

MÉTODOS PARA DETERMINAÇÃO DO NÚMERO DE ENVOLTÓRIA GEODÉSICO DE UM GRAFO*

J. Araujo

ParGO, Departamento de Matemática, Universidade Federal do Ceará
Av. Humberto Monte, s/n, Campus do Pici, Bloco 914. CEP 60440-900. Fortaleza, Ceará, Brasil
julio@mat.ufc.br

M. Campêlo & G. H. de Sousa

ParGO, Departamento de Estatística e Matemática Aplicada, Universidade Federal do Ceará
Av. Humberto Monte, s/n, Campus do Pici, Bloco 910. CEP 60440-900. Fortaleza, Ceará, Brasil
mcampelo@lia.ufc.br & gabriel.hellen@hotmail.com

RESUMO

Dado um grafo $G = (V, E)$, um subconjunto $S \subseteq V(G)$ é *convexo* (na convexidade geodésica) se, para todo $u, v \in S$, os vértices em qualquer u, v -geodésico (u, v -caminho mínimo) também pertencem a S . A *envoltória convexa* de S , denotada por $[S]$, é o menor conjunto convexo que contém S . O subconjunto S é um *conjunto de envoltória* se $[S] = V(G)$. O *número de envoltória* de G , denotado por $hn(G)$, é o menor k tal que G possui um conjunto de envoltória de cardinalidade k .

É conhecido na literatura que determinar o número de envoltória de um grafo dado é um problema *NP-Difícil*, mesmo para cubos parciais, que formam uma subclasse de bipartidos. Além disso, existem algoritmos polinomiais e limitantes para algumas classes de grafos, além de, mais recentemente, alguns resultados em Complexidade Parametrizada. Neste trabalho, dois modelos matemáticos e uma heurística para se calcular este parâmetro são estudados, implementados e comparados.

PALAVRAS CHAVE. Convexidade Geodética, Número de Envoltória, Heurística.

ABSTRACT

Given a graph $G = (V, E)$, a subset $S \subseteq V(G)$ is *convex* (in the geodetic convexity) if, for every $u, v \in S$, the vertices in any u, v -geodesic (shortest u, v -path) also belong to S . The *convex hull* of S , denoted by $[S]$, is the minimum convex set containing S . The subset S is a *hull set* if $[S] = V(G)$. The *hull number* of G , denoted by $hn(G)$, is the minimum k such that G has a hull set of cardinality k .

It is known that determining the hull number of a graph is an *NP-hard* problem, even for partial cubes, that form a subclass of bipartite graphs. Furthermore, there are polynomial-time algorithms and bounds for some graph classes, besides, more recently, some results on Parameterized Complexity. In this work, two mathematical models and a heuristic to compute such parameter are studied, implemented and compared.

KEYWORDS. Geodetic Convexity, Hull number, Heuristic.

*Este trabalho foi financiado pelo CNPq sob projetos 310234/2015-8 e 401519/2016-3.

1. Introdução

Uma classe de problemas combinatórios que tem recebido bastante atenção recentemente compreende o que se chama genericamente de problemas de contaminação ou infecção ou convexidade em grafos. Há vários tipos de contaminação consideradas na literatura, que podem ser vistos como um processo iterativo, onde, a partir de um subconjunto de vértices inicialmente contaminados, passo a passo, outros também o serão. Por exemplo, podemos considerar que dois vértices contaminados contaminam, no passo seguinte, todos aqueles que estão em algum caminho mínimo entre eles [Pelayo, 2013], ou todos aqueles que são simultaneamente vizinhos dos dois [Penso et al., 2015], etc. Também são estudadas na literatura variações em que vértices contaminados podem se tornar “sadios”, segundo alguma condição pré-determinada [Dourado et al., 2012].

Nesse contexto, um dos problemas fundamentais consiste em determinar o menor número de vértices que, inicialmente contaminados, infectam, iterativamente, todo o grafo em algum momento. Esse menor valor é conhecido como número de envoltória [Everett e Seidman, 1985]. Veja que tal problema pode modelar na prática questões como a determinação da menor quantidade de indivíduos para disseminar informações em uma rede social, ou a menor quantidade de indivíduos para disseminar uma doença ou epidemia em um local, por exemplo.

Neste trabalho, estudamos o processo de infecção em grafos relativo à convexidade geodésica. A seguir, formalizamos conceitos relacionados ao tema. Para definições sobre grafos, algoritmos e complexidade computacional não apresentadas aqui, sugerimos [West, 2000; Cormen et al., 2009; Garey e Johnson, 1990]. Apenas estudamos grafos simples, conexos e não direcionados, a menos que explicitamente dito o contrário.

Dados um grafo $G = (V, E)$ e dois vértices $u, v \in V(G)$, dizemos que um u, v -geodésico é um u, v -caminho de menor comprimento possível. Um subconjunto $S \subseteq V(G)$ é *convexo*, na convexidade geodésica, se, para todos $u, v \in S$, os vértices em qualquer u, v -geodésico também pertencem a S . A *envoltória convexa* de S , denotada por $[S]$, é o menor conjunto convexo que contém S . O subconjunto S é um *conjunto de envoltória* se $[S] = V(G)$. O *número de envoltória* de G , denotado por $hn(G)$, é o menor k tal que G possui um conjunto de envoltória de cardinalidade k .

Note que a envoltória convexa de S pode ser obtida pelo processo de infecção a partir de S , quando consideramos que dois vértices infectados u e v contaminam todos aqueles de um u, v -geodésico. Portanto, o número de envoltória é a menor quantidade de vértices a partir dos quais se infecta todo o grafo. Esse parâmetro foi definido nos anos 80 e tem sido amplamente estudado desde então. Veja, por exemplo, os trabalhos de [Everett e Seidman, 1985; Dourado et al., 2009; Araujo et al., 2013, 2016; Albenque e Knauer, 2016; Bessy et al., 2017; Marcilon, 2017].

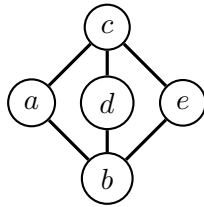


Figura 1: G_1

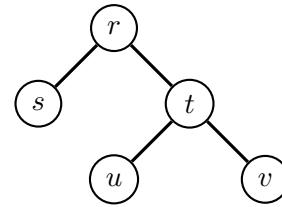


Figura 2: G_2

Ilustramos os conceitos acima com os grafos das figuras 1 e 2. Observe que no grafo G_1 há diversos conjuntos de envoltória com dois vértices. Por exemplo, $\{a, e\}$ é um conjunto de envoltória (mínimo), já que os vértices b e c serão infectados em uma primeira iteração (pois pertencem a caminhos mais curtos de a para e) e, então, d será infectado na segunda iteração, por pertencer a um b, c -geodésico. Já na Figura 2, vemos o exemplo de uma árvore. Note que, para G_2 , bem como para qualquer árvore, o conjunto de envoltória mínimo é único e consiste do seu conjunto de folhas. Assim $hn(G_1) = 2$ e $hn(G_2) = 3$.

O problema de determinar o número de envoltória de um grafo é *NP*-difícil, mesmo para grafos cubos parciais [Albenque e Knauer, 2016] e grafos cordais [Bessy et al., 2017]. Um *cubo parcial* é um subgrafo isométrico do hipercubo e, portanto, o problema é *NP*-difícil para grafos bipartidos. Uma demonstração independente desse fato pode ser encontrada em [Araujo et al., 2013]. Há na literatura diversos limitantes superiores, algoritmos polinomiais e algoritmos parametrizados para diversas classes de grafos, além de demonstrações que tais algoritmos paramétricos não existem, sob a hipótese de que $FPT \neq W[1]$ (veja alguns exemplos em [Dourado et al., 2009; Araujo et al., 2013, 2016; Marcilon, 2017]).

Porém, que conheçamos, não há formulação de programação linear-inteira com um número polinomial de restrições proposta na literatura, nem estudos que proponham e avaliem o comportamento de heurísticas polinomiais para obtenção de um conjunto de envoltória. Há entretanto a implementação de um algoritmo exponencial para obtenção do número de envoltória no SageMath [Sag] que sugere uma formulação de programação inteira com um número exponencial de restrições. Vale ressaltar que, embora a formulação seja facilmente obtida a partir da implementação no SageMath, acreditamos que a mesma não foi explicitamente proposta na literatura.

Neste trabalho, estudamos a formulação correspondente ao método implementado no SageMath e propomos uma formulação de programação inteira com uma quantidade polinomial de variáveis e restrições, bem como uma heurística polinomial para determinar o número de envoltória de um dado grafo. Em seguida, apresentamos os resultados de testes computacionais sobre instâncias correspondentes a grafos bipartidos e grafos arbitrários, tomados aleatoriamente. Comparamos então as soluções ótimas obtidas pela implementação, no CPLEX, das formulações propostas com as soluções obtidas pela heurística. Vale ressaltar que uma das formulações é resolvida por um algoritmo de planos-de-corte que corresponde ao método exponencial encontrado em [Sag]. Finalmente, na última seção, exibimos caminhos para o desenvolvimento da pesquisa sobre esse parâmetro nesta linha.

2. Algoritmo de Contaminação

Em nossa heurística e em um dos modelos, precisamos calcular a envoltória convexa de um dado conjunto S , para, por exemplo, verificar se o mesmo é um conjunto de envoltória. Em [Dourado et al., 2009], é mencionado um algoritmo polinomial para calcular tal envoltória. A partir dessa sugestão, foi implementado um algoritmo de contaminação, onde se recebe como entrada um subconjunto S com pelos menos 2 vértices, correspondendo àqueles já contaminados, e se retorna todo o conjunto resultante na contaminação geodésica aplicada a S . A seguir temos o pseudo-código:

Algorithm 1

```

1: function CONTAMINAÇÃO GEODÉSICA( $S$ )
2:    $S' \leftarrow S$ 
3:    $U \leftarrow \emptyset$ 
4:   while  $U \neq S'$  do
5:      $U \leftarrow$  vértices de todos os  $i, j$ -caminhos mínimos,  $\forall \{i, j\} \in S'$ 
6:      $S' \leftarrow S' \cup U$ 
7:   endwhile
8:   return  $S'$ 
9: end function

```

Para a descoberta dos caminhos mínimos (passo 5 do Algoritmo 1), adaptamos o clássico algoritmo de [Floyd, 1962; Warshall, 1962], que calcula o valor de todos os caminhos mínimos entre todos os pares de vértices de um grafo. Para a entrada do algoritmo, tem-se uma matriz $D_{n \times n}$, $n = |V(G)|$, onde a posição D_{ij} representa o custo da aresta ij , caso exista, e infinito caso contrário.

O algoritmo original atualiza apenas os valores de distâncias, não salvando a sequência de vértices e arestas que pertencem a um caminho mínimo entre dois vértices e desconsiderando casos em que temos mais de um caminho de mesmo custo. Como estamos interessados nos vértices internos de quaisquer caminhos mínimos, adicionamos uma matriz ao algoritmo, $H_{n \times n}$, onde cada posição H_{uv} armazenará os vértices que estão internos a um u, v -caminho mínimo, para cada par $u, v \in V(G)$. Para tanto, usamos o seguinte algoritmo:

Algorithm 2

```

1: function FLOYD-WARSHALL-MODIFICADO( $D_{n \times n}$ )
2:   for  $i = 1$  to  $n$ 
3:     for  $j = 1$  to  $n$ 
4:       for  $k = 1$  to  $n$ 
5:         if  $D_{ij} > D_{ik} + D_{kj}$  then
6:            $D_{ij} = D_{ik} + D_{kj}$ 
7:            $H_{ij} \leftarrow H_{ik} \cup H_{kj} \cup \{k\}$ 
8:         else if  $D_{ij} = D_{ik} + D_{kj}$  then
9:            $H_{ij} \leftarrow H_{ij} \cup H_{ik} \cup H_{kj} \cup \{k\}$ 
10:        endif
11:      endfor
12:    endfor
13:  endfor
14:  return  $D, H$ 
15: end function

```

3. Formulações

Propomos dois modelos de programação linear inteira para determinar o número de envoltória de um grafo. O primeiro é compacto (possui um número polinomial de variáveis e restrições); o segundo possui um número linear de variáveis, porém exponencial de restrições, em termos do número $n = |V(G)|$ de vértices do grafo.

Modelo 1: Passo de Contaminação.

Este modelo foi baseado no tempo para contaminação, ou seja, número de passos necessários para a infecção de todo o grafo. Observe que, se o grafo possuir pelo menos três vértices e não for o grafo completo, então $hn(G) \leq n - 1$ e, em no máximo $n - 2$ iterações, os demais vértices serão infectados. Defina:

- $P = n - 2$, em que P é o tempo máximo para contaminar todo o grafo;
- Variáveis:

$$\begin{aligned}
 - \quad x_i^p &= \begin{cases} 1, & \text{se } i \in V \text{ está contaminado no passo } p = 0, 1, \dots, P; \\ 0, & \text{caso contrário.} \end{cases} \\
 - \quad y_{ij}^p &= \begin{cases} 1, & \text{se } i, j \in V \text{ estão contaminados no passo } p = 0, 1, \dots, P; \\ 0, & \text{caso contrário.} \end{cases}
 \end{aligned}$$

Como o nosso objetivo é o menor número de vértices que, inicialmente contaminados, infectam iterativamente todo o grafo, então queremos que as variáveis x_i^0 somem o mínimo possível. Portanto, temos a seguinte função objetivo, que queremos minimizar:

$$\sum_{i \in V} x_i^0.$$

Para modelar a contaminação de um vértice k em determinado instante $p + 1$, temos duas possibilidades: ou ele está contaminado desde o início; ou no instante p dois vértices i e j estão contaminados e k está em algum i, j -caminho mínimo. Logo, temos a seguinte restrição:

$$x_k^{p+1} \leq \sum_{\{i,j\} \in H_k} y_{ij}^p + x_k^0, \quad \forall k \in V, p = 0, 1, \dots, P - 1,$$

onde H_k é o conjunto contendo todos os pares de vértices $\{i, j\}$, tais que k está em algum i, j -caminho mínimo. Tais conjuntos podem ser determinados a partir da saída do Algoritmo 2.

A garantia de que o conjunto contaminado inicialmente é de envoltória equivale a pedir que o processo de infecção definido acima resulte em todos os vértices contaminados no último instante. Em outros termos, devemos ter:

$$x_i^P = 1, \quad \forall i \in V.$$

A relação entre as variáveis x e y é definida por $y_{ij}^p = x_i^p x_j^p$, ou seja, o par $\{i, j\}$ está contaminado em um certo instante se, e somente se, ambos i e j estão. Dada a função objetivo, tal relação é assegurada no ótimo pelas restrições:

$$y_{ij}^p \leq x_i^p, y_{ij}^p \leq x_j^p, \quad \forall i, j \in V, p = 0, 1, \dots, P.$$

Resumindo, temos o seguinte modelo:

$$\begin{aligned} \min \quad & \sum_{i \in V} x_i^0 \\ \text{s.a: } & x_i^P = 1, & \forall i \in V. \\ & y_{ij}^p \leq x_i^p, y_{ij}^p \leq x_j^p, & \forall i, j \in V, p = 0, 1, \dots, P. \\ & x_k^{p+1} \leq \sum_{\{i,j\} \in H_k} y_{ij}^p + x_k^0, & \forall k \in V, p = 0, 1, \dots, P - 1. \\ & x_i^p \in \{0, 1\}, & \forall i \in V, p = 0, 1, \dots, P. \\ & y_{ij}^p \in \{0, 1\}, & \forall i, j \in V, p = 0, 1, \dots, P. \end{aligned}$$

Note que o modelo apresentado tem um número cúbico, com relação ao número de vértices do grafo dado como entrada, de variáveis e de restrições. Os valores das variáveis x_i^0 indicarão ao final da resolução do modelo se o vértice i participa de um conjunto de envoltória mínimo do grafo, e o número de envoltória será a soma de todas essas variáveis.

A implementação desse modelo é simples, dados os métodos mostrados anteriormente. Todos os componentes do modelo (função objetivo, restrições e variáveis) são de ordem polinomial, de modo que podemos gerá-lo em tempo polinomial.

Modelo 2: Co-convexo.

Dado um grafo G , um conjunto $S \subseteq V(G)$ é *co-convexo* se $V(G) \setminus S$ é convexo. Consequentemente, observe que pelo menos um vértice de um co-convexo S precisa estar infectado inicialmente; do contrário, como a envoltória de $V(G) \setminus S$ é o próprio conjunto, nenhum dos vértices de S se tornaria contaminado.

Consequentemente, pode-se deduzir que S' é um conjunto de envoltória se, e somente se, para todo conjunto co-convexo S , temos que $S' \cap S \neq \emptyset$ [Dourado et al., 2009]. Ou seja, um conjunto de envoltória pode ser visto como um transversal da família de co-convexos do grafo dado. Esse é o elemento chave da formulação a seguir, onde usamos as seguintes variáveis:

$$x_i = \begin{cases} 1, & \text{se } i \in V(G) \text{ é selecionado para o conjunto de envoltória;} \\ 0, & \text{caso contrário.} \end{cases}$$

A função objetivo, que queremos minimizar, é trivialmente dada por:

$$\sum_{i \in V} x_i.$$

Para a descrição das restrições, defina $CC(G)$ como a família de conjuntos co-convexos de G . Pela observação supramencionada, temos:

$$\sum_{i \in S} x_i \geq 1, \forall S \in CC(G).$$

Logo, o modelo completo é:

$$\min \sum_{i \in V} x_i \tag{1}$$

$$\text{s.a: } \sum_{i \in S} x_i \geq 1 \quad \forall S \in CC(G) \tag{2}$$

$$x_i \in \{0, 1\} \quad \forall i \in V(G) \tag{3}$$

Vale mencionar que o método implementado em [Sag] resolve implicitamente esse modelo. Em outras palavras, ele pode ser visto como a aplicação do método de planos-de-corte à formulação acima.

Pode-se diferenciar o modelo co-convexo do anterior tanto em número de variáveis como em número de restrições. Este apresenta uma vantagem quanto ao número de variáveis, que é linear no número de vértices, enquanto no anterior tal quantidade era cúbica. Entretanto, o modelo co-convexo apresenta uma desvantagem, um número exponencial de restrições, que não podem ser fornecidas diretamente ao solver.

Para a implementação do modelo (1)-(3), usamos a seguinte estratégia. Geramos inicialmente apenas uma pequena parte das restrições, ou simplesmente a restrição $\sum_{i \in V} x_i \geq 1$. Então, usamos o CPLEX para resolução do modelo relaxado (com tal subconjunto de restrições junto com a integralidade das variáveis) e verificamos se a solução produzida é um conjunto de envoltória. Para tal, utilizamos o algoritmo de contaminação proposto anteriormente. Caso afirmativo, temos uma solução ótima. Do contrário, o subconjunto de vértices não contaminados é co-convexo e leva a uma restrição violada. Esta é acrescentada ao modelo relaxado corrente e o processo, repetido. Note que a integralidade não é relaxada em nenhum momento. Um pseudocódigo desse procedimento é apresentado no Algoritmo 3.

Algorithm 3

```

1: function RESOLUÇÃO CO-CONVEXO
2:    $C \leftarrow \emptyset$ ;
3:   repeat
4:     Adicione ao modelo a restrição (2) relativa a  $V(G) \setminus C$ ;
5:     Resolva o Modelo;
6:      $S \leftarrow$  Vértices indicados pela solução;
7:      $C \leftarrow$  Contaminação Geodésica( $S$ );
8:   until  $C = V(G)$ 
9:   return  $S$ 
10: end function

```

4. Heurística

Como os modelos matemáticos podem demorar muito para serem solucionados, muitas vezes são propostos métodos heurísticos, que, apesar de não garantirem encontrar a melhor solução possível, retornam uma solução rapidamente.

A heurística que será proposta para esse problema utiliza uma estratégia gulosa, onde busca-se sempre contaminar a maior quantidade de vértices ao adicionar um novo vértice ao conjunto S^* que, após algumas iterações, será de envoltória. Para inicializar o processo, adicionamos a

esse conjunto S^* um par de vértices que são distantes entre si o máximo possível no grafo. Então, aplicamos o algoritmo de contaminação nesse conjunto e, posteriormente, observamos se algum vértice não está infectado. Em caso afirmativo, adicionamos a S^* um vértice ainda não contaminado que possui o maior dos caminhos mínimos no grafo para algum dos vértices contaminados. Repetimos esse processo até que S^* seja um conjunto de envoltória. A seguir temos um pseudo-código para este procedimento:

Algorithm 4

```

1: function HEURÍSTICA( $G$ )
2:   Encontre  $\{i, j\} \in V(G)$ , tal que  $dist_G(i, j)$  é máxima em  $G$ 
3:    $S^* \leftarrow \{i, j\}$ 
4:    $C \leftarrow$  Contaminação Geodésica( $S^*$ )
5:   while  $C \neq V(G)$ 
6:     Encontre  $k \in V(G) - C$ ,  $l \in C$ , tal que  $dist_G(k, l)$  é máxima
7:      $S^* \leftarrow S^* \cup \{k\}$ 
8:      $C \leftarrow$  Contaminação Geodésica( $S^*$ )
9:   endwhile
10:  return  $|S^*|$ 
11: end function

```

É interessante notar que esse método pode ser implementado em $O(n^3)$, o que mantém a complexidade do algoritmo de Floyd-Warshall que é utilizado no cálculo de distâncias e já possui tal complexidade.

5. Resultados Computacionais

5.1. Instâncias estudadas

Para os testes computacionais foram utilizados grafos gerados aleatoriamente, arbitrários ou bipartidos. Note que o estudo nessa classe é relevante, uma vez que mesmo em bipartidos a determinação do número de envoltória é *NP*-difícil.

Para gerar cada instância, necessitamos de dois parâmetros de entrada: a quantidade de vértices $n \in \mathbb{N}^*$; e uma de probabilidade $0 \leq p \leq 1$ relativa à ocorrência de aresta. Para gerar grafos bipartidos, separamos o conjunto de vértices em dois subconjuntos, A com $\lceil n/2 \rceil$ vértices e B com os $\lfloor n/2 \rfloor$ restantes; em seguida, criamos uma árvore geradora qualquer com esse conjunto de vértices (mantendo a propriedade de ser bipartido), de modo a garantir a conexidade do grafo. Posteriormente, para todo par de vértices $i, j \in V(G)$, tais que $i \in A$ e $j \in B$, se a aresta ij não for uma das que formaram a árvore inicialmente, gera-se um número aleatório entre 0 e 1. Caso esse número seja menor que o fator de probabilidade p , então adicionamos a aresta ij . Note que, quanto maior esse fator p , maior será a quantidade de arestas do grafo, o que reflete diretamente na densidade do mesmo. Para gerarmos grafos aleatórios quaisquer, segue-se o mesmo procedimento anterior, sendo que formamos uma árvore geradora qualquer com os n vértices e, depois, geramos as arestas para quaisquer pares $i, j \in V(G)$ no grafo.

5.2. Comparação das Formulações e Heurística

Os modelos foram resolvidos com uso do CPLEX (IBM, Versão 12.61). A implementação dos métodos descritos acima foi toda feita na linguagem de programação C++, por meio do compilador QT-Creator, versão 3.0.1. Comparamos os desempenhos dos dois modelos matemáticos e da heurística.

As instâncias foram solucionadas e os resultados, tabelados. Nas tabelas 1 e 2, as colunas representam, em ordem, o número de vértices do grafo n , o fator de probabilidade p de geração das arestas, o tempo de resolução de cada modelo (em segundos), a solução ótima, a solução da heurística, e desvio dessa em relação à solução ótima. O sinal ”-” significa que a resolução da instância com aquele método demorou mais que 1000 segundos.

Inicialmente foram testadas instâncias de grafos bipartidos, cujos resultados são mostrados na tabela a seguir:

	V	P	Co-convexo	Passo	Solução	Sol. H	desvio
1	10	0,9	0,0059	0,4278	2	2	0
2	10	0,5	0,0040	0,9753	2	2	0
3	10	0,2	6,0538	0,0120	8	8	0
4	20	0,9	0,0302	28,9300	2	2	0
5	20	0,5	0,0175	24,5068	2	2	0
6	20	0,2	1,2705	50,6619	6	6	0
7	40	0,9	0,1840	-	2	2	0
8	40	0,5	0,1161	-	2	2	0
9	40	0,2	0,0723	-	2	3	0,5
10	80	0,9	1,7256	-	2	2	0
11	80	0,5	0,9923	-	2	2	0
12	80	0,2	0,5095	-	2	2	0
média							0,0417

Tabela 1: Resultados para grafos bipartidos aleatórios.

O modelo co-convexo foi mais rápido na maior parte das instâncias que o passo de contaminação, perdendo em apenas um dos testes. Também foi observado que a heurística proposta encontrou a solução ótima em quase todas as instâncias errando em apenas uma. Uma observação com respeito a essa instância é que o grafo possui densidade baixa, indicando uma possível condição desfavorável para a heurística. Por último, o modelo passo de contaminação demora mais que o limite de tempo de 1000 segundos para resolução com grafos a partir de 40 vértices.

Também foram feitos testes em grafos aleatórios (não necessariamente bipartidos). O resultado está a seguir:

	V	P	Co-convexo	Passo	Solução	Sol. H	desvio
1	10	0,9	0,0101	0,3823	3	3	0
2	10	0,5	0,1542	0,9743	2	2	0
3	10	0,2	5,4213	0,3281	4	4	0
4	20	0,9	0,0921	17,2908	2	2	0
5	20	0,5	0,0482	24,4920	2	2	0
6	20	0,2	2,2705	38,9281	5	5	0
7	40	0,9	0,0192	-	2	2	0
8	40	0,5	0,0821,	-	2	2	0
9	40	0,2	10,2918	-	7	7	0
10	80	0,9	1,2414	-	2	2	0
11	80	0,5	2,5721	-	2	2	0
12	80	0,2	2,7126	-	2	2	0
média							0

Tabela 2: Resultados para grafos gerais aleatórios.

Para grafos aleatórios quaisquer, houve algumas mudanças significativas com relação ao tempo. O modelo passo de contaminação teve diminuição no tempo em quase todas as instâncias com as mesmas características dos grafos bipartidos, entretanto continuou a passar do limite de tempo para grafos que possuem a partir de 40 vértices. Nota-se uma demora maior nesses grafos com baixa densidade para o modelo co-convexo. Em todos os grafos com $p = 0,2$ as instâncias

demoraram mais no modelo co-convexo, quando comparado a $p = 0,5$ e $p = 0,9$. A heurística encontrou a solução ótima em todas as instâncias, mostrando uma melhora para grafos que não são bipartidos. No final o modelo co-convexo continuou sendo mais rápido que o passo de contaminação, na maior parte das instâncias testes.

6. Considerações Finais

A partir dos métodos mostrados, observa-se que o modelo co-convexo foi mais rápido em quase todas as instâncias que o passo de contaminação, perdendo em apenas um dos testes, em cada classe de grafos. Isto sugere o modelo co-convexo como o mais adequado para encontrar o conjunto de envoltória na convexidade geodésica, pois além de encontrar soluções, em média, mais rápido que o passo de contaminação, não ultrapassou o limite de tempo proposto nos testes. Para problemas reais, onde a quantidade de vértices e arestas do grafo é da ordem de centenas ou milhares, esse modelo pode ser utilizado.

Também foi observado que a heurística proposta encontrou a solução ótima em quase todas as instâncias, errando em apenas uma, onde a resposta diferiu do ótimo em uma unidade. Sendo assim, ela se apresenta como uma boa alternativa aos modelos.

Como trabalhos futuros, há pelo menos duas grandes avenidas para pesquisa. A primeira compreende um estudo aprofundado das formulações apresentadas neste trabalho, em busca de se aplicar métodos de programação inteira mais robustos para resolução de instâncias maiores. Em particular, pode-se explorar a estrutura do grafo. Estudos preliminares indicam que se a densidade do grafo dado como entrada for muito baixa, então o modelo co-convexo torna-se consideravelmente mais lento. Uma segunda linha de pesquisa consiste em considerar os inúmeros outros parâmetros de convexidade (número de intervalo, número de Carathéodory, número de Radon, número de Helly, posto, etc.), nas diversas convexidades conhecidas na literatura (P_3 , P_3^* , monofônica, de caminhos triangulados, etc.), com a devida modelagem em programação matemática e a consequente análise de resultados nas mais diversas instâncias.

Referências

- Convexity properties of graphs on sagemath. http://doc.sagemath.org/html/en/reference/graphs/sage/graphs/convexity_properties.html. Accessed: 2018-03-27.
- Albenque, M. e Knauer, K. (2016). Convexity in partial cubes: The hull number. *Discrete Mathematics*, 339(2):866 – 876. ISSN 0012-365X.
- Araujo, J., Campos, V., Giroire, F., Nisse, N., Sampaio, L., e Soares, R. (2013). On the hull number of some graph classes. *Theoretical Computer Science*, 475(Supplement C):1 – 12. ISSN 0304-3975.
- Araujo, J., Morel, G., Sampaio, L., Soares, R., e Weber, V. (2016). Hull number: P5-free graphs and reduction rules. *Discrete Applied Mathematics*, 210:171 – 175. ISSN 0166-218X. LAGOS'13: Seventh Latin-American Algorithms, Graphs, and Optimization Symposium, Playa del Carmen, México — 2013.
- Bessy, S., Dourado, M. C., Penso, L. D., e Rautenbach, D. (2017). The geodetic hull number is hard for chordal graphs. *Electronic Notes in Discrete Mathematics*, 62:291 – 296. ISSN 1571-0653. LAGOS'17 – IX Latin and American Algorithms, Graphs and Optimization.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2009). *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition. ISBN 0262033844, 9780262033848.
- Dourado, M. C., Gimbel, J. G., Kratochvíl, J., Protti, F., e Szwarcfiter, J. L. (2009). On the computation of the hull number of a graph. *Discrete Mathematics*, 309(18):5668 – 5674. ISSN 0012-365X. Combinatorics 2006, A Meeting in Celebration of Pavol Hell's 60th Birthday (May 1–5, 2006).

- Dourado, M. C., Penso, L. D., Rautenbach, D., e Szwarcfiter, J. L. (2012). Reversible iterative graph processes. *Theoretical Computer Science*, 460:16 – 25. ISSN 0304-3975.
- Everett, M. G. e Seidman, S. B. (1985). The hull number of a graph. *Discrete Mathematics*, 57(3): 217 – 223. ISSN 0012-365X.
- Floyd, R. W. (1962). Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345–. ISSN 0001-0782.
- Garey, M. R. e Johnson, D. S. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA. ISBN 0716710455.
- Marcilon, T. B. (2017). *Resultados no Tempo Máximo e no Número de Envoltória nas Convexidades P3 e Geodésica*. PhD thesis, Universidade Federal do Ceará.
- Pelayo, I. (2013). *Geodesic Convexity in Graphs*. SpringerBriefs in Mathematics. Springer New York. ISBN 9781461486992.
- Penso, L. D., Protti, F., Rautenbach, D., e dos Santos Souza, U. (2015). Complexity analysis of p3-convexity problems on bounded-degree and planar graphs. *Theoretical Computer Science*, 607:83 – 95. ISSN 0304-3975. Algorithmic Aspects in Information and Management.
- Warshall, S. (1962). A theorem on boolean matrices. *J. ACM*, 9(1):11–12. ISSN 0004-5411.
- West, D. B. (2000). *Introduction to Graph Theory*. Prentice Hall, 2 edition. ISBN 0130144002.