

GERAÇÃO DE BICLIQUES DE UM GRAFO

Vânia Maria Félix Dias^{1*}, Celina M. H. de Figueiredo², Jayme L. Szwarcfiter³

¹Universidade Federal do Rio de Janeiro, COPPE,
Caixa Postal 68530, 21945-970 Rio de Janeiro, Brasil.

²Universidade Federal do Rio de Janeiro, IM e COPPE,
Caixa Postal 68530, 21945-970 Rio de Janeiro, Brasil.

³Universidade Federal do Rio de Janeiro, IM, COPPE, e NCE,
Caixa Postal 68511, 21945-970 Rio de Janeiro, Brasil.

vm.felix@uol.com.br, celina@cos.ufrj.br, jayme@nce.ufrj.br

Abstract. *This work describes a study on the generation of bicliques of a graph. We show that it is NP-complete to test whether a subset of the vertices of a graph is part of a biclique. We also show that there is no polynomial-time delay algorithm for generating all bicliques in reverse lexicographic order, unless $P = NP$. On the other hand, we describe different polynomial-time delay algorithms for the generation of bicliques of a graph. We present an algorithm that generates all bicliques of a graph in lexicographic order. We also describe an algorithm that generates all non-induced bicliques of a graph. In addition, we propose specialized efficient algorithms for generating the bicliques of special classes of graphs.*

Resumo. *Este trabalho consiste de um estudo sobre a geração de bicliques de um grafo. Demonstramos que é NP-completo decidir se um subconjunto de vértices de um grafo é parte de uma biclique. Demonstramos também que não existe algoritmo com atraso polinomial para a geração de todas as bicliques em ordem lexicográfica reversa, a menos que $P = NP$. Por outro lado, descrevemos diferentes algoritmos, todos com atraso polinomial, para a geração das bicliques de um grafo. Apresentamos um algoritmo que gera todas as bicliques em ordem lexicográfica. Também descrevemos um algoritmo que gera todas as bicliques não induzidas de um grafo. Além disso, propomos algoritmos eficientes para a geração de bicliques de classes especiais de grafos.*

1. Introdução

1.1. Definições e Notações

Seja $G = (V, E)$ um grafo. Uma *clique* é um subconjunto de vértices mutuamente adjacentes, e um *conjunto independente* é um subconjunto de vértices mutuamente não adjacentes. Sejam $X, Y \subseteq V$ tais que $X \cap Y = \emptyset$. Dizemos que $B = X \cup Y$ é um *conjunto*

*Doutorado com bolsa do CNPq.

bipartido completo de G , quando ambos X e Y são independentes e todo vértice de X é adjacente a todo vértice de Y . Isto é, B induz um subgrafo bipartido completo em G . Dizemos que X e Y são *partes* do conjunto bipartido completo B . Se $X, Y \neq \emptyset$, então B é um *conjunto bipartido completo próprio*, i.e., B induz um subgrafo bipartido completo contendo pelo menos uma aresta de G . Caso contrário, B é chamado *conjunto bipartido completo degenerado*. Por outro lado, dizemos que $B = X \cup Y$ é um *conjunto bipartido completo não induzido* de G , se $X \cap Y = \emptyset$, e todo vértice de X é adjacente a todo vértice de Y . Dizemos que B é uma *biclique induzida* - ou simplesmente uma *biclique* - de G , se B é um conjunto bipartido completo próprio (induzido) de G e é maximal em relação a esta propriedade. Se um conjunto bipartido completo não induzido $B = X \cup Y$ é próprio e maximal, então dizemos que B é uma *biclique não induzida* de G . Observe que se G é um grafo bipartido, então toda biclique não induzida é também biclique (induzida) de G .

A Figura 1 mostra uma biclique e uma biclique não induzida. Note que $B = \{1, 4, 5\} \cup \{2\}$ é um conjunto bipartido completo próprio (induzido) maximal do grafo G , isto é, uma biclique. Por outro lado, $B = \{1, 4, 5\} \cup \{2\}$ está contido em $B' = \{1, 3, 4, 5\} \cup \{2\}$, um conjunto bipartido completo próprio (não induzido) maximal, isto é, uma biclique não induzida de G .

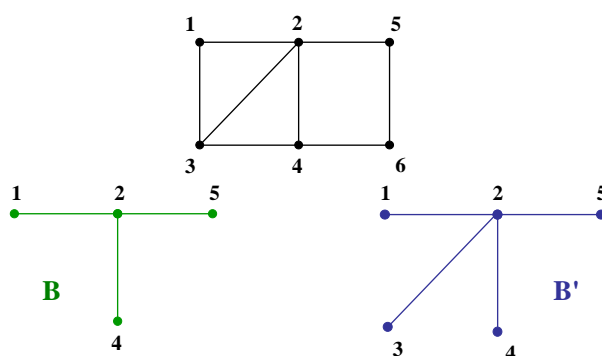


Figure 1. Biclique B induzida e biclique B' não induzida

Denotamos por \mathcal{B} o conjunto de bicliques de G . Denotamos por n e m , respectivamente, o número de vértices e de arestas de G .

1.2. Motivação

A geração de objetos de uma coleção que satisfazem determinadas propriedades é um problema bastante estudado na literatura de Algoritmos, Combinatória e Teoria dos Grafos [Byskov, 2003, D.Eppstein, 2005, Johnson et al., 1988, Makino and Uno, 2004, Tomita et al., 2004, Tsukiyama et al., 1997].

Conhecer as bicliques de um grafo é útil na solução de problemas em áreas como linguagens formais e autômatos, ordens parciais, inteligência artificial e redes de computadores [Prisner, 2000]. Além dessas, uma área que naturalmente compreende bastante

aplicações é a própria teoria dos grafos. Bicliques são empregadas na caracterização de certas classes de grafos, como grafos bipartidos cordais [Golombic and Goss, 1978]. A enumeração das bicliques de um dado grafo constitui um procedimento auxiliar no reconhecimento de certas classes de grafos [Prisner, 1997]. Resultados de limites superiores para o número de bicliques de um grafo foram estabelecidos por Prisner [Prisner, 2000]. No caso geral, o número de bicliques de um grafo é no máximo $n^{5/2}(1.618034)^n + O(1)$. Para um grafo da classe dos grafos bipartidos é demonstrado que o número máximo de bicliques de é $2^{n/2}$.

[Alexe et al., 2004] mostraram como um algoritmo para gerar todos os conjuntos independentes maximais de um grafo pode ser usado para gerar todas as bicliques não induzidas de um grafo. Além disso, descreveram um algoritmo específico para enumeração das bicliques não induzidas de um grafo, com resultados de tempo empiricamente melhores do que o obtido pela transformação descrita. Contudo, este último não representou uma melhoria da complexidade analítica. Além disso, o mesmo requer espaço exponencial. [Kloks and Kratsch, 1995] descreveram um algoritmo para listar todas as bicliques de um grafo bipartido cordal em tempo polinomial, enquanto que Eppstein [Eppstein, 1994] descreveu um algoritmo para gerar as bicliques não induzidas de grafos de arboricidade limitada. Em ambos os casos, os grafos pertencem a classes com número de bicliques polinomial em n , o número de vértices do grafo de entrada.

1.3. Publicações relativas à Tese

Durante o curso de doutorado produzimos os seguintes trabalhos: *On the generation of bicliques of a graph* [Dias et al., 2004a]; *Generating bicliques of a graph in lexicographic order* [Dias et al., 2005]; *Generating all non-induced bicliques of a graph* [Dias et al., 2004b]; *The stable marriage problem with restricted pairs* [Dias et al., 2003]; *Stable matchings with restricted pairs* [Dias et al., 2001]; *Casamentos estáveis com casais forçados e casais proibidos* [Dias and Szwarcfiter, 2000];

Observamos que dois artigos foram aceitos para publicação na revista *Theoretical Computer Science*, e que um artigo está submetido para publicação na revista *Discrete Applied Mathematics*. Estas duas revistas estão classificadas como periódicos qualis A pela CAPES.

2. Problemas de bicliques NP-completos

Nesta seção, consideramos dois problemas de decisão: PARTE DE BICLIQUE e MAIOR BICLIQUE LEXICOGRÁFICA. Os teoremas 1 e 2 estabelecem que ambos são NP-completos. Nos dois casos, através de uma transformação polinomial do bem conhecido problema de SATISFATIBILIDADE.

Dado um grafo $G = (V, E)$, dizemos que $S \subset V$ é parte de uma biclique $B = X \cup Y$ se $S = X$ ou $S = Y$. A seguir, definimos o problema de decisão relacionado.

PARTE DE BICLIQUE

Entrada: Grafo $G = (V, E)$, subconjunto $S \subset V$.

Pergunta: Existe biclique de G com parte S ?

Teorema 1 *Dado um grafo $G = (V, E)$ e um subconjunto $S \subset V$, é NP-completo decidir se S é parte de uma biclique.■*

Sejam S e T dois subconjuntos de um conjunto ordenado. Diz-se que S é *lexicograficamente menor* que T , se ocorre uma das seguintes situações: (i) seja j a primeira posição em que S e T são discordantes, neste caso $s_j < t_j$; ou (ii) os primeiros $|S|$ elementos de T coincidem com os elementos de S e $|S| < |T|$. O problema de decisão relacionado é descrito abaixo.

MAIOR BICLIQUE LEXICOGRÁFICA

Entrada: Grafo $G = (V, E)$, biclique B e uma ordem total nos vértices de V .

Pergunta: B é a maior biclique lexicográfica de G ?

Teorema 2 *Dado um grafo $G = (V, E)$, uma biclique B , e uma ordem sobre V , é Co-NP-completo decidir se B é a maior biclique lexicográfica de G .■*

Estes problemas são relevantes quando consideramos algoritmos de geração de bicliques. Em particular, a NP-completude do problema MAIOR BICLIQUE LEXICOGRÁFICA implica a inexistência de um algoritmo polinomial para determinar a maior biclique lexicográfica de um grafo, a menos que $P = NP$. Consequentemente, não existe um algoritmo com atraso polinomial para gerar as bicliques de um grafo em ordem lexicográfica reversa.

3. Geração de bicliques

Os algoritmos apresentados nesta seção, surpreendentemente, contrastam bastante com os resultados apresentados na seção anterior. Descrevemos a seguir um algoritmo que dado um grafo G e um conjunto bipartido completo B - próprio ou não - fornece em tempo polinomial a menor biclique lexicográfica de G contendo B . Mostramos ainda como este algoritmo pode ser usado para encontrar a menor biclique lexicográfica de G . Em uma das principais contribuições desta tese fornecemos um algoritmo que gera as bicliques de um grafo em ordem lexicográfica, que requer tempo $O(n^3)$ entre a enumeração de duas bicliques consecutivas em sua saída. Até o momento não era conhecido nenhum algoritmo eficiente para a geração de bicliques (induzidas) em um grafo qualquer.

3.1. Algoritmo para gerar a menor biclique lexicográfica

Seja $G = (V, E)$ um grafo, cujos vértices têm uma ordenação total em $V = \{1, \dots, n\}$. Dado um conjunto bipartido completo $B = X \cup Y$, com $X \neq \emptyset$, o algoritmo MBLB retorna a menor biclique lexicográfica B' de G que contém B , ou $B' = \emptyset$, caso não

exista biclique de G contendo B . Representamos por B^* a menor biclique lexicográfica de G . A primeira fase do algoritmo MBLB encontra o menor conjunto bipartido completo próprio B' contendo B , caso exista. Caso contrário, o algoritmo retorna $B' = \emptyset$. Na segunda fase, o algoritmo estende B' a um conjunto bipartido completo próprio maximal de G , isto é, uma biclique de G . Nas duas fases, os vértices são considerados em ordem crescente, o que garante que B' seja a menor biclique lexicográfica contendo B . A fim de encontrar B^* , a menor biclique lexicográfica de G , basta aplicar o algoritmo MBLB com $B = X \cup Y$, $X = \{k\}$ e $Y = \emptyset$, onde k é o menor vértice não isolado de G .

A complexidade de tempo do algoritmo MBLB é $O(n^2)$. É fácil ver que qualquer operação descrita no algoritmo, tanto na primeira como na segunda fase, pode ser realizada em tempo $O(n)$ para cada vértice examinado. Como cada vértice é considerado no máximo uma vez, o algoritmo pode ser implementado em tempo $O(n^2)$.

3.2. Geração de bicliques em ordem lexicográfica

Sejam B uma biclique de G e $B_j = B \cap \{1, \dots, j\}$. Dizemos que B_j é *fracamente maximal* se não existe biclique $B' \neq B$ tal que B' contém $B_j \cup \{\ell\}$, para algum $\ell \in \{1, \dots, j\} \setminus B_j$. Estabelecemos a seguinte caracterização: B é a menor biclique lexicográfica de G se e somente se B_j é fracamente maximal, para todo $j \in \{1, \dots, n\}$.

O algoritmo descrito a seguir, tendo como entrada um grafo $G = (V, E)$ e uma ordenação total sobre os vértices de V , enumera as bicliques de G em ordem lexicográfica.

Algoritmo GBOL

Entrada: Grafo $G = (V, E)$, ordenação sobre V

Saída: Enumeração de todas as bicliques de G em ordem lexicográfica

Encontre a menor biclique B^* de G

$Q \leftarrow \emptyset$

Inclua B^* na fila Q

Enquanto $Q \neq \emptyset$ faça

Retire a menor biclique lexicográfica $B = X \cup Y$ de Q

Enumerar B

Para cada vértice $j \in V \setminus B$ faça

$X_j \leftarrow X \cap \{1, \dots, j\}$

$Y_j \leftarrow Y \cap \{1, \dots, j\}$

Repita duas vezes

Se $X_j \cap N_j \neq \emptyset$ ou $Y_j \cap \overline{N}_j \neq \emptyset$ então

$X'_j \leftarrow (X_j \setminus N_j) \cup \{j\}$

$Y'_j \leftarrow Y_j \setminus \overline{N}_j$

Se $X'_j \cup Y'_j$ é fracamente maximal então

Aplique MBLB e encontre a biclique B' contendo $X'_j \cup Y'_j$

Se $B' \neq \emptyset$ e $B' \notin Q$ então inclua B' em Q

Troque os conteúdos de X_j e Y_j

Teorema 3 Dado um grafo $G = (V, E)$, o algoritmo GBOL enumera todas as bicliques de G em ordem lexicográfica.

Idéia da Prova: O algoritmo GBOL usa o algoritmo MBLB para encontrar B^* . Seja B' uma biclique incluída em Q dentro do laço

Enquanto $Q \neq \emptyset$ faça

na iteração j após uma biclique $B = X \cup Y$ ser listada. Nesse caso, B' é a menor biclique lexicográfica contendo $X'_j \cup Y'_j$, onde $X'_j = (X_j \setminus N_j) \cup \{j\}$ e $Y'_j = Y_j \setminus \overline{N}_j$, para algum $j \in V \setminus B$ que satisfaz $X_j \cap N_j \neq \emptyset$ ou $Y_j \cap \overline{N}_j \neq \emptyset$. Como B'_j é fracamente maximal, concluímos que $B'_j = X'_j \cup Y'_j$. Como $X_j \cap N_j \neq \emptyset$ ou $Y_j \cap \overline{N}_j \neq \emptyset$, a primeira posição $i < j$ na qual as bicliques B e B' são discordantes satisfaz $i \in B \setminus B'$. Consequentemente, B' é lexicograficamente maior que B . Além disso, uma biclique é listada quando é a menor lexicográfica em Q . Logo, as bicliques são listadas em ordem lexicográfica estritamente crescente. Sendo assim, nenhuma biclique é enumerada duas vezes. A prova de que todas as bicliques são listadas é feita por indução na extensão da lista de bicliques enumeradas. A primeira biclique listada pelo algoritmo é B^* . No caso geral, seja $B = X \cup Y$, $B \neq B^*$, a próxima biclique na ordenação lexicográfica das bicliques de G . Demonstramos que $B \in Q$ nesta ocasião e que, nesse caso, B será escolhida pelo algoritmo como a próxima a ser listada, o que conclui a prova. A idéia central é identificar uma biclique \tilde{B} a partir da qual B é incluída em Q , caso $B \notin Q$. ■

O Teorema 4 mostra que o algoritmo GBOL dispende tempo polinomial entre a saída de duas bicliques consecutivas na ordenação lexicográfica.

Teorema 4 O algoritmo GBOL enumera as bicliques de G com tempo de atraso polinomial $O(n^3)$.

Idéia da prova: Ao listar uma biclique B , o algoritmo GBOL realiza $O(n)$ tentativas de gerar bicliques contendo B_j , através de chamadas sucessivas do algoritmo MBLB. Como uma execução do algoritmo MBLB requer tempo $O(n^2)$, a diferença de tempo entre as saídas de duas bicliques de G listadas consecutivamente pelo algoritmo GBOL é $O(n^3)$. O mesmo limite de tempo se aplica tanto para gerar a menor biclique lexicográfica como após listar a maior biclique lexicográfica de G . Portanto, as bicliques são geradas com atraso polinomial $O(n^3)$. ■

4. Geração de bicliques não induzidas

Seja $G = (U \cup W, E)$ um grafo bipartido. Então toda biclique não induzida de G é uma biclique (induzida), sendo que uma das partes está totalmente contida em U e a outra totalmente contida em W . A seguir mostramos como um algoritmo para gerar as bicliques de um grafo bipartido pode ser usado para gerar as bicliques não induzidas de um grafo $G = (V, E)$. Dado um grafo $G = (V, E)$, onde $V = \{1, \dots, n\}$, o grafo

bipartido G' é construído do seguinte modo: G' consiste de dois conjuntos independentes $U = \{u_1, \dots, u_n\}$ e $W = \{w_1, \dots, w_n\}$, respectivamente, onde dois vértices u_i e w_j são adjacentes se e somente se $(i, j) \in E$. Claramente, *existe uma correspondência um-para-um entre as bicliques não induzidas de G e as bicliques de G' .*

4.1. Propriedades de bicliques em grafos bipartidos

Seja $G = (V, E)$ um grafo bipartido, com partições $U = \{u_1, \dots, u_p\}$ e $W = \{w_1, \dots, w_q\}$. Denotamos por $B = X \cup Y$ um conjunto bipartido completo próprio de G tal que $X \subseteq U$ e $Y \subseteq W$. Denotamos por U_j o subconjunto $\{u_1, \dots, u_j\}$ de U . Por G_j , o subgrafo de $G = (U \cup W, E)$ induzido por $U_j \cup W$, com $j \leq p$. Por \mathcal{B}_j , o conjunto de bicliques de G_j . Os Lemas 5 e 6 indicam como obter as bicliques de \mathcal{B}_j a partir de \mathcal{B}_{j-1} . Estes resultados são usados para garantir que toda biclique de G é listada exatamente uma vez pelo algoritmo.

Lema 5 *Seja $B = X \cup Y$ uma biclique de G_{j-1} . Então, ao considerarmos as bicliques de G_j , ocorre exatamente uma dentre as seguintes situações: (i) $B \notin \mathcal{B}_j$. Neste caso, temos $Y \subseteq N_{u_j}$ e $(X \cup \{u_j\}) \cup Y \in \mathcal{B}_j$. (ii) $B \in \mathcal{B}_j$. Neste caso, temos $Y \not\subseteq N_{u_j}$. Além disso, se $Y \cap N_{u_j} \neq \emptyset$ e todo $u_k \in \{u_1, \dots, u_j\} \setminus X$ satisfaz $N_{u_k} \not\subseteq (N_{u_j} \cap Y)$, então $(X \cup \{u_j\}) \cup (Y \cap N_{u_j}) \in \mathcal{B}_j$. ■*

Lema 6 *Seja $B = X \cup Y$ uma biclique em $\mathcal{B}_j \setminus \mathcal{B}_{j-1}$. Então ocorre exatamente uma dentre as seguintes situações: (i) $X = \{u_j\}$. Neste caso, $N_{u_j} \not\subseteq Y'$ para toda biclique $B' = X' \cup Y'$ em \mathcal{B}_{j-1} . (ii) $X \supset \{u_j\}$. Neste caso, $(X \setminus \{u_j\}) \cup F(X \setminus \{u_j\})$ é uma biclique de \mathcal{B}_{j-1} . ■*

A seguir descrevemos o algoritmo para listar as bicliques de um dado grafo bipartido, usando os Lemas 5 e 6.

Algoritmo BIC(X, Y, j)

Entrada: Grafo bipartido $G = (V, E)$, biclique $B = X \cup Y$ de G_{j-1}

Saída: Enumeração das bicliques $B = X \cup Y$ de G ,
tal que j é o menor vértice em X .

Se $j > p$ então Enumerar $B = X \cup Y$ $\{B \in G_p\}$

Senão

Se $N_{u_j} \subseteq Y$ então $Tree[j] \leftarrow falso$ $\{Lema 6(i)\}$

Se $Y \subseteq N_{u_j}$ então BIC($X \cup \{u_j\}, Y, j + 1$) $\{Lemas 5(i); 6(ii)\}$

Senão BIC($X, Y, j + 1$) $\{Lema 5(ii)\}$

Se $Y \cap N_{u_j} \neq \emptyset$ então

$f := 0$

Para $u_k \notin X$ e $k < j$ faça

Se $N_{u_k} \supseteq N_{u_j} \cap Y$ então $f := 1$

Se $f = 0$ então BIC($X \cup \{u_j\}, Y \cap N_{u_j}, j + 1$) ... $\{Lemas 5(ii); 6(ii)\}$

Chamada externa do algoritmo BIC:

Para $i = 1$ até p faça

Se $N_{u_i} \neq \emptyset$ e $Tree[i]$ então BIC($\{u_i\}, N_{u_i}, i + 1$)

Teorema 7 Seja $G = (V, E)$ um grafo bipartido, com $|V| = n$ e $|E| = m$. O algoritmo BIC lista todas as bicliques de G , exatamente uma vez, com tempo de atraso polinomial $O(nm)$. Além disso, o espaço total requerido pelo algoritmo é $O(n^2)$. ■

Quando consideramos classes de grafos com número polinomial de bicliques então os algoritmos propostos podem ser implementados para executar em tempo polinomial. Entre as classes conhecidas com número polinomial de bicliques estão a classe dos grafos bipartidos convexos e a classe dos grafos bipartidos biconvexos. Demonstramos também como a estrutura particular de cada uma das classes dos grafos bipartidos convexos e dos bipartidos biconvexos pode ser utilizada para reduzir o atraso entre a geração de duas bicliques, reduzindo a complexidade de tempo total do algoritmo BIC.

5. Conclusão

Nesta tese de doutorado tratamos da geração de bicliques de um grafo e da complexidade computacional deste problema combinatório. Como contribuição original, esta tese propõe algoritmos eficientes e provas de NP-completude. Demonstramos que os problemas PARTE DE BICLIQUE e MAIOR BICLIQUE LEXICOGRÁFICA são ambos NP-completos. Apresentamos algoritmos eficientes para os seguintes problemas: (1) geração de bicliques induzidas de um grafo em ordem lexicográfica; (2) geração de bicliques não induzidas de um grafo; (3) geração de bicliques de grafos bipartidos convexos e bipartidos biconvexos.

Reportamos que não há menção na literatura à existência de algoritmos eficientes para a geração de bicliques induzidas em grafos gerais, nem para as classes especiais aqui abordadas. Por outro lado, o algoritmo proposto para a geração de bicliques não induzidas em grafos gerais possui complexidade assintoticamente melhor do que um algoritmo recentemente reportado [Alexe et al., 2004]. Quando consideramos classes de grafos com número polinomial de bicliques então os algoritmos propostos podem ser implementados para executar em tempo polinomial.

Referências

- Alexe, G., Alexe, S., Crama, Y., Foldes, S., Hammer, P., and Simeone, B. (2004). Consensus algorithms for the generation of all maximal bicliques. *Discrete Applied Mathematics*, 143:11–21.
- Byskov, J. M. (2003). Algorithms for k -colouring and finding maximal independent sets. In *Proc. 14th Symp. Discrete Algorithms*, pages 456–457. ACM and SIAM.
- D.Eppstein (2005). All maximal independent sets and dynamic dominance for sparse graphs. In *16th ACM-SIAM Symp. Discrete Algorithms*, Vancouver. ACM and SIAM.
- Dias, V. M. F., Figueiredo, C., and Szwarcfiter, J. (2004a). On the generation of bicliques of a graph. In *Proceedings of CTW 2004*, pages 109–113, Menaggio, Italy. Electronic Notes in Discrete Mathematics.

- Dias, V. M. F., Figueiredo, C. M. H., and Szwarcfiter, J. L. (2004b). Generating all non-induced bicliques of a graph. *Submetido ao Discrete Applied Math*.
- Dias, V. M. F., Figueiredo, C. M. H., and Szwarcfiter, J. L. (2005). Generating bicliques of a graph in lexicographic order. *Aceito para publicação na Theoretical Computer Science*.
- Dias, V. M. F., Fonseca, G. D., Figueiredo, C. M. H., and Szwarcfiter, J. L. (2001). Stable matchings with restricted pairs. In *Proceedings of GRACO 2001*, volume 7, pages 273–287, Fortaleza, Brazil. *Electronic Notes in Discrete Mathematics*.
- Dias, V. M. F., Fonseca, G. D., Figueiredo, C. M. H., and Szwarcfiter, J. L. (2003). The stable marriage problem with restricted pairs. *Theoretical Computer Science*, 306:391–405.
- Dias, V. M. F. and Szwarcfiter, J. L. (2000). Casamentos estáveis com casais forçados e casais proibidos. *Tendências Em Matemática Aplicada e Computacional*, 1:361–372.
- Eppstein, D. (1994). Arboricity and bipartite subgraph listing algorithms. *Inform. Process. Lett.*, 51:207–211.
- Golumbic, M. and Goss, C. (1978). Perfect elimination and chordal bipartite graphs. *J. Graph Theory*, 2:155–163.
- Johnson, D. S., Yannakakis, M., and Papadimitriou, C. H. (1988). On generating all maximal independent sets. *Inform. Process. Lett.*, 27:119–123.
- Kloks, T. and Kratsch, D. (1995). Computing a perfect edge without vertex elimination ordering of a chordal bipartite graph. *Inform. Process. Lett.*, 55:11–16.
- Makino, K. and Uno, T. (2004). New algorithms for enumerating all maximal cliques. In *Proc. 9th Scand. Worksh. Algorithm Theory (SWAT2004)*, pages 260–272. *Lecture Notes in Computer Science* 3111.
- Prisner, E. (1997). Bicliques in graphs 2: Recognizing k -path graphs and underlying graphs of line digraphs. In *Proceedings of WG97*, pages 273–287, Berlin. *Lecture Notes in Computer Science* 1335.
- Prisner, E. (2000). Bicliques in graphs 1: Bounds on their number. *Combinatorica*, 20(1):109–117.
- Tomita, E., Tanaka, A., and Takahashi, H. (2004). The worst-case time complexity for generating all maximal cliques. In *Proc. 10th Int. Computing and Combinatorics Conf. (COCOON 2004)*.
- Tsukiyama, S., Ide, M., Arujoshi, H., and Ozaki, H. (1997). A new algorithm for generating the maximal independent sets. *SIAM J. Comput.*, 6(3):505–517.