Conjunto fonte máximo em grafos de comparabilidade

Marcos Fernando Andrielli
Célia Picinin de Mello

Relatório Técnico IC–96-07

Julho de 1996

Conjunto fonte máximo em grafos de comparabilidade

Marcos Fernando Andrielli* Célia Picinin de Mello†

Sumário

Um grafo de comparabilidade admite várias orientações transitivas. Cada uma delas determina um conjunto fonte para o grafo. Neste artigo propomos um algoritmo que encontra uma orientação transitiva que maximiza o conjunto fonte de um grafo de comparabilidade.

PALAVRAS CHAVES: conjunto fonte, grafo de comparabilidade, algoritmo.

1 Introdução

Diversos algoritmos resolvem o problema de encontrar, se existir, uma orientação transitiva para um grafo [1, 5, 11]. Grafos que admitem uma orientação transitiva são denominados *grafos de comparabilidade*.

Numa orientação transitiva pode-se destacar vértices que são fontes ou sumidouros, isto é, vértices com grau de entrada nulo (fontes) ou grau de saída nulo (sumidouro). Note que se um vértice v é fonte de uma orientação transitiva $\vec{G}$, v é sumidouro da orientação reversa de $\vec{G}$. Nem todo vértice de um grafo de comparabilidade é fonte de alguma orientação transitiva. Olariu [8] e Gimbel [2] caracterizaram vértices de um grafo de comparabilidade que são fontes de alguma orientação transitiva. Szwarcfiter, Mello e Figueiredo [10] descreveram propriedades que relacionam fontes com as cliques maximais do grafo. Posteriormente, em [9], caracterizaram conjunto fonte (um conjunto de vértices que são fontes de alguma orientação transitiva).

Neste artigo propomos um algoritmo para o problema de encontrar, em um grafo de comparabilidade, uma orientação transitiva com conjunto fonte de cardinalidade máxima. Este algoritmo usa a árvore de decomposição modular descrita por Spinrad [11].

O símbolo G denotará um grafo simples, $V(G)$ seu conjunto de vértices e $A(G)$, o de arestas. $\vec{G}$ denota uma orientação acíclica de G. $\vec{G}$ é *transitiva* quando $(u,v),(v,w) \in A(\vec{G})$ implica $(v,w) \in A(\vec{G})$, para todo $u,v,w \in A(G)$. O *grau de entrada* (*grau de saída*) de um vértice v em um grafo orientado é o número de arestas que possuem v como extremidade final (inicial). Uma *fonte* é um vértice com grau de entrada nulo, enquanto que um *sumidouro* é um com grau de saída nulo. Um grafo G é um *grafo de comparabilidade* se admite uma orientação transitiva. Um *conjunto fonte* (*sumidouro*) de um grafo de comparabilidade G,

*Universidade Estadual de Campinas, Instituto de Computação, Caixa Postal 6176, 13081-970 Campinas, SP, Brasil. Parcialmente financiado por SAE - Serviço de Apoio ao Estudante.

†Universidade Estadual de Campinas, Instituto de Computação, Caixa Postal 6176, 13081-970 Campinas, SP, Brasil. Parcialmente financiado por CNPq e FAPESP.

é um conjunto de vértices que são fontes (sumidouros) em alguma orientação transitiva para G. Uma *clique* (*conjunto independente*) é um conjunto de vértices de G dois a dois adjacentes (não adjacentes). Um conjunto de vértices X é *dominante* se cada vértice de $V(G) \setminus X$ é adjacente a algum vértice de X.

Na seção 2, encontram-se alguns resultados de Spinrad [11], necessários para o desenvolvimento de nosso algoritmo. Na seção 3, apresentamos uma demonstração alternativa de que um determinado tipo de módulo possui exatamente duas orientações transitivas. O algoritmo que encontra uma orientação transitiva com conjunto fonte de cardinalidade máxima será apresentado na seção 4 e, finalmente, a seção 5 contém as conclusões.

2 Preliminares

Uma decomposição modular, descoberta independentemente por Möhring [6] e Muller e Spinrad [7], é um processo para decompor um grafo. Em qualquer estágio, o subgrafo que está sendo decomposto é um módulo. Cada um dos subgrafos será decomposto recursivamente. Este processo continua até que todos os subgrafos que estão sendo decompostos contenham somente um vértice (módulos triviais).

Um *módulo* de um grafo G é um subconjunto t de vértices de $V(G)$ tal que cada vértice de $V(G) \setminus t$ é adjacente a todo vértice em t ou a nenhum vértice em t. Cada vértice de G e o conjunto $V(G)$ são módulos *triviais*. Todo módulo de um grafo pertence a exatamente uma das três classes de módulos: paralelo, serial ou vizinhança. O módulo t é *paralelo* se $G[t]$ (o subgrafo de G gerado pelos vértices de t) é desconexo; t é *serial* se $\overline{G[t]}$ é desconexo; t é vizinhança se, ambos, $G[t]$ e $\overline{G[t]}$ são conexos.

No processo de decomposição modular é construída a *árvore de decomposição modular* da seguinte forma. Considere o módulo $t = V(G)$. Se t é somente um vértice, o próprio vértice é a decomposição modular de G. Caso contrário, t é um módulo paralelo, serial ou vizinhança. Se t é um módulo paralelo (serial), crie um vértice P (S) na árvore e insira como filhos deste vértice as decomposições modulares dos componentes conexos de $G[t]$ ($\overline{G[t]}$). Se t é um módulo vizinhança, crie um vértice N na árvore e insira como filhos deste vértice as decomposições modulares dos submódulos maximais de t. A árvore de decomposição modular de um grafo G é única, a menos de isomorfismos [11].

O *grafo representante*, $R[t]$, do módulo t na decomposição modular é o grafo gerado por um subconjunto consistindo de um único vértice de cada submódulo de t na árvore de decomposição modular.

A figura 1 exibe um grafo; a figura 2 a sua árvore de decomposição modular e as figuras em 3, são os grafos representantes para os módulos N, S, P_1 e P_2 da árvore de decomposição modular da figura 2.

Observe que o grafo representante de um módulo paralelo é gerado por um conjunto independente, enquanto o de um módulo serial é gerado por uma clique. Na seção 3, mostraremos que o grafo representante de um módulo vizinhança pode ser "pensado" como gerado por um módulo vizinhança cujos submódulos maximais possuem apenas um elemento. Observe, também, que se G é um grafo de comparabilidade, o grafo representante de um módulo de G é um grafo de comparabilidade.

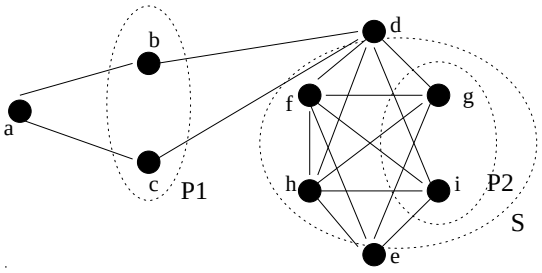

Figura 1: Um grafo G.

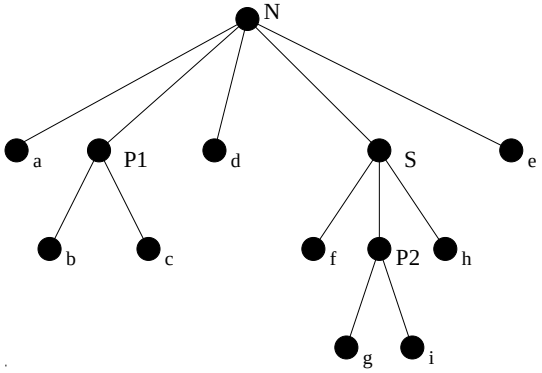

Figura 2: Árvore de decomposição modular de G

Spinrad [11] propos um algoritmo $O(n^2)$ que orienta transitivamente um grafo de comparabilidade. Seu algoritmo encontra uma orientação transitiva $\vec{G}$ para G se, e somente se, cada grafo representante na decomposição modular de G possui uma orientação transitiva.

Lema 1 *Existe uma orientação transitiva para G se, e somente se, existe uma orientação transitiva para cada grafo representante na decomposição modular de G.*

O algoritmo de Spinrad orienta cada par de vértices adjacentes u e v de G de u para v se, e somente se, os vértices, u' e v' de $R[t]$, correspondentes aos submódulos maximais de t na árvore de decomposição modular que contém u e v respectivamente, são orientados de u' para v' em $\vec{R}[t]$.

3 Unicidade da orientação transitiva para grafos representantes de módulos vizinhança

Nesta seção apresentamos uma prova alternativa de que o grafo representante de um módulo vizinhança, quando de comparabilidade, admite apenas duas orientações transitivas, sendo uma a reversa da outra. Grafos de comparabilidade que satisfazem essa condição são chamados *UPO* (unique partially orderable).

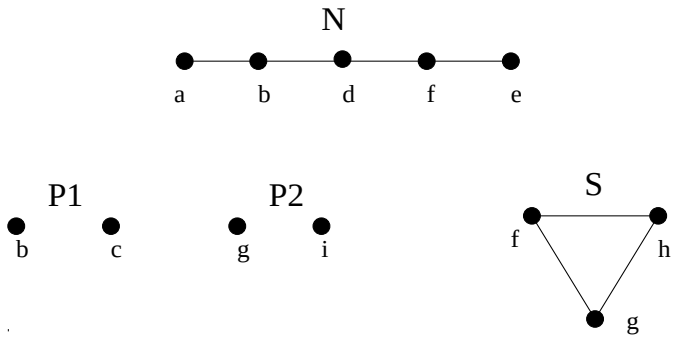

Figura 3: Grafos Representantes

Teorema 1 *Seja G um grafo de comparabilidade. G é UPO se, e somente se, todo módulo não trivial de G é um conjunto independente.*

A demonstração do Teorema 1 encontra-se em [5].

Lema 2 *Sejam t um módulo* vizinhança *de um grafo G e $R[t]$ o grafo representante de t. Se t_1 é um módulo de $R[t]$, então t_1 é trivial.*

Prova. Suponha, por absurdo, que $R[t]$ contenha um módulo t_1 não trivial. Então, $1 < |t_1| < |V(R[t])|$.

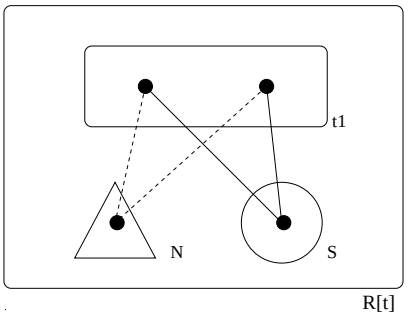

Figura 4:

Então os vértices de $R[t]$ são particionados em 3 subconjuntos: t_1, S e N onde S é formado pelos vértices de $R[t]$ adjacentes a vértices de t_1 e N pelos vértices que não são adjacentes a vértices de t_1, conforme figura 4.

Sendo t_1 não trivial, t_1 contém pelo menos 2 vértices ($|t_1| > 1$) e pelo menos um dentre os conjuntos S e N é não vazio ($|t_1| < |V(R[t])|$).

Desde que $R[t]$ é o grafo representante do módulo vizinhança t, cada vértice em $R[t]$ representa um submódulo maximal de t.

Então seja t_1' o conjunto dos vértices de t que são representados pelos vértices em t_1. t_1' é um módulo em t, e contém pelo menos dois outros submódulos, contradizendo a

maximalidade dos submódulos de t representados em t_1. ■

Observe que grafos representantes de módulos seriais ou paralelos possuem módulos distintos dos triviais.

Teorema 2 *Se G é um grafo de comparabilidade, então o grafo representante de um módulo vizinhança de G é UPO.*

Prova. Seja t um módulo vizinhança de G e $R[t]$ seu grafo representante. Então R[t] é um grafo de comparabilidade. Do lema 2, todo módulo de $R[t]$ é trivial. Logo, $R[t]$ é UPO pelo teorema 1. ■

4 O algoritmo

Nesta seção descrevemos um algoritmo que encontra uma orientação transitiva que maximiza o conjunto fonte de um grafo de comparabilidade.

Um conjunto fonte (*conjunto sumidouro*) S é *exato* quando existe uma orientação transitiva cujas únicas fontes (sumidouros) são os vértices em S.

Para um grafo de comparabilidade G e um conjunto fonte S de G são equivalentes [10, 3]:

- S é exato.
- S é um conjunto independente maximal.
- S é um conjunto dominante.

Observe que se S é um conjunto fonte máximo, S é exato. A recíproca não é verdadeira, veja o grafo de comparabilidade da figura 5.

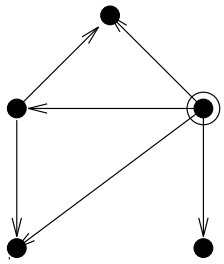

Figura 5:

Sejam t um nó da árvore de decomposição modular, t_i um filho de t na árvore de decomposição modular, $G[t]$ o grafo gerado em G pelos vértices do módulo t, $R[t]$ o grafo representante de t, $\vec{G}[t]$ uma orientação transitiva para $G[t]$ (que maximiza o número de fontes), $\vec{R}[t]$ uma orientação transitiva para $R[t]$ e v_i um vértice de $R[t]$ que representa o filho t_i de t.

O algoritmo descrito a seguir, baseado no algoritmo de Spinrad, associa a cada nó t da árvore de decomposição modular T um rótulo, $r(t)$, e um conjunto, $f(t)$, que indicam,

respectivamente, o número de fontes e o conjunto fonte de uma orientação transitiva que maximiza o número de fontes de $G[t]$. O rótulo $r(t_i)$ também estará associado ao vértice v_i de $R[t]$.

Algoritmo: *ConjuntoFonteMáximo;*

entrada: *G (grafo de comparabilidade)*

saída: *uma orientação transitiva para G com conjunto fonte máximo, o total de fontes e os vértices que são fontes da orientação encontrada.*

1. Determine a árvore de decomposição modular, T, para G;
2. Construa o grafo representante de cada nó t, de T;
3. Aplique o procedimento *OrientaNó(t)* ao nó raiz da árvore T;

Procedimento: *OrientaNó(t);*

entrada: *um nó t da árvore de decomposição modular;*

saída: *orientação transitiva de $G[t]$ que maximiza o conjunto de fontes, os vértices que são fontes e o número de fontes.*

1. Se t é uma folha v, então
   - $r(t) := 1$, $f(t) := \{v\}$.
2. Se t é um nó P, então
   - *para cada* t_i $(1 \leq i \leq k)$ filho de T *faça* OrientaNó(t_i);
   - $r(t) := \sum_{i=1}^{k} r(t_i)$, $f(t) := \bigcup_{i=1}^{k} f(t_i)$.
3. Se t é um nó S, então
   - *para cada* t_i $(1 \leq i \leq k)$ filho de T *faça* OrientaNó(t_i);
   - $r(t) := max_{1 \leq i \leq k}\{r(t_i)\}$, $f(t) := \{f(t_i) | t_i$ é um módulo onde $r(t_i)$ é máximo$\}$.
4. Se t é um nó N, então
   - *para cada* t_i $(1 \leq i \leq k)$ filho de T *faça* OrientaNó(t_i);
   - Determine as duas orientações transitivas $\vec{R}_1$ e $\vec{R}_2$ de $R[t]$. Sejam P_1 e P_2 os conjuntos dos índices dos vértices fontes de $\vec{R}_1$ e $\vec{R}_2$, respectivamente;

- *se* $(\sum_{i\in P_1} r(t_i)) \geq (\sum_{i\in P_2} r(t_i))$ *então*

$$r(t) := \sum_{i\in P_1} r(t_i),\ f(t) := \bigcup_{i\in P_1} f(t_i)$$

senão

$$r(t) := \sum_{i\in P_2} r(t_i), f(t) = \bigcup_{i\in P_2} f(t_i)$$

5. Utilize o método de Spinrad para, a partir dos grafos representantes, orientar $G[t]$.

Para cada módulo t da árvore de decomposição modular, o algoritmo proposto constrói uma orientação transitiva com conjunto fonte máximo para o subgrafo $G[t]$. Sendo G um módulo, o algoritmo devolve uma orientação transitiva, $\vec{G}$, para G, que maximiza o conjunto fonte. O teorema 3 demonstra a corretude do algoritmo.

Teorema 3 *Seja G um Grafo de Comparabilidade, o algoritmo ConjuntoFonteMáximo encontra uma orientação transitiva para G de forma a maximizar o conjunto de fontes.*

Prova. Seja G um grafo de comparabilidade. Sendo G um módulo, basta provar que o procedimento OrientaNó(t) encontra uma orientação transitiva que maximiza o conjunto fonte para cada $G[t]$.

Se t é uma folha v, tem-se $f(t) = \{v\}$ e $r(t) = 1$, por definição de fonte.

Se t não é uma folha, suponha, por indução, que os filhos $t_1, \ldots, t_k$ de t já foram visitados pelo procedimento OrientaNó(t), isto é, cada $G[t_i]$, $i = 1, \ldots, k$, possui uma orientação transitiva que maximiza o conjunto fonte.

Sendo que t não é folha, então t é um nó da árvore de decomposição modular de G que é um módulo paralelo, serial ou vizinhança de G. A demonstração do teorema 3 segue dos lemas 3, 4 e 5.

Lema 3 *Seja t um módulo paralelo e $t_1, \ldots, t_k$ seus filhos. Então $G[t]$ admite uma orientação transitiva que maximiza o conjunto fonte e $r(t) = \sum_{i=1}^{k} r(t_i)$ e $f(t) = \bigcup_{i=1}^{k} f(t_i)$.*

Prova. Seja $R[t]$ o grafo representante de t. Sendo t um módulo paralelo, $R[t]$ é um conjunto independente com k vértices. Logo, qualquer orientação transitiva $\vec{R}$ que maximiza o número de fontes em $R[t]$ possui os k vértices como fontes.

Por hipótese de indução, para cada $1 \leq i \leq k$, $G[t_i]$, possui uma orientação transitiva que maximiza o conjunto fonte. Desde que não existem arestas entre os vértices de $G[t_i]$ e $G[t_j]$, $i \neq j$, então OrientaNó(t) preserva as orientações de cada $G[t_i]$. Logo, pelo lema 1, $G[t]$ admite uma orientação que maximiza o conjunto fonte. Portanto, $f(t) = \bigcup_{i=1}^{k} f(t_i)$ e $r(t) = \sum_{i=1}^{k} r(t_i)$. ■

Lema 4 *Seja t um módulo serial, então $G[t]$ admite uma orientação transitiva que maximiza o conjunto fonte e $r(t) = max_{1\leq i\leq k}\{r(t_i)\}$ e $f(t) = \{f(t_i) | t_i$ é um módulo onde $r(t_i)$ é máximo$\}$.*

Prova. Seja $R[t]$ o grafo representante de t. Sendo t um módulo serial, $R[t]$ é uma clique. Logo, qualquer orientação transitiva de $R[t]$ possui apenas uma fonte.

Escolha para fonte, em $\vec{R}[t]$, um vértice v_i tal que $r(t_i)$ é máximo.

Por hipótese de indução, $G[t_i]$ admite uma orientação transitiva, $\vec{G}[t_i]$, com conjunto fonte, $f(t_i)$, máximo.

Pela lema 1, $G[t]$ admite uma orientação transitiva $\vec{G}[t]$ tal que $f(t_i)$ é um subconjunto do conjunto fonte de $\vec{G}[t]$.

Sendo que $f(t_i)$ é o conjunto fonte máximo em $\vec{G}[t_i]$ e está contido em um mesmo componente conexo de $\overline{G[t]}$, tem-se que todo vértice $v \in (G[t] \setminus f(t_i))$ é adjacente a algum vértice em $f(t_i)$.

Logo, $\vec{G}[t]$ é uma orientação transitiva que maximiza o número fontes em $G[t]$. Portanto, $f(t) = \{f(t_i) | t_i$ é um módulo onde $r(t_i)$ é máximo$\}$ e $r(t) = max_{1 \leq i \leq k}\{r(t_i)\}$. ■

Lema 5 *Sejam t um módulo vizinhança e $t_1, \ldots, t_k$ seus filhos; $\vec{R}_1$ e $\vec{R}_2$ as orientações transitivas do grafo representante $R[t]$; P_1 e P_2 os conjuntos dos índices dos vértices fontes de $\vec{R}_1$ e $\vec{R}_2$, respectivamente. Então*
se $(\sum_{i \in P_1} r(t_i)) \geq (\sum_{i \in P_2} r(t_i))$ então

$$r(t) = \sum_{i \in P_1} r(t_i), \ f(t) = \bigcup_{i \in P_1} f(t_i)$$

senão

$$r(t) = \sum_{i \in P_2} r(t_i), \ f(t) = \bigcup_{i \in P_2} f(t_i).$$

Prova. Escolha R_j tal que $\sum_{i \in P_j} r(t_i)$ é máximo. Suponha, sem perda de generalidade, $j = 1$.

Seja X o conjunto dos vértices em $G[t]$ pertencentes à $f(t_i)$, $i \in P_1$. Por hipótese de indução, cada $f(t_i)$ é um conjunto fonte máximo para $\vec{G}[t_i]$, $i \in P_1$.

Pelo lema 1, $G[t]$ admite uma orientação transitiva, $\vec{G}[t]$, tal que X é um subconjunto do conjunto fonte de $\vec{G}[t]$.

Seja $v \in (V(G[t]) \setminus X)$ e suponha que v é fonte de $\vec{G}[t]$. Sendo que $t_1, \ldots, t_k$ particionam $V(G[t])$, v pertence a algum t_j, $1 \leq j \leq k$.

Seja v_j o vértice representante de t_j em $R[t]$. Se $j \in P_1$, então por hipótese de indução, f_j é um conjunto fonte máximo para $\vec{G}[t_j]$ e $v \notin f(t_j)$. Logo, pelo lema 1, v não é fonte em $\vec{G}[t]$.

Se $j \notin P_1$, então pelas equivalências da página 5, existe uma aresta entre v_j e algum v_i, $i \in P_1$, em $R[t]$ (v_j não é fonte em $\vec{R_1}$). Logo, existe pelo menos uma aresta entre v e algum vértice pertencente a X em $G[t]$. Portanto, v não é fonte em $\vec{G}[t]$. ■

Na figura 6 estão os grafos representantes da árvore de decomposição modular do grafo G da figura 1. Os valores entre parênteses são os retornados pelo procedimento OrientaNó(t). No caso do nó N, estão representadas as duas orientações possíveis. A orientação escolhida foi a da direita na figura, pois preserva quatro fontes. Finalmente, a figura 7, mostra a orientação transitiva resultante, a qual maximiza o número de fontes e o conjunto fonte.

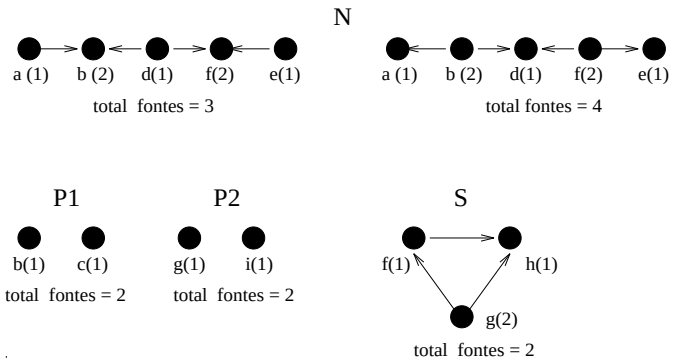

Figura 6: Grafos Representantes

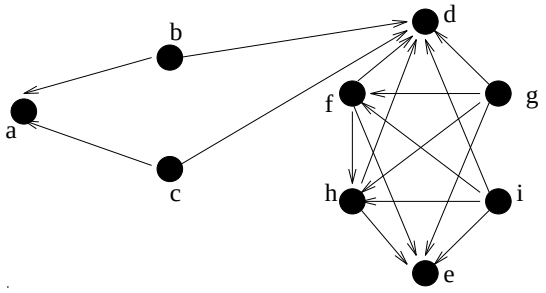

Figura 7: Grafo G orientado

5 Conclusões

Um problema relacionado, e citado em [10], consiste em dado um conjunto de vértices predeterminados W, verificar se W é um conjunto fonte máximo para um grafo de comparabilidade.

Em cada passo do algoritmo ConjuntoFonteMáximo construa $\vec{G}[t]$ com $X \subseteq f(t)$, se $X = W \cap V(G[t]) \neq \emptyset$. Então W é um conjunto fonte máximo para o grafo se, e somente se, tal construção for possível e $f(V(G)) = W$. Observe que a complexidade do algoritmo ConjuntoFonteMáximo é limitada pela complexidade de encontrar uma orientação transitiva.

Um grafo de comparabilidade pode ter várias orientações transitivas que maximizam o conjunto fonte. Um exemplo são as duas orientações transitivas distintas da figura 8 para o mesmo grafo. Com pequenas alterações, o algoritmo ConjuntoFonteMáximo, pode exibir todas as orientações transitivas. Para isso, basta considerar todas as possibilidades de vértices com valores máximo nos módulos seriais e utilizar as duas orientações no caso de módulos vizinhança, se ambas apresentarem o mesmo número de fontes preservadas.

Referências

[1] P.C. Gilmore, A.J. Hoffman, A characterization of comparability graphs and interval

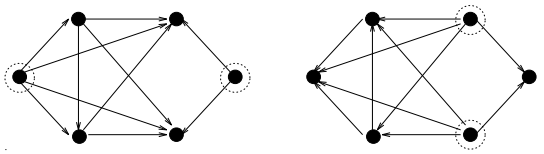

Figura 8:

graphs, *Canad. J. Math.*, **16** (1964) 539–548.

[2] J. Gimbel, Sources in posets and comparability graphs, *Order* **9** (1992) 361–365.

[3] J. Gimbel, A note on sources and sinks in comparability graphs, *Order*, submitted.

[4] M.R. Garey and D.S. Johnson, *Computer and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, (1979).

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

[6] R. H. Möhring, Algorithmic aspects of comparability graphs and interval graphs, in I. Rival, ed., *Graphs and Orders*, 41–102, D. Reidl, Dordrecht, (1985).

[7] J. Muller and J. Spinrad, On-line modular decomposition, *Tech. Rep. GIT-ICS-84/11*, Georgia Institute of Technology.

[8] S. Olariu, On sources in comparability graphs with applications, *Discrete Maths*, **110** (1992) 289–292.

[9] J.L. Szwarcfiter, C.P. de Mello, C.M.H. de Figueiredo, Sources, Sinks, Even and Odd Pairs in Comparability Graphs, *Rapport de Recherche*, L.I.R.M.M., (1993), to appear *Orders*.

[10] J.L. Szwarcfiter, C.P. de Mello, C.M.H. de Figueiredo, On finding transitive orientations with prescribed sources and sinks, *Congressus Numerantium*, **98** (1993) 191–198.

[11] J. Spinrad, On comparability and permutation graphs, *SIAM J. Comput*, **14** nº 3 (1985) 658–670.

Relatórios Técnicos – 1995

95-01 **Paradigmas de algoritmos na solução de problemas de busca multidimensional,** *Pedro J. de Rezende, Renato Fileto*

95-02 **Adaptive enumeration of implicit surfaces with affine arithmetic,** *Luiz Henrique de Figueiredo, Jorge Stolfi*

95-03 **W3 no Ensino de Graduação?,** *Hans Liesenberg*

95-04 **A greedy method for edge-colouring odd maximum degree doubly chordal graphs,** *Celina M. H. de Figueiredo, João Meidanis, Célia Picinin de Mello*

95-05 **Protocols for Maintaining Consistency of Replicated Data,** *Ricardo Anido, N. C. Mendonça*

95-06 **Guaranteeing Full Fault Coverage for UIO-Based Methods,** *Ricardo Anido and Ana Cavalli*

95-07 **Xchart-Based Complex Dialogue Development,** *Fábio Nogueira de Lucena, Hans K.E. Liesenberg*

95-08 **A Direct Manipulation User Interface for Querying Geographic Databases,** *Juliano Lopes de Oliveira, Claudia Bauzer Medeiros*

95-09 **Bases for the Matching Lattice of Matching Covered Graphs,** *Cláudio L. Lucchesi, Marcelo H. Carvalho*

95-10 **A Highly Reconfigurable Neighborhood Image Processor based on Functional Programming,** *Neucimar J. Leite, Marcelo A. de Barros*

95-11 **Processador de Vizinhança para Filtragem Morfológica,** *Ilka Marinho Barros, Roberto de Alencar Lotufo, Neucimar Jerônimo Leite*

95-12 **Modelos Computacionais para Processamento Digital de Imagens em Arquiteturas Paralelas,** *Neucimar Jerônimo Leite*

95-13 **Modelos de Computação Paralela e Projeto de Algoritmos,** *Ronaldo Parente de Menezes e João Carlos Setubal*

95-14 **Vertex Splitting and Tension-Free Layout,** *P. Eades, C. F. X. de Mendonça N.*

95-15 **NP-Hardness Results for Tension-Free Layout,** *C. F. X. de Mendonça N., P. Eades, C. L. Lucchesi, J. Meidanis*

95-16 **Agentes Replicantes e Algoritmos de Eco,** *Marcos J. C. Euzébio*

95-17 **Anais da II Oficina Nacional em Problemas Combinatórios: Teoria, Algoritmos e Aplicações,** *Editores: Marcus Vinicius S. Poggi de Aragão, Cid Carvalho de Souza*

95-18 **Asynchronous Teams: A Multi-Algorithm Approach for Solving Combinatorial Multiobjective Optimization Problems,** *Rosiane de Freitas Rodrigues, Pedro Sérgio de Souza*

95-19 **wxWindows: Uma Introdução,** *Carlos Neves Júnior, Tallys Hoover Yunes, Fábio Nogueira de Lucena, Hans Kurt E. Liesenberg*

95-20 **John von Neumann: Suas Contribuições à Computação,** *Tomasz Kowaltowski*

95-21 **A Linear Time Algorithm for Binary Phylogeny using PQ-Trees,** *J. Meidanis and E. G. Munuera*

95-22 **Text Structure Aiming at Machine Translation,** *Horacio Saggion and Ariadne Carvalho*

95-23 **Cálculo de la Estructura de un Texto en un Sistema de Procesamiento de Lenguaje Natural,** *Horacio Saggion and Ariadne Carvalho*

95-24 **ATIFS: Um Ambiente de Testes baseado em Inje,c ao de Falhas por Software,** *Eliane Martins*

95-25 **Multiware Plataform: Some Issues About the Middleware Layer,** *Edmundo Roberto Mauro Madeira*

95-26 **WorkFlow Systems: a few definitions and a few suggestions,** *Paulo Barthelmess and Jacques Wainer*

95-27 **Workflow Modeling,** *Paulo Barthelmess and Jacques Wainer*

Relatórios Técnicos – 1996

96-01 **Construção de Interfaces Homem-Computador: Uma Proposta Revisada de Disciplina de Graduação,** *F'abio Nogueira Lucena and Hans K.E. Liesenberg*

96Abs **DCC-IMECC-UNICAMP Technical Reports 1992–1996 Abstracts,** *C. L. Lucchesi and P. J. de Rezende and J.Stolfi*

96-02 **Automatic visualization of two-dimensional cellular complexes,** *Rober Marcone Rosi and Jorge Stolfi*

Instituto de Computação
Universidade Estadual de Campinas
13081-970 – Campinas – SP
BRASIL

reltec@dcc.unicamp.br