UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MARCELO PEREIRA VIEIRA

UMA HEURÍSTICA GULOSA PARA O PROBLEMA DE CLASSIFICAÇÃO GEODÉSICA

QUIXADÁ

2019

MARCELO PEREIRA VIEIRA

UMA HEURÍSTICA GULOSA PARA O PROBLEMA DE CLASSIFICAÇÃO GEODÉSICA

Monografia apresentada no curso de Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Ciência da Computação. Área de concentração: Computação.

Orientador: Prof. Me. Paulo Henrique Macêdo de Araújo

QUIXADÁ

2019

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

V716h Vieira, Marcelo Pereira.
Uma heurística gulosa para o problema de classificação geodésica / Marcelo Pereira Vieira. – 2019.
39 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2019.
Orientação: Prof. Me. Paulo Henrique Macêdo de Araújo.

1. Geodésicas - Classificação . 2. Outlier. 3. Teoria dos grafos. I. Título.

CDD 004

MARCELO PEREIRA VIEIRA

UMA HEURÍSTICA GULOSA PARA O PROBLEMA DE CLASSIFICAÇÃO GEODÉSICA

Monografia apresentada no curso de Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Ciência da Computação. Área de concentração: Computação.

Aprovada em: ___/___/___

BANCA EXAMINADORA

Prof. Me. Paulo Henrique Macêdo de Araújo (Orientador)
Universidade Federal do Ceará – UFC

Prof. Dr. Críston Pereira de Souza
Universidade Federal do Ceará - UFC

Prof. Me. Arthur Rodrigues Araruna
Universidade Federal do Ceará - UFC

Dedico este trabalho a quatro pessoas que me acompanharam e foram vitais em todas as fases da minha vida, do nascimento até hoje. Aos meus pais Vilani da Silva Pereira e Manoel Aderaldo Vieira Júnior e aos meus avós paternos Maria Amélia Facundo Vieira e Manoel Aderaldo Vieira *(in memorian)*.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que me deu o dom da vida e me abençoa todos os dias com o seu amor infinito.

À minha mãe Vilani da Silva Pereira a mulher mais forte que eu conheço, e ao meu pai, Manoel Aderaldo Vieira Junior meu exemplo de firmeza. Vocês são minha base as pessoas mais importantes da minha vida, eu amo vocês.

À minha namorada Flavia Souza Reis companheira, amiga que me deu suporte durante essa fase da minha vida.

Agradeço também a todos os meus amigos que fiz em Quixadá: Ronildo Oliveira, Leonardo Almeida, Matheus Rios, Veyber Lucas, Arthur Antunes, Marcello Alexandre, João Pedro, Flávio Yuri, Alex Sandro. Amigos que me deram força durante momentos difíceis e companheiros adversidade e a todos os outros não citados.

Quero agradecer ao meu professor orientador Paulo Henrique Macêdo Araújo, pelo empenho dedicado ao meu projeto de pesquisa. Também agradeço a todos os professores da Universidade Federal do Ceará que contribuíram com minha formação acadêmica.

“O que sabemos é uma gota; o que ignoramos é um oceano.”

(Isaac Newton)

RESUMO

Este trabalho propõe uma heurística gulosa com objetivo de gerar uma boa solução viável para o problema de Classificação Geodésica (*CG*). Esse problema foi introduzido em (ARAÚJO et al., 2019), onde os autores também apresentaram um modelo de programação linear inteira para resolvê-lo. O problema consiste em classificar informações, de acordo com dados pré-estabelecidos e representados por um grafo de similaridades. Para isso, é realizada a identificação de padrões definidos por convexidade geodésica. Neste trabalho, abordamos uma versão do problema CG com múltiplos grupos com o objetivo de identificar padrões de convexidade geodésica em grafos com maior acurácia. Analisamos a performance da heurística proposta utilizando experimentos computacionais tanto com instâncias realistas como aleatórias.

Palavras-chave: Geodésicas - Classificação, Outlier, Teoria dos grafos.

ABSTRACT

This work proposes a greedy heuristic with the goal of generating a good feasible solution for the Geodesic Classification (*GC*) problem. This problem was introduced in (ARAÚJO et al., 2019), where the authors also presented an integer linear programming model to solve it. The GC problem consists to classify new data, according to pre-established data represented by a graph of similarities. To do so, we must identify patterns defined by geodesic convexity. In this work, we deal with the multi-groups version of the *GC* problem in order to identify geodesic convexity patterns on graphs with better accuracy. We analyze the performance of the proposed heuristic using computational experiments considering realistic and random instances.

Keywords: Geodesics - Classification, Outlier, Graph theory.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CG	Convexidade Geodésica
SVM	*Support Vector Machine*
CRIO	*Classification and regression via integer optimization*

SUMÁRIO

1 INTRODUÇÃO

Diariamente, pessoas no mundo inteiro geram uma grande massa de dados, entre elas, pesquisadores trabalhando em diferentes áreas de estudo e conhecimento ou simples usuários de internet que geram um grande volume de informações sobre seus costumes, interesses e desinteresses.

Essas informações podem ser utilizadas para derivar outros dados escondidos entre tantos. Para melhor utilizar essa grande quantidade de informações, catalogam-se e classificam-se os dados em grupos considerados análogos em algum contexto a fim de prever similaridades entre amostras de informações desconhecidas.

Esse procedimento constitui a chamada *aprendizagem supervisionada* (ou *supervised learning*), que vem sendo cada vez mais utilizada para solucionar alguns dos grandes problemas da área de tecnologia da informação em *big data*.

O procedimento de *aprendizagem supervisionada* se caracteriza por duas etapas: a primeira delas consiste em classificar um conjunto de amostras inicial de tal modo a agrupar amostras de classes que possuam certas semelhanças; a segunda, em determinar a classe em que uma amostra fora do conjunto de partida melhor se encaixaria. Da primeira etapa, emerge o que chamamos de *problema de classificação*. Neste trabalho, consideramos apenas o problema de classificação com apenas duas classes. Um método clássico para separar conjuntos de amostras definidas como pontos em um espaço multi-dimensional é o chamado *Support Vector Machine* (*SVM*) (CORTES; VAPNIK, 1995). Tal método se baseia em tentar dividir o espaço de amostras com um hiperplano, separando as duas classes em dois subespaços, cada um correspondendo a uma dessas classes. Desse modo, qualquer ponto fora do conjunto de amostras inicial tem sua classe prevista de acordo com o subespaço a que ele pertence.

Em Araújo et al. (2019) foi definido um problema de classificação em grafos análogo ao problema de classificação no espaço multidimensional (*versão Euclidiana)*, o qual chamou-se de *Geodesic Classification Problem* ou *Problema de Classificação Geodésica* (*CG*), em português. Essa abordagem foi baseada no método de resolução *SVM* para classificação e suas adaptações em Bertsimas e Shioda (2007) e Corrêa, Donne e Marenco (2018). O problema *CG* é definido fazendo a partir do uso da convexidade geodésica em grafos para agrupar amostras (vértices do grafo) similares. Neste caso, um par de vértices que possuem aresta entre si é considerado similar para o contexto da aplicação; assim, é possível expressar uma relação binária entre as amostras.

O objeto de estudo deste trabalho é um método heurístico para obtenção de uma boa solução viável para o problema *CG*, cujos casos atendam às seguintes hipóteses:

- *Existe um padrão subjacente ao conjunto de amostras que pode ser expresso por uma noção de convexidade*. Trata-se de um tema relativamente recente de investigação na Teoria dos Grafos que surgiu como uma analogia ao conceito de convexidade Euclidiana. Embora existam diferentes noções de convexidade em grafos, o foco deste trabalho está no conceito de convexidade geodésica, definido através de caminhos mínimos no grafo (a analogia com a convexidade Euclidiana refere-se à distância entre pontos) (PELAYO, 2013). Nesse caso, as amostras não precisam ser caracterizadas por valores numéricos e o problema de classificação torna-se puramente combinatório. No entanto, a inspiração para este trabalho são métodos de resolução para o caso da classificação através da convexidade Euclidiana (que é o caso de quase todos os trabalhos encontrados na literatura conhecida). No caso Euclidiano, assumimos que as amostras são expressas por vetores numéricos em um espaço multidimensional e o padrão está associado à localização dessas amostras. O estudo de um problema de classificação em grafos usando convexidade geodésica tem duas motivações principais. A princípio, é de interesse teórico, uma vez que se refere à possibilidade de estabelecer novos problemas em grafos que possam trazer subsídios para métodos de resolução, inclusive para aqueles baseados na convexidade Euclidiana. A segunda motivação tem um propósito mais prático e consiste em permitir que o padrão seja expresso através de alguma relação binária sobre o conjunto de amostras.
- *O conjunto de dados pode possuir pontos discrepantes, também chamados de outliers, não previamente identificados*. A origem da discrepância pode ter diferentes razões, incluindo possíveis erros de amostragem ou características do fenômeno em estudo. Do ponto de vista da modelagem matemática, os pontos discrepantes são aqueles que se desviam do padrão subjacente do conjunto de amostras de sua classe. A presença de tais pontos é um desafio adicional. Eles precisam ser identificados e removidos do processo de determinação do padrão das amostras. O processo de eliminação de pontos discrepantes pode levar à eliminação de alguns pontos não discrepantes. Apesar disso, o uso de um mecanismo para eliminar pontos discrepantes é relevante, tendo em vista que a perda de informação que o erro na exclusão errada de alguns pontos pode causar à precisão na classificação, é pequena quando comparada ao erro induzido por um ponto discrepante qualquer.

Considerando as hipóteses acima, o problema de classificação pode ser formulado

como um problema de otimização matemática que deseja separar o espaço de amostras em conjuntos convexos e, simultaneamente, associar esses conjuntos a classes. Essa associação caracteriza a classificação obtida. A maneira mais fácil de classificar utilizando essa abordagem é chamada de separação linear, que é feita, por exemplo, no método do *SVM*. Portanto, o problema CG faz uma analogia desse método com o contexto de grafos.

Aplicações possíveis são facilmente encontradas nos campos de mineração de dados e estatísticas clássicas, como, por exemplo: classificação de texto do *Twitter*, detecção de comunidades em redes sociais, predição de similaridade de arquivos históricos, conteúdo de recomendação na *Netflix* e filtragem de *spam* para *e-mails*. Na classificação de texto no *Twitter*, por exemplo, utiliza-se ferramentas de mineração de texto que sejam capazes de identificar padrões e classificar mensagens, como para a análise sentimental, recomendação de preferências, recomendação de amigos e outros (HONG; DAVISON, 2010).

O restante deste trabalho está organizado da seguinte maneira: a Seção 2 apresenta alguns trabalhos relacionados ao problema *CG*; a Seção 3 traz notações e definições para grafos e o problema *CG*; na Seção 4, os procedimentos metodológicos do trabalho; a Seção 5 mostra os resultados dos testes aplicando a heurística proposta ao problema de *CG*; finalmente, algumas conclusões obtidas são apresentadas na Seção 6.

2 TRABALHOS RELACIONADOS

Neste capítulo, serão apresentados alguns trabalhos relacionados que contribuíram o que estão relacionado tematicamente ao presente trabalho.

Em Bertsimas e Shioda (2007), os autores propõem métodos de otimização baseados em programação inteira mista para os problemas estatísticos clássicos de classificação e regressão no espaço Euclidiano. Eles apresentam o pacote de *software* chamado *CRIO* (*Classification and regression via integer optimization*), que demonstra um potencial de impacto significativo nos métodos de resolução de problemas de otimização nas áreas de estatística e mineração de dados.

Em Corrêa, Donne e Marenco (2018), os autores exploraram os aspectos combinatórios do problema de classificação com convexidade Euclidiana usando formulações de programação inteira mista baseadas no pacote CRIO.

Em Araújo et al. (2019), os autores, inspirados na versão do problema de classificação baseado nos conceitos de convexidade euclidiana discutidos em (CORRÊA et al., 2019), definem o problema de Classificação Geodésica e apresentam um modelo de programação linear inteira para o mesmo.

Este trabalho propõe uma heurística gulosa para o problema de Classificação Geodésica, que pode ser utilizada para gerar soluções iniciais, essas, por sua vez, usadas em meta-heurísticas.

Quadro 1 – Tabela demonstrando as semelhanças entre os trabalhos.

Semelhanças	Trabalhos			
	(BERTSIMAS; SHIODA, 2007)	(CORRÊA; DONNE; MARENCO, 2018)	(ARAÚJO et al., 2019)	Trabalho proposto
Classificação Euclidiana	Sim	Sim	Sim	Não
Classificação Geodésica	Não	Não	Sim	Sim
Programação Inteira/Mista	Sim	Sim	Não	Não

Fonte: Elaborada pelo Autor

3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, apresentamos os conceitos utilizados ao longo deste trabalho. Na Seção 3.1 são apresentados alguns conceitos e notações para grafos, enquanto na Seção 3.2 o problema de Classificação Geodésica é definido.

3.1 Conceitos sobre grafos

Um grafo não-direcionado G é um par ordenado $G = (V,E)$, composto por um conjunto finito $V = \{1,2,...,n\}$, cujos elementos são chamados de *vértices*, e por um conjunto de pares não ordenados $E \subseteq \{\{u,v\} \mid u,v \in V, u \neq v\}$, cujos elementos são chamados de *arestas*. Denotamos por $n = |V|$ e $m = |E|$ a cardinalidade de cada um desses conjuntos. Dois vértices $u,v \in V$ são adjacentes (ou vizinhos) se $\{u,v\} \in E$, e não adjacentes caso contrário. Duas arestas $e,f \in E$ são adjacentes se $e \cap f \neq \emptyset$, e não adjacentes caso contrário. O grau de um vértice $v \in V$, denotado por $deg(v)$, corresponde ao número de vizinhos de v em G, isto é $|\{u \mid \{v,u\} \in E\}|$. $\Delta(G)$ denota o grau do vértice de maior grau em G e $\delta(G)$ o correspondente em grau mínimo. A densidade de G, denotada por $d(G)$, é dada por $d(G) = \frac{2m}{n(n-1)}$.

Um *caminho* em um grafo G é uma sequência de vértices distintos $P =< v_1, v_2, \ldots, v_l >$, tal que $\{v_i, v_{i+1}\} \in E(G)$ para $i = 1, \ldots, l-1$. Nesse caso, demonstramos $V(P) = \{v_1, \ldots, v_l\}$ e $E(P) = \{\{v_i, v_{i+1}\} : i = 1, \ldots, l-1\}$. Dizemos que P é um caminho entre v_1 e v_l. O *tamanho* de P é igual a $|E(P)| = |V(P)| - 1$. P é um caminho mínimo se seu tamanho possui o menor valor dentre todos os caminhos entre v_1 e v_l.

Um grafo G é considerado *conexo* se existe um caminho em G entre qualquer par de vértices. Segundo Artigas et al. (2011), a analogia entre o conceito de conjunto convexo em matemática contínua e discreta é feita considerando-se o conjunto de vértices de um grafo conexo e a distância entre dois vértices (número de arestas em um caminho mínimo entre eles) como um espaço métrico. Um subconjunto de vértices $S \subseteq V(G)$ é considerado convexo se contiver os vértices de todos os caminhos mínimos que conectam qualquer par de vértices em S. Outras definições de convexidade foram estudadas considerando-se diferentes tipos de caminho, como caminhos sem cordas (FARBER; JAMISON, 1986) ou caminhos triangulares (CHANGAT; MATHEW, 1999).

Alguns dos primeiros artigos que generalizaram os conceitos Euclidianos de conjuntos convexos à teoria dos grafos foram Harary e Nieminen (1981), **??**), Farber e Jamison

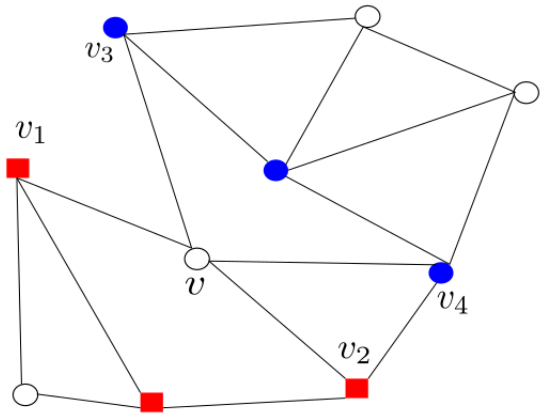
(1986). A convexidade em grafos também foi estudada sob diferentes aspectos, como conjuntos geodésicos, fecho e números de convexidade Cáceres et al. (2006), Dourado et al. (2009), Dourado:2010:RGN:2646627.2647159.

O intervalo fechado $I[v,w]$ é o conjunto de todos os vértices que pertencem a um geodésico entre v e w, dado um conjunto S, $I[S] = \bigcup_{u,v \in S} I[u,v]$. Se $I[S] = S$, então S é um *conjunto convexo*. O *fecho convexo* de S, representado por $H[S]$, é o menor conjunto convexo que contém S. Se $H[S] = V$, então S é um *conjunto de fecho*.

3.2 Problema de Classificação Geodésica

O problema de classificação geodésica para duas classes foi definido em Araújo et al. (2019). Nesse problema, são dados dois subconjuntos não-vazios $V_B, V_R \subseteq V$, $V_B \cap V_R = \emptyset$, de modo $V_{BR} = V_B \cup V_R$ representa o conjunto de amostras inicialmente classificadas. Os conjuntos V_B e V_R representam as classes *azul* e *vermelha*, respectivamente. Os vértices restantes são chamados de *vértices não classificados*, em que $V_N = V \setminus V_{BR}$ é o conjunto que contém todos eles. Um exemplo de grafo para o conjunto de amostras é ilustrado na Figura 1. Este grafo é chamado de *grafo de similaridade*.

Figura 1 – Exemplo de grafo de similaridade com: V_R = vértices quadrados; V_B = vértices circulares preenchidos; e V_N = vértices circulares não preenchidos.



Fonte: Araújo et al. (2019).

analogamente à versão Euclidiana do problema de classificação, assumimos que G é *linearmente separável em relação a* V_B *e* V_R se:

- $H[V_B] \cap V_R = \emptyset$,

- $H[V_R] \cap V_B = \emptyset$, e
- $H[V_B] \cap H[V_R] \cap V_N = \emptyset$,

e *linearmente inseparável* caso contrário. Se G é linearmente inseparável em relação a algum conjunto de amostras, podemos torná-lo linearmente separável definindo alguns vértices de V como *outlier*. Neste caso, os vértices considerados outliers não desaparecem do grafo; simplesmente suas classes não são consideradas para a determinação do padrão de convexidade geodésica. Tal situação ocorre no exemplo da Figura 1. Nesse exemplo, G é linearmente inseparável, pois $v \in H[V_B] \cap H[V_R] \cap V_N$, mas ele se torna linearmente separável se v_1, v_2, v_3 ou v_4 for considerado *outlier*. Sendo assim, o problema de classificação associado a G, V_B e V_R deseja encontrar uma classificação, dada por dois conjuntos $A_B \subseteq V_B$ e $A_R \subseteq V_R$ de vértices não-*outliers*, com o menor número de vértices considerados *outliers* (i.e., $|V_{BR} \setminus (A_B \cup A_R)|$ sendo mínimo). Esse problema é formalizado da seguinte forma: dados $V_B, V_R \subseteq V$, desejamos encontrar subconjuntos para cada classe, $A_B \subseteq V_B, A_R \subseteq V_R$ (que definem os vértices que não são outliers em cada uma delas), para que:

- $H[A_B] \cap A_R = \emptyset$,
- $H[A_R] \cap A_B = \emptyset$,
- $H[A_B] \cap H[A_R] \cap (V_N) = \emptyset$, e
- $|V_{BR} \setminus (A_B \cup A_R)|$ seja mínimo.

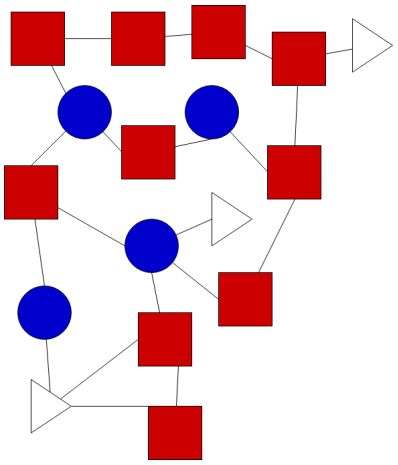
As duas primeiras restrições afiram que se um vértice inicialmente classificado está no fecho convexo da classe oposta, então o mesmo deve ser considerado como outlier, ou seja, não deve pertencer a A_B ou A_R. A terceira restrição diz que não se pode ter um vértice não classificado pertencendo ao fecho convexo das duas classes opostas pois não saberíamos como classificá-lo. Por fim, a última restrição impõe a minimização do número de outliers, a fim de obter-se uma boa acurácia.

É importante perceber que A_B e A_R determinam um mapeamento de V nas classes $\{azul, vermelha\}$ que classifica todos os vértices não classificados em $H[A_B]$ em classe azul e todos os vértices não classificados em $H[A_R]$ em classe vermelha. Vértices não classificados podem não estar no fecho de nenhuma classe. Neste caso, classificamos esses vértices,

aleatoriamente, em $V \setminus (H[A_B] \cup H[A_R])$. Observa-se que os vértices que são considerados *outliers* não são reclassificados (ou seja, movidos para a classe oposta); eles têm apenas suas classes ignoradas no processo de identificação do padrão geodésico.

Devido às restrições citadas acima, há muitos casos em que só será possível encontrar uma solução viável para o problema considerando-se todos os vértices de uma classe como *outliers*, como é ilustrado na Figura 2. Para contornar esses casos e, ainda assim, tentar encontrar um padrão que descreva as amostras, fazemos uso de uma abordagem de divisão das classes em conjuntos convexos. Dessa forma, poderemos ter L_B conjuntos convexos, determinados pelos fechos convexos dos *grupos* $A_k \subseteq V_B$, $\forall k \in \{1, \ldots, L_B\}$ e L_R conjuntos convexos, determinados pelos fechos convexos dos *grupos* $A_{k'} \subseteq V_R$, $\forall k' \in \{L_B + 1, \ldots, L_B + L_R\}$, com intuito de que conjuntos convexos de classes distintas não tenham interseção, exceto *outliers*.

Figura 2 – Exemplo de grafo de similaridade com: V_R = vértices quadrados; V_B = vértices circulares preenchidos; e V_N = vértices circulares não preenchidos.



Fonte: Elaborado pelo autor.

Para definir o problema usando as abordagens de *outliers* e divisão das classes em grupos, sejam C_B e C_R os conjuntos de índices dos grupos da classe azul e vermelha respectivamente, e considere-se $C_{BR} = C_B \cup C_R$, definimos por $K(i)$ a classe do vértice $i \in V_{BR}$ e $\bar{K}(i)$, a classe oposta. Podemos agora definir formalmente o problema de *Classificação Geodésica* ou *CG* da seguinte forma:

Dado um grafo conexo $G = (V, E)$, conjuntos de vértices inicialmente classificados V_B (vértices azuis) e V_R (vértices vermelhos), com $V_N = V \setminus (V_{BR})$, e parâmetros de limites para o número de grupos de cada classe, L_B e L_R, desejamos encontrar grupos $A_k \subseteq V_B$, $\forall k \in C_B$, e

$A_{k'} \subseteq V_R$, $\forall k' \in C_R$, tais que:

1. $A_k \cap A_{k'} = \emptyset$, $k, k' \in C_B$,

2. $A_k \cap A_{k'} = \emptyset$, $k, k' \in C_R$,

3. $H[A_k] \cap A_{k'} = \emptyset$, $k \in C_B$, $k' \in C_R$,

4. $H[A_{k'}] \cap A_k = \emptyset$, $k \in C_B$, $k' \in C_R$,

5. $H[A_k] \cap H[A_{k'}] \cap V_N = \emptyset$, $k \in C_B$, $k' \in C_R$,

6. $|V_{BR}| - |(\bigcup_k A_k \cup \bigcup_{k'} A_{k'})|$ é mínimo (o qual é o número de *outliers*).

Observe-se que as restrições acima são análogas àquelas apresentadas anteriormente e que usava apenas um grupo por classe. Porém, agora se considera as restrições para cada par de grupos de cor oposta.

Um ponto importante a destacar é que poderíamos ter definido o problema sem as restrições 1 e 2 acima: suponha $K \in \{B, R\}$ e $l \in A_k \cap A_{k'}$, $k, k' \in C_K$. Defina $\hat{A}_j = A_j, \forall j \neq k \in C_K$, e $\hat{A}_k = A_k \setminus \{l\}$. Então, $H[\hat{A}_j] \subseteq H[A_j]$ e $\hat{A}_j \subseteq A_j$, $\forall j \in C_K$, o que implica que as restrições 3, 4 e 5 continuam satisfatórias. Além disso, $(\bigcup_{j \neq k} A_j) \cup A_k = (\bigcup_{j \neq k} \hat{A}_j) \cup \hat{A}_k$, mantém o mesmo número de *outliers* e, consequentemente, o mesmo valor de função objetivo. Porém, essas restrições foram adicionadas aqui a fim de reduzir-se a ocorrência de simetrias no conjunto de soluções viáveis e, dessa maneira, diminuir o espaço de busca a ser pesquisado nos métodos de resolução, incluindo a heurística gulosa proposta neste trabalho.

4 DESENVOLVIMENTO DA HEURÍSTICA

Neste capítulo são apresentados e descritos todos os passos da execução deste trabalho.

4.1 Desenvolvimento da heurística

Esta etapa descreverá a heurística gulosa para o problema *CG*.

4.1.1 Variáveis que determinam uma solução

Para fins de implementação dos algoritmos propostos neste trabalho, definimos as seguintes variáveis para representar uma solução do problema:

$$a_{bi} \in \{0,1\}, i \in V_B, b \in C_B, \tag{4.1}$$

$$a_{ri} \in \{0,1\}, i \in V_R, r \in C_R, \tag{4.2}$$

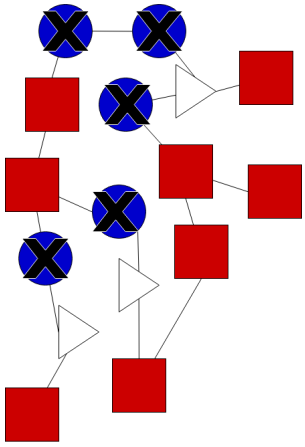
As variáveis na equação (4.1) indicam quais vértices pertencem a cada grupo da classe azul de tal modo que cada grupo A_b, $b \in C_B$ seja representado por $A_b = \{i \in V_B \mid a_{bi} = 1\}$. O equivalente vale para as variáveis na equação (4.2), as quais determinam os grupos da classe vermelha.

Dessa forma, um vértice $i \in V_{BR}$ é *outlier* para uma solução com variáveis a se $a_{ki} = 0$, para todo $k \in C_{K(i)}$.

4.1.2 Solução Trivial

Uma solução viável para o problema CG pode ser obtida trivialmente da seguinte forma: define-se como *outlier* todos os vértices em V_B se $|V_B| < |V_R|$, de tal forma que $A_k = V_R$ para algum $k \in C_R$ e $A_{k'} = \emptyset$ para $k' \in C_{BR}, k' \neq k$. Caso contrário, defina como *outliers* todos os vértices em V_R, de modo que $A_k = V_B$ para algum $k \in C_B$ e $A_{k'} = \emptyset$ para $k' \in C_{BR}, k' \neq k$. Um exemplo para essas duas definições é ilustrado na Figura 3, onde todos os vértices de V_B foram transformados em *outlier*. Com base nisso, temos uma solução viável para o problema *CG* cujo valor de função objetivo é igual a $\min\{|V_B|, |V_R|\}$.

Figura 3 – Exemplo da solução trivial: V_R = vértices quadrados; V_B = vértices circulares preenchidos; e V_N = vértices circulares não preenchidos .



Fonte: Elaborado pelo autor.

4.1.3 Configuração inicial da heurística gulosa

Distribuímos arbitrariamente e uniformemente os vértices inicialmente classificados (em V_{BR}) nos respectivos grupos de suas classes; sendo assim, as Restrições 1 e 2 do problema são satisfeitas. O pseudo-código para a realização desse procedimento é descrito no Apêndice 6.

Nessa primeira abordagem, designamos a cada novo vértice i um grupo para o mesmo através do percurso dos grupos de sua classe em sequência. Fazendo isto, distribuímos cada vértice inicialmente classificado de maneira uniforme entre os grupos de suas classes.

Entretanto, essa abordagem logo fora deixada de lado ao início dos testes, quando se constatou que, ao realizar a separação dos grupos dessa forma, vértices muito distantes ficavam em um mesmo grupo. Essa configuração, durante o cálculo do fecho convexo dos grupos, fez com o fecho convexo de alguns grupos abrangesse uma grande do grafo e, consequentemente, forçasse muitos vértices a se tornar-se *outliers*.

Assim, estabeleceu-se uma nova forma de separação para os grupos. O pseudo-código para o novo procedimento é descrito no Apêndice 10:

Nessa nova versão de atribuição para os grupos, foi levada em consideração a distância dos vértices entre si no grafo G. Para cada vértice i ainda não alocado em nenhum grupo, ordenamos todos os vértices da sua mesma classe de em relação à distância a ele de forma crescente. Então, inserimos em um grupo vazio de sua classe o vértice i e os vértices mais próximos a i ainda não alocados em nenhum grupo pertencentes à mesma classe de i. A

quantidade de vértices adicionados nesse grupo anteriormente vazio é definida por $|V_{K(i)}|/|C_{K(i)}|$, pois a distribuição desejada deve ser uniforme. Essa nova abordagem de configuração inicial reduziu a ocorrência de grupos com vértices muito distantes entre si, o que ocasionou uma diminuição de violações das restrições do tipo 3, 4 e 5.

4.2 Calculando o fecho convexo geodésico

Depois de atender às restrições de tipo 1 e 2 com a alocação dos vértices realizada da maneira descrita anteriormente, procuramos pelas possíveis restrições violadas dos tipos 3, 4 e 5 a fim de satisfazê-las. Para isso, calculamos o fecho convexo de cada grupo em cada uma das duas classes e verificamos se há violação de alguma das restrições supramencionadas do problema.

4.2.1 Restrições de tipo 3 *e* 4

Se encontrarmos uma restrição violada do tipo 3 ou do tipo 4, atestamos a existência de algum vértice $i \in V_{BR}$ pertencente ao fecho convexo de um grupo de classe oposta. Neste caso, retiramos i do grupo em que estava afim de torná-lo um *outlier*, e recalculamos o fecho convexo ao qual i pertencia. O pseudo-código para esse procedimento é descrito no Apêndice 8:

4.2.2 Restrição de tipo 5

Se encontrarmos uma restrição violada do tipo 5, temos que $H[A_k] \cap H[A_{k'}] \cap V_N \neq \emptyset$, para algum $k \in C_B$ e $k' \in C_R$. Neste caso, determinamos um vértice $v \in A_k \cup A_{k'}$ como *outlier*, retirando-o do grupo ao qual pertence. A escolha de tal vértice v é feita arbitrariamente dentro do grupo de menor cardinalidade ($\min\{|A_k|, |A_{k'}|\}$). Esse processo de remoção é repetido até que a restrição do tipo 5 considerada se torne-se satisfeita. O pseudo-código para esse procedimento é descrito no Apêndice 8:

4.3 Calculando o número de *outliers*

Satisfeitas as restrições 1-5 do problema, calculamos o número de vértices que se tornaram *outliers*. Para esse cálculo, levamos em consideração que para ser *outlier*, o vértice precisa ser inicialmente classificado e não pertencer a nenhum grupo de sua classe.

4.4 Algoritmo final

Após a implementação de todos os passos anteriores, o algoritmo que descreve a heurística gulosa para o problema CG representa a composição de todas as etapas anteriores adicionando-se um teste simples, através do qual verificamos se o número de *outliers* gerados é maior do que o número de vértices do menor conjunto V_B ou V_R. Caso seja confirmado que o número de *outliers* é maior do que a cardinalidade do menor conjunto, a solução dada é a trivial, em que o menor conjunto é atualizado como um conjunto composto completamente de *outliers*. O pseudo-código da heurística gulosa é apresentado no Apêndice 8:

4.5 Experimentos computacionais e avaliação

Implementada a heurística gulosa descrita na seção anterior, realizamos experimentos computacionais com um grande número de instâncias geradas aleatoriamente e algumas instâncias realistas de aplicações práticas.

Para cada instância testada, analisamos o valor da solução obtida pela heurística gulosa, o tempo de execução e a acurácia. A análise é feita através da avaliação do comportamento da heurística quando se altera os parâmetros de número de grupos permitidos $|C_B| = L_B$ e $|C_R| = L_R$.

5 RESULTADOS

Nesta seção, faz-se um relato detalhado do que se obteve com os experimentos computacionais.

5.1 Ambiente de testes

Os testes foram realizados em uma máquina virtual com a seguinte configuração: CPU Intel Xeon CPU E5, 2G de memória RAM com S.O. Ubuntu 16.04 Xenial.

5.2 Instâncias sintéticas Euclidianas

As *instâncias sintéticas Euclidianas* derivam de instâncias geradas para a versão Euclidiana do problema de classificação. Tais instâncias foram geradas com o propósito de ser linearmente separáveis quando alguns poucos pontos são considerados outliers. Para cada instância gerada dessa forma, derivam-se outras 10 para o problema CG a seguir: construímos o grafo de similaridade associado usando a transformação definida em Zaki e Jr (2014), em que cada ponto se torna um vértice. Dessa forma, escolhemos aleatoriamente 30% dos vértices para se tornarem-se não classificados (eles formam o conjunto de validação).

Essa transformação do *dataset* em um grafo é descrito da seguinte forma: cada vértice i do grafo representa um ponto x_i do conjunto de dados da versão Euclidiana. Para criar as arestas, primeiro calculamos a similaridade entre os pontos. Para isso, utilizamos a função de kernel de Gauss, dada por:

$$a_{ij} = exp\left\{\frac{-||x_i - x_j||^2}{2}\right\},$$

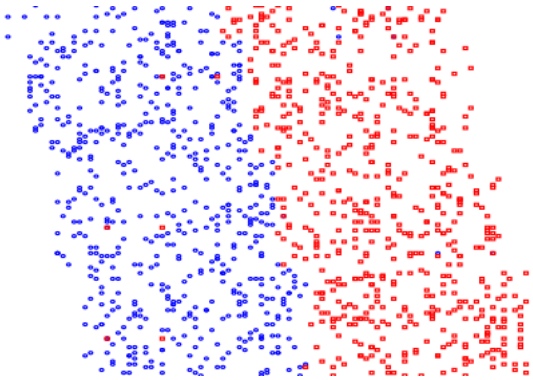
onde a fórmula $||x_i - x_j||^2 = \sum_{k=1}^{2}(x_i(k) - x_j(k))^2$ representa o quadrado da distância Euclidiana entre os pontos x_i e x_j. Assim, cada aresta (x_i, x_j) tem um peso de similaridade a_{ij} correspondente ao valor de similaridade entre x_i e x_j. Em seguida, para cada vértice i, foi calculado o vetor q de vértices mais próximos em termos do valor de similaridade, dado por:

$$N_q(i) = \{j \in V \mid j \neq i, a_{ij} \geq a_{iq}\},$$

onde a_{iq} representa o valor de similaridade entre i e seu q-ésimo vizinho mais próximo (usamos q = 10% de $|V|$). Desse modo, uma aresta é adicionada entre os vértices i e j se, e somente se, ambos, i e j, são suficientemente próximos entre si, isto é, $j \in N_q(i)$ e $i \in N_q(j)$. Finalmente, se

o resultado for um grafo desconexo, adicionamos as q arestas mais próximas entre cada par de componente conexa, tornando o grafo igualmente conexo. Na Figura 4 temos a ilustração de uma instância Euclidiana que, transformada em grafo com o procedimento descrito, torna-se uma instância sintética Euclidiana.

Figura 4 – Exemplo de instância Euclidiana



Fonte: Elaborada pelo Autor

5.3 Instâncias realistas

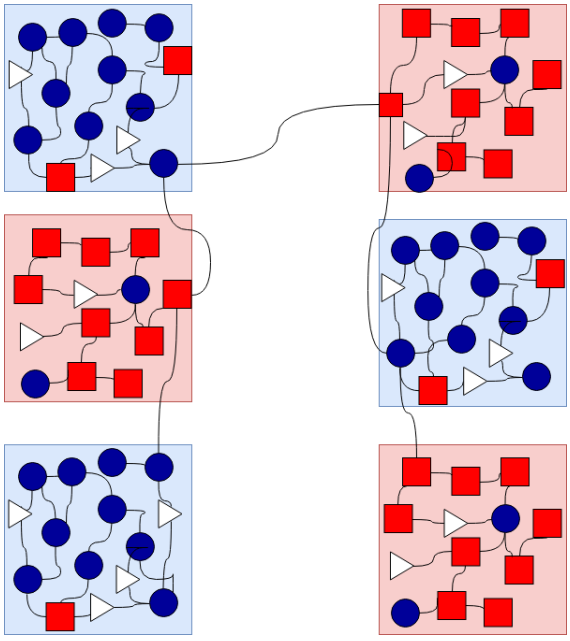
As *instâncias realistas* derivam-se de instâncias da versão Euclidiana de um conjunto de dados realistas da doença de Parkinson (LITTLE et al., 2007), os quais são encontrados em <https://archive.ics.uci.edu/ml/datasets.html>. Nesse conjunto de dados, cada ponto fornece informações de pacientes utilizadas na prevenção de novos diagnósticos da doença de Parkinson. O conjunto de dados é composto por 195 pontos, cada um deles contendo informações de medição de voz biomédica de uma pessoa divididas em 22 atributos. Entre esses 195 pontos, 48 pontos estão relacionados a pessoas saudáveis e 147 a pessoas com doença de Parkinson. E a transformação dos dados para grafo é feita de maneira similar à apresentada para as instâncias sintéticas Euclidianas.

5.4 Instâncias sintéticas de grafo

Nos testes posteriores, utilizou-se uma nova leva de instâncias, chamadas de *sintéticas de grafo*, desta vez variando o número de vértices do grafo de 100 até 1000. Essas instâncias foram modeladas de modo a simular o padrão geodésico e aproveitar ao máximo a heurística proposta pelo presente trabalho, sendo expressas de uma maneira um pouco diferente daquelas utilizadas preliminarmente.

Para a criação de uma instância sintética de grafo de dimensão 3, gera-se aleatoriamente três componentes conexas para cada classe, azul e vermelha, cada componente com a mesma quantidade de vértices $|V|/6$ (se sobrar vértices eles são colocados na última componente). A primeira componente azul é ligada com a primeira componente vermelha através de apenas uma aresta (u, v). A primeira componente vermelha também se liga através de uma única aresta com a segunda componente azul e assim se segue como ilustrado na Figura 5. Observe que é sempre o mesmo vértice que faz a ligação de uma componente com suas componentes vizinhas de classes opostas. Além disso, a primeira componente azul não se liga à última componente vermelha. Por fim, são escolhidos 30% dos vértices, que passarão a ser não classificados, compondo o conjunto de validação.

Figura 5 – Exemplo de grafo de similaridade onde os vértices quadrados vermelhos são os vértices de V_R e os vértices circulares azuis são os vértices de V_B e os demais vértices circulares triangulares brancos são vértices de V_N.



Fonte: Elaborada pelo Autor

5.5 Resultados constatados

A seguir são apresentadas as tabelas contendo os resultados obtidos:

Quadro 2 – Tabelas demonstrando o tempo de execução para as Instâncias sintéticas de grafos.

nome da instância	$\|C_k\|$				
	1	2	3	4	5
syntheticGraph_v100_d11_l3_br70-1	0.081526	0.050697	0.032301	0.048545	0.036645
syntheticGraph_v200_d11_l3_br70-1	1.7649	1.1051	0.7423	0.74525	0.61962
syntheticGraph_v300_d11_l3_br70-1	4.8822	4.1542	2.8371	3.3354	2.5732
syntheticGraph_v400_d11_l3_br70-1	15.467	19.43	15.572	13.889	10.661
syntheticGraph_v500_d11_l3_br70-1	49.267	59.737	58.813	46.24	44.462
syntheticGraph_v1000_d11_l3_br70-1	Null	Null	Null	Null	Null

Fonte: Elaborada pelo Autor

Quadro 3 – Tabelas demonstrando a acurácia para as Instâncias sintéticas de grafos.

nome da instância	$\|C_k\|$				
	1	2	3	4	5
syntheticGraph_v100_d11_l3_br70-1	86.67	86.67	90.00	86.67	86.67
syntheticGraph_v200_d11_l3_br70-1	95.00	95.00	95.00	95.00	95.00
syntheticGraph_v300_d11_l3_br70-1	88.89	88.89	88.89	88.89	88.89
syntheticGraph_v400_d11_l3_br70-1	88.33	89.17	89.17	89.17	89.17
syntheticGraph_v500_d11_l3_br70-1	91.33	91.33	90.67	73.33	73.33
syntheticGraph_v1000_d11_l3_br70-1	Null	Null	Null	Null	Null

Fonte: Elaborada pelo Autor

Quadro 4 – Tabelas demonstrando o valor-solução para as Instâncias sintéticas de grafos.

nome da instância	$\|C_k\|$					OPT
	1	2	3	4	5	
syntheticGraph_v100_d11_l3_br70-1	31	20	9	17	14	6
syntheticGraph_v200_d11_l3_br70-1	68	68	62	57	51	13
syntheticGraph_v300_d11_l3_br70-1	99	76	76	76	76	20
syntheticGraph_v400_d11_l3_br70-1	136	136	136	136	136	26
syntheticGraph_v500_d11_l3_br70-1	173	173	173	173	173	33
syntheticGraph_v1000_d11_l3_br70-1	Null	Null	Null	Null	Null	66

Fonte: Elaborada pelo Autor

Quadro 5 – Tabelas demonstrando o tempo de execução para as Instâncias sintéticas Euclidianas.

nome da instância	$\|C_k\|$				
	1	5	10	20	30
syntheticDim2_v104_d7_l1_br70-1	0.087714	0.070392	0.091582	0.050397	0.014237
syntheticDim2_v104_d7_l1_br70-2	0.070761	0.074461	0.1009	0.056555	0.027574
syntheticDim2_v104_d7_l1_br70-3	0.1411	0.079228	0.048865	0.021785	0.020112
syntheticDim2_v104_d7_l1_br70-4	0.12068	0.1067	0.067653	0.022362	0.008561
syntheticDim2_v104_d7_l1_br70-5	0.095585	0.11209	0.083741	0.021436	0.009221
syntheticDim2_v104_d7_l1_br70-6	0.10003	0.094891	0.053598	0.014845	0.009086
syntheticDim2_v104_d7_l1_br70-7	0.092823	0.080754	0.04992	0.028147	0.037774
syntheticDim2_v104_d7_l1_br70-8	0.055474	0.056935	0.086447	0.021876	0.018401
syntheticDim2_v104_d7_l1_br70-9	0.097811	0.093824	0.053186	0.027971	0.014381
syntheticDim2_v104_d7_l1_br70-10	0.12305	0.084024	0.084095	0.01989	0.00932

Fonte: Elaborada pelo Autor

Quadro 6 – Tabelas demonstrando a acurácia para as Instâncias sintéticas Euclidianas.

nome da instância	$\|C_k\|$				
	1	5	10	20	30
syntheticDim2_v104_d7_l1_br70-1	45.16	45.16	45.16	45.16	51.61
syntheticDim2_v104_d7_l1_br70-2	51.61	51.61	51.61	51.61	90.32
syntheticDim2_v104_d7_l1_br70-3	54.84	54.84	58.06	93.55	93.55
syntheticDim2_v104_d7_l1_br70-4	38.71	38.71	38.71	38.71	45.16
syntheticDim2_v104_d7_l1_br70-5	48.39	48.39	48.39	61.29	96.77
syntheticDim2_v104_d7_l1_br70-6	51.61	51.61	51.61	77.42	90.32
syntheticDim2_v104_d7_l1_br70-7	45.16	45.16	45.16	45.16	64.52
syntheticDim2_v104_d7_l1_br70-8	35.48	35.48	35.48	35.48	61.29
syntheticDim2_v104_d7_l1_br70-9	45.16	45.16	45.16	45.16	96.77
syntheticDim2_v104_d7_l1_br70-10	61.29	61.29	61.29	90.32	90.32

Fonte: Elaborada pelo Autor

Quadro 7 – Tabelas demonstrando o valor-solução para as Instâncias sintéticas Euclidianas.

nome da instância	$\|C_k\|$				
	1	5	10	20	30
syntheticDim2_v104_d7_l1_br70-1	33	33	33	33	22
syntheticDim2_v104_d7_l1_br70-2	35	35	35	35	4
syntheticDim2_v104_d7_l1_br70-3	36	36	26	19	4
syntheticDim2_v104_d7_l1_br70-4	31	31	31	31	21
syntheticDim2_v104_d7_l1_br70-5	34	34	34	27	5
syntheticDim2_v104_d7_l1_br70-6	36	36	36	9	4
syntheticDim2_v104_d7_l1_br70-7	33	33	33	33	26
syntheticDim2_v104_d7_l1_br70-8	30	30	30	30	14
syntheticDim2_v104_d7_l1_br70-9	33	33	33	33	7
syntheticDim2_v104_d7_l1_br70-10	33	33	33	3	3

Fonte: Elaborada pelo Autor

Quadro 8 – Tabelas demonstrando o tempo de execução para as Instâncias reais de Parkinson.

nome da instância	$\|C_k\|$				
	1	10	20	30	50
parkinsons_v195_d5_l1_br70-1	0.8477	1.092	0.77804	0.26305	0.31261
parkinsons_v195_d5_l1_br70-2	1.4015	1.3869	0.74755	0.27788	0.17168
parkinsons_v195_d5_l1_br70-3	0.82322	1.0735	0.95872	0.21336	0.31554
parkinsons_v195_d5_l1_br70-4	0.7301	1.0228	0.49161	0.27904	0.20022
parkinsons_v195_d5_l1_br70-5	0.68624	1.2789	0.60959	0.29779	0.26347
parkinsons_v195_d5_l1_br70-6	0.59624	1.3298	0.5976	0.50423	0.24798
parkinsons_v195_d5_l1_br70-7	0.79593	1.2732	0.7605	0.3998	0.19153
parkinsons_v195_d5_l1_br70-8	0.91253	0.98816	0.78604	0.31154	0.21312
parkinsons_v195_d5_l1_br70-9	38	1.574	0.71618	0.47334	0.20153
parkinsons_v195_d5_l1_br70-10	0.84743	1.3804	0.89548	0.31529	0.2023

Fonte: Elaborada pelo Autor

Quadro 9 – Tabelas demonstrando a acurácia para as Instâncias reais de Parkinson.

nome da instância	$\|C_k\|$				
	1	10	20	30	50
parkinsons_v195_d5_l1_br70-1	86.21	22.41	22.41	86.21	86.21
parkinsons_v195_d5_l1_br70-2	68.97	68.97	43.10	68.97	68.97
parkinsons_v195_d5_l1_br70-3	77.59	77.59	32.76	77.59	77.59
parkinsons_v195_d5_l1_br70-4	70.69	70.69	70.69	70.69	70.69
parkinsons_v195_d5_l1_br70-5	87.93	86.21	32.76	74.14	74.14
parkinsons_v195_d5_l1_br70-6	75.86	75.86	75.86	75.86	75.86
parkinsons_v195_d5_l1_br70-7	79.31	27.59	25.86	79.31	79.31
parkinsons_v195_d5_l1_br70-8	75.86	75.86	32.76	75.86	75.86
parkinsons_v195_d5_l1_br70-9	82.76	82.76	29.31	82.76	82.76
parkinsons_v195_d5_l1_br70-10	75.86	27.59	27.59	75.86	75.86

Fonte: Elaborada pelo Autor

Quadro 10 – Tabelas demonstrando o valor-solução para as instâncias realistas de Parkinson.

nome da instância	$\|C_k\|$				
	1	10	20	30	50
parkinsons_v195_d5_l1_br70-1	40	40	40	40	40
parkinsons_v195_d5_l1_br70-2	30	30	30	29	29
parkinsons_v195_d5_l1_br70-3	35	35	35	35	35
parkinsons_v195_d5_l1_br70-4	31	31	31	31	34
parkinsons_v195_d5_l1_br70-5	33	33	33	33	33
parkinsons_v195_d5_l1_br70-6	34	34	34	34	34
parkinsons_v195_d5_l1_br70-7	35	35	35	34	34
parkinsons_v195_d5_l1_br70-8	34	34	34	34	34
parkinsons_v195_d5_l1_br70-9	38	38	38	38	38
parkinsons_v195_d5_l1_br70-10	34	34	34	34	34

Fonte: Elaborada pelo Autor

5.6 Análises dos resultados

Após a realização dos testes, decidiu-se compará-los de três maneiras distintas: por tempo, pelo valor da solução e pela acurácia, sempre levando em consideração os números de grupos em C_B e C_R. Essas comparações são mostradas abaixo:

5.6.1 Por tempo

Na avaliação de tempo de execução, analisamos a variação de tempo para cada instância de teste conforme fosse alterado o número de grupos de C_B e C_R. Com isso, podemos notar, em geral, uma considerável diminuição de tempo de execução em todas as instâncias, conforme fosse aumentado o número de grupos. Em alguns casos, a diminuição de tempo chegou a superar 50%. Ocorreram também casos em que o tempo sofreu um leve aumento inferior a 20%; nesses casos, ao aumentar-se novamente o valor do número de grupos, o tempo voltou a baixar, com exceção da instância sintética de grafos para 1000 vértices cujo algoritmo não conseguiu terminar a execução. Podemos verificar que o tempo de execução é bastante baixo, visto que se demora apenas alguns segundos para instâncias de tamanho médio. Logo, analisando o parâmetro de tempo de execução podemos concluir que nossa heurística obtém resultados satisfatórios.

5.6.2 Acurácia

Na avaliação da acurácia, analisamos a quantidade de predições corretas dentro do conjunto de validação de cada instância. Para isso, mostramos o cálculo de quantidade de predições corretas por tamanho do conjunto de validação, dado em porcentagem. Verificamos que para as instâncias sintéticas Euclidianas, a heurística apresentou uma acurácia com valor baixo. Isso indica possivelmente que tais instâncias não possuíam um padrão que poderia ser expresso por convexidade geodésica. Já para as instâncias de Parkinson, mesmo com um número baixo de grupos permitidos, constataram-se bons valores de acurácia, o que demonstra um potencial da abordagem geodésica em grafos para classificação em instâncias reais. Finalmente, para as instâncias sintéticas de grafo, podemos constatar o que poderíamos prever: os melhores resultados são alcançados quando se tem $|C_B| = |C_R| = 3$ já que as instâncias geradas foram feitas para simular o padrão geodésico onde as amostras poderiam ser separados em três componentes convexas de cada classe.

5.6.3 Valor da solução

Na análise do valor da solução (número de *outliers*), conseguimos verificar que quanto maior o número de grupos permitidos, menor o número de *outliers* na solução retornada pela heurística, como esperado. Porém, isso não acontece sempre nas instâncias sintéticas de grafo devido à heurística gulosa fazer uso de todos os grupos permitidos, mesmo que não seja a melhor opção. Podemos notar também que a variação do valor de solução nas instâncias realistas de Parkinson é muito pequena, mesmo para uma grande quantidade de grupos. Isso mostra uma evidência de que para tais instâncias, o número de *outliers* ótimo está próximo do valor da solução retornada pela heurística gulosa, demonstrando que ela traz resultados satisfatórios.

6 CONCLUSÃO

Neste trabalho, apresentamos uma heurística gulosa e realizamos testes computacionais para avaliar sua performance. Após todos os testes e análises foi concluído que a heurística proposta é promissora e traz resultados satisfatórios para instâncias de pequeno e médio porte. Porém, a mesma ainda necessita de muitas melhorias, principalmente por conta de sua simplicidade. Pelos experimentos, temos evidências de que o problema se mostra difícil até para a obtenção de uma simples solução viável melhor que a solução trivial, como foi visto nos testes que para muitas instâncias o número de grupos impactava diretamente no tempo, mas não muito no valor da solução. Vale ressaltar que mesmo não sendo indicada para instâncias muito grandes no quesito valor solução, para o quesito tempo de execução, a heurística pode vir a ser uma ótima alternativa. Além disso, a acurácia da abordagem geodésica e sua implementação heuristicamente através da heurística gulosa deste trabalho se mostrou bastante promissora e incentiva o estudo e aprofundamento de tais ideias em outros contextos.

Como trabalhos futuros, podemos citar o melhoramento da heurística gulosa proposta por este trabalho, a elaboração de novas heurísticas tomando as ideias apresentadas aqui como base e a aplicação das soluções retornadas em meta-heurísticas a fim de obter soluções ainda melhores.

REFERÊNCIAS

ARAÚJO, P. H. M.; CAMPÊLO, M. B.; CORRÊA, R. C.; LABBÉ, M. The geodesic classification problem on graphs. **Latin & American Algorithms, Graphs and Optimization Symposium**, 2019.

ARTIGAS, D.; DANTAS, S.; DOURADO, M. C.; SZWARCFITER, J. L. Partitioning a graph into convex sets. **Discrete Mathematics**, v. 311, n. 17, p. 1968–1977, 2011. Disponível em: https://doi.org/10.1016/j.disc.2011.05.023 . Acesso em: 20 set 2018.

BERTSIMAS, D.; SHIODA, R. Classification and regression via integer optimization. **Operations Research**, INFORMS, v. 55, n. 2, p. 252–271, 2007.

CáCERES, J.; HERNANDO, C.; MORA, M.; PELAYO, I. M.; PUERTAS, M. L.; SEARA, C. On geodetic sets formed by boundary vertices. **Discrete Math.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 306, n. 2, p. 188–198, fev. 2006. ISSN 0012-365X. Disponível em: http://dx.doi.org/10.1016/j.disc.2005.12.012 . Acesso em: 12 set 2018.

CHANGAT, M.; MATHEW, J. On triangle path convexity in graphs. **Discrete Mathematics**, v. 206, n. 1, p. 91 – 95, 1999. ISSN 0012-365X. Disponível em: http://www.sciencedirect.com/science/article/pii/S0012365X9800394X . Acesso em: 12 out 2018.

CORRÊA, R. C.; BLAUM, M.; MARENCO, J.; KOCH, I.; MYDLARZ, M. An integer programming approach for the 2-class single-group classification problem. **Latin & American Algorithms, Graphs and Optimization Symposium**, 2019.

CORRÊA, R. C.; DONNE, D. D.; MARENCO, J. On the combinatorics of the 2-class classification problem. **Discrete Optimization**, 2018. ISSN 1572-5286. In press.

CORRÊA, R. C.; DONNE, D. D.; MARENCO, J. On the combinatorics of the 2-class classification problem. **Discrete Optimization**, Elsevier, 2018.

CORTES, C.; VAPNIK, V. Support-vector networks. In: **Machine Learning**. [S.l.: s.n.], 1995. p. 273–297.

DOURADO, M. C.; GIMBEL, J. G.; KRATOCHVÍL, J.; PROTTI, F.; SZWARCFITER, J. L. On the computation of the hull number of a graph. **Discrete Math.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 309, n. 18, p. 5668–5674, set. 2009. ISSN 0012-365X. Disponível em: http://dx.doi.org/10.1016/j.disc.2008.04.020 . Acesso em: 15 out 2018.

FARBER, M.; JAMISON, R. E. Convexity in graphs and hypergraphs. **SIAM Journal on Algebraic Discrete Methods**, v. 7, n. 3, p. 433–444, 1986. Disponível em: https://doi.org/10.1137/0607049 . Acesso em: 19 out 2018.

HARARY, F.; NIEMINEN, J. Convexity in graphs. **J. Differential Geom.**, Lehigh University, v. 16, n. 2, p. 185–190, 1981. Disponível em: https://doi.org/10.4310/jdg/1214436096 , Acesso em: 23 out 2018.

HONG, L.; DAVISON, B. D. Empirical study of topic modeling in twitter. In: **Proceedings of the First Workshop on Social Media Analytics**. New York, NY, USA: ACM, 2010. (SOMA '10), p. 80–88. ISBN 978-1-4503-0217-3. Disponível em: http://doi.acm.org/10.1145/1964858.1964870 . Acesso em: 25 out 2018.

LITTLE, M. A.; MCSHARRY, P. E.; ROBERTS, S. J.; COSTELLO, D. A.; MOROZ, I. M. Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection. **BioMedical Engineering OnLine**, v. 6, n. 1, p. 23, Jun 2007. ISSN 1475-925X. Disponível em: https://doi.org/10.1186/1475-925X-6-23 . Acesso em: 27 out 2018.

PELAYO, I. M. **Geodesic Convexity in Graphs**. New York: Springer-Verlag, 2013.

ZAKI, M. J.; JR, W. M. **Data Mining and Analysis**: Fundamental concepts and algorithms. New York, NY, USA: Cambridge University Press, 2014. ISBN 0521766338, 9780521766333.

APÊNDICE A: DIVISÃO 1

Algoritmo 1: Algoritmo de divisão 1

```
1  início
2      para cada i ∈ V_BR faça
3          if i ∈ V_B then
4              Coloca i no próximo grupo da classe Azul. Caso já esteja no último grupo azul,
                volte para o primeiro.
5          end
6          if i ∈ V_R then
7              Coloca i no próximo grupo da classe Vermelha. Caso já esteja no último grupo
                vermelho, volte para o primeiro.
8          end
9      fim
10 fim
```

Fonte: Elaborada pelo Autor.

APÊNDICE B: DIVISÃO 2

Algoritmo 2: Algoritmo de divisão 2

```
1 início
2     para cada i ∈ V_BR faça
3         if i não está em nenhum grupo ainda then
4             Gera-se a vizinhança de i em relação à distância de forma crescente;
5             Para um grupo vazio da classe de i, coloca-se i e os vértices mais próximos a i
                que ainda não estiverem em nenhum grupo e que possuem a mesma classe até
                completar a quantidade |V_K(i)|/|C_K(i)|.;
6         end
7     fim
8 fim
```

Fonte: Elaborada pelo Autor.

APÊNDICE C: RESTRIÇÕES 3 E 4

Algoritmo 3: Algoritmo para as restrições 3 e 4

```
1 início
2     para cada i ∈ V_BR faça
3         if i ∈ H[A_k'] para algum k' ∈ C_K̄(i) then
4             remover i do grupo de sua classe a qual pertence;
5             recalcular o fecho convexo do grupo A_k';
6         end
7     fim
8 fim
```

Fonte: Elaborada pelo Autor.

APÊNDICE C: RESTRIÇÃO 5

Algoritmo 4: Algoritmo para a restrição 5

```
1 início
2     para cada i ∈ V_N faça
3         if i ∈ H[A_k] ∩ H[A_k'] para um k ∈ C_B e um k' ∈ C_R then
4             remover um vértice v do grupo de menor cardinalidade entre A_k e A_k';
5             recalcular o fecho convexo do grupo modificado;
6         end
7     fim
8 fim
```

Fonte: Elaborada pelo Autor.

APÊNDICE E: HEURÍSTICA

Algoritmo 5: Algoritmo final da heurística gulosa

início
    Alocar os vértices de V_{BR} nos grupos de maneira uniforme;
    Satisfazer restrições do tipo 3 e 4;
    Satisfazer restrições do tipo 5;
    Calcular o número de *outliers*;
    if *número de outliers é menor que* $\min\{|V_B|, |V_R|\}$ **then**
        retorna o número de *outliers*;
    else
        retorna a solução trivial;
    end
fim

Fonte: Elaborada pelo Autor.