CEFET/RJ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO

UM ESTUDO COMPUTACIONAL COMPARATIVO ENTRE ALGORITMOS DE AGRUPAMENTO E DE DETECÇÃO DE COMUNIDADES CONSIDERANDO DADOS DA EDUCAÇÃO

Daiana Medeiros da Silva

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenheira de Produção e Sistemas, do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ, como parte dos requisitos necessários à obtenção do título de Mestre em Engenheira de Produção e Sistemas.

Orientadora: Carla Silva Oliveira
Coorientador: José André de Moura Brito

Rio de Janeiro
Janeiro de 2021

UM ESTUDO COMPUTACIONAL COMPARATIVO ENTRE ALGORITMOS DE AGRUPAMENTO E DE DETECÇÃO DE COMUNIDADES CONSIDERANDO DADOS DA EDUCAÇÃO

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenheira de Produção e Sistemas, do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ, como parte dos requisitos necessários à obtenção do título de Mestre em Engenheira de Produção e Sistemas.

Daiana Medeiros da Silva

Aprovado por:

_____

Presidente, Professora Dra. Carla Silva Oliveira (ENCE) (Orientadora)

_____

Professor Dr. José André de Moura Brito (ENCE) (Coorientador)

_____

Professor Dr. Celso Marques da Silva Junior (CEFET/RJ)

_____

Professor Dr. Gustavo Silva Semaan (UFF)

Rio de Janeiro
Janeiro de 2021

Ficha catalográfica elaborada pela Biblioteca Central do CEFET/RJ

S586 Silva, Daiana Medeiros da
Um estudo computacional comparativo entre algoritmos de agrupamento e de detecção de comunidades considerando dados da educação / Daiana Medeiros da Silva — 2021.
77f. + anexo : il. color. , enc.

Dissertação (Mestrado) Centro Federal de Educação Tecnológica Celso Suckow da Fonseca , 2021.
Bibliografia : f. 72-77
Orientadora: Carla Silva Oliveira
Coorientador: José André de Moura Brito

1. Algoritmos computacionais. 2. Teoria dos grafos. 3. Engenharia – Processamento de dados 4. Análise espectral. 5. Base de dados. I. Oliveira, Carla Silva (Orient.). II. Brito, José André de Moura (Coorient.). III. Título.

CDD 005.1

Elaborada pela bibliotecária Tania Mello – CRB/7 nº 5507/04

AGRADECIMENTOS

Agradeço a Deus que é a essência da minha vida e a razão do meu existir. Obrigada meu Deus por ter me dado forças e saúde para chegar até aqui. Obrigada por cada lágrima derramada que serviram de aprendizado e superação nessa longa caminhada.

Quero agradecer a todas as pessoas que foram importantes na minha vida e me ajudaram nesta jornada. Cada um teve sua participação em algum momento seja na forma de apoio, conselho, ajuda ou orientação.

Ao meu marido que esteve ao meu lado em todos os momentos, muitos de incertezas e de lágrimas. Obrigada por sempre me ouvir, mesmo que o assunto era sempre o mesmo: “não vou conseguir terminar o mestrado”. Obrigada por ser este companheiro e parceiro para todas as horas. Muito obrigada por fazer tudo que tivesse ao seu alcance para me dar o suporte necessário para concluir este objetivo. Agradeço também aos meus filhos: Mateus e Sophia embora muito pequenos me deram uma força extraordinária para seguir em frente.

A minha mãe, Norma, por todo apoio, carinho e dedicação.

Aos meus orientadores, muito obrigada professora Carla e professor José André pelo conhecimento compartilhado, pelos conselhos e por toda a paciência que tiveram na minha orientação.

Aos membros da banca professores Gustavo e Celso por todas as contribuições que foram enriquecedoras.

Aos docentes do CEFET-RJ, obrigada por toda contribuição e ensinamento. Agradeço ao professor Augusto Reis pelo apoio e ajuda. Agradeço a todos os funcionários da secretária, ao Fernando por toda ajuda.

“Bem sei eu que tudo podes,
e que nenhum dos teus propósitos
pode ser impedido.”(*Jó 42:2)*

RESUMO

UM ESTUDO COMPUTACIONAL COMPARATIVO ENTRE ALGORITMOS DE AGRUPAMENTO E DE DETECÇÃO DE COMUNIDADES CONSIDERANDO DADOS DA EDUCAÇÃO

Daiana Medeiros da Silva

Orientadores:

Carla Silva Oliveira, D.Sc.
José André de Moura Brito, D.Sc.

Resumo da Dissertação de Mestrado submetida ao Programa de Pós-graduação em Engenheira de Produção e Sistemas, do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ, como parte dos requisitos necessários à obtenção do título de Mestre em Engenheira de Produção e Sistemas.

Esta dissertação tem por objetivo geral realizar um estudo computacional comparativo entre três algoritmos de agrupamento (*k-means*, *PAM* e *CLARA*) e dois algoritmos de Detecção de Comunidades (*Fast Greedy* e *Walktrap*), considerando dados da área da Educação. Para analisar as soluções produzidas pelos cinco algoritmos, foram utilizados dois índices associados, respectivamente, aos critérios de validação relativo e externo, sejam eles: Silhueta e Rand. Para tal, foi implementado um algoritmo para construção de redes, que foram produzidas a partir das bases de dados reais de duas maneiras distintas, a saber: através dos quartis das distâncias dos objetos e da Árvore Geradora Mínima. Dentre os principais resultados, destacam-se aqueles provenientes dos algoritmos *PAM* e *CLARA*, que quando aplicados a um conjunto de base de dados de porte pequeno e médio não diferem quanto aos seus resultados, em relação ao índice de Silhueta. Em relação ao índice de Rand, quando é comparado o desempenho dos algoritmos *PAM* e *CLARA*, frente aos algoritmos de Detecção de Comunidades: *Fast Greedy* e *Walktrap*, apresentou maior similaridade entre os grupos formados pelo algoritmo *CLARA* e as comunidades do algoritmo *Fast Greedy*, do que os grupos formados pelo algoritmo *PAM* e as comunidades produzidas pelo algoritmo *Fast Greedy*.

Palavras-chave:
Análise de Agrupamentos; Detecção de Comunidades; *k-means*; *PAM*; *Fast Greedy*.

Rio de Janeiro
Janeiro de 2021

ABSTRACT

A COMPARATIVE COMPUTATIONAL STUDY BETWEEN CLUSTER ANALYSIS AND COMMUNITY DETECTION ALGORITHMS CONSIDERING EDUCATION DATA

Daiana Medeiros da Silva

Advisors:

Carla Silva Oliveira, D.Sc.
José André de Moura Brito, D.Sc.

Abstract of Dissertation submitted to Programa de Pós-graduação em Engenheira de Produção e Sistemas of Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ, as partial of fulfillment of the requirements for the degree Master.

The general objective of this dissertation is to perform a comparative computational study between three clustering algorithms (k-means, PAM and CLARA) and two Community Detection algorithms (Fast Greedy and Walktrap), considering data from the education area. To analyze the solutions produced by the five algorithms, two indexes associated, respectively, with the relative and external validation criteria were used, be they: Silhouette and Rand. For this, an algorithm was implemented for network construction, which were produced from the real databases in two different ways, that is: through the quartiles of the distances of the objects and the Minimum Generating Tree. Among the main results, we highlight those from the PAM and CLARA algorithms, which when applied to a small and medium-sized database set do not differ in relation to their results, in relation to the Silhouette index. In relation to the Rand index, when comparing the performance of the PAM and CLARA algorithms, compared to the Community Detection algorithms Fast Greedy and Walktrap, it presented greater similarity between the groups formed by the CLARA algorithm and the communities of the Fast Greedy algorithm, than the groups formed by the PAM algorithm and the communities produced by the Fast Greedy algorithm.

Keywords:
Cluster Analysis; Community Detection; k-means; PAM; Fast Greedy

Rio de Janeiro
Janeiro 2021

Lista de Abreviaturas e Siglas

AAG	Análise de Agrupamentos
AGM	Árvore Geradora Mínima
CLARA	*Clustering Large Applications*
CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico
ENADE	Exame Nacional de Desempenho do Estudantes
ENEM	Exame Nacional do Ensino Médio
INEP	Instituto Nacional de Estudos e Pesquisa Educacionais Anísio Teixeira
KDD	*Knowledge Discovey in Databases*
PA	Problema de Agrupamento
PAA	Problema de Agrupamento Automático
PAC	Problema de Agrupamento Clássico
PAM	*Partitioning Around Medoids*

Lista de Figuras

Figura 1: Grafos *G* e *H*..... 5

Figura 2: Grafo $\boldsymbol{G1}$..... 6

Figura 3: Grafo $\boldsymbol{G2}$..... 8

Figura 4: Aplicação do Algoritmo PRIM..... 9

Figura 5: AGM 9

Figura 6:Similaridade entre os grupos e dentro dos grupos. 11

Figura 7: Etapas do KDD. 15

Figura 8: Etapas do processo de agrupamento 16

Figura 9: Duas possíveis soluções de agrupamento com 2 *medoids* 23

Figura 10: Exemplo de três comunidades de uma rede..... 25

Figura 11: Grafo $\boldsymbol{G3}$..... 28

Figura 12: Comunidades $\boldsymbol{C1}$ e $\boldsymbol{C2}$ 28

Figura 13: Exemplo de dendograma ou árvore hierárquica 31

Figura 14: Grafo $\boldsymbol{G4}$..... 35

Figura 15: Etapas da Formação de duas comunidades utilizando o algoritmo Walktrap. 37

Figura 16: Representação da aplicação do algoritmo Walktrap 38

Figura 17: Dendograma utilizando Algoritmo Walktrap 38

Figura 18: Etapas da metodologia utilizada 47

Figura 19: Comparação dos métodos através do índice de Silhueta 53

Figura 20: Comparação dos Algoritmos através do índice de Rand..... 54

Figura 21: Percentual das bases de dados para as quais foram produzidos valores de Rand acima a 0,7 56

Figura 22:Comparação do algoritmo Clara versus PAM em relação ao índice de Silhueta. Medida de dissimilaridade: Euclidiana e grafo gerado via quartil. 57

Figura 23:Comparação do algoritmo PAM versus o algoritmo CLARA utilizando a distância de Manhattan..... 58

Figura 24:Valores do índice de Silhueta segundo valores dos grupos obtidos.....60

Figura 25:Valores do índice de Silhueta segundo valores dos grupos obtidos.....61

Figura 26:Comparação dos algoritmos *PAM* e *CLARA* em relação índice de Rand 62

Figura 27: Comparação dos algoritmos através do índice de Silhueta com criação dos grafos via quartis e AGM. 64

Lista de Tabelas

Tabela 1: Número de soluções para o problema de agrupamento usando o número de *Stirling* 14

Tabela 2:Exemplo de variáveis quantitativas com ordens de grandeza distintas 19

Tabela 3: Média e desvio-padrão das variáveis Renda e Idade 19

Tabela 4: Variáveis padronizadas 19

Tabela 5: Verificação se os vértices $\boldsymbol{vi}$ e $\boldsymbol{vj}$ estão na mesma comunidade 29

Tabela 6: Autovalores λ da matriz P e os respectivos autovetores da matriz P. 35

Tabela 7: Autovalores da matriz P e os respectivos autovetores da matriz $\boldsymbol{PT}$. 36

Tabela 8: Valores das distâncias r e $\Delta\sigma$ 37

Tabela 9: Representação dos objetos nos grupos 41

Tabela 10: Exemplo da comparação dos pares entre os grupos e as partições 41

Tabela 11: Grupos exemplos 43

Tabela 12: Tempos computacionais dos algoritmos utilizando: quartil e AGM e a distância Euclidiana 66

Tabela 13: Tempos computacionais dos algoritmos utilizando: Quartil e AGM e a distância de Manhattan 68

Lista de Quadros

Quadro 1: Tipos de atributos 18

Quadro 2: Interpretação do Coeficiente de Silhueta 43

Quadro 3: Quantidade de registros da Base de Dados 51

Quadro 4: Descrição das configurações do experimento I..... 52

Quadro 5: Comparação dos Algoritmos através do índice de Rand 77

Quadro 6: Comparação dos tempos Computacionais de Execução dos Algoritmos..... 78

Quadro 7: Comparação dos tempos Computacionais de Execução dos Algoritmos..... 79

Quadro 8: Comparação dos tempos Computacionais de Execução dos Algoritmos..... 80

Quadro 9: Comparação dos tempos Computacionais de Execução dos Algoritmos..... 81

Sumário

Capítulo I - Introdução 1

I.1 Motivação 2
I.2 Objetivo Geral 3
I.2.1 Objetivos específicos 3
I.3 Organização da Dissertação 4

Capítulo II - Conceitos básicos de Grafos 5

II.1 Teoria dos Grafos 5

Capítulo III – Análise de Agrupamentos 10

III.1 Análise de Agrupamentos 10
III.2 Problema de Agrupamento 11
III.3 Processo de Descoberta de Conhecimento em Bases de Dados 14
III.3.1 Procedimento para formação de agrupamentos 16
III.3.2 Seleção e Tratamento de dados 17
III.4 Algoritmos 20
III.4 .1 Algoritmo *k-means* 21
III.4 .2 Algoritmo *PAM* (*Partitioning Around Medoids*) 22
III.4 .3 Algoritmo *CLARA* (*Clustering Large Applications*) 23

Capítulo IV – Detecção de Comunidades em Redes 25

IV.1 Detecção de Comunidades 25
IV.2 Modularidade 27
IV.3 Algoritmos de Detecção de Comunidades 29
IV. 3 .1 Algoritmo *Fast Greedy* 30
IV. 3 .2 Algoritmo *Walktrap* 33

Capítulo V – Critério de Validação 39

V.1 Conceitos básicos 39
V.1.1 Índice de Rand 39
V.1.3 Índice de Silhueta 41

Capítulo VI – Metodologia Utilizada 45

VI.1 Procedimentos para análise de dados 45

Capítulo VII – Resultados e Análises 48

VII.1 Experimentos 48

VII.1.1 Base de Dados 49
VII.2 Análises 52
VII.2.1 Apresentação dos resultados 52
VII.2.2 Apresentação dos resultados – Comparação dos algoritmos de agrupamento e de Detecção de Comunidades 53
VII.2.3 Apresentação dos resultados – Comparação dos algoritmos *PAM* e *CLARA* 56
VII.2.4 Apresentação dos resultados – Comparação da AGM e Quartis 63
VII.2.5 Apresentação dos resultados – Comparação dos tempos computacionais 65
VII.3 Conclusões 69
REFERÊNCIAS 71
Anexo I 77
Anexo II 78
Anexo III 79
Anexo III 80
Anexo IV 81

Capítulo I - Introdução

A prática de agrupar os objetos de acordo com as suas semelhanças é a base de grande parte da ciência, ao passo que, organizar dados em agrupamentos é um dos modos mais fundamentais que configura a compreensão e aprendizado (JAIN & DUBES, 1988). Entretanto, o problema de organizar tais dados não é uma tarefa trivial, tendo em vista que, diante do volume expressivo de dados que é produzido diariamente e disponibilizado, por exemplo, na internet, faz-se necessário uma maneira automática de organizar, separar e classificar tais dados, de forma a produzir informações relevantes para a realização de análises por parte de órgãos privados e públicos.

Uma possível alternativa para tratar esse problema diz respeito à aplicação da Análise de Agrupamentos, que corresponde a uma técnica de análise multivariada que agrega um conjunto de algoritmos que possibilitam alocar os objetos de uma base de dados em grupos sem a necessidade, a priori, do conhecimento de suas classes ou suas categorias (JAIN & DUBES, 1988).

Existem diversas aplicações reais concernentes às áreas de psicologia, ciências sociais, biologia, medicina, marketing, estatística, mineração de dados (JAIN & DUBES, 1988), (KUMAR; STEINBACH; TAN, 2009), (HAN; KAMBER; PEI, 2012), dentre outras, que podem resolvidas mediante a aplicação da análise de agrupamentos.

Uma alternativa muito utilizada para analisar dados diz respeito à aplicação de algoritmos de Detecção de Comunidades em Redes. Adicionalmente, a utilização de técnicas de Detecção de Comunidades em Redes tem se mostrado uma valiosa ferramenta de análise, cuja aplicação tem sido estudada tanto pela academia quanto pela indústria (WANG et. al., 2015); (DIAS et. al., 2019). Mais especificamente, redes são definidas como “sistemas físicos, biológicos ou sociais caracterizados por um conjunto de entidades bem definidas que se integram e interagem dinamicamente entre si” (TEIXEIRA; LIMA; ABREU, 2008).

Considerando essa definição, pode-se representar**,** através de redes, diversas estruturas de dados: redes de coautoria, redes de pesquisa, redes de esgoto, redes de telefonia, redes de computadores, entre outras. As redes podem ser modeladas a partir da utilização de uma estrutura matemática denominada grafo, o qual é formado por dois conjuntos: os vértices/nós que correspondem aos objetos de um grafo/rede e as arestas

que representam às ligações/conexões entre os vértices, ((COPPIN, 2017) apud (SOUSA; PRATA, 2019)). Através dessa estrutura, pode-se obter variadas informações quanto à topologia dessas redes como, por exemplo, o fluxo de informação que passa pelos nós e/ou clusters de nós (COSTA et al., 2013).

Por sua vez, estudar e analisar redes modeladas por grafos com substancial número de vértices e arestas como, por exemplo, a rede de contato do *twitter*, com mais de 50 milhões de indivíduos, ou mesmo uma rede de neurônios, demanda, com frequência, a utilização de métodos associados à Teoria dos Grafos. Estes métodos também são constantemente utilizados para análises de redes de coautorias e colaborações (PROCACI, 2015); (MAGALHÃES, 2013); (SEMAN et al., 2020). Mais especificamente, no exemplo do *twitter*, cada um dos 50 milhões de indivíduos é representado por um vértice no grafo e cada ligação/conexão entre os indivíduos corresponde a uma aresta do grafo. Neste sentido, as redes reais, representadas por grafos, apresentam, naturalmente, uma alta dimensionalidade no que diz respeito ao número de vértices e de arestas que devem ser analisados. Ainda neste sentido, após a representação de uma rede mediante um grafo, um dos objetivos é analisar as relações, entre os vértices/objetos que compõem tal rede, por exemplo, considerando uma maior importância de um ou mais vértices em detrimento a outros.

Em vista disso, um dos problemas associados à área análise de redes via Teoria dos Grafos é o de detecção de comunidades, ou seja, o particionamento do grafo que modela a rede em clusters (MENTZ et al., 2007).

Por esta razão, os algoritmos de Análise de Agrupamentos e de Detecção de Comunidades podem ser utilizados como uma importante ferramenta auxiliar de análise, em diferentes tipos de redes tais como: Qualidade, como instrumento de apoio à melhoria da qualidade dos serviços aos clientes (MOORI; MARCONDES; AVILA, 2002); Educação, como instrumento de apoio na avaliação da formação técnica profissionalizante para o setor industrial (SANTOS et al., 2014); *World Wide Web*, como instrumento de apoio para melhorar os resultados disponibilizados por sites de busca (HAMMOUDA, 2002).

I.1 Motivação

Estudos que utilizam algoritmos de Análise de Agrupamentos e de Detecção de

Comunidades, a fim de auxiliar pesquisas e análises têm crescido em diversas áreas. Em específico, na área da educação há diversos desafios que podem ser solucionados diante destas ferramentas de análise, uma vez que, ao lidar com uma grande quantidade de dados e, concomitantemente, conseguir extrair informações precisas para tomadas de decisões não é uma tarefa trivial.

A exemplo, extrair informações relevantes de base dados, como do Exame Nacional do Ensino Médio (ENEM) ou do Exame Nacional de Desempenho de Estudantes (ENADE) e, mapear o aprendizado dos estudantes, realizar comparativos entre escolas/universidades públicas e privados, identificar a deficiência dos estudantes em uma disciplina específica ou ainda mapear: a evasão escolar, perfil de ingressos e egressos, perfil socioeconômico etc., são informações importantes que servem de apoio na tomada de decisões de gestores da área educacional. Entretanto, para obter tais informações é necessário aplicar estas ferramentas a fim encontrar grupos de alunos com uma maior similaridade.

Dada a importância e do crescente volume de dados desta área, a presente dissertação utilizou dados oriundos da área educacional para realizar alguns experimentos e análises. Ademais, para extrair informações que sejam concisas e úteis, torna-se necessário obter bases de dados que contenham informações que tratem de um mesmo assunto.

I.2 Objetivo Geral

Esta dissertação tem por objetivo geral realizar um estudo computacional comparativo entre três algoritmos de agrupamento (*k-means*, *PAM* e *CLARA*) e dois algoritmos de Detecção de Comunidades (*Fast Greedy* e *Walktrap*), considerando dados da área da Educação. A comparação entre os algoritmos, quanto às estruturas de grupos e comunidades formados, será realizada através do cálculo de índices - Silhueta e Rand, associados aos critérios de validação.

I.2.1 Objetivos específicos

De modo a alcançar o objetivo geral proposto, os objetivos específicos desta dissertação são:

- Construir redes mediante uso das bases de dados reais da Educação de duas

maneiras distintas, a saber: usando os quartis das distâncias dos objetos associados às bases de dados e a árvore geradora mínima;

- Segmentar as bases de dados a partir da aplicação dos os algoritmos de agrupamento *k-means*, *CLARA* e *PAM* e segmentar as bases de dados a partir da aplicação dos algoritmos de detecção de comunidades *Fast Greedy* e *Walktrap*;

- Avaliar as soluções produzidas pelos cinco algoritmos, tendo por base o cálculo de índices associados aos critérios de validação relativo e externo, sejam eles: Silhueta e índice de Rand;

- Comparar e avaliar os resultados produzidos pelos algoritmos de agrupamento e de detecção de comunidades;

- Verificar se a troca do algoritmo *PAM* pelo *CLARA* acarretaria alguma mudança significativa de qualidade, no que tange aos índices de Silhueta e Rand;

- Avaliar o impacto nas soluções produzidas pelos algoritmos de agrupamento e de detecção de comunidades, considerando, como entrada, dois tipos de rede: árvore geradora mínima (AGM) e quartis;

I.3 Organização da Dissertação

Esta dissertação está organizada em sete capítulos. No Capítulo II são apresentados os conceitos básicos da Teoria dos Grafos. No Capítulo III são apresentados os conceitos básicos de Análise de Agrupamentos de dados, incluindo uma breve descrição dos três algoritmos de agrupamento que foram utilizados neste trabalho: *k-means*, *CLARA* e *PAM*. No Capítulo IV, são apresentados os conceitos básicos de Detecção de Comunidades em Redes e os algoritmos de detecção de comunidades: *Fast Greedy* e *Walktrap*. No Capítulo V, são descritos os critérios de validação. No Capítulo VI, é apresentada a metodologia da pesquisa. Por fim, o Capítulo VII traz a descrição dos experimentos, a análise e as conclusões desta dissertação.

Capítulo II - Conceitos básicos de Grafos

Neste capítulo são apresentados os conceitos básicos da Teoria dos Grafos e do Algoritmo de Prim necessários para a compreensão do texto e do problema que foi o objeto de estudo desta dissertação. As principais fontes bibliográficas para o desenvolvimento deste capítulo são encontradas em (BONDY & MURTY, 1976) e (GOLDBARG & GOLDBARG, 2012).

II.1 Teoria dos Grafos

Um grafo é uma estrutura $G= (V,E)$, onde V é um conjunto discreto cujos elementos são denominados de vértices ou nós e, E é um conjunto de subconjuntos de elementos de V, cujos elementos são denominados de ligações, arcos ou arestas. O grafo G é de ordem n quando $|V|=n$ e tamanho m se $|E|=m$. Um grafo pode ser definido como direcionado (ou orientado) quando existe uma orientação nas ligações. Neste caso, as arestas são denominadas de arcos. Quando não existe orientação nas arestas o grafo é não orientado ou não direcionado e quando existe uma valoração das arestas, diz-se que o grafo é valorado ou ponderado. A Figura 1 exibe um grafo G orientado e um grafo H não orientado.

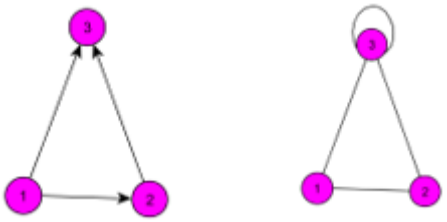


Figura 1: Grafos G e H

Um vértice v_i de G é denominado *adjacente* ou *vizinho* a outro vértice v_j quando existe uma ligação entre eles. Quando uma aresta liga um vértice a ele mesmo esta aresta é denominada de *laço* (conforme o vértice 3 no grafo H da Figura 1). Um *grafo* é denominado *vazio* quando $|E|=0$ e denominado de *grafo nulo* quando $|V|=0$.

Denomina-se de grafo *trivial* ou *singleton* o grafo que possui apenas um único vértice. Um *caminho* é uma sequência de vértices $v_1, \ldots, v_i, \ldots, v_s$ de modo que

$(v_i, v_{i+1}) \in E, \forall\ 1 \leq i \leq s-1$. O comprimento de um caminho entre dois vértices v_i e v_j é o número de arestas existentes nesse caminho. Denomina-se de distância *geodésica* o comprimento do menor caminho entre dois vértices v_i e v_j a qual denotamos por $d_{v_i v_j}$. Um ciclo é um caminho fechado, ou seja, é um caminho cujo vértice inicial coincide com o vértice final. Um grafo *G* é denominado conexo se para cada par de vértices existe um caminho ligando esses vértices. Caso contrário o grafo é denominado desconexo.

O grau de um vértice $v_i \in V$, denotado por $d(v_i)$, é o número de vértices adjacentes ao vértice v_i, ou ainda é o número de vizinhos de v_i. O grau mínimo de *G*, denotado por $\delta(G)$ é $\delta(G) = min\{d(v_i)/\ v_i \in V\}$ e o grau máximo de *G*, denotado por $\Delta(G)$ é $\Delta(G) = max\{d(v_i)/\ v_i \in V\}$.

Existem várias maneiras para representar um grafo. Uma delas é por matrizes, como, por exemplo: matriz de adjacência, Laplaciana, Laplaciana sem sinal, Laplaciana Normalizada e de Incidência.

A primeira alternativa de representar grafos através de matrizes é extremamente útil e, ao mesmo tempo, facilita o seu armazenamento. Além disso, essas representações podem ser facilmente manipuladas em computadores. A matriz de adjacência de um grafo não orientado *G*, denotada por $A(G) = [a_{ij}]$, é uma matriz de ordem *n*, sendo a entrada $a_{ij} = 1$ se $(v_i, v_j) \in E$, $a_{ij} = 0$, caso contrário. A Figura 2 exibe um grafo G_1 com 5 vértices e 5 arestas cuja matriz de adjacências é apresentada a seguir:

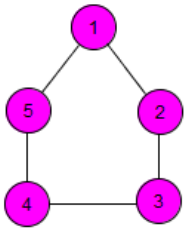


Figura 2: Grafo $\boldsymbol{G_1}$

$$A(G_1) = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Dada uma matriz quadrada *M* de ordem *n*, um escalar λ é denominado um autovalor de *M* se existe um vetor não-nulo *x*, denominado autovetor de *M*, tal que $Mx = \lambda x$. Para determinarmos os autovalores de *M* é necessário encontrar, inicialmente, o polinômio característico associado a *M*, $p(\lambda)$, definido através da seguinte equação: $p(\lambda) = det(\lambda I - M) = 0$, onde *I* é a matriz identidade de ordem *n*. As raízes dessa equação são denominadas de autovalores da matriz *M*.

Dado dois grafos $G = (V, E)$ e $H = (V', E')$, *H* é denominado *subgrafo* de *G* se $V' \subseteq V$ e $E' \subseteq E$. O subgrafo $H = (V', E')$ é considerado um *subgrafo* gerador de *G* quando $V' = V$ e $E' \subseteq E$.

Uma árvore *T* é um grafo conexo que não contém ciclos. Uma árvore geradora de um grafo *G* é um subgrafo gerador conexo e que não contém ciclos. Neste contexto, todo grafo conexo possui pelo menos uma árvore geradora. O custo de uma árvore geradora *T*, de um grafo ponderado *G*, é dado através da soma de todos os custos das arestas da árvore geradora *T*. Uma árvore geradora mínima (AGM), T_{Min}, é a árvore geradora de menor custo, dentre todas as possíveis árvores geradoras obtidas a partir de *G*. Mais especificamente, dado um grafo *G* com custos associados às arestas, a AGM obtida a partir de *G* corresponde ao subgrafo gerador cuja soma dos custos das arestas é mínima.

A Figura 3a exibe um grafo G_2 com 10 vértices e 15 arestas e uma árvore geradora mínima de G_2 é apresentada na Figura 3b. Pode-se observar que o custo de T_{Min} de G_2 é igual a 32.

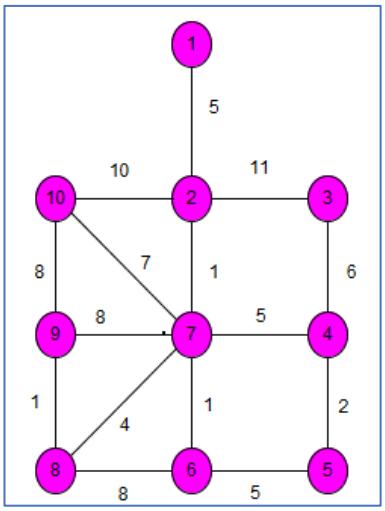


Figura 3a: Grafo G_2

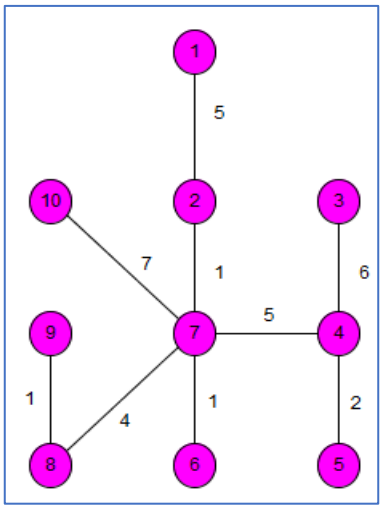


Figura 3b: T_{Min} de G_2

Figura 3: Grafo G_2
Fonte: Adaptado de (GOLDBARG & GOLDBARG, 2012)

O algoritmo de *PRIM*, desenvolvido em 1930, (GOLDBARG & GOLDBARG, 2012) tem como objetivo produzir árvores geradoras mínimas a partir de um grafo ponderado, conexo, não orientado e que não contenha laços.

O algoritmo é desenvolvido da seguinte maneira: inicialmente, deve-se escolher um vértice qualquer do grafo original. Após esta escolha, novas arestas devem ser adicionadas gradativamente à árvore em construção, observando o menor peso/custo de cada aresta nele incidente, até ser formada a AGM.

A Figura 4 ilustra a aplicação do algoritmo de *PRIM* em um grafo com 11 vértices e 18 arestas cujos passos são descritos a seguir: na iteração 1 escolhe-se um vértice inicial, por exemplo, é escolhido o vértice 1, S = {1}; posteriormente, na iteração 2, deve-se escolher uma aresta que tenha o menor peso/custo que incida no vértice 1, como pode ser visto na Figura 4, a aresta (1, 11). Assim, o conjunto *S* é atualizado, ou seja, S = {1; 11}, e deve-se realizar nova escolha de uma aresta que incida no vértice 1 ou 11, com menor custo e que alcance a um vértice que não esteja no conjunto S. Desta forma, na iteração 3, é incluída a aresta (1, 8), então S = { 1; 11 ; 8}. Esse procedimento é feito até que o conjunto *S* contenha todos os vértices do grafo e assim se obtém uma árvore geradora mínima, a qual é apresentada na Figura 5.

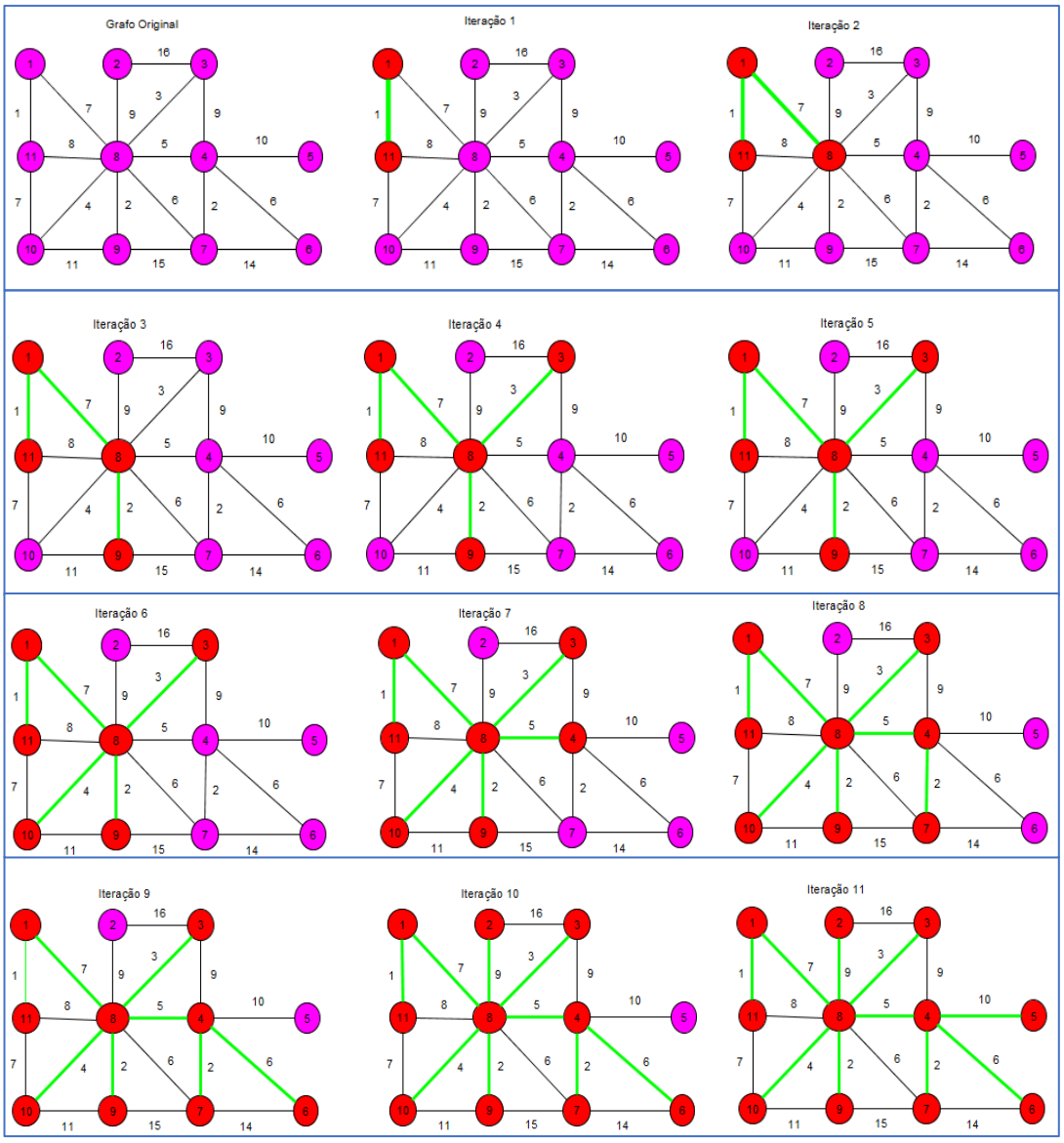


Figura 4: Aplicação do Algoritmo PRIM
Fonte: Os Autores.

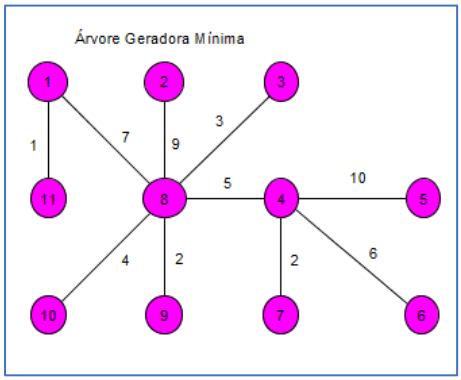


Figura 5: AGM
Fonte: Os Autores.

Capítulo III – Análise de Agrupamentos

Neste capítulo são apresentados os conceitos básicos de análise de agrupamentos assim como do problema de agrupamento. Além disto, é exposto o processo de descoberta de conhecimento em bases de dados, o procedimento para formação de agrupamentos, seleção e tratamento de dados. Por fim, são apresentados três algoritmos de análise de agrupamento utilizados nesta dissertação, quais sejam: *k-means*, *PAM* (*Partitioning Around Medoids*) e *CLARA* (*Clustering Large Applications*).

III.1 Análise de Agrupamentos

A prática de agrupar os objetos de acordo com as suas semelhanças é a base de diferentes áreas do conhecimento. De modo que, organizar um conjunto de dados em grupos é uma das maneiras mais fundamentais que configura a compreensão e o aprendizado (JAIN & DUBES, 1988).

Ademais, a prática de agrupar objetos (registros, observações, padrões) de acordo com as suas semelhanças, além de ser algo importante**,** está presente diariamente no nosso cotidiano (ALVES; OLIVEIRA; BRITO, 2014). Por exemplo, na escola agrupamos os alunos em classes, de acordo com a sua série ou faixa etária; na pesquisa, no levantamento de dados de uma população podemos agrupar as pessoas de acordo com o grau de escolaridade, sexo, renda, religião etc.

O agrupamento de dados pode ser utilizado, também, na pesquisa de mercado, a fim de reconhecer padrões sobre um determinado nicho. Já o profissional do marketing pode utilizar a Análise de Agrupamentos para descobrir grupos distintos entre a bases de dados que compõem os clientes, e assim alocar a um mesmo grupo clientes que tenham padrões de compra semelhantes (HAN; KAMBER; PEI, 2012).

A Análise de Agrupamentos (AAG) pode ser definida como uma técnica de Análise Multivariada que agrega um conjunto de algoritmos que são aplicados para agrupar objetos (JAIN & DUBES, 1988), de forma que os objetos semelhantes (de acordo com uma métrica calculada a partir dos atributos dos objetos) sejam alocados a um mesmo grupo e que objetos não semelhantes entre si sejam alocados a grupos distintos. Deste modo, os agrupamentos serão melhores à medida que, a semelhança (ou homogeneidade) aumente dentro de um grupo e a diferença (ou heterogeneidade) entre os grupos aumente

(KUMAR; STEINBACH; TAN, 2009), (KAUAI; PÉCORA; LIMA, 2020).

Atualmente, existem diversas aplicações reais associadas, por exemplo, às áreas psicologia e outras ciências sociais, biologia, medicina, marketing, estatística, mineração de dados (JAIN & DUBES, 1988), (KUMAR; STEINBACH; TAN, 2009), (HAN; KAMBER; PEI, 2012), sendo tais aplicações abordadas/resolvidas mediante aplicação da Análise de Agrupamentos. Ainda neste sentido, a resolução de tais aplicações, mediante a aplicação da AAG, remete à resolução de um problema de agrupamento.

III.2 Problema de Agrupamento

Dado um conjunto de dados *X* formado por *n* objetos e com *p* atributos (características), no problema de agrupamento (PA) o objetivo é particionar o conjunto *X* em *k* grupos, de forma que os objetos que pertencem a um mesmo grupo sejam homogêneos entre si, e os objetos pertencentes a grupos distintos sejam heterogêneos, ou seja, que a similaridade dos objetos dentro de cada cluster seja maximizada (variação intracluster) e a similaridade entre os objetos entre os cluster distintos deve ser minimizada (variação intercluster) (LAROSE, 2004). A Figura 6 ilustra esta variação intracluster e a variação intercluster.

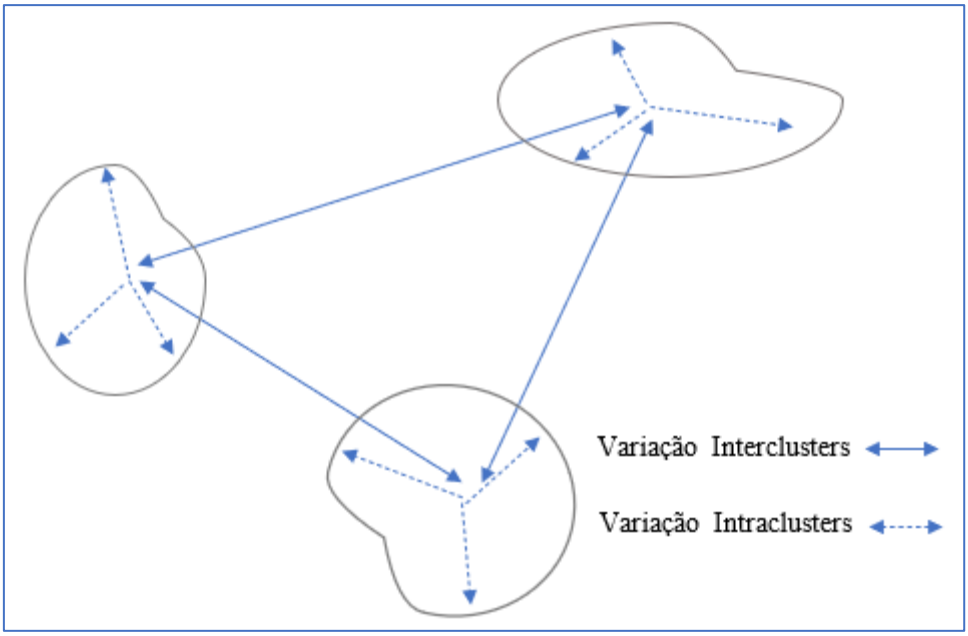


Figura 6:Similaridade entre os grupos e dentro dos grupos
Fonte: Larose (2004).

A fim de verificar a homogeneidade dentro de um grupo faz-se necessário utilizar alguma métrica, como, por exemplo, a distância Euclidiana ou distância de Manhattan. Essas métricas são descritas no subitem (III.3.1). É importante observar que**,** ao utilizar métricas diferentes na aplicação de determinado algoritmo para um mesmo conjunto de dados**,** podem ser produzidos agrupamentos bem distintos quanto aos objetos alocados aos grupos e ao grau de homogeneidade em cada grupo.

De acordo com (HANSEN; JAUMARD, 1997), o problema de agrupamento é definido, formalmente, da seguinte maneira: dado um conjunto X, formado por n objetos, onde $X = \{x_1, x_2, \cdots, x_n\}$ no qual, para cada objeto x_i existe p atributos, $x_i = \{x_i^1, x_i^2, \cdots, x_i^p\}$, deve-se construir, a partir X, k grupos/clusters (C_j), onde $C = \{C_1, C_2, \cdots, C_k\}$.

Na literatura, o problema de agrupamento pode ser definido ainda de duas formas: problema de agrupamento clássico (PAC) e problema de agrupamento automático (PAA) (ou ainda clusterização automática). Para o primeiro problema é necessário que o parâmetro *k* (número de clusters), seja definido a priori. Entretanto, quando este parâmetro não é definido a priori, e é pretendido determinar ou identificar o valor considerado ideal para *k*, o problema é dito de agrupamento automático (NALDI, 2011), (SEMAAN, 2013).

Para o PAC as restrições descritas pelas equações (III.1), (III.2) e (III.3), (HAN; KAMBER; PEI, 2012) devem ser satisfeitas.

$$\bigcup_{j=1}^{k} C_j = X \tag{III.1}$$

$$C_l \cap C_j = \emptyset \quad l, j = \{1,2,\cdots,k\}\ l \neq j \tag{III.2}$$

$$C_j \neq \emptyset\ j = \{1,2,\cdots,k\} \tag{III.3}$$

A equação (III.1) diz respeito à restrição de que a união de todos os clusters C_j gerados através do conjunto X deve ser igual ao próprio conjunto X; a equação (III.2) diz respeito à restrição de que nenhum elemento pode pertencer a mais de um cluster e a

equação (III.3) mostra que não deve existir nenhum cluster vazio.

No que diz respeito ao processo de formação dos grupos:

> "Os grupos são formados a partir de um princípio indutivo, também conhecido como função-objetivo ou critério de agrupamento. A função-objetivo é utilizada para selecionar um modelo ou estrutura capaz de identificar os grupos que melhor se ajustem a um determinado conjunto de dados. Uma vez determinada a função-objetivo, o problema de agrupamento consiste em otimizar esta função para o conjunto de dados em estudo. Na maioria das vezes, encontrar a solução ótima para este problema a partir da análise de possíveis soluções é intratável ou apresenta uma complexidade muito elevada para grandes conjuntos de dados" (NALDI, 2011, p. 8).

O conceito de solução ótima (ótimo global) apresentado acima pode ser entendido ainda como: a partir de um conjunto de dados, deve–se encontrar grupos/*clusters* que sejam mais homogêneos de acordo com alguma métrica. Porém, encontrar a solução (ou agrupamento) ótima global, ou seja, que produz o menor valor possível para função-objetivo definida a priori, não é uma tarefa fácil. Supondo, por exemplo, que fosse aplicado um método de enumeração exaustiva que permitisse gerar todas as possíveis alocações dos *n* objetos aos *k* grupos, o número de soluções possíveis (para o PAA), é dado através do número de *Stirling* de segundo tipo, definido pela equação (III.4) (LIU, 1968):

$$Ns(n,k) = \frac{1}{k!}\sum_{j=0}^{n}(-1)^{i}\binom{k}{j}(k-j)^{n} \qquad \text{(III.4)}$$

De modo a ilustrar a complexidade do problema, considere os valores de *Ns* na Tabela 1 para valores relativamente pequenos de *n*.

Tabela 1: Número de soluções para o problema de agrupamento usando o número de *Stirling*

n	k	$Ns(n,k)$
10	2	511
10	3	9.330
10	4	34.105
20	2	524.287
20	3	580.606.446
20	4	45.232.115.901
40	2	549.755.813.887
40	3	2.026.277.026.753.674.246
40	4	50.369.882.873.307.917.364.901

Fonte: Os Autores

Da Tabela 1, pode-se observar que, à medida que o número n (de objetos) aumenta o número de soluções possíveis cresce exponencialmente, ou seja, o número de maneiras em que os *n* objetos podem ser agrupados em *k* grupos cresce substancialmente. Tal fato torna proibitiva a aplicação de um método de enumeração exaustiva para encontrar a melhor solução associada ao *k* grupos.

Assim sendo, renuncia-se ao ótimo global e são considerados os algoritmos de agrupamento cujo objetivo é solucionar o problema de otimização. Tais algoritmos produzem soluções correspondentes a ótimos locais de boa qualidade demandando um baixo tempo computacional.

III.3 Processo de Descoberta de Conhecimento em Bases de Dados

Obter informações que estão armazenadas em grandes bases de dados não é uma tarefa simples de ser realizada. No entanto, existe na literatura um processo conhecido como KDD (do inglês *Knowledge Discovey in Databases*), cujo significado é a descoberta

de conhecimento em base de dados, o qual é “um processo não trivial, interativo e iterativo, que envolve várias etapas, com muitas decisões, tomadas pelo usuário, a fim de identificar padrões: válidos, novos, potencialmente úteis e compreensíveis” (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

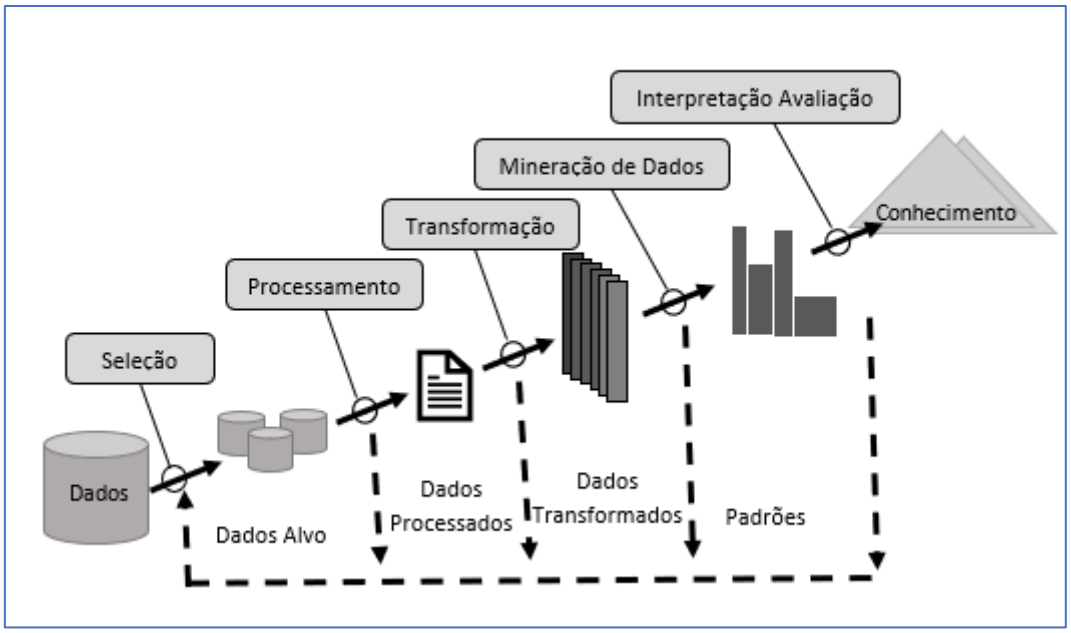


Figura 7: Etapas do KDD
Fonte: (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

A Figura 7, ilustra as cinco etapas do KDD, que consiste em: (i) seleção, (ii) processamento, (iii) transformação, (iv) mineração de dados e, por fim, a (v) interpretação/avaliação. A primeira etapa, seleção, consiste na seleção de dados o que permite que apenas atributos relevantes sejam considerados. Na segunda etapa, processamento, os dados são preparados de maneira a eliminar possíveis repetições, e/ou valores discrepantes. A terceira etapa consiste em realizar a transformação dos dados considerando, por exemplo, padronização de atributos quantitativos. A quarta etapa, mineração dos dados, tem por objetivo aplicar métodos tais como: modelagem, análise de agrupamento e regras de associação. Após esta etapa de mineração dos dados, os dados deverão ser analisados e validados. Vale ressaltar que este processo, além de ser iterativo, também é interativo (SILVA, Leandro, 2015).

III.3.1 Procedimento para formação de agrupamentos

Conforme descrito anteriormente, o processo de agrupamento tem como objetivo agrupar *n* objetos de uma base de dados, em *k* grupos, sendo tais objetos representados por pessoas, palavras, países, flores, entre outros. Cada um destes objetos possui *p* atributos, que podem ser: altura, peso, cor, temperatura e outros (KAUFMAN & ROUSSEEUW, 1990).

Para melhor compreensão, na Figura 8 são descritas as etapas do processo de agrupamento.

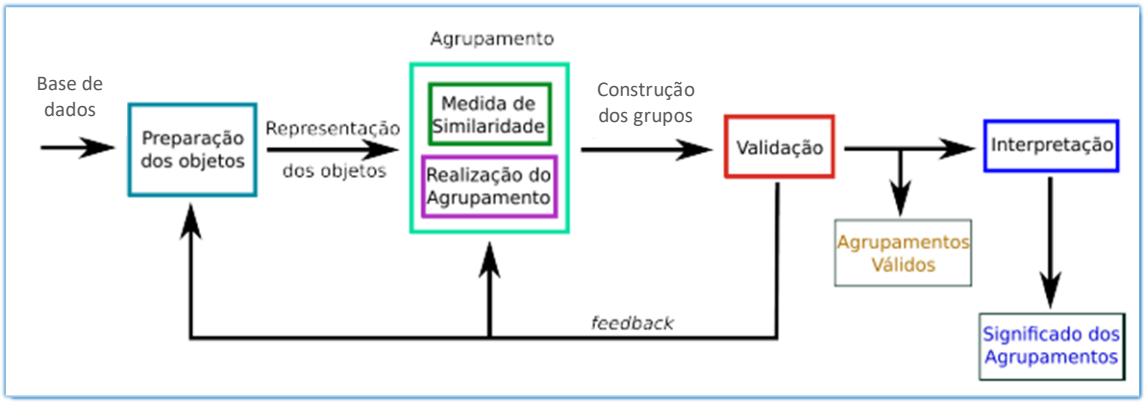


Figura 8: Etapas do processo de agrupamento
Fonte: NALDI (2011)

A Figura 8 corresponde a uma ilustração da visão geral do processo de construção de um agrupamento, desde a entrada da base de dados até a interpretação dos grupos gerados. De uma maneira geral, este processo se divide em: preparação dos dados, agrupamento, validação e, por fim, a interpretação dos agrupamentos. Estas etapas estão descritas a seguir:

- **Preparação de dados:** consiste em verificar a necessidade de aplicar algum tipo de transformação nos dados como a padronização (caso de variáveis quantitativas). Além disso, é nesta fase que é determinado a maneira como a similaridade entre os objetos é definida, o que pode ser através da matriz de distâncias (descrita na Seção III.3.1);

- **Medida de similaridade/dissimilaridade:** nesta etapa é realizado o cálculo da medida de similaridade/dissimilaridade entre todos os pares de objetos da base de dados;

- **Agrupamento:** neste momento deve ser escolhido, dentre os diversos algoritmos de agrupamento existentes, aqueles que serão utilizados, e por fim aplicá-los ao conjunto de dados;

- **Validação:** em síntese, esta etapa se concentra em verificar a qualidade dos agrupamentos, por meio do uso de medidas estatísticas que auxiliam a verificação, através de um ou alguns índices associados, por exemplo, os critérios de validação relativo ou externo.

- **Interpretação:** constitui-se na compreensão dos agrupamentos formados, de modo a verificar a existência de padrões ou características que representem cada grupo.

Na próxima subseção é apresentado o procedimento para realizar a seleção e tratamentos de dados e a subseção III.4 diz respeito aos algoritmos de agrupamento.

III.3.2 Seleção e Tratamento de dados

Os objetos da base de dados podem, geralmente, ser representados por vetores de atributos (características). Os atributos, por sua vez, podem ser divididos em dois tipos, a saber: atributos quantitativos (ou numéricos) e atributos qualitativos (ou categóricos). Os qualitativos assumem valores finitos, e se dividem em: nominais e ordinais. Os atributos qualitativos nominais representam características que não apresentam ordem, e os atributos qualitativos ordinais representam características que refletem uma sequência ou uma ordem. Já os quantitativos assumem valores em conjuntos finitos ou infinitos e se dividem em: contínuos e discretos. O Quadro 1 ilustra, em síntese, o descrito acima, onde os atributos são características de um objeto.

Quadro 1: Tipos de atributos

Tipos de Atributos		
Qualitativos ou Categóricos	Nominais	Profissão; Cor de cabelo; Cor de olhos; Religião, Sexo; Estado civil; fumante/não fumante; Doente/sadio.
	Ordinais	Grau de instrução (1º grau, 2º grau e 3º grau)
Quantitativos ou Numéricos	Contínuos	Altura; Tempo; Peso
	Discretos	Número de filhos; Número de clientes, Número de peças, Número de máquinas.

A preparação dos dados, o qual representa a primeira etapa do processo de agrupamento, tem como principal objetivo propiciar a análise dos tipos de objetos pertencentes à base de dados, a fim de selecionar apenas os objetos ou atributos que possuam maior relevância para realização do agrupamento e, desta forma, pode-se eliminar/excluir registros (objetos) irrelevantes a análise (VALE, 2005).

Assim sendo, alguns dos registros (objetos) presentes na base de dados não serão utilizados para realização do agrupamento, posto que estes podem conter (LAROSE,2004; DONI, 2004):

- Registros com falta de informação (*missing values*) em relação a algum atributo;
- Outliers;
- Dados duplicados;
- Erros de digitação.

A fim de selecionar apenas registros (objetos) que sejam relevantes, o conjunto de dados deve ser submetido a um pré-processamento, de maneira a remover da base de dados objetos que sejam irrelevantes como os que foram citados acima, (LAROSE,2004).

Após os objetos da base serem submetidos a este pré-processamento, é necessário verificar a base de dados e observar se as variáveis quantitativas associadas aos objetos estão na mesma unidade, ou na mesma escala. Caso estejam, pode-se seguir para o passo seguinte da análise de agrupamentos, caso contrário, é necessário realizar uma transformação das variáveis, isto é, uma padronização das variáveis. Para padronizar as *n*

observações da *k*-esima variável x_{ik} em relação ao *i*-ésimo registro, deve-se subtrair a média e dividir pelo desvio padrão. Dado um conjunto com n objetos e p variáveis/atributos tem-se que a equação (III.5), define z_{ik} – correspondente ao valor padronizado da i-ésima observação da j-ésima variável :

$$z_{ik} = \frac{x_{ik} - \mu_k}{\sigma_k} \qquad \text{(III.5)}$$

Na equação (III.5), $\mu_k e\ \sigma_k$ representam, respectivamente, a média e o desvio-padrão da *k*-ésima variável.

Para exemplificar como é realizada a padronização segue, o exemplo apresentado na Tabela 2.

Tabela 2:Exemplo de variáveis quantitativas com ordens de grandeza distintas

Cidadão	Renda (R$)	Idade (anos)
A	6.000	30
B	7.000	28
C	8.000	50
D	9.000	55

A Tabela 3 traz a média e o desvio-padrão das variáveis renda e idade em anos e Tabela 4 ilustra as variáveis já padronizadas.

Tabela 3: Média e desvio-padrão das variáveis Renda e Idade

Variável	Média	Desvio-padrão
Renda (R$)	7,50	1118,03
Idade (anos)	40,75	11,90

Tabela 4: Variáveis padronizadas

Variáveis padronizadas	
Renda (R$)	Idade (anos)
-1,34	-0,90
-0,45	-1,07
0,45	0,78
1,30	1,19

O passo seguinte em relação à etapa de padronização consiste em calcular a distância entre os objetos (função dos atributos definidos), de modo a quantificar a similaridade, através de medidas de similaridade, $s(x_i, x_j)$ (ou de dissimilaridade, $d(x_i, x_j)$).

Quanto mais próximo de zero for $s(x_i, x_j)$, mais diferentes são os objetos em função de seus *p* atributos. Já para a dissimilaridade vale o caso oposto, quanto mais próximo $d(x_i, x_j)$ for de zero, mais parecidos em relação aos seus atributos são os objetos x_i e x_j (KAUFMAN & ROUSSEUW, 1990).

Desta forma, definimos a matriz de dissimilaridades como sendo uma matriz quadrada, denotada por *D*, sendo cada entrada de D denotada por d_{ij}, tal que $d_{ij} = d(x_i, x_j)$, e $d(x_i, x_j)$ indica a dissimilaridade entre os objetos x_i e x_j. Para atributos do tipo quantitativo, as medidas de dissimilaridade mais utilizadas são as distâncias: Euclidiana, de Manhattan e de Mahalanobis.

A distância Euclidiana entre dois objetos, calculada a partir de seus *p* atributos, é representada pela equação (III.6), (HAN; KAMBER; PEI, 2012):

$$d(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2} \qquad \text{(III.6)}$$

A distância de Manhattan entre os objetos, calculada a partir de *p* atributos, é representada pela equação (III.7), (HAN; KAMBER; PEI, 2012):

$$d(x_i, x_j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}| \qquad \text{(III.7)}$$

III.4 Algoritmos

Os algoritmos de agrupamento podem ser classificados, basicamente, em hierárquicos e não hierárquicos (ou particionais). Os algoritmos do tipo hierárquico caracterizam-se pelo número de grupos (k) não ser fixado a priori e ainda são divididos em: aglomerativos e divisivos.

No caso dos algoritmos não hierárquicos, o número de grupos é definido a priori. Como exemplos de algoritmos não-hierárquicos temos o *k-means*, *PAM* e *CLARA*, que

serão descritos nas próximas subseções.

Estes algoritmos foram escolhidos em virtude da boa aceitação e grande utilização pela literatura. Além disso, tais algoritmos estão implementados nos pacotes *stats* e *cluster* do *software R*, o qual foi utilizado para realização de todos os experimentos computacionais realizados nesta dissertação.

III.4 .1 Algoritmo *k-means*

O algoritmo *k-means* é um dos mais conhecidos da literatura e mais utilizado em problemas práticos (ALVES; OLIVEIRA; BRITO, 2014), (JAIN, 2010), (FRÄNTI; SIERANOJA, 2018), (KAUAI; PÉCORA; LIMA, 2020).

Uma vez escolhido o valor de *k*, posteriormente, deve-se calcular as distâncias de cada objeto a cada um dos centroides (médias), dos objetos de cada um dos grupos. Os objetos são alocados ao grupo associado ao centroide que estiver a menor distância calculada. Em seguida, são identificados os *k* grupos formados e então, são recalculados os novos centroides de cada novo grupo (JAIN & DUBES, 1988).

Após esta etapa, são calculadas novamente as distâncias de cada objeto aos centroides a fim de, encontrar a menor distância entre eles, e, o próximo passo consiste em realocar esses objetos nos novos grupos. Este processo continua até que não haja mais mudanças significativas em relação aos valores dos centroides.

A seguir são apresentados os passos básicos do algoritmo *k-means*, considerando que: *k* representa o número de grupos; x_{i} um objeto; g_{j} um grupo de índice j; c_j o centroide do grupo g_{j} ; $d(g_{\mathrm{j}}, x_{\mathrm{i}})$ a distância Euclidiana entre o grupo de índice *j* e o objeto x_{i}, (FREI, 2006), (JAIN & DUBES, 1988).

1- Separar os *n* objetos em *k* grupos, de maneira aleatória;

2- Calcular os centroides (médias) dos objetos de cada grupos em relação aos atributos;

$$c_j = \frac{1}{nj} \sum_{x_{\mathrm{i}} \in g_{\mathrm{j}}} x_{\mathrm{i}}, \quad j = ,1\ldots, k \qquad (III.9)$$

3- Associar cada um dos *n* objetos ao grupo cujo centroide está mais próximo, conforme a fórmula a seguir;

$$\sum_{j=1}^{k}\sum_{x\in c_j} d\left(g_{\mathrm{j}}, x_{\mathrm{i}}\right)^2 \qquad (III.10)$$

4- Atualizar o centroide de cada grupo que recebeu o novo objeto e do grupo que perdeu o objeto;

5- Repetir os passos 3 e 4 até que não haja nenhuma mudança significativa nas posições dos centroides, considerando duas iterações seguidas.

Quanto às principais características deste algoritmo, pode-se destacar que é suscetível a valores atípicos e trabalha com variáveis quantitativas, sendo de fácil implementação, (ALVES; OLIVEIRA; BRITO, 2014), (JAIN & DUBES, 1988).

III.4 .2 Algoritmo *PAM* (*Partitioning Around Medoids*)

O Algoritmo *PAM* é formalmente definido da seguinte maneira: seja o conjunto *X* com *n* objetos, $X = \{x_1, x_2, \dots, x_n\}$. A partir deste grupo, inicialmente, deve se escolher *k* objetos aleatoriamente, a fim de formar o conjunto de *medoid*s, $M = \{\mathrm{medoid}_1, \mathrm{medoid}_2, \dots, \mathrm{medoid}_\mathrm{k}\}$. Posteriormente**,** deve-se alocar cada um dos $(n-k)$ objetos restantes ao grupo cujo *medoid* esteja mais próximo (função da medida de distância escolhida). Este processo deve ser repetido até que não haja mudanças significativas na função objetivo em síntese, o algoritmo tem os seguintes passos:

1- Selecionar, dentre os *n* objetos disponíveis de *X*, os *k* objetos que definem um conjunto *M* de *medoid*s;
2- Associar os $(n-k)$ objetos restantes ao seu *medoid* mais próximo;
3- Substituir os *k medoid*s de forma a minimizar a soma das distâncias dos $(n-k)$ objetos ao seu *medoid* mais próximo;
4- Repetir os passos 2 e 3 até que não haja nenhuma mudança significativa nas posições dos *medoids*.

O algoritmo de agrupamento *PAM*, utilizado para formar os *k* grupos tendo por base os *k-medoids*, pode ser considerado como uma variação do algoritmo *k-means*.

No algoritmo *k-means*, a priori, *k* centroides devem ser escolhidos, entretanto, no

método *k-medoids* essa referência ao centro de gravidade do grupo não é levada em consideração, mas, sim, a utilização de *k* objetos representativos denominados de *medoids*.

Já em relação ao algoritmo *k-means*, o centroide não é um dos objetos do grupo, mas, sim, a média em relação aos atributos dos objetos alocados ao grupo (CAMILO & SILVA, 2009).

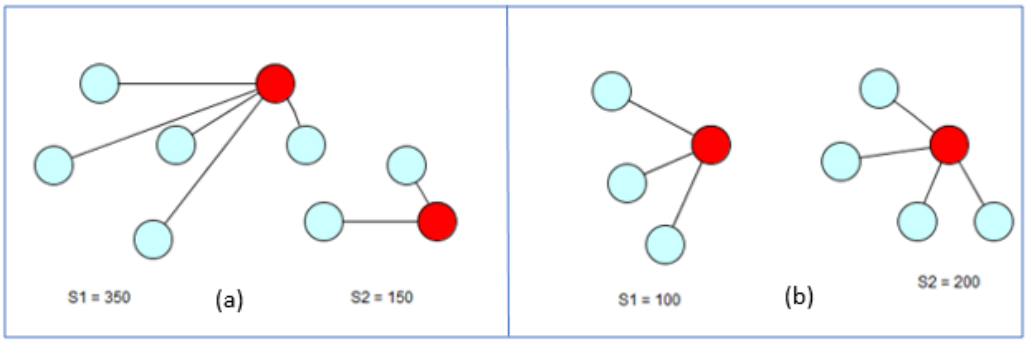


Figura 9: Duas possíveis soluções de agrupamento com 2 *medoids*
Fonte: (BRITO; SEMAAN; BRITO, 2015)

A Figura 9 ilustra duas possíveis soluções (a) e (b) a partir da aplicação do algoritmo, o qual deve alocar nove objetos em 2 grupos, para tal deve-se obter 2 *medoids*. Nesta ilustração os objetos na cor vermelha representam os *medoids*, as retas que ligam os objetos correspondem à distância entre eles; já S1 e S2 representam a soma das distâncias dos objetos a seus respectivos *medoids* (BRITO; SEMAAN; BRITO, 2015).

Como pode ser observado no passo 2 do algoritmo, deve-se associar o objeto ao *medoid* que esteja mais próximo, considerando alguma métrica de distância. Desta maneira, as soluções representadas na Figura 9 seriam: Figura 9.a soma total igual a 500 e Figura 9.b soma total igual a 300. Para este exemplo, a solução a ser escolhida seria a apresentada pela Figura 9.b, a qual representa o menor valor das distâncias de todos os objetos aos seus respectivos *medoids*.

III.4 .3 Algoritmo *CLARA* (*Clustering Large Applications*)

O algoritmo *CLARA* criado e desenvolvido por KAUFMAN & ROUSSEEUW em 1989 é uma variação do algoritmo *PAM*, e é utilizado para conjunto de dados maiores (muitos objetos), visto que, a aplicação do algoritmo *PAM* é mais viável para base de

dados de pequeno a médio porte.

O algoritmo *CLARA* tem por objetivo lidar com amostras do conjunto original de dados *X*, de maneira alternativa como é executado o algoritmo *PAM* o qual trabalha com todo o conjunto de dados. Desta maneira, o algoritmo *CLARA* seleciona, d do conjunto *X* , *q* amostras aleatórias de tamanho *n´* (*n´< n*).

Após ter selecionado as *q* (mínimo indicado pelos autores é 5) amostras aleatórias da base de dados, deve-se aplicar o algoritmo *PAM* em cada uma das amostras. Posteriormente, em cada amostra, considerando os *k-medoids* obtidos em função dos *n´* objetos, realiza-se a alocação dos (*n - n´)* objetos restantes ao *medoid* mais próximo. A solução escolhida corresponde àquela com menor valor de função objetivo.

Capítulo IV – Detecção de Comunidades em Redes

Este capítulo tem por objetivo apresentar os conceitos de detecção de comunidades, modularidade e, por fim, apresentar dois algoritmos de detecção de comunidades que são utilizados nesta dissertação, quais sejam: *Fast Greedy* e *Walktrap*.

IV.1 Detecção de Comunidades

Redes são definidas como "sistemas físicos, biológicos ou sociais caracterizados por um conjunto de entidades bem definidas que se integram e interagem dinamicamente entre si" (TEIXEIRA; LIMA; ABREU, 2008). Considerando essa definição, podemos representar, através de redes, diversas estruturas: redes de coautoria, redes de esgoto, redes de telefonia, redes de computadores, entre outras. As redes podem ser modeladas a partir da utilização de uma estrutura matemática denominada grafo $G = (V, E)$. Mediante tal representação, pode-se obter variadas informações no que concerne à topologia dessas redes como, por exemplo, o fluxo de informação que passa pelos nós e clusters de nós (COSTA et al., 2013).

Em muitas redes uma propriedade comum é a existência de comunidades (*clusters* ou grupos) com alta interação de suas arestas com seus respectivos nós, havendo, em contrapartida, uma baixa densidade nas relações entre os grupos. Esta propriedade é conhecida na literatura como "estrutura de comunidade". A Figura 10 ilustra uma rede na qual há uma alta densidade de arestas, dentro de cada comunidade, e uma baixa densidade de arestas entre as comunidades, (NEWMAN & GIRVAN, 2004).

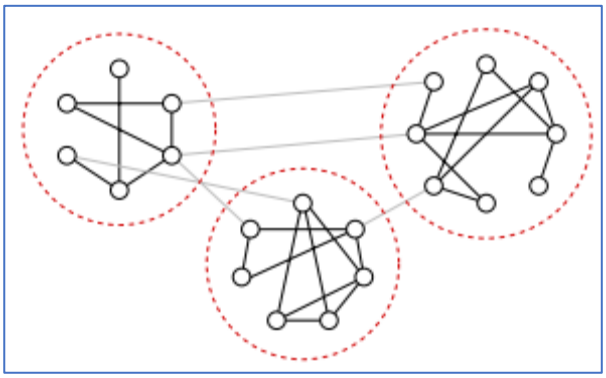


Figura 10: Exemplo de três comunidades de uma rede
Fonte: Newman & Girvan (2004)

De acordo com ((GOLDSCHMIDT *et al*., 2015) apud (AMARAL, 2020)) o principal objetivo da detecção de comunidades é encontrar comunidades, isto é, subconjuntos de vértices de uma rede/grafo, dos quais estejam fortemente conectadas entre si.

Já no que se refere ao problema de detecção de comunidades é um problema parecido com problema de corte ou de particionamento em grafos ((KARGER (2000), KERNIGHAN & LIN (1970), FIDUCCIA & MATTHEYSES (1982) apud LINARES (2013)**)**. O problema de detecção de comunidades tem como objetivo procurar comunidades que possuam alguma similaridade entre seus nós, ou seja, que desempenhem funções similares entre seus nós.

Vale ressaltar que o tamanho e o número de comunidades de uma rede são desconhecidos a priori. Assim sendo, de forma a determinar esses valores, faz-se necessário aplicar algum algoritmo de detecção de comunidades (ALVES; OLIVEIRA; BRITO, 2014).

Admitindo que C_1, C_2,$\cdots$, C_k sejam comunidades obtidas a partir da aplicação de um algoritmo na rede, essas comunidades devem satisfazer as seguintes propriedades (FIDUCCIA & MATTHEYSES, 1982):

$$\text{I. } C_i \cap C_j = \emptyset \quad i,j \ = \{1,2,\cdots,k\}\ i \neq j \tag{IV.1}$$

$$\text{II.} \cup C_i = V \tag{IV.2}$$

onde a Equação (IV.1) diz respeito à propriedade, que nenhum elemento pode pertencer a mais de uma comunidade e a Equação (IV.2) mostra que a união das comunidades é igual ao conjunto de vértices da rede estudada e representada pelo grafo *G=(V,E)*.

De acordo com LINARES (2013), um dos principais problemas relacionados à detecção de comunidades é: como pode ser definido a melhor divisão da rede em comunidades, visto que, em redes reais, geralmente não há um conhecimento prévio de informações importantes tais como, o número e o tamanho das comunidades preexistentes. Desta maneira, não existe nenhuma regra geral que possa ser aplicada para definir a melhor divisão de uma rede real em comunidades. A fim de resolver esse problema, Newman & Girven (2004) propuseram uma medida, denominada

Modularidade a qual é apresentada na próxima subseção.

IV.2 Modularidade

De acordo com (CLAUSET *et al.*, 2004), o conceito de modularidade foi um dos primeiros a aparecer, quando os pesquisadores começaram a avaliar formas de definir o problema de agrupamento em grafos, no que tange ao conceito de comunidades.

Por sua vez, nos últimos anos, as inúmeras aplicações reais que remetem ao problema de detecção de comunidades têm motivado o estudo e a proposta de vários algoritmos que resolvem o problema de determinar a modularidade máxima em uma rede. Outra motivação para o estudo desses métodos diz respeito à complexidade intrínseca a esse problema, que pertence à classe de problemas NP-difíceis.

Dentre os vários algoritmos propostos na literatura, destacamos, neste trabalho, o algoritmo *Fast Greedy* o qual é descrito na próxima subseção.

A modularidade permite mensurar a qualidade de uma possível divisão da rede em comunidades. Em outras palavras, se é significativa ou não essa divisão. Essa medida, de acordo com (CLAUSET; NEWMAN; MOORE, 2004), é definida da seguinte maneira:

$$Q = \frac{1}{2m} \sum_{v_i v_j} \left[a_{ij} - \frac{d(v_i)d(v_j)}{2m} \right] \delta\left(c_{v_i}, c_{v_j}\right) \qquad \text{(IV.3)}$$

onde $a_{v_i v_j}$ representa cada entrada da matriz de adjacência e v_i e v_j correspondem aos vértices associados ao grafo que modela a rede, m o número de arestas do grafo, $d(v_i)\ e\ d(v_j)$ representam o grau do vértice v_i e do vértice v_j e, $\delta\left(C_{v_i}, C_{v_j}\right) = 1$ quando $v_i\ e\ v_j$ pertencem à mesma comunidade, caso contrário $\delta\left(C_{v_i}, C_{v_j}\right) = 0$.

Assim, podemos observar que, quando o valor da modularidade (Q) encontra-se muito próximo de zero, isto é um indicativo de que não existe uma estrutura de comunidade forte. Em contrapartida, quando o valor de Q é próximo de 1 é um indicativo da existência de uma estrutura de comunidade forte. Porém, em muitas aplicações reais os valores de Q, obtidos a partir da aplicação de um algoritmo de detecção, não excedem a 0.7 e valores mais altos não ocorrem de maneira frequente, (ALVES; OLIVEIRA; BRITO, 2014).

Com o objetivo de exemplificar o cálculo da modularidade, considere o grafo *G4* da Figura 11, dividido em duas comunidades C_1 e C_2 cujos vértices são $\{1; 5\}$ e $\{2; 3; 4\}$, respectivamente conforme Figura 12.

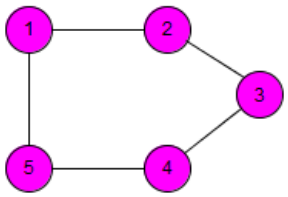


Figura 11: Grafo $\boldsymbol{G_3}$

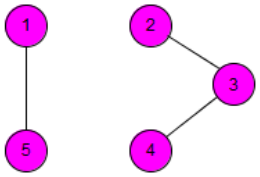


Figura 12: Comunidades $\boldsymbol{C_1}$ e $\boldsymbol{C_2}$

Para determinarmos o valor da *modularidade*, *Q*, considerando essa divisão, apresentamos, a seguir, a matriz de adjacência associada ao grafo, G_3.

$$A(G_3) = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

O número de arestas do grafo é, $m = 5$ e $d(v_1) = 2$; $d(v_2) = 2$; $d(v_3) = 2$; $d(v_4) = 2$ e $d(v_5) = 2$. A Tabela 5, traz os valores de $\delta(v_i, v_j)$, indicando se v_i e v_j estão na mesma comunidade:

Tabela 5: Verificação se os vértices $\boldsymbol{v_i}$ e $\boldsymbol{v_j}$ estão na mesma comunidade

$\boldsymbol{v_i}$	$\boldsymbol{v_j}$	$\boldsymbol{\delta(v_i, v_j)}$
$\boldsymbol{v_1}$	v_2	0
$\boldsymbol{v_1}$	v_3	0
$\boldsymbol{v_1}$	v_4	0
$\boldsymbol{v_1}$	v_5	1
$\boldsymbol{v_2}$	v_1	0
$\boldsymbol{v_2}$	v_3	1
$\boldsymbol{v_2}$	v_4	1
$\boldsymbol{v_2}$	v_5	0
$\boldsymbol{v_3}$	v_1	0
$\boldsymbol{v_3}$	v_2	1
$\boldsymbol{v_3}$	v_4	1
$\boldsymbol{v_3}$	v_5	0
$\boldsymbol{v_4}$	v_1	0
$\boldsymbol{v_4}$	v_2	1
$\boldsymbol{v_4}$	v_3	1
$\boldsymbol{v_4}$	v_5	0
$\boldsymbol{v_5}$	v_1	1
$\boldsymbol{v_5}$	v_2	0
$\boldsymbol{v_5}$	v_3	0
$\boldsymbol{v_5}$	v_4	0

Substituindo os valores na Equação (IV.3), temos:

$$Q = \frac{1}{2*5}\left(\left(1 - \frac{2*2}{2*5}\right) * 0 + \left(0 - \frac{2*2}{2*5}\right) * 0 + \cdots + \left(1 - \frac{2*2}{2*5}\right) * 0 + \left(0 - \frac{2*2}{2*5}\right) * 1\right) = 0{,}08.$$

Para as comunidades $\boldsymbol{C_1}$ e $\boldsymbol{C_2}$ o valor da modularidade desta partição é *Q= 0,08*. Este valor encontra-se muito próximo de zero; o que é um indicativo que há uma baixa probabilidade de a rede estar dividida em uma estrutura de comunidade forte.

IV.3 Algoritmos de Detecção de Comunidades

Nesta subsecção são apresentados os dois algoritmos de detecção de comunidades em redes utilizados nesta dissertação, o *Fast Greedy* e o *Walktrap*.

IV. 3 .1 Algoritmo *Fast Greedy*

O *Fast Greedy* foi proposto por (CLAUSET; NEWMAN; MOORE, 2004) como sendo um algoritmo utilizado para a detecção de comunidades, o qual maximiza a medida de modularidade definida na Equação IV.3, fazendo uso de um algoritmo guloso, como pode ser visto em (Botelho & Sousa, 2015). Este algoritmo foi baseado no algoritmo já proposto por Newman em 2004.

De modo a entender melhor o funcionamento do algoritmo**,** algumas explicações são expostas, a seguir**:**

- Inicialmente o *Fast Greedy* considera cada vértice do grafo que modela a rede como uma comunidade;
- O algoritmo encontra as variações na modularidade (ΔQ), a qual é definida posteriormente, e ao encontrar os maiores valores de ΔQ, realiza então a junção das comunidades que possuem os maiores valores em relação a *Q*. É importante observar que este procedimento só é realizado se houver uma ligação entre essas comunidades, ou seja, se existirem uma ou mais arestas que as unam. De acordo com (CLAUSET; NEWMAN; MOORE, 2004), a união de duas comunidades sem que haja uma aresta em comum, não irá resultar em incremento no valor da modularidade;
- O *Fast Greedy* armazena e atualiza os valores de ΔQ_{ij} em uma matriz, ΔQ quecontém as variações de modularidade e ΔQ_{ij} representa cada elemento dessa matriz;
- O algoritmo utiliza a estrutura *max-heap H* para armazenar o maior elemento de cada linha da matriz ΔQ.

Conforme comentando acima, o *Fast Greedy* inicialmente considera cada vértice do grafo que modela a rede como o único membro de uma comunidade e, iterativamente, novos membros são adicionados à comunidade. A associação de mais de um vértice a uma certa comunidade produz um maior incremento no valor da modularidade – de forma que para fazer a junção de duas comunidades é necessário encontrar o maior valor da variação de modularidade (ΔQ_{ij}), a qual é definida da seguinte maneira:

$$\Delta Q_{ij} = \begin{cases} \frac{1}{2m} - \frac{d(v_i)d(v_j)}{(2m)^2} & se\ v_i, v_j \text{ estiverem conectados} \\ 0 \quad \text{caso contrário} \end{cases} \qquad \text{(IV.4)}$$

Sendo m correspondente ao número de arestas do grafo e $d(v_i)$ é o grau do vértice v_i e analogamente $d(v_j)$ representa o grau do vértice v_j. De acordo com (CLAUSET; NEWMAN; MOORE, 2004), as etapas do *Fast Greedy* são definidas da seguinte maneira:

1 – Inicialmente são calculados os valores iniciais para a variação de modularidade (ΔQ_{ij}) para o par de vértices v_i, v_j, e são calculados os valores de $\frac{d(v_i)}{2m}$. Feito isso, será armazenado o maior elemento de cada linha da matriz ΔQ , no *max-heap*;

2 – Posteriormente, é identificado o maior valor de ΔQ_{ij} do *max-heap H*, e então deve se juntar as comunidades correspondentes e, concomitantemente, deve-se atualizar a matriz ΔQ, o *max-heap H* e o vetor contendo os valores de $\frac{d(v_i)}{2m}$, e então incrementar Q através de ΔQ_{ij};

3 – Por fim, o passo 2 é repetido até que reste apenas uma única comunidade.

No *Fast Greedy* a entrada do algoritmo é a rede a ser estudada, e como saída o algoritmo retorna o valor da modularidade Q e o número de comunidades. Outra saída possível do algoritmo é o dendograma apresentado na Figura 13, que corresponde a uma representação hierárquica da rede em comunidades.

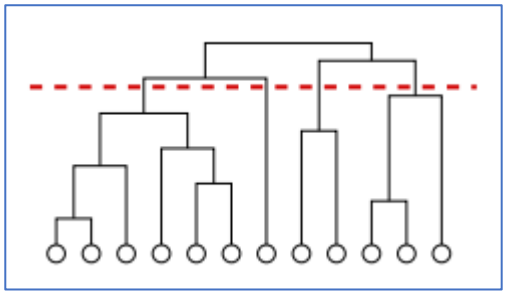


Figura 13: Exemplo de dendograma ou árvore hierárquica
Fonte: NEWMAN & GIRVAN, 2004

Nesta figura os círculos representam os vértices individuais da rede. Estes vértices se unem a fim de formar comunidades cada vez maiores, como é indicado pelas linhas, até por fim, chegar ao todo, no qual todos os vértices estão unidos em uma única comunidade. Ainda na Figura 13, o corte indicado pela linha tracejada em vermelho indica a formação de quatro comunidades.

A fim de exemplificar a execução do algoritmo *Fast Greedy* considere a matriz

de adjacência *A*, e a matriz ΔQ, construida a apartir da Equação (IV.4), apresentadas a seguir:

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix};$$

$$\Delta \mathrm{Q} = \begin{pmatrix} 0 & 0{,}051 & 0{,}020 & 0 & 0 & 0 \\ 0{,}051 & 0 & 0{,}020 & 0 & 0 & 0 \\ 0{,}020 & 0{,}020 & 0 & -0{,}025 & 0 & 0 \\ 0 & 0 & -0{,}025 & 0 & 0{,}020 & 0{,}020 \\ 0 & 0 & 0{,}020 & 0{,}020 & 0 & 0{,}051 \\ 0 & 0 & 0 & 0{,}020 & 0{,}051 & 0 \end{pmatrix}$$

A partir da matriz ΔQ é selecionado o maior elemento em cada uma de suas linhas e, posteriormente, verifica-se o maior valor para ΔQ_{ij}. Para este exemplo, os valores máximos são $\Delta Q_{12} = 0{,}051$ e $\Delta Q_{65} = 0{,}051$, e como ΔQ_{12} e ΔQ_{65} possuem valores iguais, independe a escolha do vértice para unir a comunidade inicial, C_0. Na iteração 1, escolhe-se então, ΔQ_{12} e os vértices 1 e 2 para formarem a comunidade inicial, C_0.

Posteriormente, é atualizada a matriz ΔQ da seguinte maneira: é removida a *i*-ésima linha e a *i*-ésima coluna *i*. No exemplo acima são excluídas a linha e a coluna correspondentes ao vértice 1, e são atualizados os valores $\Delta Q'_{jC_0}$ para nova comunidade utilizando a *j*-ésima linha e *j*-ésima coluna. Neste exemplo são atualizados os valores para linha e coluna correspondente ao vértice 2. É importante ressaltar que os outros valores de ΔQ permanecem inalterados. A seguir é apresentada a matriz $\Delta Q'$ associada ao exemplo apresentando acima. A linha em vermelho, representa os valores que devem ser excluídos e os valores em negrito representam a atualização para $\Delta Q'_{2C_0}$, através da Equação (IV.5):

$$\Delta Q' = \begin{pmatrix} 0 & 0{,}051 & 0{,}020 & 0 & 0 & 0 \\ 0{,}051 & \mathbf{0} & \mathbf{-0{,}11} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0{,}020 & \mathbf{-0{,}11} & 0 & -0{,}025 & 0 & 0 \\ 0 & \mathbf{0} & -0{,}025 & 0 & 0{,}020 & 0{,}020 \\ 0 & \mathbf{0} & 0{,}020 & 0{,}020 & 0 & 0{,}051 \\ 0 & \mathbf{0} & 0 & 0{,}020 & 0{,}051 & 0 \end{pmatrix}$$

É importante destacar que a cada etapa do algoritmo *Fast Greedy* consiste na seguinte observação: juntar duas comunidades sem que haja uma aresta em comum não produz incremento em ΔQ_{ij}. Desta maneira, é necessário apenas armazenar os valores de ΔQ_{ij} para as comunidades *i* e *j* que estão ligadas por uma ou mais arestas.

Posteriormente, para a atualização de $\Delta Q'$ é necessário verificar se deverá juntar a comunidade C_0 a um novo vértice, o vértice 3, ou se este vértice deve fazer parte da formação de uma outra comunidade, para tal são comparados os valores $\Delta Q'$ para o vértice 3 : $\Delta Q'_{3C_0} = -0{,}11$ e $\Delta Q'_{34} = -0{,}025$, observasse que o maior incremento em ΔQ_{ij} é $\Delta Q'_{3C_0}$. Sendo assim, na iteração 2, é feita a junção da comunidade C_0 ao vértice 3, e é formada uma nova comunidade C_1. Este procedimento é repetido até todos os vértices terem sido explorados e haver uma única comunidade.

IV. 3 .2 Algoritmo *Walktrap*

O *Walktrap* é um algoritmo baseado no conceito de caminhada aleatória desenvolvido por (PONS & LAPATY, 2006). A ideia principal desse algoritmo é realizar caminhos aleatórios na rede, com o intuito de que esses caminhos permaneçam dentro da mesma estrutura de comunidade. O algoritmo funciona da seguinte maneira: é atribuída uma medida de similaridade entre as comunidades, denotada por uma distância *r*. Esta medida possibilita o cálculo da distância entre os vértices e/ ou comunidades baseadas no caminho percorrido pelo *random walk* entre dois vértices e/ou comunidades (PORTO, 2014).

A Equação (IV. 5) apresenta como é definida a distância entre duas comunidades, (PONS & LAPATY, 2006):

$$r_{C_1C_2} = \sqrt{\sum_{j=1}^{n} \frac{\left(P_{C_1 j}^{t} - P_{C_2 j}^{t}\right)}{d(v_j)}} \qquad \text{(IV. 5)}$$

onde $d(v_j)$ é o grau do vértice v_j e $P_{C_1 j}^{t}$ é a probabilidade da comunidade C_1 ir para o vértice v_j em *t* passos. A matriz de transição do processo do caminho aleatório é dada por P, e cada elemento desta matriz é calculado através da seguinte expressão: $P_{ij} = \frac{a_{ij}}{d(v_i)}$, sendo a_{ij} o elemento da matriz de adjacência que se encontra na *i*-ésima linha e *j*-ésima coluna e $d(v_i)$ é o grau do vértice v_i. Já a entrada associada à *i*-ésima linha e a *j*-ésima coluna da matriz de transição P^t, representa a probabilidade do vértice v_i ir para o vértice v_j através do caminho aleatório com tamanho *t*, definido através da seguinte expressão (IV. 6), (PONS & LAPATY, 2006):

$$P_{ij}^{t} = \sum_{\alpha=1}^{n} \lambda_{\alpha}^{t} v_{\alpha}(i) u_{\alpha}(j) \qquad \text{(IV. 6)}$$

onde λ é o autovalor das matrizes P e P^T, v é o autovetor da matriz P associando ao autovalor λ e u é o autovetor da matriz P^T associando ao autovalor λ.

Com estas medidas de distância e probabilidade definidas, o algoritmo segue o mesmo princípio do *Fast Greedy*, que é a de um algoritmo de clusterização hierárquico, o qual apresenta uma abordagem aglomerativa

As etapas do algoritmo *Walktrap* são descritas abaixo. como pode ser visto em (PONS & LAPATY, 2006). Em um primeiro momento, define-se a partição contendo apenas um vértice, $P_1 = \{\{v_i\}, v_i \in V\}$ e então são calculadas as distâncias entre todos os vértices adjacentes a v_i. Em seguida, o algoritmo segue as seguintes, etapas:

• Escolher duas comunidades C_1 e C_2 em P_j de acordo com o critério baseado nas distâncias entre as comunidades. Para que duas comunidades sejam unidas, o valor do quadrado das distâncias entre as comunidades, expressão (IV. 7), deve ser mínimo:

$$\Delta\sigma\,(C_1, C_2) = \frac{1}{n} \frac{|C_1||C_2|}{|C_1|+|C_2|} r_{C_1C_2}^2 \qquad \text{(IV. 7)}$$

• Fazer a junção dessas duas comunidades em uma nova comunidade

$C_3 = C_1 \cup C_2$ e assim é criada uma nova partição: $P_{j+1} = (P_j \setminus \{C_1, C_2\}) \cup \{C_3\}$;

• Atualizar as distâncias entre as comunidades. Depois de *n - 1* passos, o algoritmo termina e é obtido P_n = V.

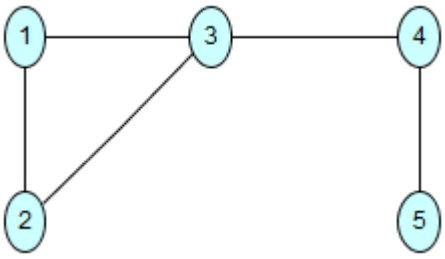


Figura 14: Grafo $\boldsymbol{G_4}$
Fonte: Os Autores.

A fim de exemplificar a execução deste algoritmo considere o grafo G_6 da Figura 14. Assim a matriz de adjacência, *A* e a matriz de transição, *P*, são apresentadas a seguir:

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad ; \quad P = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 1/3 & 1/3 & 0 & 1/3 & 0 \\ 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Os graus dos vértices do grafo G_4 são: *d* (v_1) = 2; *d* (v_2) = 2; *d* (v_3) = 3; *d* (v_4) = 2; *d* (v_5) =1. Desta maneira são apresentados os autovalores da matriz *P*, λ, e os autovetores da matriz *P*, v, associados aos autovalores λ na Tabela 6.

Tabela 6: Autovalores $\boldsymbol{\lambda}$ da matriz *P* e os respectivos autovetores da matriz *P*.

	$\boldsymbol{v_1}$		$\boldsymbol{v_2}$		$\boldsymbol{v_3}$		$\boldsymbol{v_4}$		$\boldsymbol{v_5}$
	1,00		-0,12		-0,34		-0,70		0,34
	1,00		-0,12		-0,34		0,70		0,34
$\boldsymbol{\lambda_1}$=1,00	1,00	λ_2=-0,86	0,30	λ_3=0,65	-0,10	λ_4=-0,50	0,00	λ_4=-0,29	-0,54
	1,00		-0,60		0,47		0,00		-0,19
	1,00		0,70		0,72		0,00		0,66

Fonte: os autores.

Abaixo é apresentado a transposta da matriz de transição, P^T, e na Tabela 7 os autovetores da matriz P^T, u, associados aos autovalores λ:

$$P^T = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{3} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{3} & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 1 \\ 0 & 0 & 0 & \frac{1}{2} & 0 \end{pmatrix}$$

Tabela 7: Autovalores da matriz P e os respectivos autovetores da matriz $\boldsymbol{P^T}$.

	$\boldsymbol{u_1}$		$\boldsymbol{u_2}$		$\boldsymbol{u_3}$		$\boldsymbol{u_4}$		$\boldsymbol{u_5}$
	0,42		-0,13		-0,43		-0,70		0,33
	0,42		-0,13		-0,43		0,70		0,33
$\boldsymbol{\lambda_1}$=1,00	0,64	λ_2= - 0,85	0,56	λ_3= 0,65	-0,20	λ_4= - 0,50	0,00	λ_4= - 0,29	-0,79
	0,42		-0,69		0,60		0,00		-0,19
	0,21		0,40		0,46		0,00		0,32

Fonte: os autores.

Considerando $t = 1$, para este exemplo, é calculado a P^1 do grafo G_6 através da expressão (IV. 6):

$$P^1 = \begin{pmatrix} 0{,}225 & 0{,}725 & 0{,}824 & 0{,}239 & 0{,}119 \\ 0{,}725 & 0{,}225 & 0{,}824 & 0{,}239 & 0{,}119 \\ 0{,}545 & 0{,}545 & 0{,}380 & 0{,}530 & 0{,}129 \\ 0{,}238 & 0{,}238 & 0{,}824 & 0{,}245 & 0{,}585 \\ 0{,}238 & 0{,}238 & 0{,}359 & 0{,}171 & 0{,}126 \end{pmatrix}$$

Após ser calculada a matriz de transição P^1, em um primeiro momento, o algoritmo é inicializado com uma partição contendo apenas um vértice, $P_1 = \{v_1, v_1 \in V\}$ e, posteriormente, são calculadas as distâncias entre todos os vértices adjacentes ao vértice v_1. Desta maneira, a primeira etapa do algoritmo *Walktrap* se dá através da escolha aleatória de um vértice para formar a primeira comunidade. Após esta escolha foi realizado o cálculo da distância entre o vértice v_1 e seus vértices adjacentes, ou seja, foi calculada a distância do vértice $v_1$ao vértice v_2 e do vértice v_1 ao vértice v_3, através da expressão (IV. 5).

Em um segundo momento**,** após calcular a distância entre estes vértices, é necessário juntar um novo vértice à comunidade inicial, C_0. A escolha do vértice para ser acrescentado à comunidade é realizada através do menor valor do quadrado das distâncias entre o vértice e a comunidade.

Para o exemplo acima, os valores de $\Delta\sigma\,(3, C_0)$ e $\Delta\sigma\,(2, C_0)$ são iguais independente da escolha do vértice para unir a comunidade. Escolhe-se, então, o vértice v_2. Após esta junção é então atualizada a partição P_2= $\{v_1, v_2\}$. Posteriormente, é atualizada a distância entre o vértice v_3 a comunidade C_1, e o vértice v_3 é adicionado à comunidade C_2, e novamente é atualizada a partição P_3= $\{v_1, v_2, v_3\}$. Este procedimento é repetido até serem formadas todas as comunidades, e o algoritmo termina com P_5 = $\{v_1, v_2, v_3, v_4, v_5\}$. A seguir é apresentada a Figura 15, que ilustra a formação de duas comunidades aplicando o algoritmo *Walktrap* ao grafo G_6 da Figura 15 e a Tabela 8 com os valores obtidos das distâncias *r* e $\Delta\sigma$.

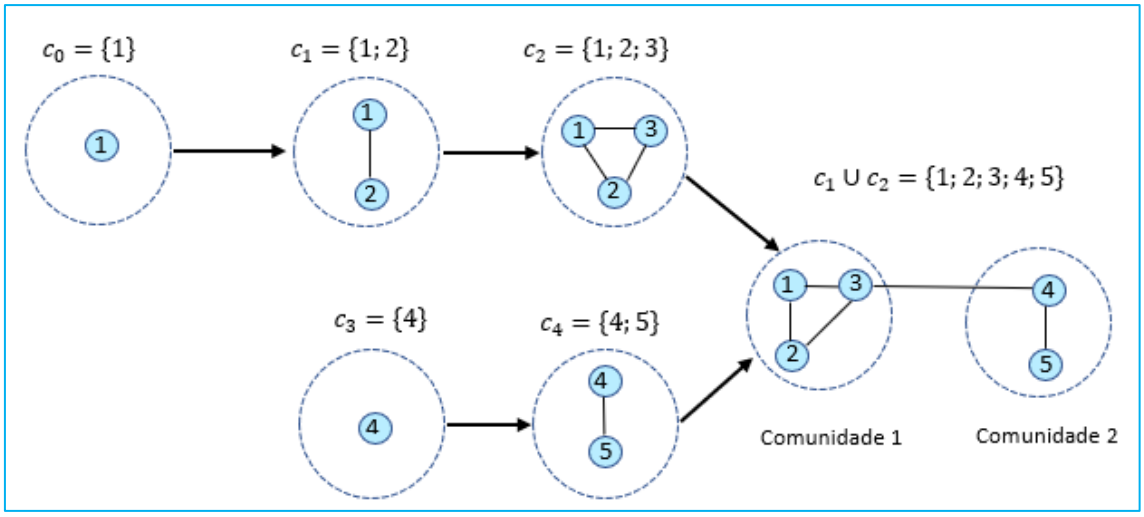


Figura 15: Etapas da Formação de duas comunidades utilizando o algoritmo Walktrap.
Fonte: Os Autores.

Tabela 8: Valores das distâncias r e Δσ

Partição	**Distância entre as comunidades**	**Δσ**
Partição1	$r_{12} = 0{,}50$	$\Delta\sigma\,(3, C_0) \cong 0{,}02$
	$r_{13} = 0{,}42$	$\Delta\sigma\,(2, C_0) \cong 0{,}02$
Partição 2	$r_{3C_1} = 0{,}33$	$\Delta\sigma\,(3, C_1) \cong 0{,}01$
	$r_{34} = 0{,}63$	$\Delta\sigma\,(3, 4) \cong 0{,}04$
Partição 4	$r_{4C_2} = 0{,}51$	$\Delta\sigma\,(4, C_2) \cong 0{,}03$
	$r_{45} = 0{,}84$	$\Delta\sigma\,(4, 5) \cong 0{,}07$
	$r_{C_2C_3} = 0{,}40$	$\Delta\sigma\,(C_2C_3) \cong 0{,}03$

Fonte: Os Autores.

De modo a exemplificar a execução do algoritmo de detecção de comunidades *Walktrap*, através do *software R*, a Figura 16 ilustra a divisão de uma rede, contendo 21 vértices e 114 arestas, em duas comunidades que estão representadas pelas cores vermelho e azul.

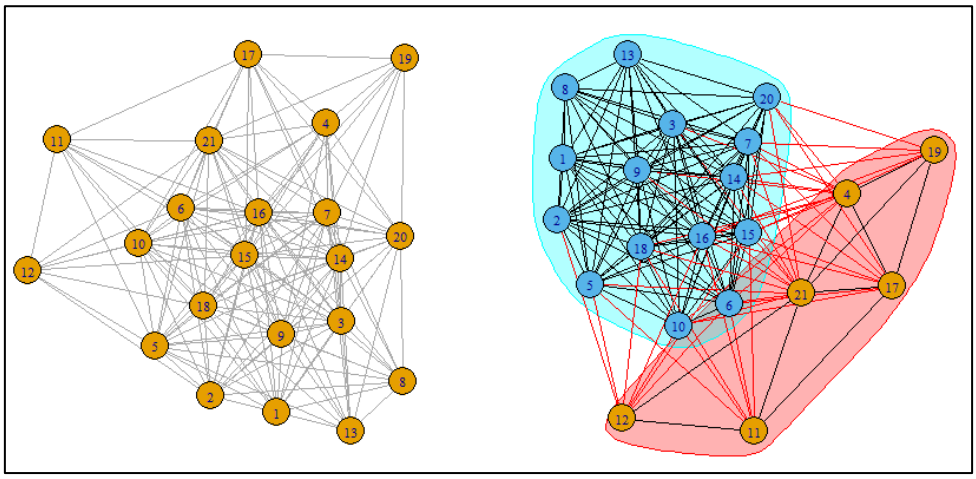


Figura 16: Representação da aplicação do algoritmo Walktrap
Fonte: Os Autores

A Figura 17, ilustra a representação hierárquica da rede em comunidades. Os números na parte inferior representam os vértices indviduais da rede e os retângulos em laranja e azul delimitam cada uma das duas comunidades.

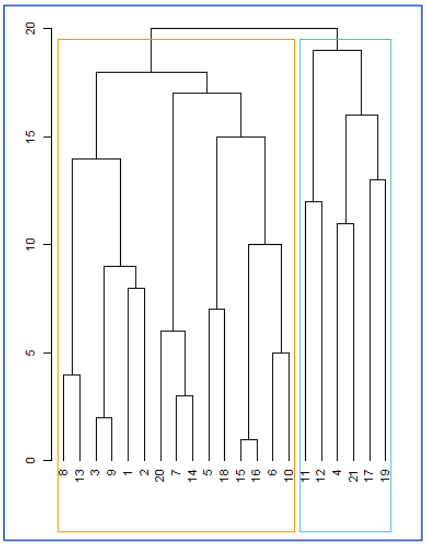


Figura 17: Dendograma utilizando Algoritmo Walktrap
Fonte: Os Autores

Capítulo V – Critério de Validação

Este capítulo tem por objetivo apresentar o conceito de critério de validação e os índices de validação que são utilizados nesta dissertação: índice de Rand e índice de Silhueta.

V.1 Conceitos básicos

Conforme apresentado anteriormente, ao aplicar os algoritmos de agrupamento são produzidos k grupos (k≥2). Contudo, alguns fatores podem influenciar na qualidade dos grupos produzidos, como a escolha da medida de similaridade ou, até mesmo, um dos parâmetros utilizados no algoritmo. Por esta razão é importante avaliar a qualidade resultados (agrupamentos) produzidos pelos algoritmos.

De modo geral, isso é feito tendo por base índices estatísticos que permitem medir quantitativamente a qualidade dos agrupamentos gerados. Estes índices estão associados a critérios de validação, (Oliveira, 2008).

Os critérios de validação são divididos em três tipos: externos, internos e relativos, (Jain & Dubes (1988), Oliveira (2008)). O Critério Externo de Validação avalia a qualidade dos agrupamentos gerados a partir de alguma estrutura externa conhecida, a priori, mais especificamente, em relação a uma solução conhecida a priori e à solução produzida por um algoritmo do conjunto de dados e o Critério Interno avalia o grau de compatibilidade entre a estrutura de grupos e os dados, utilizando apenas os próprios dados. Já o Critério Relativo de Validação avalia**,** dentre duas ou mais divisões de um conjunto**,** qual representaria melhor o conjunto de dados. Este critério também é utilizado, principalmente, para determinar o melhor valor para um parâmetro utilizado pelo algoritmo como, por exemplo, o número de grupos (k) a ser adotado, ou ainda comparar resultados de algoritmos diferentes, quando aplicados em uma mesma base de dados.

V.1.1 Índice de Rand

O índice de Rand está associado ao critério externo de validação, sendo sua aplicação condicionada ao conhecimento dos grupos gerados pelo algoritmo de

agrupamento e da estrutura de grupo pré-especificada, ou seja, para o cálculo deste índice, é necessário obter informações de todos os pares de objetos pertencentes aos grupos gerados pelo algoritmo de agrupamento e da estrutura de grupo pré-especificada.

Desta forma, define-se como devem ser comparadas as estruturas de grupos pré-especificada, conhecida a priori, e a estrutura de grupos gerados pelo algoritmo de agrupamento, da seguinte maneira: sejam *M* o conjunto dos grupos (com a alocação dos objetos) gerados por um algoritmo de agrupamento; *F* estrutura de grupos pré-especificada, conhecida a priori e *A, B, C e D*, os termos utilizados no critério de comparação entre estes grupos (SILVA; PERES; BOSCARIOLI, 2016), mais especificamente, *A* representa a quantidade de pares que pertencem ao mesmo grupo de *M* e à estrutura de grupos pré-especificada em *F*, *B* a quantidade de pares que pertencem ao mesmo grupo de *M* e à estrutura de grupos pré-especificada *F* diferentes, *C* a quantidade de pares que pertencem a grupos diferentes e à mesma estrutura de grupos pré-especificada *F* e *D* a quantidade de pares que pertencem a grupos diferentes e a estrutura de grupos pré-especificada diferentes.

A partir destas definições, o índice de Rand é definido pela seguinte equação:

$$R = \frac{(A + D)}{(A + B + C + D)} \qquad (V.1)$$

Um valor alto para o índice de Rand indica um elevado grau de similaridade entre os grupos e sua estrutura pré-especificada, ou seja, em relação à solução produzida por um algoritmo e uma solução conhecida a priori. É importante ressaltar que o valor máximo que este índice pode assumir é 1 e o valor mínimo é zero (SILVA; PERES; BOSCARIOLI, 2016).

Para exemplificar o cálculo do índice de Rand, considere: a estrutura de grupos *F* pré-especificada $F_1 = \{x_1; x_2; x_3; x_5\}$ e $F_2 = \{x_4; x_6; x_7\}$ e os grupos *M* gerados pelo algoritmo de agrupamento $M_1 = \{x_1; x_2; x_3 ; x_4\}$ e $M_2 = \{x_5; x_6; x_7\}$ ilustrado na Tabela 9.

Tabela 9: Representação dos objetos nos grupos

F_1	F_2	M_1	M_2
x_1	x_4	x_1	x_5
x_2	x_6	x_2	x_6
x_3	x_7	x_3	x_7
x_5	------	x_4	------

Fonte: Adaptado de (SILVA; PERES; BOSCARIOLI, 2016).

Para realizar os cálculos e obter os valores dos termos A, B, C e D, devemos comparar os pares de objetos considerando conjuntos M e F, como pode ser visto na Tabela 10.

Tabela 10: Exemplo da comparação dos pares entre os grupos e as partições

Comparação dos pares	Quantidade de pares
$A = \{(x_1, x_2); (x_1, x_3); (x_2, x_3); (x_6, x_7)\}$	#(A) = 4
B = $\{(x_1, x_4); (x_5, x_6); (x_5, x_7); (x_2, x_4); (x_3, x_4)\}$	#(B) = 5
C = $\{(x_1, x_5); (x_2, x_5); (x_3, x_5); (x_4, x_6); (x_4, x_7)\}$	#(C) = 5
D = $\{(x_1, x_6); (x_1, x_7); (x_2, x_6); (x_2, x_7); (x_3, x_6); (x_3, x_7); (x_4, x_5)\}$	#(D) = 7

Fonte: Adaptado de (SILVA; PERES; BOSCARIOLI, 2016).

Com os resultados obtidos acima, o índice de Rand é dado por:

$$R = \frac{(A+D)}{(A+B+C+D)} = \frac{(4+7)}{(4+5+5+7)} = \frac{11}{21} = 0{,}5238.$$

V.1.3 Índice de Silhueta

A aplicação do índice de Silhueta, proposto por Rousseeuw (1987), permite avaliar os agrupamentos encontrados, no que tange à qualidade (pertinência) da alocação dos objetos aos grupos (ROUSSEEUW, 1987). Mais especificamente, esse índice evidência o posicionamento dos objetos nos grupos, o que permite identificar se cada um dos destes objetos estão bem posicionados, ou seja, se foram alocados ao grupo ideal. Para cada objeto x_i o valor da Silhueta, $S(x_i)$, é calculado através da seguinte expressão:

$$S(x_i) = \frac{b(x_i) - a(x_i)}{\max\{a(x_i), b(x_i)\}}, \qquad (V.3)$$

onde

- $a(x_i) = \frac{1}{|g_z|} \sum d(x_i, x_j), \quad \forall x_i \neq x_j,\ x_i\ e\ x_j \in\ g_z$ $(V.4)$
- $d_{externa}(x_i, g) = \frac{1}{|g_t|} \sum d(x_i, x_j), \quad x_i \notin g_t\ ,\ \forall x_j \in\ g_t$ $(V.5)$
- $b(x_i) = \min d_{externa}(x_i, g_t)\ ,\ g_t\ \neq\ g_z\ , x_i\ \in\ g_t$ $(V.6)$

Nestas equações, temos que $b(x_i)$ é o mínimo da distância externa entre o objeto x_i em relação a todos os objetos do grupo mais próximo a ele. Já o termo $a(x_i)$ representa a dissimilaridade média do objeto x_i em relação a todos os objetos do mesmo grupo; $|g_z|$ representa a quantidade de objetos do grupo g_z e analogamente, $|g_t|$ representa a quantidade de objetos do grupo g_t.

O valor da Silhueta $S(x_i)$, de cada objeto varia em [-1,1]. Se observarmos os possíveis valores de $S(x_i)$, verificamos que, se $a(x_i) > b(x_i)$, $S(x_i)$ assume valores negativos, o que significa que os objetos do grupo g_z estão mais distantes dos objetos do seu próprio grupo e mais próximos aos objetos do seu grupo adjacente, o que não torna interessante valores de Silhueta deste tipo. Entretanto, se verificarmos $a(x_i) < b(x_i)$, o valor do coeficiente de Silhueta fica positivo, o que indica que o objeto está bem posicionado em relação ao seu grupo e que os objetos do grupo g_z estão bem próximos dos objetos do seu próprio grupo. Além disso, existe a possiblidade de que $a(x_i) = b(x_i)$, ou seja, $S(x_i) = 0$.

Alternativamente ao índice para cada objeto, utiliza-se, para a avaliar a solução produzida pelo algoritmo, a média das silhuetas ($\bar{S}$), conforme a expressão apresenta na equação (V.7), onde *n* é o número de objetos da base de dados e $S(x_i)$ é o coeficiente de Silhueta do objeto x_i.

$$\bar{S} = \frac{\sum_{i=1}^{n} S(x_i)}{n} \qquad (V.7)$$

Quando a Silhueta média tem um valor acima de 0.7, implica boa estrutura de grupos. Valores na faixa de 0.51 e 0.7 indicam uma estrutura razoável de grupos. Já valores entre 0.26 e 0.5 indicam uma estrutura fraca (KAUFMAN & ROUSSEEUW, 1990, p. 88). Tais informações estão sumarizadas no Quadro 2.

Quadro 2: Interpretação do Coeficiente de Silhueta

Silhueta Média	Interpretação
0,71 – 1,00	Estrutura Forte
0,51 – 0,70	Estrutura Razoável
0,26 – 0,50	Estrutura Fraca
≤ 0,25	Não foi encontrada uma estrutura substancial

Fonte: Kaufman & Rousseeuw 1990, p. 88.

A Tabela 11 ilustra três grupos com seus respectivos objetos e suas coordenadas no plano cartesiano. Por exemplo, o primeiro grupo possui 4 objetos, $\{x_1, x_2, x_3, x_4\}$, com as suas respectivas coordenadas no plano cartesiano {(1,1), (2,2), (3,3), (3,5)}.

Tabela 11: Grupos exemplos

Grupo 1	Grupo 2	Grupo 3
x_1= (1,1)	x_5 = (5,6)	x_8= (8,2)
x_2 = (2,3)	x_6 = (5,8)	x_9 = (9,3)
x_3 = (3,3)	x_7= (6,7)	x_{10} = (7,2)
x_4 = (3,5)	----------	----------

De modo a exemplificar como é realizado o cálculo da Silhueta, a seguir apresentamos o cálculo detalhado para identificar o coeficiente de Silhueta para o objeto x_3 do grupo 1, g_1: primeiramente é necessário calcular a média da distância de cada objeto x_3 em relação aos outros objetos pertencentes ao mesmo grupo. Assim temos que, $d\ (x_3, x_1)$ = 2,83; $d\ (x_3, x_2)$ = 1,00; $d\ (x_3, x_4)$ = 2,00. Então, a dissimilaridade média é: $a(x_3) = \frac{2{,}83 + 1{,}00 + 2{,}00}{3}$ = 1,94. No próximo passo é necessário realizar o cálculo da distância Euclidiana do objeto x_3 para cada um dos objetos pertencentes aos outros grupos, os quais são dados pelos seguintes valores.

1. $d\ (x_3, x_5)$ = 3,60; $d\ (x_3, x_6)$ = 5,38; d (x_3, x_7) = 5,00. Assim, $d_{externa}(x_3, g_2) = \frac{3,60+5,38+5,00}{3}$ = 4,66.
2. $d\ (x_3, x_8)$ = 5,09; $d\ (x_3, x_9)$ = 6,00; $d\ (x_3, x_{10})$ = 4,12. Então, $d_{externa}(x_3, g_3) = \frac{5,09+6,00+4,12}{3}$ = 5,07.

O valor de $b(x_3)$, conforme a Equação (V.6), é o mínimo entre $d_{externa}(x_3, g_2)$ e $d_{externa}(x_3, g_3)$. Sendo assim, $b(x_3)$ = 4,66. Logo, ao comparar o objeto x_3, no que tange sua relação aos grupos externos, o grupo mais próximo ao objeto x_3 é g_2. Por fim, conforme a Equação $(V.3)$ temos:

$$S(x_3) = \frac{b(x_3) - a(x_3)}{\max\{a(x_3), b(x_3)\}} = \frac{4,66 - 1,94}{4,66} = 0,58$$

No exemplo apresentado acima a Silhueta apresenta valor dentro da faixa de 0,51 e 0,70 o que indica uma estrutura razoável de agrupamento.

Capítulo VI – Metodologia Utilizada

O presente capítulo traz a descrição d o procedimento utilizado para subsidiar a Dissertação em questão, no que tange à metodologia utilizada.

VI.1 Procedimentos para análise de dados

Nesta dissertação o objetivo foi analisar e comparar o desempenho de dois algoritmos de Detecção de Comunidades em redes (*Fast Greedy* e *Walktrap*) frente a três algoritmos de agrupamento não-hierárquicos: *k-means, PAM e CLARA*. Para realizar tal análise foram consideradas as quatro etapas descritas a seguir: (i) obtenção dos dados; (ii) geração das redes; (iii) aplicação dos algoritmos: *k-means, PAM, CLARA, Walktrap* e *Fast Greedy*; (iv) aplicação dos critérios de validação: índice de Rand e Silhueta.

Na primeira etapa, os dados foram obtidos através de uma busca em sites que têm base dados reais, ligados à área da educação, a saber: Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e o Instituto Nacional de Estudos e Pesquisa Educacionais Anísio Teixeira (INEP). A partir desta pesquisa foram obtidas 29 bases cuja descrição é apresentada mais à frente.

Já a segunda etapa, associada à geração das redes, consistiu em construir grafos a partir das bases de dados supracitadas. É importante destacar que, para aplicar os algoritmos de agrupamento, foi necessário ter apenas as bases de dados, e realizar os procedimentos de pré-processamento descritos na Subseção III.3.2. Em contrapartida, para aplicar os algoritmos de Detecção de Comunidades faz-se necessário ter, como dado de entrada, um grafo. Desta maneira, foram gerados os grafos associados a estas bases de dados, já adquiridas. Para tal, optou-se por realizar dois experimentos para a construção destes grafos, quais sejam: (1) construir o grafo utilizando quartis e (2) construir o grafo mediante árvore geradora mínima. A seguir esses procedimentos são descritos.

Experimento de Construção I: através das bases de dados obtidas na Etapa 1, a partir dos *n* objetos com *p* atributos foram calculadas as distâncias d_{ij}, entre todos os objetos tomados dois a dois, sendo produzida uma matriz, $\left(D = \left[d_{ij}\right]_{nxn}\right)$.

A partir dessas matrizes, foi possível criar um grafo completo, $G = (V, E)$, onde

os vértices são os objetos da base e suas arestas têm, como pesos, as distâncias d_{ij} correspondentes aos vértices v_i e v_j de *V*. Posteriormente à criação desse grafo, foi criado um subgrafo gerador de G, G', tal que $G' = (V, E')$ e $E' \subset E$. Para construção de G', mais especificamente, de quais arestas (E) de G iriam compor G', foram calculados, em um primeiro passo, os quartis das distâncias de *D*. Assim, cada objeto corresponderá a um vértice de G' e dois vértices v_i e v_j de G' são conectados por uma aresta, se a distância (d_{ij}) entre eles for menor ou igual ao 3º quartil das distâncias. Desta forma, foi possível criar a matriz de adjacência *A* do grafo *G*`.

O segundo experimento, via árvore geradora mínima (AGM), foi realizado da seguinte maneira: a partir do grafo completo, o qual fora gerado de forma semelhante do experimento I, foi determinado a mediana, β, das distâncias de *D*. Posteriormente, foi aplicado o algoritmo PRIM que gera a AGM, *T*.

Após ser criada a AGM, *T*, foram adicionadas a esta árvore novas arestas (E) do grafo completo, G. Desta forma, foi possível criar o grafo G'', tal que $G'' = (V, E'')$ e $E'' \subset E$. Para construção de G'', mais especificamente, quais arestas (E) de G irão compor G'', foram adicionadas arestas à AGM, *T*, de acordo com a seguinte condição, a saber: foram calculadas as distâncias (d_{ij}) entre todos os vértices v_i e v_j desconectados pertencentes à AGM, T , e se $(d_{ij}) \leq \beta$, então as arestas foram adicionadas à AGM, *T*, definindo assim as arestas de E''. Após serem adicionadas, a esta árvore, todas as arestas cuja distâncias estejam abaixo da mediana, foi possível criar o grafo $G'' = (V, E'')$.

Os dois experimentos foram realizados mediante uma função implementada no *software* livre *R*.

Na terceira etapa, foram aplicados a todos aos *subgrafos* geradores, G' *e* G'' , criados na etapa anterior, os três algoritmos de agrupamento *k-means*, *CLARA* e *PAM* e dois algoritmos de Detecção de Comunidades: *Fast Greedy* e *Walktrap*.

Para tal, foi necessário utilizar os seguintes pacotes disponíveis no software *R*: *igraph, cluster, stats*. Ao aplicar o algoritmo *Fast Greedy* através da função *cluster_fast_greedy*, por meio do pacote *igraph*, a função retorna: o número de comunidades detectadas; o valor da modularidade (*Q*) e um vetor de dimensão *n* onde cada posição contém o valor correspondente à comunidade a qual cada objeto está alocado. Já para aplicar os algoritmos *k-means*, *CLARA* e *PAM* é necessário fornecer

como dados de entrada o número *k* de grupos, a matriz de distâncias para os algoritmos *PAM* e *CLARA e* a matriz com os dados originais *nxp* para o algoritmo *k-means*. Ressalta-se que, nos experimentos realizados como os algoritmos de agrupamento, o valor de *k* foi definido a partir do número de comunidades produzidos pelo algoritmo *Fast Greedy*.

Na quarta etapa são aplicados os índices de Silhueta e de Rand. Na quinta etapa, as análises foram realizadas da seguinte maneira:

• Comparar os valores do índice de Silhueta e do índice de Rand, a fim de avaliar os melhores agrupamentos gerados entre os algoritmos *k-means*, *PAM*, *CLARA*, *Walktrap* e *Fast Greedy*;

• Comparar os tempos computacionais de execução dos algoritmos;

• Verificar se, ao trocar o algoritmo *CLARA* em detrimento do *PAM* resulta em algum ganho de qualidade em relação aos índices de Silhueta e de Rand.

A Figura 18 apresenta as etapas da metodologia utilizada. A Figura 18.a ilustra as etapas do primeiro experimento I, baseado no cálculo dos quartis, e a Figura 18.b apresenta etapas do experimento II, via árvore geradora mínima (AGM).

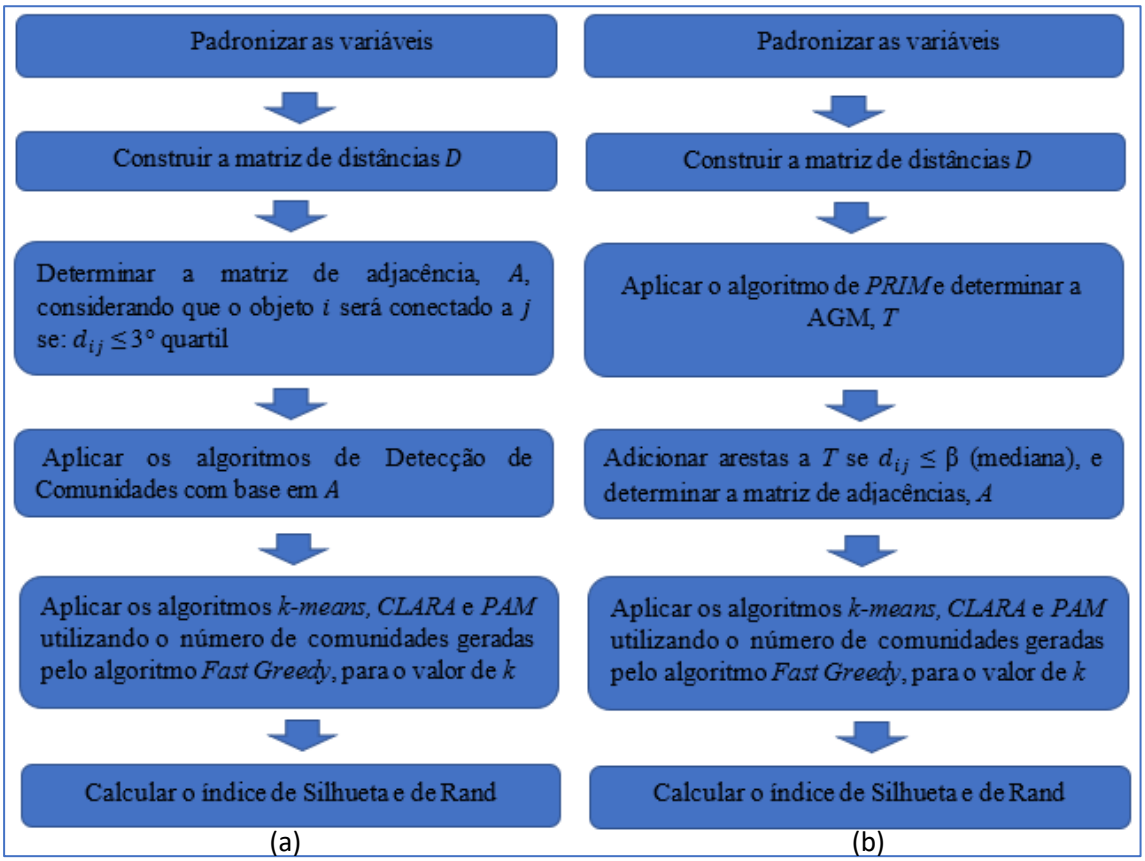


Figura 18: Etapas da metodologia utilizada

Capítulo VII – Resultados e Análises

VII.1 Experimentos

Todos os experimentos para esta dissertação foram realizados em um computador dotado de processador Intel I5 de 2.4 GHz, com 4 GB de memória RAM e sistema operacional Windows 8. O *software* utilizado para a análise é o *R* v. 3.5.2 (*R Core Team*, 2019)

Os resultados foram obtidos utilizando os pacotes *igraph*, *cluster*, *clv*, *parallel*, *dplyr*, *tidyverse*. Foram selecionadas 29 bases de dados reais (descritas no Quadro 3) de porte pequeno, médio e grande (em relação ao número de registros, ou seja, objetos), sendo todas ligadas à educação. Para tal, foi necessário realizar os procedimentos de pré-processamento de remoção de registros com valores faltantes e outliers encontrados nas bases de dados.

Após esta etapa foi realizado o experimento 1: em um primeiro momento, foram executados todos os algoritmos para as 19 bases de dados menores e foram executados os algoritmos para uma amostra das 10 bases de dados maiores. As 10 bases de grande de porte têm o número de objetos na faixa de 8.823 a 57.717, o que demandou um alto tempo computacional, no que se refere à execução dos algoritmos. Desta maneira, a fim de contornar esta situação, e realizar as análises em tempo hábil, foi realizado o experimento 2: consistiu em selecionar amostras de tamanhos diferentes do conjunto de dados, de modo a aplicar os algoritmos e obter os resultados em um menor tempo computacional.

Mais especificamente, foram selecionadas 100 amostras de tamanhos diferentes (1.000, 2.000 e 3.000) destas 10 bases de dados de grande de porte. Posteriormente à seleção dessas amostras, foram aplicados os algoritmos: *PAM*, *CLARA*, *Walktrap* e *Fast Greedy*. Num segundo momento foram calculadas duas medidas estáticas de posição a saber, média e mediana para: o índice Silhueta, o índice Rand e a modularidade. Finalizado estes cálculos, foi feita uma comparação dos resultados obtidos do experimento 1.

Após realizar o experimento 2 observou-se que os valores referentes à média e mediana, encontrados das amostras, cujo resultado foi obtido mediante aplicação os algoritmos: *PAM*, *CLARA*, *Walktrap* e *Fast Greedy*, não diferiram significativamente dos valores encontrados na execução de uma única amostra do conjunto de dados, ou seja, do

experimento 1. Desta maneira, os resultados dos experimentos computacionais para esta dissertação se deram através das saídas geradas do experimento 1. Mais especificamente o experimento 1 considerou todas as bases de dados, incluído as bases que foram citadas no experimento 2.

VII.1.1 Base de Dados

Conforme já comentado, para esta dissertação foram utilizadas 29 bases de dados reais (descrição no Quadro 3) . Todas essas bases de dados foram obtidas através de uma busca em dois sites que têm base dados reais, ligados à área da educação, a saber: Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e Instituto Nacional de Estudos e Pesquisa Educacionais Anísio Teixeira (INEP).

O Quadro 3 traz para as 29 bases dados as seguintes informações: base de dados, a quantidade de registros (objetos) em cada base de dados antes da remoção de valores faltantes (NAs) e outliers; a quantidade de registros após a retirada de valores faltantes; a quantidade de registros após a retirada de *outliers* e número de variáveis (todas quantitativas) em cada uma das bases.

Após a obtenção de todas as bases de dados descritas no Quadro 3, os dados contidos nestas bases foram divididos por região, no tange às cinco grandes regiões geográficas do Brasil: Centro Oeste, Norte, Nordeste, Sudeste e Sul.

No que se refere às bases do Programa Institucional de Bolsas de Iniciação Científica (PIBIC), PIBIC - por Estado, de Produtividade em Pesquisa - por Instituição de Ensino (Área: Exatas), Produtividade em Pesquisa - por Estado (Todas as Áreas) e Produtividade em Pesquisa - por Instituição de Ensino (Todas as Áreas), os dados utilizados para compor estas bases foram retirados do *site* do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Do *site* do CNPq, as informações foram retiradas do Mapa de Investimentos**,** o qual disponibiliza o acesso às informações sobre as bolsas que estão em vigência, sendo possível extrair informações quantitativas sobre as bolsas fornecidas. Estes quantitativos estão detalhados por: UF, área do conhecimento, instituição de ensino, entre outros.

Já em relação às bases Exame Nacional do Ensino Médio (Enem), foram retiradas do *site* do Instituto Nacional de Estudos e Pesquisa Educacionais Anísio Teixeira (INEP).

Mais especificamente, do *site* do INEP, as informações foram retiradas dos Indicadores Educacionais o qual disponibiliza dados a respeito do desempenho dos estudantes da Educação Superior, Enade, e do desempenho dos estudantes do Ensino Médio, Enem. Além disso, foram retiradas informações para compor a base de dados Avaliação Nacional de Alfabetização (ANA), da Avaliações e Exames Educacionais referente ao resultado do Sistema de Avaliação da Educação Básica (Saeb), o qual disponibiliza informações do desempenho educacional dos estudantes do 3° ano do ensino fundamental das escolas públicas brasileiras. E, por fim, para compor as bases de dados Instituições de Educação Superior (IES), as informações foram retiradas do Censo da Educação Superior, o qual é realizado anualmente pelo Inep, e disponibiliza informações sobre as instituições de educação superior que ofertam cursos de graduação e sequenciais de formação específica e ainda disponibiliza as informações de seus alunos e docentes.

Quadro 3: Quantidade de registros da Base de Dados

Base de Dados	Quantidade de Registros	Quantidade de Registros após retirada de_Nas	Quantidade de Registros após a retirada de outliers	Número de Variáveis
ENADE_CENTRO_OESTE	16.466	14.247	14.247	16
ENADE_NORTE	20.805	16.688	16.688	16
ENADE_NORDESTE	59.245	49.812	49.812	16
ENADE_SUDESTE	70.100	57.717	57.717	16
ENADE_SUL	25.788	22.767	22.767	16
ANA_CENTRO_OESTE	35	35	31	13
ANA_NORTE	66	66	60	13
ANA_NORDESTE	87	87	70	13
ANA_SUDESTE	41	41	32	13
ANA_SUL	31	31	25	13
ENEM_POR_ESCOLA_CENTRO_OESTE	9.103	5220	5.173	5
ENEM_POR_ESCOLA_NORTE	8.823	4955	4.889	5
ENEM_POR_ESCOLA_NORDESTE	28.901	15.895	15.621	5
ENEM_POR_ESCOLA_SUDESTE	49.498	26.341	25.890	5
ENEM_POR_ESCOLA_SUL	21.578	12.881	12.755	5
IES_ALUNOS	286	286	222	4
IES_DOCENTES	286	286	250	7
IES_GERAL	286	286	253	7
IES_CENTRO_OESTE	19	19	13	14
IES_NORTE	24	24	17	14
IES_NORDESTE	59	59	48	14
IES_SUDESTE	153	153	100	14
IES_SUL	31	31	27	14
PIBIC_1	27	27	26	1
PIBIC_2	356	356	319	1
PRODUTIVIDADE_1	192	192	192	1
PRODUTIVIDADE_2	25	25	23	1
PRODUTIVIDADE_3	27	27	27	1
PRODUTIVIDADE_4	529	529	529	1

Fonte: Os Autores.

VII.2 Análises

Conforme já comentado, através do experimento I, foi possível criar dois grafos distintos, para diferentes análises. O primeiro grafo foi criado através dos quartis e o segundo grafo mediante a árvore geradora mínima (AGM). Para estes grafos foram realizados oito tipos distintos de experimentos, considerando o tipo de grafo construído versus algoritmo aplicado versus medida de dissimilaridade, conforme ilustra o Quadro 4:

Quadro 4: Descrição das configurações do experimento I

Tipo de Experimento	Algoritmo	Construção do Grafo	Medida de Dissimilaridade
1	*PAM*	Via quartil	Distância Euclidiana
2	*CLARA*	Via quartil	Distância Euclidiana
3	*PAM*	Via AGM	Distância Euclidiana
4	*CLARA*	Via AGM	Distância Euclidiana
5	*PAM*	Via quartil	Distância de Manhattan
6	*CLARA*	Via quartil	Distância de Manhattan
7	*PAM*	Via AGM	Distância de Manhattan
8	*CLARA*	Via AGM	Distância de Manhattan

Fonte: Os autores.

VII.2.1 Apresentação dos resultados

De modo a comparar os resultados dos algoritmos: *Fast Greedy*, *Walktrap*, *k-means*, *PAM* e *CLARA*, em um primeiro momento, foram aplicados, às bases de dados, os algoritmos *Fast Greedy* e *Walktrap*. Como entrada para esses dois algoritmos, foi dado o grafo construído através dos quartis e da árvore geradora mínima e as saídas geradas foram o valor da modularidade (Q) e o número de comunidades.

Em um segundo momento, foram aplicados às bases de dados os algoritmos de agrupamento, *k-means*, *CLARA* e *PAM*. Para aplicar esses algoritmos, faz-se necessário

o conhecimento, a priori, do número de grupos *k*. Sendo assim, para realizar a análise desses cinco algoritmos e possibilitar uma comparabilidade entre os algoritmos de detecção de comunidade e os algoritmos de agrupamento**,** foi utilizado o mesmo número de comunidades geradas pelo algoritmo *Fast Greedy*, para o valor de *k*. A justificativa para utilização do resultado do algoritmo *Fast Greedy* se dá pelo fato de ser um dos mais utilizados e conhecidos na literatura.

VII.2.2 Apresentação dos resultados – Comparação dos algoritmos de agrupamento e de Detecção de Comunidades

De modo a comparar os algoritmos foi calculada, em um primeiro momento, a Silhueta média , conforme pode ser visto na Figura 19.

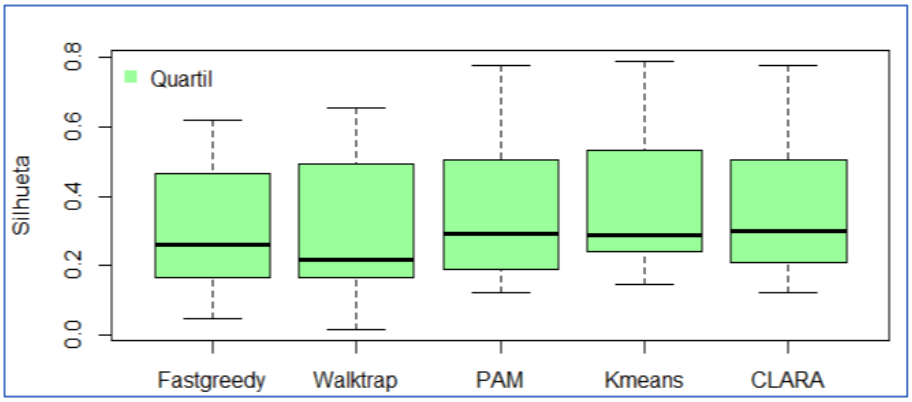


Figura 19: Comparação dos métodos através do índice de Silhueta
Fonte: os autores

Na Figura 19 cada *boxplot* foi obtido através da distribuição das Silhuetas médias, derivadas da aplicação dos algoritmos *Fast Greedy*, *Walktrap*, *PAM*, *CLARA* e *k-means* nas 29 bases de dados. Para tal, adotou-se a distância Euclidiana como medida de dissimilaridade e foram utilizados os quartis das distâncias dos objetos tomados dois a dois para criar o grafo e, consequentemente, aplicar os algoritmos de Detecção de Comunidades.

Foi obtido um valor máximo de 0,787 para o índice de Silhueta média, o que indica que os objetos estão bem alocados, já que valores dentro do intervalo de 0,71 a 1,00 indicam uma estrutura de agrupamento forte. Valores máximos dentro deste intervalo podem ser observados apenas nos algoritmos de agrupamento *PAM* e *k-means* de acordo

com a Figura 19. Os maiores valores, os limites superiores, do índice de Silhueta média são do algoritmo *PAM* e *k-means* 0,774 e 0,787, respectivamente. Para os algoritmos *Fast Greedy* e *Walktrap* os valores máximos, limites superiores do *boxplot*, estão na faixa de 0,619 a 0,653. Valores estes que indicam a formação de uma estrutura razoável dos objetos.

Já ao observar o valor da mediana representada pela linha interna do *boxplot*, observa-se, novamente, que os maiores valores medianos das Silhuetas médias foram produzidos pelos algoritmos *k-means* e *PAM* frente aos valores das medianas dos algoritmos *Fast Greedy* e *Walktrap*. Em contrapartida, observando os valores obtidos destas medianas para o *Fast Greedy*, *Walktrap, PAM* e *k-means*, que são respectivamente: 0,261; 0,217; 0,290; 0,286, entende-se que, em termos medianos, foram formadas estruturas de agrupamento fracas, , com exceção do algoritmo Walktrap que apresentou um valor abaixo deste intervalo.

Quando é realizada a comparação destes mesmos algoritmos através de outro índice de validação, mais especificamente o índice de Rand, os resultados encontrados quanto à alocação dos objetos aos grupos são bem similares, conforme pode ser visto através da Figura 20.

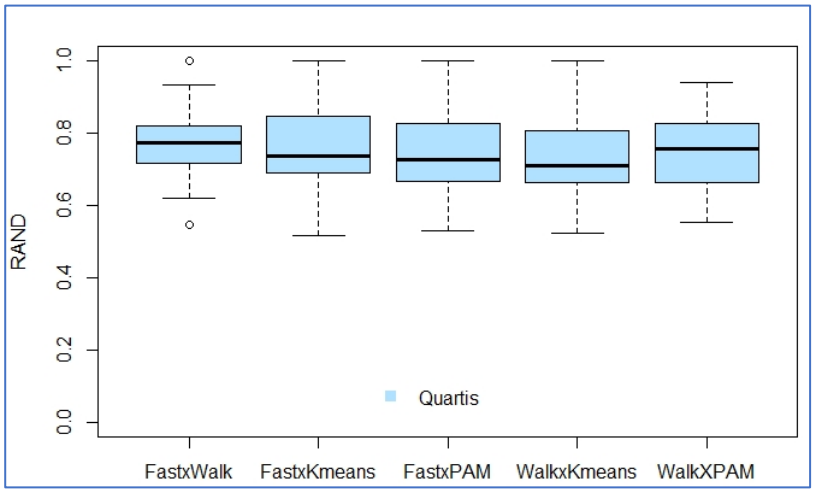


Figura 20: Comparação dos Algoritmos através do índice de Rand
Fonte: os autores

Aqui, o índice Rand foi utilizado para comparar grupos ou clusters gerados por um algoritmo de agrupamento ou um algoritmo de detecção de comunidades a grupos a partir dos resultados dos índices de Rand obtidos para as bases de dados, da seguinte maneira: os clusters gerados a partir da aplicação do algoritmo *Fast Greedy* (conhecidos

a priori) foram comparados aos clusters gerados da aplicação do algoritmo *Walktrap*, o segundo *boxplot* foi criado através da comparação dos clusters gerados da aplicação do algoritmo *Fast Greedy* aos grupos gerados da aplicação do algoritmo de agrupamento *k-means*, o terceiro *boxplot* foi feito através da comparação dos clusters gerados da aplicação do algoritmo *Fast Greedy* aos grupos gerados da aplicação do algoritmo de agrupamento *PAM*, o quarto e o quinto *boxplot* seguiram procedimentos análogos aos anteriores.

Tendo em vista que este índice varia no intervalo [0,1], e que os valores próximos a 1 indicam que existe alto grau de similaridade quanto às soluções (alocação dos objetos aos grupos) produzidas pelos algoritmos, pelos resultados apresentados na Figura 20 (valores do índice de Rand iguais ou superiores a 0,7) há indícios de uma alta similaridade entre os grupos e clusters obtidos da aplicação dos algoritmos.

Os resultados destas comparações estão expostos, de uma maneira mais detalhada, no Anexo I, onde estão descritos os valores do índice de Rand, separadamente por base de dados, assim como, por valor de *k*.

A Figura 21, ilustra o percentual das bases de dados para as quais foram obtidos valores iguais ou superiores a 0,70 do índice de Rand. A partir desta figura é possível observar que, em 79 % das bases de dados, foram obtidos valores iguais ou superiores a 0,70 do índice de Rand, mais especificamente, como pode ser visto na comparação entre os algoritmos *Fast Greedy* e o algoritmo *Walktrap*. O que significa uma boa qualidade, em relação a este índice, na aplicação destes algoritmos em grande parte das bases de dados analisadas.

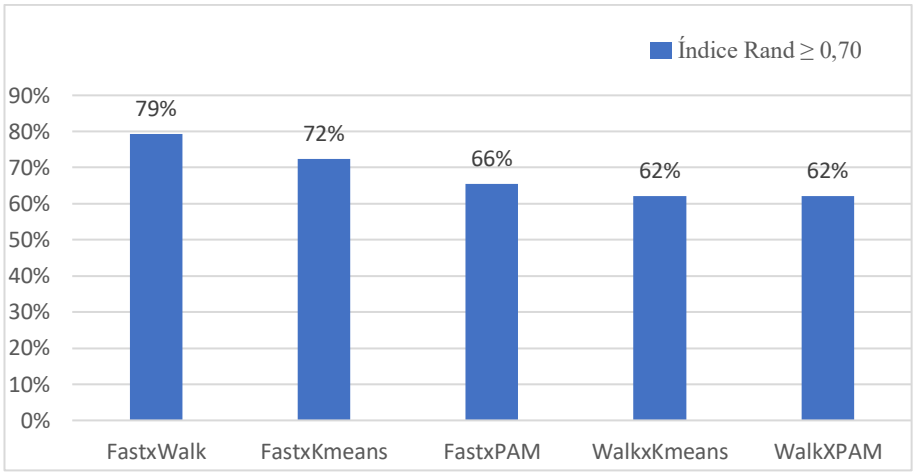


Figura 21: Percentual das bases de dados para as quais foram produzidos valores de Rand acima a 0,7
Fonte: Os autores

Ainda em relação à Figura 21, pode-se observar que, ao comparar o algoritmo de Detecção de Comunidades, *Fast Greedy*, com os algoritmos de agrupamento *k-means* e *PAM* , resultados melhores são encontrados (72% e 66%, respectivamente) frente ao resultado encontrado pelo algoritmo de Detecção de Comunidades, *Walktrap*, quando comparado aos algoritmos de agrupamento *k-means* e *PAM* (62% e 62%, respectivamente).

VII.2.3 Apresentação dos resultados – Comparação dos algoritmos *PAM* e *CLARA*

Outro experimento realizado foi utilizar o algoritmo *CLARA* no lugar do algoritmo *PAM*, de modo a identificar se a troca destes algoritmos acarretaria alguma mudança significativa nos resultados, mais especificamente, analisar se as soluções produzidas a partir da aplicação do algoritmo *CLARA* seriam melhores quando comparadas às soluções do algoritmo *PAM*, no que concerne aos índices de Silhueta e Rand.

Através da Figura 22, é possível perceber que, a partir dessa troca, não foram observadas grandes alterações nos resultados, no que se refere ao índice de Silhueta média (aplicados às 29 bases de dados). É importante destacar que o número de grupos adotados para esta análise corresponde ao número de comunidades geradas pelo algoritmo *Fast Greedy*, como fora descrito na metodologia. Os números de grupos adotados foram k = 2, 3 e 4. É possível perceber que os resultados dos índices de Silhueta são bem semelhantes tanto para o algoritmo *CLARA* quanto para o algoritmo *PAM*.

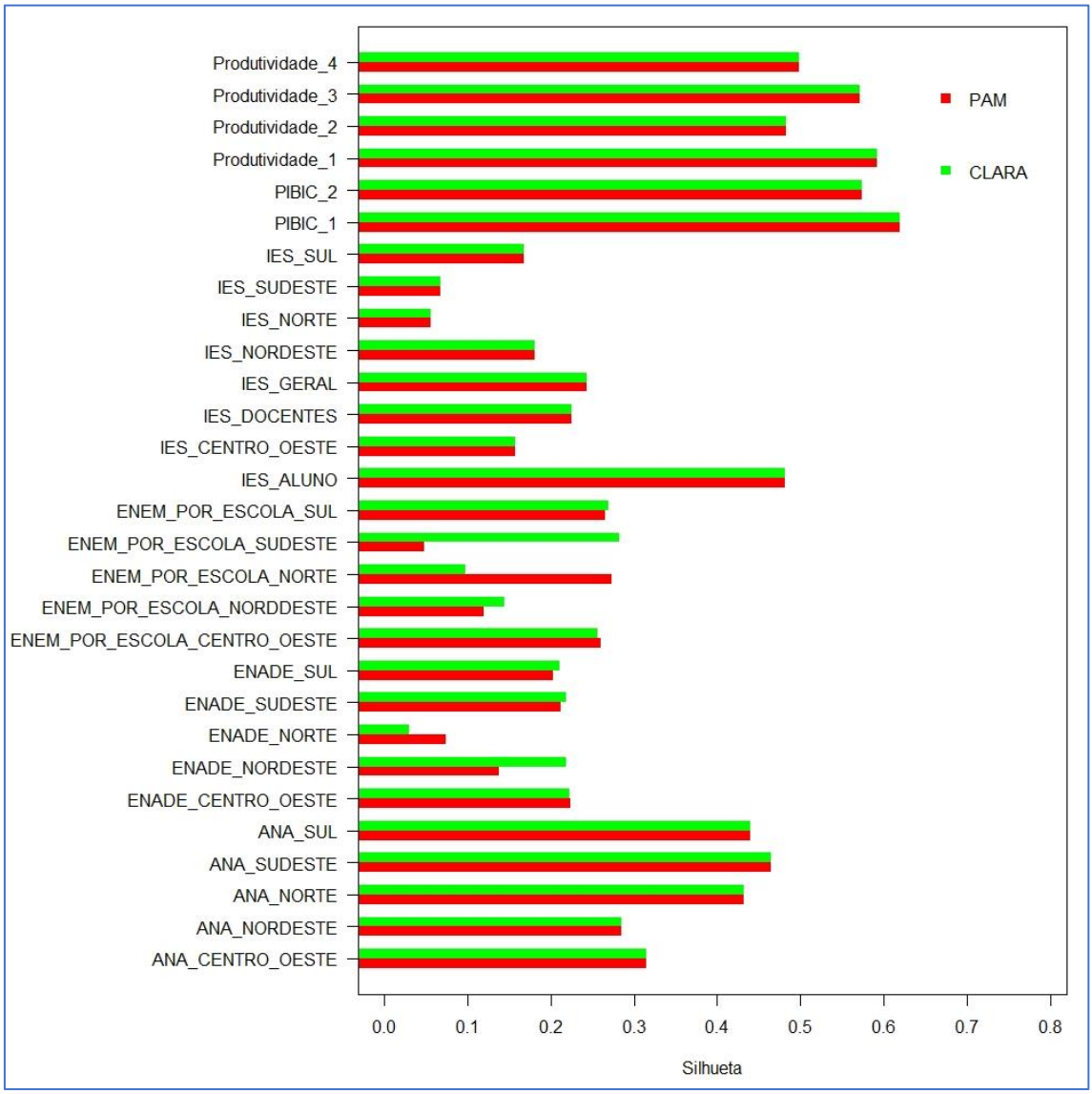


Figura 22:Comparação do algoritmo Clara versus PAM em relação ao índice de Silhueta. Medida de dissimilaridade: Euclidiana e grafo gerado via quartil.
Fonte: Os autores

A Figura 22 apresenta um gráfico de barras que ilustra a comparação dos algoritmos *CLARA*, versus o algoritmo *PAM*, em relação ao índice de Silhueta média, observando que a medida de dissimilaridade utilizada foi a distância Euclidiana. É possível verificar, com mais clareza, os pontos em que os algoritmos obtiveram desempenhos iguais, divergentes ou similares. Em 20 das bases de dados analisadas, os dois algoritmos tiveram resultados similares.

Já nas bases de dados de maior porte, os desempenhos destes algoritmos foram divergentes. Em 5 bases de dados o algoritmo *CLARA* produziu resultados melhores do

que o algoritmo *PAM* e em 4 bases de dados o algoritmo *PAM* teve melhor desempenho do que o algoritmo *CLARA* quanto ao índice de Silhueta.

A Figura 23 ilustra a comparação dos algoritmos *CLARA* versus o algoritmo *PAM*, em relação ao índice de Silhueta média. Para esta análise é utilizada a distância de Manhattan, como medida de dissimilaridade e, a fim de verificar se ao utilizar esta medida resultaria em algum ganho em relação à qualidade dos agrupamentos gerados pelos algoritmos.

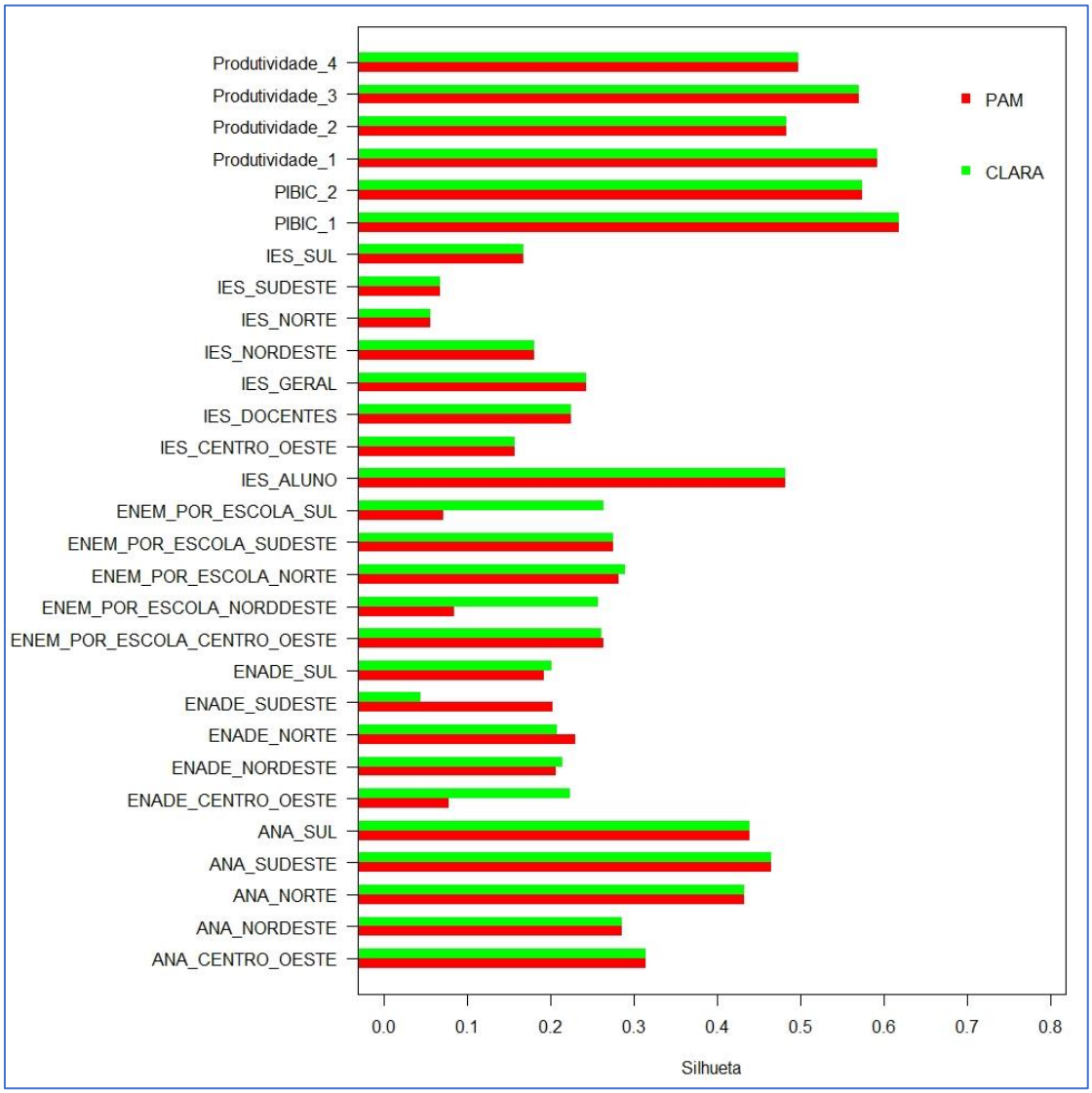


Figura 23:Comparação do algoritmo PAM versus o algoritmo CLARA utilizando a distância de Manhattan.
Fonte: Os autores.

Da Figura 23, depreende-se que, em 20 bases de dados, os algoritmos *PAM* e *CLARA* apresentaram resultados iguais para o índice de Silhueta. Já em relação às 9 bases de dados restantes, observa-se que em 3 bases de dados o algoritmo *PAM* obteve o valor do índice de Silhueta maior do que o algoritmo *CLARA*, e em 6 bases de dados o algoritmo *CLARA* obteve o valor do índice de Silhueta maior do que o *PAM*.

Desta forma, percebe-se que para as bases de dados estudadas, quando utilizada a medida dissimilaridade, distância de Manhattan, os resultados para ambos os algoritmos: *CLARA* e *PAM*, não apresentam grandes alterações. Este fato indica que pode-se usar um algoritmo em detrimento ao outro sem grandes alterações no que tange à qualidade dos agrupamentos.

A Figura 24 apresenta a relação entre o valor do índice de Silhueta e o número de grupos *k*, produzidos pelos algoritmos. Para esta análise considerou-se a distância Euclidiana como medida de dissimilaridade e os grafos foram gerados através do quartil das distâncias dos objetos tomados dois a dois.

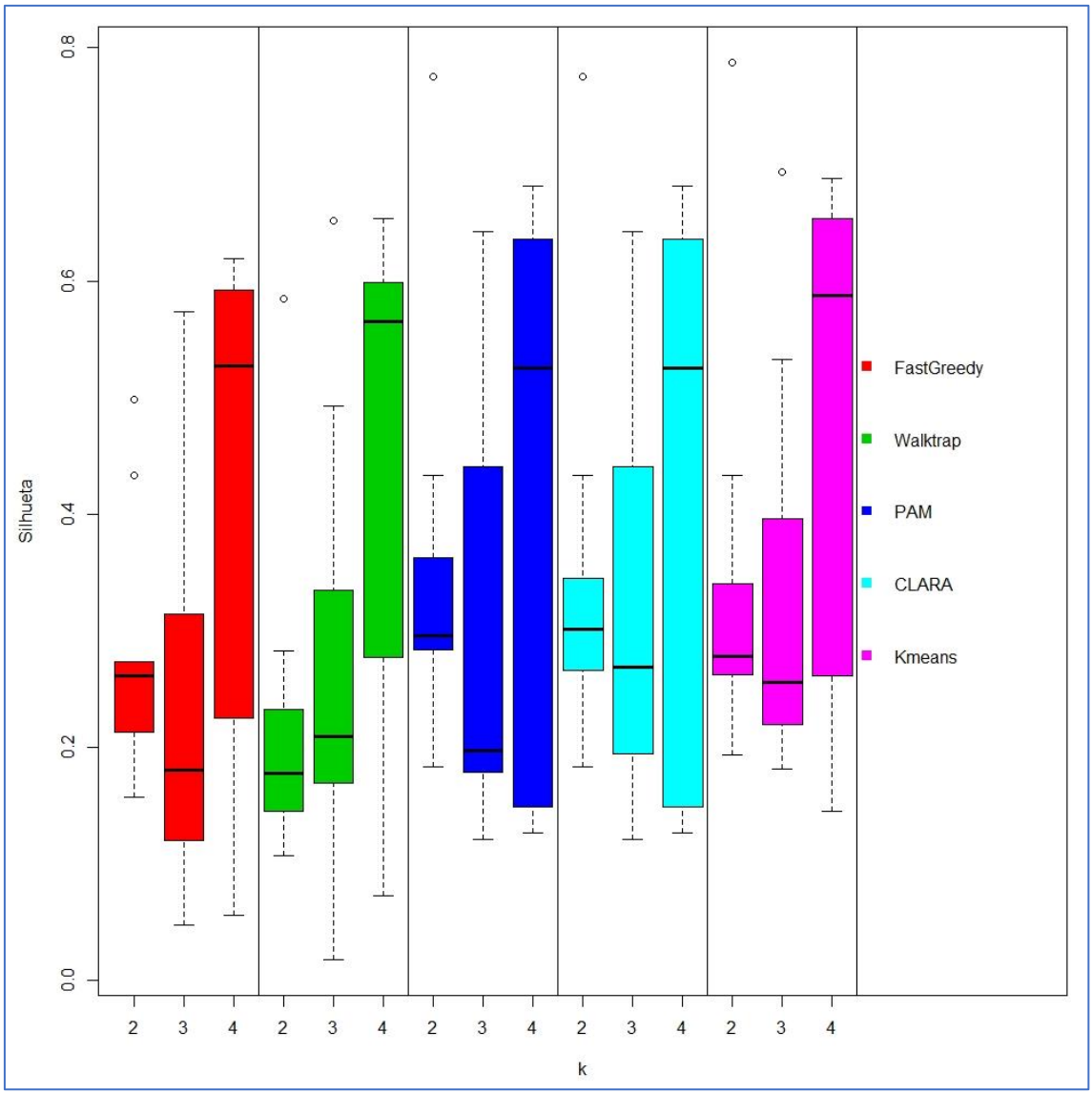


Figura 24:Valores do índice de Silhueta segundo valores dos grupos obtidos.
Fonte: Os autores.

Após definir o número de comunidades, *k*, através da aplicação do algoritmo *Fast Greedy*, este número foi utilizado como parâmetro de entrada para a aplicação dos algoritmos de agrupamento *k-means*, *PAM* e *CLARA*. Como pode ser visto na Figura 25, os valores de *k* (números de grupos) produzidos para os algoritmos são *k* =2, *k* =3, *k* = 4. Pode-se perceber que os melhores resultados quanto ao índice de Silhueta estão associados a 4 grupos.

Já a Figura 25, apresenta o gráfico *boxplot* construído pela relação do valor do índice de Silhueta com o número de grupos *k*, produzidos pelos algoritmos. Mas, para esta análise, considerou-se a distância Euclidiana como medida de dissimilaridade e os grafos foram gerados através da AGM.

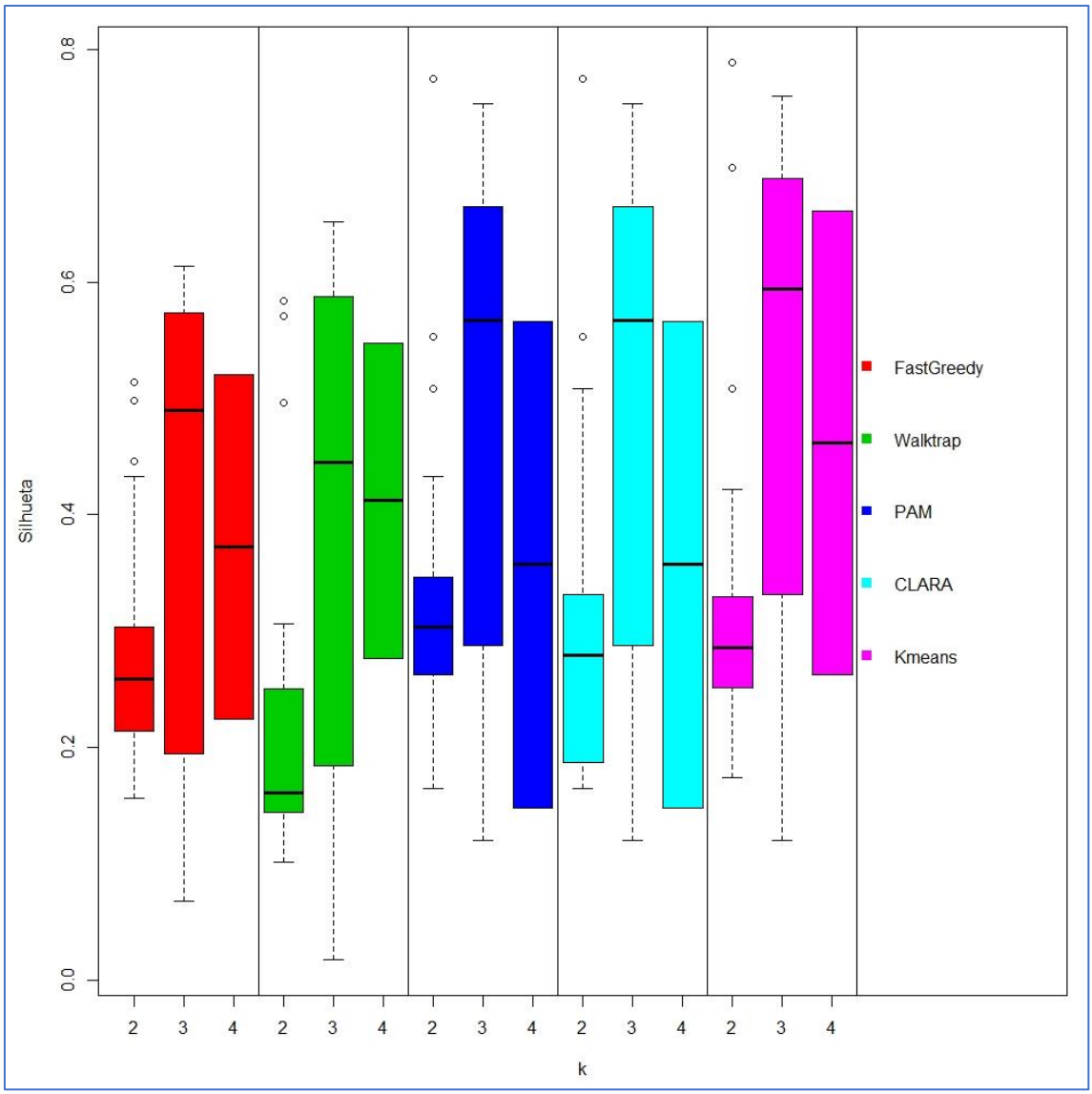


Figura 25:Valores do índice de Silhueta segundo valores dos grupos obtidos.
Fonte: Os autores.

Ao utilizar o índice de Rand, para comparar os resultados do algoritmo *PAM* e o algoritmo *CLARA* frente aos algoritmos de Detecção de Comunidades apresenta-se a Figura 26.

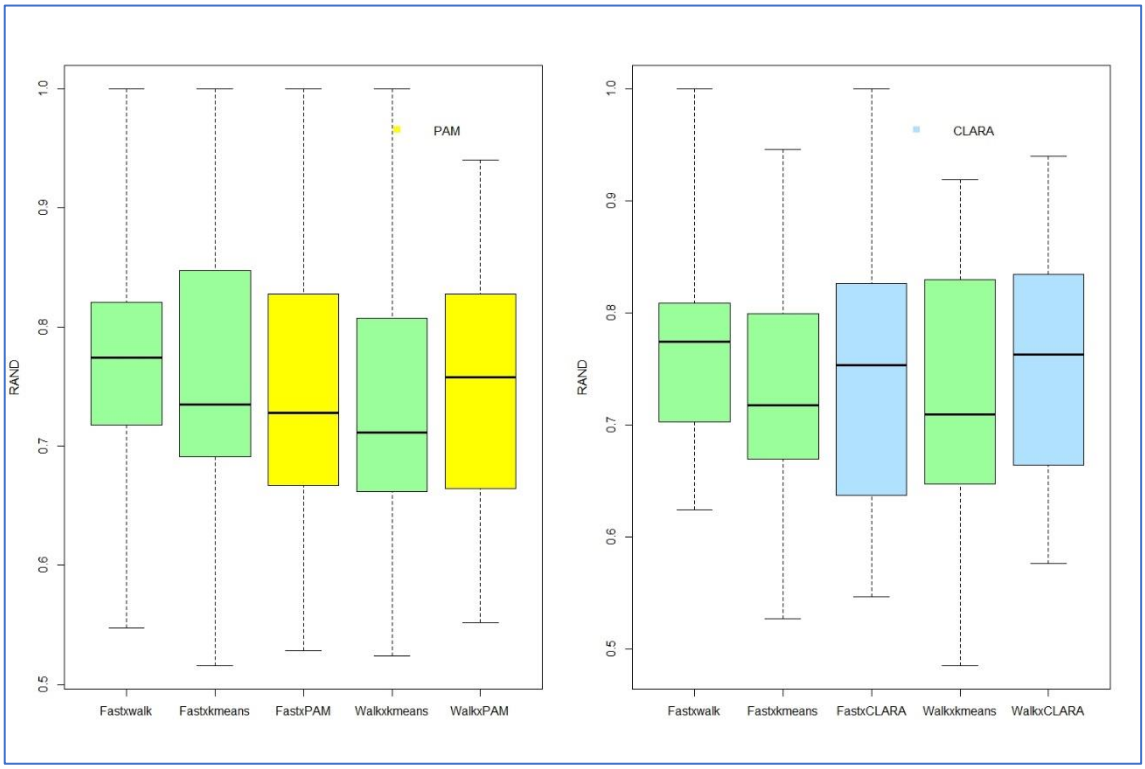


Figura 26:Comparação dos algoritmos *PAM* e *CLARA* em relação índice de Rand.
Fonte: Os autores.

Para este experimento foi utilizada a distância Euclidiana e para geração dos grafos para os algoritmos de Detecção de Comunidades utilizou-se o quartil dos objetos tomados dois a dois. Através da Figura 26 é possível observar que, quando comparado o algoritmo de Detecção de Comunidades *Fast Greedy* com os algoritmos *PAM* e *CLARA*, observa-se que o desempenho do algoritmo *CLARA* frente ao *Fast Greedy*, é melhor em relação ao desempenho do algoritmo *PAM* frente ao *Fast Greedy*. Já em relação à análise dos algoritmos *PAM* e *CLARA* quando comparados ao algoritmo *Walktrap*, não foram observadas diferenças relevantes na troca do algoritmo *PAM* pelo algoritmo *CLARA* frente ao algoritmo de Detecção de Comunidades *Walktrap*.

Em síntese, estes resultados mostram que, para as bases de dados estudadas nesta dissertação, o algoritmo *PAM* e *CLARA* quando aplicados a conjunto de dados de médio a pequeno porte não diferem quanto aos seus resultados, em relação ao índice de Silhueta.

Em relação ao índice de Rand, quando é comparado o desempenho dos algoritmos *PAM* e *CLARA*, frente aos algoritmos de Detecção de Comunidades: *Fast Greedy* e *Walktrap*, verificou-se uma maior similaridade entre os grupos formados pelo algoritmo *CLARA* e as comunidades do algoritmo *Fast Greedy*, do que pelos grupos formados pelo algoritmo *PAM* e as comunidades do algoritmo *Fast Greedy*. No que se refere à comparação das comunidades geradas pelo algoritmo *Walktrap* e os grupos gerados pelos algoritmos *PAM* e *CLARA* não foram observadas diferenças relevantes que evidenciassem ganho de qualidade, na troca dos algoritmos *PAM* e *CLARA*.

VII.2.4 Apresentação dos resultados – Comparação da AGM e Quartis

De modo a avaliar o impacto dos algoritmos de agrupamento e de Detecção de Comunidades quanto à criação de grafos, de maneira distinta, gerados através da AGM e através dos quartis, esta subsecção apresenta as soluções da aplicação dos algoritmos.

Para tal, em um primeiro momento, é realizada a geração dos grafos através do quartil, das distâncias dos objetos tomados dois a dois, para as 29 bases de dados. Em seguida, são aplicados os algoritmos de Detecção de Comunidades e de agrupamento para cada um dos grafos gerados. Posteriormente, são aplicados os índices de Silhueta e de Rand, de modo a analisar a qualidade das comunidades geradas pelos algoritmos de Detecção de Comunidades (*Fast Greedy* e *Walktrap*) e dos agrupamentos gerados pelos algoritmos agrupamento (*k-means*, *PAM* e *CLARA*). Já em um segundo momento, é realizada a geração dos grafos através da árvore geradora mínima, para as 29 bases de dados. Posteriormente, são aplicados os algoritmos de Detecção de Comunidades e de agrupamento para cada um dos grafos gerados através da AGM. E por fim, aplica-se os índices de Silhueta e de Rand.

Os procedimentos, descritos acima, visam analisar se grafos gerados de maneira distinta, a partir de uma mesma base de dados, resultam em algum ganho em relação às soluções produzidas dos algoritmos de agrupamento e Detecção de Comunidades, no que se refere ao índice de Silhueta e de Rand.

A Figura 27 apresenta o gráfico *boxplot* obtido através da distribuição das 29 Silhuetas médias, derivadas da aplicação dos algoritmos: *Fast Greedy*, *Walktrap*, *PAM* e *k-means*, nas 29 bases de dados. Os *boxplots* em verde sumarizam os resultados produzidos pelos algoritmos tendo como dado de entrada os grafos gerados via quartil, já

os *boxplots* em azul sumarizam os resultados produzidos pelos algoritmos a partir dos grafos gerados via AGM.

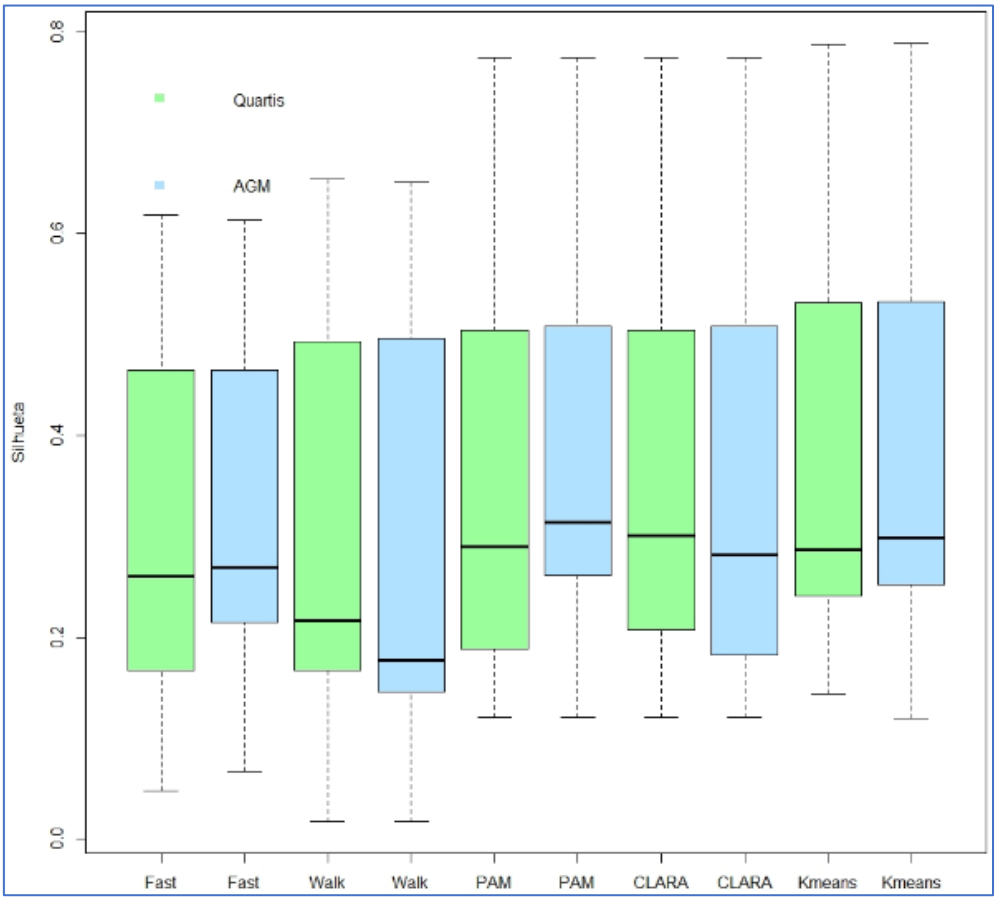


Figura 27: Comparação dos algoritmos através do índice de Silhueta
com criação dos grafos via quartis e AGM.
Fonte: Os autores

No que se refere ao algoritmo de Detecção de Comunidade *Fast Greedy* foi observada uma maior dispersão dos resultados quando o grafo é construído através dos quartis**,** quando comparado aos mesmos resultados produzidos através da AGM.

Para o algoritmo *Walktrap* houve ganhos, mesmo que pequenos, quanto à qualidade do algoritmo em relação ao índice de Silhueta quando aplicado a grafos gerados via quartil em detrimento a grafos gerados via AGM.

Em relação aos algoritmos de Agrupamento: o algoritmo *PAM* apresentou melhores soluções, no que se refere ao valor da mediana para AGM. No algoritmo *CLARA*, as melhores soluções apresentadas foram através dos quartis. E por fim, o algoritmo *k-means* apresentou melhores soluções através da AGM.

Já no que se refere à qualidade dos resultados dos algoritmos de Detecção de Comunidades e de agrupamento, quanto às soluções produzidas em relação ao índice de Rand, quando aplicado a grafos gerados via quartil e grafos gerados através da AGM, destaca-se que para os algoritmos *Fast Greedy* e *Walktrap* quando comparados às suas comunidades, em relação ao índice Rand, observa-se que quando o grafo é construído através dos quartis houve uma maior dispersão do conjunto de dados do que, em relação, aos grafos gerados através da AGM.

Mais especificamente, ao comparar o algoritmo *Fast Greedy* com o algoritmo *k-means*, observa-se uma maior dispersão do conjunto de dados, para grafos gerados através do quartil. Resultado oposto é visto quanto à comparação do algoritmo *Fast Greedy* em relação ao algoritmo de agrupamento *PAM*, sendo observada uma maior dispersão do conjunto de dados quando esse algoritmo é aplicado a grafos gerados através da AGM, e sendo observado maiores valores para o índice de Rand em comparação a grafos gerados através do quartil.

VII.2.5 Apresentação dos resultados – Comparação dos tempos computacionais

A presente subseção traz uma comparação dos algoritmos quanto aos seus tempos de computacionais. Para estes experimentos computacionais foram utilizados grafos gerados através do quartil das distâncias dos objetos tomados dois a dois e utilizando como medida de dissimilaridade as distâncias Euclidiana e de Manhattan. A Tabela 12 apresenta os tempos computacionais de execução, em segundos, obtidos pelos algoritmos para cada base de dados. Para estes cálculos é utilizada a distância Euclidiana, como medida de dissimilaridade.

Tabela 12: Tempos computacionais dos algoritmos utilizando: quartil e AGM e a distância Euclidiana

Tempos Computacionais de Execução dos Algoritmos										
Base de Dados	**Utilizando o quartil para gerar o grafo**					**Utilizando a AGM para gerar o grafo**				
	Fast Greedy	***Walktrap***	***PAM***	***CLARA***	***k-means***	***Fast Greedy***	***Walktrap***	***PAM***	***CLARA***	***k-means***
ANA_CENTRO _OESTE	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,01	0,00	0,00
ANA_NORDESTE	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ANA_NORTE	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ANA_SUDESTE	0,00	0,00	0,00	0,02	**0,02**	0,02	0,00	0,00	0,00	0,00
ANA_SUL	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ENADE_CENTRO _OESTE	**2,48**	4,11	6,33	6,32	0,00	2,22	4,11	6,17	6,21	0,17
ENADE_NORDESTE	2,20	4,02	**6,57**	6,17	0,02	2,21	4,00	6,09	6,26	0,00
ENADE_NORTE	2,25	4,12	6,25	6,31	0,00	2,17	4,14	6,16	6,28	0,00
ENADE_SUDESTE	2,15	3,91	6,03	6,17	0,00	2,18	3,97	6,07	6,17	0,00
ENADE_SUL	2,30	4,13	6,25	6,33	0,00	**2,37**	4,17	6,11	6,12	**0,30**
ENEM_POR_ESCOLA _CENTRO_OESTE	2,22	4,25	6,34	**6,39**	0,00	2,06	**4,33**	**6,56**	6,52	0,00
ENEM_POR_ESCOLA _NORDDESTE	2,09	4,24	6,06	6,21	0,00	2,06	4,23	6,38	**6,53**	0,01
ENEM_POR_ESCOLA _NORTE	2,08	**4,30**	6,25	6,17	0,01	2,16	4,23	6,05	6,12	0,00
ENEM_POR_ESCOLA _SUDESTE	0,66	1,74	2,17	2,22	0,00	0,46	1,65	2,10	2,20	0,00
ENEM_POR_ESCOLA _SUL	0,68	1,75	2,12	2,26	0,00	0,48	1,55	2,14	2,20	0,00
IES_ALUNO	0,02	0,01	0,07	0,06	0,00	0,00	0,03	0,01	0,01	0,00
IES_CENTRO_OESTE	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
IES_DOCENTES	0,00	0,06	0,03	0,03	0,00	0,02	0,03	0,03	0,03	0,00
IES_GERAL	0,01	0,05	0,03	0,03	0,00	0,00	0,04	0,02	0,01	0,00
IES_NORDESTE	0,00	0,00	0,00	0,00	0,02	0,00	0,00	0,00	0,00	0,00
IES_NORTE	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
IES_SUDESTE	0,00	0,00	0,01	0,02	0,00	0,00	0,00	0,00	0,00	0,00
IES_SUL	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
PIBIC_1	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
PIBIC_2	0,02	0,06	0,03	0,03	0,00	0,01	0,07	0,03	0,03	0,00
Produtividade_1	0,00	0,01	0,04	0,00	0,00	0,00	0,02	0,02	0,02	0,00
Produtividade_2	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Produtividade_3	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Produtividade_4	0,05	0,17	0,09	0,08	0,00	0,03	0,16	0,09	0,07	0,00

Fonte: Os autores

Na Tabela 12 os valores destacados em negrito representam os maiores valores de tempo de processamento observados para cada algoritmo. Observa-se que os tempos computacionais, para os grafos gerados através do quartil, ficaram entre 0 e 6,57 segundos. Já os tempos computacionais de execução dos algoritmos gerados a partir da AGM, das distâncias dos objetos tomados dois a dois e, utiliza como medida de dissimilaridade a distância Euclidiana, ficaram entre 0 e 6,56 segundos.

A Tabela 13 apresenta os tempos computacionais de execução, em segundos, obtidos pelos algoritmos para cada base de dados, e utiliza a distância de Manhattan, como medida de dissimilaridade. Os tempos computacionais de execução dos algoritmos ficaram entre 0 e 6,33 segundos, em relação aos grafos gerados através dos quartis. Já para os grafos gerados através da AGM os tempos computacionais ficaram entre 0 e 6,56 segundos.

No que tange aos tempos computacionais de execução dos algoritmos de agrupamento e de Detecção de Comunidades, no geral, o algoritmo *k-means* apresentou melhores resultados frente aos algoritmos de Detecção de Comunidade e aos *PAM* e *CLARA* (algoritmos de agrupamento).

Tabela 13: Tempos computacionais dos algoritmos utilizando: Quartil e AGM e a distância de Manhattan

Tempos Computacionais de Execução dos Algoritmos										
Base de Dados	**Utilizando o quartil para gerar o grafo**					**Utilizando a AGM para gerar o grafo**				
	Fast Greedy	***Walktrap***	***PAM***	***CLARA***	***k-means***	***Fast Greedy***	***Walktrap***	***PAM***	***CLARA***	***k-means***
ANA CENTRO OESTE	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ANA_NORDESTE	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,00	0,00	0,00
ANA NORTE	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,00	0,00	0,00
ANA_SUDESTE	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ANA_SUL	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,00	0,00	0,00
ENADE CENTRO OESTE	2,56	4,06	6,22	**6,31**	0,00	**2,37**	4,14	6,11	6,14	0,13
ENADE NORDESTE	2,22	4,00	6,23	6,21	0,00	2,35	4,00	6,01	6,16	0,00
ENADE_NORTE	2,19	4,06	**6,33**	6,27	**0,01**	2,22	4,23	6,05	6,06	0,00
ENADE_SUDESTE	2,29	3,96	6,07	6,25	0,00	2,14	3,98	6,08	6,08	0,00
ENADE SUL	**2,60**	4,05	6,08	6,16	0,00	2,25	4,32	6,10	6,31	**0,26**
ENEM_POR ESCOLA CENTRO_OESTE	2,22	4,25	6,26	6,20	0,00	2,13	4,39	**6,56**	**6,41**	0,02
ENEM POR ESCOLA NORDDESTE	2,13	4,19	6,26	6,20	0,00	2,15	4,33	6,21	6,21	0,01
ENEM POR ESCOLA NORTE	2,00	**4,43**	5,92	6,21	0,00	2,03	**4,41**	6,11	5,93	0,00
ENEM POR ESCOLA SUDESTE	0,52	1,75	2,11	2,25	0,00	0,40	1,66	2,16	2,10	0,00
ENEM POR ESCOLA SUL	0,52	1,67	2,12	2,13	0,00	0,46	1,57	2,18	2,19	0,00
IES_ALUNO	0,02	0,03	0,05	0,07	0,00	0,00	0,03	0,03	0,02	0,00
IES CENTRO OESTE	0,00	0,00	0,00	0,02	0,00	0,00	0,00	0,00	0,00	0,00
IES_DOCENTES	0,02	0,03	0,01	0,03	0,00	0,00	0,05	0,03	0,03	0,00
IES_GERAL	0,02	0,04	0,02	0,04	0,00	0,02	0,03	0,01	0,03	0,00
IES_NORDESTE	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
IES NORTE	0,00	0,00	0,00	0,00	0,00	0,00	0,01	0,00	0,00	0,00
IES_SUDESTE	0,00	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
IES_SUL	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,00
PIBIC_1	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
PIBIC 2	0,01	0,05	0,03	0,04	0,00	0,01	0,07	0,03	0,05	0,00
Produtividade_1	0,02	0,01	0,02	0,01	0,00	0,00	0,02	0,00	0,00	0,02
Produtividade_2	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Produtividade 3	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Produtividade_4	0,03	0,16	0,09	0,09	0,00	0,01	0,17	0,08	0,19	0,00

VII.3 Conclusões

Nesta dissertação foram realizados experimentos em 29 bases de dados ligadas à educação, visando realizar um estudo computacional comparativo entre três algoritmos de agrupamento (*k-means*, *PAM* e *CLARA*) e dois algoritmos de Detecção de Comunidades (*Fast Greedy* e *Walktrap*) . A comparação entre os algoritmos, quanto às estruturas de grupos e comunidades formados, foi feita através do cálculo de índices, Silhueta e Rand, associados aos critérios de validação.

Mais especificamente, os objetivos dos experimentos propostos para esta dissertação estiveram pautados em: construir redes através das bases de dados reais de duas maneiras distintas (através dos quartis das distâncias dos objetos e da AGM); segmentar as bases de dados a partir da aplicação dos algoritmos de agrupamento (*k-means*, *CLARA* e *PAM*) e de Detecção de Comunidades (*Fast Greedy* e *Walktrap*); avaliar as soluções produzidas pelos cinco algoritmos, tendo por base o cálculo de índices associados aos critérios de validação relativo e externo, sejam eles: Silhueta e índice de Rand; comparar e avaliar os resultados produzidos pelos algoritmos de agrupamento e de Detecção de Comunidades; verificar se a troca do algoritmo *PAM* pelo *CLARA* acarretaria alguma mudança significativa de qualidade, no que tange aos índices de Silhueta e Rand e por fim, avaliar o impacto dos algoritmos de agrupamento e de Detecção de Comunidades quanto à criação de redes geradas através da AGM e através dos quartis das distâncias dos objetos.

Quando a análise se concentrou em verificar se há diferença na troca dos algoritmos *PAM* e *CLARA*, percebe-se que, para as bases de dados estudadas, quando utilizado a medida dissimilaridade, distância de Manhattan, os resultados os algoritmos *CLARA* e *PAM* não apresentam grandes alterações.

Em síntese, os resultados mostram que, para as bases de dados estudadas nesta dissertação, o algoritmo *PAM* e *CLARA* quando aplicados a conjunto de dados de porte pequeno e médio não diferem quanto aos seus resultados, em relação ao índice de Silhueta. Em relação ao índice de Rand, quando é comparado o desempenho dos algoritmos *PAM* e *CLARA*, frente aos algoritmos de Detecção de Comunidades: *Fast Greedy* e *Walktrap*, apresentasse maior similaridade entre os grupos formados pelo algoritmo *CLARA* e as comunidades do algoritmo *Fast Greedy*, do que os grupos

formados pelo algoritmo *PAM* e as comunidades do algoritmo *Fast Greedy*. No que se refere à comparação das comunidades geradas pelo algoritmo *Walktrap* e os grupos gerados pelos algoritmos *PAM* e *CLARA*, não foram observadas diferenças relevantes que evidenciassem ganho de qualidade na troca desses dois algoritmos.

Quando a análise foi realizada através da relação do valor do índice de Silhueta em relação ao número de grupos *k*, produzidos pelos algoritmos, observou-se que maiores valores de *k* resultaram em melhores valores para o índice de Silhueta, na aplicação dos algoritmos quando é utilizado a distância Euclidiana como medida de dissimilaridade e quando os grafos foram gerados através do quartil das distâncias dos objetos tomados *dois* a dois. Quando é utilizada a mesma medida de dissimilaridade, mas o grafo foi gerado através da AGM, observou-se melhores resultados concentrados em grupos com $k = 3$.

Para trabalhos futuros será considerada a aplicação do algoritmo de agrupamento *k-means* com o *k* variando, em um intervalo, de 2 a 5 e aplicar os algoritmos de Detecção de Comunidade: *Fast Greedy* e *Walktrap*, posteriormente a esta etapa e a partir de dos resultados gerados comparar o número de comunidades geradas pelo algoritmo de agrupamento *k-means* e os algoritmos de Detecção de Comunidades, em específico verificar se coincide o número de agrupamento e o número de comunidades gerados. Recomenda-se, ainda, utilizar outro índice de validação a exemplo, o índice de Dunn.

REFERÊNCIAS

ALVES, I.; OLIVEIRA, C. S.; BRITO, J. A. M. Um estudo do problema de detecção de comunidades em redes. **Revista Eletrônica Sistemas & Gestão**, v. 9 n. 4, pp. 566-576, 2014.

AMARAL, W. S. **Análise de Grafos para Apoio em Auditoria de Licitações Públicas**. 2020. Dissertação (Mestrado em Ciência da Computação) - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, Rio de Janeiro, 2020.

BONDY, J. A.; MURTY U. S. R. Graph Theory with Applicatons. Department of Combination and Optimization, University of Waterloo, Ontario, Canada, 1976.

BOTELHO, G. M.; SOUSA, F. B. Comparação de Algoritmos de Detecção de Comunidades em Redes Complexas. Instituto de Ciências Matemáticas e de Computação São Carlos - SP – Brasil. Universidade de São Paulo (USP). Disponível em: http://wiki.icmc.usp.br/images/8/81/Trabalho2GlendaFabiano.pdf. Acessado em: 25/08/2015.

BRITO, J.A., SEMAAN, G.S., BRITO, L.R. Resolução do Problema dos K-Medoids Via Algoritmo Genético de Chaves Aleatórias Viciadas. **Revista Pesquisa Naval**, vol. 31, n. 27, p.126-142, 2015.

CAMILO, C. O.; SILVA, J. C. Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas. Goiás: INF/UFG, 2009. 18p. (Relatório Técnico, RT-INF-001-09).

CLAUSET, A., NEWMAN, M. E., MOORE, C. “Finding community structure in very large networks”, **Physical Review E.**, vol. 70, n. 6, 2004.

COSTA, J. M. S., TEIXEIRA, P. M., BARBASTEFANO, R. G., SOUZA, C. G., LIMA, L. S. Aplicação da análise de redes sociais em uma rede de publicações sobre gestão da

cadeia de suprimentos. XXXIII Encontro Nacional de Engenharia de Produção, Bahia, 2013.

CRUZ, M. D. **O problema de Clusterização Automática**. 2010. Tese (Doutorado em Engenharia de Sistemas e Computação) – Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2010.

DIAS, M. V.; BRAZA, P. A.; SILVA, E. B.; GOLDSCHMIDTA, R. R. Detecção de comunidades baseada em informações contextuais em redes homogêneas com atributos. **Revista Militar de Ciência e Tecnologia**, Rio de Janeiro, V.36, n. 2, p. 14–22, 2019.

DONI, M. V. Análise de Cluster: Métodos Hierárquicos e de Particionamento. Monografia. Universidade Presbiteriana Mackenzie. São Paulo, 2004.

FAYYAD, U. M.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From Data Mining to Knowledge Discovery in Databases. **Artificial Intelligence**, 1996.

FRÄNTI, P.; SIERANOJA, S. K-means properties on six clustering benchmark datasets. **Applied Intelligence**, v. 48, n. 12, p. 4743-4759, 2018.

FREI, F. Introdução à análise de agrupamentos – Teoria e prática, Editora: Unesp, 2006.

FREITAS, L. Q. **Medidas de Centralidade em Grafos**. Dissertação (Mestrado em Engeharia de Produção) – Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2010.

FIDUCCIA, C., MATTHEYSES, R. M. (1982), “A linear-time heuristic for improving network partitions”. In: 19th IEE Conference on Design Automation.

FILHO, J. A. A. **Definição Automática da Quantidade de Atributos Selecionados em Tarefas de Agrupamento de Dados**. 2013. Tese (Doutorado em Ciências) - Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo. São Carlos, 2013.

GOLDBARG, M.; GOLDBARG, E. Grafos: Conceitos, Algoritmos e Aplicações, Rio de Janeiro, Editora Elsevier, 2012.

HAMMOUDA, K. M. **Web Mining: Identifying Document Structure for Web Document Clustering.** 2002. Dissertation (Master of Applied Science in Systems Design Engineering) - Department of Systems Design Engineering. University of Waterloo, Canada, 2002.

HAN, J., KAMBER, M., PEI, J. Data Mining: Concepts and Techniques: Concepts and Techniques. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2012.

HANSEN, P.; JAUMARD B. Cluster Analysis and Mathematical Programming, Mathematical Programming, 1997.

JAIN, A., DUBES, R. Algorithms for Clustering Data. Prentice Hall. 1988.

JAIN, Anil K. Data clustering: 50 years beyond K-means. Pattern recognition letters, v. 31, n. 8, p. 651-666, 2010.

KAUAI, F; PÉCORA, J. E. JR.; LIMA, L. S. Desenvolvimento de uma metaheurística híbrida para o problema de agrupamento de dados. In: ConBREpro, 2020, Curitiba. X Congresso Brasileiro de Engenharia de Produção, 2020.

KAUFMAN, L.; ROUSSEEUW, P. J. An introduction to cluster analysis. John Wiley and Sons. 1990.

KAUFMAN, L.; ROUSSEEUW P.J. Finding Groups in Data – An Introduction to Cluster Analysis. Wiley-Interscience Publication. 1989.

KUMAR, V.; STEINBACH, M.; TAN, P. N. Introdução ao Data Mining – Mineração de Dados. Ciência Moderna. 2009.

LAROSE, D. T. Discovering Knowledge in Data: An Introduction to Data Mining. Wiley-Interscience. 2004.

LINARES, O. A. C. **Segmentação de Imagens de alta Dimensão por meio de Algortimos de Detecção de Comunidades e Super Pixels**. 2013. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional) - Instituto de Ciência Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2013.

LIU, C. L. Introduction to Combinatirial Mathematics (Computer Science Sereies). Mcgraw-Hill College. 1968.

MAGALHÃES, C.V.C; SANTOS, R.E.S.; SILVA, F.Q.B.; GOMES, A.S. Caracterizando a Pesquisa em Informática na Educação no Brasil: Um Mapeamento Sistemático das Publicações do SBIE. XXIV Simpósio Brasileiro de Informática na Educação - SBIE, 2013.

MARTINS, R. M. **Utilização de técnicas de análise de agrupamento do risco de geada no estado do paraná para a cultura do milho safrinha**. 2008. Tese (Doutorado em Agronomia). Faculdade de Ciências Agronômicas da Universidade Estadual Paulista. Botucatu, 2008.

MENTZ, J.; CALVO, R.; SENO, E. R. M.; ROMERO, R. A. F.; LIANG, Z. Redes Complexas: Conceitos e aplicações. Instituto de Ciências Matemáticas e de Computação. Relatório Técnico, n. 290, 2007.

MOORI, R. G.; MARCONDES, R. C.; AVILA, R. T. A análise de agrupamentos como instrumento de apoio à melhoria da qualidade dos serviços aos clientes. Rev. adm. contemp. [online]. vol.6, n.1, pp. 63-84, ISSN 1982-7849, 2002.

NALDI, M. C. **Técnicas de combinação para agrupamento centralizado e distribuído de dados**. 2011. Tese (Doutorado em Ciências de Computação e Matemática Computacional), Universidade de São Paulo, 2011.

NEWMAN, M. E. J. Girvan, M. Finding and evaluating community structure in networks. **Physical Review E**, 69 (026113), 2004.

OLIVEIRA, T. B. S. **Clusterização de dados utilizando técnicas de redes complexas e computação bioinspirada**. 2008. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional) -Instituo de Ciências Matemática e de Computação – Universidade de São Paulo. São Carlos, 2008.

PONS, P.; LATAPY, M. Computing communities in large networks using random walks. **Journal of Graph Algorithms Applications**, v. 10, n. 2, p. 191–218, 2006.

PORTO, S. M. **Metodologia para a Evolução de Comunidades em Redes Complexas Dinâmicas**. 2014. Dissertação (Mestrado em Computação Aplicada). Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2014.

R Core Team (2019). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.URL https://www.R-project.org/.

ROUSSEEUW, P. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. **Journal of Computational and Applied Mathematics**. Elsevier Science Publishers B. V., Amsterdam, The Netherlands, v. 20, n. 1, p. 53–65, nov. 1987.

SANTOS, M.; GUIZZO, C. S. P.; SAMPAIO, R. R.; CORDEIRO, A. J. A. Aplicação da análise de agrupamento na avaliação da formação técnica profissionalizante para o setor industrial. XVLI Simpósio brasileiro de pesquisa operacional, Bahia, 2014.

SEMAAN, G. S. **Algoritmos para o Problema de Agrupamento Automático**. 2013. Tese (Doutorado em Computação) - Universidade Federal Fluminense, Niterói, 2013.

SEMAAN, G. S.; CORREA, D. A. ; RIBAS, S. G. ; SILVA, E. F. ; BRITO, J. A. M. Redes de Colaboração e Contribuições da Renote. Renote. **Revista Novas Tecnologias na Educação**, v. 18, p. 1-10, 2020.

SILVA, L. F. **Uma Análise Híbrida para Detecção de Anomalias da Mama Usando Séries Temporais de Temperatura**. 2015. Tese (Doutorado em Computação). Universidade Federal Fluminense, Niterói, 2015.

SILVA, Leandro. A. **Mineração de dados: uma abordagem introdutória e ilustrada**. ISBN: 978-85-8293-059-5. 1. Ed. São Paulo: Editora Mackenzie, 2015.

SILVA, L. A.; PERES, S.M.; BOSCARIOLI, C. **Introdução à Mineração de Dados com Aplicação em R**. ISBN: 978-85-352-8446-1 - 1ed. - Rio de Janeiro: Elsevier, 2016.

SOUSA, R. N.; PRATA, D. N. Resumo Automático de Textos Jurídicos usando Grafos com Vocabulário Controlado e Algoritmo K-Means com Words Embedding. **Revista ESMAT**, ano 11, n. 18, p. 65-80, Edição Especial 2019.

TEIXEIRA, L. S.; LIMA, L. S.; ABREU, N. M. M. Grafos que modelam redes confiáveis. XL SBPO: A Pesquisa Operacional e o uso racional de recursos hídricos. João Pessoa, PB, Brasil, 2008.

VALE, M. N. **Agrupamentos de dados: avaliação de métodos e desenvolvimento de aplicativo para análise de grupos**. 2005. Dissertação (Mestrado em Engenharia Elétrica). Pontífica Universidade Católica. Rio de Janeiro, 2005.

WANG, P.; XU, B.; WU, Y.; ZHOU, X. Link prediction in social networks: the state-of-the-art. SCIENCE CHINA Information Sciences, 2015.

Anexo I

O presente anexo apresenta o Quadro 5 com os valores para o índice de Rand para as 29 bases de dados, assim como o valor de *k*. Para este experimento foi utilizada como medida de dissimilaridade a distância Euclidiana e a geração dos grafos via quartil.

Quadro 5: Comparação dos Algoritmos através do índice

Base de Dados	k	FastxWalk	FastxKmeans	FastxPAM	WalkxKmeans	WalkXPAM
ANA_CENTRO_OESTE.csv	3	0,77419	0,70107	0,75698	0,66451	**0,93978**
ANA_NORDESTE.csv	3	0,79089	0,54865	0,82774	0,64016	0,83478
ANA_NORTE.csv	2	0,77966	**1,00000**	**1,00000**	0,77966	0,77966
ANA_SUDESTE.csv	3	**1,00000**	0,70968	0,90322	0,70967	0,90322
ANA_SUL.csv	3	0,92333	0,76000	0,76000	0,78333	0,78333
ENADE_CENTRO_OESTE.csv	2	0,83876	0,67254	0,56293	0,64573	0,60680
ENADE_NORDESTE.csv	3	0,54729	0,71795	0,71389	0,71142	0,70749
ENADE_NORTE.csv	3	0,64587	0,71534	0,70686	0,66566	0,66786
ENADE_SUDESTE.csv	2	0,62021	0,63379	0,54101	0,66503	0,55941
ENADE_SUL.csv	2	0,72261	0,55667	0,52832	0,63582	0,58492
ENEM_POR_ESCOLA _CENTRO_OESTE.csv	2	0,70700	0,93424	0,63587	0,73470	0,55216
ENEM_POR_ESCOLA _NORDDESTE.csv	3	0,82044	0,78348	0,71641	0,92401	0,78983
ENEM_POR_ESCOLA_ NORTE.csv	2	0,63364	0,90490	0,96461	0,64405	0,63463
ENEM_POR_ESCOLA _SUDESTE.csv	3	0,68451	0,71389	0,67132	0,61822	0,74013
ENEM_POR_ESCOLA _SUL.csv	2	0,73602	0,85602	0,73366	0,85750	0,93030
IES_ALUNO.csv	16	0,80808	0,80196	0,82605	0,80742	0,78341
IES_CENTRO_OESTE.csv	2	0,71795	0,71794	0,84615	0,53846	0,84615
IES_DOCENTES.csv	4	0,72453	0,74322	0,67485	0,70223	0,58650
IES_GERAL.csv	3	0,84048	0,79227	0,72790	0,86238	0,81011
IES_NORDESTE.csv	3	0,90336	0,84751	0,66666	0,83244	0,67819
IES_NORTE.csv	4	0,80882	0,69118	0,72058	0,66176	0,67647
IES_SUDESTE.csv	3	0,67247	0,62891	0,61504	0,72396	0,66415
IES_SUL.csv	3	0,92877	0,51567	0,06296	0,52421	0,66096
PIBIC_1.csv	4	0,93230	0,93230	0,81538	**1,00000**	0,82769
PIBIC_2.csv	3	0,79710	0,77283	0,89408	0,91906	0,88263
Produtividade_1.csv	4	0,79001	0,94591	0,94832	0,83685	0,83419
Produtividade_2.csv	4	0,74703	0,73518	0,75494	0,76679	0,76284
Produtividade_3.csv	4	0,74000	0,93000	0,87000	0,79000	0,75666
Produtividade_4.csv	2	0,73102	0,59906	0,63757	0,70458	0,75751

Anexo II

O presente anexo apresenta o Quadro 6 com a comparação dos tempos computacionais dos algoritmos, em segundos, para as 29 bases de dados, utilizando como medida de dissimilaridade a distância Euclidiana e a geração dos grafos via quartil.

Quadro 6: Comparação dos tempos Computacionais de Execução dos Algoritmos

Tempos Computacionais de Execução dos Algoritmos					
Base de Dados	***Fast Greedy***	***Walktrap***	***PAM***	***CLARA***	***k-means***
ANA_CENTRO_OESTE.csv	0,02	0,00	0,00	0,00	0,00
ANA_NORDESTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_NORTE.csv	0,02	0,00	0,00	0,00	0,00
ANA_SUDESTE.csv	0,00	0,00	0,00	0,02	**0,02**
ANA_SUL.csv	0,00	0,00	0,00	0,00	0,00
ENADE_CENTRO_OESTE.csv	**2,48**	4,11	6,33	6,32	0,00
ENADE_NORDESTE.csv	2,20	4,02	**6,57**	6,17	0,02
ENADE_NORTE.csv	2,25	4,12	6,25	6,31	0,00
ENADE_SUDESTE.csv	2,15	3,91	6,03	6,17	0,00
ENADE_SUL.csv	2,30	4,13	6,25	6,33	0,00
ENEM_POR_ESCOLA_CENTRO_OESTE.csv	2,22	4,25	6,34	**6,39**	0,00
ENEM_POR_ESCOLA_NORDDESTE.csv	2,09	4,24	6,06	6,21	0,00
ENEM_POR_ESCOLA_NORTE.csv	2,08	**4,30**	6,25	6,17	0,01
ENEM_POR_ESCOLA_SUDESTE.csv	0,66	1,74	2,17	2,22	0,00
ENEM_POR_ESCOLA_SUL.csv	0,68	1,75	2,12	2,26	0,00
IES_ALUNO.csv	0,02	0,01	0,07	0,06	0,00
IES_CENTRO_OESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_DOCENTES.csv	0,00	0,06	0,03	0,03	0,00
IES_GERAL.csv	0,01	0,05	0,03	0,03	0,00
IES_NORDESTE.csv	0,00	0,00	0,00	0,00	0,02
IES_NORTE.csv	0,00	0,00	0,00	0,00	0,00
IES_SUDESTE.csv	0,00	0,00	0,01	0,02	0,00
IES_SUL.csv	0,00	0,00	0,00	0,00	0,00
PIBIC_1.csv	0,00	0,00	0,00	0,00	0,00
PIBIC_2.csv	0,02	0,06	0,03	0,03	0,00
Produtividade_1.csv	0,00	0,01	0,04	0,00	0,00
Produtividade_2.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_3.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_4.csv	0,05	0,17	0,09	0,08	0,00

Anexo III

O presente anexo apresenta o Quadro 7 com a comparação dos tempos computacionais dos algoritmos, em segundos, para as 29 bases de dados, utilizando como medida de dissimilaridade a distância Manhattan e a geração dos grafos via quartil.

Quadro 7: Comparação dos tempos Computacionais de Execução dos Algoritmos

Tempos Computacionais de Execução dos Algoritmos					
Base de Dados	***Fast Greedy***	***Walktrap***	***PAM***	***CLARA***	***k-means***
ANA_CENTRO_OESTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_NORDESTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_NORTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_SUDESTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_SUL.csv	0,00	0,00	0,00	0,01	0,00
ENADE_CENTRO_OESTE.csv	2,56	4,06	6,22	**6,31**	0,00
ENADE_NORDESTE.csv	2,22	4,00	6,23	6,21	0,00
ENADE_NORTE.csv	2,19	4,06	**6,33**	6,27	**0,01**
ENADE_SUDESTE.csv	2,29	3,96	6,07	6,25	0,00
ENADE_SUL.csv	**2,60**	4,05	6,08	6,16	0,00
ENEM_POR_ESCOLA_CENTRO_OESTE.csv	2,22	4,25	6,26	6,20	0,00
ENEM_POR_ESCOLA_NORDDESTE.csv	2,13	4,19	6,26	6,20	0,00
ENEM_POR_ESCOLA_NORTE.csv	2,00	**4,43**	5,92	6,21	0,00
ENEM_POR_ESCOLA_SUDESTE.csv	0,52	1,75	2,11	2,25	0,00
ENEM_POR_ESCOLA_SUL.csv	0,52	1,67	2,12	2,13	0,00
IES_ALUNO.csv	0,02	0,03	0,05	0,07	0,00
IES_CENTRO_OESTE.csv	0,00	0,00	0,00	0,02	0,00
IES_DOCENTES.csv	0,02	0,03	0,01	0,03	0,00
IES_GERAL.csv	0,02	0,04	0,02	0,04	0,00
IES_NORDESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_NORTE.csv	0,00	0,00	0,00	0,00	0,00
IES_SUDESTE.csv	0,00	0,02	0,00	0,00	0,00
IES_SUL.csv	0,00	0,00	0,00	0,00	0,00
PIBIC_1.csv	0,00	0,00	0,00	0,00	0,00
PIBIC_2.csv	0,01	0,05	0,03	0,04	0,00
Produtividade_1.csv	0,02	0,01	0,02	0,01	0,00
Produtividade_2.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_3.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_4.csv	0,03	0,16	0,09	0,09	0,00

Fonte: Os autores.

Anexo III

O presente anexo apresenta o Quadro 8 com a comparação dos tempos computacionais dos algoritmos, em segundos, para as 29 bases de dados, utilizando como medida de dissimilaridade a distância Euclidiana e a geração dos grafos via AGM.

Quadro 8: Comparação dos tempos Computacionais de Execução dos Algoritmos

Tempos Computacionais de Execução dos Algoritmos					
Base de Dados	***Fast Greedy***	***Walktrap***	***PAM***	***CLARA***	***k-means***
ANA_CENTRO_OESTE.csv	0,00	0,00	0,01	0,00	0,00
ANA_NORDESTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_NORTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_SUDESTE.csv	0,02	0,00	0,00	0,00	0,00
ANA_SUL.csv	0,00	0,00	0,00	0,00	0,00
ENADE_CENTRO_OESTE.csv	2,22	4,11	6,17	6,21	0,17
ENADE_NORDESTE.csv	2,21	4,00	6,09	6,26	0,00
ENADE_NORTE.csv	2,17	4,14	6,16	6,28	0,00
ENADE_SUDESTE.csv	2,18	3,97	6,07	6,17	0,00
ENADE_SUL.csv	**2,37**	4,17	6,11	6,12	**0,30**
ENEM_POR_ESCOLA_CENTRO_OESTE.csv	2,06	**4,33**	**6,56**	6,52	0,00
ENEM_POR_ESCOLA_NORDDESTE.csv	2,06	4,23	6,38	**6,53**	0,01
ENEM_POR_ESCOLA_NORTE.csv	2,16	4,23	6,05	6,12	0,00
ENEM_POR_ESCOLA_SUDESTE.csv	0,46	1,65	2,10	2,20	0,00
ENEM_POR_ESCOLA_SUL.csv	0,48	1,55	2,14	2,20	0,00
IES_ALUNO.csv	0,00	0,03	0,01	0,01	0,00
IES_CENTRO_OESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_DOCENTES.csv	0,02	0,03	0,03	0,03	0,00
IES_GERAL.csv	0,00	0,04	0,02	0,01	0,00
IES_NORDESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_NORTE.csv	0,00	0,00	0,00	0,00	0,00
IES_SUDESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_SUL.csv	0,00	0,00	0,00	0,00	0,00
PIBIC_1.csv	0,00	0,00	0,00	0,00	0,00
PIBIC_2.csv	0,01	0,07	0,03	0,03	0,00
Produtividade_1.csv	0,00	0,02	0,02	0,02	0,00
Produtividade_2.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_3.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_4.csv	0,03	0,16	0,09	0,07	0,00

Fonte: Os autores.

Anexo IV

O presente anexo apresenta o Quadro 9 com a comparação dos tempos computacionais dos algoritmos, em segundos, para as 29 bases de dados, utilizando como medida de dissimilaridade a distância Manhattan e a geração dos grafos via AGM**.**

Quadro 9: Comparação dos tempos Computacionais de Execução dos Algoritmos

Tempos Computacionais de Execução dos Algoritmos					
Base de Dados	***Fast Greedy***	***Walktrap***	***PAM***	***CLARA***	***k-means***
ANA_CENTRO_OESTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_NORDESTE.csv	0,00	0,02	0,00	0,00	0,00
ANA_NORTE.csv	0,00	0,02	0,00	0,00	0,00
ANA_SUDESTE.csv	0,00	0,00	0,00	0,00	0,00
ANA_SUL.csv	0,00	0,00	0,00	0,00	0,00
ENADE_CENTRO_OESTE.csv	**2,37**	4,14	6,11	6,14	0,13
ENADE_NORDESTE.csv	2,35	4,00	6,01	6,16	0,00
ENADE_NORTE.csv	2,22	4,23	6,05	6,06	0,00
ENADE_SUDESTE.csv	2,14	3,98	6,08	6,08	0,00
ENADE_SUL.csv	2,25	4,32	6,10	6,31	**0,26**
ENEM_POR_ESCOLA_CENTRO_OESTE.csv	2,13	4,39	**6,56**	**6,41**	0,02
ENEM_POR_ESCOLA_NORDDESTE.csv	2,15	4,33	6,21	6,21	0,01
ENEM_POR_ESCOLA_NORTE.csv	2,03	**4,41**	6,11	5,93	0,00
ENEM_POR_ESCOLA_SUDESTE.csv	0,40	1,66	2,16	2,10	0,00
ENEM_POR_ESCOLA_SUL.csv	0,46	1,57	2,18	2,19	0,00
IES_ALUNO.csv	0,00	0,03	0,03	0,02	0,00
IES_CENTRO_OESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_DOCENTES.csv	0,00	0,05	0,03	0,03	0,00
IES_GERAL.csv	0,02	0,03	0,01	0,03	0,00
IES_NORDESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_NORTE.csv	0,00	0,01	0,00	0,00	0,00
IES_SUDESTE.csv	0,00	0,00	0,00	0,00	0,00
IES_SUL.csv	0,00	0,00	0,00	0,02	0,00
PIBIC_1.csv	0,00	0,00	0,00	0,00	0,00
PIBIC_2.csv	0,01	0,07	0,03	0,05	0,00
Produtividade_1.csv	0,00	0,02	0,00	0,00	0,02
Produtividade_2.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_3.csv	0,00	0,00	0,00	0,00	0,00
Produtividade_4.csv	0,01	0,17	0,08	0,19	0,00

Fonte: Os autores.