SBM

Matemática Contemporânea, Vol. 45, 11–21

Geração de Obstruções Minimais de Grafos-$(2,1)$

Matheus S. D'Andrea Alves **Loana Tito Nogueira**
Raquel S. F. Bravo **Uéverton S. Souza**

Resumo

Um grafo é dito um grafo-(k,ℓ) se o seu conjunto de vértices pode ser particionado em k conjuntos independentes e ℓ cliques. Reconhecer se um dado grafo G é um grafo-(k,ℓ) é um problema NP-completo para todo par k,ℓ tal que $\max\{k,\ell\} \geq 3$. Por outro lado, reconhecer se um dado G é um grafo-$(2,1)$ é uma tarefa executável em tempo polinomial. Embora o reconhecimento de grafos-$(2,1)$ seja polinomial, pouco se sabe sobre as características da família dos subgrafos proibidos dos grafos-$(2,1)$, também conhecida como a família das obstruções minimais de grafos-$(2,1)$. Neste trabalho, demonstramos algumas propriedades inerentes às obstruções de grafos-$(2,1)$, e em seguida as utilizamos para desenvolver um algoritmo eficiente de enumeração das obstruções minimais de grafos-$(2,1)$ com até nove vértices, facilmente extensível para grafos de ordem maior.

2000 AMS Subject Classification: 05C85, 05C75, 05C30.

Key Words and Phrases: Grafos-(k,ℓ), subgrafos proibidos, obstruções minimais, grafos-(2,1).

Este trabalho possui auxílio dos seguintes fomentos de pesquisa: CNPq, CAPES, and FAPERJ.

1 Introdução

Uma das principais motivações do estudo de classes de grafos é o fato de que diversos problemas, que são difíceis para grafos em geral, tornam-se tratáveis quando restritos a classes especiais de grafos. Assim, busca-se delimitar a partir de que ponto um determinado problema pode ser resolvido de forma eficiente. Particularmente, o problema de particionamento em grafos tem despertado muito interesse devido às pesquisas de grafos perfeitos [4] e também pela procura de algoritmos eficientes para o reconhecimento de determinadas classes de grafos. O problema de partição de grafos pode ser descrito como tendo por objetivo particionar o conjunto dos vértices de um grafo em subconjuntos $V_1, V_2, \ldots, V_k$ onde $V_1 \cup V_2 \cup \ldots \cup V_k = V$ e $V_i \cap V_j = \emptyset$, $i \neq j$, $1 \leq i \leq k$ e $1 \leq j \leq k$, exigindo-se, porém, algumas propriedades sobre estes subconjuntos de vértices. Estas propriedades podem ser *internas*, como por exemplo exigir que os vértices de cada subconjunto V_i sejam dois-a-dois adjacentes (isto é, V_i seja uma clique) ou dois-a-dois não-adjacentes (isto é, V_i seja um conjunto independente), ou *externas*, onde as exigências são feitas sobre os pares de $V_i \times V_j$, podem ser adjacentes ou não-adjacentes entre si. Brandstädt [1] definiu uma classe de grafos, a classe dos *grafos*-(k, ℓ) como sendo aquela formada pelos grafos cujo conjunto de vértices pode ser particionado em k conjuntos independentes e ℓ cliques. Brandstädt [1, 2] considerou em particular as classes de grafos-$(2, 1)$, grafos-$(1, 2)$ e grafos-$(2, 2)$, apresentando algoritmos polinomiais para reconhecê-las. Por outro lado, sabe-se que reconhecer grafos-(k, ℓ) para $k \geq 3$ ou $\ell \geq 3$ é NP-Completo [1]. Como exemplo, podemos destacar a classe dos grafos-$(k, 0)$, que corresponde ao problema de reconhecer se um dado grafo é k-colorível (seu reconhecimento é NP-Completo para $k \geq 3$).

Embora o reconhecimento de grafos-$(2, 1)$ seja polinomial, pouco se sabe sobre as características da família das obstruções minimais de grafos-$(2, 1)$. Sendo assim, o objetivo deste trabalho é demonstrar algumas propriedades inerentes às obstruções de grafos-$(2, 1)$, e em seguida fazer uso destas para

desenvolver um algoritmo eficiente de enumeração das obstruções minimais de grafos-$(2,1)$ com até nove vértices, facilmente extensível para grafos de ordem maior.

Uma versão preliminar deste trabalho foi apresentada em [5].

2 Preliminares

Dado um grafo simples $G = (V, E)$, denotamos por $\overline{G}$ o complemento de G. Para $V' \subseteq V$, denotamos por $G[V']$ o subgrafo de G induzido por V'. Uma *clique* (*conjunto independente*) é um subconjunto de vértices que induz um subgrafo completo (sem arestas), não necessariamente maximal e denotada por K_p (I_p) uma clique (conjunto independente) de p vértices.

Um grafo G é um *grafo-*(k, ℓ) se o conjunto de vértices V pode ser partiticionado em k conjuntos independentes e ℓ cliques.

Uma obstrução de um grafo G é um subgrafo $G' \subseteq G$ o qual impede que G tenha certa propriedade.

Um conjunto S é dito **minimal** em relação a uma determinada propriedade Π se S satisfaz Π, e não existe $S' \subset S$ que satisfaça Π. Desta forma, uma obstrução minimal de um grafo-$(2,1)$ é qualquer grafo G minimal com relação a propriedade de *não ser um grafo-*$(2,1)$. Ou seja, G é um grafo que não é $(2,1)$ e qualquer subgrafo induzido de G é $(2,1)$.

3 Compreensão Geral

Uma parte crucial para o desenvolvimento do algoritmo que será apresentado aqui é a observação de que toda obstrução minimal de um grafo-$(2,1)$ é necessariamente um grafo-$(2,2)$ e um grafo-$(3,1)$ simultaneamente. Um lema trivialmente extraído desse fato é o de que toda obstrução minimal de um grafo-$(2,1)$ contém dois conjuntos independentes, uma clique e um conjunto com um único vértice.

O teorema e sua prova são dados a seguir.

Teorema 1. *Toda obstrução minimal de um grafo-*$(2,1)$ *é necessariamente um grafo-*$(2,2)$ *e um grafo-*$(3,1)$ *simultaneamente, onde uma das partições é composta por um único vértice.*

Demonstração. Seja G uma obstrução minimal de grafo-$(2,1)$. Isso significa que existe um vértice $v \in V(G)$ tal que a sua remoção torna G um grafo-$(2,1)$. Suponha G' como um subgrafo induzido por $V(G) \setminus \{v\}$. Pela definição de minimal, podemos garantir que o grafo G' pode ser particionado em dois conjuntos independentes e uma clique, pois G é minimal. Portanto, $\{v\}$ é simultaneamente um conjunto independente e uma clique, logo G é um grafo-$(2,2)$ e G é um grafo-$(3,1)$. ■

Teorema 1 nos permite construir obstruções minimais de grafos-$(2,1)$ da seguinte forma: primeiramente constrói-se um grafo G' que é $(2,1)$; em seguida, gera-se uma obstrução G a partir da adição de um vértice especial v em G', que deverá ser ligado a alguns vértices já existentes de forma apropriada. Notem que nossa proposta não é aplicar a ingênua abordagem de a partir de um dado n implementar um algoritmo exaustivo que enumera todos os grafos não isomorfos de tamanho máximo n, e em seguida testar quais destes são obstruções minimais. Em nossa abordagem poucas ligações (em especial as referentes ao último vértice) serão testadas por força bruta. Para tal, e com o objetivo de reduzir o tempo de execução e o número de grafos isomorfos a serem testados, faremos uso de características intrínsecas aos grafos-$(2,1)$.

4 Características intrínsecas a um grafo-$(2,1)$

Lema 1 nos mostra que obstruções de grafos-$(2,1)$ só existem a partir de 6 vértices. Também é possível mostrar que todo grafo-$(2,1)$ utilizado para gerar uma obstrução contém o subgrafo $2K_2$ (esqueleto #0 da Figura 1).

Lema 1. *[6] Não existem obstruções dos grafos-*$(2,1)$ *com menos de 6 vértices.*

Demonstração. Um grafo G é (2,1) se, e somente se, o mesmo contém uma clique C que intersecta todo ciclo ímpar de G. Portanto, basta demonstrar que nenhum grafo com até 5 vértices possui ciclos ímpares disjuntos. Como o menor ciclo ímpar formado em um grafo simples sem laço é um ciclo de tamanho 3 (C_3), em um grafo com no máximo 5 vértices ao se formar um C_3 sobram apenas 2 vértices ainda disjuntos desse C_3, tornando assim impossível a construção de outro ciclo ímpar disjunto do primeiro. Logo, como C_3 é uma clique, não existem obstruções de grafos-$(2,1)$ com menos de 6 vértices. ■

Lema 2. *Se G é obstrução minimal de grafos-$(2,1)$ então:*

(a) Para qualquer $v \in V(G)$, o conjunto de vértices de $G' = G[V \setminus \{v\}]$ pode ser particionado em dois conjuntos independentes, S_1, S_2, e uma clique, C, onde $|V(C)| \geq 2$ e existe uma aresta entre algum vértice de S_1 e algum vértice de S_2.

(b) G não contém vértices de grau menor ou igual a 1.

Demonstração. (a) Suponha que não existe aresta entre S_1 e S_2. Então, G' é um grafo-$(1,1)$ e portanto G é um grafo-$(2,1)$ - contradição. Além disso, se $|V(C)| = 1$ temos os seguintes casos: (i) se $u \in V(C)$ possui grau zero em G' então G' é bipartido e G é um grafo-$(2,1)$ - contradição; (ii) se $u \in V(C)$ possui grau maior ou igual a um então podemos inserir um vértice $w \in N(u)$ $(w \neq v)$ em C mantendo alguma aresta entre os conjuntos independentes, caso contrário G seria um grafo-$(2,1)$. (b) Seja G uma obstrução minimal e $w \in V(G)$ um vértice de grau menor ou igual a um. Por questão de minimalidade, $G[V \setminus \{w\}]$ é um grafo-$(2,1)$. Neste caso, w poderá ser adicionado ou a S_1 ou a S_2 o que implica em G ser $(2,1)$, uma contradição. ■

O primeiro passo do nosso algoritmo baseia-se em gerar, a partir do grafo $2K_2$, grafos chamados de esqueletos internos, os quais são todos os subgrafos não isomorfos obtidos de todas as possíveis combinações de adição de arestas no subgrafo $2K_2$ inicial. A Figura 2 ilustra todos os oito possíveis esqueletos internos de G'.

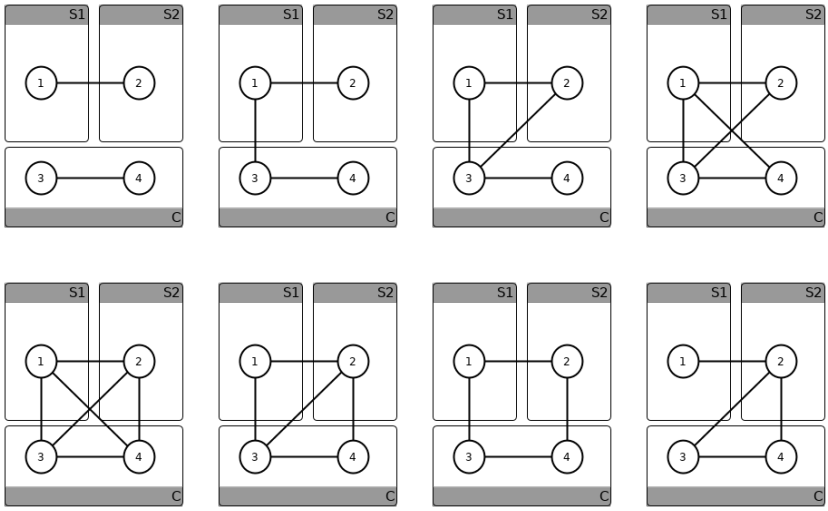

Figura 1: Esqueletos #0, ..., #7, respectivamente.

5 Obtenção da obstrução

Note que, se desejamos construir uma obstrução com até n vértices, sabemos agora a disposição de cinco desses vértices. O vértice v, que quando adicionado será responsável pela obstrução, e os vértices pertencentes ao $2K_2$ do esqueleto descrito no Lema 2. Portanto, existem ainda $n-5$ vértices que não conhecemos suas disposições em G, onde para determiná-las geramos todos os grafos não-isomorfos com os vértices restantes, os quais nomeamos de *Externos*.

Supondo $n = 9$, temos pelo Teorema 1 e pelo Lema 2 a disposição de 5 vértices. Seja $n' = n - 5$ o número de vértices ainda sem disposição, formamos os *Externos* como o grupo com todos os grafos não-isomorfos com $n' = 4$ vértices.

Cada instância formada por um *Externo*, v e $2K_2$ será então explorada pelo algoritmo em busca de uma obstrução. A Figura 3, representa uma possível configuração inicial com $n = 9$.

O próximo passo é adicionar os vértices do grafo *Externo* (veja Figura 2a)

ao esqueleto interno de todas as formas possíveis mantendo o esqueleto como um grafo-$(2,1)$. Para isso, cada vértice de *Externo* é adicionado à uma das partições já existentes do grafo-(S_1, S_2, C), respeitando a estrutura do Externo e as regras da partição, onde o mesmo será incluído. Por exemplo, se um vértice u pertencente ao grupo de vértice de *Externo* tenta ser incluído em um dos conjuntos independentes (S_1, S_2), porém um de seus vizinhos em Externo já está incluído no mesmo conjunto independente, então não é permitido que u faça parte desse conjunto, pois assim ele deixaria de ser um conjunto independente. A Figura 2 ilustra essa etapa.

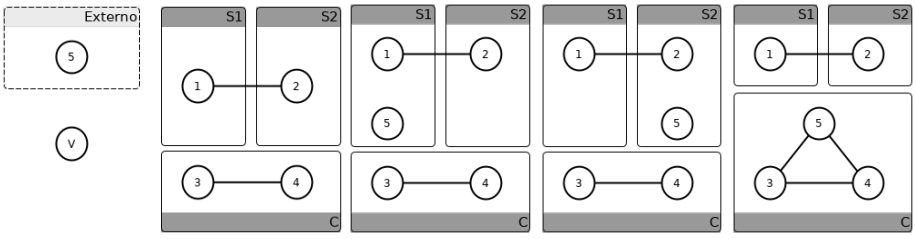

Figura 2: (a) Uma possível configuração inicial; (b) possíveis esqueletos com a adição do vértice do grafo *Externo*, esqueletos #0, #1 e #2, respectivamente.

Agora, usaremos o terceiro grafo da Figura 2b (*novo esqueleto* #2) como continuação do nosso exemplo. O próximo passo é a enumeração de todas as possibilidades de adição de arestas dos vértices $u \in V(Externo)$ para os outros vértices do esqueleto interno sem quebrar a característica da partição onde ele está inserido. Por Exemplo, um vértice $u \in S_1$ não pode adicionar uma aresta para um vértice $w \in S_1$, aos grafos obtidos após a adição dessas arestas daremos o nome de *compostos*. Nos compostos resultantes removemos vértices com grau igual a 0, e se após tais remoções o esqueleto composto encontra-se com menos de cinco vértices, então o descartamos. Tal operação é segura devido ao Lema 2. A Figura 3 ilustra

essa etapa.

Agora consideramos o grafo #0 da Figura 3. A partir dos compostos obtidos acima, adicionamos o vértice v e enumeramos todas as possíveis combinações de adições de arestas de v ao composto. Os grafos gerados por esse passo foram chamados de *candidatos*. Com os candidatos com n vértices gerados, aplicamos o algoritmo de Brandstädt [1, 2] em cada um deles obtendo aqueles que são obstruções, e em seguida testamos a condição de minimalidade.

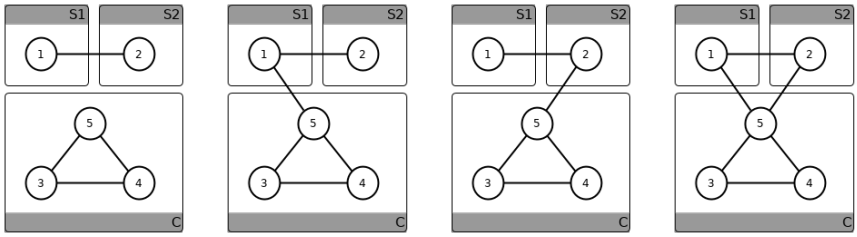

Figura 3: Compostos #0, #1, #2 e #3 obtidos a partir do grafo #2 da Fig. 2b.

Teorema 2. *O algoritmo descrito gera todas as obstruções minimais.*

Demonstração. Pelo Teorema 1, sabemos que todas as obstruções minimais de grafos-$(2,1)$ são simultaneamente um grafo-$(2,2)$ e um grafo-$(3,1)$ onde uma das partições contém apenas um vértice. Seja v esse vértice de G. Sabemos também que $G' = G[V \setminus \{v\}]$ é um grafo-$(2,1)$. Nosso algoritmo é composto, basicamente, pelos seguintes passos: (i) enumerar todos os possíveis grafos G'; (ii) para cada G' adicionar o vértice v; (iii) testar todas as possíveis adições de arestas para v, verificando quais geram uma obstrução minimal.

Portanto para a corretude do algoritmo basta mostrar que todos os possíveis grafos G' são, de fato, gerados pelo nosso algoritmo. Os esqueletos internos $\#0, \ldots, \#7$ são todos os possíveis subgrafos não isomorfos induzidos pelos quatro vértices do subgrafo $2K_2$ inicial. Como toda obstrução

contém como subgrafo induzido um desses esqueletos, nosso algoritmo tem como propriedade chave enumerar todas as obstruções que podem ser obtidas a partir da fixação de cada um desses esqueletos. Neste ponto, falta analisar os vértices de $G'[V(G') \setminus \{1,2,3,4\}]$. Para $|V(G')| = 8$ temos 4 vértices em $G'[V(G') \setminus \{1,2,3,4\}]$, dando-nos 12 possíveis grafos não isomorfos que denominamos externos. Combinando esses 12 possíveis grafos, os 8 esqueletos, e as combinações de arestas entre eles obtemos todos os grafos G' possíveis. Note que, se algum vértice fica isolado, o mesmo é removido antes da inserção de v e portanto as obstruções com $6, 7$ e 8 vértices também são geradas. ■

6 Considerações Finais

Neste trabalho demonstramos algumas propriedades de obstruções de grafos-$(2,1)$, e em seguida fizemos uso destas para obter um algoritmo competitivo de enumeração das obstruções minimais de grafos-$(2,1)$ com até nove vértices, facilmente extensível para grafos de ordem maior. Para ilustrar quão úteis e promissoras são as etapas definidas para o nosso algoritmo, primeiramente observe que existem 2^{36} grafos distintos com até 9 vértices (vértices rotulados), e portanto a enumeração de cada um deles seguida de uma posterior filtragem de casos isomorfos e uma verificação de obstrução minimal de $(2,1)$, certamente não nos fornece um método executável na prática. Nossa abordagem visa, portanto, eliminar uma grande gama de casos isomorfos em nossa análise. Isso é possível explorando decisões que podem ser facilmente tomadas, uma vez que propriedades intrínsecas às obstruções de grafos-$(2,1)$ são conhecidas.

Removendo duplicadas (indivíduos indênticos), usando esqueletos externos de tamanho exatamente quatro, e eliminando vértices de grau zero nos candidatos, foram gerados apenas 22mil grafos em 94min sumarizando 8mb de dados. Para ilustrar quão impactantes são as decisões tomadas no projeto de nosso algoritmo, uma versão onde não era considerada a remoção de vértices de grau zero nos candidatos e considerava-se esqueletos

externos de tamanho entre um e quatro vértices, gerou 8gb de dados em um espaço de 6h.

Após a obtenção das 22mil obstruções minimais, uma segunda geração foi obtida a partir da eliminação de casos isomorfos, resultando em 1026 obstruções minimais não isomorfas de grafos-$(2,1)$ em 80min num arquivo final de 500Kb. Toda implementação dos algoritmos foi feita na Linguagem Java, utilizando o framework JGraphT e em um segundo momento em javascript com o framework vis.js para a geração da visualização do resultado. Para eliminação dos casos isomorfos implementamos o algoritmo VF2 de Cordella et al. [3]. Em nossos experimentos foi utilizado uma máquina Lenovo e431, com Intel core i5 e 4gb de memória RAM. As obstruções geradas podem ser visualizadas e acessadas em:
`http://www2.ic.uff.br/~ueverton/obstrucao21/obstrucoes_html/obstrucoes.html`;
`http://www2.ic.uff.br/~ueverton/obstrucao21/Apendice.pdf`.

Referências

[1] **Bransdstädt, A.** Partitions of graphs into one or two independent sets and cliques. *Discrete Mathematics* 152 (1996) 47 – 54.

[2] **Bransdstädt, A.** The complexity of some problems related to graph 3-colorability. *Discrete Applied Mathematics* 89 (1998) 59 – 73.

[3] **Cordella, L. P., Foggia P., Sansone C., Vento M.** A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.10 (2004) 1367–1372.

[4] **Golumbic, M. C.** *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, New York, 1980.

[5] **Alves, M. D., Bravo, R., Tito, L., Souza, U. S.** Geração de Obstruções Minimais de Grafos-(2,1). *XLVIII Simpósio Brasileiro de Pesquisa Operacional*, 2016, Vitória.

[6] **Silva, J. S.** Obstruções minimais de grafos-$(2,1)$. *Dissertação de mestrado*, Universidade Federal Fluminense (2011).

Matheus S. D'Andrea Alves
UFF, Niterói, Brasil.
mad@id.uff.br

Raquel Bravo, Loana Tito Nogueira,
Uéverton S. Souza
UFF, Niterói, Brasil.
raquel@ic.uff.br,
loana@ic.uff.br,
ueverton@ic.uff.br