fora das comunidades que participam e cada um dos grupos tendem a estar organizados hierarquicamente, isto é comunidades dentro de comunidades.

Uma grande quantidade de algoritmos têm sido desenvolvidos para definir e identificar comunidades em redes sociais e de informações (Girvan e Newman, 2002; Radicchi et al., 2004). Muitas vezes também é assumido que as comunidades obedecem a uma estrutura recursiva, em que grandes comunidades podem futuramente serem divididas em comunidades menores (Clauset et al., 2004; Guimerà et al., 2007).

O problema de identificar comunidades se assemelha à tradicional detecção de agrupamentos (Fortunato, 2010; Getoor e Diehl, 2005b). Entretanto, por se tratar de redes complexas é necessário algoritmos que levem em conta a ligação entre os elementos, não sendo possível o uso dos algoritmos de detecção de agrupamentos tradicionais citados no Capítulo 2.

A ideia do algoritmo de identificação de comunidades é particionar a rede em subgrafos menores por meio da remoção de um número mínimo de arestas. As comunidades encontradas podem ter seus elementos disjuntos, isto é, um nó que participa em um grupo não deve participar em outro, aceitando-se apenas algumas sobreposições (Fortunato, 2010). Esta situação pode ser vista na vida real, por exemplo: as pessoas nem sempre participam de apenas um grupo social. Boas revisões sobre detecção de comunidades podem ser encontradas em (Fortunato, 2010; Leskovec et al., 2008).

A estrutura de comunidades, como exemplificado pela Figura 3.6, foi observada também em diversas redes reais, como (Girvan e Newman, 2002; Newman, 2006; Ravasz et al., 2002). Entretanto, esta forma de organização das redes complexas parece não valer para as redes complexas volumosas, com centenas de milhares de nós, como mostrado em (Leskovec et al., 2008), que demonstra que esta estrutura hierárquica está presente somente nas redes complexas pequenas, isto é redes com poucas centenas de nós.

As grandes redes complexas tendem a não possuírem conjuntos de nós interligados formando comunidades muito bem definidas (Figura 3.6). Na verdade, as grandes redes

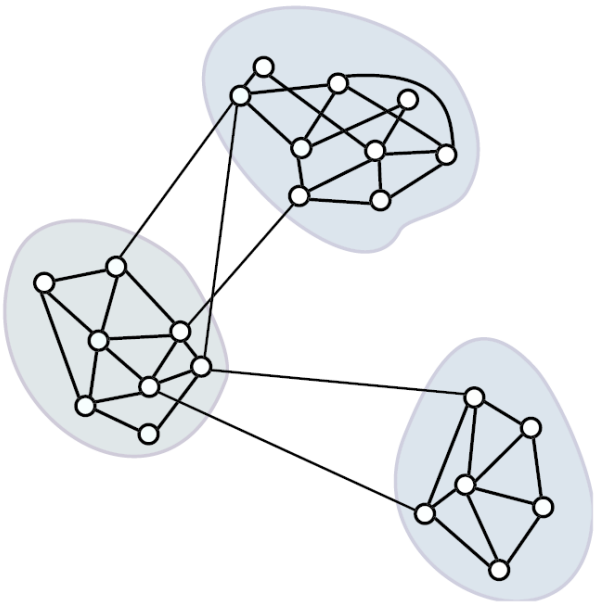


Figura 3.6: Grafo apresentando 3 comunidades do modo tradicional (Leskovec et al., 2008).

tendem a apresentar uma estrutura chamada "Centro-Periferia" (Borgatti e Everett, 2000; Holme, 2005), que em ciência da computação é conhecida também pelo nome "*jellyfish*" (Tauro et al., 2001) ou "*octopus*" (Chung e Lu, 2006) e que é exemplificada na Figura 3.7. Este conceito significa que uma rede é composta por um grande e denso conjunto de nós (core/centro) ligados entre si que basicamente não tem nenhuma estrutura de comunidade hierárquica, isto é, não podem ser quebrados em comunidades menores. Assim, a estrutura Centro-Periferia sugere o oposto da estrutura de comunidade hierárquica, e parece ser o mais encontrado em redes complexas de grande escala (Leskovec et al., 2008) e também em redes de computadores chamados Sistemas Autônomos (Siganos, 2006).

Também em (Leskovec et al., 2008) os autores mostram que as comunidades tendem a ser pequenas, com não mais do que 100 nós, e pouco conectadas com o restante da rede. O valor 100 é conhecido como o número de Dunbar, que é o número máximo de relacionamentos que uma pessoa consegue administrar (Dunbar, 1998).

3.3.5 Resistência a ataques

Uma questão que tem sido considerada como foco de pesquisa recente é a definição de quão robusta é uma rede. Uma rede pode representar o relacionamento interpessoal. Assim,

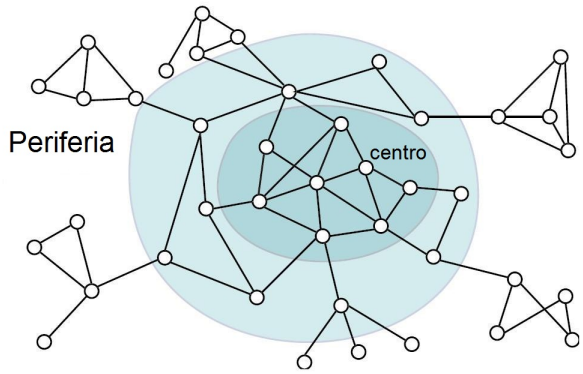
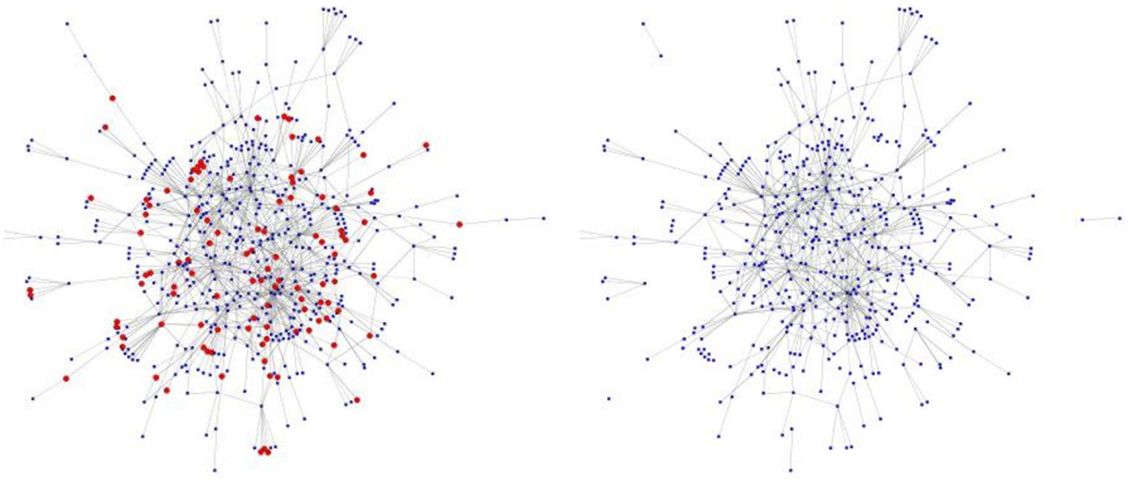


Figura 3.7: Grafo apresentando a topologia "Centro-Periferia"(Leskovec et al., 2008).

as pessoas seriam os nós e o relacionamento as arestas. Imagine por exemplo que os nós da rede sejam marcados ou removidos conforme as pessoas contraiam uma doença ou descubram uma informação. Este tipo de análise é conhecido como resistência a ataques.

A Internet, por exemplo, é altamente robusta, já que há diferentes rotas ligando os computadores e/ou roteadores (nó), fazendo com que a informação tenha mais de um caminho para navegar de um nó para o outro. Mesmo se um roteador falhar, o sistema é capaz de refazer a rota e fazer com que a informação chegue ao seu destino. De fato, na Internet sempre haverá um conjunto de roteadores que não estará funcionando em um dado momento. Entretanto, o importante é que a Internet como um todo continuará funcionando mesmo com uma grande quantidade de falhas (Wu et al., 2007).



(a) (b)

Figura 3.8: Exemplo de uma rede complexa que tem 20% dos seus nós removidos aleatóriamente: (a) é a rede original e os nós marcados em vermelhos são os nós que serão removidos e (b) é a rede (a) já com os nós removidos.

Assim, descobrir qual é o efeito da falha de uma certa fração de nós na conectividade do restante da rede tem sido o objetivo de inúmeros pesquisadores. Se ao invés da rede representar computadores ela representar pessoas, as falhas em uma rede podem representar a quantidade de pessoas que contraíram alguma doença, por exemplo uma gripe.

Se os nós de uma rede são removidos de modo aleatório, como mostrado na Figura 3.8, o efeito é usualmente pequeno. Isto se deve ao fato de que as redes possuem um alto número de nós de grau um e, em uma remoção aleatória, esses nós teriam uma alta probabilidade de serem removidos. A remoção de nós de grau um não altera a estrutura da rede já que eles se encontram na periferia da rede. A Figura 3.8 (b) apresenta a remoção aleatória de 20% dos nós de uma rede e como é notado, a rede continua robusta. Contudo, se os nós são removidos de um modo cuidadoso, isto é, segundo alguma ordenação que não seja aleatória, o dano causado na rede pode ser grande.

Em (Albert et al., 2000; Cohen et al., 2000) a resistência das redes é analisada segundo duas maneiras diferentes:

- A remoção aleatória, chamada de **falha**;
- A remoção segundo a ordenação dos nós pelo seu grau, chamada de **ataque**;

As redes livres de escala são muito resistentes a falhas (remoção aleatória de nós), mas elas são substancialmente menos robusta quanto a um ataque. Isto acontece por que durante as falhas, uma grande quantidade de nós removidos são nós de grau um, já que estes são maioria nas redes reais. Já os ataques removem os nós de alto grau, fazendo com que a rede se torne desconexa muito mais rápidamente. Na Seção 3.4 será apresentada a comparação quanto a resistência a falhas e ataques entre as redes reais e a rede sintética *Erdös-Rényi*.

As falhas e ataques às redes são geralmente analisadas quanto ao diâmetro e ao tamanho (número de nós) da maior componente conexa. O diâmetro aumenta conforme mais nós são removidos, mas esse aumento é mais lento nas redes livres de escala do que nas randômicas. Similarmente, o tamanho da maior componente conexa (GCC) decai mais devagar nas redes livres de escala do que nas randômicas.

Muitos dos trabalhos de como a informação flui e influencia nós através da rede são feitos no contexto de epidemiologia e no de disseminação de doenças (Bailey, 1975; Chakrabarti et al., 2008). Os trabalhos de disseminação de doenças em redes e imunização focam frequentemente em determinar o valor que é chamado *Epidemic Threshold* (Bailey, 1975), um valor crítico da transmissão de vírus sobre o qual uma epidemia é criada.

A comunidade de epidemiologia tem desenvolvido modelos chamados SIR - Suscetível/Infectado/Recuperado (*Susceptivle/Infective/Removed*) e SIS - Suscetível/Infectado/Suscetível (*Susceptivle/Infective/Susceptivle*) (Bailey, 1975) de uma infecção. Suscetível (S), significa que uma pessoa pode contrair uma doença mas ainda não a tem; Infectado, significa que a pessoa tem a doença e pode transmiti-la, e Recuperado (R) significa que a pessoa já esta recuperada e não pode nem passar a doença nem contraí-la novamente, ou que a pessoa morreu. O modelo *SIS* é o ideal para uma gripe comum, já que os nós (isto é, pessoas) podem estar infectadas, se recuperarem e serem infectadas novamente.

Trabalhos recentes mostram que o *Epidemic Threshold* de um grafo no modelo SIS é $\beta/\delta = 1/\lambda_1$, na qual β é a proporção de nascimento do vírus (nós infectados) e δ a proporção de morte do vírus (nós curados ou mortos) e λ_1 é o primeiro autovalor da rede original (Chakrabarti et al., 2008). Acima do *Epidemic Threshold* a rede está em perfeito funcionamento, abaixo do *Epidemic Threshold* a rede representa uma situação de epidemia.

3.3.6 Predição de Ligações

A predição de ligações pode ser definida como, dado um "*snapshot*" de uma rede complexa em um tempo t, quer se prever com uma certa acurácia as arestas que irão surgir na rede complexa no tempo futuro $t+1$. Contudo, esta técnica também pode ser usada, de modo não trivial, para a encontrar de arestas que não foram encontradas durante a coleta dos dados usados para a construção do grafo.

Dentre as técnicas de predição de ligação destacam-se as baseadas em propriedades estruturais do grafo (Huang, 2006; Liben-Nowell e Kleinberg, 2003). Um exemplo

interessante é apresentado em (Clauset et al., 2008), em que a descoberta de grupos, isto é comunidades, em redes complexas é usado para o auxílio na identificação de ligações faltantes, já que pares de nós pertencentes a uma mesma comunidade têm mais chance de serem conexos entre si do que pares de nós pertencentes a comunidades diferentes. Este método se diferencia da predição de ligação tradicional, pois normalmente esta tarefa visa descobrir arestas que virão a existir na rede complexa quando esta evoluir (crescimento do número de nós e arestas com o passar do tempo) e não uma aresta perdida na construção da rede complexa. Entretanto, este método não funciona para todos os tipos de redes complexas, já que o método não consegue detectar comunidades em redes complexas que não possuam grupos muito definidos. Há uma coleção de trabalhos que vem usando e desenvolvendo algoritmos na área de predição de ligações (Acar et al., 2009; Hasan et al., 2006; Kashima et al., 2009; Kunegis e Lommatzsch, 2009; Lu e Zhou, 2009).

Uma das dificuldades da predição de ligações é que as redes complexas tendem a ser esparsas. Para driblar esta dificuldade, outros modelos fazem uso não só de propriedades estruturais do grafo mas também de características relacionais baseadas nos atributos dos nós do grafo. Esta abordagem é mais conhecida na área de aprendizado relacional ou aprendizado multi-relacional, que tem por objetivo não só o uso da estrutura dos grafos, mas também a descrição dos mesmos por meio de uma base de dados relacional ou lógica relacional ou de primeira ordem. Assim, além das ligações entre as tuplas formando um grafo, também há características, isto é, informações, relacionadas aos nós do grafo (Getoor e Diehl, 2005a; Hasan et al., 2006; Popescul et al., 2003; Taskar et al., 2004). Entretanto, esta informação complementar dos nós e arestas nem sempre esta disponível, o que inviabiliza a aplicação desses algoritmos nesses casos.

3.4 Modelos de grafos

Além dos estudos empíricos das grandes redes, há um trabalho considerável em modelos para geração de redes complexas. Ambos modelos, determinísticos e estocásticos, têm sido explorados. Os modelos de geração de redes complexas sintéticas fazem com que pela inclusão de arestas, a rede possua uma propriedade ou uma distribuição estatística global.

Os modelos são ferramentas úteis para avaliarem uma hipótese sobre o processo de formação das redes complexas. Redes sintéticas são úteis para a construção de algoritmos melhores e mais eficientes, já que torna possível a construção de redes muito grandes.

Dentre os modelos probabilístico para redes complexas, o *Erdös-Rényi* (Erdos e Renyi, 1960) é o mais estudado. Neste modelo, cada par de nós tem uma probabilidade independente e idêntica de serem conectados. O estudo desse modelo levou a uma teoria matemática muito rica. Contudo, este gerador produz redes que não seguem muitas das propriedades identificadas nas redes complexas reais.

O modelo *Erdös-Rényi* é também conhecido como modelo randômico e há duas maneiras diferentes de construção da rede para este modelo: $\mathcal{G}_{\mathcal{N},\mathcal{M}}$ e $\mathcal{G}_{N,p}$.

- $\mathcal{G}_{\mathcal{M},\mathcal{N}}$é a rede randômica gerada contendo N nós e M arestas. Neste modelo é possível que se tenha mais de uma rede contendo a mesma quantidade de nós e arestas, já que a ligação entre os nós é feita de maneira aleatória. Neste modelo M arestas são criadas entre pares de nós escolhidos aleatoriamente do conjunto N.

- $\mathcal{G}_{N,p}$ é a rede randômica contendo N nós, sendo que os pares de nós são conectado com uma probabilidade p. Assim, cada nó $v_i \in \mathcal{V}$ é visitado e conectado a outro nó $v_j \in \mathcal{V}$ formando uma aresta com probabilidade p.

Ambos os modelos são equivalentes, contudo por ser mais simples o modelo $\mathcal{G}_{N,p}$ é o mais usado para as formulações matemáticas.

Uma característica importante analisada nas redes randômicas (*Erdös-Rényi*) é a chamada "fase de transição". Nesta fase, a rede é pouco conectada, isto é, ela possui uma grande componente conexa, na qual a presença de ciclos é rara, e se for removida alguma fração de nós, mesmo que pequena, a grande componente conexa será destruída e haverá apenas componentes conexas pequenas e de tamanho similares. Por isso, este momento é conhecido como fase de transição, pois, antes dele a rede é conexa e depois dele a rede é composta por pequenas componentes conexas. A fase de transição também é conhecida como "Percolation" ou "Ponto Crítico" (Kong et al., 2006), pois neste ponto a estrutura da rede muda abruptamente. Este ponto também é o mesmo observado em fenômenos

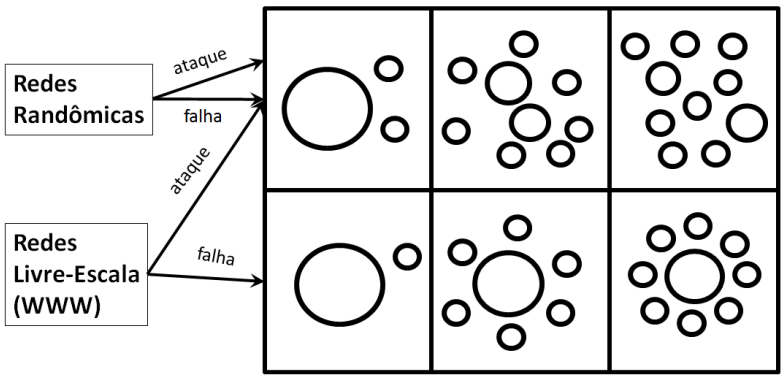


Figura 3.9: Comparação do comportamento do tamanho da rede quanto a remoção de nós aleatória e nós de alto grau. (Albert et al., 2000)

naturais, que podem ou não ser mapeados como grafos, como a tensão mecânica que causa terremotos, fogo em florestas, avalanches, etc (Bailey, 1975; Bak, 1996; Chen et al., 1991).

Nas redes randômicas (*Erdös-Rényi*) a "fase de transição" acontece quando $\frac{1}{|\mathcal{V}|} * \sum_{i=1}^{|\mathcal{V}|} d(v_i) = |\mathcal{V}| * p = 1$. Quanto $\frac{1}{|\mathcal{V}|} * \sum_{i=1}^{|\mathcal{V}|} d(v_i) < 1$ a rede é formada por pequenas componentes conexas e com $\frac{1}{|\mathcal{V}|} * \sum_{i=1}^{|\mathcal{V}|} d(v_i) > 1$ a rede se torna cada vez mais conexa. A distribuição do grau de uma rede *Erdös-Rényi* segue uma distribuição exponencial, a distribuição de Poisson (Durrett, 2007; Erdos e Renyi, 1960).

Além da diferença quanto a distribuição do grau, o coeficiente de clusterização em redes reais é mais alto se comparado com o de outras redes sintéticas, como as randômicas, com características semelhantes, isto é, mesmo número de nós, arestas e média do grau do nó (Dorogovtsev et al., 2002; Ravasz e Barabási, 2003).

Outra característica é quanto ao comportamento típico das de redes, uma livre de escala e a outra randômica, quanto as falhas e ataques, ilustrado na Figura 3.9. As circunferências representam o tamanho da rede quanto as suas componentes conexas, as circunferências menores representam as componentes conexas menores.

Como ilustrado, as redes *Erdös-Rényi* apresentam o mesmo comportamento quanto a falhas e ataques. Isso acontece pois as redes *Erdös-Rényi* não tem a distribuição de grau seguindo uma lei de potência, isto é, os nós não tem grau muito maior que a média do grau dos nós da rede. Já as redes reais, são resistente a falhas, devido à grande presença de nós de grau um, que não afetam o comportamento da rede quando removidos aleatoriamente. Entretanto, durante um ataque, quando os nós de alto grau são removidos, a rede tem

comportamento semelhante as redes *Erdös-Rényi*.

Com o avanço no estudo das redes complexas, outros modelos geradores de redes sintéticas que as tornam mais realísticas foram criados. Apesar de em muitas situações ser difícil distinguir uma rede real de uma sintética, os modelos existentes ainda não atingiram uma maturidade igual a que existe na geração de dados sintéticos tradicionais. Dentre os modelos clássicos estão o *Preferential Attachment* (Barabasi e Albert, 1999) e o *Small-World* (Watts e Strogatz, 1998). Estes e outros modelos são detalhados a seguir.

O modelo *Preferential Attachment* (Barabasi e Albert, 1999; Barabási et al., 2002; Kleinberg et al., 1999; Kumar et al., 1999; Winick e Jamin, 2002), tem a sua construção baseada no lema "ricos se tornam cada vez mais ricos". Isto é devido ao fato de que os novos nós inseridos na rede são conectados preferencialmente a nós de alto grau já existentes na rede. Este comportamento leva a uma distribuição do grau ser uma lei de potência e à rede possuir um diâmetro pequeno. Com isso, o diâmetro cresce lentamente com a inserção de novos nós, o que viola a propriedade que diz que o diâmetro diminui com o crescimento da rede.

Outra família de geradores de redes complexas tem por objetivo a propriedade de diâmetro pequeno e agrupamento local dos nós (triângulos) nas redes. Entre esses modelos destacamos o *Small-World* (Watts e Strogatz, 1998) e o *Waxman* (Waxman, 1988). Uma terceira família de modelos demonstra que a distribuição de grau segue uma lei de potência porque um nó tenta otimizar a sua conectividade com seus vizinhos. Dentre esses modelos tem o *Forest Fire*(Leskovec et al., 2007a), *Copying Model*(Kumar et al., 2000) e *Winner does not take it all* (Pennock et al., 2002).

Em suma, a grande maioria dos geradores focam em apenas uma propriedade estática, e acaba negligenciando outras. Assim, um conjunto de novos geradores que se baseiam em mais de uma propriedade estática das redes complexas vem surgindo nos últimos anos. Esses modelos também tem como objetivos simular o processo evolutivo das redes. Esses geradores são classificados como recursivos, como o RTM (Akoglu et al., 2008) e o RTG (Akoglu e Faloutsos, 2009).

Há geradores que visam a transformação de outros tipos de dados em redes com-

plexas. Um exemplo é a construção de uma rede na qual os nós representam as cidades e as arestas a distância em quilômetros entre as cidades. Um número k pode ser usado como limitante para o número de cidades a que cada cidade se liga. Neste caso, cada cidade (nó da rede) tem $d_{out} = k$. Esse gerador é chamado de Modelo *k nearest neighbor model* ou $k-$vizinhança mais próxima, na qual k é o número de vizinhos que cada nó se conecta. O modelo *knn* também faz uso de uma função de distância que mede quão distante dois objetos do conjunto estão. A seguir é apresentada uma definição formal do modelo *knn*.

Definição 2. ***Rede k-vizinhos mais próximos:*** *Dado um conjunto de dados $\mathbb{S}$ de cardinalidade N e uma função de distância $\mathcal{DF}$, o grafo* knn *é definido como o conjunto de nós $V = \{\mathbb{S}\}$ e o conjunto de arestas é dado por $\mathcal{E} = \{\langle s_1, s_2\rangle \,|s_1 \in \mathbb{S} \wedge s_2 \in k\text{-}NNq(\mathbb{S}, s_1, k)\}$, sendo $k\text{-}NNq(\mathbb{S}, s_1, k)$ o resultado da consulta aos k-vizinhos mais próximos de s_1.*

Em um grafo *knn* todos os k vizinhos mais próximo de um elemento s_1 são conectados a ele. O fato do elemento s_1 ter o elemento s_2 como vizinho não implica que o elemento s_2 tenha o elemento s_1 como um dos seus k vizinhos mais próximos. Por isso este modelo produz grafos naturalmente direcionados e apenas a distribuição do grau de entrada pode seguir uma lei de potência, já que o grau de saída é sempre k. Existem duas abordagens para transformar um grafo *knn* direcionado em um grafo *knn* não direcionado. A primeira, e mais simples, é não considerar a direção das arestas e tratar o grafo como não direcionado. A segunda, mais elaborada, considera somente como arestas no grafo *knn* não direcionado as arestas nas quais no grafo *knn* original possuem as duas direções, isto é, uma aresta (s_i, s_j) vai existir no grafo *knn* não direcionado se as arestas (s_i, s_j) e (s_j, s_i) existirem no grafo *knn* direcionado. O grafo gerado pelo último método é denominado "*mutual k-nearst neighbor graph*". Apesar do modelo *knn* ser pouco explorado na literatura (Paterson e Yao, 1992), sua aplicabilidade já foi comprovada na descoberta de anomalias (Hautamaki et al., 2004), agrupamentos (Maier et al., 2009) e construção e análise de redes sociais (Lathia et al., 2008).

3.5 Evolução das Redes Complexas

Como apresentado até aqui, o estudo de padrões, como a avaliação de aderência à leis de potência para a distribuição do grau, identificação de comunidades, o fenômeno *Small World*, e outros, têm tido como foco redes estáticas, isto é, um único "snapshot" de uma grande rede ou um conjunto pequeno de "snapshots". Contudo, com o amadurecimento da área de mineração em redes complexas, novos questionamentos surgem, entre eles: Como as redes complexas evoluem? Quais são os padrões considerados "normais" para este crescimento?

Modelos como o *Preferential Attachment* (Barabasi e Albert, 1999; Newman, 2003), *Small-World* (Broder et al., 2000; Milgram, 1967; Watts e Strogatz, 1998) assumem que a média do grau de saída d_{out} permanece constante com o tempo e que o diâmetro cresce lentamente. Entretanto, estudos mais recentes como em (Akoglu et al., 2008; Leskovec et al., 2007a; McGlohon et al., 2008) mostram que o diâmetro tende a diminuir com o crescimento da rede e a média do grau de saída tende a aumentar. Além disso, também foi verificado empiricamente que a maior componente conexa passa de pouco conectada (aspecto de uma árvore) para muito conectada na rede e o primeiro autovalor tem seu comportamento associado a uma lei de potência com o expoente menor que 0.5

As leis derivadas desses estudos, sendo algumas delas leis de potência, são apresentadas a seguir:

- **Lei de Densificação** (DPL - *densification power law*): O número de nós $V(t)$ e o número de arestas $E(t)$ crescem durante a evolução da rede de maneira super-linear, que segue uma lei de potência $E(t) \propto N(t)^{\alpha}$, com $\alpha > 1$ (Leskovec et al., 2007a).

- **Diâmetro pequeno e decrescente** (SSDL - *small and skrink diameter law*): o diâmetro (efetivo) de uma rede deve ser pequeno (Albert et al., 1999) com um possível pico no ponto chamado "Gelling Point", que é o ponto com o maior diâmetro (McGlohon et al., 2008). Além disso, o diâmetro deve diminuir com o tempo (Leskovec et al., 2007a).

- **Primeiro autovalor** (λ_1PL - λ_1 *power law*): o maior autovalor $\lambda_1(t)$ da matriz de adjacência de uma rede versus o número de arestas $E(t)$ durante a evolução da rede segue uma lei de potência: $\lambda_1(t) \propto E(t)^{\alpha}$, com $\alpha < 0.5$ (Akoglu et al., 2008).

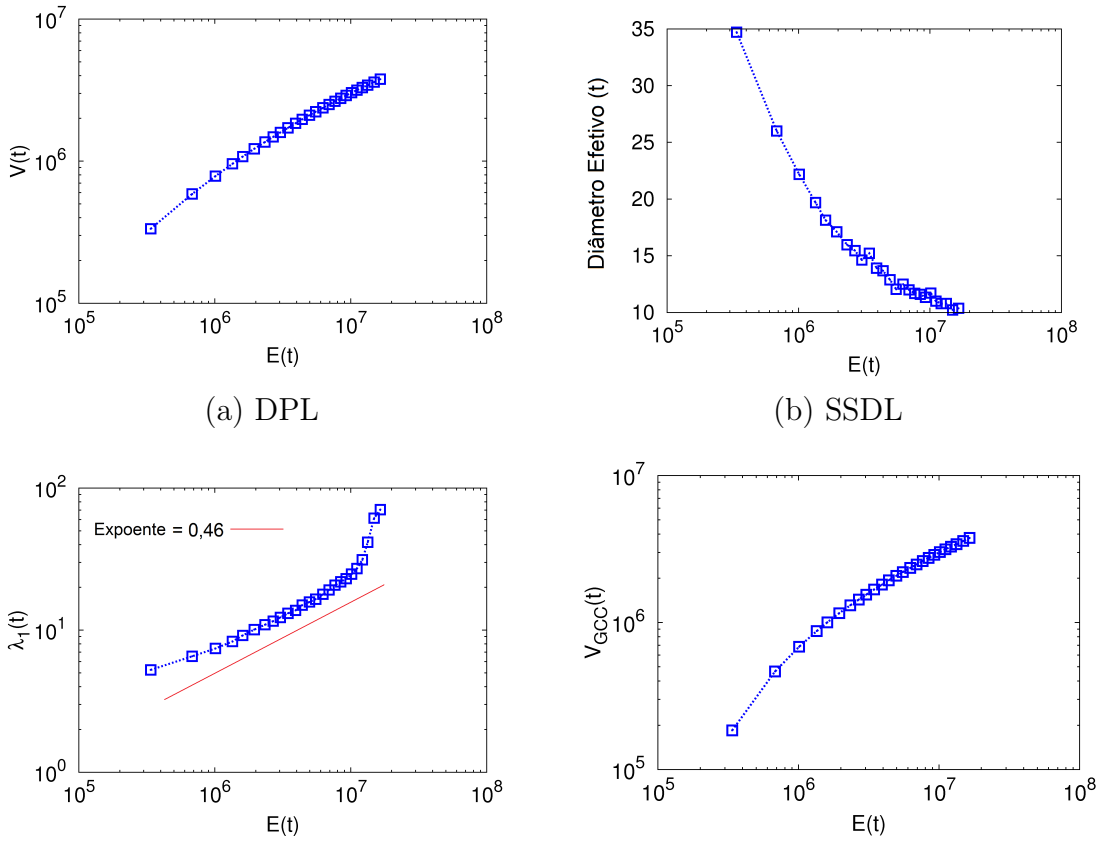
- **Grande Componente Conexa** (GCCL - *GCC Law*): a maior componente conexa de uma rede cresce em número de nós $N_{GCC}(t)$ versus o número total de arestas $E(t)$ da rede com o tempo (McGlohon et al., 2008). Isto significa que com a evolução a tendencia é todos os nós da rede se tornarem uma única componente conexa.

O estudo da evolução de redes complexas é de suma importância, pois permite o desenvolvimento de modelos mais precisos e a implementação de cenários "*O que aconteceria se*". Além disso, a evolução ajuda em detecção de anomalias, amostragem de redes e predição do futuro da rede. A Figura 3.10 mostra a evolução da rede que é o conjunto de Patentes U.S. mantida pela *National Bureau of Economic Research* (Leskovec et al., 2007a) com $|\mathcal{V}| = 3.774.362$ e $|\mathcal{E}| = 16.512.782$ de 1975 até 1999. As arestas deste conjunto são as referências entre as patentes.

3.6 Considerações finais

Neste Capítulo foi apresentada a conceituação da teoria dos grafos utilizada pelas redes complexas e as principais propriedades estatísticas e os algoritmos utilizados na área de redes complexas. As redes complexas estão cada vez mais presentes nos sistemas computacionais e com isso o seu entendimento torna-se cada vez mais necessário. Assim, são importantes a descoberta de novas leis de potência, o desenvolvimento de novos modelos e ferramentas que consigam analisar como as redes evoluem ou novos padrões.

As principais contribuições desta tese na área de redes complexas são apresentadas no Capítulo 5. Estas contribuições são representadas pelo algoritmo `ShatterPlots` que permite a classificação de uma rede em sintética ou real por meio da deleção aleatória de aresta até o ponto chamado `ShatterPoint` (fase de transição das redes reais e sintéticas) e pelo método `FCR`, que faz a análise de cliques de tamanho 4 e 5 para a diferenciação da topologia de nós que possuam quantidade de triângulos semelhantes. Nesses trabalhos



(a) DPL (b) SSDL

(c) λ_1PL (d) GCCPL

Figura 3.10: Evolução da rede complexa Patentes U.S. segundo as leis de evolução. Em (a) o número de nós e arestas crescem proporcionalmente, (b) o diâmetro diminui com a evolução da rede, (c) o seu primeiro autovalor cresce com expoente menor que 0.5 e em (d) o número de nós da GCC cresce até todos os nós da rede formarem uma única componente conexa

são apresentadas leis bastante uteis e interessantes. Também é apresentada a extensão do coeficiente de clusterização para a identificação de nós com topologia diferente mas com grande influência no primeiro autovalor da matriz de adjacência do grafo.

Parte II

Trabalhos Desenvolvidos

Capítulo 4

Biased Box Sampling - BBS

4.1 Considerações iniciais

Grande parte das técnicas de redução de tuplas está baseada na amostragem de dados uniforme, na qual cada elemento tem a mesma probabilidade de ser selecionado. No entanto, a uniformidade na distribuição de valores e a independência entre atributos são propriedades bastante incomuns em dados do mundo real. Usualmente, dados reais são caracterizados pela ampla presença de correlações entre seus atributos e pela não uniformidade na sua distribuição.

A detecção de agrupamentos foi escolhida dentre os processos de mineração de dados para ser explorada neste trabalho, pois, a não uniformidade é bastante marcante. A amostragem de dados é um forte paradigma na redução de dados, usada para viabilizar a complexidade inerente da tarefa de detecção de agrupamentos, aumentando a velocidade dos algoritmos. No caso de dados com grandes variações no tamanho dos agrupamentos, a amostragem balanceada tende a ser melhor. Nesta amostragem, a probabilidade de um objeto ser incluído depende da densidade local do agrupamento.

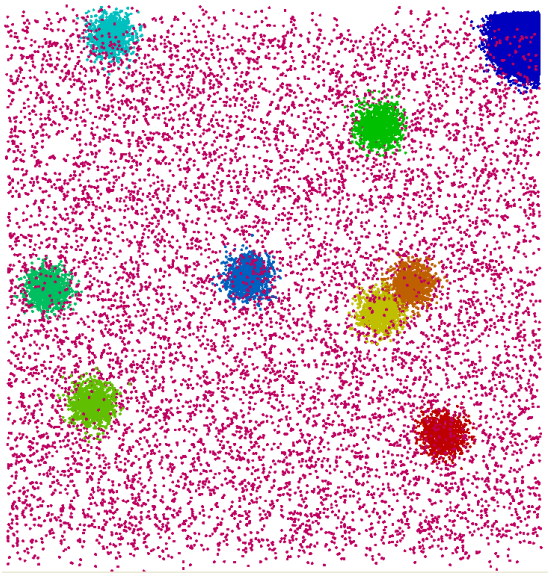


Figura 4.1: Conjunto de dados 20-dimensional (projeção 2D) contendo um agrupamento com 50 mil elementos, oito com mil elementos e 15% de ruído.

Na Figura 4.1 é mostrado um exemplo de um conjunto de dados distribuídos de maneira não uniforme. Esse conjunto contém nove agrupamentos, um contendo 50 mil elementos e outros oito contendo mil elementos, além de 15% de ruído distribuído uniformemente. Extraindo-se uma amostra de 1% deste conjunto, esta amostra irá conter um agrupamento com cerca de quinhentos elementos, e outros oito contendo cerca de dez elementos bem como algum ruído. Nota-se que, numa situação real, o número de agrupamentos será descoberto apenas depois do algoritmo de detecção de agrupamentos ter sido executado, portanto esta informação não está disponível para o processo de amostragem.

Entretanto, neste exemplo, o algoritmo de detecção de agrupamento não será capaz de encontrar agrupamentos tão pequenos. O problema é que quando a representação de um agrupamento em um conjunto de dados é significantemente menor que dos outros agrupamentos, o algoritmo de detecção de agrupamentos pode perder agrupamentos pequenos, misturando-os com o ruído. Uma questão que se abre é: *"Como uma amostragem de um conjunto de dados multidimensional pode ser realizada sem perder agrupamentos, mesmo que este estejam desbalanceados (isto é, o número de elementos é bem diferente), sem ter nenhum conhecimento prévio sobre os agrupamentos?"*.

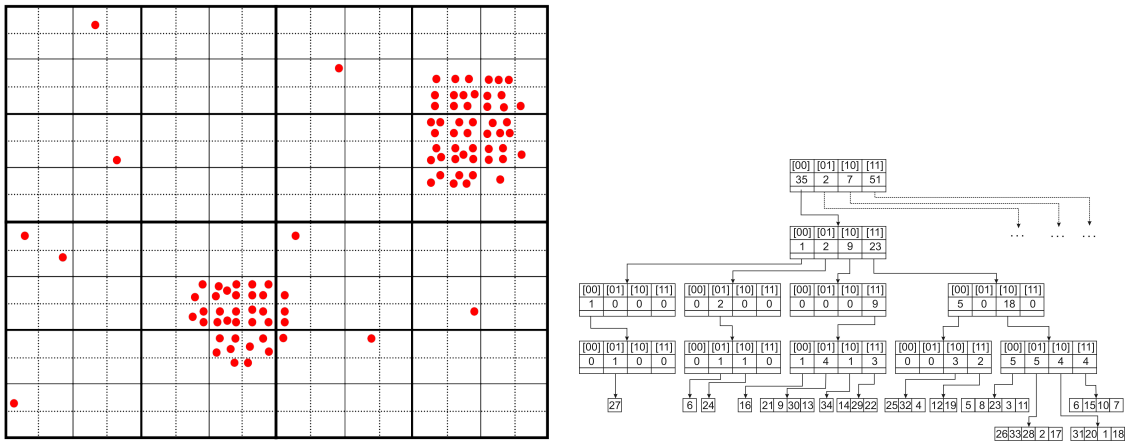
Baseado nessa questão desenvolveu-se uma nova técnica que realiza uma amostragem de dados balanceada pela densidade local. O algoritmo **BBS - Biased Box**

Sampling, foi desenvolvido para implementar essa técnica, apresentando uma complexidade linear $O(N \cdot E)$ em relação a quantidade de objetos N e a quantidade de atributos E. Os experimentos realizados mostram a sua eficiência na redução do tempo de execução da amostragem balanceada, sem afetar a precisão dos algoritmos de detecção de agrupamentos usados em sequência.

4.2 Biased Box Sampling

A principal ideia do algoritmo Biased Box Sampling (BBS) é dividir o espaço em 2^E regiões, sendo que cada atributo divide o espaço ao meio, e conta-se o número de elementos do conjunto de dados que pertence a cada região. A Figura 4.2 (a) mostra um conjunto de dados de duas dimensões representado em um hiper-reticulado. A estrutura que implementa o hiper-reticulado é uma árvore chamada de *MG-Tree* (*Multi Grid Tree* - Árvore Multi Resolução) apresentada na Figura 4.2 (b). Cada região que possui mais elementos que um dado *threshold* δ é recursivamente dividida, gerando como estrutura uma *"hyper-quad tree"*, a *MGc-tree* (*Multi Grid Compact Tree* - Árvore Multi Resolução Compacta) como representado graficamente pela Figura 4.3 (b). Quando uma região não alcança o dado *threshold*, ela é representada por um nó folha e seus elementos são amostrados. Os nós folhas podem estar em diferentes níveis, refletindo a densidade do espaço de cada região. A execução da amostragem uniforme em cada nó folha irá recuperar mais elementos de regiões menos densas, refletindo a variação da densidade sobre o espaço de dados completo.

O algoritmo BBS tem três partes. A primeira cria a estrutura de multi-resolução, implementada como uma árvore. Ela é similar ao algoritmo LiBOC (Traina et al., 2000), a principal modificação é a necessidade de armazenar os elementos em cada nó folha. Como esta nova técnica tem custo computacional constante para cada ponto, este passo tem a mesma complexidade que o LiBOC, ou seja $O(N \cdot E)$. O resultado é uma árvore - *MG-tree* - onde cada nó não folha armazena a contagem de elementos pertencente a região correspondente, uma identificação da região é de zero até 2^E ponteiros para a próxima resolução que divide a corrente. Um nó folha armazena apenas os ponteiros para



(a) (b)

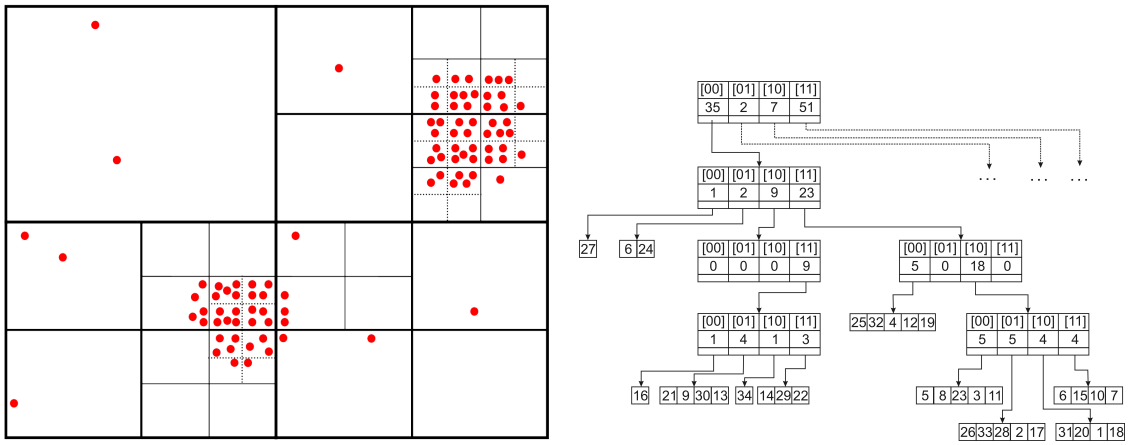
Figura 4.2: (a) Hiper-reticulado de 2 dimensões e 4 níveis (b) MG-Tree (*Multi Grid Tree* - Árvore Multi Resolução)

os elementos que corresponde àquela região, a qual formam uma lista indexada. Nota-se que, nesse passo, a árvore tem em cada nó folha uma resolução $R-1$. A construção da *MG-tree* é ilustrada graficamente pela Figura 4.2 (a) e a sua construção é representada na Figura 4.2 (b). O valor R define o número máximo de resoluções que o algoritmo deve tentar para obter uma boa amostragem. O mínimo é o limite de como uma boa amostragem deve ser, e o máximo é o limite da alta densidade da amostra do conjunto de dados. Se R for ajustado muito alto, o algoritmo irá requerer mais memória para operar (já que a complexidade de memória requerida pelo algoritmo é $O(N \cdot E \cdot R)$). No entanto, depois do *threshold* o aumento de R não irá melhorar a amostra. A primeira parte do BBS é mostrado como Algoritmo 4.2.1.

Algoritmo 4.2.1 Biased Box Sampling: Criação da árvore - BBSCT

Entrada: Conjunto de dados com N elementos e E atributos, número de níveis R
Saída: árvore multi-resolução - $MG-tree$

1: Normalize os atributos do conjunto de dados dentro de um cubo unitário
2: **para** para cada ponto t no conjunto de dados **faça**
3: **para** $r = 1/2^j,\ j = 1, 2, 3, .., R-1$ **faça**
4: Selecione a célula no próximo nível onde t existe como i
5: Incremente o contador $C_{i,r}$;
6: **se** nível $= R-1$ **então**
7: Insira ponto t em uma lista indexada de células i
8: **fim-se**
9: **fim-para**
10: **fim-para**



(a) (b)

Figura 4.3: (a) Hiper-reticulado de 2 dimensões e 4 níveis concatenado (b) MGc-tree - *Multi Grid Compact Tree - Árvore Multi Resolução Compacta*

A segunda parte do algoritmo BBS é responsável por reduzir as regiões mais profundas da Árvore Multi-Resolução, menos densas, transformando a *MG-tree* gerada pelo Algoritmo 4.2.1 em uma árvore condensada *MGc-tree*. Esta parte, mostrada como Algoritmo 4.2.2, olha para cada célula na árvore onde o número de elementos é menor que o threshold δ. Quando tais células são encontradas, as listas das células filhas são concatenadas na lista indexada, transformando seus pais em nós folhas. O resultado é a árvore compacta *MGc-tree*, como representada pelas Figuras 4.3 (a) e (b), que tem todos os nós folhas com aproximadamente o mesmo número de elementos.

Algoritmo 4.2.2 Biased Box Sampling: Join List - BBSJL
Entrada: *MG*-tree e taxa de amostragem *Ratio*
Saída: Estrutura concatenada - $MGc-tree$
1: $\delta = E * 100/(2 * Ratio)$
2: **enquanto** Célula $\neq$ NULL **faça**
3: **se** Célula do próximo nível $\neq$ NULL **então**
4: **se** Nível atual $<$ R **então**
5: Chama $BBSJL(Nvel\ Atual + 1, Ratio)$
6: **senão se** $C_{r,i} < \delta$ **então**
7: Concatena lista indexada de células irmãs e transforma os pais em nós folhas;
8: **fim-se**
9: **fim-se**
10: Célula recebe próximo nível de célula;
11: **fim-enquanto**

O valor do *threshold* (δ) determina o número mínimo de elementos que determinada

célula deve conter para que seja vantajoso descer para o próximo nível da árvore. Esse valor é estimado no passo 5 do Algoritmo 4.2.2 como $\delta = E * 100/(2 * Ratio)$, ou seja, diretamente proporcional ao número de atributos (E) e inversamente proporcional a taxa de amostragem ($Ratio$), pois, quanto menor a taxa de amostragem mais elementos deve ter a célula para que ao menos um ponto seja recuperado ($100/Ratio$) e quanto maior o número de atributos mais os elementos serão divididos no próximo nível. O próximo passo é realizar a amostragem.

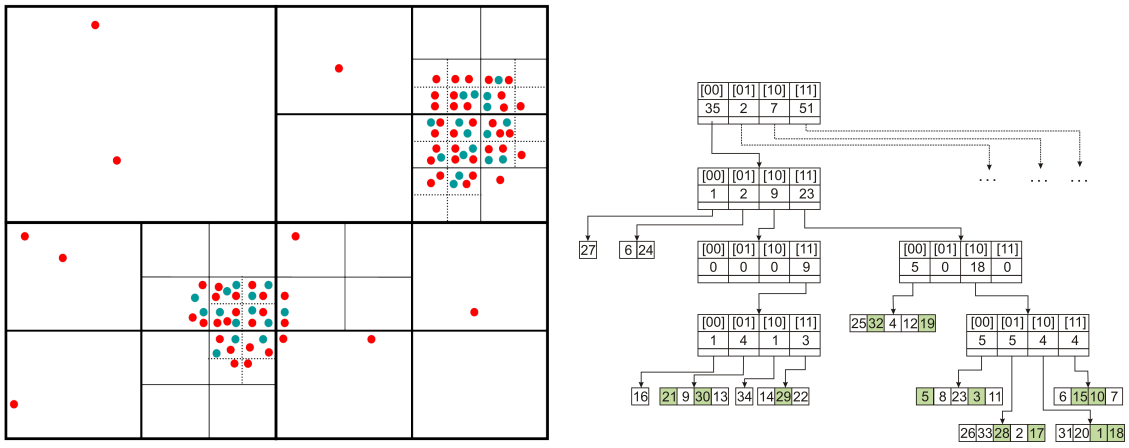
A terceira (e última) parte do algoritmo BBS executa efetivamente a amostragem em multi-resolução. O algoritmo percorre a árvore até o nó folha da *MGc-tree*, calcula o tamanho da amostra para aquele nó utilizando a diferença entre o nó folha e o máximo número de níveis de árvore para aumentar o número de elementos selecionados (passo 7 do Algoritmo 4.2.3). O *oversample* por sua vez é determinado pela diferença entre o número de níveis da árvore (R) e o nível atual, valorizando os níveis mais altos da árvore. Essa valorização permite que a amostragem realizada recupere os agrupamentos com pouca representatividade no conjunto. Mesmo com essa valorização dos níveis superiores, os elementos ruidosos em grande parte dos experimentos foram eliminados pelo algoritmo, pois estes tendem a estarem distribuídos de forma esparsa, já a partir do nível zero da árvore. O passo 8 usa a lista concatenada para escolher elementos da amostra. Na Figura 4.4 (a) é mostrado em destaque (verde) os elementos selecionados pelo processo de amostragem no Hiper-Reticulado e na Figura 4.4 (b) os mesmos elementos selecionados na árvore MGc-tree.

O algoritmo BBS, mostrado como Algoritmo 4.2.4, recebe como entrada o conjunto de elementos para ser amostrado, a quantidade de atributos, o numero de níveis e a taxa de amostragem, chama as partes mostradas como Algoritmos 4.2.1, 4.2.2 e 4.2.3 e retorna um vetor indexando com os elementos escolhidos para amostra do conjunto original.

Algoritmo 4.2.3 Biased Box Sampling: Extrai amostra - BBSES

Entrada: Estrutura de grid concatenada - $MGc-tree$ e taxa de amostragem $Ratio$
Saída: vetor com os índices dos elementos amostrados

1: **enquanto** Células $\neq$ NULL **faça**
2: **se** Célula do próximo nível $\neq$ NULL e First = NULL **então**
3: Chama $BBSES(Nvel\ Atual + 1, Ratio)$
4: **senão se** First $\neq$ NULL **então**
5: $tamanhodaamostra = Ratio * C_{r,i}/100$
6: **se** $Nvel\ Atual - R \neq 0$ **então**
7: $tamanhodaamostra = tamanhodaamostra * 2 * (R - Nvel\ Atual)$
8: **fim-se**
9: Seleciona um ponto da lista indexada e insere no resultado;
10: **fim-se**
11: Célula recebe a próxima célula;
12: **fim-enquanto**



(a) Hiper-reticulado (b) MGc-Tree

Figura 4.4: (a) Escolha dos elementos no Hiper-reticulado (b) Escolha dos elementos na MGc-Tree

Algoritmo 4.2.4 Biased Box Sampling

Entrada: Conjunto de dados com N elementos, E atributos, numero de níveis R e taxa de amostragem $Ratio$
Saída: Conjunto de dados com os elementos selecionados

Chama BBSCT(Conjunto de dados, R)
Chama BBSJL(MG-tree, Ratio)
Chama BBSES(MGc-tree, Ratio)

4.3 Resultados Obtidos

Foram realizados experimentos com diversos conjuntos de dados, que estão descritos a seguir. A mesma metodologia experimental foi aplicada para todos os conjuntos de dados. No primeiro passo foram geradas amostras com o algoritmo proposto BBS e com os algoritmos apresentados na Seção 2.3.2: DBS (*Density Biased Sampling*), GBS (*Grid Biased Sampling*) e a amostragem uniforme (US), gerando quatro conjuntos de amostra para cada conjunto de dados amostrado. Em um segundo passo, foi avaliada a qualidade das amostras obtidas quanto à preservação dos agrupamentos de dados em relação ao conjunto de dados original. Cada experimento foi repetido dez vezes, gerando dez amostras com os mesmos parâmetros. Os valores apresentados são a média do processamento de cada uma das dez amostras.

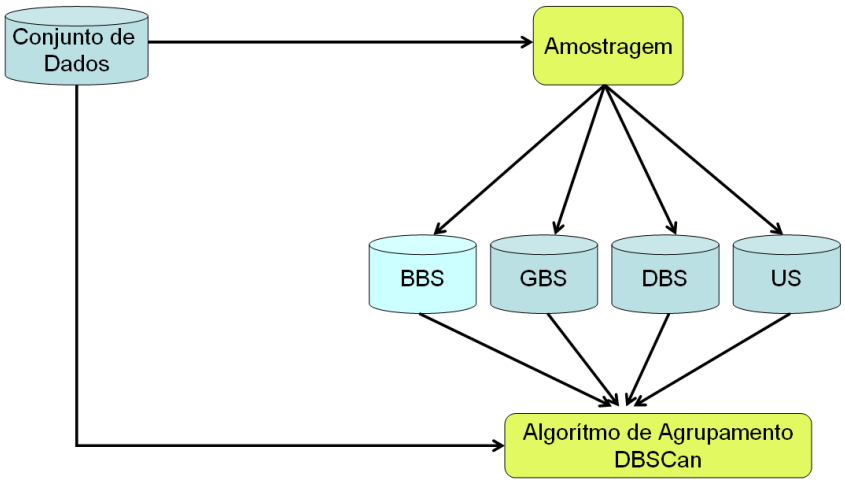


Figura 4.5: Metodologia utilizada para avaliação do algoritmo BBS

Os parâmetros do DBS foram ajustados conforme indicado em (Kollios et al., 2003). O parâmetro α foi ajustado do seguinte modo: para conjuntos de dados contendo ruído e várias densidades (incluindo agrupamentos pequenos), α foi atribuído com -0.25. Para conjuntos de dados para várias densidades mas sem ruído o α foi ajustado para -0.5. O outro parâmetro é o número de estimadores de kernel que foram utilizados, definido como 1000 seguindo a recomendação do autor. Entretanto, podemos notar que para a utilização do algoritmo DBS, é necessário um conhecimento prévio da distribuição do conjunto de dados, já que o parâmetro α varia segundo a distribuição do conjunto, isto

é, se tem ruído ou agrupamentos com tamanhos diferentes. No algoritmo GBS, apenas o parâmetro e necessita ser ajustado, para o qual foi utilizado o valor 0.5 segundo (Palmer e Faloutsos, 2000).

Para avaliar a precisão da técnica, foi aplicado o algoritmo de agrupamento DBScan (Ester et al., 1996), sobre os conjuntos de dados originais e sobre as respectivas amostras. O DBScan foi escolhido por não requerer o número de agrupamentos pré-determinado como parâmetro, isto é, ele pode encontrar os agrupamentos baseados nas propriedades dos dados, e por isso ele é apropriado para avaliar a qualidade das amostras. Outro fator importante é que o algoritmo DBScan consegue identificar ruído no conjunto de dados, o que não é possível com algoritmos baseados em particionamento e hierárquicos. O DBSCAN baseia-se na densidade local para descobrir os agrupamentos, e ele pode detectar ruído. A ferramenta WEKA[1] possui uma implementação do DBScan. O algoritmo requer os seguintes parâmetros: o MinPts que é o número mínimo de elementos que um agrupamento deve ter, e o raio ϵ que define a distância máxima para determinar se dois elementos são vizinhos ou não. Esses parâmetros foram ajustados da seguinte maneira: o valor de MinPts foi definido como sendo a taxa da amostragem aplicada ao número de elementos do menor agrupamento existente no conjunto de dados. O raio ϵ foi experimentalmente ajustado para cada um dos conjuntos.

Um resumo das informações dos conjuntos de dados utilizados nos experimentos é apresentado na Tabela 4.1. O conjunto OneBig foi gerado utilizando a ferramenta DBGen desenvolvida pelo grupo GBDI. Assim, para o OneBig foi possível variar a dimensão do conjunto e a quantidade de ruído presente. A quantidade de ruído inserida é sempre uma porcentagem do número de pontos já existente no conjunto.

Nome	# Elementos	Agrupamentos	E	Ruído
UniformClusters	10.000	5	2	Sim
Pendigits	10.992	8	16	Sim
OneBig	54000	9	5-100	variado

Tabela 4.1: Conjunto de dados utilizado para os experimentos

[1] http://www.cs.waikato.ac.nz/ml/weka/

4.3.1 Conjunto Uniforme

O primeiro conjunto a ser testado foi o "UniformClusters", proposto como um conjunto adequado para teste em (Kollios et al., 2003). Ele possui cinco agrupamentos em um espaço bidimensional, sendo um grande círculo com 30000 elementos, dois pequenos círculos cada um com dez mil elementos, duas elipses, cada uma com 20000 elementos, conectadas por uma ponte de ruído com mil elementos e mais 9000 elementos de ruído espalhado pelo espaço total, como descrito em (Kollios et al., 2003). Neste conjunto, tanto a distribuição intra-agrupamento como o ruído são uniformemente distribuídos. Por ser um conjunto bidimensional, ele pode ser visualizado facilmente, o que permite interpretar intuitivamente os resultados dos algoritmos.

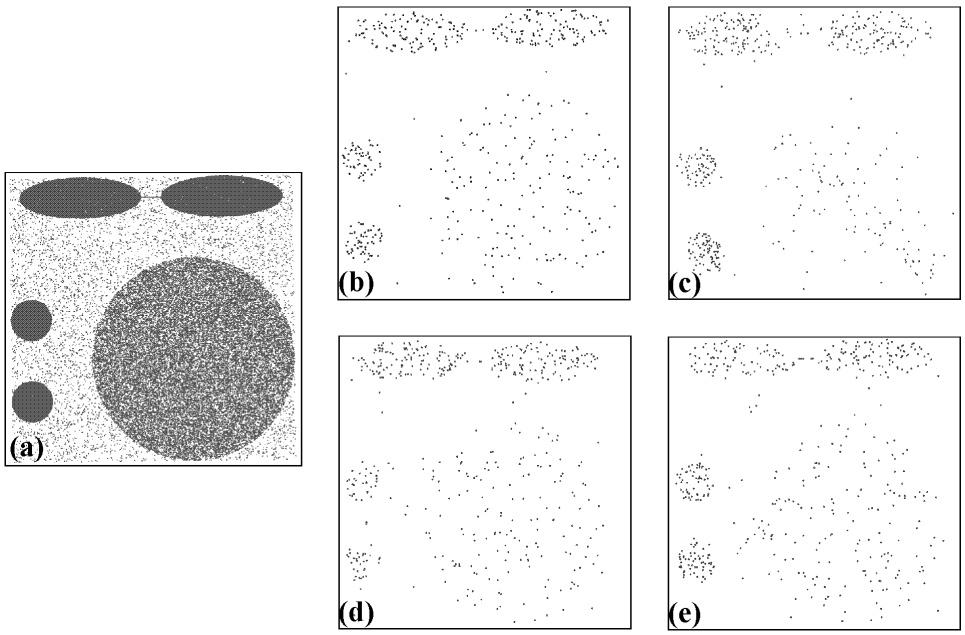


Figura 4.6: Visualização do conjunto "UniformClusters". (a)Conjunto original; e amostras com 0.5% de taxa de amostragem obtidas dos algoritmos: (b) BBS, (c) DBS, (d) GBS; (e) US.

A Figura 4.3.1 mostra a visualização do conjunto "UniformClusters", tanto o conjunto original (Figura 4.3.1(a)) quanto as amostras com tamanho de 0.5%(obtidas pelos algoritmos (b): BBS Figura 4.3.1, (c): DBS Figura 4.3.1, (d): GBS Figura 4.3.1 e (e): US Figura 4.3.1). Como se pode ver, as amostras são diferentes mas todas permitiram ao DBSCAN encontrar quatro agrupamentos. Contudo, visualmente pode-se ver que a

amostra gerada pelo BBS é mais próxima do conjunto original. O DBScan encontrou quatro agrupamentos em todos os conjuntos de amostras do "UniformClusters" e no conjunto original, apesar do conjunto original possuir cinco agrupamentos. Este resultado foi devido à ponte de ruído entre dois dos agrupamentos, a qual é preservada por todas os métodos de amostragem.

4.3.2 Conjunto Pendigits

O Conjunto "Pendigits"(obtido do UCI Machine Learning Repository[2]) é um conjunto bem conhecido que ilustra a eficiência do BBS para conjunto de dados reais. Para este conjunto foi aplicada uma metodologia um pouco diferente, já que o número de agrupamentos não era previamente conhecido. Primeiramente o DBScan foi aplicado ao conjunto de dados original, sendo o número mínimo de elementos (MinPts) ajustado para 10% do conjunto de dados. Para ϵ, vários valores foram avaliados, mas o DBScan foi capaz de encontrar 8 agrupamentos no conjunto de dados original apenas quando o ϵ foi ajustado para 0.4. O conjunto "Pendigits" possui avaliações de reconhecimento de escrita de dígitos, assim o número 8 foi escolhido por ser o valor mais próximo de 10. Devido a isso, o valor 0.4 foi o escolhido para os experimentos.

Tabela 4.2: Número de agrupamentos encontrados no conjunto "Pendigits".

Variação do tamanho da amostra				
Algoritmo	3%	4%	5%	10%
BBS	7	7	7	8
DBS	5	5	6	6
GBS	3	4	7	6
US	2	3	6	2

O resultado deste experimento é mostrado na Tabela 4.2. Para esse conjunto de dados reais o *BBS* obteve o melhor resultado em relação a todos os demais algoritmos de amostragem. A amostra de 10% do *BBS* permitiu que o mesmo resultado que o conjunto completo fosse obtido pelo DBSCAN, localizando 8 agrupamentos. Os outros

[2]http://www.ics.uci.edu/-mlearn/MLSummary.html

algoritmos não conseguiram encontrar mais do que 7 agrupamentos para a mesma taxa de amostragem.

4.3.3 Conjunto OneBig

O conjunto de dados denominado "OneBig" foi testado em diferentes configurações. A principal característica desse conjunto de dados é a grande diferença de tamanho entre seus agrupamentos. Este conjunto possui um agrupamento maior contendo 50000 elementos e outros 8 agrupamentos cada um com 500 elementos, isto é a diferença quanto ao número de elementos entre o maior e os menores é de 100 vezes. Para este conjunto foram criadas diversas configurações, variando-se a quantidade de ruído e a dimensão.

Tabela 4.3: Número de agrupamentos encontrados no conjunto "OneBig" com amostragem de 2%, 20 dimensões e variação da quantidade de ruído.

	Porcentagem de Ruído						
Algoritmo	0%	10%	20%	30%	40%	60%	80%
BBS	9	9	9	9	9	9	9
DBS	9	9	9	9	9	3	1
GBS	5	4	5	5	5	4	4
US	5	6	5	6	6	6	7

Na Tabela 4.3 são apresentados os resultados da variação de ruído no conjunto de dados "OneBig" para uma amostragem de 2%. Como esperado, os efeitos do ruído é significativamente notável nos casos do US, DBS e GBS. O outro fator que afeta bastante os algoritmos é a distribuição dos conjuntos de dados. Os algoritmos testados retornaram elementos dos agrupamentos não encontrados, mas a quantidade não foi suficiente para o DBScan encontrar todos os agrupamentos, isto é, a quantidade de pontos foi menor que a quantidade especificada no parâmetro MinPts. O algoritmo DBS retornou número de elementos suficientes apenas até 40% de ruído. Com 60% e 80% de ruído apenas o algortimo BBS retornou elementos suficientes para que o DBScan encontrasse todos os elementos. Os parâmetros para o DBScan foram ajustados da seguinte maneira, o MinPts=10, isto é 2% do tamanho do agrupamento menor, e o $\epsilon = 0.15$, o mesmo utilizado no conjunto de dados original. A distribuição gaussiana foi escolhida por ser a distribuição

mais próximas dos conjuntos de dados reais.

Tabela 4.4: Número de agrupamentos encontrados no conjunto "OneBig" com amostragem de 2%, 30% de ruído e variação da dimensionalidade.

Variação da Dimensionalidade					
Algoritmo	5	10	20	40	100
BBS	8	9	9	9	9
DBS	9	9	9	9	9
GBS	5	6	4	6	5
US	7	6	6	6	7

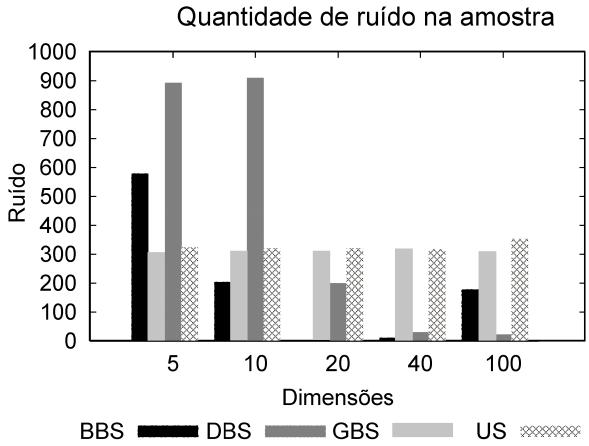


Figura 4.7: Quantidade de ruído para o conjunto "OneBig" com amostragem de 2%, 30% de ruído e variação da dimensionalidade.

Outro experimento realizado foi avaliar o impacto da dimensionalidade sobre o conjunto "OneBig", como apresentado na Tabela 4.4. O ruído foi ajustado para 30% e a taxa de amostragem em 2%. Claramente podemos notar que para 5 dimensões, apenas o algoritmo DBS retornou elementos suficientes de todos os agrupamentos, sendo o BBS o segundo melhor retornando 8 agrupamentos. Os algoritmo GBS e US não produziram resultados corretos em nenhumas das dimensionalidades, sendo fortemente influenciados pelos 30% de ruído. O algoritmo DBS conseguiu sempre recuperar todos os agrupamentos entretanto a quantidade de pontos considerados ruído que foram retornado foi maior que o BBS. Pode-se notar analisando o gráfico da Figura 4.7 que para 40 dimensões o BBS não retornou nenhum elemento de ruído.

4.3.4 Escalabilidade

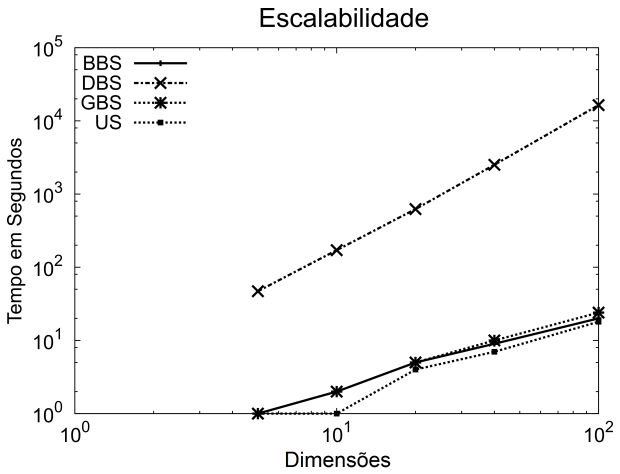


Figura 4.8: Escalabilidade dos algoritmos de amostragem quanto à dimensionalidade (escala logarítmica)

O gráfico apresentado na Figura 4.8 mostra a escalabilidade quanto ao número de dimensões do algoritmo BBS em relação ao demais algoritmos. Além de mostrar uma grande eficiência no aumento da representatividade, recuperando em grande parte dos experimentos uma maior quantidade de elementos dos agrupamentos, podemos constatar pelo gráfico que o desempenho do algoritmo BBS é tão bom quanto as demais técnicas, sendo superior ao DBS. O algoritmo DBS tem o pior tempo, pois, o aumento da dimensionalidade faz com com que a leitura de cada ponto pelo algoritmo DBS seja custosa. A execução do conjunto original no algoritmo DBScan demorou quase 8 horas para encontrar todos os agrupamentos. Com isso, podemos concluir que o algoritmo BBS é rápido e eficiente para a amostragem de dados de grandes bases, seja no número de atributos ou de tuplas, principalmente quando a presença de ruído é elevada. O seu custo é linear no número de elementos N e no número de dimensões E, assim como os algoritmo GBS e US. O algoritmo DBS tem seu custo elevado, já que, quanto maior o número de estimadores de kernel utilizado e dimensões, maior a quantidade de processamento utilizado.

4.4 Considerações Finais

Este capítulo apresentou o algoritmo BBS - Biased Box Sampling, que realiza amostragem de maneira eficiente e com excelentes resultados em conjuntos de dados desbalanceados e com alta presença de ruído, isto é, conjuntos que possuam grupos de elementos em quantidade diferentes. Os resultados obtidos com os experimentos mostram o bom desempenho do algoritmo BBS, comparado com os algoritmos GBS, DBS e US. As comparações foram realizadas utilizando o conjunto real "Pendigits" e diversas configurações de conjuntos gerados sinteticamente. Os experimentos mostraram também que o algoritmo de amostragem balanceada BBS é escalável quanto ao número de dimensões do conjunto e robusto quanto a quantidade de ruído presente no conjunto, produzindo um resultado eficiente e rápido.

A amostragem de dados é uma tarefa importante, já que, os conjuntos de dados crescem cada vez mais em volume (número de tuplas) e número de dimensões. Além disso, os conjuntos de dados reais apresentam grupos de elementos com densidades diferentes e grande quantidade de elementos considerados como ruído. Assim, ter um algoritmo linear que seja capaz de reduzi-los e elimine o ruído, mas mantenha a qualidade da amostra gerada é de suma importância. Como pode ser visto, o algoritmo BBS cumpre todos os requisitos necessários para um bom algoritmo de amostragem.

Capítulo
5

Mineração de Redes Complexa: algoritmos, leis e padrões

5.1 Considerações Iniciais

As redes complexas estão por toda parte, tais como, as redes sociais, as biológicas e as de colaboração. Encontrar padrões nestas redes é uma tarefa de mineração importante, dado que isto ajuda a descobrir anomalias (exceções) e regiões de interesse.

Tradicionalmente, as redes de tamanho pequeno são analisadas localmente para ajudar a responder questões sobre o comportamento ou as propriedades de um nó em particular na rede. Apesar desses métodos serem bastante expressivos, eles frequentemente falham por não serem escaláveis em redes com milhões de nós e arestas. Além disso, muitas vezes é necessário trabalhar com redes de tamanho grande, já que, como visto no Capítulo 3, as propriedades estruturais tendem a ser diferentes para redes maiores. Com isso, a mineração de redes complexas nos últimos anos mudou o foco para a analise da rede como um todo ao invés da visão antiga centralizada em alguns nós da rede.

O principal objetivo desta tese foi explorar grandes conjuntos de dados por meio

do desenvolvimento de técnicas que permitam a mineração dessas grandes redes. Este objetivo foi mantido nos conjuntos de dados representados como grafos, isto é, as redes complexas.

Baseado na abordagem de redução de dados, foi desenvolvido um método chamado `ShatterPlots` que, permite a distinção de redes complexas entre reais e sintéticas, por meio da remoção aleatória de arestas da rede até esta atingir o maior valor para o diâmetro efetivo, chamado `ShatterPoint`.

Após a exploração da redução de grafos, foi analisado o armazenamento de grafos em sistemas gerenciadores de base de dados relacionais (SGBDR) que são ferramentas importantes e poderosas no armazenamento e na recuperação de grandes volumes de dados de forma eficiente e confiável. O armazenamento foi explorado para a tarefa de recuperação de cliques de tamanho 4 e 5 denominados κ_4 e κ_5, por meio do método `FCR - Fast Clique Retrieval`. A identificação dos cliques de tamanho 4 e 5 é uma tarefa importante, pois permite a identificação de grupos potenciais para a venda de serviços comuns ou recomendação de produtos em sistemas de recomendação. O uso de cliques de tamanho 4 e 5 em uma extensão do coeficiente de clusterização, transforma esta equação em uma equação mais rica, que ajuda em outras tarefas de mineração.

A Seção 5.2 apresenta uma descrição de todas as redes complexas utilizadas nos experimentos realizados com os métodos e algoritmos desenvolvidos. A Seção 5.3 apresenta o algoritmo `ShatterPlots` juntamento com os experimentos realizados e a Seção 5.4 apresenta o trabalho relativo à recuperação de κ_4 e κ_5 com a abordagem `FCR - Fast Clique Retrieval`.

5.2 As redes complexas utilizadas

As redes utilizadas nesta tese estão descritas na Tabela 5.1. As colunas são respectivamente o número, o símbolo que representam as redes, o número de nós e arestas e a descrição da redes, este último também apresenta o nome e a referência da rede.

As redes sintéticas foram geradas usando os algoritmos descritos nos artigos que as propões. Para o modelo BR foi usado o modelo apresentado em (Ravasz et al., 2002)

Número	Símbolo	Nó	Arestas	Descrição
Redes Sociais Online				
1	◇	75.877	405.739	Epinions network (Richardson et al., 2003)
2	◇	33.696	180.811	Enron email net (Klimt e Yang, 2004)
Colaboração Acadêmica (co-autor)				
3	∗	21.363	91.286	Arxiv `cond-mat` (Leskovec et al., 2007a)
4	∗	11.204	117.619	Arxiv `hep-ph` (Leskovec et al., 2007a)
Redes de informação (citações)				
5	x	34.401	420.784	Arxiv `hep-th` citations (Gehrke et al., 2003)
6	x	32.384	315.713	Blog citation (Leskovec et al., 2007b)
7	x	238.305	297.338	Patente network (14 anos) (Leskovec et al., 2007a)
Redes Web				
7	⊙	319.717	1.542.940	Stanford – UC Berkeley
8	⊙	855.802	4.291.352	Google web graph (Google, 2002)
Amazon rede de compras				
9	⊡	473.315	3.505.519	Snapshot 2 (Clauset et al., 2004)
Bipartidos (autores-artigos)				
10	+	54.498	131.123	Arxiv `astro-ph` (Leskovec et al., 2007a)
Internet				
11	■	13.579	37.448	AS Oregon (Leskovec et al., 2007a)
12	■	26.389	52.861	CAIDA AS
11	■	22.963	48.436	AS de M. Newman
13	■	62.561	147.878	Gnutella, 31 Mar 2000 (Ripeanu et al., 2002)
Reticulados - Grids				
14	♦	4.941	6.594	Power Grid western US (Watts e Strogatz, 1998)
redes sintéticas				
15	▽	2D - Synthetic Grid		
16	▼	*Erdös-Rényi* random graphs (Erdos e Renyi, 1960)		
17	▽	BR - Barabasi Hierarchical Model (Ravasz et al., 2002)		
18	▽	SW - SmallWorld (Watts e Strogatz, 1998)		
19	▽	PA - Preferential Attachment (Barabasi e Albert, 1999)		

Tabela 5.1: Conjuntos de dados considerado nesta tese. O símbolo e o número em cada linha são utilizados nas figuras das próximas seções.

com 3, 4 e 7 níveis para cada uma das três redes. Para o modelo *Erdös-Rényi* foi usado o modelo descrito em (Durrett, 2007) utilizando o modelo $G(\mathcal{V}, \mathcal{E})$ O número de nós e arestas usados foram $|\mathcal{V}| = 1k, 2k, 10k, 100k$ e $|\mathcal{E}| = 5k, 14k, 50k, 400k$ respectivamente.

As redes *Preferential Attachment* (PA) foram criadas utilizando o modelo descrito em (Barabasi e Albert, 1999) usando 3k e 4k nós e 3 e 5 como parâmetro para grau. Nas redes *Small World* (SW), o gerador segue o modelo apresentado em (Watts e Strogatz, 1998) usando como parâmetros: número de nós $|\mathcal{V}| = 5k, 8k, 8k$, grau igual $d = 5, 6, 3$ e

probabilidade de rescrita $p = 0.4, 0.9, 0.5$. Os grids-2D foram criados com os tamanhos 30x30, 50x50 e 1000x1000 sem a ligação das extremidades. Todos as redes foram geradas como não direcionadas.

As redes reais foram coletadas de diversas fontes. A seguir será detalhado a formação de cada uma das redes reais:

Redes Sociais Online: A classe das redes sociais online incluem a rede Epinions, que expressa a relação interpessoal "quem confia em quem" (http:\www.epinions.com), e a rede Email Enron que é um conjunto de mensagens de e-mail entre dois endereços.

Colaboração Acadêmica: Baseado no base de dados arXiv sendo os nós pessoas e as arestas a conexão entre pessoas que publicaram pelo menos um artigo juntas.

Redes de informação (citações): As redes complexas de citações foram baseadas em quem cita quem. A rede hep-th (high energy physics theory) é baseada nos dados do KDD Cup 2003. A rede Blog consiste de citações entre Blogs. A rede Patente considera a lista de patente americana mantida pela National Bureau of Economic Research. Os dados consiste em 37 anos (Janeiro de 1975 à Dezembro de 1999).

Redes Web: A classe de redes baseadas na Web consiste de redes complexas, na qual, nós representam páginas Web e as arestas representam os links entre as páginas.

Amazon rede de compras: A rede representa a informação sobre compras em lojas, nos quais os nós representam itens de compra e as aresta representam os itens frequentemente comprados juntos.

Bipartidos (autores-artigos) Baseado no base de dados arXiv foi construído uma rede complexa. Há um nó para cada artigo e um nó para cada pessoa que é autora de pelo menos um artigo na base arXiv, e uma aresta conectando pessoas para artigos da sua autoria.

Internet A rede de roteadores que compreende a Internet pode ser organizado em redes chamadas Sistemas Autônomos (AS). Cada AS troca informações com alguns

vizinhos (pares). A rede Gnutella consiste de vários sistemas autônomos obtidos de diferentes fontes como a rede de troca de informação ponto-a-ponto.

5.3 ShatterPlots

O objetivo do algoritmo `ShatterPlots` é encontrar padrões em redes complexas, reais e sintéticas, por meio da remoção de arestas. Intuitivamente, dada uma rede $\mathcal{G}$ a cada passo i, Q_t arestas são escolhidas aleatoriamente e removidas da rede. A cada remoção um conjunto de medidas como diâmetro efetivo, autovalor da matriz de adjacência, número de nós e arestas, etc são coletados da rede e o processo continua até não haver mais nenhuma aresta na rede. Os nós que se tornam isolados, isto é, não possuem nenhuma aresta conectada a eles, são removidos da rede. Esse processo, que corresponde ao algoritmo `ShatterPlots`, é apresentado graficamente na Figura 5.1.

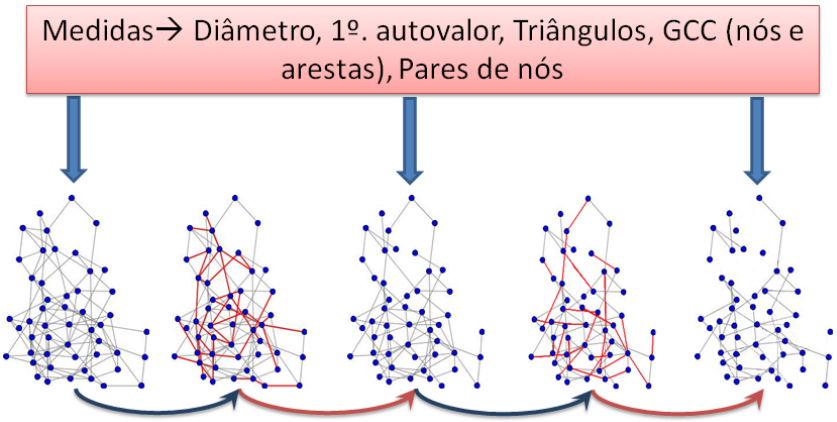


Figura 5.1: Exemplificação gráfica do algoritmo `ShatterPlots`. As arestas em vermelho representam aquelas a serem removidas.

O processo realizado pelo `ShatterPlots` simula um conjunto de falhas ocorridas sistematicamente na comunicação dos nós da redes, esse processo é usualmente medido por meio do diâmetro efetivo da rede que se altera. Inicialmente o diâmetro cresce conforme as arestas são removidas, já que, nós que eram conexos se tornam desconectados. Entretanto, após um certo número de arestas serem removidas, a rede se quebra por completo e, com isso, o diâmetro diminui formando um pico no seu maior valor. Este pico é conhecido como

"Fase de transição" ou "Ponto Crítico" e como visto no Capítulo 3 é apresentado pelas redes (*Erdös-Rényi*). No algoritmo `ShatterPlots` esse ponto é chamado de `ShatterPoint`.

O objetivo do algoritmo `ShatterPlots` é encontrar padrões que permitam distinguir entre redes reais e sintéticas no `ShatterPoint`. Assim, este problema pode ser formulado da seguinte maneira:

Problema 1. *Dada uma rede complexa grande e esparsa, verificar se ela é real ou sintética.*

Com o problema principal definido, surgem novas questões que devem ser investigadas. Estas questões são chamadas de filosóficas (PhQ) e são apresentadas a seguir.

PhQ1 As redes reais possuem `ShatterPoint`?

PhQ2 Se sim, nas redes reais o `ShatterPoint` ocorre rápido ou não?

Como os experimentos apresentados na Seção 5.3.2 irão mostrar, as redes reais apresentam `ShatterPoint`. Além disso, ao contrário de fenômenos naturais, como avalanches, placas tectônicas e outros, que estão no ponto crítico, as redes complexas estão bem distante deste ponto, isto é, é necessário a remoção de muitas arestas para uma rede alcançar o `ShatterPoint`.

A Figura 5.2 apresenta medidas como diâmetro efetivo, número de pares alcançáveis e número de nós pertencentes à maior componente conexa (GCC) em três redes reais durante a remoção aleatória de arestas pelo algoritmo `ShatterPlots`. Todas as medidas são em relação a quantidade de arestas existente na rede. Como pode ser observado, apenas o diâmetro mostra um pico nos gráficos apresentados na parte superior da Figura 5.2. O `ShatterPoint`, definido como sendo o ponto em que um pico se forma no diâmetro, é marcado por uma linha em todos os gráficos apresentados na Figura 5.2. As outras medidas também possuem uma fase de transição, contudo, elas não aparecem de maneira clara nos gráficos.

No `ShatterPoint`, a rede é pouco conectada, isto é, qualquer aresta removida tornará a rede desconexa. Com isso podemos definir que:

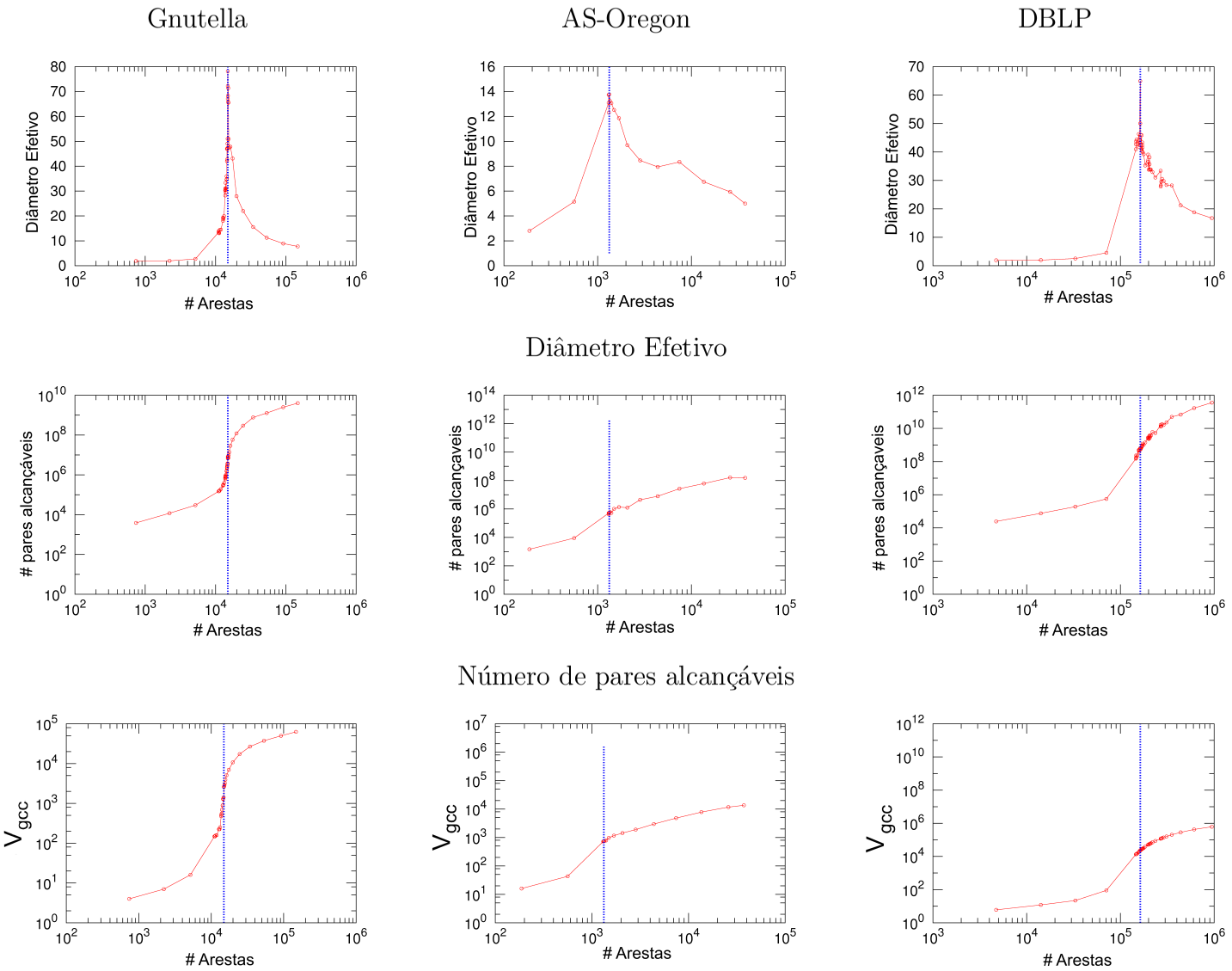


Figura 5.2: Medidas estruturais da rede feitas durante a remoção aleatória de arestas. As redes apresentam `ShatterPoint` em todas as medidas mas somente o diâmetro apresenta um pico.

Definição 3. *O* `ShatterPoint` *é o estágio em que o diâmetro efetivo tem o maior valor.*

Definição 4. *Para as redes* Erdös-Rényi *o* `ShatterPoint` *definido coincide com a fase de transição definida na literatura.*

Esta ultima definição é importante, pois as redes *Erdös-Rényi* serão usadas como validação para os padrões encontrados.

Todas as medidas no `ShatterPoint` são representadas pelas letras tendo *sp* subscrito. Assim, tem-se $\mathcal{V}_{gccsp}, \mathcal{E}_{gccsp}, Npairs_{sp}, \mathcal{V}_{sp}, \mathcal{E}_{sp}, D_{sp}, d_{maxsp}(v_i), \Delta_s$ e $\lambda_{1,sp}$ que são nós e arestas da componente conexa, número de pares alcançáveis, nós e arestas da rede completa, diâmetro, maior grau, número de triângulos e primeiro autovalor, todos obtidos no `ShatterPoint`.

Tendo respondido as questões filosóficas, surgem outras questões, chamadas de exploratórias (ExQ), são:

ExQ1 Qual a proporção $|\mathcal{E}_{sp}| \,/\, |\mathcal{E}|$ (isto é, a fração de arestas no `ShatterPoint`)? Esta fração, chamada "*Edges shattering ratio*", depende do tamanho da rede ou não?

ExQ2 O que representa a relação $|\mathcal{V}_{sp}|$ vs. $|\mathcal{V}|$?

ExQ3 Redes sintéticas tem o mesmo comportamento no `ShatterPoint`? Ou elas seguem leis diferentes?

ExQ4 O que pode ser concluído sobre $|\mathcal{E}_{sp}|$ vs $|\mathcal{V}_{sp}|$? E sobre a maior componente conexa no `ShatterPoint`?

Todas essas questões serão respondidas na Seção 5.3.2. Intuitivamente, o algoritmo `ShatterPlots` mede no `ShatterPoint` se com a remoção de uma grande quantidade de arestas a rede tende a se tornar uma longa cadeia de arestas ou um conjunto de nós com grande quantidade de nós de grau um conectados a eles e quão rápido a rede chega neste estágio. A rapidez para chegar no `ShatterPlots`, isto é, a quantidade de arestas removidas, e o formato da rede no `ShatterPoint` permite a extração de padrões valiosos que permite separar as redes em reais e sintéticas.

5.3.1 Descrição do algoritmo `ShatterPlots`

Nesta subseção é apresentado o algoritmo `ShatterPlots`. Contudo, ao invés do algoritmo começar com a rede complexa e remover arestas aleatoriamente, ele começa com uma rede vazia e insere arestas aleatoriamente. O Algoritmo 5.3.1 apresenta os detalhes.

Algoritmo 5.3.1 `ShatterPlots` Adaptativo

Entrada: Rede $\mathcal{G}(\mathcal{V}, \mathcal{E})$
Saída: `ShatterPoint` e as medidas neste ponto

1: Embaralhe a lista de aresta $|\mathcal{E}|$
2: Crie a rede temporária $H(\mathcal{V},\emptyset)$, com $\mathcal{V}$ nós
3: $\epsilon = 0.005$ ou $\epsilon = 1/\lambda_1$
4: $t = 0$
5: $Q_t = \epsilon * |\mathcal{E}|$
6: **enquanto** $H\ != \mathcal{G}$ **faça**
7: Insira Q_t arestas aleatoriamente em H
8: $t = t + 1$
9: Medir propriedades estruturais de H (diâmetro, componente conexa, λ_1, etc.)
10: D_t = diâmetro efetivo H
11: **se** $t > 1$ **então**
12: **se** $D_t - D_{t-1} \geq 1$ **então**
13: $Q_t = Q_{t-1}/2$
14: **senão se** $D_t - D_{t-1} \leq -1$ **então**
15: $Q_t = 2 * Q_{t-1}$
16: **fim-se**
17: **senão se** $\epsilon = 1/\lambda_1$ **então**
18: $Q_t = 0.005 * |\mathcal{E}|$
19: **fim-se**
20: **fim-enquanto**

A ideia é embaralhar a lista de aresta da rede $\mathcal{G}$ e construir o grafo temporário H adicionando a cada iteração uma quantidade Q_t de arestas aleatoriamente. Após cada inserção, medidas estruturais são feitas na rede, como diâmetro, número de pares alcançáveis, números de triângulos, primeiro autovalor da matriz de adjacência e número de nós e arestas da maior componente conexa. Este processo é repetido até a rede estar completa, isto é, contendo todos os nós e arestas da lista de aresta inicial.

Idealmente as propriedades da rede complexa seriam computadas após a inserção de cada aresta. Contudo, esta abordagem seria muito demorada, por isso um conjunto de arestas é inserido a cada iteração. A questão que surge é, qual é a quantidade apropriada de arestas para cada inserção de maneira que o `ShatterPoint` não seja perdido? A

resposta para esta questão é um método adaptativo. O método começa com um pequeno número de arestas e se não há uma mudança maior que um *threshold* no diâmetro a quantidade de arestas a ser inserida aumenta. De modo oposto, a quantidade de arestas diminui se o diâmetro começa a crescer de modo a alcançar o `ShatterPoint`. O mesmo processo poderia ser aplicado de maneira inversa, isto é, ao invés de inserir arestas, elas seriam removidas.

Escalabilidade: o Algoritmo `ShatterPlots` Adaptativo é escalável quanto ao número total de arestas da rede $|\mathcal{E}|$. Assim, o método é capaz de processar redes complexas grandes. O algoritmo é ainda mais rápido, até 8 vezes, se o padrão *Eigenvalue* , apresentado na seção 5.3.2, for utilizado como valor inicial para Q_t.

Primeiro, assume-se que a inserção de uma aresta é uma operação de tempo constante. Isto é verdade para a maioria das implementações de redes complexas. Em algumas implementações, ela pode ser logarítmica/linear na média do grau do nó, mas como as redes reais são esparsas, isto é praticamente constante. Segundo, por meio da função de aproximação de vizinhança usada no algoritmo ANF, o diâmetro efetivo é calculado em tempo linear $O(E)$ no número de arestas. O diâmetro efetivo é utilizado por ser mais robusto quanto a medida entre pares de nós em uma rede (Seção 3.3.3).

Contudo, isto não resolve o problema imediatamente: se uma implementação simplista for usada e o algoritmo inserir um número constante de arestas a cada passo, então o algoritmo será quadrático no número de arestas $O(E^2)$ ($O(E)$ de iterações do `ShatterPlots`, e o fator de $O(E)$ para cada execução do algoritmo ANF).

O algoritmo `ShatterPlots` é um método adaptativo, pois ele ajusta o número de arestas inseridas na rede. Este ajuste faz com que o número de arestas inseridas diminua se o diâmetro aumentar mais que um certo *threshold* e aumente se o diâmetro se manteve estável. Isto torna constante o número de iterações do algoritmo, fazendo com que ele seja escalável quanto ao número de arestas.

Há duas versões do algoritmo `ShatterPlots`. A primeira chamada `Proportional ShatterPlots` que tem o valor inicial Q_t igual a $\epsilon = 0.005$ e a outra versão chamada de `Eigenvalue ShatterPlots` usa $1/\lambda_1$ como valor inicial para o ϵ no Q_t. Segundo

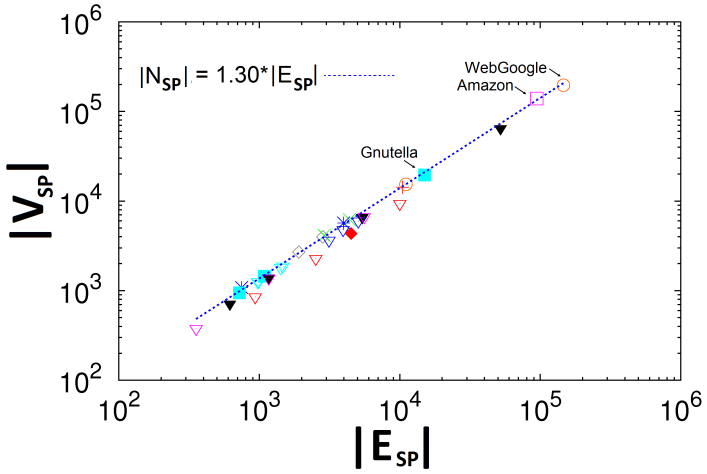
o padrão *Eigenvalue* , apresentado na seção 5.3.2, todas as redes estão acima da linha $\mathcal{E}_{sp}/\mathcal{E} = 1/\lambda_1$, então $\mathcal{E}_{sp}$ é maior que $1/\lambda_1 * \mathcal{E}$ no `ShatterPoint`. Assim, o valor inicial ϵ pode ser $1/\lambda_1$. Para `Eigenvalue ShatterPlots` nenhuma das redes da extensiva coleção utilizada perdeu o `ShatterPoint`. Entretanto, no caso do `Eigenvalue ShatterPlots` perder o `ShatterPoint`, uma solução fácil seria a aplicação do algoritmo `Proportional ShatterPlots` entre 0 e valor $Q_t = 1/\lambda_1$. Na seção escalabilidade (5.3.3) será apresentado os tempos das duas versões do algoritmo `ShatterPlots`.

5.3.2 Experimentos

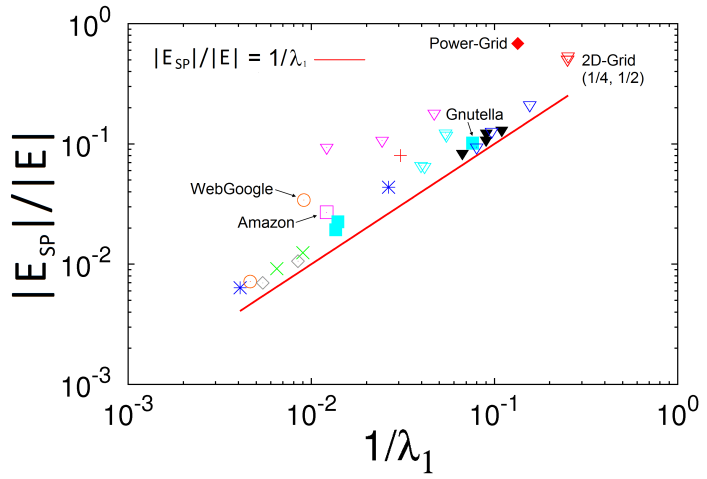
As Figuras 5.3 e 5.4 mostram os gráficos das propriedades estruturais medidas no `ShatterPoint`. Os eixos estão em escala linear-linear na figura (d) e log-log em (a), (b), (c), (e) e (f). As linhas teóricas/esperadas estão presentes (todas com coeficiente de correlação acima de 0.98) quando o gráfico parece apresentar uma forte correlação. As linhas em azul representam os resultados obtidos e as linhas vermelhas o resultado teórico esperado. Cada experimento é a média de 10 execuções.

Como pode ser visto na Figura 5.3 (a), todas as redes testadas apresentaram `ShatterPoint`, sendo que a razão nó-aresta para todas elas no `ShatterPoint`, $\mathcal{V}_{sp}/\mathcal{E}_{sp}$, segue uma linha com o ângulo sendo 1.30. Isto significa que, no `ShatterPoint`, o número de nós $\mathcal{V}_{sp}$ é cerca de 30% maior que $\mathcal{E}_{sp}$. Nesta relação os nós isolados não são considerados, o que demonstra que a rede neste ponto é composta por diversas componentes conexas pequenas. Esta observação responde as questões PhQ2 e parte das questões ExQ3 e ExQ4.

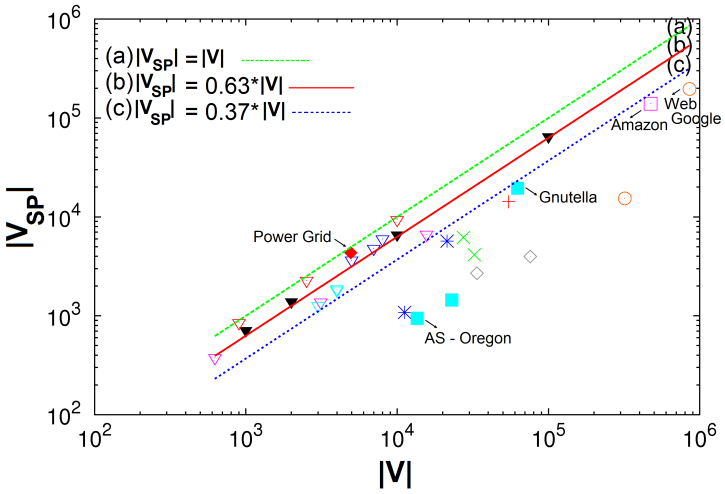
Foi observado também que o `ShatterPlots` aplicado as redes *Erdös-Rényi* levam o `ShatterPoint` a ser a "Fase de transição" predita na teoria para essas redes. Em todas as redes *Erdös-Rényi* o valor de $\mathcal{E}_{sp}$ no `ShatterPoint` satisfaz a condição $\mathcal{E}_{sp} = \mathcal{V}/2$ e $\mathcal{V}_{sp} = \mathcal{V} * (1 - 1/e)$, que é exatamente a condição de fase de transição apresentada na Seção 3.4.



(a) $|\mathcal{V}_{sp}|$ vs. $|\mathcal{E}_{sp}|$

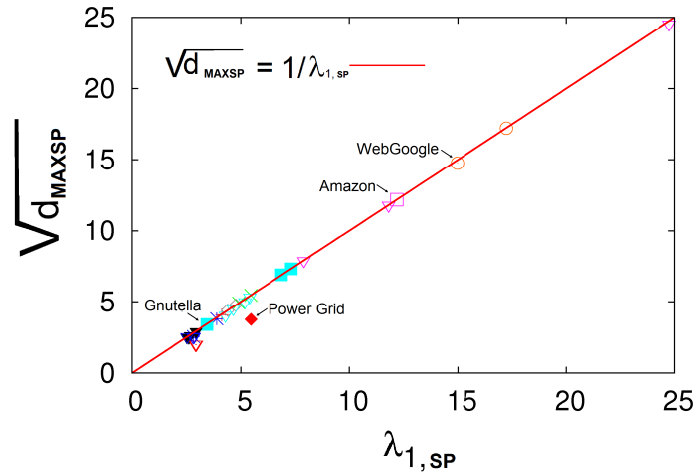


(b) $\mathcal{E}_{sp}/\mathcal{E}$ vs. $1/\lambda_1$

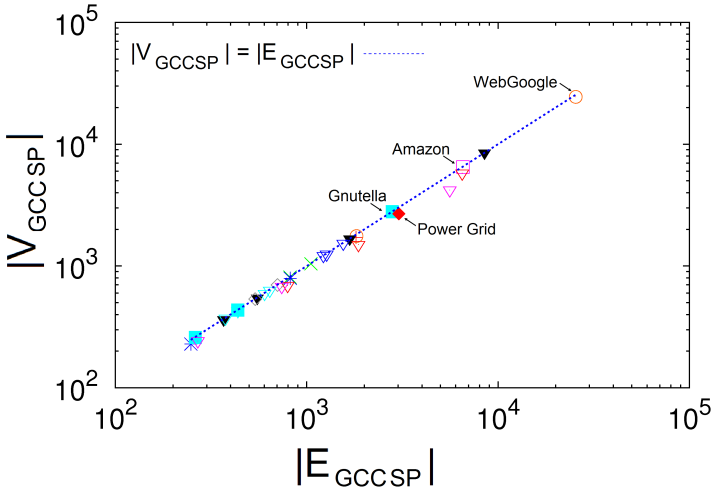


(c) $|\mathcal{V}_{sp}|$ vs. $|\mathcal{V}|$

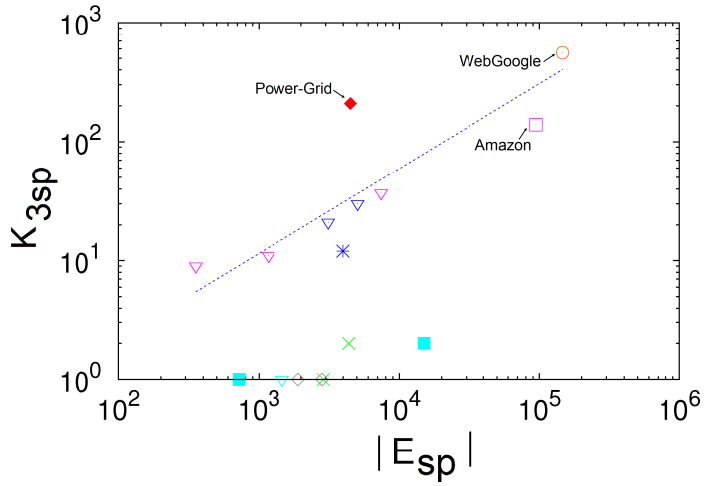
Figura 5.3: Propriedades estruturais no (`ShatterPoint`). As redes sintéticas são representadas em triângulos; *Erdös-Rényi* em triângulos pretos. (a)número de nós não isolados ($\mathcal{V}_{sp}$) versus número atual de arestas $\mathcal{E}_{sp}$ no `ShatterPoint` (padrão *30-per-cent*); (b) número de arestas atuais $\mathcal{E}_{sp}$ sobre o número de arestas total $\mathcal{E}$ versus um sobre λ_1, primeiro autovalor da rede original (padrão *Eigenvalue* pattern); Amazon esta desviando da linha bem como as redes sintéticas BR, PA e 2D-grid. (c) número de nós sobreviventes, isto é, não isolados $\mathcal{V}_{sp}$ versus número total de nós da rede original $\mathcal{V}$ (padrão *NodeShatteringRatio*) 2D-grid estão a cima da linha do *Erdös-Rényi*; SW estão juntos com o *Erdös-Rényi*, BR e PA estão acima da linha "c" e abaixo da linha "b"



(d) $\sqrt{d_{sp}}$ vs. $\lambda_{1,sp}$



(e) $\mathcal{V}_{gccsp}$ vs. $\mathcal{E}_{gccsp}$



(f) $\kappa_{3_{sp}}$ vs. $\mathcal{E}_{sp}$

Figura 5.4: (d)a raiz quadrada do maior grau no `ShatterPoint` $d_{maxsp}(v_i)$ versus $\lambda_{1,sp}$ no `ShatterPoint` (padrão *Root-degree*); (e) número de nós N_{gccsp} versus número de arestas E_{gccsp} da GCC no `ShatterPoint` (padrão *TreeGCC*); (f) número de Triângulos $\kappa_{3_{sp}}$ versus número de arestas $\mathcal{E}_{sp}$ (padrão *TriangleRatio*). A rede Power Grid tem um número desproporcional de triângulos. Apenas as redes com um ou mais triângulos são apresentadas nesse gráfico.

Padrão *30-per-cent*

Padrão 1 (*30-per-cent*)**.** *O* `ShatterPoint` *de todas as redes ocorre quando* $\mathcal{V}_{sp}$ *é 30% maior que* $\mathcal{E}_{sp}$*.*

Justificativa Teórica Para as redes *Erdös-Rényi* o padrão *30-per-cent* pode ser provado:

Nas redes *Erdös-Rényi* na fase de transição tem se que:

$$\mathcal{E}_{sp} = 1/2 * \mathcal{V}_{sp} * e/(e-1) = 0.79 * \mathcal{V}_{sp} \tag{5.1}$$

na qual $e = 2.718$. Identicamente, $\mathcal{V}_{sp} = 1.26\mathcal{E}_{sp}$, o que é bem próximo aos 30%.

Prova: no `ShatterPoint` tem-se $\mathcal{V}_{sp} = \mathcal{V} * (1 - 1/e)$ e $\mathcal{E}_{sp} = \mathcal{V}/2$. Substituindo $\mathcal{V}$, completa-se a prova. **CQD**

Discussão: é uma surpresa todas as redes seguam este padrão, pois como será visto a seguir, para todos os outros padrões encontrados no `ShatterPoint` as redes reais se diferem bastante das redes *Erdös-Rényi*.

Exceções: este é um dos poucos padrões que parece ser universal. Assim, este padrão não ajuda a encontrar exceções nas redes, bem como distingui-las entre reais e sintéticas. Entretanto, os padrões apresentados a seguir o fazem.

Padrão *Eigenvalue*

Seja $\mathcal{E}_{sp}/\mathcal{E}$ definido como *Edges Shattering Ratio*, que é a fração de arestas que a rede contém no `ShatterPoint`. A Figura 5.3 (b) mostra que a porcentagem de arestas mantidas na rede no `ShatterPoint` tem uma forte correlação com $1/\lambda_1$. Esta observação responde a questão ExQ1. De fato, este padrão mostra que o *Edge Shattering Ratio* não depende do tamanho da rede, mas sim do primeiro autovalor da matriz de adjacência da rede. Então tem-se:

Padrão 2 (*Eigenvalue*)**.** *A fração de arestas*

$$\mathcal{E}_{sp}/\mathcal{E} = ct * 1/\lambda_1, \tag{5.2}$$

na qual ct é uma constante e λ_1 é o primeiro autovalor da matriz de adjacência da rede.

Justificativa Teórica: o *Edge Shattering Ratio* é a porcentagem de arestas que ainda cria um grande componente conexa. $1/\lambda_1$ é conhecido como o "Epidemic Treshold" do modelo SIS (Susceptível-Infectado-Susceptível) apresentado na Seção 3.3.5.

Discussão: a razão $\mathcal{E}_{sp}/\mathcal{E}$ é também conhecida como *Percolation Threshold*, apresentado na Seção 3.4, é a fase de transição das redes sintéticas 2D-grids. Para os 2D-grids o *Percolation Threshold* das arestas é bem definido como sendo 0.5 (Kesten, 1980), sendo assim é esperado que as redes sintéticas 2D-grids sejam excessões neste padrão.

Excessões: neste padrão pode-se ver que algumas redes sintéticas como PA, BR e 2D-grids desviam da tendência formada pelas redes reais.

Padrão *NodeShatteringRatio*

A Figura 5.3 (c) mostra o padrão *Node Shattering Ration* que é a relação entre nós no `ShatterPoint` $\mathcal{V}_{sp}$ versus o número de nós da rede original $\mathcal{V}$. Três linhas foram desenhadas na Figura 5.3 (c). A linha (a) - pontilhada - é exatamente $\mathcal{V} = \mathcal{V}_{sp}$ que é o limite máximo, a linha (b) - linha sólida - é a linha teórica das rede *Erdös-Rényi* e a linha (c) - linha tracejada - é $\mathcal{V}_{sp} = 0.37 * \mathcal{V}$ é a linha em que todas as redes reais estão abaixo dela. Como pode se ver, este padrão responde as questões ExQ2 e ExQ3.

Padrão 3. *As redes sintética são próximas a* $\mathcal{V}_{sp} = 0.63 * \mathcal{V}$*.*

Justificativa teórica: Como já visto, todas as redes *Erdös-Rényi* na fase de transição têm

$$\mathcal{V}_{sp} = \mathcal{V} * (1 - 1/e) \tag{5.3}$$

e $(1 - 1/e) = 0.63$, onde $e = 2.718$.

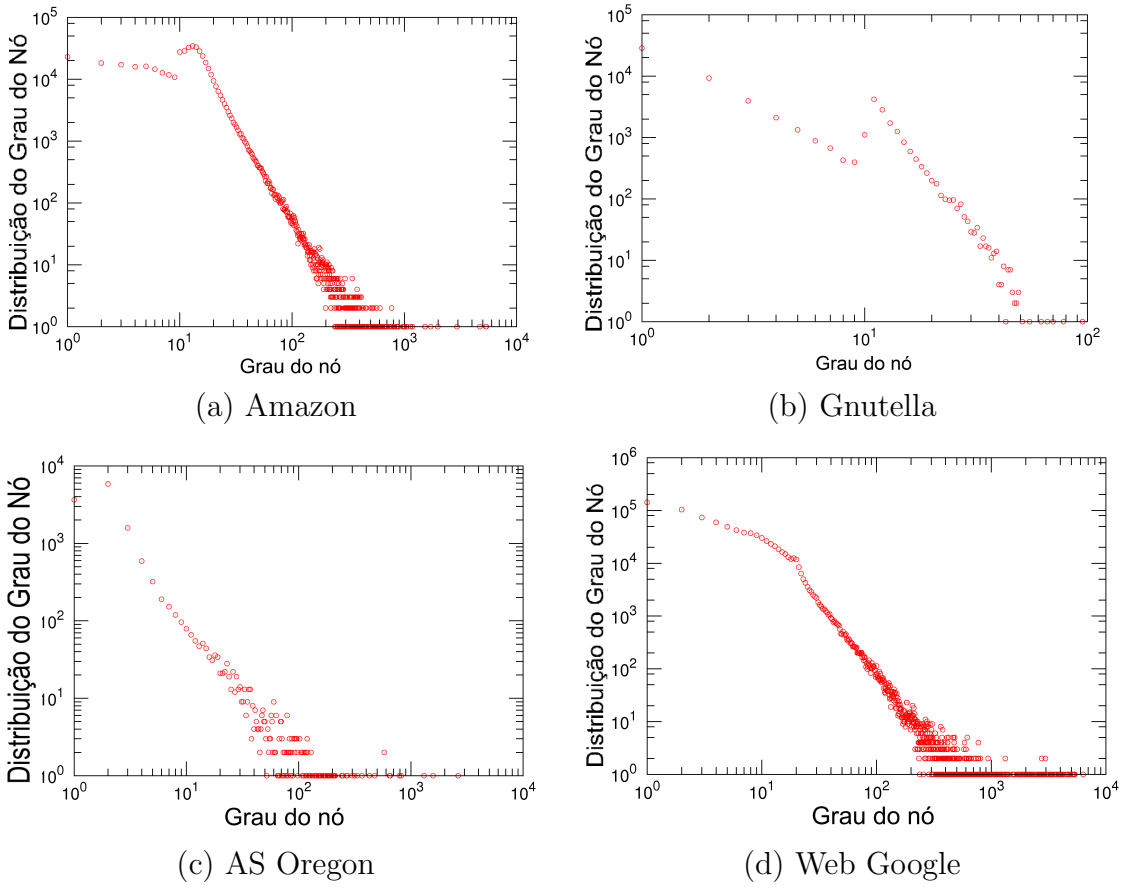
Discussão: a explicação é que a maioria das redes reais tem muitos nós com grau igual a um, $d(v_i) = 1$, o que faz as redes terem uma distribuição do grau do nó que segue uma lei de potência, e estes nós tem uma alta probabilidade de estarem isolados no `ShatterPoint`. Um exemplo é a rede AS-Oregon, na qual a distribuição do grau é apresentada na Figura 5.5 (c). Por outro lado, redes como os 2D-grids têm a maioria dos nós com o grau igual a quatro, e as redes *Erdös-Rényi* tem pouca variação na distribuição do grau do seus nós, isto é, os nós têm o grau próximo a média do grau. Todos essas redes têm poucos nós isolados quando elas estão no `ShatterPoint`. As redes 2D-grids possuem menos nós isolados que as redes *Erdös-Rényi*. Este é o motivo que faz as redes 2D-grids (triângulos laranjas) estarem acima da linha das redes *Erdös-Rényi* (triângulos pretos), isto é, eles têm no `ShatterPoint` muito mais nós conexos que as redes reais. Contudo, algumas redes, como Amazon e Gnutella, são mascaradas, isto significa que, elas têm uma distribuição do grau do nó que não segue fielmente uma lei de potência, como mostra as Figuras 5.5 (a) e (b), respectivamente. Para essas redes pode ser visto que o `ShatterPoint` ocorre mais rápido que em outras redes como AS-Oregon (Figura 5.3 (c)).

Excessões: o padrão *NodeShatteringRatio* é provavelmente o melhor detector de redes reais, sintéticas e mascaradas, pelo menos no conjunto de redes estudadas. Note se que, todos as redes sintéticas estão perto da linha "b" e acima da linha "c" - $\mathcal{V}_{sp} = 0.37 * \mathcal{V}$ - na Figura 5.3(c).

Padrão *Root-degree*

A Figura 5.4 (d) apresenta o primeiro autovalor da matriz de adjacência da rede no `ShatterPoint`, chamado de $\lambda_{1,sp}$, versus $\sqrt{d_{maxsp}(v_i)}$, que é a raiz quadrada do maior grau do nó da rede no `ShatterPoint`. Nesta figura também é mostrado a linha de tendencia que segue a equação $\lambda_{1,sp} = \sqrt{d_{maxsp}(v_i)}$.

Padrão 4. *Todas as redes seguem* $\lambda_{1,sp} \geq \sqrt{d_{maxsp}(v_i)}$.



(a) Amazon (b) Gnutella

(c) AS Oregon (d) Web Google

Figura 5.5: Distribuição do grau do nós das redes: (a) Amazon, (b) Gnutella, (c) AS Oregon e (d) Web Google

Como apresentado na seção 3.3.3, os autovalores comprovam a tendência apresentada pelas redes estudadas.

Justificativas Teóricas: todas as redes que tendem a estar pouco conectadas e se assemelham a uma árvore tem-se: $\lambda_1(G) = \sqrt{d_{max}(v_i)}$

Discussão: o teorema 2 apresentado na Seção 3.3.3 vale para qualquer rede, incluindo aquelas no `ShatterPoint`.

Especificamente para as redes *Erdös-Rényi* (triângulos pretos) tem se que o autovalor $\lambda_{1,sp}$ é constante e independente do número de nós $\mathcal{V}$ da rede.

Padrão 5. *O valor $\lambda_{1,sp}$ para as redes* Erdös-Rényi *estudadas é constante:* ≈ 2.8

A rede Power-Grid esta abaixo da linha que representa a tendência seguida pelas redes na Figura 5.4 (d). Isto significa que no `ShatterPoint` ela é uma rede bem conectada, já que, quanto mais distante o valor λ_1 for da $\sqrt{d_{max}(v_i)}$ mais bem conectada uma rede será. Na Figura 5.6 é mostrado o nó de maior grau da rede Power-Grid na rede original (Figura 5.6 (a)) e no `ShatterPoint` (Figura 5.6 (b)). Pode-se ver que este nó ainda continua com alguns triângulos e muitas conexões inclusive no `ShatterPoint`, onde espera-se nós pouco conexos. Por ser um processo aleatório, é esperado que os nós de maior grau sejam os nós que terão mais arestas removidas no `ShatterPlots`, já que, eles são os nós com maior número de arestas. Além disso, olhando-se o padrão *NodeShatteringRatio* a rede Power Grid esta muito perto da linha "a" da Figura 5.3(c), o que significa que $\mathcal{V}_{sp}$ é próximo $\mathcal{V}$.

Padrão *TreeGCC*

Na Figura 5.4 (e) é mostrado que a maioria das redes no `ShatterPoint` tem aproximadamente a mesma quantidade de arestas ($|E_{gccsp}|$) e nós ($|N_{gccsp}|$) na maior componente conexa (GCC). Assim, esse padrão responde a segunda parte da ExQ4.

Padrão 6. *A grande componente conexa de todas as redes no* `ShatterPoint` *tem* $E_{gccsp} \cong N_{gccsp}$.

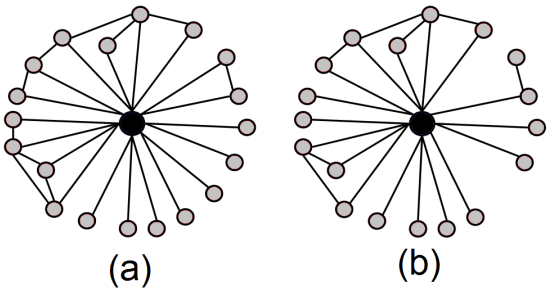


Figura 5.6: Nó de maior grau da rede Power-Grid: (a) Rede original ($|\mathcal{V}| = 20$ e $|\mathcal{E}| = 29$) e (b) `ShatterPoint` ($|\mathcal{V}_{sp}| = 19$ e $|\mathcal{E}_{sp}| = 25$)

Discussão: sabe-se que acima do `ShatterPoint` a rede é bem conectada e abaixo ela é completamente desconectada. Assim, no `ShatterPoint` espera-se que a rede seja pouco conectada. Isto significa que, a remoção de uma pequena quantidade de arestas de uma rede no `ShatterPoint` fará a rede se tornar completamente desconectada. Por este padrão pode se ver que a GCC no `ShatterPoint` parece-se com um árvore. Entretanto, algumas redes como o Power Grid estão bem abaixo da linha da Figura 5.4 (e), mostrando-se mais conectadas que uma árvore. A Figura 5.6, que mostra o nó com maior grau da rede Power Grid, é um exemplo que os nós desta rede ainda permanece bem conectados mesmo no `ShatterPoint`. A diferença entre esse padrão e o *30-per-cent* é que neste os nós e arestas fora da GCC são ignorados enquanto no padrão *30-per-cent* eles são incluídos. Note-se que em nenhum padrão os nós isolados são considerados.

Padrão *TriangleRatio*

Na Figura 5.4 (f) é mostrado que no `ShatterPoint` a maioria das redes têm poucos triângulos. As redes que apresentam uma quantidade nula de triângulos no `ShatterPoint` não foram incluídas no gráfico já que os eixos são logarítmicos.

Padrão 7. *As redes no* `ShatterPoint` *tem poucos ou nenhum triângulo* ($\kappa_{3,sp} \approx 0$).

Excessões: neste padrão a rede Power Grid é a exceção.

Discussão: no `ShatterPoint` espera-se que as redes sejam pouco conectadas, e isto pode ser visto nos padrões *TreeGCC*, que diz que a GCC parece com uma árvore, e no

padrão *Root-degree* , em que é mostrado que o valor do $\lambda_{1,sp}$ é fortemente correlacionado com o maior grau da rede no `ShatterPoint`.

Para a rede Power Grid, foi observado que ela esta abaixo da linha de tendencia dos dados na Figura 5.4 (e), o que significa que ela tem mais arestas do que nós na GCC, isto é, ela é mais conectada que uma árvore. Também ela tem o valor de $\lambda_{1,sp}$ maior que $\sqrt{d_{maxsp}(v_i)}$ como visto na Figura 5.4 (d). Isto significa que, o autovalor dela não esta correlacionado com o maior grau, o que também prova que ela é mais conectada que uma árvore (Figura 5.6 (b)).

Observa-se também, que a relação entre o número inicial de triângulos (κ_3) da rede Power Grid é muito maior que a das outras redes. Por exemplo, inicialmente a rede Power Grid tem $\kappa_3 = 651$ enquanto Web Google tem $\kappa_3 = 13,356,298$, já no `ShatterPoint` a rede Power Grid tem $\kappa_{3sp} = 209$ enquanto a rede Web Google tem $\kappa_{3sp} = 556$.

Como pode ser visto todos os padrões estão interligados e refletem o comportamento das redes no `ShatterPoint`. Os modelos de geração de redes sintéticas testados mostram que ainda existem falhas no cumprimento de todas as propriedades estatísticas das redes complexas reais e, por isso, esta área ainda necessita de mais trabalho. Assim, o uso de alguns modelos de redes complexas sintéticas para o estudo de propriedades das redes complexas pode levar a conclusões errôneas que interfiram no resultado da mineração.

5.3.3 Escalabilidade

O algoritmo `ShatterPlots` é uma ferramenta rápida que necessita apenas de uma leitura na lista de aresta em cada iteração. O número de iterações depende de quão rápido possa-se chegar ao `ShatterPoint`.

A escalabilidade das duas versões do algoritmo `ShatterPlots` é apresentada na Figura 5.7, que apresenta o tempo de execução versus o quantidade inicial de arestas das redes. As redes utilizadas foram as redes sintéticas *Erdös-Rényi*, por se tratar de um experimento controlado. O número de aresta total de cada uma das redes utilizadas é $|\mathcal{E}|$ = 50k, 200k, 300k, 500k, 600k e o número de nós $|\mathcal{V}|$= 10k, 40K, 60k, 80k, 100k,

respectivamente. Os experimentos rodaram em uma Quad Xeon (2.66 GHz), com 8Gb de RAM, rodando Linux (Ubuntu) e os resultados são médias de 10 execuções.

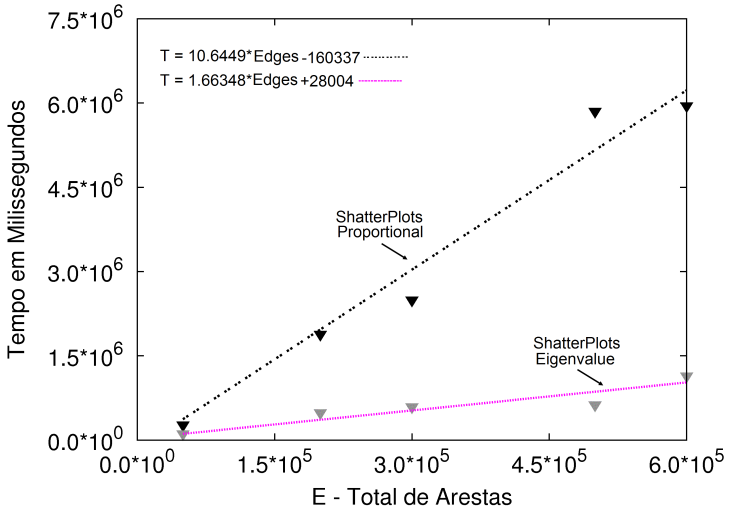


Figura 5.7: Escalabilidade

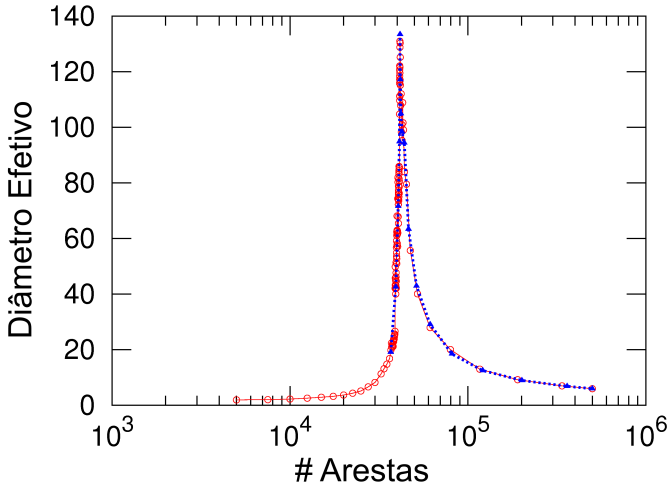


Figura 5.8: `ShatterPlots` da rede *Erdös-Rényi* com 500k arestas usando `Eigenvalue ShatterPlots` (triângulos azuis) e `Proportional ShatterPlots` (círculos vermelhos).

Os mesmos conjuntos de dados foram utilizados em ambos os algoritmos. O algoritmo `Proportional ShatterPlots` é representado pelos triângulos pretos e `Eigenvalue ShatterPlots` pelos triângulos cinzas. As linhas traçadas (tracejada em preto e a sólida em vermelho) demonstram que o algoritmo escala de maneira linear quanto ao tamanho da rede, sendo o `Eigenvalue ShatterPlots` significantemente mais rápido (até 8x) que

o `Proportional ShatterPlots`.

A Figura 5.8 apresenta a execução de uma das redes sintéticas pelo algoritmo `Proportional ShatterPlots` e `Eigenvalue ShatterPlots`. Nota-se que ambos encontraram o mesmo `ShatterPoint`, sendo que o algoritmo `Eigenvalue ShatterPlots` começa com um grande valor para a primeira inserção de arestas diminuindo assim o número de iterações, sendo o `Proportional ShatterPlots` com 129 iterações e o `Eigenvalue ShatterPlots` com 17 iterações.

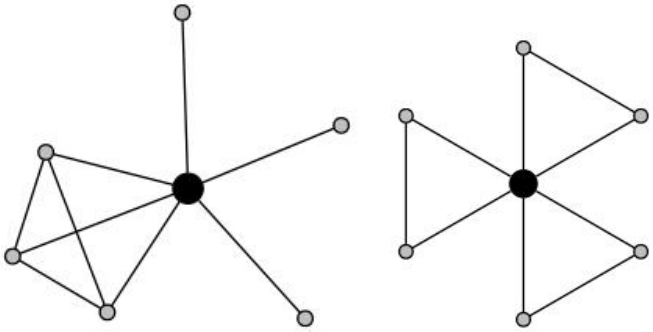
5.4 A técnica `FCR - Fast Clique Retrieval`

Os sistemas gerenciadores de base de dados (SGBD) são ferramentas poderosas para o armazenamento de grande volume de dados e vem sendo usada com sucesso por diversas instituições há mais de quarenta anos. Este sucesso pode ser explicado pela simplicidade do modelo relacional, que armazena os dados e que é constituído por relações, e pelo surgimento de uma linguagem de manipulação simples e eficiente, a SQL (Structured Query Language). Como apresentado no Capitulo 3, dados representado como grafos têm crescido em termos de aplicações e volume de dados. As relações entre tabelas em um banco de dados relacional pode ser vista como um grafo. Com isso, muitas vezes é desejável minerar os dados armazenados em um banco de dados com tarefas de mineração de grafos, sem ter que extrair o grafo do banco de dados e armazená-lo em outra estrutura, isto é, quer se minerar o grafo diretamente no banco de dados.

Como visto na Seção 3.3.3, a descoberta de triângulos é uma tarefa importante para as redes complexas, pois, eles são a parte principal do coeficiente de clusterização, indicam transitividade e podem ajudar na identificação de comunidades.

Contudo, em situações como a apresentada na Figura 5.9, somente o número de triângulos ou o coeficiente de clusterização não são suficientes para diferenciar nós com topologias diferentes. Isto ocorre por que, apesar dos exemplos apresentados na Figura 5.9 possuírem topologias distintas, eles possuem o mesmo grau e mesmo quantidade de triângulos, o que significa que eles possuem o mesmo valor para o coeficiente de clusterização ($= 0.2$). Identificar topologias diferentes é importante para sistemas de recomendação

e venda de produtos e serviços, dado que um nó como o representado pelo ponto preto na Figura 5.9 (a) pode indicar uma relação mais participativa entre o nó (por exemplo uma pessoa) e um conjunto seleto de nós (por exemplo amigos) ao contrário do que é representado pela Figura 5.9 (b).



(a) (b)

Figura 5.9: Duas representações de nós com o centro em preto com suas respectivas vizinhanças são apresentados em (a) e (b). Os nós centrais (em preto) como mesmo valor para o coeficiente de clusterização 0.2 mas topologias diferentes.

Em termos computacionais, o problema de identificar nós com topologias distintas é definido como verificar se um nó participa de um clique de tamanho maior que 3 (κ_3), isto é, tamanho 4 (κ_4) e 5 (κ_5). Além da contagem e listagem dos cliques κ_4 e κ_5, este trabalho também propõe uma extensão do coeficiente de clusterização que incorpore a contagem desses cliques para descobrir, de fato, se um dado nó ou a rede esta próxima de se tornar um clique, já que como mostrado na Figura 5.9, o coeficiente de clusterização tradicional não é suficiente.

Observou-se também que, os cliques κ_5 e κ_4, assim como o κ_3, seguem diversas leis de potência, sejam elas em relação ao grau dos nós, a distribuição dos cliques ou a relação entre cliques, por exemplo κ_4 vs. κ_3.

Outra observação importante é que os cliques κ_l, com $l \in \{3, 4, 5\}$, são mais influente do que o grau do nó em relação ao primeiro autovalor (λ_1) da matriz de adjacência da rede.

Contudo, mesmo em redes esparsas, a recuperação de cliques κ_4 e κ_5 é uma tarefa que, se mal implementada, pode ser muito custosa. Além disso, o tamanho das redes complexas tem crescido consideravelmente nos últimos anos, o que torna ainda mais

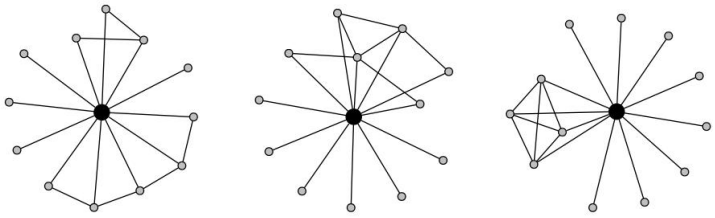
importante o uso de armazenamento eficiente.

As seções seguintes apresentam as observações, experimentos e o método desenvolvido, chamado `FCR - Fast Clique Retrieval`, que utiliza um SGBD para a contagem e listagem dos cliques κ_5 e κ_4.

5.4.1 Coeficiente de Clusterização Genérico

A Figura 5.10 apresenta três subgrafos que representam os nós u, v e w e a vizinhança de cada um deles. Cada um dos nós apresentados tem o mesmo grau ($d(u) = d(v) = d(w) = 12$) e o mesmo número de triângulos ($\kappa_3(u) = \kappa_3(v) = \kappa_3(w) = 6$), consequentemente o mesmo coeficiente de clusterizaçao $\frac{2*6}{13*(13-1)} = 0.077$. Contudo, eles tem diferentes números de κ_4 ($\kappa_4(u) = 0$, $\kappa_4(v) = 1$, $\kappa_4(w) = 4$) e κ_5 ($\kappa_5(u) = \kappa_5(v) = 0$, $\kappa_5(w) = 1$).

Em uma rede social os nós u, v e w podem ser visto como pessoas que possuem um tipo de relacionamento diferente entre seus amigos (nós vizinhos). O nó com topologia em que a quantidade de $\kappa_4(v_i)$ e $\kappa_5(v_i)$ é maior que outros mas com o mesmo $d(v_i)$, como o nó w, pode representar um nó com uma vizinhança que possui um relacionamento mais forte, isto é, muito dos seus amigos são amigos entre si. Esta identificação pode ser usada como referência para sistemas de recomendação para a venda de um serviço ou produto para essa vizinhança com relacionamento mais forte, já que, um relacionamento interpessoal forte pode influenciar mais a compra de produtos e serviços similares entre as pessoas.



(a) Nó central u (b) Nó central v (c) Nó central w

Figura 5.10: Três subgrafos na qual o nó central u, v, w tem o mesmo grau $d(u) = d(v) = d(w) = 12$ e $\kappa_3(u) = \kappa_3(v) = \kappa_3(w) = 6$: (a) o nó central u tem apenas triângulos, (b) um κ_4 centrado no v, (c) quatro κ_4 e um κ_5 centrado no nó w

Para enfatizar a diferença da conectividade de um nó e a sua vizinhança, como mostrado na Figura 5.10, foi adicionado ao coeficiente de clusterização original a contagem de cliques de tamanho maiores, criando assim uma nova equação chamada `Coeficiente`

de Clusterização Generalizado.

A generalização da definição do coeficiente de clusterização para cliques de tamanhos l pode ser feita usando a combinação de arestas que podem formar um clique de tamanho l, como $\binom{d(v_i)}{l-1} = \frac{d(v_i)!}{(l-1)!\cdot(d(v_i)-(l-1))!}$. Assim, o Coeficiente de Clusterização Generalizado é definido na Equação 5.4 para um nó v_i.

$$C_k(v_i) = \sum_{l=3}^{k} \frac{\kappa_l(v_i)}{\binom{d(v_i)}{l-1}} \tag{5.4}$$

sendo $\kappa_l(v_i)$ o número de cliques de tamanho l e $d(v_i)$ é o grau do nó v_i.

O *Coeficiente de Clusterização Generalizado* considerando os cliques até κ_5 ($l = 5$) é apresentado na equação 5.5.

$$\begin{aligned} C_5(v_i) = \frac{2 * \kappa_3(v_i)}{(d(v_i) * (d(v_i) - 1))} + \\ \frac{6 * \kappa_4(v_i)}{(d(v_i) * (d(v_i) - 1) * (d(v_i) - 2))} + \\ \frac{24 * \kappa_5(v_i)}{(d(v_i) * (d(v_i) - 1) * (d(v_i) - 2) * (d(v_i) - 3))} \end{aligned} \tag{5.5}$$

5.4.2 Leis de potência dos Cliques

Como visto no Capitulo 3 o primeiro autovalor da matriz de adjacência da rede complexa tem uma grande importância na mineração das redes, especialmente para o estudo do *Epidemic Threshold*. Na literatura, o nó com maior grau é usado como o nó que mais influencia no primeiro autovalor da matriz de adjacência da rede complexa. Apesar disto não ser sempre verdade, o nó com maior grau tem sido usado por ser a melhor aproximação, pois não há métodos eficientes para se descobrir o nó com maior influencia, a não ser testando a remoção de cada um dos nós e computando o primeiro autovalor, o que torna esta método inadequado para grandes grafos. Também é conhecido que o nó que mais influencia o autovalor são nós que, quando removidos, tendem a deixarem os nós vizinhos mais desconectados, mas não isolados.

Intuitivamente, nós que possuem uma grande quantidade de cliques κ_l com $l = 3, 4, 5$, isto é, que tem o maior valor para o `Coeficiente de Clusterização Generalizado`, tendem, quando removidos, a causarem uma grande quantidade de quebras no grafo e assim ser mais influente no primeiro autovalor da matriz de adjacência da rede do que os nós com alto grau. Assim, é proposta a lei chamada `eigenvalue drop`1 que será comprovada por experimentos nas próximas seções.

Lei 1 (`eigenvalue drop`)**.** *Os nós com maiores valores para* `Coeficiente de Clusterização Generalizado`*, tendem a ser mais influentes no primeiro autovalor da matriz de adjacência de uma rede* (λ_1)*.*

Outra relação importante de ser analisada é a dos cliques versus o grau do nó. Quanto maior o grau de um nó maior deve ser a quantidade de cliques de tamanho $l = 3, 4, 5$ que este nó possui e esta relação segue uma lei de potência chamada `Grau-Clique` (Lei 2).

Lei 2 (`Grau-Clique`)**.** *A relação entre a média do número de cliques de tamanho l versus o grau ϕ de um conjunto de nós de uma rede segue uma lei de potência com expoente $\alpha > 0$.*

$$\overline{\kappa_l} = \phi^{\alpha} \tag{5.6}$$

sendo $\overline{\kappa_l}$ a média de cliques de tamanho l, ϕ é um certo grau e $\alpha > 0$ é o expoente da lei.

Como visto nas seções anteriores, os triângulos, cliques de tamanho 3, se conectam para formar cliques de tamanho maior, isto é, 4. Da mesma maneira os cliques de tamanho 4 se conectam para formar cliques de tamanho 5. Esta relação de transformação também segue uma lei de potência chamada `Clique-Clique` (Lei 3), pois, quanto maior o número de cliques de tamanho l maior também será o número de cliques de tamanho $l + 1$. Contudo, conforme o tamanho l dos cliques aumenta, maior tende a ser o número de nós que se desviam da tendencia.

Lei 3 (Clique-Clique)**.** *Um dado número de cliques de tamanho l que usualmente se conectam formando cliques de tamanho j com $j > l$ seguindo uma lei de potência com $\alpha > 0$.*

$$\overline{\kappa}_j = {\kappa_l}^{\alpha} \tag{5.7}$$

sendo $\overline{\kappa}_j$ a média de cliques de tamanho j e κ_l o número de cliques de tamanho l com $t < j$.

Assim como a distribuição do grau do nó da rede, a distribuição dos cliques também segue uma lei de potência, isto é, muito nós têm poucos cliques enquanto alguns nós possuem muitos cliques. Isto se deve ao fato de que nós de alto grau possuem muitos cliques mas a quantidade de nós de alto grau na rede é baixa, enquanto que, nós de baixo grau possuem pouco cliques e estes são a maioria na rede. A Lei 4 chamada Distribuição-Clique apresenta esta lei de potência.

Lei 4 (Distribuição-Clique)**.** *A distribuição dos cliques κ_l em uma rede segue uma lei de potencia com o expoente $\alpha > 0$.*

$$P(\kappa_l) = {\kappa_l}^{-\alpha} \tag{5.8}$$

5.4.3 Experimentos

Nesta seção são apresentados os resultados dos experimentos para avaliação do método FCR proposto e a comprovação das leis apresentadas na seção anterior. A avaliação foi realizada usando um SGBDR *open source*, o PostgreSQL 8.3.7, em um computador equipado com processador Intel Core2 Quad 2.83GHZ e 4Gb de RAM.

O método FCR foi testado com as redes AS, Epinions, Email Enron e Amazon, descritas na Seção 5.2.

Influência do autovalor

Para medir essa influencia é definido o valor `eigenvalue drop`, que tem por objetivo medir o quanto um nó contribui para o primeiro autovalor da matriz de adjacência da rede. Este valor é computado subtraindo do primeiro autovalor da matriz de adjacência da rede o valor do primeiro autovalor da matriz de adjacência da rede sem um dos seus nós. Então, o λ_{Drop} é dados pela fórmula: $\lambda_{Drop} = \lambda_1 - \lambda'_1$.

O experimento do λ_{Drop} foi realizado da seguinte maneira: o primeiro autovalor (λ_1) do grafo $\mathcal{G}$ é medido, então um nó v_i é removido do grafo $\mathcal{G}$ juntamente com todas as arestas ligadas a ele e então o primeiro autovalor (λ'_1) é medido novamente e o `eigenvalue drop` é computado $\lambda_{Drop_i} = \lambda_1 - \lambda_{1'_i}$. O vértice v_i é colocado de volta juntamente com as suas arestas e outro nó v_{i+1} é removido, e então, $\lambda_{1'_{i+1}}$ é computado novamente como $\lambda_{Drop_{i+1}}$. O processo é repetido até todos os nós desejados serem removidos.

O nó com a maior influência no primeiro autovalor do grafo tem o maior `eigenvalue drop` (λ_{Drop_i}) e usualmente ele é o nó com maior grau, já que este nó apesar de ter muitos nós de grau um conectados a ele, tende a ser o nó com o maior número de $\kappa_l(v_i)$. Contudo, isto nem sempre é verdade. Além disso, se há uma lista de nós com o mesmo grau, uma outra informação se faz necessária para escolher um nó desta lista.

Neste trabalho é mostrado que o nó com maior valor para o `Coeficiente de Clusterização Generalizado`, isto é, o maior valor para $\kappa_l(v_i)$ com $l = 3, 4e5$, tende a ser o nó que tem a maior influência no `eigenvalue drop` e por isso é uma aproximação melhor que o nó com o maior grau.

$\kappa_3(v_i)$	$\kappa_4(v_i)$	$\kappa_5(v_i)$	$C_5(v_i)$	λ_{Drop}
194	1116	4896	0,2762	0,026889
130	426	1176	0,1687	0,015364
104	444	1464	0,1399	0,013307
102	456	1608	0,1384	0,017628
38	54	24	0,0459	0,007135
10	12	0	0,0119	0,004271
10	0	0	0,0115	0,002229

Tabela 5.2: Número de $\kappa_3(v_i)$, $\kappa_4(v_i)$, $\kappa_5(v_i)$, Coeficiente de Clusterização Genérico($C_5(v_i)$) e `eigenvalue drop` para 7 nós da rede AS todos os nós com grau igual $d(v_i) = 30$. O valor real do primeiro autovalor da rede é $66,1191$

A Tabela 5.2 apresenta o resultado de um dos experimentos para comprovar a influência do primeiro autovalor da matriz de adjacência da rede. Para isso foram escolhidos 7 nós com o mesmo grau ($d(v_i) = 30$) da rede AS e computado o `eigenvalue drop` de cada um deles. O resultado é apresentado na Tabela 5.2, ordenada segundo o valor de `eigenvalue drop`, que mostra o número $\kappa_3(v_i)$, $\kappa_4(v_i)$, $\kappa_5(v_i)$, o `Coeficiente de Clusterização Generalizado` $C_5(v_i)$ (Equação 5.4) e `eigenvalue drop` - λ_{Drop} - para cada um dos 7 nós. Dois nós com o mesmo grau, $d(v_i) = 30$, e o mesmo número de triângulos, $\kappa_3(v_i) = 10$, afetam o primeiro autovalor de maneiras diferentes. O nó que tem $\kappa_4(v_i) = 12$ tem um maior `eigenvalue drop` que o nó com $\kappa_4 = 0$, ambos possuem $\kappa_5 = 0$. Assim, o $C_k(v_i)$ pode ser usado no lugar do número do número κ_l.

O mesmo ocorre com o nó que tem $\kappa_3(v_i) = 194$. Este nó possui um maior `eigenvalue drop` já que ele também tem o maior número de κ_4 e κ_5 que qualquer outro nó da Tabela 5.2.

$\kappa_3(v_i)$	$\kappa_4(v_i)$	$\kappa_5(v_i)$	$d(v_i)$	$C_5(v_i)$	λ_{Drop}
35488	**768948**	**12029568**	**1026**	**0,03447**	**2,5623**
31284	535146	7171536	1143	0,02433	1,7953
27342	547116	8794824	1099	0,02307	1,6402
26128	503850	8285952	1244	0,01716	1,3806
26802	506598	8381280	1367	0,01455	1,3568
18434	227466	2641776	1261	0,01171	0,8100
22530	393942	6130728	1245	0,01475	0,7601
17560	250668	3336960	1068	0,01561	0,6695
12758	235818	3881784	924	0,01526	0,5568
896	**3330**	**10680**	**1383**	**0,00047**	**0,0016**

Tabela 5.3: Número de $\kappa_3(v_i)$, $\kappa_4(v_i)$, $\kappa_5(v_i)$, grau do nó, `Coeficiente de Clusterização Generalizado` e `eigenvalue drop` para os 10 nós com maior grau da rede Email−Enron. O valor real do primeiro autovalor deste grafo é $118,4175$

Outro experimento é apresentado na Tabela 5.3. Neste experimento são considerados os 10 nós de maior grau da rede Email-Enron. A Tabela 5.3 é ordenada segundo o valor de `eigenvalue drop`. O nó com maior grau ($d_{max}(v_i) = 1383$) afeta menos o primeiro autovalor que o 9^o nó em grau ($d(v_i) = 1026$), mas que possui o maior valor de `Coeficiente de Clusterização Generalizado`. O mesmo padrão pode ser observado em outros nós da Tabela 5.3.

Os experimentos acima mostram que realmente há uma forte correlação entre o `Coeficiente de Clusterização Generalizado` de um nó e o `eigenvalue drop`, sendo este uma aproximação melhor para nós que influenciam o primeiro autovalor da matriz de adjacência do que o nó com o maior grau, confirmando assim a Lei 1.

Leis de potência dos cliques

A Figura 5.11 apresenta o distribuição do grau para a avaliação do conjunto de dados e a Figura 5.12 apresenta o número de nós com grau ϕ ($\mathcal{V}_\phi = \{v_i \in \mathcal{V} | d(v_i) = \phi\}$) versus a média $\kappa_l(v_i)$ para $l \in \{3, 4, 5\}$. Os gráficos da Figura 5.12 exibem uma correlação entre o grau e a distribuição dos cliques.

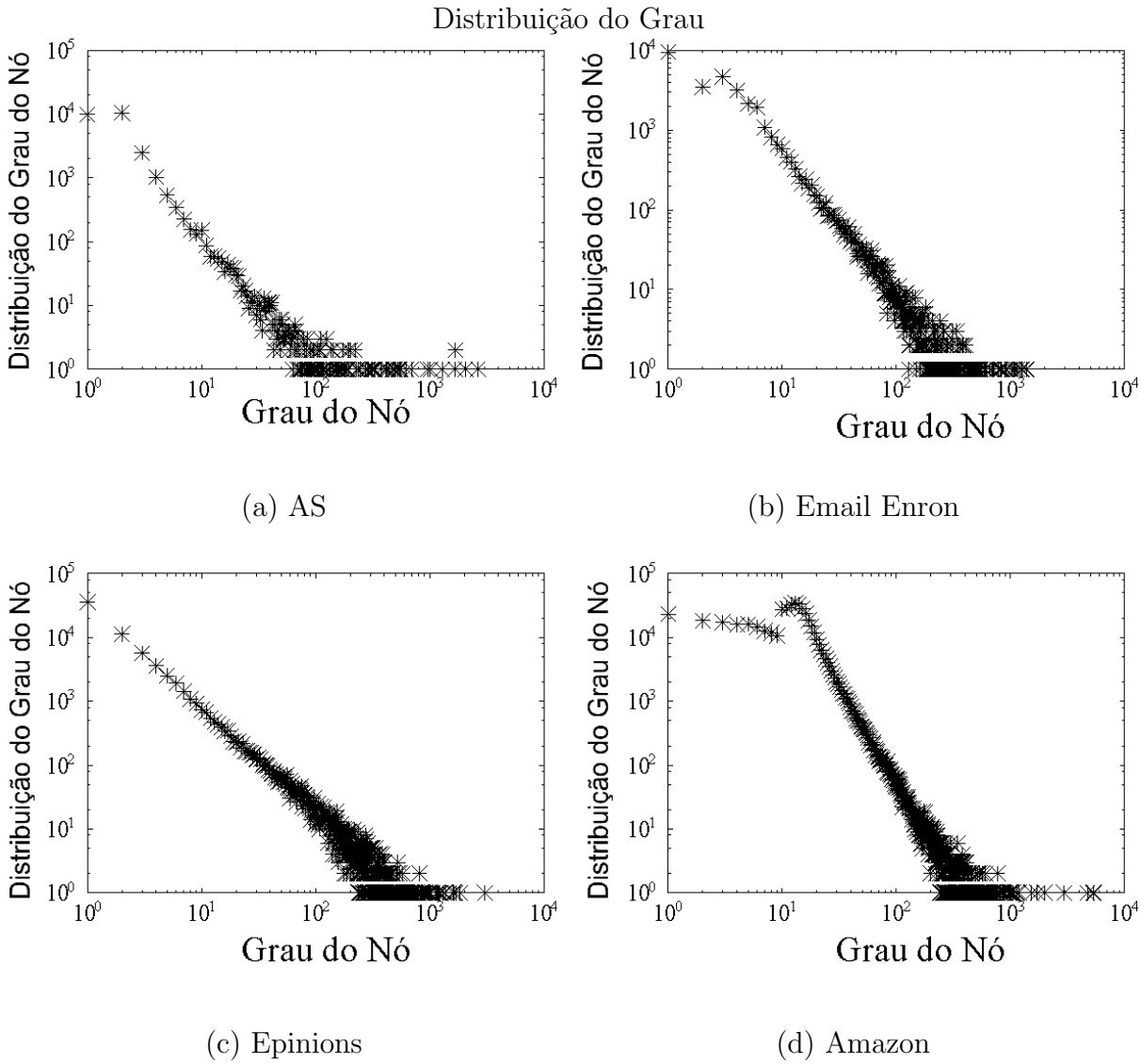
Com o aumento de l mais nós tendem a desviar da tendência mostrada na Figura 5.12, dado que em muitas redes os nós de alto grau possuem muitos nós de grau um conectados a eles. Assim, a probabilidade dos nós de alto grau participarem em cliques maiores decai. Contudo, mesmo com alguns nós desviando do padrão a correlação entre os cliques e o grau do nó segue a lei de potência `Grau-Clique` (Lei 2).

Os nós que desviam desta lei, os pontos destacados nas colunas (b) e (d) da Figura 5.12, são nós com menos cliques do tamanho l que o número de cliques esperados. O nó em destaque na rede Email-Enron é o nó de maior grau e que tem menos influência no `eigenvalue drop` dentre os 10 nós de maior grau. Uma observação interessante ocorre que a rede Amazon, Figure 5.12 (d), que parece seguir duas tendências diferentes, uma até o grau 10 e uma outra acima de 10. O número 10, é o grau em que a rede Amazon tem uma grande mudança na tendência da distribuição do grau, como pode ser visto na Figura 5.12 (d).

Rede	$\kappa_3(G)$	$\kappa_4(G)$	$\kappa_5(G)$
AS	205.590	1.184.544	9.169.320
Email Enron	4.345.594	56.177.760	697.083.240
Epinions	9.746.886	139.281.528	2.090.091.840
Amazon	39.772.974	204.902.448	973.945.920

Tabela 5.4: O valor de $\kappa_3(\mathcal{G})$, $\kappa_4(\mathcal{G})$ e $\kappa_5(\mathcal{G})$ na redes AS, Email Enron, Epinions e Amazon.

A Tabela 5.4 apresenta os número de κ_3, κ_4 e κ_5 para AS, Email Enron, Epinions



(a) AS

(b) Email Enron

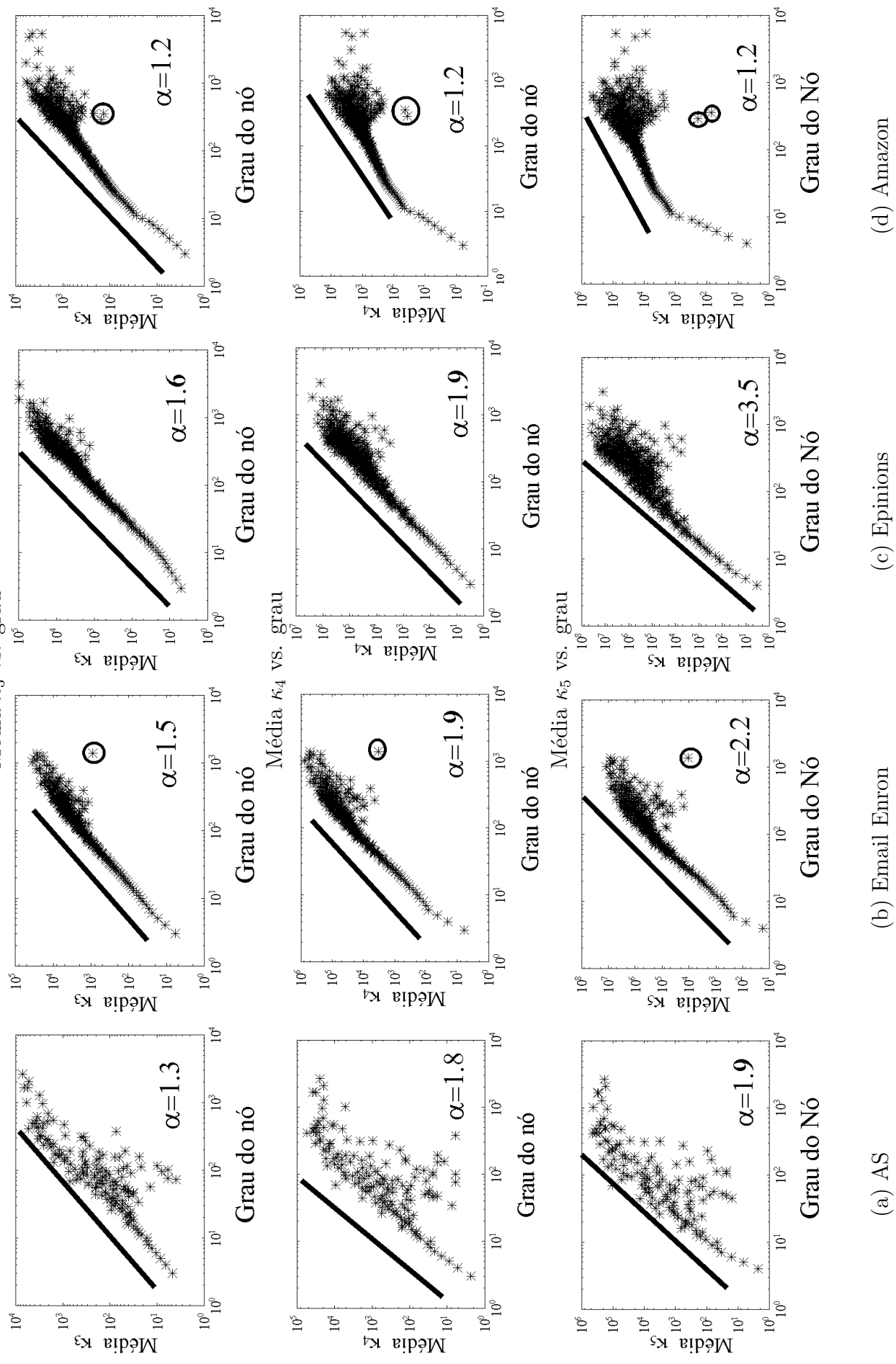
(c) Epinions

(d) Amazon

Figura 5.11: Distribuição do grau das redes AS, Email Enron, Epinions e Amazon.

e Amazon. Como pode ser observado, a rede Amazon tem o maior número de κ_3 e κ_4. Contudo, a rede Epinions tem muito mais κ_5 que a Amazon apesar de ter menos nós e arestas. Isto significa que, a rede Epinions é muito mais conectada que a Amazon. Além disso, a rede Epinions e Email Enron tem poucos nós desviando da tendência.

Apesar de todas as redes apresentarem pontos desviando da tendência conforme o tamanho dos cliques aumenta, pode ser observado que as redes desviam de maneiras diferentes. Na rede AS (Figura 5.12 coluna (a)) muitos dos nós que desviam da tendência não são os de alto grau. Estes nós tendem a ser aqueles que estão na periferia da rede. Por outro lado, em outras redes como as Email Enron, Epinions e Amazon representadas nos



(a) AS (b) Email Enron (c) Epinions (d) Amazon

Figura 5.12: Média de $\kappa_l(v_i)$ ($l \in 3, 4, 5$) vs grau do nó das redes AS, Email Enron, Epinions e Amazon.

gráficos da Figura 5.12 colunas (b),(c) e (d) respectivamente, muito dos nós que desviam são os nós de alto grau, especialmente na rede Email Enron. Na Figura 5.12 também pode ser visto que estas redes tem grande parte dos nós desviando por volta do grau 100, que é o melhor tamanho para comunidades, como apresentado no Capítulo 3.

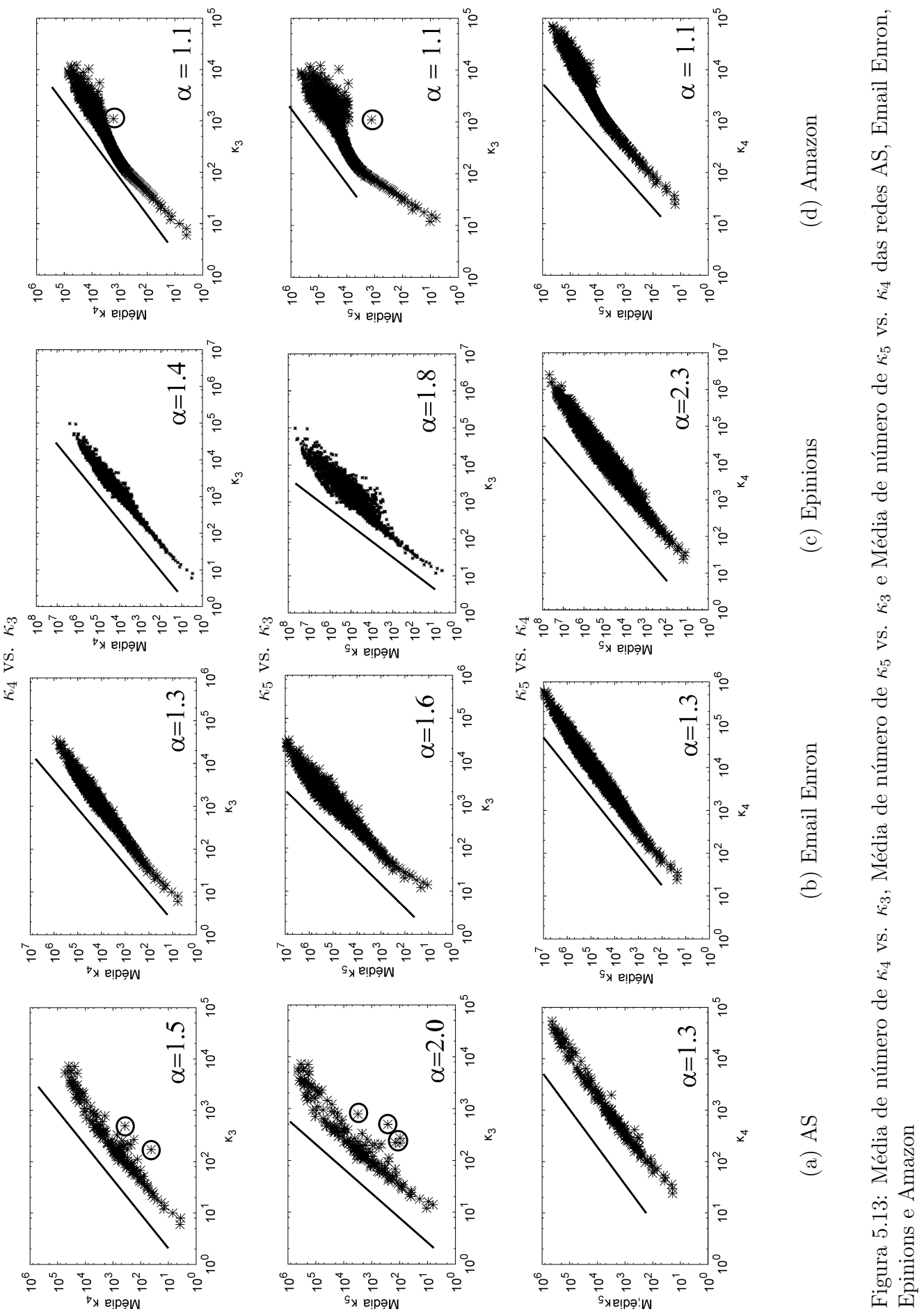
Os triângulos de um nó tendem a se conectarem entre si formando cliques maiores tais como κ_4 e κ_5. A Figura 5.13 mostra esta correlação de κ_4 e κ_5 versus κ_3 e κ_5 versus κ_4, que segue a lei de potência `Clique-Clique` (Lei 3)

O grande número de $\kappa_4(v_i)$ e $\kappa_5(v_i)$ nas redes estudadas demonstra que as redes são bem conectadas e exibem uma estrutura de comunidade. Especialmente as redes Email Enron e Epinions, nas quais são redes sociais e tem seus gráficos bem parecidos.

Contudo, o número de $\kappa_3(v_i)$, $\kappa_4(v_i)$ e $\kappa_5(v_i)$ que um nó possui é bem inferior ao número máximo de cada um desses cliques que cada nó poderia ter. Por isso, pode-se dizer que uma rede complexa apesar de ser bem conectada esta longe de ser um grafo completo.

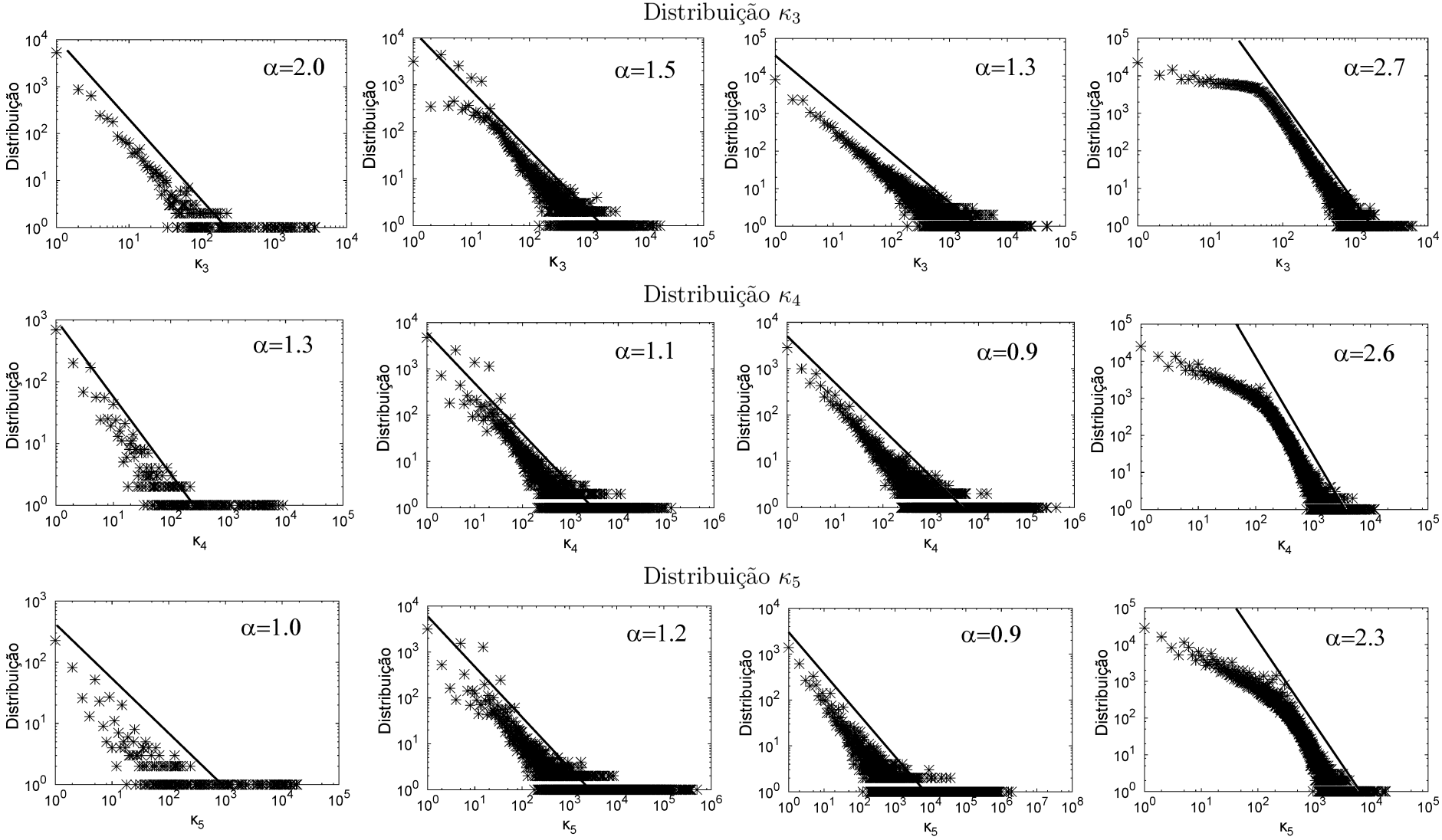
Os pontos que desviam desta lei são nós que tem poucos cliques que participam em cliques de tamanho maiores. Alguns pontos da Figura 5.13 colunas (a) e (d) que desviam da tendência foram destacados. Esses pontos representam a quantidade de cliques de tamanho 3 que um nó participa versus a média de cliques de tamanho 4 e 5 que esses cliques formaram. O média de κ_4 e κ_5 formados é menor do que se espera, assim uma parte significante desses nós tem uma topologia com mais triângulos desconexos entre si do que se espera. O mesmo acontece para a relação κ_5 vs. κ_4. Isto é um indicativo de que um nó participa em mais de um grupo, ou seja, as comunidades tendem a ter sobreposição.

Outra observação constatada é que a distribuição dos cliques de fato segue a lei de potência `Distribuição-Clique` (Lei 4), como apresenta a Figura 5.14. A distribuição dos cliques tem o expoente da lei de potencia similar ao expoente da distribuição do grau (α) e as mesmas "quebras", especialmente para cliques menores como o κ_3. Um exemplo é a rede Amazon, que tem um "quebra" no grau 10 e esta quebra aparece na distribuição dos cliques. Contudo, com o aumento do valor de l a quebra tende a desaparecer na distribuição do grau.



(a) AS (b) Email Enron (c) Epinions (d) Amazon

Figura 5.13: Média de número de κ_4 vs. κ_3, Média de número de κ_5 vs. κ_3 e Média de número de κ_5 vs. κ_4 das redes AS, Email Enron, Epinions e Amazon



(a) AS (b) Email Enron (a) Epinions (b) Amazon

Figura 5.14: Distribuição do κ_l com $l = 3, 4, 5$ para as redes (a) As, (b) Email Enron, (c) Epinions e (d) Amazon em cada coluna. A distribuição dos cliques segue uma lei de potência com o grau similar a distribuição do grau e as mesmas "quebras" como no caso da distribuição do grau do nó da rede Amazon.

5.4.4 Método `FCR`

Esta seção tem por objetivo apresentar o algoritmo que conta e lista o número de $\kappa_4(v_i)$ e $\kappa_5(v_i)$ de um nó v_i. As redes complexas são modeladas como grafos que podem ser vistos como relações em um banco de dados. O método `FCR - Fast Clique Retrieval` explora o gerenciamento e otimizações providas por um Sistema Gerenciador de Base de Dados Relacional (SGBDR) para encontrar κ_4 e κ_5 em um grafo. O esquema convencional de armazenamento de grafos, que armazena somente a lista de arestas, chamado neste trabalho de "Método Tradicional", requer muitas operações de junção para recuperar todos os cliques. Além disso, as redes grandes demandam um elevado custo de processamento. O método `FCR` quebra o grafo que representa a rede em subgrafos menores, permitindo a recuperação eficiente de κ_4 e κ_5 para cada nó.

No método `FCR`, o grafo é primeiramente dividido em N subgrafos, sendo N o número de nós. Um subgrafo $\mathcal{G}_{s_i} = (\mathcal{V}_{s_i}, \mathcal{E}_{s_i}), i \in \{1, \dots N\}$ é o subgrafo centrado no nó v_i, tal que $\mathcal{V}_{s_i} = \{u | u \in \mathcal{V} \wedge (v_i, u) \in \mathcal{E}\}$ e $\mathcal{E}_{s_i} = \{(u, w) | (u, w) \in \mathcal{E} \wedge u, w \in \mathcal{V}_{s_i}\}$, que contém a vizinhança do nó v_i e cada aresta conectada a estes nós. Então, cada aresta de $\mathcal{E}_{s_i}$ é armazenada em uma tabela adicional chamada `Subgrafos`. Assim, a base de dados contém somente duas tabelas, uma com a lista de aresta contendo todas as arestas de $\mathcal{G}$ e a outra, chamada `Subgrafos`, que armazena apenas as arestas de cada $\mathcal{G}_{s_i}$. Para cada aresta armazenada na tabela `Subgrafos` há o nó de saída e o nó de destino da aresta e um identificador (*grafoi*), que é o índice i do subgrafo G_{s_i}. Índices, árvores B, foram criados em ambas as tabelas, para todos os atributos, para que o otimizador do SGBDR possa ser utilizada. O processo de criação e preenchimento da tabela `Subgrafos` é apresentado pelo Algoritmo 5.4.1.

A Figura 5.15 apresenta um exemplo de um grafo $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ e o nó $v_i = 1$ representado pelo ponto preto no grafo, que será usado como exemplo para explicar como o método `FCR` funciona. A lista de arestas de $\mathcal{G}$, exemplificada pela Figura 5.15, é armazenada em uma base de dados relacional como uma tabela, na qual uma parte dela, é mostrada pela Figura 5.22(a). Note-se que como o grafo é não direcionado ambas as arestas (v_i, v_j) e (v_j, v_i) são armazenadas.

Algoritmo 5.4.1 Criação da `Subgrafos`

Entrada: O grafo de entrada $\mathcal{G}(\mathcal{V}, \mathcal{E})$
Saída: A base de dados com duas tabelas: `Subgrafos` e tabela de arestas

```
1: Cria-se a tabela de arestas
2: Insere as arestas do grafo G na tabela de arestas
3: Cria a tabela Subgrafos
4: Divide-se o grafo G em |V| = N subgrafos
5: para i=0 para N faça
6:    para todo arestas conectando a vizinhança do nó v_i faça
7:       Identifica cada aresta como i
8:       Insere cada aresta na Subgrafos
9:    fim-para
10: fim-para
11: Cria índices em todas as tabelas
```

As arestas $\mathcal{E}_{s_i}$ que compõe cada subgrafo $\mathcal{G}_{s_i}$ são armazenadas na tabela `Subgrafos`. A tabela `Subgrafos` armazena as arestas de todos os $|\mathcal{V}| = N$ subgrafos, exceto aqueles que não possuem arestas conectando os seus vizinhos (nós 5, 7 e 8 da Figura 5.15). Parte da tabela `Subgrafos` do grafo da Figura 5.15 é apresentada na Tabela 5.22 (b) que mostra algumas arestas dos subgrafos $\mathcal{G}_{s_i}$.

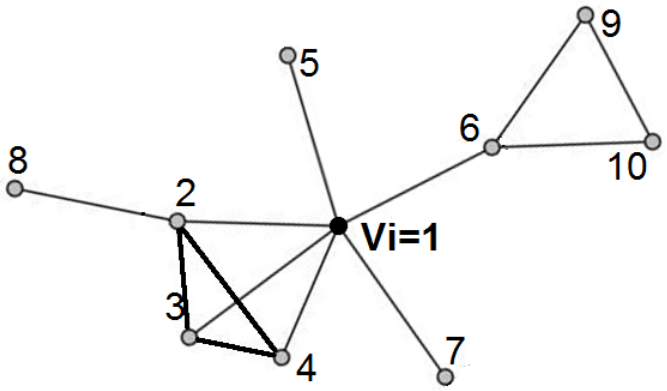


Figura 5.15: Um grafo G e o nó $v_i = 1$ que tem as arestas de seus subgrafos representadas por linhas grossas.

A ideia geral para encontrar todos os κ_4 de um nó v_i é a seguinte:

1. a vizinhança $\mathcal{N}_{v_i}$ é recuperada;
2. para cada aresta (v_i, v_j) recupere todas as arestas (v_i, v_j, z) na tabela `Subgrafos`
3. verifique se a combinação do índice z dois-a-dois de todas as arestas recuperadas são arestas existentes em $\mathcal{G}$

Instanciando-se para encontrar todos os $\kappa_4(v_i)$ do nó $v_i = 1$ da Figura 5.15 tem-se:

1. as arestas $(1,2;1,3;1,4;1,5;1,6;1,7)$, que são a vizinhança de v_i, são recuperadas.

2. para cada aresta recuperada é verificado na tabela `Subgrafos` se ela pertence a algum subgrafo. Para a aresta $(1,2)$ é recuperada as seguintes tuplas: $(1,2,3;1,2,4;)$ dado que estas arestas são parte da vizinhança dos nós 3 e 4.

3. então, existe um $\kappa_4(v_i)$ se existe aresta entre os nós recuperados, isto é, no exemplo checar se existe a aresta $(3,4)$.

O SQL para recuperar todos os κ_4 da rede Epinios é apresentado a seguir. Primeiro é apresentada a consulta no modo tradicional em que são necessários 11 junções para a recuperação de todos os κ_4.

O κ_4 é composto por 6 arestas (a,b,c,d,e,f) que correspondem a cada uma das tabelas na consulta representada na Figura 5.16, as junções correspondem aos nós do grafo que formam κ_4. A intuição da consulta SQL é representada com o grafo da Figura 5.17.

```
SELECT a.src, a.dest, b.dest, c.dest
FROM epinions a, epinions b, epinions c, epinions d, epinions e,
     epinions f
WHERE and a.src  =  b.src  and a.src  =  c.src  and d.src  = a.dest
      and d.dest =  c.dest and e.src  =  b.dest and e.dest = c.dest
      and f.src  =  a.dest and f.dest =  b.dest and a.dest <> b.dest
      and a.dest <> c.dest and b.dest <> c.dest;
```

Figura 5.16: Consulta SQL para recuperar todos os κ_4 da rede Epinios

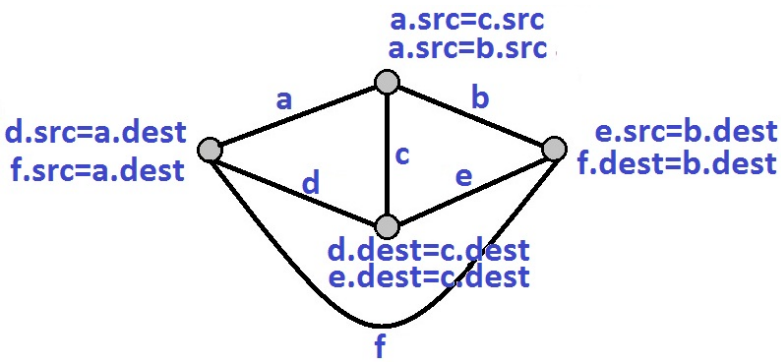


Figura 5.17: Grafo correspondente a Consulta SQL da Figura 5.16

Utilizando a tabela `Subgrafos` são necessárias apenas 4 junções como representado na consulta da Figura 5.18, que é mais clara e eficiente, evitando a necessidade de checar a existência da mesma aresta mais de uma vez.

```
SELECT a.src, a.dest, m.src, m.dest
FROM epinions a, epinionssubgraph m, epinionssubgraph n
WHERE a.src = m.grafoi and a.dest = n.grafoi and m.src = n.src
      and m.dest = n.dest;
```

Figura 5.18: Consulta SQL para recuperar todos os κ_4 da rede Epinios utilizando a tabela `Subgrafos`

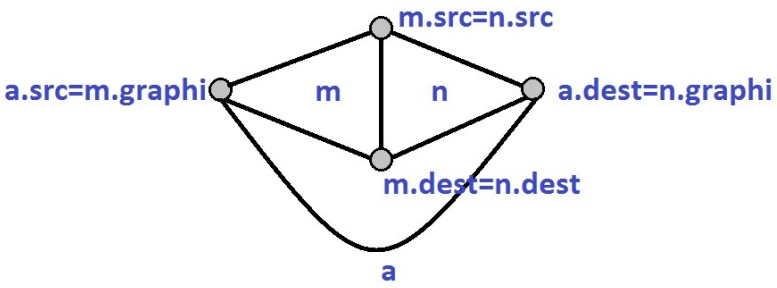


Figura 5.19: Grafo correspondente a Consulta SQL da Figura 5.18

Para encontrar todos os κ_5 de um nó v_i são necessários os seguintes passos:

1. a vizinhança $\mathcal{N}_{v_i}$ é recuperada;
2. para cada aresta (v_i, v_j) recupere todas as arestas (v_i, v_j, z) na tabela `Subgrafos`;
3. verifique se a combinação do índice z três-a-três, (z_1, z_2, z_3), corresponde a tuplas existentes na tabela `Subgrafos`;

Na verdade, utilizando a tabela `Subgrafos` é necessário apenas descobrir se (z_1, z_2, z_3) existe na tabela `Subgrafos` ao invés de ter que checar se (z_1, z_2), (z_1, z_3) e (z_2, z_3) exitem na tabela de arestas.

O SQL para a recuperação de todos os κ_5 da rede Epinions também é apresentado. Primeiro o modo tradicional que requer ainda mais junções que a recuperação de κ_4, ao todo são 21 junções.

O método `FCR` necessita apenas de 7 junções tornando a redução computacional ainda maior.

```
SELECT a.src, a.dest, b.dest, c.dest, d.dest
FROM epinions a, epinions b, epinions c, epinions d, epinions e,
     epinions f, epinions g, epinions h, epinions i, epinions j
WHERE a.src = b.src and a.src = c.src and a.src = d.src
      and a.dest =  e.src  and a.dest =  g.src  and a.dest =  i.src
      and b.dest =  e.dest and b.dest =  h.src  and b.dest =  j.src
      and c.dest =  g.dest and c.dest =  j.dest and c.dest =  f.src
      and d.dest =  f.dest and d.dest =  h.dest and d.dest =  i.dest
      and c.dest <> d.dest and a.dest <> b.dest and a.dest <> c.dest
      and a.dest <> d.dest and b.dest <> c.dest and b.dest <> d.dest;
```

Figura 5.20: Consulta SQL para recuperar todos os κ_5 da rede Epinios utilizando a tabela `Subgrafos`

```
SELECT z.src, z.dest, z.grafoi, a.src, a.dest
FROM epinionssubgraph z, epinionssubgraph a,
     epinionssubgraph b, epinionssubgraph c
WHERE z.src = a.grafoi and z.dest = b.grafoi and z.grafoi = c.grafoi
      and a.src  = b.src  and a.src  = c.src
      and a.dest = b.dest and a.dest = c.dest;
```

Figura 5.21: Consulta SQL para recuperar todos os κ_5 da rede Epinios

Na prática, para encontrar todos os κ_5 de um nó é o mesmo procedimento para encontra todos os κ_4 de um nó. A diferença é que é necessário checar apenas a existência de três arestas que são conectadas entre si ao invés de checar uma aresta em κ_4. Encontrar três arestas conexas no método `FCR` é simples dado que cada nó tem as arestas que conectam a sua vizinhança armazenadas na tabela `Subgrafos`. O tamanho em número de linhas da tabela `Subgrafos`, para as redes testadas no experimento de escalabilidade, é apresentado na Figura 5.23 (a).

5.4.5 Escalabilidade

Esta Seção apresenta a avaliação do método `FCR - Fast Clique Retrieval` e mostra que ele é até 4 e 12 vezes mais rápido que o método tradicional para consultar todos os κ_4 e κ_5, respectivamente. A Figura 5.23 (c) e (d) e a Figura 5.24 apresentam os resultados dos experimentos para avaliar o quão rápido o método `FCR` é para consultar todos os κ_4 e κ_5. Todos os tempos são medidos em segundos e os valores mostrados são médias de 3 execuções, com o cache sendo limpo após cada execução. O método `FCR` é

SRC	DEST
1	2
1	3
1	4
1	5
1	6
2	1
2	3
3	4
...	...

(a) - Lista de arestas

SRC	DEST	grafo i
2	3	1
2	4	1
3	2	1
3	4	1
4	2	1
4	3	1
1	2	3
1	2	4
...	...	...

(b) - tabela `Subgrafos`

Figura 5.22: Tabela (a) é a lista de aresta tradicional (a única usada no método tradicional). Tabela (b) é a tabela `Subgrafos`, que armazena todas as arestas dos subgrafos $\mathcal{G}_{s_i}$.

representado pela linha contínua e o método tradicional pela linha pontilhadas. O banco de dados utilizado é o PostgreSQL 8.3.7. A Figura 5.23 (a) apresentam informações sobre a tabela `Subgrafos` e a tabela de arestas e a Figura 5.23 (b) apresenta as informações sobre a quantidade de cliques sendo a linha contínua a quantidade de κ_4 e a pontilhada a quantidade de κ_5.

Para controlar os experimentos foram usadas redes baseadas no conjunto de cidades americanas (EUA), que é composto por latitudes e longitudes de 25.375 cidades. As redes usadas na avaliação da escalabilidade foram criados usando o modelo *knn* variando $k \in \{4, 5, 7, 10, 15, 20, 25\}$. Assim, cada rede tem o mesmo número de nós mas diferentes quantidades de arestas, como apresentado na Figura 5.23. O maior ganho do algoritmo é alcançado com $k = 25$, isto é, consultando todos os κ_4 usando `FCR` e 4 vezes mais rápido (Figura 5.23 (c)) e consultando todos os κ_5 usando `FCR` e 12 mais rápido do que usando o método tradicional (Figura 5.23 (d)).

No segundo experimento de escalabilidade foi executada uma consulta para mil nós da rede Amazon (o maior grafo real em número de nós e arestas utilizado neste trabalho) que foram escolhidos randomicamente. A Figura 5.24 apresenta a média de tempo por consulta para o método `FCR` e tradicional. Antes de cada consulta, o cache do computador e do banco de dados foram limpos. Como pode ser visto, o método `FCR` é 8 vezes mais rápido para κ_4 e 9 mais rápido para κ_5, quando comparado com o método tradicional, que

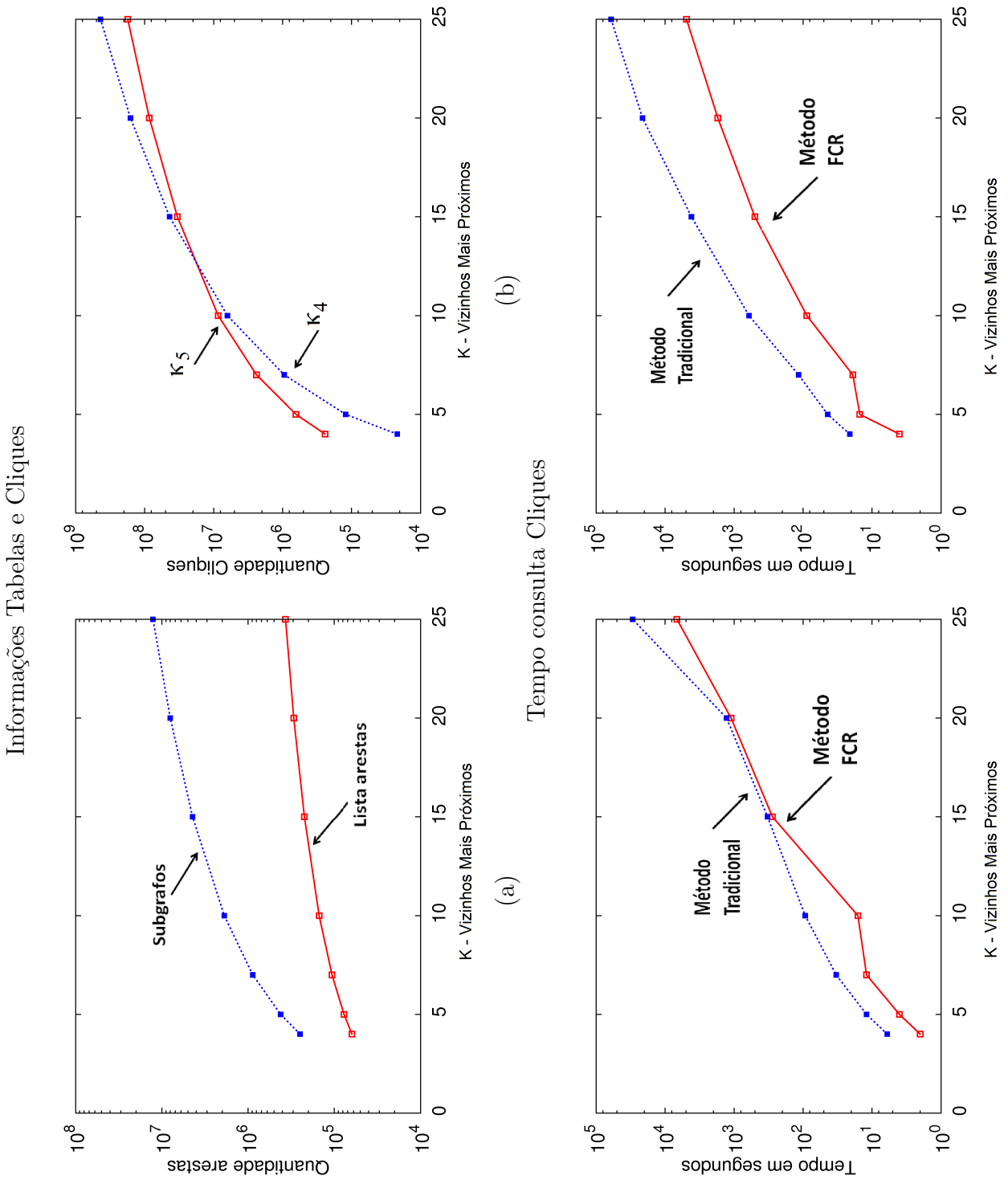


Figura 5.23: (a) Informações sobre a quantidade de tuplas nas tabelas: `Subgrafos` e de arestas; (b) Informações sobre a quantidade de κ_4 e κ_5; (c) Tempo de consulta dos κ_4 para o método tradicional e o `FCR`; (d) Tempo de consulta dos κ_5 para o método tradicional e o `FCR`;

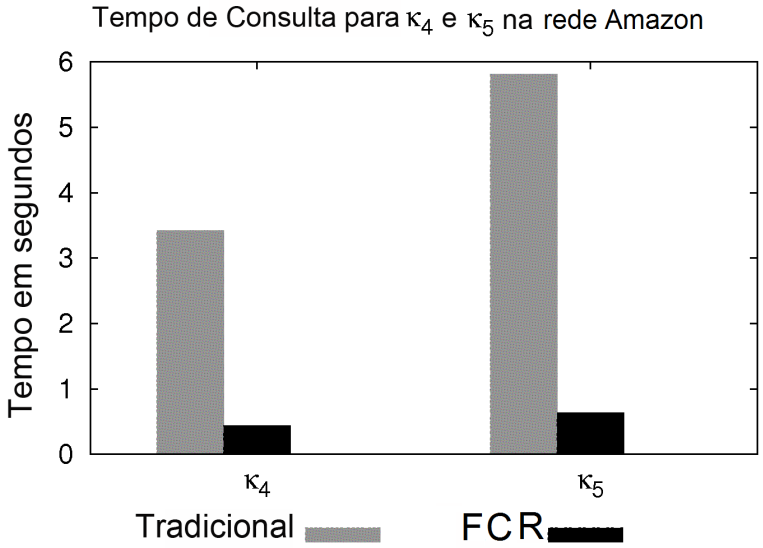


Figura 5.24: Média de mil consultas na rede Amazon

utiliza apenas a tabela com a lista de arestas no SGBD. Como os experimentos mostram, o `FCR` é um método eficiente para recuperar os κ_4 e κ_5 das redes.

5.5 Considerações Finais

A mineração de grafos que modelam redes complexas tem se tornado uma atividade importante na área computacional, já que este tipo de mineração permite representar não só os elementos de um conjunto, mas também as ligações entre esses elementos. O trabalho na área de redes complexas desenvolvido nesta tese trouxe diversas contribuições. Entre elas, destaca-se o algoritmo `ShatterPlots` que permite a separação entre redes complexas entre reais e sintéticas por meio de padrões interessantes descobertos no `ShatterPoint`.

Também foi visto que os cliques κ_4 e κ_5 seguem leis de potencias, aqui nomeadas de `Grau-Clique`, `Clique-Clique` e `Distribuição-Clique`. Além disso, a extensão do coeficiente de clusterização, para a inclusão de cliques de tamanho maiores, tornou-se uma equação importante para a identificação de nós com topologias diferentes. O uso dos SGBDRs para armazenamento e apoio à mineração de redes complexas mostrou ser adequado e útil abrindo uma nova frente de exploração para o armazenamento de redes complexas, já que com o crescimento do tamanho das redes complexas, apenas o uso da memória principal se torna inviável.

Parte III

Conclusões

Capítulo 6

Conclusões

6.1 Considerações Iniciais

Nos últimos anos não só o seu volume dos dados tem aumentado significativamente, mas também a maneira de como eles são representados. As redes complexas são um exemplo de uma nova necessidade de representação que é imposta aos SGBDs. O processo de descoberta de conhecimento para analisar esses dados também deve ser interativo e iterativo, fazendo com que algoritmos de mineração de dados que tenham um grande custo computacional se tornem inviáveis para serem utilizados em conjunto de dados volumosos, sejam eles multidimensionais ou modelados como grafos, como no caso das redes complexas. Com isso, existe a necessidade de novos algoritmos, mais escaláveis que os algoritmos tradicionais da área de grafos, para a mineração desses novos conjuntos de dados.

Dados multidimensionais usualmente não seguem uma distribuição uniforme, possuem muitos atributos e muitos elementos considerados ruídos, fazendo com que algoritmos como os de detecção de agrupamentos se tornem ainda mais custosos e o pré-

processamento uma tarefa ainda mais importante.Para as redes complexas, há a necessidade de se desenvolver novos algoritmos para a extração de propriedades afim de propiciar o entendimento do comportamento das redes complexas bem como a maneira como ocorrem as ligações entre os elementos, permitindo assim a criação de melhores geradores de redes complexas sintéticos. Além disso, também há a necessidade do estudo de novos meios de armazenamento para as redes complexas.

6.2 Principais Contribuições

O objetivo desta tese é propor métodos que possam trabalhar com grandes volumes de dados. Dentre as contribuições se destacam a remoção seletiva de dados, visando reduzir o volume de dados a ser processado. Primeiramente foi proposto o algoritmo *Biased Box Sampling - BBS* para reduzir grandes conjuntos de dados multidimensionais que tenham agrupamentos de dados desbalanceados, ruídos e uma grande quantidade de atributos. Os resultados dos experimentos sobre esse algoritmo mostram que as amostras obtidas não só permite aumentar a velocidade da execução dos algoritmos de detecção de agrupamento mas em muitos casos contribuem para melhorar a qualidade dos resultados.

Em seguida, o algoritmo `ShatterPlots`, por meio da remoção aleatória progressiva de arestas, permite analisar redes complexas de grande tamanho e classificá-las em reais e sintéticas. Isso é feito extraindo-se diversas propriedades interessantes no `ShatterPoint`, que é o ponto no processo do `ShatterPlots` em que a rede possui o maior diâmetro efetivo. Por fim, foi feito um estudo de cliques de tamanho 4 e 5 (κ_4 e κ_5) mostrando que esses cliques também são importantes para a analise topológica da rede, agilizando assim a descoberta de nós influentes no primeiro autovalor da matriz de adjacência do grafo. Também foram encontradas as leis de potências `Grau-Clique`, `Clique-Clique` e `Distribuição-Clique`, que são importante para a análise da rede em relação aos cliques, pois nós que desviam dos padrões encontrados são excessões e devem ser melhores analisados. Para a descoberta dos cliques foi desenvolvido o método `FCR - Fast Clique Retrieval` que faz uso dos SGBDRs para a mineração das redes complexas. Também foi proposta a extensão do coeficiente de clusterização, que o torna uma equação mais rica

para a identificação de nós que realmente tendem a formar cliques nas redes complexas.

Concluindo, esta tese fez um amplo e valioso estudo em grandes conjuntos de dados, englobando o desenvolvimento de diversos métodos para a mineração de dados e redes complexas com resultados bastante interessantes. Também foi apresentado que os bancos de dados podem ser ferramentas uteis para o armazenamento, consulta e mineração de grandes redes complexas.

6.3 Publicações

As publicações originarias desta tese estão listadas abaixo:

- Appel, Ana Paula e Paterlini, Adriano Arantes e Sousa, Elaine Parros Machado de e Traina, Caetano, Jr., "Amostragem Balanceada Baseada na Correlação Fractal" II Workshop em Algoritmos e Aplicações de Mineração de Dados (II WAAMD) junto ao 21o. Simpósio Brasileiro de Banco de Dados (SBBD'06), 2006,1-8p, Florianópolis, SC

- Appel, Ana Paula e Paterlini, Adriano Arantes e Sousa, Elaine Parros Machado de e Traina, Agma Juci Machado e Traina, Caetano, Jr., Biased Box Sampling - A Density-Biased Sampling for Clustering, ACM Symposium on Applied Computing (SAC'2007), 2007,2p.,Seoul, Korea;

- Appel, Ana Paula e Paterlini, Adriano Arantes e Sousa, Elaine Parros Machado de e Traina, Agma Juci Machado e Traina, Caetano, Jr., A Density-Biased Sampling Technique to Improve Cluster Representativeness, Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, 366-373p;

- Appel, Ana Paula e Faloutsos, Christos e Traina, Caetano, Jr., "Graph Mining Techniques: focusing on discerning real from synthetic graphs", Book Chapter Proposal, Graph Data Management: Techniques and Applications (proposta aprovada).

Publicações originadas de trabalhos colaborativos:

- Appel, Ana Paula e Chakrabarti, Deepayan e Faloutsos, Christos e Kumar, Ravi e Leskove, Jure e Tomkins, Andrew, "ShatterPlots: a fast tool for mining large graphs", Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 3, 2009, Sparks, Nevada, USA, 1-12p;

- Kang, U e Tsourakakis, Charalampos E., Appel, Ana Paula, Faloutsos, Christos and Leskove, Jure, "Radius plots for mining Tera-byte Scale Graphs: Algorithms, Patterns and Observations", SIAM Conference on Data Mining (SDM10), April 29-May 1, 2010, Columbus, Ohio , 1-12p;

- Kang, U e Tsourakakis, Charalampos E., Appel, Ana Paula, Faloutsos, Christos and Leskove, Jure, "Radius plots for mining Tera-byte Scale Graphs: Algorithms, Patterns and Observations", ACM Transactions on Knowledge Discovery from Data Special Issue on Large-scale Data Mining: Theory and Applications, 1-12p (aceito para publicação);

6.4 Trabalhos Futuros

Os principais direcionamentos de trabalho futuro para este trabalho de pesquisa são listados a seguir:

6.4.1 Algoritmo *Biased Box Sampling - BBS*

- Gerar um novo algoritmo que se alie a técnica desenvolvida, o algoritmo *BBS*, com técnicas de redução de atributos para reduzir o conjunto de dados não só quanto ao número de elementos mas também quanto ao número de atributos.

- Adaptar o algoritmo *Biased Box Sampling - BBS* para que seja possível realizar a amostragem de dados em *data streams*. Muitas vezes é necessário computar estatísticas sobre *streams* de dados em um espaço reduzido, sendo necessária a escolha dinâmica de alguns elementos representativos do conjunto. As *streams* de dados também podem conter ruído e apresentar conjunto de elementos com densidades

diferentes, sendo assim, um algoritmo de amostragem de dados que leve em conta essas características mais eficiente que a amostragem uniforme.

- Gerar um novo algoritmo que utilize a técnica do algoritmo *BBS* para encontrar agrupamentos de dados. O algoritmo *BBS* mostrou reduzir eficientemente a quantidade de ruído no conjunto de dados permitindo que algoritmos de detecção de agrupamento encontre os agrupamentos mais facilmente. Assim, é possível adaptar o algoritmo *BBS* para não só remover o ruído e selecionar um conjunto de elementos menor mas também já encontra os agrupamentos em definitivo.

6.4.2 Algoritmo `ShatterPlots`

- Análise do comportamento do `ShatterPoint` em relação ao raio das redes ao invés da utilização do diâmetro efetivo;

- Análise do comportamento do `ShatterPoint` quanto a evolução das redes complexas, isto é, ver se conforme o crescimento da rede o `ShatterPlots` se desloca ou permanece constante e se esse comportamento é igual para as redes reais e geradores sintéticos que produzem redes com evolução como RTM e o RTG.

- Análise de outras abordagens de remoção de arestas. Um exemplo seria ordenar as arestas por meio de pesos, como a soma do grau dos nós que a aresta liga. Verificar se esta nova abordagem produz o `ShatterPoint` e o analisar o comportamento das medidas extraídas no `ShatterPoint` nesse novo `ShatterPoint`.

6.4.3 Método `FCR - Fast Clique Retrieval`

- Avaliar se a utilização de hints e reordenação das junção ajudam a otimizar as consultas SQL para a recuperação de cliques 4 e 5;

- Adaptar o método `FCR` para que seja possível a utilização da arquitetura Map/Reduce (paralelização) ou o uso de SGBDs paralelos para aumentar a velocidade das consultas e melhorar o suporte a grandes redes complexas. Também

será interessante avaliar o armazenamento em colunas ao invés do tradicional armazenamento de linhas;

Referências Bibliográficas

Abello, J.; Buchsbaum, A. L.; Westbrook, J. R. A functional approach to external graph algorithms. In: *Algorithmica*, Springer-Verlag, 1998, p. 332–343.

Acar, E.; Dunlavy, D. M.; Kolda, T. G. Link prediction on evolving data using matrix and tensor factorizations. In: Saygin, Y.; Yu, J. X.; Kargupta, H.; Wang, W.; Ranka, S.; Yu, P. S.; Wu, X., eds. *ICDM Workshops*, IEEE Computer Society, 2009, p. 262–269.
Disponível em http://dblp.uni-trier.de/db/conf/icdm/icdmw2009.html#AcarDK09

Agrawal, R.; Srikant, R. Fast algorithms for mining association rules in large databases. In: *Proc. of VLDB Conf.*, Santiago, Chile, 1994, p. 487–499.

Akoglu, L.; Faloutsos, C. Rtg: a recursive realistic graph generator using random typing. *Data Min. Knowl. Discov.*, v. 19, n. 2, p. 194–209, 2009.

Akoglu, L.; McGlohon, M.; Faloutsos, C. Rtm: Laws and a recursive generator for weighted time-evolving graphs. In: *ICDM*, 2008, p. 701–706.

Albert, R.; Jeong, H.; Barabási, A.-L. Diameter of the World-Wide Web. *Nature*, v. 401, p. 130–131, 1999.

Albert, R.; Jeong, H.; Barabási, A.-L. Error and attack tolerance of complex networks. *Nature*, v. 406, p. 378–381, 2000.

Appel, A. P.; Chakrabarti, D.; Faloutsos, C.; Kumar, R.; Leskove, J.; Tomkins, A. Shatterplots: a fast tool for mining large graphs. In: *Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 3, 2009, Sparks, Nevada, USA*, SIAM, 2009, p. 1–12.

Appel, A. P.; Paterlini, A. A.; de Sousa, E. P. M.; Traina, A. J. M.; Jr., C. T. A density-biased sampling technique to improve cluster representativeness. In: *Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, Proceedings*, Springer, 2007a, p. 366–373 (*Lecture Notes in Computer Science*, v.4702).

Appel, A. P.; Paterlini, A. A.; Sousa, E. P. M. d.; Traina, Caetano, J. Amostragem balanceada baseada na correlação fractal. In: *II Workshop em Algoritmos e Aplicações de Mineração de Dados (II WAAMD) junto ao 21ž Simpósio Brasileiro de Banco de Dados (SBBD'06)*, Florianópolis, SC: SBC, 2006, p. 1–8.

Appel, A. P.; Paterlini, A. A.; Sousa, E. P. M. d.; Traina, Caetano, J.; Traina, A. J. M. Biased box sampling - a density-biased sampling for clustering. In: *ACM Symposium on Applied Computing (SAC'2007)*, Seoul, Korea, 2007b, p. 2p.

Bailey, N. T. J. *The mathematical theory of infectious diseases and its applications*. 2nd ed. Hafner Press, 1975.

Bak, P. How nature works : The science of self-organized criticality. 1996.

Bar-Yossef, Z.; Gurevich, M. Random sampling from a search engine's index. *J. ACM*, v. 55, n. 5, p. 1–74, 2008.

Bar-Yossef, Z.; Kumar, R.; Sivakumar, D. Reductions in streaming algorithms, with an application to counting triangles in graphs. In: *SODA*, 2002.

Barabasi, A. L.; Albert, R. Emergence of scaling in random networks. *Science*, v. 286, n. 5439, p. 509–512, 1999.

Disponível em http://view.ncbi.nlm.nih.gov/pubmed/10521342

Barabási, A. L.; Jeong, H.; Néda, Z.; Ravasz, E.; Schubert, A.; Vicsek, T. Evolution of the social network of scientific collaborations. *Physica A*, v. 311, 2002.

Batista, G. E. A. P. A.; Prati, R. C.; Monard, M. C. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, v. 6, n. 1, p. 20–29, 2004.

Becchetti, L.; Boldi, P.; Castillo, C.; Gionis, A. Efficient semi-streaming algorithms for local triangle counting in massive graphs. In: *KDD*, 2008, p. 16–24.

Berkhin, P. *Survey of clustering data mining techniques*. Relatório Técnico, Accrue Software, San Jose, CA, 2002.
Disponível em http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.3739

Bi, Z.; Faloutsos, C.; Korn, F. The DGX distribution for mining massive, skewed data. In: *KDD*, 2001, p. 17–26.

Bollobás, B.; Riordan, O. The diameter of a scale-free random graph. *Combinatorica*, v. 24, n. 1, p. 5–34, 2004.

Borgatti, S. P.; Everett, M. G. Models of core/periphery structures. *Social Networks*, v. 21, n. 4, p. 375 – 395, 2000.
Disponível em http://www.sciencedirect.com/science/article/B6VD1-3YHG8BC-4/2/8fd4a0f3d191bf9fec68e9b7a3f1347d

Braverman, V.; Ostrovsky, R.; Zaniolo, C. Optimal sampling from sliding windows. In: *PODS '09: Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, New York, NY, USA: ACM, 2009, p. 147–156.

Broder, A.; Kumar, R.; Maghoul, F.; Raghavan, P.; Rajagopalan, S.; Stata, R.; Tomkins, A.; Wiener, J. Graph structure in the web: experiments and models.

In: *Proceedings of the Ninth International World-Wide Web Conference (WWW9, Amsterdam, May 15 - 19, 2000 - Best Paper)*, Reston, VA: Foretec Seminars, Inc. (of CD-ROM), 2000.

Disponível em http://www9.org/w9cdrom/160/160.html

Caldarelli, G.; Satorras, P. R.; Vespignani, A. Structure of cycles and local ordering in complex networks. *The European Physical Journal B - Condensed Matter*, v. 38, n. 2, p. 183–186, 2004.

Disponível em http://dx.doi.org/10.1140/epjb/e2004-00020-6

Carpineto, C.; Osiński, S.; Romano, G.; Weiss, D. A survey of web clustering engines. *ACM Comput. Surv.*, v. 41, n. 3, p. 1–38, 2009.

Carvalho, A. C. P. d. L. F. d.; Brayner, A.; Loureiro, A.; Furtado, A. L.; von Staa, A.; de Lucena, C. J. P.; de Souza, C. S.; Medeiros, C. M. B.; Lucchesi, C. L.; e Silva, E. S.; Wagner, F. R.; Simon, I.; Wainer, J.; Maldonado, J. C.; de Oliveira, J. P. M.; Ribeiro, L.; Velho, L.; Gonçalves, M. A.; Calani, M. C.; Mattoso, M.; Ziviani, N.; Navaux, P. O. A.; da Silva Torres, R.; Almeida, V. A. F.; Jr., W. M.; Kohayakawa, Y. Grandes desafios da pesquisa em computação no brasil 2006-2016. 2006.

Chakaravarthy, V. T.; Pandit, V.; Sabharwal, Y. Analysis of sampling techniques for association rule mining. In: *ICDT '09: Proceedings of the 12th International Conference on Database Theory*, New York, NY, USA: ACM, 2009, p. 276–283.

Chakrabarti, D.; Faloutsos, C. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, v. 38, n. 1, p. 2, 2006.

Chakrabarti, D.; Wang, Y.; Wang, C.; Leskovec, J.; Faloutsos, C. Epidemic thresholds in real networks. *ACM Trans. Inf. Syst. Secur.*, v. 10, n. 4, p. 1–26, 2008.

Chakrabarti, D.; Zhan, Y.; Faloutsos, C. R-mat: A recursive model for graph mining. In: *Fourth SIAM International Conference on Data Mining*, 2004.

Disponível em http://www.cs.cmu.edu/~{}christos/PUBLICATIONS/siam04.pdf

Chakravarthy, S.; Pradhan, S. Db-fsg: An sql-based approach for frequent subgraph mining. In: *Database and Expert Systems Applications, 19th International Conference, DEXA 2008, Turin, Italy, September 1-5, 2008. Proceedings*, Springer, 2008, p. 684–692 (*Lecture Notes in Computer Science*, v.5181).

Chaudhuri, S.; Das, G.; Narasayya, V. Optimized stratified sampling for approximate query processing. *ACM Trans. Database Syst.*, v. 32, n. 2, p. 9, 2007.

Chawla, N. V.; Hall, L. O.; Joshi, A. Wrapper-based computation and evaluation of sampling methods for imbalanced datasets. In: *UBDM '05: Proceedings of the 1st international workshop on Utility-based data mining*, New York, NY, USA: ACM, 2005, p. 24–33.

Chawla, N. V.; Japkowicz, N.; Kotcz, A. Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, v. 6, n. 1, p. 1–6, 2004.

Chehreghani, M. H.; Abolhassani, H.; Chehreghani, M. H. Improving density-based methods for hierarchical clustering of web pages. *Data Knowl. Eng.*, v. 67, n. 1, p. 30–50, 2008.

Chen, B.; Haas, P.; Scheuermann, P. A new two-phase sampling based algorithm for discovering association rules. In: *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2002, p. 462–468.

Chen, K.; Bak, P.; Obukhov, S. P. Self-organized criticality in a crack-propagation model of earthquakes. *Phys. Rev. A*, v. 43, n. 2, p. 625–630, 1991.

Chen, L.-S.; Hsu, C.-C.; Chang, Y.-S. Mds: a novel method for class imbalance learning. In: *ICUIMC '09: Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, New York, NY, USA: ACM, 2009, p. 544–549.

Chuang, K.-T.; Chen, H.-L.; Chen, M.-S. Feature-preserved sampling over streaming data. *ACM Trans. Knowl. Discov. Data*, v. 2, n. 4, p. 1–45, 2009.

Chung, F.; Chung, F.; Chung, F.; Lu, L.; Lu, L. The average distances in random graphs with given expected degrees. *Internet Mathematics*, v. 1, p. 15879–15882, 2002.

Chung, F.; Lu, L. *Complex graphs and networks*. American Mathematical Society, 2006.
Disponível em http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0821836579

Chung, F. R. K. *Spectral graph theory*. 1994.

Clauset, A.; Moore, C.; Newman, M. E. J. Hierarchical structure and the prediction of missing links in networks. *Nature*, v. 453, n. 7191, p. 98–101, 2008.
Disponível em http://arxiv.org/abs/0811.0484

Clauset, A.; Newman, M. E. J.; Moore, C. Finding community structure in very large networks. *Physical Review E*, v. 70, p. 066111, 2004.
Disponível em doi:10.1103/PhysRevE.70.066111

Clauset, A.; Shalizi, C. R.; Newman, M. E. J. Power-law distributions in empirical data. *SIAM Review*, v. 51, n. 4, p. 661–704, 2009.
Disponível em http://dx.doi.org/10.1137/070710111

Cochran, W. G. *Sampling techniques*. Second ed. New York: John Wiley and Sons, 1977.

Cohen, R.; Erez, K.; ben Avraham, D.; Havlin, S. Resilience of the internet to random breakdowns. *Phys. Rev. Lett.*, v. 85, n. 21, p. 4626–4628, 2000.

Crovella, M. E.; Bestavros, A. Self-similarity in world wide web traffic: evidence and possible causes. *IEEE/ACM Trans. Netw.*, v. 5, n. 6, p. 835–846, 1997.

Dill, S.; Kumar, R.; Mccurley, K. S.; Rajagopalan, S.; Sivakumar, D.; Tomkins, A. Self-similarity in the web. *ACM Trans. Internet Technol.*, v. 2, n. 3, p. 205–223, 2002.

Dorogovtsev, S. N.; Goltsev, A. V.; Mendes, J. F. F. Pseudofractal scale-free web. *Physical Review E*, v. 65, p. 066122, 2002.
Disponível em http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0112143

Dunbar, R. *Grooming, gossip, and the evolution of language.* Harvard Univ Press, 1998.
Disponível em http://www.isrl.uiuc.edu/~amag/langev/paper/dunbar98groomingGossip.html

Durrett, R. *Random graph dynamics.* Cambridge: Cambridge University Press, 2007.
Disponível em http://www.math.cornell.edu/~durrett/RGD/RGD.html

Erdos, P.; Renyi, A. On the evolution of random graphs. *Publication of the Mathematical Institute of the Hungarian Acadamy of Science*, v. 5, p. 17–67, 1960.

Ertoz, L.; Steinbach, M.; Kumar, V. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In: Barbara, D.; Kamath, C., eds. *Proceedings of the Third SIAM International Conference on Data Mining (SDM 2003)*, Society for Industrial and Applied Mathematics, 2003 (*Proceedings in Applied Mathematics*, v.112).
Disponível em http://www.siam.org/meetings/sdm03/proceedings/sdm03_05.pdf

Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Simoudis, E.; Han, J.; Fayyad, U. M., eds. *Proceedings of the Second International Conference on KDD-96*, AAAI Press, 1996, p. 226–231.

Faloutsos, M.; Faloutsos, P.; Faloutsos, C. On power-law relationships of the internet topology. In: *SIGCOMM 1999*, Cambridge, Massachusetts: ACM Press, 1999, p. 251–262.

FAYYAD, U.; HAUSSLER, D.; STOLORZ, P. Mining scientific data. *Commun. ACM*, v. 39, n. 11, p. 51–57, 1996a.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI Magazine*, v. 17, p. 37–54, 1996b.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. The kdd process for extracting useful knowledge from volumes of data. *Commun. ACM*, v. 39, n. 11, p. 27–34, 1996c.

FERNANDES, S.; KAMIENSKI, C.; KELNER, J.; MARIZ, D.; SADOK, D. A stratified traffic sampling methodology for seeing the big picture. *Comput. Netw.*, v. 52, n. 14, p. 2677–2689, 2008.

FLAXMAN, A.; FRIEZE, A.; FENNER, T. High degree vertices and eigenvalues in the preferential attachment graph. *Internet Math.*, v. 2, n. 1, p. 1–19, 2005.
Disponível em http://www.ams.org/mathscinet-getitem?mr=2166274

FODOR, I. *A survey of dimension reduction techniques*. Relatório Técnico, 2002.

FORTUNATO, S. Community detection in graphs. *Physics Reports*, v. 486, n. 3-5, p. 75–174, 2010.
Disponível em http://dx.doi.org/10.1016/j.physrep.2009.11.002

FRONCZAK, A.; HOLYST, J. A.; JEDYNAK, M.; SIENKIEWICZ, J. Higher order clustering coefficients in barabasi-albert networks. *Physica A*, v. 316, n. 1, p. 688–694, 2002.
Disponível em http://dx.doi.org/10.1016/S0378-4371(02)01336-5

GEHRKE, J.; GINSPARG, P.; KLEINBERG, J. Overview of the 2003 KDD Cup. *SIGKDD Explorations*, v. 5, n. 2, p. 149–151, 2003.

GETOOR, L.; DIEHL, C. P. Introduction to the special issue on link mining. *SIGKDD Explor. Newsl.*, v. 7, n. 2, p. 1–2, 2005a.

GETOOR, L.; DIEHL, C. P. Link mining: a survey. *ACM SIGKDD Explorations*, v. 7, n. 2, p. 3–12, 2005b.

GIRVAN, M.; NEWMAN, M. E. J. Community structure in social and biological networks. In: *Proc. Natl. Acad. Sci. USA*, 2002.

GOOGLE Google programming contest. http://www.google.com/programming-contest/, 2002.

GUHA, S.; RASTOGI, R.; SHIM, K. Cure: an efficient clustering algorithm for large databases. In: *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, New York, NY, USA: ACM Press, 1998, p. 73–84.

GUIMERÀ, R.; SALES-PARDO, M.; AMARAL, L. A. Module identification in bipartite and directed networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, v. 76, n. 3 Pt 2, 2007.
Disponível em http://view.ncbi.nlm.nih.gov/pubmed/17930301

HAAS, P.; SESHADRI, J. F. N. S.; STOKES, L. Sampling-based estimation of the number of distinct values of an attribute. In: *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, Zurich, Suíça: Morgan Kaufmann, 1995, p. 311–322.

HAIR, J. F.; TATHAM, R. L.; ANDERSON, R. E.; BLACK, W. *Multivariate data analysis (5th edition)*. 5th ed. Prentice Hall, 1998.
Disponível em http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0138948585

HAN, J.; KAMBER, M. *Data mining - concepts and techniques*. 1st edition ed. New York: Morgan Kaufmann Publishers, 2000.

HASAN, M. A.; CHAOJI, V.; SALEM, S.; ZAKI, M. Link prediction using supervised learning. In: *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

HAUTAMAKI, V.; KARKKAINEN, I.; FRANTI, P. Outlier detection using k-nearest neighbour graph. In: *17th ICPR 04*, Washington, DC, USA: IEEE, 2004, p. 430–433.

HOLME, P. Core-periphery organization of complex networks. *Phys. Rev. E*, v. 72, n. 4, p. 046111, 2005.

HUANG, Z. Link prediction based on graph topology: The predictive value of the generalized clustering coefficient. In: *Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (LinkKDD2006)*, 2006.
Disponível em http://www.personal.psu.edu/faculty/h/u/huz2/Zan/papers/link.linkKDD06.pdf

HUBERMAN, B. A.; ADAMIC, L. A. Internet: Growth dynamics of the world-wide web. *Nature*, v. 401, n. 6749, p. 131, 1999.
Disponível em http://dx.doi.org/10.1038/43604

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: A review. *ACM Computing Surveys*, v. 31, n. 3, p. 265–323, 1999.

JEONG, H.; TOMBOR, B.; ALBERT, R.; OLTVAI, Z. N.; BARABÁSI, A.-L. The large-scale organization of metabolic networks. *Nature*, v. 407, 2000.

JOSHI, S.; JERMAINE, C. Sampling-based estimators for subset-based queries. *The VLDB Journal*, v. 18, n. 1, p. 181–202, 2009.

KASHIMA, H.; KATO, T.; YAMANISHI, Y.; SUGIYAMA, M.; TSUDA, K. Link propagation: A fast semi-supervised learning algorithm for link prediction. In: *SDM*, SIAM, 2009, p. 1099–1110.
Disponível em http://dblp.uni-trier.de/db/conf/sdm/sdm2009.html#KashimaKYST09

KAUFMAN, L.; ROUSSEEUW, P. J. *Finding groups in data: An introduction to cluster analysis (wiley series in probability and statistics)*. Wiley-Interscience, 2005.
Disponível em http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0471735787

KESTEN, H. The critical probability of bond percolation on the square lattice equals 1/2. *Communications in Mathematical Physics*, v. 74, p. 41–59, 1980.

Kleinberg, J. M.; Kumar, R.; Raghavan, P.; Rajagopalan, S.; Tomkins, A. S. The web as a graph: measurements, models, and methods. 1999.

Klimt, B.; Yang, Y. Introducing the enron corpus. In: *CEAS*, 2004.

Kollios, G.; Gunopulos, D.; Koudas, N.; Berchtold, S. Efficient biased sampling for approximate clustering and outlier detection in large data sets. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, v. 15, n. 5, p. 1170–1187, 2003.

Kong, J. S.; Rezaei, B. A.; Sarshar, N.; Roychowdhury, V. P.; Boykin, P. O. Collaborative spam filtering using e-mail networks. *Computer*, v. 39, n. 8, p. 67–73, 2006.

Kotsiantis, S.; Kanellopoulos, D. Discretization techniques: A recent survey. *International Transactions on Computer Science and Engineering*, v. 32, n. 1, p. 47–58, 2006.

Kriegel, H.-P.; Kröger, P.; Zimek, A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data*, v. 3, n. 1, p. 1–58, 2009.

Krishnamurthy, V.; Faloutsos, M.; Chrobak, M.; Cui, J.-H.; Lao, L.; Percus, A. G. Sampling large internet topologies for simulation purposes. *Computer Networks*, v. 51, n. 15, p. 4284–4302, 2007.

Kumar, R.; Raghavan, P.; Rajagopalan, S.; Sivakumar, D.; Tomkins, A.; Upfal, E. Stochastic models for the web graph. In: *Proc. 41st IEEE Symp. on Foundations of Computer Science*, 2000.

Kumar, R.; Raghavan, P.; Rajagopalan, S.; Tomkins, A. Trawling the web for emerging cyber-communities. *Comput. Networks*, v. 31, n. 11-16, p. 1481–1493, 1999. Disponível em http://dx.doi.org/10.1016/S1389-1286(99)00040-7

KUNEGIS, J.; LOMMATZSCH, A. Learning spectral graph transformations for link prediction. In: *Proc. Int. Conf. in Machine Learning*, 2009.
Disponível em http://www.dai-labor.de/index.php?id=1034&pubID=419

KURAMOCHI, M.; KARYPIS, G. An efficient algorithm for discovering frequent subgraphs. *IEEE Transactions on Knowledge and Data Engineering*, v. 16, n. 9, p. 1038–1051, 2004.

LATAPY, M. Main-memory triangle computations for very large (sparse (power-law)) graphs. *Theor. Comput. Sci.*, v. 407, n. 1-3, p. 458–473, 2008.

LATHIA, N.; HAILES, S.; CAPRA, L. kNN CF: a temporal social network. In: *ACM RecSys08*, New York, NY, USA: ACM, 2008, p. 227–234.

LEE, C.-H.; ZAÏANE, O. R.; PARK, H.-H.; HUANG, J.; GREINER, R. Clustering high dimensional data: A graph-based relaxed optimization approach. *Inf. Sci.*, v. 178, n. 23, p. 4501–4511, 2008.

LESKOVEC, J.; FALOUTSOS, C. Sampling from large graphs. *KDD*, p. 631–636, 2006.

LESKOVEC, J.; KLEINBERG, J.; FALOUTSOS, C. Graphs over time: densification laws, shrinking diameters and possible explanations. In: *Proc. of ACM SIGKDD*, Chicago, Illinois, USA: ACM Press, 2005, p. 177–187.

LESKOVEC, J.; KLEINBERG, J. M.; FALOUTSOS, C. Graph evolution: Densification and shrinking diameters. *ACM TKDD*, v. 1, n. 1, p. 1 – 40, 2007a.

LESKOVEC, J.; LANG, K. J.; DASGUPTA, A.; MAHONEY, M. W. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *CoRR*, v. abs/0810.1355, 2008.

LESKOVEC, J.; MCGLOHON, M.; FALOUTSOS, C.; GLANCE, N. S.; HURST, M. Patterns of cascading behavior in large blog graphs. In: *SDM*, 2007b.

Liben-Nowell, D.; Kleinberg, J. The link prediction problem for social networks. In: *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, New York, NY, USA: ACM, 2003, p. 556–559.

Ling, Y. An evaluation of sampling-based size estimation methods for selections in database systems. In: *Proceedings of the International Conference on Data Engineering (ICDE)*, Taipei, Taiwan: IEEE Computer Society, 1995, p. 532–539.

Lu, L.; Zhou, T. Role of weak ties in link prediction of complex networks. In: *Proceedings of the 1st ACM International Workshop on Complex Networks in Information and Knowledge Management (CNIKM)*, Hong Kong, China, 2009.

Lu, Z.; Rughani, A. I.; Tranmer, B. I.; Bongard, J. Informative sampling for large unbalanced data sets. In: *GECCO '08: Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation*, New York, NY, USA: ACM, 2008, p. 2047–2054.

MacQueen, J. B. Some methods for classification and analysis of multivariate observations. In: Cam, L. M. L.; Neyman, J., eds. *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, 1967, p. 281–297.

Maier, M.; Hein, M.; von Luxburg, U. Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters. *Theor. Comput. Sci.*, v. 410, n. 19, p. 1749–1764, 2009.

McGlohon, M.; Akoglu, L.; Faloutsos, C. Weighted graphs and disconnected components: patterns and a generator. In: *KDD*, 2008, p. 524–532.

Mihail, M.; Papadimitriou, C. H. On the eigenvalue power law. In: *RANDOM '02: Proceedings of the 6th International Workshop on Randomization and Approximation Techniques*, London, UK: Springer-Verlag, 2002, p. 254–262.

Milgram, S. The small world problem. *Psychology Today*, v. 2, p. 60–67, 1967.

NANOPOULOS, A.; THEODORIDIS, Y.; MANOLOPOULOS, Y. Indexed-based density biased sampling for clustering applications. *Data and Knowledge Engineering*, v. 57, n. 1, p. 37–63, 2006.

NEWMAN, M. E. J. The structure and function of complex networks. *SIAM Review*, v. 45, p. 167–256, 2003.

NEWMAN, M. E. J. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, v. 46, p. 323, 2005.
Disponível em doi:10.1080/00107510500052444

NEWMAN, M. E. J. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, v. 103, n. 23, p. 8577–8582, 2006.
Disponível em http://dx.doi.org/10.1073/pnas.0601602103

NEWMAN, M. E. J.; BARABÁSI, A. L.; WATTS, D. J., eds. *The structure and dynamics of networks*. Princeton University Press, 2006.

NG, R. T.; HAN, J. Efficient and effective clustering methods for spatial data mining. In: BOCCA, J. B.; JARKE, M.; ZANIOLO, C., eds. *International Conference on Very Large Databases (VLDB)*, Santiago de Chile, Chile: Morgan Kaufmann, 1994, p. 144–155.

OLKEN, F.; ROTEM, D. Random sampling from databases - a survey. 1995.

PAGE, L.; BRIN, S.; MOTWANI, R.; WINOGRAD, T. *The pagerank citation ranking: Bringing order to the web*. Relatório Técnico, Stanford Digital Library, 1998.

PALMER, C. R.; FALOUTSOS, C. Density biased sampling: An improved method for data mining and clustering. In: *ACM SIGMOD International Conference on Management of Data*, San Diego, CA: ACM Press, 2000, p. 82–92.

PALMER, C. R.; GIBBONS, P. B.; FALOUTSOS, C. Anf: A fast and scalable tool for data mining in massive graphs. In: *ACM SIGKDD International Conference on*

Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada: ACM Press, 2002, p. 81–90.

Paterson, M.; Yao, F. F. On nearest-neighbor graphs. In: *19th ICALP 92*, London, UK: Springer-Verlag, 1992, p. 416–426.

Pennock, D. M.; Flake, G. W.; Lawrence, S.; Glover, E. J.; Giles, C. L. Winners don't take all: Characterizing the competition for links on the Web. *Proceedings of the National Academy of Sciences*, v. 99, n. 8, p. 5207–5211, 2002.

Popescul, A.; Popescul, R.; Ungar, L. H. Statistical relational learning for link prediction. 2003.

Radicchi, F.; Castellano, C.; Cecconi, F.; Loreto, V.; Parisi, D. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, v. 101, n. 9, p. 2658–2663, 2004.
Disponível em http://dx.doi.org/10.1073/pnas.0400054101

Ravasz, E.; Barabási, A.-L. Hierarchical organization in complex networks. *Phys. Rev. E*, v. 67, n. 2, p. 026112, 2003.

Ravasz, E.; Somera, A. L.; Mongru, D. A.; Oltvai, Z. N.; Barabasi, A. L. Hierarchical organization of modularity in metabolic networks. *Science*, v. 297, n. 5586, p. 1551–1555, 2002.
Disponível em http://dx.doi.org/10.1126/science.1073374

Redner, S. How popular is your paper? an empirical study of the citation distribution. 1998.
Disponível em http://arxiv.org/abs/cond-mat/9804163

Reka, A.; Barabási Statistical mechanics of complex networks. *Rev. Mod. Phys.*, v. 74, p. 47–97, 2002.
Disponível em http://arxiv.org/abs/cond-mat/0106096

Rezende, S. O. *Sistemas inteligentes - fundamentos e aplicações*. Manole, 2003.

RICHARDSON, M.; AGRAWAL, R.; DOMINGOS, P. *Trust management for the semantic web*, v. 2870. 351–368 p., 2003.
Disponível em http://www.metapress.com/link.asp?id=DM0N8R10RL4U430Q

RIPEANU, M.; FOSTER, I.; IAMNITCHI, A. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal*, v. 6, n. 1, 2002.

SCHROEDER, M. *Fractals, chaos, power laws: Minutes from an infinite paradise*. W. H. Freeman, 1991.

SIGANOS, GEORGOS; SUDHIR L TAURO, M. F. Jellyfish: A conceptual model for the as internet topology. In: *Journal of Communications and Networks*, 2006, p. 339Ű350.

TASKAR, B.; WONG, M.; ABBEEL, P.; KOLLER, D. Link prediction in relational data. 2004.
Disponível em http://citeseer.ist.psu.edu/630066.html

TAURO, S. L.; PALMER, C.; SIGANOS, G.; FALOUTSOS, M. A simple conceptual model for the internet topology. In: *Global Internet, San Antonio, Texas*, 2001.

TOIVONEN, H. Sampling large databases for association rules. In: VIJAYARAMAN, T. M.; BUCHMANN, A. P.; MOHAN, C.; SARDA, N. L., eds. *The International Journal on Very Large Databases*, Morgan Kaufmann Publishers, 1996, p. 134–145.

TRAINA, C. J.; TRAINA, A. J. M.; WU, L.; FALOUTSOS, C. Fast feature selection using fractal dimension. In: BECKER, K.; DE SOUZA, A. A.; DE SOUZA FERNANDES, D. Y.; BATISTA, D. C. F., eds. *XV Simpósio Brasileiro de Banco de Dados, 2-4 Outubro 2000, João Pessoa, Paraíba, Brasil, Anais.*, 2000, p. 158–171.

TSOURAKAKIS, C. E. Fast counting of triangles in large real networks without counting: Algorithms and laws. In: *ICDM '08*, Washington, DC, USA: IEEE Computer Society, 2008, p. 608–617.

Tsourakakis, C. E.; Kang, U.; Miller, G. L.; Faloutsos, C. Doulion: counting triangles in massive graphs with a coin. In: *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA: ACM, 2009, p. 837–846.

Watts, D. J.; Strogatz, S. H. Collective dynamics of 'small-world' networks. *Nature*, v. 393, n. 6684, p. 440–442, 1998.
Disponível em http://dx.doi.org/10.1038/30918

Waxman, B. M. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, v. 6, n. 9, 1988.

Winick, J.; Jamin, S. *Inet-3.0: Internet Topology Generator.* Relatório Técnico CSE-TR-456-02, University of Michigan, Ann Arbor, 2002.
Disponível em http://topology.eecs.umich.edu/inet/

Wu, J.; Zhang, Y.; Mao, Z. M.; Shin, K. G. Internet routing resilience to failures: analysis and implications. In: *CoNEXT '07: Proceedings of the 2007 ACM CoNEXT conference*, New York, NY, USA: ACM, 2007, p. 1–12.

Xiong, H.; Pandey, G.; Steinbach, M.; Kumar, V. Enhancing data analysis with noise removal. *IEEE Trans. on Knowl. and Data Eng.*, v. 18, n. 3, p. 304–319, 2006.

Yan, X.; Han, J. gSpan: Graph-based substructure pattern mining. In: *ICDM*, 2002.

Zaki, M. J.; Parthasarathy, S.; Li, W.; Ogihara, M. *Evaluation of sampling for data mining of association rules.* Relatório Técnico TR617, 1996.
Disponível em citeseer.ist.psu.edu/zaki96evaluation.html

Zhang, C.; Yang, Q.; Liu, B. Guest editors' introduction: Special section on intelligent data preparation. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, v. 17, n. 9, p. 1163–1165, 2005.

Zhang, T.; Ramakrishnan, R.; Livny, M. Birch: An efficient data clustering method for very large databases. In: Jagadish, H. V.; Mumick, I. S., eds. *ACM*

SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada: ACM Press, 1996, p. 103–114 (*SIGMOD Record 25(2)*, v.1).