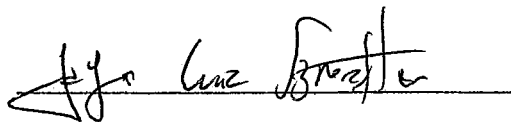


# ENUMERAÇÃO DOS CONJUNTOS INDEPENDENTES MAXIMAIS DE GRAFOS

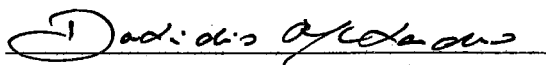
*Mónica Rosa Villanueva Ilufi*

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO EM ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

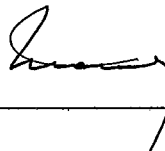
Aprovada por:



Prof. Jayme Luiz Szwarcfiter, Ph.D.  
(presidente)



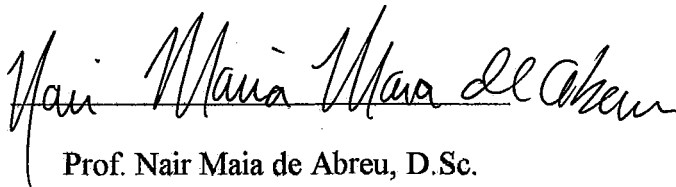
Prof. Dalcidio Morais Claudio, Dr.Rer.Nat.



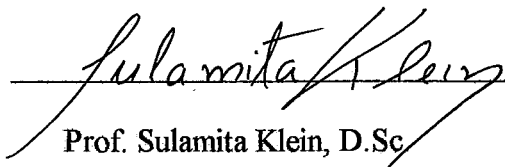
Prof. Nelson Maculan, D.Sc.



Prof. Célia Picinin de Mello, D.Sc.



Prof. Nair Maia de Abreu, D.Sc.



Prof. Sulamita Klein, D.Sc.

RIO DE JANEIRO, RJ - BRASIL  
MAIO DE 1996

VILLANUEVA ILUFI, MÓNICA ROSA

Enumeração dos conjuntos independentes maximais de grafos  
[Rio de Janeiro] 1996.

VII, 145p., 29,7 cm

(COPPE/UFRJ, D.Sc., Engenharia de Sistemas e Computação, 1996)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Conjuntos independentes maximais. 2. Caminho induzido.

3. Ciclo induzido. 4. Grafo grade completa de duas linhas.

5. Grafo de intervalo. 6. Grafo triangularizado.

7. Grafo separável por cliques. 8. Grafo série-paralelo a dois terminais

I. COPPE/UFRJ      II. Título (série)

*Ao Nano*

## AGRADECIMENTOS

Agradeço:

ao Professor Jayme Szwarcfiter pela motivação, paciência, compreensão e orientação durante o trabalho na tese e em todos os anos de estudo de pós-graduação;

aos Professores integrantes da banca;

ao Professor Nelson Maculan por seu apoio constante e amizade;

ao grupo de trabalho e amigos, especialmente Carmen, Célia, Celina, Márcia e Sula, pelo apoio, comentários, discussões e incentivo no trabalho;

a Márcia Cerioli, minha embaixadora no programa;

aos colegas e amigos do programa, especialmente Cláudia L., Claudia B., Clícia, Cristina B., Oscar P. e Philippe M., pela amizade, companheirismo e experiências compartilhadas;

aos Professores do Programa de Engenharia de Sistemas e Computação da COPPE/UFRJ;

a COPPE e ao povo brasileiro pela possibilidade de realizar a pós-graduação e de viver em seu país;

aos colegas da Universidad de La Frontera, Temuco, Chile, e a todos os que possibilitaram a realização da minha pós-graduação, em geral;

a minha família.



Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para obtenção do grau de Doutor em Ciências (D.Sc.)

## ENUMERAÇÃO DOS CONJUNTOS INDEPENDENTES MAXIMAIS DE GRAFOS

*Mónica Rosa Villanueva Ilufi*  
MAIO, 1996

Orientador: Prof. Jayme Luiz Szwarcfiter  
Programa: Engenharia de Sistemas e Computação.

Neste trabalho estuda-se o problema de *enumeração da família de conjuntos independentes maximais de um grafo dado*, considerando-se também o problema de *contagem do número de conjuntos independentes maximais (c.i.m.) existentes em um grafo dado*. O problema de determinar o *número de conjuntos independentes maximais de um grafo qualquer é #P-completo*.

Desenvolvem-se algoritmos para construir a família de conjuntos independentes maximais de *caminhos induzidos, ciclos induzidos, grafos grade completas de duas linhas, grafos de intervalo e grafos triangularizados*; transformando esse problema no problema de se determinar todos os caminhos maximais fonte-sumidouro num digrafo que depende da classe particular considerada. Determina-se, além disso, a equação de recorrências que caracteriza o número de c. i. m. e sua solução. Modificando-se o algoritmo para gerar todos os c. i. m. de um grafo de intervalo, é possível contar o número deles. A complexidade total de cada algoritmo é polinomial no tamanho do grafo e no número de c.i.m., no entanto a complexidade por c.i.m. é linear no número de vértices.

Logo, trabalha-se com a estratégia de operar dois grafos para os quais dispõe-se da família de c.i.m. de cada um deles. Desenvolve-se um algoritmo para gerar a família de c.i.m. de um *grafo separável por cliques* e de um *grafo série-paralelo a dois terminais*, aproveitando a árvore de decomposição do grafo e as operações de unir dois grafos por conexão série, por um conjunto comum a ambos e por conexão paralela.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

## ENUMERATION OF MAXIMAL INDEPENDENTE SETS OF GRAPHS

*Mónica Rosa Villanueva Ilufi*  
MAY, 1996

Thesis Supervisor: Prof. Jayme Luiz Szwarcfiter  
Department: Computer Science and System Engineering.

In this work, it is studied the problem of *enumerating the family of maximal independent sets of a given graph*, it is also consider the problem of *counting the number of maximal independent sets of a given graph*. The problem of determining the *number* of maximal independent sets of any graph is **#P-complete**.

It is developed algorithms to construct the family of maximal independent sets for *induced paths, induced cycles, complete grid of two lines, interval graphs and triangulated graphs*; transforming this problem in the problem of generating all the maximal source-sink paths of an oriented graph which depends on the particular class of graph. It is also determined the recurrence equation which characterized the number of maximal independent sets of a complete grid of two lines. Modifying the algorithm to generate all the maximal independent sets of an interval graph, it is possible to count them. The total complexity of each algorithm is polynomial in the size of the graph and in the number of maximal independent sets of the graph.

Working with the strategy of operate two graphs, it is developed algorithms to construct the family of maximal independent sets for clique separable graphs and for two terminal series-parallel graphs, given the tree decomposition of the graph and the family of maximal independent sets of the primitive graphs, in polynomial time of the size of the graph and of the number of maximal independent sets of the graph.

## ÍNDICE

<b>1. Introdução</b> .....	1
1.1. O problema .....	1
1.2. Definições básicas e notação .....	2
1.2.1. Grafos .....	2
1.2.2. Complexidade .....	5
1.3. Conteúdo da tese .....	6
<b>2. Contagem de Conjuntos Independentes Maximais</b> .....	8
2.1. Limite superior para o número de c.i.m. ....	8
2.2. Número de c.i.m. para árvores .....	15
2.3. Número de c.i.m. para casos particulares .....	21
<b>3. Geração de Conjuntos Independentes Maximais</b> .....	23
3.1. Algoritmo de Paull e Unger .....	23
3.2. Algoritmo de Tsukiyama, Ide, Ariyoshi e Shirawaka .....	26
3.3. Algoritmo de Loukakis .....	30
3.4. Algoritmo de Johnson, Yannakakis e Papadimitriou .....	35
3.5. Algoritmos para grafos particulares .....	37
<b>4. Solução por transformação do problema</b> .....	38
4.1. Caminhos sem corda .....	38
4.2. Ciclos sem corda .....	46
4.3. Grafos grade completas de duas linhas .....	56
4.4. Grafos de intervalo .....	66
4.5. Grafos triangularizados .....	88
<b>5. Solução por operações em grafos</b> .....	97
5.1. Grafos com uma articulação .....	97
5.2. Grafos com um conjunto articulação clique .....	104
5.3. Grafos separáveis por cliques .....	112
5.4. Grafos com conexão paralela .....	119
5.5. Grafos série-paralelo a dois terminais .....	134
<b>6. Conclusões</b> .....	138
<b>7. Referências Bibliográficas</b> .....	142

## Capítulo 1

# INTRODUÇÃO

Neste capítulo é apresentado o problema de estudo do trabalho, formulam-se todas as definições necessárias para o desenvolvimento do tema, e se descreve a distribuição do trabalho nos capítulos seguintes.

### 1.1. O Problema

Os problemas básicos de combinatória podem ser descritos como: *Quantos elementos têm um conjunto dado?* e *Qual é o conjunto dos elementos que satisfazem determinadas propriedades?* O primeiro problema corresponde à *contagem* das configurações com as propriedades desejadas para pertencer ao conjunto, e o segundo corresponde à *enumeração* dessas configurações.

A configuração particular de interesse aqui é um *conjunto independente* ou *conjunto estável* de um grafo. Um conjunto independente de um grafo é um subconjunto de vértices tal que não são adjacentes dois a dois; um conjunto independente é maximal se não existe outro conjunto independente que o contenha em forma própria; um conjunto independente é *máximo* ou *maior* se não existe outro de cardinalidade maior.

Em relação aos conjuntos independentes de um grafo  $G$  dado, tem-se os problemas:

- determinar o tamanho  $\alpha(G)$  do maior conjunto independente de  $G$
- determinar o número  $i(G)$  de conjuntos independentes maximais de  $G$
- gerar a família  $F(G)$  de conjuntos independentes maximais de  $G$ .

O tamanho  $\alpha(G)$  corresponde ao maior número de elementos de  $V$  não relacionados entre si. O número  $i(G)$  dá a quantidade de combinações maximais de elementos não adjacentes dois a dois. A família  $F(G)$  de conjuntos independentes maximais representa todas as combinações maximais, sem duplicação, dos elementos de  $V$  não adjacentes dois a dois.

O problema de determinar o tamanho  $\alpha(G)$  do maior conjunto independente de um grafo  $G$  é *NP-completo* [Cook, 1973; Garey e Johnson, 1979]. O problema de determinar o número  $i(G)$  de conjuntos independentes maximais de um grafo  $G$  qualquer, é *#P-completo* [Simon, 1977; Valiant, 1979].

Este trabalho trata dos problemas de contagem e de geração de todos os conjuntos independentes maximais em classes particulares de grafos.

Lawler (1976a) prova que o número cromático de um grafo, isto é: o número mínimo de cores necessárias para colorir os vértices de um grafo de modo que vértices adjacentes possuam cores diferentes, pode ser obtido por um algoritmo de complexidade de pior caso de  $O(nm[1+3\sqrt{3}]^n)$ , utilizando a partição dos vértices em conjuntos independentes maximais.

## 1.2. Definições básicas e notação

### 1.2.1 Grafos

Um *grafo não direcionado*  $G(V,E)$  consiste de um conjunto finito não vazio  $V$  e um conjunto  $E$  de pares não ordenados de elementos distintos de  $V$ . Os elementos de  $V$  são os *vértices* e os de  $E$  são as *arestas* de  $G$ , respectivamente. Denota-se  $|V| = n$ ,  $|E| = m$ . Quando for necessário especificar o grafo corrente, denota-se  $V(G)$  ou  $V_G$  e  $E(G)$  ou  $E_G$  ao conjunto de vértices e ao conjunto de arestas de  $G$ , respectivamente. Uma aresta é denotada pelo par  $(v,w)$  de vértices que a formam, ela é *incidente* aos vértices  $v$  e  $w$ . Dessa forma,  $v$  e  $w$  são os *extremos* da aresta e eles são *adjacentes*.

Caso interesse a ordem em cada aresta, então  $(v,w)$  é diferente de  $(w,v)$  e o grafo é chamado *grafo direcionado* ou *digrafo*  $\vec{G}(V,\vec{E})$ , diz-se que  $(v,w)$  está *orientada* de  $v$  a  $w$ . É costume falar de vértice ou *nó*, e de aresta ou *arco*, nesse caso.

Em geral, se fala de grafo em vez de grafo não direcionado, a não ser quando for necessário especificar.

Um grafo representa-se geometricamente associando cada vértice com um ponto distinto e cada aresta com uma linha unindo os pontos correspondentes a seus extremos.

O grafo  $\bar{G}(V,\bar{E})$  onde  $\bar{E} = \{(x,y) \in V \times V \mid x \neq y, (x,y) \notin E\}$  é o *grafo complemento* de  $G(V,E)$ .

Um grafo é *completo* se existe uma aresta entre cada par de seus vértices.

Para um elemento  $v$  de  $V$ , denota-se  $\text{Adj}(v)$  ao conjunto de vértices adjacentes a  $v$ . Chama-se *grau* do vértice  $v$  ao número de vértices adjacentes a ele, e denota-se  $\text{gr}(v)$ . Ou seja  $\text{gr}(v) = |\text{Adj}(v)|$ . Um vértice com grau zero é denominado *isolado*, um vértice com grau igual ao número de vértices é denominado *universal*.

Dados  $v_1$  e  $v_k$  em  $V$ , chama-se *caminho* de  $v_1$  a  $v_k$  a uma seqüência de vértices  $v_1, v_2, \dots, v_k$  tal que  $(v_j, v_{j+1}) \in E$ , para  $1 \leq j \leq k-1$  com  $k > 1$ . O *comprimento* do caminho é o número de arestas que o compõem. Um *ciclo* é um caminho  $v_1, v_2, \dots, v_k, v_{k+1}$  tal que  $v_{k+1} = v_1$  com  $k \geq 3$ . Uma *corda* de um caminho é uma aresta não pertencente ao caminho e que une dois vértices dele. Um *caminho* entre  $v$  e  $w$  sem cordas é dito *induzido*. Analogamente um *ciclo* é *induzido* se não possuir cordas. O caminho induzido e o ciclo induzido de  $k$  vértices denotam-se  $P_k$  e  $C_k$ , respectivamente.

Se um grafo não contém nenhum ciclo então diz-se que o grafo é *acíclico*.

Um *subgrafo*  $H(V',E')$  de um grafo  $G(V,E)$  é um grafo tal que  $V' \subseteq V$  e  $E' \subseteq E$ . Se  $E'$  for o conjunto de arestas de  $G$  que contêm todas as arestas existentes em  $E$  com ambos os extremos em  $V' = A$ , diz-se que  $H$  é o subgrafo induzido por  $A$  e denota-se  $H = G_A$ .

Um grafo é *conexo* se existe caminho entre cada par de seus vértices; caso contrário é *desconexo*. Um grafo sem arestas é chamado totalmente desconexo. Se um grafo é desconexo, chamam-se *componentes conexas* a todo subgrafo induzido dele que seja conexo e não esteja contido em nenhum outro subgrafo conexo.

Em um grafo conexo, denomina-se *distância*  $d(v,w)$  dos vértices  $v$  e  $w$  ao comprimento do menor caminho entre  $v$  e  $w$ .

Dado  $G(V,E)$  e um vértice  $v \in V$ , denota-se  $G - v$  ao grafo obtido ao retirar de  $G$  o vértice  $v$  e todas as arestas incidentes a  $v$ . O grafo  $G - H$ , obtido de  $G(V,E)$  pela remoção do subgrafo  $H(V',E')$ , é o subgrafo induzido por  $V - V'$ . Dado  $B(V^*,E^*)$ , o grafo  $G + B$  tem o conjunto de vértices  $V \cup V^*$  e o conjunto de arestas  $E \cup E^*$ .

Uma *árvore* é um grafo conexo acíclico. Distinguem-se dentro de uma árvore os vértices com grau igual a um, os quais são chamados de *folhas*. Uma árvore é denominada *enraizada* quando algum vértice  $r$  é escolhido como especial,  $r$  é chamado *raiz* da árvore. Dados dois vértices  $v$  e  $w$  de uma árvore enraizada  $T$  de raiz  $r$ ; se  $v$  pertence ao caminho de  $r$  a  $w$  em  $T$ , então  $v$  é *ancestral* de  $w$  e  $w$  é *descendente* de  $v$ . Se além disso  $(v,w) \in E$  então  $v$  é *pai* de  $w$  e  $w$  é *filho* de  $v$ . Se é relevante a ordem em que os filhos de cada vértice são considerados então a árvore é enraizada e *ordenada*. Uma *árvore binária* é uma árvore enraizada ordenada em que cada vértice possui, no máximo, dois filhos.

Dada  $T(V,E)$  uma árvore enraizada e  $v \in V$ , uma subárvore  $T_v$  de  $T$  é a árvore enraizada cuja raiz é  $v$ , definida pelo subgrafo induzido em  $T$  pelos descendentes de  $v$ .

Em um grafo direcionado, define-se o *grau de entrada*  $g_e(v)$  de um vértice  $v$  como o número de arestas que possuem  $v$  como extremo final; analogamente, o *grau de saída*  $g_s(v)$  de um vértice  $v$  como o número de arestas que possuem  $v$  como extremo inicial. Uma *fonte* é um vértice com grau de entrada nulo, enquanto que um *sumidouro* é um com grau de saída nulo.

Em uma árvore direcionada, para as folhas  $g_e(v) = 1$  e  $g_s(v) = 0$ ; se existe um e só um vértice com  $g_e(v) = 0$ , esse vértice é a raiz da árvore.

Dado um grafo direcionado  $D$ , diz-se que o vértice  $v$  *alcança* ou *atinge* o vértice  $w$  se existe algum caminho de  $v$  a  $w$  em  $D$ , sendo  $w$  *alcançável* ou *atingível* de  $v$ . A *redução transitiva*  $D_R$  de um dígrafo  $D$  é o menor dígrafo que preserva a alcançabilidade de  $D$ ; também é conhecida como *Diagrama de Hasse* de  $D$ .

Um *conjunto independente* ou *conjunto estável* de  $G$  é um subconjunto de vértices tal que não são adjacentes dois a dois.  $I$  é um *conjunto independente maximal* (c.i.m.) se  $I$  é um conjunto independente e todo vértice de  $V - I$  é adjacente a, pelo menos, um vértice de  $I$ . Um conjunto independente é *máximo* ou *maior* se não existe outro de cardinalidade maior. Todo conjunto independente máximo é maximal.

Uma *clique* é um subconjunto  $C$  de vértices tal que todo par de seus elementos é adjacente dois a dois.  $C$  é *maximal* se não existe outra clique  $C'$  de  $G$  que contenha  $C$  de forma própria. Uma clique é *máxima* ou *maior* se não existe outra de maior cardinal. Toda clique máxima é maximal.

O *tamanho* de uma clique máxima de  $G$  é denotado por  $\omega(G)$  e o de um maior conjunto independente de  $G$  por  $\alpha(G)$ . O *número* de conjuntos independentes maximais de  $G$  denota-se  $i(G)$ . A *família* de todos os conjuntos independentes maximais de  $G$  denota-se  $F(G)$ . Dessa forma,  $i(G) = |F(G)|$ .

Por exemplo, o grafo completo  $K_n$  tem uma só clique maximal que corresponde ao grafo todo, logo  $\omega(G) = n$ , cada vértice  $v_j$  induz um c.i.m.  $I_j = \{v_j\}$ , assim  $i(G) = n$ ,  $\alpha(G) = 1$ ,  $F(G) = \{ \{v_1\}, \{v_2\}, \dots, \{v_n\} \}$ . Por outra parte, o grafo totalmente desconexo  $I_n$  tem um só c.i.m.  $I$  que corresponde ao conjunto de vértices de  $I_n$ , cada vértice  $v_j$  induz uma clique maximal  $C_j = \{v_j\}$ , assim  $F(G) = \{ \{v_1, v_2, \dots, v_n\} \}$ ,  $i(G) = 1$ ,  $\omega(G) = 1$ ,  $\alpha(G) = n$ .

Um caso fácil é o grafo estrela com conjunto de vértices  $V = \{u, v_1, \dots, v_n\}$ , onde  $u$  é um vértice universal e cada vértice  $v_j$  só está ligado com  $u$ . Neste caso, toda aresta  $(u, v_j)$  corresponde a uma clique maximal,  $F(G) = \{ \{u\}, \{v_1, v_2, \dots, v_n\} \}$ ,  $\omega(G) = 2$ ,  $i(G) = 2$ ,  $\alpha(G) = n$ .

Um grafo  $G(V, E)$  é *bipartido* quando o seu conjunto de vértices  $V$  puder ser particionado em dois subconjuntos  $V_1, V_2$  tais que toda aresta de  $G$  une um vértice de  $V_1$  a outro de  $V_2$ ; denota-se  $V = V_1 \cup V_2$ .

Um *grafo grade completa*  $G_{n,p}$  tem como conjunto de vértices o conjunto de pares ordenados  $(i, j)$ , com  $1 \leq i \leq n$ ,  $1 \leq j \leq p$ . Existe uma aresta entre os vértices  $v = (v_i, v_j)$  e  $w = (w_i, w_j)$  se  $|v_i - w_i| = 1$  e  $|v_j - w_j| = 0$  ou  $|v_i - w_i| = 0$  e  $|v_j - w_j| = 1$ .

Um grafo  $G(V, E)$  é de *intervalo* se existe uma correspondência 1-1 entre o seu conjunto de vértices  $V$  e uma família de intervalos reais  $\{I_v\}_{v \in V}$ , de forma que, para todo par de vértices distintos  $u, v$  tem-se:  $(u, v) \in E$  se e somente se  $I_u \cap I_v \neq \emptyset$ ; isto é, os vértices estão ligados se os intervalos correspondentes se interceptam. A família  $\{I_v\}_{v \in V}$  é chamada representação de  $G$  por intervalos.

Um grafo  $G(V, E)$  é *arco-circular* se existe uma correspondência 1-1 entre o seu conjunto de vértices  $V$  e uma família  $A = \{A_v\}_{v \in V}$  de arcos de circunferência de um círculo, de forma que, para todo par de vértices distintos  $u, v$  tem-se:  $(u, v) \in E$  se e somente se  $A_u$  e  $A_v$  se sobrepõem; isto é, os vértices estão ligados se os arcos de circunferência correspondentes se interceptam.  $A$  é denominado modelo arco-circular de  $G$ .

Um grafo  $G(V, E)$  é *círculo* se existe uma correspondência 1-1 entre o seu conjunto de vértices  $V$  e uma família  $A = \{A_v\}_{v \in V}$  de arcos de cordas de um círculo, de forma que, para todo par de vértices distintos  $u, v$  tem-se:  $(u, v) \in E$  se e somente se as cordas  $A_u$  e  $A_v$  correspondentes se interceptam.

Um grafo  $G(V, E)$  é *triangularizado* ou *cordal* se cada ciclo de comprimento maior do que três possui uma corda.

Um grafo  $G(V, E)$  é de *comparabilidade* se admite uma orientação transitiva de suas arestas; isto é, uma orientação  $\vec{E}$  tal que no grafo direcionado  $\vec{G}(V, \vec{E})$  para todo  $x, y, z$  em  $V$  tem-se: se  $(x, y)$  e  $(y, z)$  estão em  $\vec{E}$  então necessariamente  $(x, z)$  está em  $\vec{E}$ .

Dada uma orientação acíclica  $\vec{E}$  de um grafo não direcionado  $G(V, E)$ , se existe um caminho dirigido  $v_i - v_j$  em  $G$ , então  $v_i$  é um *ancestral* de  $v_j$  e  $v_j$  é um *descendente* de  $v_i$ . denota-se  $\langle v_i, v_j \rangle$  ao conjunto de vértices que são simultaneamente descendentes de  $v_i$  e ancestrais de  $v_j$ .  $\vec{E}$  é denominada *localmente transitiva* se os subgrafos induzidos por  $\langle v_i, v_j \rangle$  são de comparabilidade. Nesse caso, diz-se que  $G$  é de *comparabilidade local*.

Um grafo  $G(V, E)$  é *separável por cliques* se  $G$  é tipo 1, isto é:  $V$  pode ser particionado em dois conjuntos  $V = V_1 + V_2$  com  $V_1 \cup V_2 = V$ ,  $V_1 \cap V_2 = \emptyset$  tal que  $G_{V_1}$  é um grafo bipartido conexo com  $\text{card}(V_1) \geq 3$ ,  $G_{V_2}$  é uma clique e em  $G$ , todo vértice de  $V_1$  é adjacente a todo vértice de  $V_2$ ; ou  $G$  é tipo 2, isto é:  $G$  é um grafo  $k$ -partido completo, para algum  $k$ ,  $V = V_1 + V_2 + \dots + V_k$  com  $V_i \cap V_j = \emptyset \forall i \neq j$ , todo  $V_i$  é um conjunto independente e  $\forall i \neq j$  todo vértice de  $V_i$  é adjacente a todo vértice de  $V_j$ ; ou  $G$  tem um corte de vértices  $K$  que é uma clique, isto é:  $G - K$  é desconexo e os subgrafos  $G_i$  induzidos por  $C_i \cup K$  são separáveis por cliques, onde  $C_i$  com  $i = 1, \dots, k$  são as componentes conexas de  $G - K$ .

O grafo  $G$  obtido por conexão série dos grafos  $G_1(V_1, E_1, v_{11}, v_{12})$  e  $G_2(V_2, E_2, v_{21}, v_{22})$  com  $V_1 \cap V_2 = \emptyset$ , onde  $v_{11}$  e  $v_{12}$  são os vértices terminais de  $G_1$ ,  $v_{21}$  e  $v_{22}$  são os vértices terminais de  $G_2$ , identificando os vértices  $v_{12}$  e  $v_{21}$ , é dado por:  $V_G = V_1 \cup V_2$ ,  $E_G = E_1 \cup E_2$ , com vértices terminais  $v_{11}$  e  $v_{22}$ . O grafo  $G$  obtido por conexão paralela dos grafos  $G_1(V_1, E_1, v_{11}, v_{12})$  e  $G_2(V_2, E_2, v_{21}, v_{22})$  com  $V_1 \cap V_2 = \emptyset$ , onde  $v_{11}$  e  $v_{12}$  são os vértices terminais de  $G_1$ ,  $v_{21}$  e  $v_{22}$  são os vértices terminais de  $G_2$ , é dado por:  $V_G = V_1 \cup V_2$ ,  $E_G = E_1 \cup E_2$ , com vértices terminais  $v_1$  e  $v_2$  ( $v_1 = v_{11} = v_{21}$ ,  $v_2 = v_{12} = v_{22}$ ).

Um grafo  $G$  é *série-paralelo a dois terminais* se ele pode ser obtido por um número finito de aplicações sucessivas de conexões série e/ou conexões paralelas, a partir do grafo série-paralelo básico ou mínimo  $G_b$ , consistente de dois vértices ligados por uma aresta simples.

Sejam  $S_1 = x_1, x_2, \dots, x_p$  e  $S_2 = y_1, y_2, \dots, y_q$  duas seqüências de inteiros; diz-se que  $S_1$  é lexicograficamente menor do que  $S_2$ , e denota-se  $S_1 <_L S_2$ , quando:

existe  $1 \leq j \leq p, q$  tal que  $x_k = y_k$  para todo  $k < j$  e  $x_j < y_j$ ,

ou então:  $p < q$  e para todo  $1 \leq k \leq p$  tem-se  $x_k = y_k$ .

## 1.2.2 Complexidade

Para exprimir analiticamente a complexidade de um algoritmo utiliza-se a notação  $O$  para limite superior e a notação  $o$  para limite inferior.

Dadas duas funções não negativas  $f$  e  $g$  de variável inteira positiva, diz-se que  $f$  é da ordem de  $g$ , denota-se  $f \approx O(g)$ , quando existem constantes  $c$  e  $n_0 > 0$  tais que  $f(n) \leq cg(n)$  para todo  $n \geq n_0$ . Se  $f(x)/g(x) \rightarrow 0$  quando  $x \rightarrow \infty$  então  $f(x) \approx o(g(x))$ . Se existem três constantes positivas  $a, b, c$  tais que  $a \leq |f(x)/g(x)| \leq b$  para  $|x| \geq c$  então  $f(x) = \Theta(g(x))$ .

Dado que o número de conjuntos independentes maximais de um grafo qualquer pode ser exponencial ( $i(G) \leq 3 n^{n/3}$ ), então ao avaliar a complexidade dos algoritmos para contar e enumerar os conjuntos independentes maximais, deve considera-se este fato. Johnson, Yannakakis e Papadimitriou (1988) consideram três tipos de complexidades de tempo:

(a) *Tempo total polinomial*: o tempo necessário para enumerar todas as configurações é limitado superiormente por um polinômio no tamanho do grafo e no número de c.i.m. do grafo.

(b) *Tempo incremental polinomial*: dadas várias configurações, o tempo para gerar a próxima é polinomial no tamanho do grafo e no tamanho dos conjuntos já obtidos.

(c) *Retardo polinomial*: o tempo necessário entre enumerar uma configuração e a seguinte está limitado superiormente por um polinômio no tamanho do grafo.



### 1.3. Conteúdo da tese

No presente trabalho, desenvolve-se algoritmos para construir a família de conjuntos independentes maximais no caso particular de caminhos induzidos, ciclos induzidos, grafos grade completas de duas linhas, grafos de intervalo, grafos triangularizados, grafos separáveis por cliques e grafos série-paralelo a dois terminais. Também se caracteriza o número de conjuntos independentes maximais no caso particular de caminhos induzidos, ciclos induzidos, grafos grade completas de duas linhas e grafos de intervalo.

No capítulo 2 resume-se os resultados, encontrados na literatura, relacionados com o número  $i(\mathbf{G})$  de conjuntos independentes maximais de um grafo  $\mathbf{G}$ .

No capítulo 3 descrevem-se quatro algoritmos, da literatura, para enumerar todos os conjuntos independentes maximais de um grafo qualquer.

No capítulo 4 apresenta-se os algoritmos desenvolvidos para construir a família de conjuntos independentes maximais no caso particular de caminhos induzidos, ciclos induzidos, grafos grade completas de duas linhas, grafos de intervalo e grafos triangularizados. Todos estes algoritmos baseiam-se na idéia de transformar o problema de construir a família de c.i.m do grafo dado no problema de determinar todos os caminhos maximais fonte-sumidouro num digrafo que depende da classe particular considerada.

Na seção 4.1 desenvolve-se um algoritmo para gerar todos os conjuntos independentes maximais de um *caminho induzido*, de tempo linear para cada conjunto independente maximal. Para esta classe de grafos tem-se um limite superior para o número de c.i.m. dado por Füredi (1987). Utilizando a mesma idéia dos caminhos induzidos, é possível gerar a família de c.i.m. de um *ciclo induzido* em tempo linear para cada c.i.m., apresentado na seção 4.2; Füredi (1987), também, apresenta um limite superior para o número de c.i.m. nesta classe de grafos. Em ambos casos, a complexidade total do algoritmo é polinomial no tamanho do grafo e no número de c.i.m. do grafo. Da mesma forma, na seção 4.3 desenvolve-se um algoritmo de tempo linear para cada c.i.m. de *grafos grade completas de duas linhas*; a complexidade total do algoritmo é polinomial no tamanho do grafo e no número de c.i.m. do grafo. Determina-se, além disso, a equação de recorrências que caracteriza o número de c.i.m. neste caso, e a solução dessa equação que dá o número de c.i.m. de uma grade completa de duas linhas (Villanueva, 1995).

Na seção 4.4, apresenta-se o algoritmo de Leung (1984) para gerar todos os conjuntos independentes maximais de um grafo de intervalo. Generalizando a idéia do algoritmo para gerar os c.i.m. de um caminho induzido, obtém-se um algoritmo para um *grafo de intervalo* com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele, no entanto a complexidade por c.i.m. é linear no tamanho do grafo. Este algoritmo é mais eficiente que o desenvolvido por Leung. Modificando esse algoritmo é possível contar o número de c.i.m. de um grafo de intervalo dado (Villanueva, 1993). Apresenta-se, também, o algoritmo de Leung (1984) para gerar todos os conjuntos independentes maximais de um grafo arco-circular para o qual ele utiliza o algoritmo de grafos de intervalo.

Na seção 4.5, descreve-se, o algoritmo desenvolvido por Leung (1984) para gerar todos os conjuntos independentes maximais de um grafo triangularizado. Generalizando a idéia do algoritmo para gerar os c.i.m. de um caminho induzido, obtém-se um algoritmo para um *grafo triangularizado* com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele; a complexidade por c.i.m. é linear no número de vértices, isto representa uma melhora em relação ao algoritmo de Leung.

No capítulo 5 trabalha-se com a estratégia de operar dois grafos para os quais dispõe-se da família de c.i.m. de cada um deles. Dependendo do tipo de operação nos dois grafos que compõem o grafo de interesse, constrói-se a família de c.i.m. dele. Na seção 5.1 considera-se a operação *conexão série* de dois grafos com um vértice comum a ambos, o qual é uma articulação do grafo resultante. Primeiro descreve-se o algoritmo desenvolvido por Boulala e Uhry (1979) para gerar um conjunto independente de peso máximo, desenvolve-se então um algoritmo para gerar a família de c.i.m. de um grafo obtido como conexão série de outros dois grafos, dadas as famílias de c.i.m. de esses dois grafos menores, com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele. Na seção 5.2 generaliza-se a conexão série, isto é a construção de um grafo com uma articulação ao caso de um *grafo com um conjunto articulação que é uma clique* primeiro descreve-se o algoritmo desenvolvido por Boulala e Uhry (1979) para gerar um conjunto independente de peso máximo; logo, obtém-se um algoritmo para gerar a família de c.i.m. de um grafo obtido como conexão série de outros dois grafos identificando uma clique comum em ambos, dadas as famílias de c.i.m. de esses dois grafos menores, com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele.

Na seção 5.3 desenvolve-se um algoritmo para gerar a família de c.i.m. de um grafo separável por cliques, aproveitando a árvore de decomposição por cliques e a operação de unir dois grafos por um conjunto comum a ambos (Villanueva, 1994).

Na seção 5.4 considera-se a operação *conexão paralela* de dois grafos com dois vértices comuns a ambos, os quais são denominados vértices terminais do grafo resultante; primeiro descreve-se o algoritmo desenvolvido por Boulala e Uhry (1979) para gerar um conjunto independente de peso máximo. Logo, desenvolve-se um algoritmo para gerar a família de c.i.m. de um grafo obtido como conexão paralela de outros dois grafos, dadas as famílias de c.i.m. de esses dois grafos menores, com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele.

Na seção 5.5 desenvolve-se um algoritmo para gerar a família de c.i.m. de um *grafo série-paralelo a dois terminais*, aproveitando a árvore de decomposição do grafo e as operações de unir dois grafos por conexão série e por conexão paralela.

No capítulo 6, apresenta-se as conclusões deste trabalho. Inclui-se um quadro resumo dos algoritmos encontrados na literatura e os algoritmos propostos aqui, em classes particulares de grafos, para enumerar todos os conjuntos independentes maximais de um grafo dado, junto com sua complexidade e o(os) autor(es). Além disso, indicam-se algumas linhas de pesquisa em relação ao problema estudado neste trabalho.

Finalmente, no capítulo 7 se da a lista de referências bibliográficas citadas no trabalho.

## Capítulo 2

### CONTAGEM DE CONJUNTOS INDEPENDENTES MAXIMAIS

O problema de contar todas as configurações que satisfazem uma determinada propriedade é freqüente em combinatória. Em particular, em Teoria de Grafos, o problema de contar todos os conjuntos independentes maximais de um grafo dado tem captado a atenção e estudo dos pesquisadores. Füredi (1987), Griggs, Grinstead e Guichard (1988), Liu (1994) apresentam limites superiores para este número em um grafo qualquer. Wilf (1986), Sagan (1988), Meir e Moon (1988) consideram o caso de árvores. Hujter e Tuza (1993) estudam os grafos sem triângulos. Liu (1993) se preocupa dos grafos bipartidos.

Dada a relação entre conjunto independente e clique, o problema de determinar o número de conjuntos independentes maximais de um grafo qualquer pode reduzir-se ao problema de determinar o número de cliques maximais do grafo. Moon e Moser (1965) estudam este número para um grafo qualquer; Szwarcfiter e Barroso (1991) provam que o número de cliques maximais de um grafo círculo pode ser calculado em tempo  $O(nm)$ .

Neste capítulo apresenta-se resultados da literatura, relativos ao número de conjuntos independentes maximais de um grafo. Considera-se, primeiro, o caso geral de um grafo qualquer. Em seguida, o caso particular de árvores, para as quais Wilf (1986) apresenta um limite superior e desenvolve um algoritmo de tempo linear para calcular esse número. Finalmente, considera-se outros casos particulares, para os quais dispõe-se de um limite superior do número de c.i.m. do grafo.

#### 2.1. Limite superior para o número de c.i.m.

Moon & Moser (1965) determinam o número máximo  $m(n)$  de cliques maximais em um grafo com  $n$  vértices. Este número é:

$$m(n) = \begin{cases} 3^{n/3} & \text{se } n = 0(\text{mod } 3) \\ 4 \times 3^{\lfloor n/3 \rfloor - 1} & \text{se } n = 1(\text{mod } 3) \\ 2 \times 3^{\lfloor n/3 \rfloor} & \text{se } n = 2(\text{mod } 3) \end{cases} \quad \text{ou} \quad m(n) = \begin{cases} 3^t & \text{se } n = 3t \\ 4 \times 3^{t-1} & \text{se } n = 3t + 1 \\ 2 \times 3^t & \text{se } n = 3t + 2 \end{cases}$$

Como corolário tem-se que para todo  $G(V,E)$  com  $\text{card}(V) = n$ ,  $i(G) \leq 3n/3$ .

Eles apresentam os grafos extremos que atingem este número, figura 2.1, onde  $K_j$  indica o grafo completo de  $j$  vértices,  $m$  é o único inteiro que satisfaz

$$n = 2^m + 2m + [\log m] + (1 + 3) \text{ com } 0 \leq l \leq 2^m,$$

$$t = [\log m] + 1,$$

$$h = 2^{m-1} - 2^t - t + 1;$$

denomina-se A, B e C aos vértices da primeira, segunda e terceira coluna, respectivamente, e D ao conjunto de  $2^{m-1} + 1$  vértices dentro do retângulo na coluna B. Cada vértice de A é unido a todos os outros vértices de A, em forma análoga para B e C. Cada vértice  $a$  tipo A é unido a cada vértice de B não contido num grafo completo ligado por uma linha com  $a$ . Cada vértice  $c$  tipo C é unido a cada vértice de D não contido num grafo completo ligado por uma linha com  $c$ .

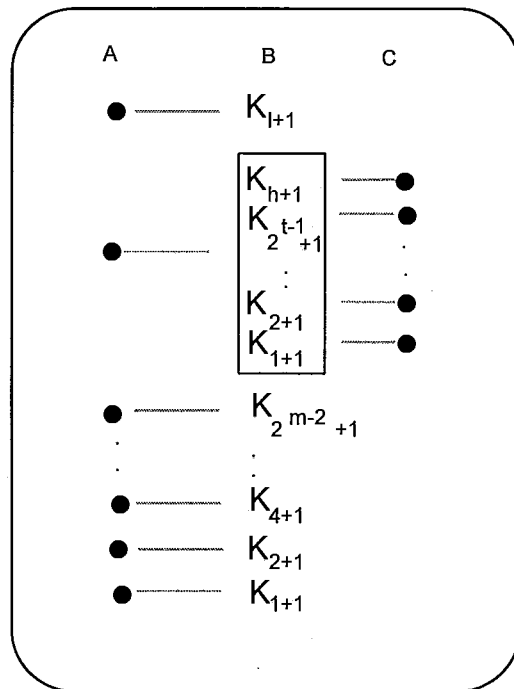


Fig. 2.1: Grafos extremos de Moon & Moser

Estes grafos correspondem aos grafos conexos de  $n$  vértices que possuem o maior número de cliques maximais e este numero é  $m(n)$ . Esses grafos são grafos círculo, isto é, o grafo interseção de  $n$  cordas de um círculo, onde a cada corda associa-se um vértice e dois vértices são adjacentes se as cordas correspondentes se interceptam.

Considerando que  $C$  é clique em  $G$  se e somente se  $C$  é conjunto independente em  $\bar{G}$ , então o número máximo de conjuntos independentes maximais de um grafo de ordem  $n$  é  $m(n)$  e os grafos extremos que atingem este número são  $H_n$ , apresentados na figura 2.2, que correspondem aos complementos dos grafos de Moon & Moser e não são conexos.

$$H_n = \begin{cases} tK_3 & \text{se } n = 3t \\ (t-1)K_3 + 2K_2 \text{ o } (t-1)K_3 + K_4 & \text{se } n = 3t + 1 \\ tK_3 + K_2 & \text{se } n = 3t + 2 \end{cases}$$

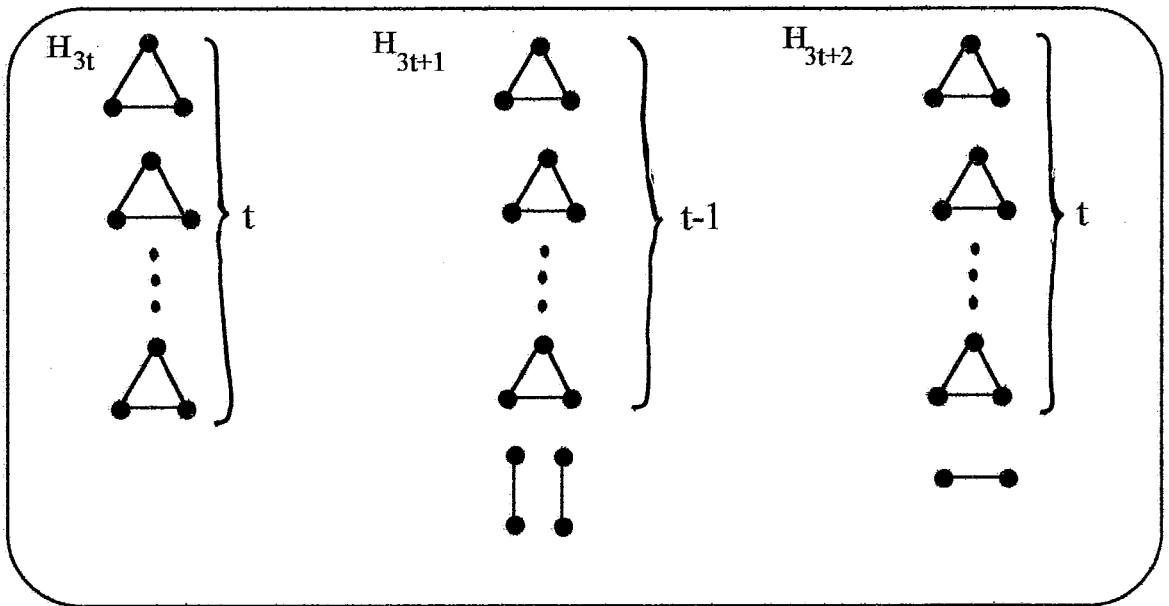


Fig. 2.2: Grafos complementos de grafos extremos de Moon & Moser

Füredi (1987) prova que, dado  $G$  com  $n$  vértices então: ou  $G$  é isomorfo a  $H_n$  e portanto  $i(G) = m(n)$  ou  $i(G) \leq r(n)$ , onde  $r(n)$  limita o número de conjuntos independentes maximais em um grafo de  $n$  vértices, dado pela função

$$r(n) = \begin{cases} m(n-1) & \text{se } 2 \leq n \leq 5 \\ 8x3^{t-3} & \text{se } n = 3t \geq 6 \\ 11x3^{t-3} & \text{se } n = 3t + 1 \geq 6 \\ 16x3^{t-2} & \text{se } n = 3t + 2 \geq 6 \end{cases}$$

Os grafos extremos que atingem este limite com  $i(G) = r(n)$  são:  $F_2 = 2 K_1$  com  $i(F_2)=2$ ,  $F_3 = P_3$  ou  $K_2 + K_1$  com  $i(F_3) = 2$ ,  $F_4 = P_4$  ou  $K_3 + K_1$  com  $i(F_4) = 3$ ,  $F_5 = C_5$  com  $i(F_5) = 5$ ,  $F_6 = 3 K_2$  com  $i(F_6) = 8$ ,  $F_7 = C_5 + K_2$  com  $i(F_7) = 10$ ; apresentados na figura 2.3, e para  $n \geq 8$  tem-se

$$F_n = \begin{cases} (t-2) K_3 + 3 K_2 & \text{se } n = 3t \\ (t-3) K_3 + 5 K_2 & \text{se } n = 3t + 1 \\ (t-2) K_3 + 4 K_2 & \text{se } n = 3t + 2, \end{cases}$$

mas eles não são conexos.

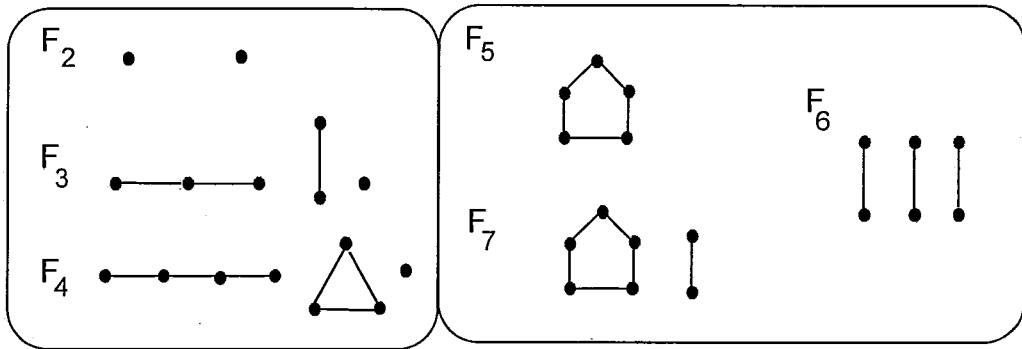


Fig. 2.3: Grafos extremos  $F_2, \dots, F_7$  de Füredi

Füredi apresenta, também, o seguinte limite superior  $c_0(n)$  para  $i(G)$  no caso que  $G$  é conexo de ordem  $n \geq 50$ :

$$c_0(n) = \begin{cases} 2x 3^{t-1} + 2^{t-1} & \text{se } n = 3t \\ 3^t + 2^{t-1} & \text{se } n = 3t + 1 \\ 4x 3^{t-1} + 3x 2^{t-2} & \text{se } n = 3t + 2. \end{cases}$$

Além disso, conjectura que é válido para todo  $n$  e constrói os grafos extremos  $T_n$ , apresentados na figura 2.4, que correspondem à versão conexa dos grafos  $H_n$  de Moon e Moser, da forma seguinte:

Para  $n = 3t$ :

Seja  $G = t K_3$ ,  $x \in V$ .

Unir  $x$  com uma aresta a cada uma das  $t-1$  componentes conexas de  $G$ .

Para  $n = 3t + 1$ :

Seja  $G = K_1 + t K_3$ ,  $x$  o vértice isolado.

Unir  $x$  através de uma aresta a  $t-1$  copias de  $K_3$  e com três arestas à outra componente  $K_3$ .

Para  $n = 3t + 2$ :

Seja  $G = K_4 + (t-1) K_3 + K_1$ ,  $x$  o vértice isolado.

Unir  $x$  com três arestas a uma componente  $K_3$  e com uma aresta a cada uma das outras componentes conexas.

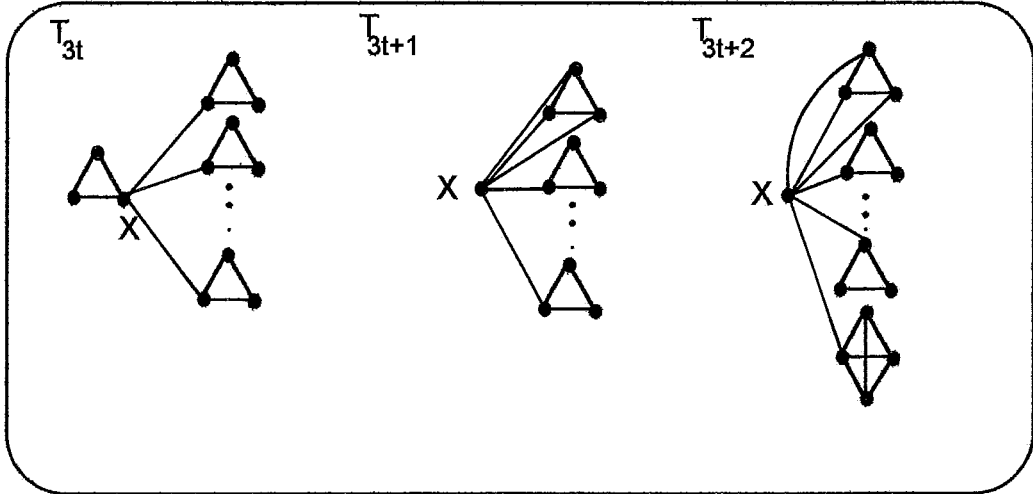


Fig. 2.4: Grafos extremos de Füredi

Ele apresenta como exemplos o caso do caminho sem cordas  $P_n$  e do ciclo sem cordas  $C_n$ . Tem-se  $i(P_n) \leq 2a^{n-2}$ ,  $i(C_n) \leq 3a^{n-3}$ , onde  $a$  é a única solução real da equação  $1+a=a^3$  e  $n$  é o número de vértices.

Ele prova que: dado um grafo  $G$  conexo com  $n$  vértices e grau máximo  $\Delta(G) \leq 6$ , então  $i(G) \leq 3^{n/3} 1.009^{-n+3}$ .

Por outro lado Griggs, Grinstead e Guichard (1988) determinam um limite superior para o número  $i(G)$  de conjuntos independentes maximais de um grafo conexo qualquer  $G$  com  $n$  vértices e caracterizam os grafos extremos que atingem este limite. Eles apresentam a seguinte função limite superior para  $i(G)$ :

$$h(n) = \begin{cases} n & \text{se } n < 6 \\ 2x3^{n/3-1} + 2^{n/3-1} & \text{se } n \geq 6, n = 0(\text{mod } 3) \\ 3\lfloor n/3 \rfloor + 2\lfloor n/3 \rfloor - 1 & \text{se } n \geq 6, n = 1(\text{mod } 3) \\ 4x3\lfloor n/3 \rfloor - 1 + 3x2\lfloor n/3 \rfloor - 2 & \text{se } n \geq 6, n = 2(\text{mod } 3) \end{cases}$$

que coincide com o limite  $c_o(n)$  determinado por Füredi, considerando  $t = \lfloor n/3 \rfloor$ .

Se  $n \leq 4$  o grafo completo  $K_n$  de  $n$  vértices é o único grafo conexo com  $h(n)$  conjuntos independentes maximais, neste caso  $i(K_n) = h(n) = n$ .

Se  $n = 5$   $K_5$ ,  $C_5$  e os dois grafos da figura 2.5 têm  $h(n)$  c.i.m. com  $i(G) = h(n) = 5$ .

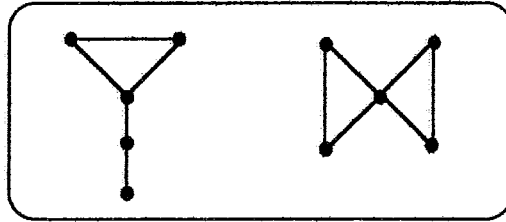


Fig. 2.5: Grafos com  $i(G) = 5$

Se  $n \geq 6$ , eles constroem os grafos extremos com  $i(G) = h(n)$ , apresentados na figura 2.6, segundo a seguinte regra:

Se  $n \equiv 0 \pmod{3}$ :

Seja  $G = n/3 K_3$ ,  $v$  um vértice qualquer de um  $K_3$  qualquer.  
 Unir  $v$  com uma aresta a cada uma das outras  $(n/3-1)$  componentes de  $G$ .  
 O grafo resultante é igual ao grafo  $T_{3t}$  construído por Füredi neste caso.

Se  $n \equiv 1 \pmod{3}$ :

Seja  $G = (n-4)/3 K_3 + K_4$ ,  $v$  um vértice de  $K_4$ .  
 Unir  $v$  com um vértice de cada  $K_3$ .  
 O grafo resultante é igual ao grafo  $T_{3t+1}$  construído por Füredi, o  $K_4$  é obtido pelo  $K_1$  e o  $K_3$  ao qual é ligado com três arestas.

Se  $n \equiv 2 \pmod{3}$ :

Seja  $G = (n-8)/3 K_3 + 2 K_4$ ,  $v$  um vértice de um dos  $K_4$ .  
 Unir  $v$  com um vértice do outro  $K_4$  e com um vértice de cada  $K_3$ .  
 O grafo resultante é o grafo  $T_{3t+2}$  construído por Füredi, ao unir o vértice isolado com um  $K_3$  através de três arestas forma-se um  $K_4$ .

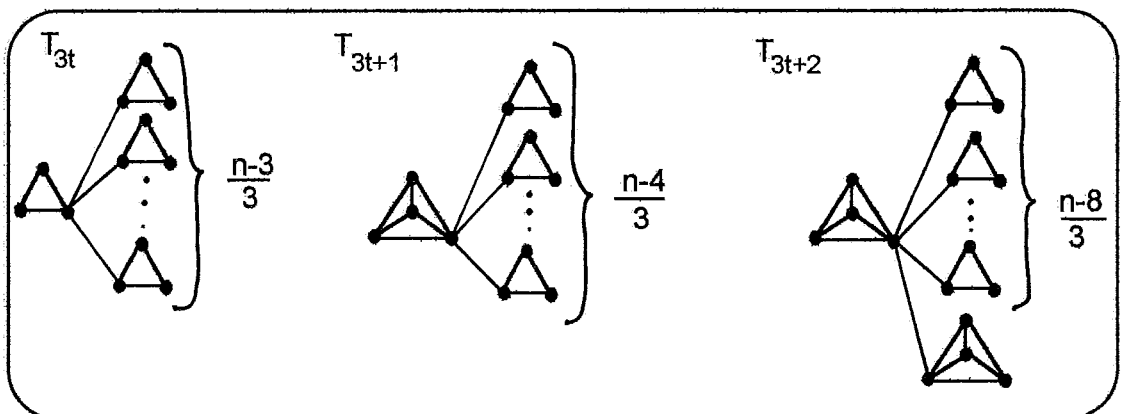


Fig. 2.6: Grafos extremos de Griggs, Grinstead e Guichard



Além disso, Griggs, Grinstead e Guichard provam que, para cada  $n$ , este é o único grafo conexo com  $h(n)$  c.i.m..

Liu (1994) prova duas conjecturas relativas ao maior número de c.i.m. de um grafo de ordem  $n$ , formuladas por P. Erdős quem junto com Moser foram os primeiros a perguntar pelo número máximo de cliques maximais de um grafo e os grafos extremos com este número.

Considerando  $i(\emptyset) = 1$ , Liu prova que

$$i(G) \leq i(G - v) + i(G - v - \text{Adj}(v)) \quad \forall v \in V.$$

Ele generaliza o limite dado por Füredi provando que, dado  $c \in \mathbf{Z}^+ \exists t > 0, t \in \mathbf{R}$  tal que: se  $G$  é um grafo conexo de  $n$  vértices com grau máximo  $\Delta(G) \leq c$  então

$$i(G) \leq 3^{n/3} (1+t)^{-n+3}.$$

Ele prova, também, que  $\exists M \in \mathbf{Z}^+$  e  $\exists c(M) \in \mathbf{Z}^+$  tal que: se  $G$  é conexo de ordem  $n$  com grau máximo  $\Delta(G) \geq c(M)$  então  $i(G) \leq 3^{n/3} (1 + (1/M \Delta(G)))^{-n+3}$ .

Liu prova a primeira conjectura que diz:

Para uma família  $\Phi = \{ G / G \text{ é conexo} \}$ , se  $\Delta_\Phi(n) = o(n)$  então  $i_\Phi(n) = o(3^{n/3})$ ,

dada a função  $\Delta_\Phi: \mathbf{Z}^+ \rightarrow \mathbf{Z}^+ \cup \{0\}$  por  $\Delta_\Phi(n) = \max\{\Delta(G) / G \in \Phi, |V(G)| = n\}$ .

Na figura 2.7 exibe-se um grafo  $G$  de  $n$  vértices com  $i(G) = \Theta(3^{n/3})$ , para  $k > 0$  inteiro qualquer considera-se  $n = (3n_1 + 1)k$ , tem-se  $\Delta(G) = n_1 + 2 = (n - k) / (3k) + 2$ .

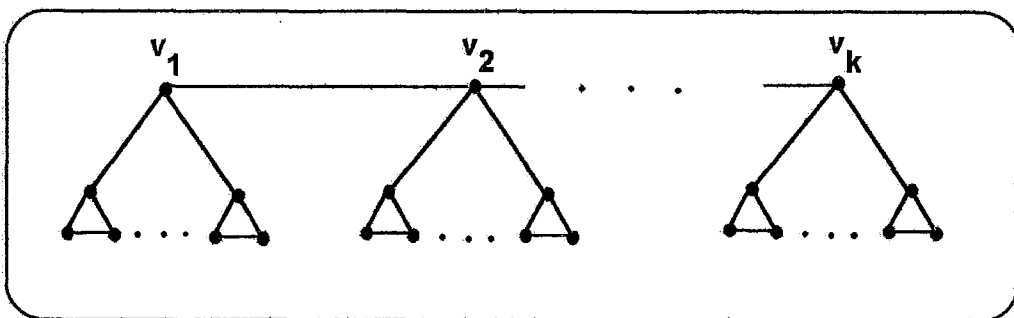


Fig. 2.7: Grafo de Liu

A segunda conjectura de Erdős é:

Se  $\Phi$  é uma família de grafos tal que  $l_{\Phi}(n) = \infty$  se  $n \rightarrow \infty$  então  $i_{\Phi}(n) = o(3^{n/3})$ , onde  $l_{\Phi}(n)$  é o comprimento de um caminho mais longo.

Liu prova o seguinte resultado que é mais forte:

Existe um número real  $t > 0$  tal que se  $G$  é um grafo de ordem  $n$  então

$$i(G) \leq 3^{n/3} (1+t)^{-d+3}$$

onde  $d$  é o comprimento de um caminho mais longo de  $G$ .

Para provar esta propriedade precisa-se do seguinte fato:

Se  $H$  é um subgrafo induzido de  $G$  então  $i(H) \leq i(G)$ ,

e da seguinte igualdade:

Para um grafo  $G(V,E)$ ,  $x, y \in V$ , se  $x$  e  $y$  são adjacentes então

$$i(G(x;y)) = i(G) + i(y) - i(x) + \beta(x)$$

onde:  $G(x;y)$  é o grafo obtido ao retirar de  $G$  as arestas incidentes a  $x$  exceto  $(x,y)$  e acrescentar as arestas unindo  $x$  a cada vértice de  $Adj(y) - \{x\}$ ,

$i(x)$  denota o número de c.i.m. de  $G$  contém  $x$ ,

$\beta(x)$  denota o número de c.i.m. em  $G - x$  que estão contidos em  $G - x - Adj(x)$ .

Dado qualquer inteiro positivo  $c$ , seja  $G = K_{c+1} + H_{n-c-1}$ , com  $H_n$  grafo complemento dos grafos extremos de Moon e Moser, então  $G$  tem um caminho mais longo de comprimento  $c$  e  $i(G) = \Theta(3^{n/3})$ .

## 2.2. Número de c.i.m. para árvores

Wilf (1986) determina o número máximo de conjuntos independentes maximais que uma árvore de  $n$  vértices pode ter. Este é dado pela função

$$f(n) = \begin{cases} 2^{n/2-1} & \text{se } n \geq 2 \text{ par} \\ 2^{(n-1)/2} & \text{se } n \text{ ímpar} \\ 1 & \text{se } n = 0 \end{cases}$$

Ele exhibe as árvores extremas que atingem este número, apresentadas na figura 2.8.

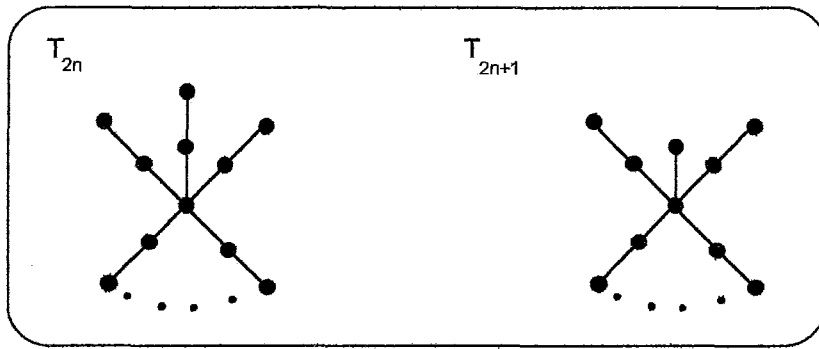


Fig. 2.8: Árvores extremas de Wilf

Sagan (1988) apresenta uma prova mais simples que a dada por Wilf, para o número de c.i.m. de uma árvore dada com  $n$  vértices. Ele caracteriza as árvores que atingem este limite como os bastões de comprimento 0 se  $n$  é ímpar e os bastões de comprimento 1 e 3 se  $n$  é par.

Um *bastão* de comprimento 1 é obtido da seguinte construção:

- 1) começar com um caminho  $P$  de 1 vértices
- 2) acrescentar caminhos de comprimento dois a vértices de  $P$  com grau 1.

Na figura 2.9 exibem-se os primeiros elementos da família de bastões de comprimento 0.

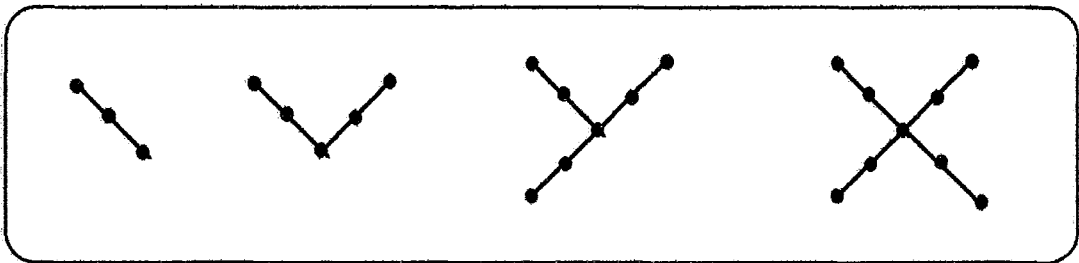


Fig. 2.9: Bastões de comprimento 0

Na figura 2.10 encontram-se alguns elementos da família de bastões de comprimento 1.

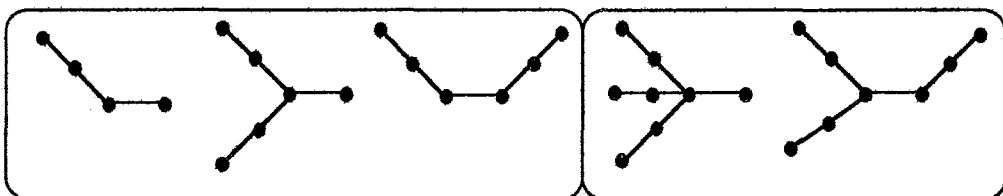


Fig. 2.10: Bastões de comprimento 1

Ele, também, prova que o mínimo valor de  $i(T)$  dentre todas as árvores  $T$  de  $n$  vértices é 2 e a única árvore que o atinge é a árvore-estrela  $B_{1,n-1}$  da figura 2.11.

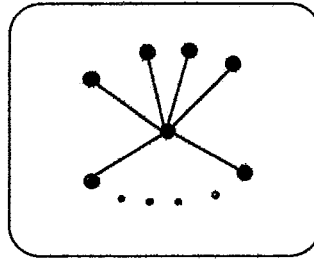


Fig. 2.11: Árvore-estrela  $B_{1,n-1}$

Meir e Moon (1988) estudam o comportamento assintótico do número médio  $e(n)$  de c.i.m. e do tamanho médio  $\mu(n)$  dos c.i.m., numa árvore enraizada de  $n$  vértices, considerando as árvores pertencentes a certas famílias. Eles determinam que:

$$e(n) \sim q E^n \quad \text{se } n \rightarrow \infty$$

$$\mu(n) \sim S n \quad \text{se } n \rightarrow \infty$$

onde  $q$ ,  $E$  e  $S$  são constantes que dependem da família de árvores considerada. Por exemplo, para a família de árvores binárias tem-se:

$$\lim_{k \rightarrow \infty} [e(2k+1)]^{1/(2k+1)} = 1,2499\dots$$

$$\lim_{k \rightarrow \infty} \frac{\mu(2k+1)}{2k+1} = 0,4824\dots$$

Zito (1991) prova que, o número de conjuntos independentes máximos de uma árvore de  $n$  vértices está limitado por

$$i_{\alpha}(T_n) = \begin{cases} 2^{(n-3)/2} & \text{se } n > 1 \text{ ímpar} \\ 1 + 2^{(n-2)/2} & \text{se } n \text{ par} \end{cases}$$

e constrói as árvores extremas que atingem este número, apresentadas na figura 2.12.

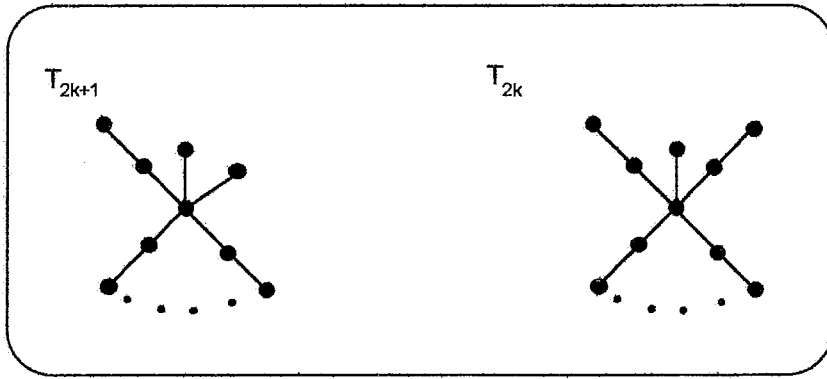


Fig. 2.12: Árvores extremas de Zito

Wilf (1986), também, desenvolve um algoritmo de tempo linear para calcular o número de conjuntos independentes maximais de uma dada árvore  $T(V,E)$ . A idéia do algoritmo é calcular para cada vértice  $v$  de  $T$  o número de c.i.m. da subárvore  $T_v$  enraizada em  $v$ , começando pelas folhas e em ordem não crescente da distância da raiz a  $v$ .

Seja  $r$  a raiz de  $T$ , consideram-se as arestas de  $T$  orientadas de forma que a raiz é fonte e as folhas são sumidouros. Para cada vértice  $v$  de  $T$  define-se:

$$F(v) = \{ \text{filhos de } v \} = \{ w \in V / (v,w) \in \vec{E} \}$$

$$N(v) = \{ \text{netos de } v \} = \{ z \in V / \exists w \in F(v) \text{ e } (w,z) \in \vec{E} \}$$

$$\mu_v = |\{ \text{c.i.m. de } T_v \}| = i(T_v)$$

$$\nu_v = |\{ \text{c.i.m. de } T_v / v \text{ não pertence ao c.i.m.} \}|$$

Tem-se:

$$\nu_v = \prod_{w \in F(v)} \mu_w - \prod_{w \in F(v)} \nu_w$$

$$\mu_v = \nu_v + \prod_{z \in N(v)} \mu_z$$

$$\mu_f = 1, \nu_f = 0 \text{ para } f \text{ folha}$$

$$\mu_r = i(T)$$

$$\nu_r = i(T(r \notin I))$$

$$\prod_{z \in N(r)} \mu_z = i(T(r \in I)).$$

Para implementar o algoritmo precisa-se do vetor  $d$  de distâncias da raiz  $r$  de  $T$  aos vértices  $v_1, v_2, \dots, v_n$  numerados em forma não crescente de  $d(v)$  com  $v_n = r$ . A cada vértice  $v$  associa-se um rótulo  $(\mu_v, \upsilon_v)$ .

O método de Wilf é implementado segundo o algoritmo 2.1 descrito a seguir:

**Algoritmo 2.1: Contagem dos c.i.m. de uma árvore**

Dada a árvore  $T(V,E)$  enraizada em  $r$ ,  $|V| = n$ ,  $|E| = m$ , o vetor  $d$  de distâncias à raiz

1. introduzir um filho fictício a cada folha
2. para todo  $v$  vértice fictício efetuar
  - 2.1 rotular  $v$  com  $(1,0)$
  - 2.2  $\mu_v = 1$   
 $\upsilon_v = 0$
3. para  $j = 1$  até  $n$  efetuar
  - 3.1 se  $v_j$  é folha então rotular  $v_j$  com  $(1,0)$
  - 3.2 caso contrário efetuar
    - $F(v_j) = \{ \text{filhos de } v_j \}$
    - $N(v_j) = \{ \text{netos de } v_j \}$
    - $A = 1$
    - $B = 1$
    - para todo  $w \in F(v_j)$  efetuar
      - $A = A \times \mu_w$
      - $B = B \times \upsilon_w$
    - $\upsilon_{v_j} = A - B$
    - $A = 1$
    - para todo  $w \in N(v_j)$  efetuar
      - $A = A \times \mu_w$
      - $\mu_{v_j} = \upsilon_{v_i} + A$
    - rotular  $v_j$  com  $(\mu_{v_j}, \upsilon_{v_j})$ .

***Complexidade:***

A complexidade total do algoritmo 2.1 é  $O(nm)$ , isto é tem tempo polinomial no tamanho do grafo.

Por exemplo, para a árvore  $T$  da figura 2.13 tem-se:

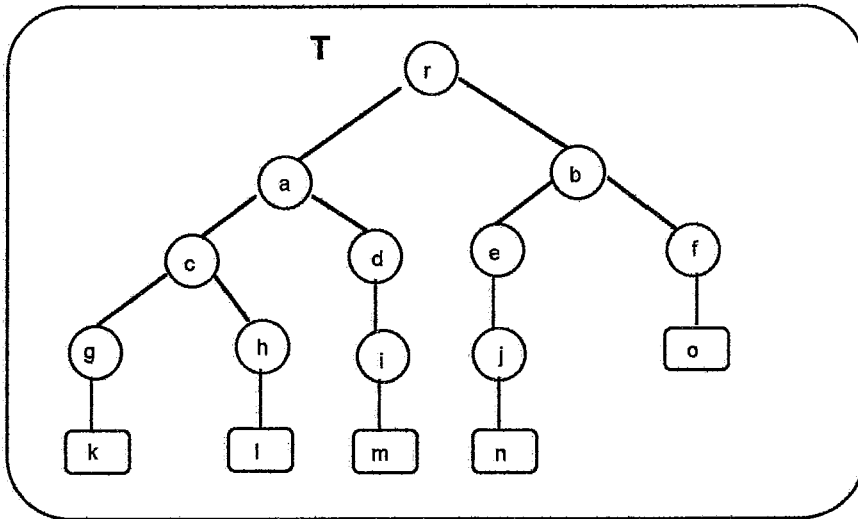


Fig. 2.13: Exemplo contagem de c.i.m. em árvores (algoritmo de Wilf)

v	d(v)
r	0
a	1
b	1
c	2
d	2
e	2
f	2
g	3
h	3
i	3
j	3

vértices fictícios: **k, l, m, n, o**

ordem nos vértices segundo d(v): **j, i, h, g, f, e, d, c, b, a, r**

v	$u_v$	$\mu_v$
o	0	1
n	0	1
n	0	1
l	0	1
k	0	1
j	0	1
i	0	1
h	0	1
g	0	1
f	0	1
e	1	2
d	1	2
c	1	2
b	2	3
a	3	4
r	6	14

Efetivamente, a família de c.i.m. da árvore  $\mathbf{T}$  é:

$$F(\mathbf{T}) = \{ \{r,c,d,e,f\}, \{r,c,d,f,j\}, \{r,c,e,f,i\}, \{r,d,e,f,g,h\}, \{r,c,f,i,j\}, \{r,e,f,g,h,i\}, \{r,d,f,g,h,j\}, \\ \{r,f,g,h,i,j\}, \{a,b,g,h,i,j\}, \{a,e,f,g,h,i\}, \{a,f,g,h,i,j\}, \{b,c,d,j\}, \{b,c,i,j\}, \{b,d,g,h,j\} \}$$

logo  $i(\mathbf{T}) = 14$ .

### 2.3. Número de c.i.m. para casos particulares

Liu (1993) determina que, o número máximo de c.i.m. de um grafo bipartido com  $n$  vértices é  $b(n) = 2^{\lfloor n/2 \rfloor}$ , e os únicos grafos extremos que atingem este limite são os grafos

$$B'_{n,r} = \begin{cases} tK_2 & \text{se } n = 2t \\ rK_2 + T_{2(t-r)+1} & \text{se } n = 2t + 1 \end{cases}$$

onde  $n \geq 2$ ,  $0 \leq r \leq \lfloor n/2 \rfloor$  e  $T_k$  são as árvores extremas de Wilf.

Ele prova que no caso de grafos bipartidos conexos, o numero máximo de c.i.m. é o limite dado por Wilf para as árvores e os únicos grafos extremos que atingem esse limite são  $T_n$  no caso acíclico e no caso em que os grafos possuem ciclos são os grafos  $B_n$  da figura 2.14 com  $r \geq 3$  e  $t \geq 2$ .



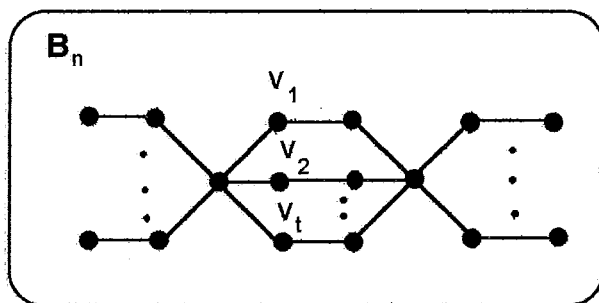


Fig. 2.14: Grafos bipartidos conexos com ciclos extremos

Hujter e Tuza (1993) provam que: se  $G$  é um grafo sem triângulos com  $n \geq 4$  vértices então

$$i(G) \leq \begin{cases} 2^{n/2} & \text{se } n \text{ par} \\ 5 \times 2^{(n-5)/2} & \text{se } n \text{ ímpar} \end{cases}$$

A igualdade é válida se e somente se  $G$  é isomorfo ao grafo  $E_n$  apresentado na figura 2.15, onde

$$E_n = \begin{cases} \frac{n}{2} K_2 & \text{se } n \text{ par} \\ C_5 + (n-5) K_1 & \text{se } n \text{ ímpar} \end{cases}$$

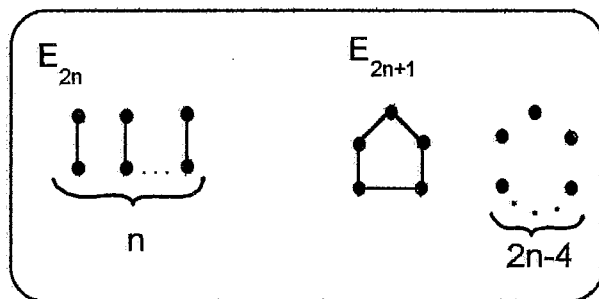


Fig. 2.15: Grafos sem-triângulos extremos.

## Capítulo 3

### GERAÇÃO DE CONJUNTOS INDEPENDENTES MAXIMAIS

O problema de gerar todas as configurações que satisfazem uma determinada propriedade é freqüente em combinatória. Em particular, em Teoria de Grafos, o problema de gerar todos os conjuntos independentes maximais de um grafo dado tem captado a atenção e estudo dos pesquisadores. Os algoritmos de Paull e Unger (1959), Tsukiyama et alii (1977), Loukakis e Tsouros (1981), Loukakis (1983) e Johnson et alii (1988) resolvem este problema.

Dada a relação entre conjunto independente e clique, o problema de gerar a família de conjuntos independentes maximais de um grafo qualquer pode reduzir-se ao problema de gerar a família de cliques maximais do grafo. Este problema pode ser resolvido com os algoritmos de Marcus (1964), Mulligan e Corneil (1972), Akkoyunlu (1973), Bron e Kerbosch (1973) e Osteen (1974).

Neste capítulo são apresentados quatro algoritmos da literatura, para gerar a família de conjuntos independentes maximais de um grafo qualquer. Os algoritmos considerados são os de Paull e Unger (1959), Tsukiyama et alii (1977), Loukakis (1983) e Johnson et alii (1988).

#### 3.1. Algoritmo de Paull e Unger (1959) (descrito em Lawler, 1976b)

Dado o grafo  $G(V,E)$  com conjunto de vértices  $V = \{v_1, v_2, \dots, v_n\}$ , Paull e Unger fazem uma construção recursiva para gerar a família  $F(G)$  de c.i.m. de  $G$ .

Eles definem:

$F_j = \{ \text{família de c.i.m. do subgrafo } G_j \text{ induzido por } \{v_1, \dots, v_j\} \}$   
com isto  $F(G) = F_n$ .

Por exemplo, para o grafo  $G$  da figura 3.1 com conjunto de vértices  $V = \{1,2,3,4,5\}$ , tem-se  $F_1 = \{ \{1\} \}$ ,  $F_2 = \{ \{1\}, \{2\} \}$ ,  $F_3 = \{ \{1,3\}, \{2\} \}$ ,  $F_4 = \{ \{1,3\}, \{1,4\}, \{2\} \}$ ,  
 $F_5 = \{ \{1,3\}, \{1,4\}, \{1,5\}, \{2,5\} \}$ .

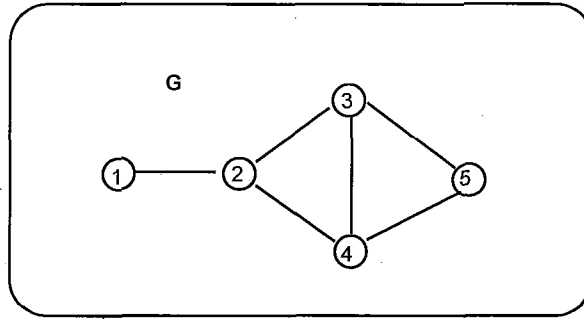


Fig. 3.1: Exemplo do Lawler

A idéia é construir  $F_{j+1}$  a partir de  $F_j$ .

**Observação 3.1:**

1.- Dado um c.i.m.  $I$  em  $F_j$ ;

se  $I \cap \text{Adj}(v_{j+1}) = \emptyset$  então  $v_{j+1}$  pode ser agregado a  $I$ , logo  $I \cup \{v_{j+1}\} \in F_{j+1}$ ,

se  $I \cap \text{Adj}(v_{j+1}) \neq \emptyset$  então  $v_{j+1}$  não pode ser agregado a  $I$  e portanto  $I \in F_{j+1}$ .

2.- Dado um c.i.m.  $I'$  em  $F_{j+1}$ ;

se  $v_{j+1} \notin I'$  então  $I'$  é um c.i.m. de  $G_j$ , isto é  $I' \in F_j$ ,

se  $v_{j+1} \in I'$  então  $I' - \{v_{j+1}\} \subseteq I$  para algum  $I \in F_j$  tal que  $I' = I - \text{Adj}(v_{j+1}) \cup \{v_{j+1}\}$ .

**Observação 3.2:**

Um c.i.m.  $I' \in F_{j+1}$  pode ser obtido a partir de dois c.i.m. diferentes  $I_1$  e  $I_2$  de  $F_j$ . Isto é:

$$I' - \{v_{j+1}\} = I_1 - \{v_{j+1}\} = I_2 - \{v_{j+1}\}.$$

Para o exemplo da figura 3.1,  $I' = \{1,5\} \in F_5$  é gerado por  $I_1 = \{1,3\}$  e  $I_2 = \{1,4\}$  com  $I_1, I_2 \in F_4$ .

Define-se o multiconjunto

$$F'_{j+1} = \{ I' / I' = I - \text{Adj}(v_{j+1}) \cup \{v_{j+1}\}, I \in F_j \}.$$

Dada a observação 3.2,  $F'_{j+1}$  é um multiconjunto porque pode conter conjuntos com duplicadas.

Tem-se

$$F_{j+1} \subseteq F_j \cup F'_{j+1}.$$

Para obter  $F_{j+1}$  a partir de  $F_j \cup F'_{j+1}$  é preciso eliminar os conjuntos que são duplicata de outros c.i.m. e aqueles conjuntos que não são maximais.

Dada a família  $F_j$  de c.i.m. do subgrafo  $G_j$  induzido por  $\{v_1, \dots, v_j\}$ , Paull e Unger constroem a família  $F_{j+1}$  segundo o seguinte procedimento 3.1:

**Procedimento 3.1: Procedimento de Paull e Unger**

Passo 1: construir  $F'_{j+1}$

para todo  $I \in F_j$ :  $F'_{j+1} = F'_{j+1} \cup \{ I - \text{Adj}(v_{j+1}) \cup \{v_{j+1}\} \}$

Passo 2: eliminar conjuntos não maximais e duplicadas.

O método de Paull e Unger pode ser implementado segundo o algoritmo 3.1 descrito a seguir:

**Algoritmo 3.1: Algoritmo de Paull e Unger**

Dado  $G(V,E)$  conexo,  $\text{Adj}(v_j) \forall v_j \in V$ ,  $|V| = n$ ,  $|E| = m$

1.  $F_1 = \{v_1\}$
2. para  $j = 1$  até  $n - 1$  efetuar
  - 2.1  $F'_{j+1} = \emptyset$
  - 2.2 para todo  $I \in F_j$  efetuar
    - $F'_{j+1} = F'_{j+1} \cup \{ I - \text{Adj}(v_{j+1}) \cup \{v_{j+1}\} \}$
  - 2.3  $F = F_j \cup F'_{j+1}$
  - 2.4 eliminar de  $F$  os conjuntos duplicata de outros c.i.m. e os conjuntos não maximais
  - 2.5  $F_{j+1} = F$
- 3  $F(G) = F_n$ .

**Complexidade:**

Passo 1:  $O(1)$

Passo 2: é executado  $O(n)$  vezes

2.1:  $O(1)$

2.2:  $O(n \cdot i(G))$  pois  $|\text{Adj}(v_{j+1})| \leq n$ ,  $|F_j| \leq |F_n| = i(G)$

2.3:  $O(1)$

2.4:  $O(n[i(G)]^2)$  pois  $|F_j| \leq |F_{j+1}| \leq |F_n| = i(G)$   
precisa comparar  $|F_j|$   $|F_{j+1}|$  conjuntos

2.5:  $O(1)$

total do passo 2:  $O(n^2[i(G)]^2)$

Passo 3:  $O(1)$

A complexidade total do algoritmo 3.1 é  $O(n^2[i(G)]^2)$ , isto é tem tempo total polinomial no tamanho do grafo e no número de c.i.m. do grafo. A complexidade de retardo do algoritmo não é polinomial no tamanho do grafo. O espaço requerido é  $O(m + n \cdot i(G))$ .

Para o grafo  $G$  da figura 3.1 tem-se:

j	$F_j$	$F'_{j+1}$	F
1	{1}	{2}	{1},{2}
2	{1},{2}	{1,3},{3}	{1},{2},{1,3},{3}
3	{2},{1,3}	{4},{1,4}	{2},{1,3},{4},{1,4}
4	{2},{1,3},{1,4}	{2,5},{1,5},{1,5}	{2},{1,3},{1,4},{2,5},{1,5},{1,5}
5	{1,3},{1,4},{2,5},{1,5}		

logo  $F(G) = \{ \{1,3\}, \{1,4\}, \{2,5\}, \{1,5\} \}$ .

Lawler, Lenstra e Rinnooy Kan (1980) apresentam uma generalização do algoritmo de Paull e Unger para a construção de  $F_{j+1}$  a partir de  $F_j$ , descartando os conjuntos independentes não maximais e os conjuntos duplicadas de outros c.i.m.. Eles apresentam o seguinte procedimento para este alvo:

**Procedimento 3.2: Procedimento de Lawler, Lenstra e Rinnooy Kan**

- Passo 1: para cada  $I \in F_j$  determinar todos os conjuntos independentes  $I'$  que são maximais em relação a  $I \cup \{v_{j+1}\}$
- Passo 2: para cada  $I'$ : se  $I'$  é maximal para  $G_{j+1}$  então  $I' \in F_{j+1}$
- Passo 3: para cada  $I'$  que seja maximal se ele já está em  $F_{j+1}$  então descartá-lo.

**3.2. Algoritmo de Tsukiyama, Ide, Ariyoshi e Shirawaka (1977)**

Dado o grafo  $G(V,E)$ , Tsukiyama et alii constroem a família  $F_{j+1}$  de c.i.m. do subgrafo  $G_j$  induzido por  $\{v_1, \dots, v_j\}$  eliminando conjuntos repetidos e conjuntos independentes não maximais, de forma mais eficiente que Lawler et alii.

No passo 3, em vez de comparar  $I'$  com todos os elementos já incorporados a  $F_{j+1}$ ,  $I'$  é introduzido em  $F_{j+1}$  se e só se ele é obtido do menor lexicográfico  $I$  de  $F_j$  entre todos os c.i.m. de  $F_j$  que podem gerar  $I'$ .

**Observação 3.3:**

1. Dado  $I \in F_j$ , o conjunto independente  $I' = I - \text{Adj}(v_{j+1}) \cup \{v_{j+1}\}$  de  $G_j$  é maximal se e somente se

$$\forall k < j + 1, v_k \notin I \text{ tem-se } \text{Adj}(v_k) \cap I' \neq \emptyset.$$

Caso contrário: para  $v_k$  tal que  $\text{Adj}(v_k) \cap I' = \emptyset$  tem-se  $I' \cup \{v_k\}$  é c.i.m. de  $G_{j+1}$  e portanto  $I'$  não é maximal.

Isto acontece no exemplo da figura 3.1, ao construir  $F_3$  tem-se

$$\begin{aligned} j &= 2 & \mathbf{I} &= \{2\} & \mathbf{I}' &= \{3\} \\ k &= 1 & 1 &\notin \mathbf{I}' & \text{Adj}(1) &= \{2\} & \mathbf{I}' &= \emptyset \\ & & \text{efetivamente} & \{1,3\} & \text{é c.i.m de } & \mathbf{G}_3. \end{aligned}$$

2.- Para evitar c.i.m. duplicados, dado  $\mathbf{I} \in F_j$ , o c.i.m.  $\mathbf{I}' = \mathbf{I} - \text{Adj}(\mathbf{v}_{j+1}) \cup \{\mathbf{v}_{j+1}\}$  é introduzido em  $F_{j+1}$  se e somente se  $\mathbf{I} \cap \text{Adj}(\mathbf{v}_{j+1})$  é lexicograficamente menor que todos os outros  $\mathbf{T} \cap \text{Adj}(\mathbf{v}_{j+1})$  com  $\mathbf{T} \in F_j$  tal que  $\mathbf{T} \cap \text{Adj}(\mathbf{v}_{j+1}) = \mathbf{I} \cap \text{Adj}(\mathbf{v}_{j+1})$ .

Isto é:  $\mathbf{I} \in F_j$  satisfaz a condição lexicográfica se e somente se

$$\begin{aligned} \nexists k < j+1, \mathbf{v}_k \notin \mathbf{I} \text{ tal que } \text{Adj}(\mathbf{v}_k) \cap [\mathbf{I} - \text{Adj}(\mathbf{v}_{j+1})] &= \emptyset \\ \text{e } \text{Adj}(\mathbf{v}_k) \cap \{\mathbf{v}_1, \dots, \mathbf{v}_{k-1}\} \cap [\mathbf{I} \cap \text{Adj}(\mathbf{v}_{j+1})] &= \emptyset \end{aligned}$$

ou equivalentemente:

$$\forall k < j+1, \mathbf{v}_k \notin \mathbf{I} \text{ tem-se } \text{Adj}(\mathbf{v}_k) \cap [\mathbf{I} - [\text{Adj}(\mathbf{v}_{j+1}) \cap \{\mathbf{v}_{k+1}, \dots, \mathbf{v}_j\}]] \neq \emptyset.$$

Isto acontece no exemplo da figura 3.1, ao construir  $F_5$  os c.i.m.  $\{1,3\}$  e  $\{1,4\}$  de  $F_4$  induzem o c.i.m.  $\{1,5\}$ :

$$\begin{aligned} \mathbf{I} &= \{1,3\} & \mathbf{I} - \text{Adj}(5) &= \{1\} & \mathbf{I} \cap \text{Adj}(5) &= \{3\} \\ \mathbf{T} &= \{1,4\} & \mathbf{T} - \text{Adj}(5) &= \{1\} & \mathbf{T} \cap \text{Adj}(5) &= \{4\} \end{aligned}$$

$\mathbf{I} \cap \text{Adj}(5)$  é lexicograficamente menor que  $\mathbf{T} \cap \text{Adj}(5)$ , logo  $\mathbf{I}' = \mathbf{I} - \text{Adj}(5) \cup \{5\} = \{1,5\}$  é incorporado a  $F_5$  enquanto  $\mathbf{T} - \text{Adj}(5) \cup \{5\}$  não o é, evitando assim as duplicatas.

Para implementar o algoritmo, constrói-se uma árvore binária enraizada em  $\{\mathbf{v}_1\}$ , de altura  $n$ , cujas arestas são construídas em cada passo ao fazer uma busca em profundidade na árvore. Cada vértice no nível  $j$  da árvore corresponde a um c.i.m.  $\mathbf{I}$  de  $F_j$ . O nó  $\mathbf{I} \in F_j$  pode ter no máximo dois filhos; se  $\mathbf{I} \cap \text{Adj}(\mathbf{v}_{j+1}) = \emptyset$ , isto é  $\mathbf{I}$  não contém vértices adjacentes a  $\mathbf{v}_{j+1}$ , então  $\mathbf{I}$  tem um único filho  $\mathbf{I}' = \mathbf{I} \cup \{\mathbf{v}_{j+1}\}$ . Caso contrário,  $\mathbf{I} \cap \text{Adj}(\mathbf{v}_{j+1})$  é não vazio, o filho direito é  $\mathbf{I}$  e tem filho esquerdo  $\mathbf{I}'$  com  $\mathbf{I}' = \mathbf{I} - \text{Adj}(\mathbf{v}_{j+1}) \cup \{\mathbf{v}_{j+1}\}$  se  $\mathbf{I}'$  é maximal e satisfaz a condição lexicográfica.

Este método é descrito no algoritmo 3.2 utilizando uma pilha  $Q$  cujo último elemento corresponde ao nó corrente que está sendo examinado.

**Algoritmo 3.2: Algoritmo de Tsukiyama et alii**

Dado  $G(V,E)$  conexo,  $\text{Adj}(v_j) \forall v_j \in V$ ,  $|V| = n$ ,  $|E| = m$

1.  $I = \{v_1\}$   
 $j = 1$   
 $Q = I$
2. enquanto  $Q \neq \emptyset$  e existam elementos não marcados em  $Q$  efetuar
  - 2.1 seja  $I$  o último elemento não marcado de  $Q$   
 marcar  $I$   
 $j = \text{NIVEL}(I)$
  - 2.2 se  $I \cap \text{Adj}(v_{j+1}) \neq \emptyset$  então efetuar  
 $\text{FILHODIR}(I) = I$   
 $Q = Q \cup \text{FILHODIR}(I)$   
 $\text{NIVEL}(I) = j + 1$
  - 2.3  $I' = I - \text{Adj}(v_{j+1}) \cup \{v_{j+1}\}$   
 $k = 1$
  - 2.4 enquanto  $k < j + 1$  efetuar  
 se  $v_k \notin I$  então  
 se  $I' \cap \text{Adj}(v_k) = \emptyset$  então  $k = j + 100$  ( $I'$  não maximal)  
 $k = k + 1$
  - 2.5 se  $k = j + 1$  então efetuar  
 $k = 1$   
 enquanto  $k < j + 1$  efetuar  
 se  $v_k \notin I$  então  
 se  $\text{Adj}(v_k) \cap [I - [\text{Adj}(v_{j+1}) \cap \{v_{k+1}, \dots, v_j\}]] = \emptyset$   
 então  $k = j + 100$  ( $I'$  não menor lexicográfico)  
 $k = k + 1$
  - 2.6 se  $k = j + 1$  então efetuar  
 $\text{FILHOESQ}(I) = I'$   
 $Q = Q \cup \text{FILHOESQ}(I)$   
 $\text{NIVEL}(I) = j + 1$
  - 2.7 se  $j = n - 1$  então efetuar  
 imprimir  $\text{FILHOESQ}(I)$  se existir  
 imprimir  $\text{FILHODIR}(I)$  se existir  
 retirar  $I$ ,  $\text{FILHOESQ}(I)$  e  $\text{FILHODIR}(I)$  de  $Q$ .

**Complexidade:**

Passo 1:  $O(1)$

Passo 2: é executado  $O(i(G))$  vezes

2.1:  $O(1)$

2.2:  $O(n)$

2.3:  $O(1)$

2.4:  $O((m + n) n)$

2.5:  $O((m + n) n)$

2.6:  $O(1)$

2.7:  $O(1)$

total do passo 2:  $O(nm \ i(G))$ .

A complexidade total do algoritmo 3.2 é  $O(nm \ i(G))$ , possui tempo polinomial no tamanho do grafo e no número de c.i.m. do grafo. No entanto, o tempo existente entre a geração de um c.i.m. e o próximo corresponde ao tempo de passar de uma folha da árvore à seguinte, da esquerda para a direita, este tempo é  $O(nm)$ , isto é tem retardo polinomial entre dois c.i.m.. O espaço requerido é  $O(m + n)$ .

A execução do algoritmo 3.2 no grafo  $G$  da figura 3.1 fornece os seguintes resultados, o subíndice no conjunto  $I$  em  $Q$  indica o nível de  $I$  na árvore binária:

j	Q	I	FILHOESQ(I)	FILHODIR(I)	c.i.m.
1	$\{1\}_1^*, \{1\}_2, \{2\}_2$	$\{1\}$	$\{2\}$	$\{1\}$	
2	$\{1\}_1^*, \{1\}_2, \{2\}_2^*, \{2\}_3$	$\{2\}$		$\{2\}$	
3	$\{1\}_1^*, \{1\}_2, \{2\}_2^*, \{2\}_3^*, \{2\}_4$	$\{2\}$		$\{2\}$	
4	$\{1\}_1^*, \{1\}_2, \{2\}_2^*, \{2\}_3^*, \{2\}_4^*, \{2,5\}_5$	$\{2\}$	$\{2,5\}$		$\{2,5\}$
2	$\{1\}_1^*, \{1\}_2, \{2\}_2^*, \{2\}_3^*, \{2\}_4^*, \{2,5\}_5^*, \{1,3\}_3$	$\{1\}$	$\{1,3\}$		
3	$\{1\}_1^*, \{1\}_2, \{2\}_2^*, \{2\}_3^*, \{1,3\}_3^*, \{1,3\}_4, \{1,4\}_4$	$\{1,3\}$	$\{1,4\}$	$\{1,3\}$	
4	$\{1\}_1^*, \{1\}_2, \{2\}_2^*, \{2\}_3^*, \{1,3\}_3^*, \{1,3\}_4, \{1,4\}_4^*, \{1,4\}_5$	$\{1,4\}$		$\{1,4\}$	$\{1,4\}$
4	$\{1\}_1^*, \{1\}_2, \{2\}_2^*, \{2\}_3^*, \{1,3\}_3^*, \{1,3\}_4^*, \{1,3\}_5, \{1,5\}_5$	$\{1,3\}$	$\{1,5\}$	$\{1,3\}$	$\{1,5\}$ $\{1,3\}$

A árvore construída durante o processo para o grafo da figura 3.1 é exibida na figura 3.2.

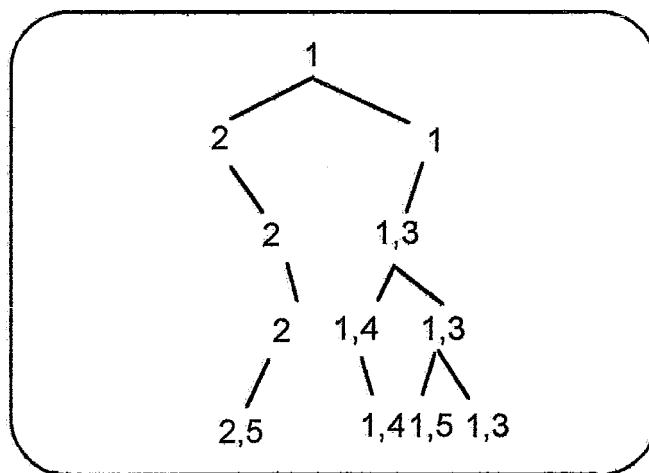


Fig. 3.2: Árvore associada ao grafo  $G$  (algoritmo de Tsukiyama et alii)

logo  $F(G) = \{ \{2,5\}, \{1,4\}, \{1,5\}, \{1,3\} \}$ .



### 3.3. Algoritmo de Loukakis (1983)

Loukakis faz backtracking para construir a família de c.i.m. de um grafo dado  $G(V,E)$  com conjunto de vértices  $V = \{v_1, v_2, \dots, v_n\}$ . O procedimento representa-se em uma árvore  $T_G$ .

Para isto ele define em cada passo da busca os conjuntos a seguir, os quais caracterizam o nó  $x$  corrente:

$W_x^+$  é um conjunto independente

$W_x^- = \{ \text{vértices já considerados na busca e que não podem pertencer a nenhum c.i.m. que contenha } W_x^+ \}$

$W_x' = \{ \text{vértices candidatos a serem incorporados a } W_x^+ \}$ .

Denotando  $\text{Adj}(W_x^+) = \{ v \in V / \exists w \in W_x^+ \text{ com } (v,w) \in E \}$  tem-se:

$$W_x' = V - [W_x^+ \cup W_x^- \cup \text{Adj}(W_x^+)]$$

$$V = W_x^+ \cup \text{Adj}(W_x^+) \cup W_x^- \cup W_x'.$$

O processo começa no vértice  $u$  de menor grau.

Ao ramificar a partir de um nó  $x$  da árvore de busca com conjuntos  $W_x^+$ ,  $W_x^-$  e  $W_x'$  já determinados, escolhe-se um vértice  $u \in W_x'$  a ser unido a  $W_x^+$ , seja  $y$  o filho de  $x$  na árvore  $T_G$  então:

$$W_y^+ = W_x^+ \cup \{u\}$$

$$W_y' = W_x' - N[u]$$

se  $W_x^- \subseteq \text{Adj}(W_x^+)$  e  $\exists u \in W_x'$  tal que  $\text{Adj}(u) \cap W_x' = \emptyset$  então todo c.i.m. que contenha  $W_x^+$  deve conter  $u$ .

Quando não é possível acrescentar mais vértices a  $W_x^+$ , isto é  $W_x' = \emptyset$ , então  $W_x^+$  pode ser maximal, faz-se backtracking retirando o último vértice  $v$  incorporado a  $W_x^+$  e volta a  $z$  o pai( $x$ ), atualiza-se

$$W_z^- = W_x^- \cup \{v\}$$

$$W_z' = W_x' \cup [\text{Adj}(v) \cap W_x'].$$

**Observação 3.4:**

Ao fazer backtracking ( $W' = \emptyset$ ) observa-se:

- se  $W_x^- \subseteq Adj(W_x^+)$  então  $V = W_x^+ \cup Adj(W_x^+)$  e portanto  $W_x^+$  é c.i.m. de G
- se  $W_x^- \not\subseteq Adj(W_x^+)$  então  $\exists u \in W_x^- - Adj(W_x^+)$  com  $W_x^+ \cap Adj(u) = \emptyset$  logo  $W_x^+ \cup \{u\}$  é um conjunto independente e portanto  $W_x^+$  não é maximal.

Um vértice  $u \in W_x^-$  é denominado fechado se  $u \in Adj(W_x^+)$ , caso contrário  $u$  é denominado aberto.

Com isto, tem-se:

**Propriedade 3.1:**

$W_x^+$  é c.i.m. se e somente se  $W_x^- = \emptyset$  e  $\forall u \in W_x^-$  é fechado.

Para implementar o algoritmo, Loukakis utiliza um conjunto M tal que M(j) é positivo se  $v_j \in W_x^+$ , M(j) é negativo se  $-v_j \in W_x^-$ ; o conjunto Z que corresponde ao conjunto de vértices abertos e o parâmetro

$$t = \begin{cases} 0 & \text{se } W_x^- \subseteq Adj(W_x^+) \\ 1 & \text{se não} \end{cases}$$

O algoritmo 3.3 implementa o método de Loukakis, constrói a árvore de busca a partir de um nó raiz artificial R com  $W'_R = V$ ,  $W^+_R = W^-_R = \emptyset$ .

**Algoritmo 3.3: Algoritmo de Loukakis**

Dado  $G(V,E)$  conexo,  $Adj(v) \forall v \in V$ ,  $|V| = n$ ,  $|E| = m$

1.  $M = \emptyset$   
 $u = R$   
 $W'_R = V$   
 $Z = \emptyset$   
 $t = 0$   
 $k = R$
2. determinar o vértice u de menor grau
3.  $x = u$   
 $pai(u) = R$   
 $M = M \cup \{u\}$   
 $W'_u = W'_{pai(u)} - N[u]$

4. enquanto  $[\{x\} \cup \text{Adj}(x)] \not\subseteq Z$  efetuar
- 4.1 enquanto  $W'_u \neq \emptyset$  efetuar
- 4.1.1 se  $t = 1$  então efetuar
- $$Z = Z - [\{k\} \cup \text{Adj}(u)]$$
- se  $Z = \emptyset$  então  $t = 0$
- caso contrário efetuar
- $$k = Z(1)$$
- se  $\text{Adj}(k) \cap W'_u \neq \emptyset$  então efetuar
- determinar  $a \in \text{Adj}(k) \cap W'_u$
- $$\text{pai}(a) = u$$
- $$u = a$$
- $$t = 1$$
- 4.1.2 se  $W'_u \neq \emptyset$  então efetuar
- $$a = W'_u(1)$$
- $$\text{pai}(a) = u$$
- $$u = a$$
- se  $\text{Adj}(u) \cap W'_{\text{pai}(u)} = \emptyset$  então marcar  $u$
- $$M = M \cup \{u\}$$
- $$W'_u = W'_{\text{pai}(u)} - N[u]$$
- 4.2 imprimir  $M - \{j / -j \in M\}$
- 4.3 seja  $v$  o elemento de  $M$  com sinal  $+$  e não marcado que está mais à direita
- $$Z = \{v\} \cup \{j / -j \in M, j \in \text{Adj}(v)\}$$
- $$R(v) = \{j / j \in M \text{ ou } -j \in M \text{ e } j \text{ ou } -j \text{ estão à direita de } v\}$$
- $$M = M - R(v)$$
- trocar  $v$  por  $-v$  em  $M$
- $$W'_{\text{pai}(v)} = W'_v \cup (R(v) \cup [\text{Adj}(v) \cap W'_{\text{pai}(v)}])$$
- $$x = M(1)$$
- $$k = Z(1)$$
- 4.4 se  $\text{Adj}(k) \cap W'_{\text{pai}(v)} = \emptyset$  então ir a 4.3
- 4.5 caso contrário efetuar
- determinar  $u \in \text{Adj}(k) \cap W'_{\text{pai}(v)}$
- $$\text{pai}(u) = \text{pai}(v)$$
- $$t = 1$$
- se  $\text{Adj}(u) \cap W'_{\text{pai}(u)} = \emptyset$  então marcar  $u$
- $$M = M \cup \{u\}$$
- $$W'_u = W'_{\text{pai}(u)} - N[u].$$

**Complexidade:**

Passo 1:  $O(1)$

Passo 2:  $O(nm)$

Passo 3:  $O(n)$

Passo 4: é executado  $O(i(G))$  vezes

4.1: é executado  $O(n)$  vezes

4.1.1:  $O(n + m)$

4.1.2:  $O(m)$

4.2:  $O(1)$

4.3:  $O(n + m)$

4.4:  $O(n)$

4.5:  $O(n + m)$

total do passo 4:  $O([nm + n + m] i(G)) = O(nm i(G))$ .

A complexidade total do algoritmo 3.3 é  $O(nm i(G))$ , possui tempo polinomial no tamanho do grafo e no número de c.i.m. do grafo. No entanto, o tempo existente entre a geração de um c.i.m. e o próximo corresponde ao tempo de passar de uma folha da árvore à seguinte, da esquerda para a direita. Este tempo é  $O(nm)$ , isto é tem retardo polinomial entre dois c.i.m.. O espaço requerido é  $O(m + n)$ .

Por exemplo para o grafo  $G$  da figura 3.1 tem-se os resultados a seguir:

u	M	pai(u)	$W'_u$	t	c.i.m.
R	{ }		{1,2,3,4,5}	0	
1	{1}	R	{3,4,5}		
3	{1,3}	1	{ }		{1,3}
4	{1,-3}	1	{4,5}	1	
1			{4,5}		
4	{1,-3,4}		{ }	0	{1,4}
1	{1,-3,-4}		{5}	1	
5	{1,-3,-4,5*}		{ }	0	{1,5}
1	{-1}		{2,3,4,5}	1	
2	{-1,2}	R	{5}	0	
5	{-1,2,5*}	2			{2,5}
R					

A árvore construída durante o processo para o grafo da figura 3.1 é exibida na figura 3.3.

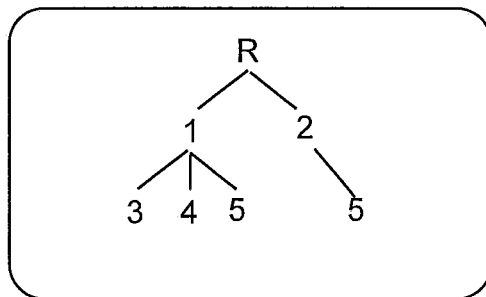


Fig. 3.3: Árvore associada ao grafo  $G$  (algoritmo de Loukakis)

logo  $F(G) = \{ \{1,3\}, \{1,4\}, \{1,5\}, \{2,5\} \}$ .

Loukakis apresenta como exemplo o grafo  $G_L$  da figura 3.4, com os resultados a seguir

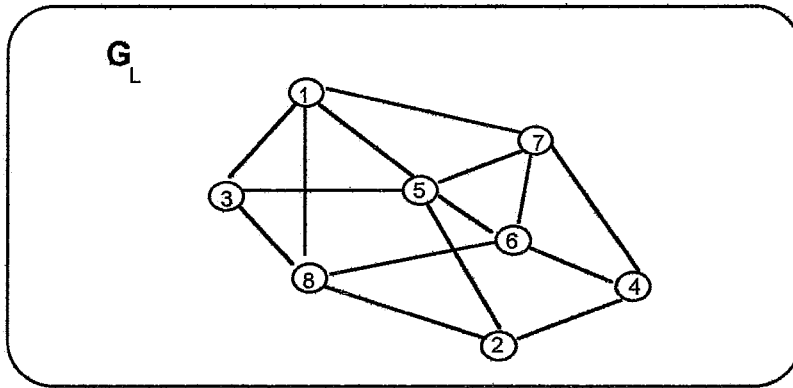


Fig. 3.4: Exemplo de Loukakis

u	M	pai(u)	$W'_u$	t	c.i.m.
R	{}		{1,2,3,4,5,6,7,8}	0	
2	{2}	R	{1,3,6,7}		
1	{2,1}	2	{6}		
6	{2,1,6*}	1	{}		{2,1,6}
2	{2,-1}		{3,6,7}		
3	{2,-1,3*}	2	{6,7}	1	
6	{2,-1,3*,6}	3	{}	0	{2,3,6}
3	{2,-1,3*,-6}		{7}		
7	{2,-1,3*,-6,7*}	3	{}		{2,3,7}
R	{-2}		{3,6,7,1,4,5,8}		
4	{-2,4}	R	{3,1,5,8}		
3	{-2,4,3}	4	{}		{4,3}
	{-2,4,-3}				
4			{1,5,8}		
1	{-2,4,-3,1}	4	{}	1	{4,1}
4	{-2,4,-3,-1}		{5,8}		
5	{-2,4,-3,-1,5*}	4	{8}	0	
8	{-2,4,-3,-1,5*,8*}	5	{}		{4,5,8}
R					

A árvore construída durante o processo para o grafo  $G_L$  da figura 3.4 é exibida na figura 3.5.

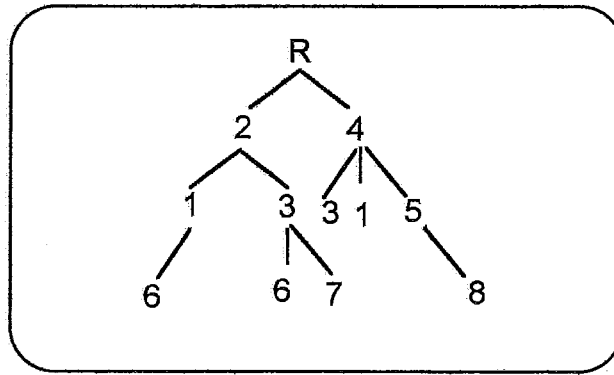


Fig. 3.5: Árvore associada ao grafo  $G_L$  (algoritmo de Loukakis)

logo  $F(G) = \{ \{2,1,6\}, \{2,3,6\}, \{2,3,7\}, \{4,3\}, \{4,1\}, \{4,5,8\} \}$ .

Loukakis e Tsouros (1981) apresentam um algoritmo de busca em profundidade para, dado um grafo gerar a família de c.i.m em ordem lexicográfica. O procedimento é similar ao de Loukakis (1983) utilizando os conjuntos  $W_x^+$ ,  $W_x^-$  e  $W_x'$  correspondentes a um conjunto independente, aos vértices já considerados na busca e que não podem pertencer a nenhum c.i.m. que contenha  $W_x^+$  e aos vértices candidatos a serem incorporados a  $W_x^+$  de modo a obter um c.i.m. que contenha todo vértice de  $W_x^+$  mas nenhum vértice de  $W_x^-$ .

O esquema lexicográfico é obtido com as seguintes regras:

- ramificar um nó  $x$  que seja o de menor índice em  $W_x'$
- fazer backtracking no último vértice introduzido em  $W_x^+$ .

No esquema lexicográfico os vértices são examinados numa ordem fixa, isto produz um aumento na complexidade do algoritmo, mas a complexidade total fica da mesma ordem, isto é  $O(nm i(G))$ .

### 3.4. Algoritmo de Johnson, Yannakakis e Papadimitriou (1988)

Dado o grafo  $G(V,E)$ , eles desenvolvem um algoritmo para gerar todos os conjuntos independentes maximais do grafo  $G$ , em ordem lexicográfica com retardo polinomial entre dois conjuntos consecutivos.

Eles, também, provam que não existe um algoritmo com retardo polinomial entre dois conjuntos para gerar a família de c.i.m. de um grafo dado na ordem lexicográfica reversa, a menos que  $P = NP$ .

Para gerar a família  $F(G)$  de c.i.m. de  $G$  em ordem lexicográfica, Johnson et alii primeiro determinam o c.i.m. lexicograficamente menor, depois constroem o próximo c.i.m. dados o anterior e o c.i.m. no subgrafo  $G_j$  induzido por  $\{v_1, \dots, v_j\}$ , e assim por diante.

Eles utilizam uma pilha  $Q$  para armazenar os c.i.m. construídos no processo, mas se um c.i.m. já está em  $Q$  então ele é descartado.

O algoritmo 3.4 implementa o algoritmo de Johnson et alii.

**Algoritmo 3.4: Algoritmo de Johnson et alii**

Dado  $G(V,E)$  conexo,  $\text{Adj}(v_j) \forall v_j \in V$ ,  $|V| = n$ ,  $|E| = m$

1. Determinar  $I^*$  o primeiro c.i.m. de  $G$  lexicograficamente menor
2. inserir  $I^*$  em  $Q$
3. enquanto  $Q \neq \emptyset$  efetuar
  - 3.1 seja  $I$  o mínimo lexicográfico não marcado de  $Q$
  - 3.2 marcar  $I$   
imprimir  $I$
  - 3.3 para todo  $v_j \in V$  efetuar
    - se existe  $v_i \in I$  com  $i < j$ ,  $(v_i, v_j) \in E$  então efetuar
      - $I_j = I \cap \{v_1, \dots, v_j\}$
      - se  $I_j - \text{Adj}(v_j) \cup \{v_j\}$  é c.i.m. de  $G_j$  então efetuar
        - determinar  $T$  o primeiro c.i.m. de  $G$  lexicograficamente menor que contém  $I_j - \text{Adj}(v_j) \cup \{v_j\}$
        - incluir  $T$  em  $Q$  se ele não está.

**Complexidade:**

Passo 1:  $O(n + m)$

considerar os vértices na ordem  $v_1, \dots, v_n$ ; se  $v_j$  não é adjacente a nenhum vértice já introduzido em  $I^*$  então unir  $v_j$  a  $I^*$

Passo 2:  $O(1)$

Passo 3: é executado  $i(G)$  vezes

3.1:  $O(n \log[i(G)])$

3.2:  $O(1)$

3.3:  $O(n[n + m + n]) \Rightarrow O(nm)$

total do passo 3:  $O(nm i(G))$ .

A complexidade total do algoritmo 3.4 é  $O(nm i(G))$ , com retardo polinomial  $O(nm)$  entre dois c.i.m. gerados em ordem lexicográfica, listando cada c.i.m. quando ele é gerado. Então necessita espaço  $O(n)$ .

Por exemplo para o grafo  $G$  da figura 3.1, o algoritmo 3.4 produz os resultados a seguir:  $I^* = \{1,3\}$

T	Q	I	c.i.m.
	$\{1,3\}^*$	$\{1,3\}$	$\{1,3\}$
$\{2,5\}$	$\{1,3\}^*, \{2,5\}$	$\{2,5\}$	$\{2,5\}$
$\{1,4\}$	$\{1,3\}^*, \{2,5\}, \{1,4\}$		
$\{1,5\}$	$\{1,3\}^*, \{2,5\}, \{1,4\}, \{1,5\}$		
	$\{1,3\}^*, \{2,5\}, \{1,4\}^*, \{1,5\}$	$\{1,4\}$	$\{1,4\}$
	$\{1,3\}^*, \{2,5\}, \{1,4\}^*, \{1,5\}^*$	$\{1,5\}$	$\{1,5\}$
	$\{1,3\}^*, \{2,5\}^*, \{1,4\}^*, \{1,5\}^*$	$\{2,5\}$	$\{2,5\}$

O algoritmo 3.4 constrói a família  $F(G) = \{ \{1,3\}, \{1,4\}, \{1,5\}, \{2,5\} \}$  nessa ordem.

Na tabela 3.1 apresenta-se um resumo dos algoritmos desenvolvidos para construir a família de conjuntos independentes maximais de um grafo dado e suas complexidades.

**Tabela 3.1: Progresso nos algoritmos para gerar  $F(G)$**

AUTOR	ANO	COMPLEXIDADE TOTAL	ESPAÇO
Paull e Unger	1959	$O(n^2 [i(G)]^2)$	$O(m + n i(G))$
Tsukiyama et alii	1977	$O(nm i(G))$	$O(n + m)$
Loukakis e Tsouros	1981	$O(nm i(G))$	$O(n + m)$
Loukakis	1983	$O(nm i(G))$	$O(n + m)$
Johnson et alii	1988	$O(nm i(G))$	$O(n)$

### 3.5. Algoritmos para grafos particulares

Gupta, Lee e Leung (1982) desenvolvem um algoritmo para determinar um conjunto independente máximo em tempo  $O(n \log(n))$ , para um grafo de intervalo dado através da representação por intervalos. Leung (1984) faz backtracking no algoritmo de Gupta, Lee e Leung e gera todos os conjuntos independentes maximais de um grafo de intervalo em tempo  $O(n + b)$ , onde  $b$  é a soma do número de vértices de todos os conjuntos independentes maximais. Leung apresenta, também, um algoritmo de complexidade  $O(n+b)$  para o caso de um grafo arco-circular dado, com  $b$  igual à soma do número total de vértices de todos os conjuntos independentes maximais de  $G$ , utilizando o algoritmo para grafos de intervalo. Ele desenvolve um algoritmo de tempo  $O((n+e)i(G))$  para gerar a família de conjuntos independentes maximais de um grafo cordal dado, onde  $e$  é o número de arestas do grafo.

Kashiwabara et alii (1992) desenvolvem um algoritmo de tempo  $O(n^{2,5} + b)$  para gerar todos os conjuntos independentes máximos de um grafo bipartido, com  $b$  igual à soma do número total de vértices de todos os conjuntos independentes máximos de  $G$ .



## Capítulo 4

### SOLUÇÃO POR TRANSFORMAÇÃO DO PROBLEMA

Neste capítulo apresentam-se os algoritmos desenvolvidos para construir a família de conjuntos independentes maximais no caso particular de caminhos induzidos, ciclos induzidos, grafos grade completas de duas linhas, grafos de intervalo e grafos triangularizados. Todos estes algoritmos baseiam-se na idéia de transformar o problema de construir a família de c.i.m do grafo dado no problema de determinar todos os caminhos maximais fonte-sumidouro num dígrafo que depende da classe particular considerada.

#### 4.1. Caminhos sem cordas

Um caminho sem cordas ou caminho induzido  $P_n$  é uma seqüência de vértices  $v_1, \dots, v_n$  tal que  $(v_i, v_{i+1}) \in E$  para  $i = 1, \dots, n-1$ , e não existe nenhuma outra aresta em  $P_n$ .

O caminho  $P_n$  é desenhado de forma horizontal, os seus vértices são denotados por  $1, 2, \dots, n$  da esquerda à direita, como é mostrado na figura 4.1.

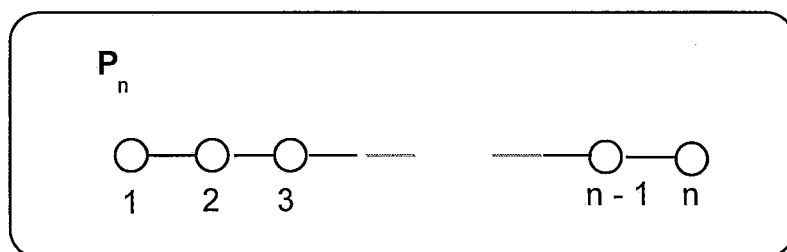


Fig. 4.1: Caminho  $P_n$

Um conjunto independente maximal de  $P_n$  é descrito pela seqüência de vértices que o compõem. Por exemplo para  $P_5$ ,  $I=1,3,5$  representa o c.i.m.  $I=\{1,3,5\}$ .

Seja:  $P_n$  um caminho de  $n$  vértices sem cordas,

$F(P_n)$  a família de conjuntos independentes maximais de  $P_n$ ,

$i(P_n) = \text{card}(F(P_n)) =$  número de c.i.m. de  $P_n$ .

Considere por exemplo  $P_7$ , a família de c.i.m. de  $P_7$  é

$F(P_7) = \{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6\}, \{2,4,7\}, \{2,5,7\}\}$ , o número de c.i.m. de  $P_7$  é  $i(P_7) = 7$ .

Na tabela 4.1 exibe-se a família  $F(P_n)$  e o número  $i(P_n)$  de c.i.m. dos caminhos  $P_1$  a  $P_9$ .

**Tabela 4.1:** C.i.m. de  $P_1, P_2, \dots, P_9$

$n$	$F(P_n)$	$i(P_n)$
1	{1}	1
2	{1}; {2}	2
3	{1,3}; {2}	2
4	{1,3}; {1,4}; {2,4}	3
5	{1,3,5}; {1,4}; {2,4}; {2,5}	4
6	{1,3,5}; {1,3,6}; {1,4,6}; {2,4,6}; {2,5}	5
7	{1,3,5,7}; {1,3,6}; {1,4,6}; {1,4,7}; {2,4,6}; {2,4,7}; {2,5,7}	7
8	{1,3,5,7}; {1,3,5,8}; {1,3,6,8}; {1,4,6,8}; {1,4,7}; {2,4,6,8}; {2,4,7}; {2,5,7}; {2,5,8}	9
9	{1,3,5,7,9}; {1,3,5,8}; {1,3,6,8}; {1,3,6,9}; {1,4,6,8}; {1,4,6,9}; {1,4,7,9}; {2,4,6,8}; {2,4,6,9}; {2,4,7,9}; {2,5,7,9}; {2,5,8}	12

**Observação 4.1:**

Considerando cada c.i.m  $I$  como uma seqüência de vértices  $I = v_1, v_2, \dots, v_k$  observa-se:

- o primeiro vértice da seqüência é 1 ou 2
- o último vértice da seqüência é  $(n-1)$  ou  $n$
- o vértice seguinte ao vértice  $i$ , se existir, é o vértice  $(i+2)$  ou o  $(i+3)$ .

### Algoritmo para gerar a família $F(P_n)$ de c.i.m de $P_n$

**Lema 4.1:**

A família  $F(P_n)$  de c.i.m. de  $P_n$  corresponde ao conjunto de todos os caminhos das raízes até as folhas das duas árvores binárias  $T_1$  e  $T_2$ , mostradas na figura 4.2, onde  $T_j$  é enraizada em  $j$ ; cada vértice  $i$  tem dois filhos  $i+2$  e  $i+3$ , para  $1 \leq i \leq n-3$ , o vértice  $n-2$  tem um só filho rotulado  $n$ ; os vértices rotulados  $n-1$  ou  $n$  são as folhas das árvores, para  $j=1,2$ .

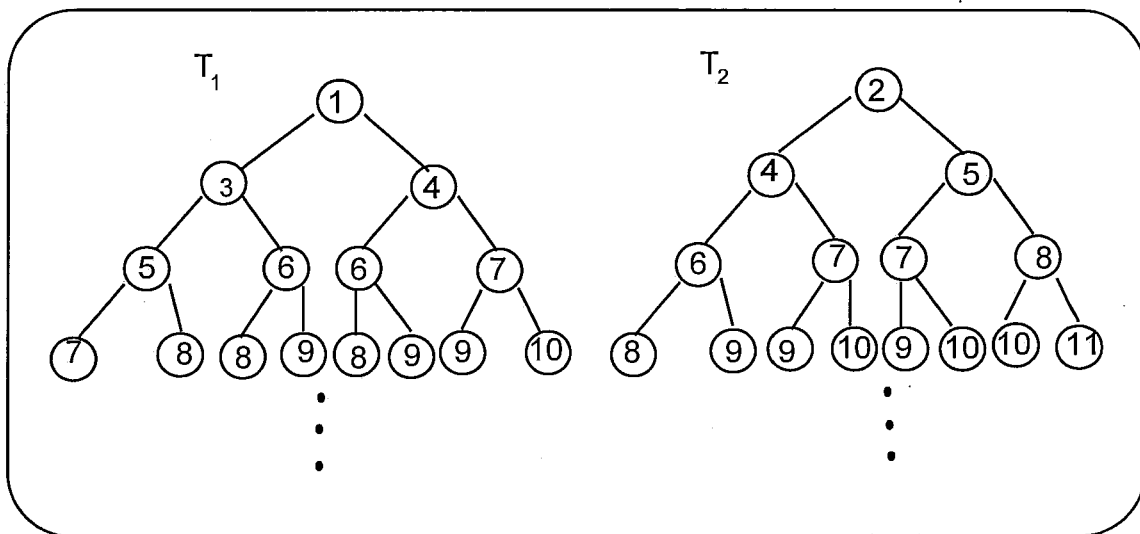


Fig. 4.2: Árvores  $T_1$  e  $T_2$

**Prova:**

Dadas as árvores binárias  $T_1$  e  $T_2$  construídas segundo o lema 4.1, tem-se que cada caminho  $I = v_1, v_2, \dots, v_k$  em  $T_j$  desde a raiz  $j = v_1$  até uma folha  $v_k$  corresponde a um c.i.m de  $P_n$  pois, em  $T_j$  existe a aresta  $(v_p, v_q)$  se e somente se  $(v_p, v_q) \notin E(P_n)$ , isto é sse  $v_q \neq v_p + 1$ , logo  $I$  é conjunto independente e  $I$  é maximal pois não é possível acrescentar mais vértices a  $I$ , dado que: a raiz  $v_1$  é 1 ou 2,  $(1,2) \in E(P_n)$ , se  $v_p \in I$  então  $v_{p+2} \in I$  ou  $v_{p+3} \in I$ ,  $(v_p, v_{p+1}) \in E(P_n)$ , a folha é  $n-1$  ou  $n$  e  $(n-1, n) \in E(P_n)$ . Os caminhos tem que ser raiz-folha, caso contrário o c.i.m. não é maximal.

Dado um c.i.m.  $I = \{v_1, v_2, \dots, v_k\}$  de  $P_n$  com  $v_1 < v_2 < \dots < v_k$ , a seqüência  $I$  dada por  $v_1, v_2, \dots, v_k$  corresponde a um caminho raiz-folha em  $T_j$ , pois como  $I$  é maximal então  $v_1$  é 1 ou 2,  $v_k$  é  $n-1$  ou  $n$ , se  $v_p \in I$  então  $v_{p+2} \in I$  ou  $v_{p+3} \in I$ , não existe  $(v_p, v_{p+1}) \in E(D)$  pois  $(i, i+1) \in E(P_n)$ , não existe  $(v_p, v_q) \in E(D)$  com  $v_q > v_p + 3$  pois não seria maximal.



Por exemplo, na figura 4.3 apresenta-se o caminho  $P_7$ , as árvores  $T_1$  e  $T_2$  associadas a  $P_7$ . Os caminhos da raiz até as folhas na árvore  $T_1$  são  $\{1,3,5,7\}$ ,  $\{1,3,6\}$ ,  $\{1,4,6\}$ ,  $\{1,4,7\}$ ; na árvore  $T_2$  tem-se os caminhos  $\{2,4,6\}$ ,  $\{2,4,7\}$  e  $\{2,5,7\}$  desde a raiz até as folhas.

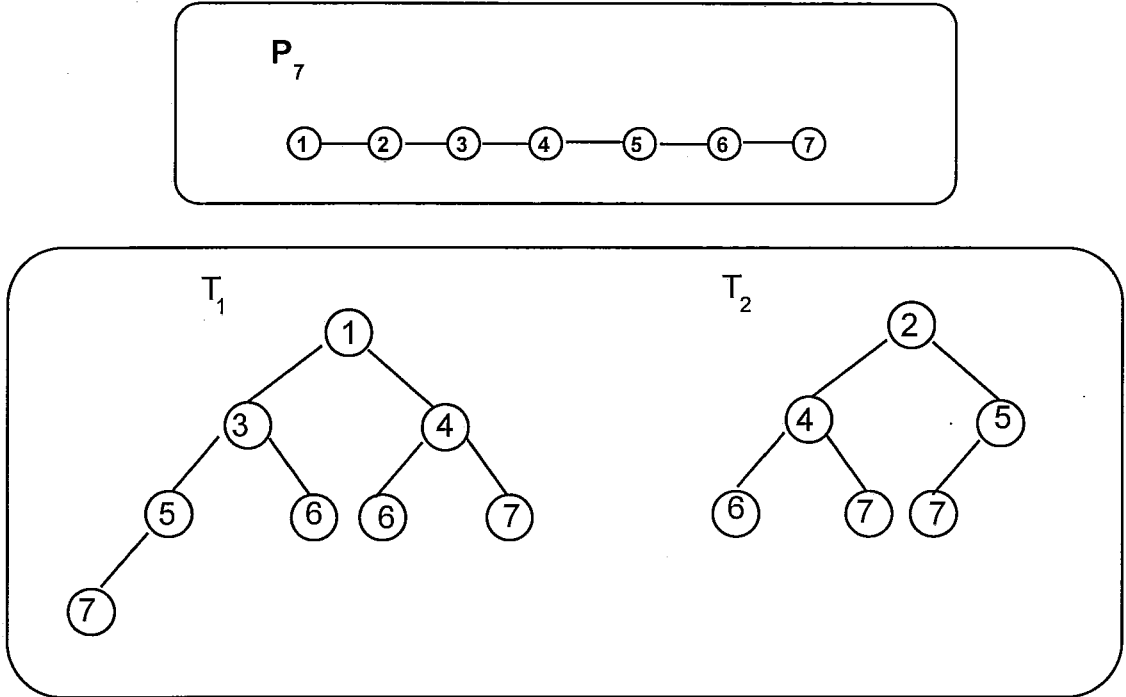


Fig. 4.3: c.i.m. de  $P_7$

Logo  $F(P_7) = \{ \{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6\}, \{2,4,7\}, \{2,5,7\} \}$ ,  $i(P_7) = 7$ .

**Observação 4.2:**

O lema 4.1 descreve um algoritmo para gerar  $F(P_n)$ , construindo as árvores binárias  $T_1$  e  $T_2$ , o qual tem complexidade de espaço exponencial. Para evitar este problema, transformam-se as duas árvores em um dígrafo acíclico  $D$  com conjunto de vértices  $V(D)$  igual ao de  $P_n$ ,  $V(D) = \{1, 2, \dots, n\}$ ; para  $1 \leq v < w \leq n$ , existe a aresta  $(v,w)$  em  $E(D)$  se e somente se  $w = v + 2$  ou  $w = v + 3$ . Dessa forma, as fontes de  $D$  são 1 e 2, os sumidouros são  $n - 1$  e  $n$ . Esta transformação é expressa no lema 4.2, logo após o exemplo 4.4, a prova desse lema é omitida por ser similar à do lema 4.1.

Por exemplo, na figura 4.4 apresenta-se o dígrafo  $D$  associado a  $P_7$ . Os caminhos fonte-sumidouro em  $D$  são  $\{1,3,5,7\}$ ,  $\{1,3,6\}$ ,  $\{1,4,6\}$ ,  $\{1,4,7\}$ ,  $\{2,4,6\}$ ,  $\{2,4,7\}$  e  $\{2,5,7\}$ .

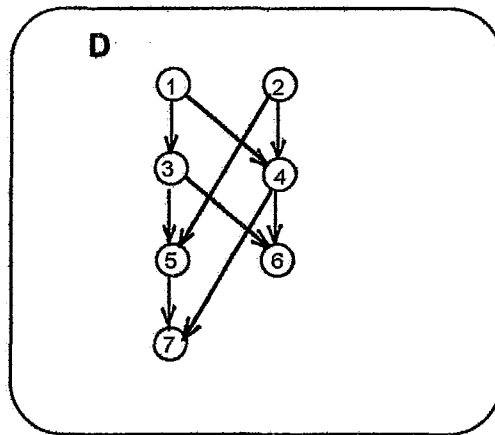


Fig. 4.4: Dígrafo associado a  $P_7$

**Lema 4.2:**

A família  $F(P_n)$  de c.i.m. de  $P_n$  corresponde ao conjunto de caminhos maximais fonte-sumidouro do dígrafo acíclico  $D$  com conjunto de vértices  $V(D) = \{1, 2, \dots, n\}$ ; existe a aresta  $(v,w)$  em  $E(D)$  se e somente se  $w = v + 2$  ou  $w = v + 3$ . As fontes de  $D$  são 1 e 2, os sumidouros são  $n - 1$  e  $n$ .

**Algoritmo:**

Dado  $n$ , desenvolve-se um algoritmo para construir a família  $F$  de c.i.m de  $P_n$ , utilizando a caracterização pelo dígrafo  $D$  descrito no lema 4.2. O algoritmo 4.1 implementa este procedimento.

**Algoritmo 4.1: Família de c.i.m. de um caminho induzido**

Dado  $n$

1. Construir o dígrafo  $D$  com  $V(D) = V(G)$  e  $(v,w) \in E(D) \Leftrightarrow w = v + 2$  ou  $w = v + 3$
2. Gerar todos os caminhos fonte-sumidouro de  $D$ .

**Complexidade:**

Passo 1:  $O(n)$

Passo 2:  $O(n)$  por caminho, ao fazer busca em profundidade irrestrita (ver observação 4.3).

A complexidade total do algoritmo 4.1 é  $O(n \cdot i(P_n))$ , possui tempo polinomial no tamanho do grafo e no número de c.i.m. do grafo. No entanto, o tempo por c.i.m. é  $O(n)$ . O espaço requerido é  $O(n + m)$ .

**Observação 4.3:**

Uma busca em profundidade irrestrita (B.I.P.) em um digrafo  $D$  é um processo sistemático de se percorrer os vértices e as arestas de  $D$ , de forma que:

- marca-se o vértice inicial que é uma fonte (é a raiz da busca)
- no passo geral, seleciona-se algum vértice marcado  $v$  incidente a alguma aresta  $(v,w)$  ainda não explorada, marca-se  $w$
- o vértice marcado é escolhido como aquele mais recentemente alcançado na busca, dentre todos os vértices marcados
- cada aresta é visitada um número finito e qualquer de vezes
- apenas os vértices e as arestas alcançáveis a partir da raiz são incluídos na busca.

Utiliza-se uma pilha  $Q$  para armazenar os vértices marcados. Cada vez que ingressa à pilha  $Q$  um sumidouro, tem-se armazenado em  $Q$  um caminho fonte-sumidouro (na ordem contrária que ingressou a  $Q$ ).

Para obter todos os caminhos fonte-sumidouro, o procedimento de busca deve aplicar-se a partir de cada uma das fontes de  $D$ .

Seja  $S(v) = \{ w \in V / (v,w) \in E(D) \}$  o conjunto de sucessores de  $v$ . O passo 2 do algoritmo 4.1 pode corresponder ao algoritmo 4.2 a seguir.

**Algoritmo 4.2: B.I.P. para caminhos fonte-sumidouro**

Dado o digrafo  $D_R$   
 desmarcar todos os vértices  
 definir uma pilha  $Q$   
 para todo  $s$  fonte de  $D_R$  efetuar  
      $P(s)$

**Procedimento**  $P(v)$   
 colocar  $v$  na pilha  $Q$   
 se  $v$  é sumidouro então efetuar  
     listar o conteúdo da pilha  $Q$   
     retirar  $v$  de  $Q$   
 caso contrário efetuar  
     marcar  $v$   
     para todo  $w$  em  $S(v)$  efetuar  
         se  $w \notin Q$  então  $P(w)$   
     retirar  $v$  de  $Q$   
     desmarcar  $v$ .

Por exemplo, para o caminho  $P_8$  obtém-se o digrafo  $D$  da figura 4.5. Os caminhos fonte-sumidouro em  $D$  são  $\{1,3,5,7\}$ ,  $\{1,3,5,8\}$ ,  $\{1,3,6,8\}$ ,  $\{1,4,6,8\}$ ,  $\{1,4,7\}$ ,  $\{2,4,6,8\}$ ,  $\{2,4,7\}$ ,  $\{2,5,7\}$  e  $\{2,5,8\}$ .

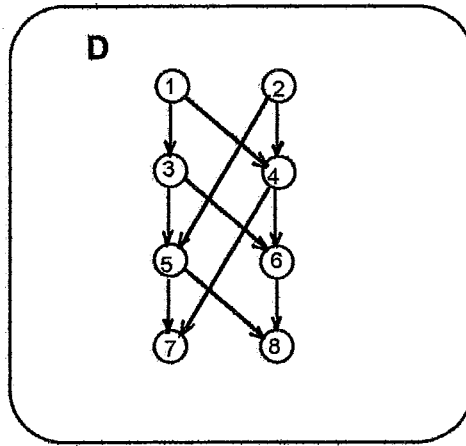


Fig. 4.5: Dígrafo associado a  $P_8$

### Número de c.i.m. de $P_n$

#### Lema 4.3:

O número de c.i.m. de um caminho sem cordas  $P_n$  é dado pela relação:

$$\begin{aligned}
 i(P_n) &= i(P_{n-2}) + i(P_{n-3}) & n \geq 4, \\
 i(P_1) &= 1, \\
 i(P_2) &= 2, \\
 i(P_3) &= 2.
 \end{aligned}$$

#### Prova:

Seja  $I \in F(P_{n-2})$ ,  $I = \{v_1, v_2, \dots, v_t\}$ , existem duas possibilidades:

(a)  $v_{n-2} \in I$ ,  $t = n-2$ , então  $I \cup \{n\} \in F(P_n)$

(b)  $v_{n-3} \in I$ ,  $t = n-3$ , então  $I \cup \{n\} \in F(P_n)$

Seja  $I \in F(P_{n-3})$ ,  $I = \{v_1, v_2, \dots, v_t\}$ , há dois casos:

(c)  $v_{n-3} \in I$ ,  $t = n-3$ , então  $I \cup \{n-1\} \in F(P_n)$

(d)  $v_{n-4} \in I$ ,  $t = n-4$ , então  $I \cup \{n-1\} \in F(P_n)$ .

O c.i.m.  $I$  do caso (b) é igual ao c.i.m.  $I$  do caso (c), depois do vértice  $(n-3)$  em  $I$  pode seguir o  $(n-1)$  ou o  $n$ . Para construir  $F(P_n)$  considera-se que ao c.i.m.  $I$  do caso (b) acrescenta-se o vértice  $n$  e ao c.i.m.  $I$  do caso (c) o  $(n-1)$ .

O c.i.m.  $I$  do caso (d) está contido no c.i.m.  $I$  do caso (a), depois do vértice  $(n-4)$  em  $I$  pode seguir o  $(n-2)$  ou o  $(n-1)$ , depois do vértice  $(n-2)$  em  $I$  só pode seguir o  $n$ ; considerando o c.i.m.  $I$  de (d),  $I \cup \{n-2\}$  corresponde ao c.i.m.  $I$  do caso (a). Para gerar  $F(P_n)$  considera-se que ao c.i.m.  $I$  do caso (a) acrescenta-se  $\{n\}$  e ao c.i.m.  $I$  do caso (d) é acrescentado  $\{n-1\}$ .



**Observação 4.4:**

A prova do lema 4.3 fornece um algoritmo recursivo para construir  $F(P_n)$  a partir de  $F(P_{n-2})$  e  $F(P_{n-3})$ .

Por exemplo, para obter  $F(P_7)$  a partir de  $F(P_4)$  e de  $F(P_5)$ , tem-se:

$$F(P_5) = \{ \{1,3,5\}, \{1,4\}, \{2,4\}, \{2,5\} \} \quad F(P_4) = \{ \{1,3\}, \{1,4\}, \{2,4\} \}$$

- (a)  $\{1,3,5\} \in F(P_5) \Rightarrow \{1,3,5,7\} \in F(P_7)$   
 $\{2,5\} \in F(P_5) \Rightarrow \{2,5,7\} \in F(P_7)$
- (b)  $\{1,4\} \in F(P_5) \Rightarrow \{1,4,7\} \in F(P_7)$   
 $\{2,4\} \in F(P_5) \Rightarrow \{2,4,7\} \in F(P_7)$
- (c)  $\{1,4\} \in F(P_4) \Rightarrow \{1,4,6\} \in F(P_7)$   
 $\{2,4\} \in F(P_4) \Rightarrow \{2,4,6\} \in F(P_7)$
- (d)  $\{1,3\} \in F(P_4) \Rightarrow \{1,3,6\} \in F(P_7)$ .

**Observação 4.5:**

A equação de recorrências

$$i(P_n) = i(P_{n-2}) + i(P_{n-3})$$

é equivalente à equação  $1 + x = x^3$

apresentada por Füredi (1987) cujas soluções são:  $x_1 = 1,32472$

$$x_2 = -0,662359 + 0,56228i$$

$$x_3 = -0,662359 - 0,56228i.$$

**Observação 4.6:**

O número de c.i.m. de  $P_n$  satisfaz a relação:

$$i(P_n) \leq 2 \alpha^{n-2},$$

onde  $\alpha = x_1$  é a única solução real da equação  $1 + x = x^3$ .



## 4.2. Ciclos sem cordas

Um ciclo sem cordas ou ciclo induzido  $C_n$  é um grafo com  $n$  vértices  $v_1, v_2, \dots, v_n$  tal que  $(v_i, v_{i+1}) \in E$  para  $i = 1, \dots, n$ , considerando  $v_{n+1} = v_1$ ,  $n \geq 3$ ; e tal que não existe aresta entre nenhum outro par de vértices.

Os vértices de  $C_n$  são denotados por  $1, 2, \dots, n$ .

Na figura 4.6 apresentam-se os ciclos sem cordas  $C_3$  e  $C_4$ .

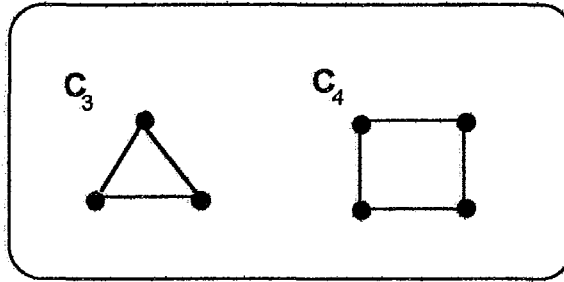


Fig. 4.6: Ciclos  $C_3$  e  $C_4$

Seja:  $C_n$  um ciclo de  $n$  vértices sem cordas,

$F(C_n)$  a família de conjuntos independentes maximais de  $C_n$ ,

$i(C_n) = \text{card}(F(C_n)) =$  número de c.i.m. de  $C_n$ .

Um conjunto independente maximal é descrito pela seqüência de vértices que o compõem. Por exemplo para  $C_4$ ,  $I = 1,3$  representa o c.i.m.  $I = \{1,3\}$ . A família de c.i.m. de  $C_4$  é  $F(C_4) = \{\{1,3\}, \{2,4\}\}$ .

Na tabela 4.2 apresenta-se a família  $F(C_n)$  e o número  $i(C_n)$  de conjuntos independentes maximais dos ciclos induzidos  $C_3$  a  $C_9$ .

Tabela 4.2: C.i.m. de  $C_3, \dots, C_9$

$n$	$F(C_n)$	$i(C_n)$
3	$\{1\}; \{2\}; \{3\}$	3
4	$\{1,3\}; \{2,4\}$	2
5	$\{1,3\}; \{1,4\}; \{2,4\}; \{2,5\}; \{3,5\}$	5
6	$\{1,3,5\}; \{1,4\}; \{2,4,6\}; \{2,5\}; \{3,6\}$	5
7	$\{1,3,5\}; \{1,3,6\}; \{1,4,6\}; \{2,4,6\}; \{2,4,7\}; \{2,5,7\}; \{3,5,7\}$	7
8	$\{1,3,5,7\}; \{1,3,6\}; \{1,4,6\}; \{1,4,7\};$ $\{2,4,6,8\}; \{2,4,7\}; \{2,5,7\}; \{2,5,8\}; \{3,5,8\}; \{3,6,8\}$	10
9	$\{1,3,5,7\}; \{1,3,5,8\}; \{1,3,6,8\}; \{1,4,6,8\}; \{1,4,7\}; \{2,4,6,8\};$ $\{2,4,6,9\}; \{2,4,7,9\}; \{2,5,7,9\}; \{2,5,8\}; \{3,5,7,9\}; \{3,6,9\}$	12

**Observação 4.7:**

Considerando cada c.i.m.  $I$  como uma seqüência de vértices  $I = v_1, v_2, \dots, v_k$ , observa-se:

- o primeiro vértice da seqüência é 1 ou 2 ou 3
- o último vértice da seqüência é  $n$  ou  $(n-1)$  ou  $(n-2)$
- depois do vértice  $i$  pode seguir o vértice  $(i+2)$  ou o  $(i+3)$ .

**Algoritmo para gerar a família  $F(C_n)$  de c.i.m. de  $C_n$**

Para gerar a família  $F(C_n)$  de conjuntos independentes maximais de  $C_n$ , trabalha-se com dois caminhos que são subgrafos induzidos de  $C_n$ :  $P_{n-1}$  e  $P_{2,n}$  onde  $P_{n-1}$  é o caminho sem cordas com vértices 1, 2, ...,  $n-1$  e  $P_{2,n}$  é o caminho sem cordas com vértices 2, 3, ...,  $n$ .

Por exemplo, para  $C_3$  consideram-se os caminhos  $P_2$  e  $P_{2,3}$  apresentados na figura 4.7, para  $C_4$  os caminhos  $P_3$  e  $P_{2,4}$  da mesma figura.

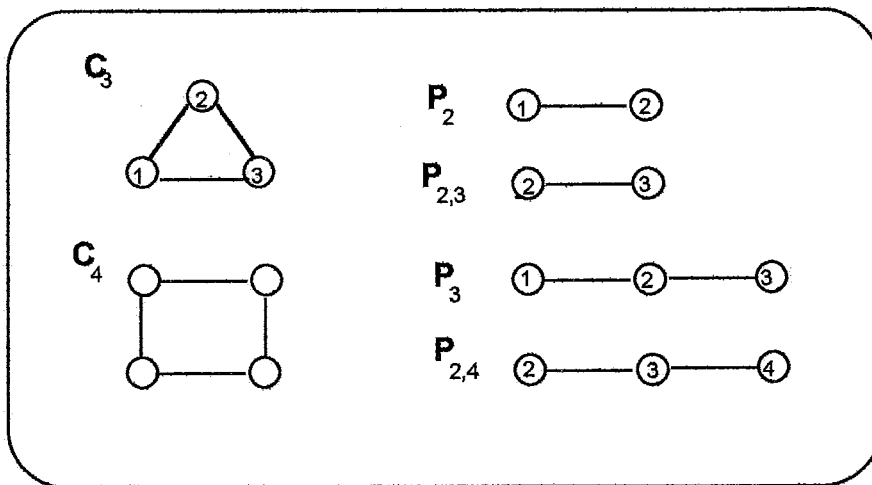


Fig. 4.7: Ciclos  $C_3$  e  $C_4$ , caminhos  $P_2$ ,  $P_{2,3}$ ,  $P_3$  e  $P_{2,4}$

Segundo lema 4.1:

- a família  $F(P_{n-1})$  de c.i.m. de  $P_{n-1}$  corresponde ao conjunto de caminhos da raiz até as folhas das árvores  $T_1$  e  $T_2$  seguintes:

$T_1$  tem raiz 1, folhas  $n-1$  e  $n-2$ ,  $T_2$  tem raiz 2, folhas  $n-1$  e  $n-2$ ,

em  $T_1$  e  $T_2$  o vértice  $i$  tem dois filhos:  $i+2$  e  $i+3$  para  $i \leq n-4$ , o vértice  $n-3$  tem só um filho o  $n-1$ ,

- a família  $F(P_{2,n})$  de c.i.m. de  $P_{2,n}$  corresponde ao conjunto de caminhos da raiz até as folhas das árvores  $T_3$  e  $T_4$  seguintes:

$T_3$  tem raiz 2, folhas  $n-1$  e  $n$ ,  $T_4$  tem raiz 3, folhas  $n-1$  e  $n$ ,

em  $T_3$  e  $T_4$  o vértice  $i$  tem dois filhos:  $i+2$  e  $i+3$ ,  $i \leq n-3$ , o vértice  $n-2$  tem, apenas, um filho o  $n$ .

**Observação 4.8:**

$T_2$  é uma subárvore de  $T_3$ .

Se  $n$  par:

em  $F(P_{n-1})$  aparecem os caminhos  $2, 4, 6, \dots, n-2$  e  $2, 5, \dots, n-1$  de  $T_2$ ,  
em  $F(P_{2,n})$  aparecem os caminhos  $2, 4, 6, \dots, n-2, n$  e  $2, 5, \dots, n-1$  de  $T_3$ .

Se  $n$  ímpar:

em  $T_1$  está o caminho:  $1, 3, 5, \dots, n-2$  e o caminho  $1, 3, 6, \dots, n-1$ ,  
em  $T_4$  está o caminho:  $3, 5, \dots, n$  e o caminho  $3, 6, \dots, n-1$ ,  
em  $F(P_{n-1})$  aparecem os caminhos  $2, 4, \dots, n-1$  e  $2, 5, \dots, n-2$  associados a  $T_2$ ,  
em  $F(P_{2,n})$  aparecem os caminhos  $2, 4, \dots, n-1$  e  $2, 5, \dots, n$  associados a  $T_3$ .

**Lema 4.4:**

A família  $F(C_n)$  de c.i.m do ciclo sem cordas  $C_n$  corresponde ao conjunto de caminhos da raiz até as folhas das árvores  $A_1$  e  $A_2$ , e ao conjunto de caminhos da raiz até a folha  $n$  da árvore  $A_3$ , as quais são descritas abaixo.

$A_1$  tem raiz  $1$ , folhas  $n-2$  e  $n-1$ ,

$A_2$  tem raiz  $2$ , folhas  $n-1$  e  $n$ ,

$A_3$  tem raiz  $3$ , folhas  $n-1$  e  $n$ ,

em  $A_1, A_2$  e  $A_3$  o vértice  $i$  tem dois filhos:  $i+2$  e  $i+3$  se  $i < n-3$ ,

em  $A_1$  o vértice  $n-3$  tem o filho  $n-1$ ,

em  $A_2$  e  $A_3$  o vértice  $n-3$  tem os filhos  $n-1$  e  $n$ ,

em  $A_2$  e  $A_3$  o vértice  $n-2$  tem o filho  $n$ .

**Prova:**

Dadas as árvores binárias  $A_1, A_2$  e  $A_3$  construídas segundo o lema 4.4, tem-se:

- cada caminho  $I = v_1, v_2, \dots, v_k$  em  $A_j$ , para  $j = 1, 2$ , desde a raiz  $j = v_1$  até uma folha  $v_k$  corresponde a um c.i.m de  $C_n$  pois, em  $A_j$  existe o arco  $(v_p, v_q)$  se e somente se a aresta  $(v_p, v_q) \notin E(C_n)$ , isto é sse  $q \neq p + 1$ , com  $q < n$ , logo  $I$  é conjunto independente e  $I$  é maximal pois não é possível acrescentar mais vértices a  $I$ , dado que: a raiz  $v_1$  é  $1$  ou  $2$ ,  $(1, 2) \in E(C_n)$ ; se  $v_p \in I$  então  $v_{p+2} \in I$  ou  $v_{p+3} \in I$ ,  $(v_p, v_{p+1}) \in E(C_n)$ , e a folha é  $n-2$  ou  $n-1$  em  $A_1$  dado que  $(1, n), (n-1, n) \in E(C_n)$ , e em  $A_2$  a folha é  $n-1$  ou  $n$ . Os caminhos tem que ser raiz-folha, caso contrário o c.i.m. não é maximal.

- cada caminho  $I = v_1, v_2, \dots, v_k$  em  $A_3$  desde a raiz  $v_1 = 3$  até a folha  $v_k = n$  corresponde a um c.i.m de  $C_n$  pois, em  $A_3$  existe o arco  $(v_p, v_q)$  se e somente se  $(v_p, v_q) \notin E(C_n)$ , isto é sse  $q \neq p + 1$ , com  $q < n$ , logo  $I$  é conjunto independente e  $I$  é maximal pois não é possível acrescentar mais vértices a  $I$ , dado que:  $(1, n) \in E(C_n)$ . Se  $v_p \in I$  então  $v_{p+2} \in I$  ou  $v_{p+3} \in I$ , pois  $(v_p, v_{p+1}) \in E(C_n)$ . Um caminho  $I'$  desde a raiz até a folha  $n-1$  em  $A_3$  não é maximal pois está incluído no c.i.m.  $I = I' \cup \{1\}$ .

Dado um c.i.m.  $I = \{v_1, v_2, \dots, v_k\}$  de  $C_n$ , a seqüência  $I = v_1, v_2, \dots, v_k$  corresponde a um caminho raiz-folha em  $A_j$ , pois como  $I$  é maximal então  $v_1$  é  $1, 2$  ou  $3$ ,  $v_k$  é  $n-2, n-1$  ou  $n$ , se  $v_p \in I$  então  $v_{p+2} \in I$  ou  $v_{p+3} \in I$ , não existe  $(v_p, v_{p+1}) \in E(D)$  pois  $(i, i+1) \in E(C_n)$ , e não existe  $(v_p, v_q) \in E(D)$  com  $v_q > v_p + 3$  pois não seria maximal.



Por exemplo, na figura 4.8 exibe-se o ciclo  $C_7$ , as árvores  $A_1, A_2, A_3$  associadas a  $C_7$ , os caminhos raiz-folha de  $A_1$  e de  $A_2$ , e os caminhos da raiz até a folha marcada 7 da árvore  $A_3$ .

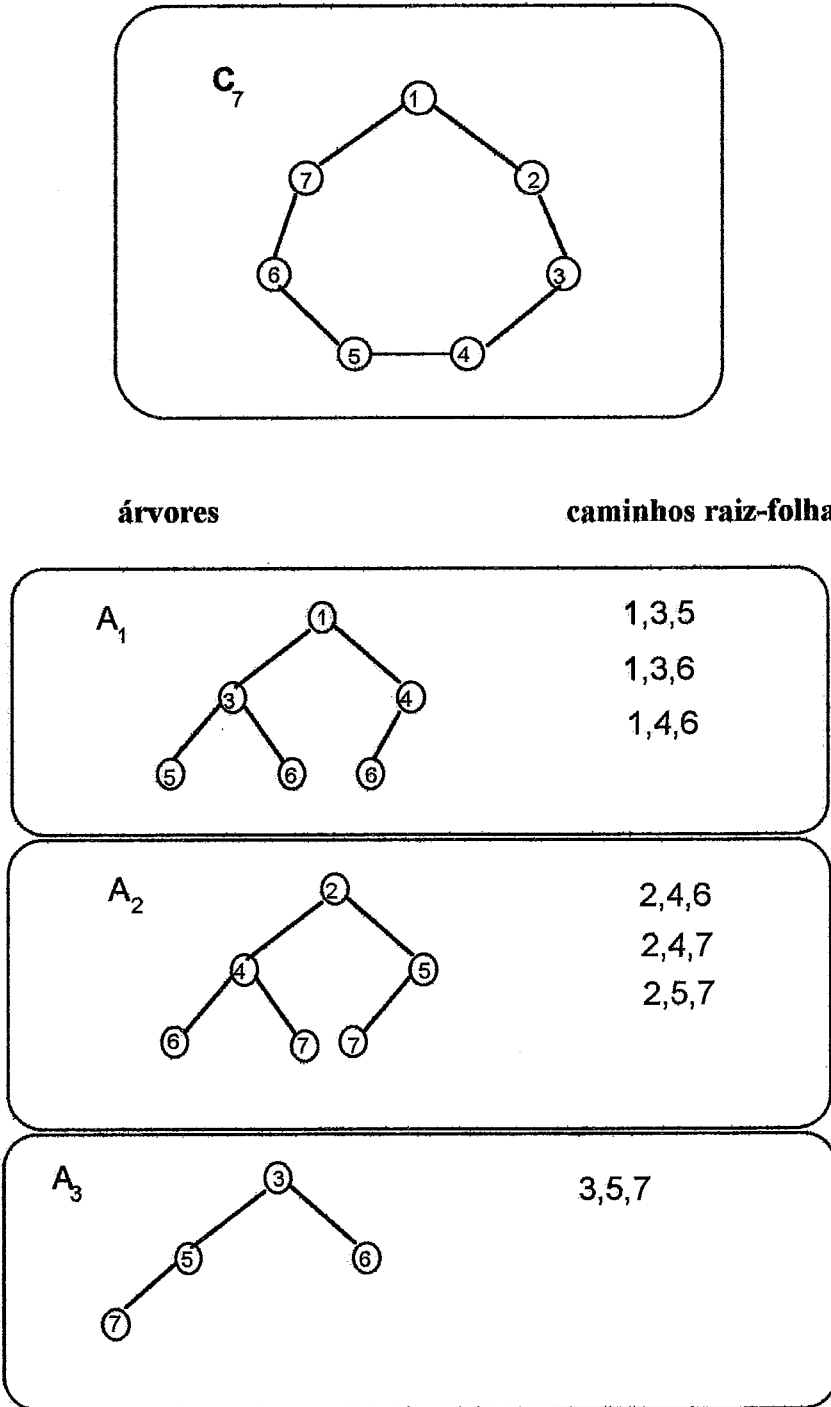


Fig. 4.8: c.i.m. de  $C_7$

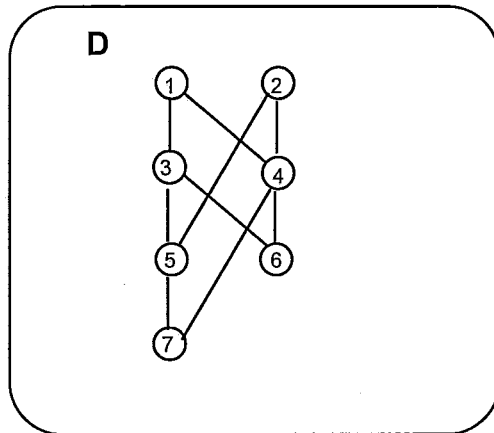
Logo  $F(C_7) = \{ \{1,3,5\}; \{1,3,6\}; \{1,4,6\}; \{2,4,6\}; \{2,4,7\}; \{2,5,7\}; \{3,5,7\} \}$ ,  
 $i(C_7) = 7$ .

**Observação 4.9:**

O lema 4.4 descreve um algoritmo para gerar  $F(C_n)$ , construindo as árvores binárias  $A_1$ ,  $A_2$  e  $A_3$ , o qual tem complexidade de espaço exponencial. Para evitar este problema, transformam-se as três árvores em um digrafo acíclico  $D$ , com conjunto de vértices  $V(D)$  igual ao de  $C_n$ ,  $V(D) = V(C_n) = \{1, 2, \dots, n\}$ ; existe a aresta  $(v, w)$  em  $E(D)$  se e somente se  $w = v + 2$  ou  $w = v + 3$ . Dessa forma, as fontes de  $D$  são 1 e 2, os sumidouros de  $D$  são  $n - 1$  e  $n$ . Esta transformação é exprimida no lema 4.5 que vem logo após o exemplo, dispensando a prova pois esta é similar à prova do lema 4.4. Este digrafo é o mesmo que o construído no caso de caminho induzido.

Ao gerar os caminhos fonte-sumidouro de  $D$ , é possível conferir se a fonte utilizada é 1 e o sumidouro é  $n$ , nesse caso o caminho gerado pode induzir dois c.i.m.:  $\{1, 3, \dots, n-2\}$  e  $\{3, 5, \dots, n\}$ ; se o caminho gerado contém o vértice 3 então  $\{3, 5, \dots, n\}$  é c.i.m. de  $C_n$ ; senão, isto é o caminho começa em 1 mas não contém o vértice 3 então ele é descartado pois não é c.i.m. de  $C_n$ . Também, deve-se conferir se  $n - 2$  está no caminho, nesse caso  $\{1, 3, \dots, n - 2\}$  é c.i.m..

Por exemplo, na figura 4.9 apresenta-se o digrafo  $D$  associado a  $C_7$ . Os caminhos maximais fonte-sumidouro em  $D$  são  $\{1, 3, 5, 7\}$ ,  $\{1, 3, 6\}$ ,  $\{1, 4, 6\}$ ,  $\{1, 4, 7\}$ ,  $\{2, 4, 6\}$ ,  $\{2, 4, 7\}$ ,  $\{2, 5, 7\}$ ; onde nem todos esses caminhos correspondem a c.i.m. de  $C_7$  dado que  $(1, 7) \in E(C_7)$ . Considerando o caminho  $I = \{1, 4, 7\}$ , observa-se que ele contém o vértice 1 mas não contém o vértice 3, logo ele não é considerado pois  $I - \{1\} = \{4, 7\}$  não é maximal.

Fig. 4.9: Digrafo associado a  $C_7$ **Lema 4.5:**

A família  $F(C_n)$  de c.i.m. de  $C_n$  obtém-se a partir do conjunto de caminhos maximais fonte-sumidouro do digrafo acíclico  $D$  com  $V(D) = \{1, 2, \dots, n\}$  e tal que existe a aresta  $(v, w)$  em  $E(D)$  se e somente se  $w = v + 2$  ou  $w = v + 3$ . As fontes de  $D$  são 1 e 2, os sumidouros são  $n - 1$  e  $n$ . Se o caminho fonte-sumidouro de  $D$  começa na fonte 1 e termina no sumidouro  $n$ , deve-se conferir: primeiro, se o caminho gerado contém o vértice 3 então  $\{3, 5, \dots, n\}$  é c.i.m. de  $C_n$ ; segundo, se o caminho gerado contém o vértice  $n - 2$  então  $\{1, 3, \dots, n - 2\}$  é c.i.m. de  $C_n$ .

**Algoritmo:**

Dado  $n$ , desenvolve-se um algoritmo para construir a família  $F$  de c.i.m de  $C_n$ , utilizando a caracterização pelo dígrafo  $D$  descrito no lema 4.5. O algoritmo 4.3 implementa este procedimento.

**Algoritmo 4.3: Família de c.i.m. de um ciclo induzido**

Dado  $n$

1. Construir o dígrafo  $D$  com  $V(D) = V(G)$  e  $(v,w) \in E(D) \Leftrightarrow w = v + 2$  ou  $w = v + 3$
2. Gerar a família  $F$  de todos os caminhos fonte-sumidouro de  $D$
3. Para todo caminho  $I = v_1, v_2, \dots, v_k$  de  $F$  efetuar
  - 3.1 se  $v_1 = 1$  e  $v_k = n$  então efetuar retirar  $I$  de  $F$
  - se  $v_2 = 3$  então efetuar  $F = F \cup [I - \{v_1\}]$
  - se  $v_{k-1} = n - 2$  então efetuar  $F = F \cup [I - \{v_k\}]$ .

**Complexidade:**

Passo 1:  $O(n)$

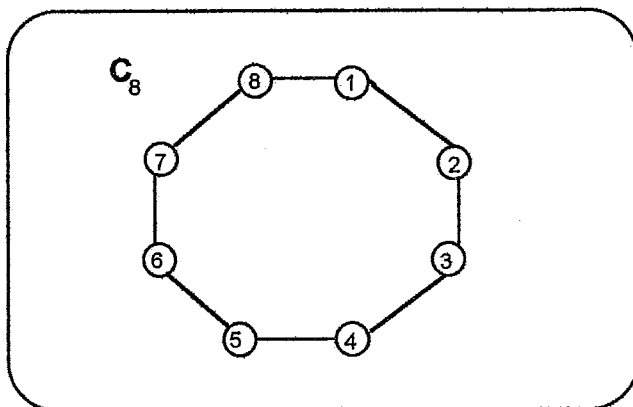
Passo 2:  $O(n)$  por caminho, ao fazer busca em profundidade irrestrita (ver observação 4.3)

Passo 3: é executado  $i(C_n)$  vezes

3.1:  $O(n)$

A complexidade total do algoritmo 4.3 é  $O(n \cdot i(C_n))$ , possui tempo polinomial no tamanho do grafo e no número de c.i.m. do grafo. No entanto, o tempo por c.i.m. é  $O(n)$ . O espaço requerido é  $O(n + m)$ .

Na figura 4.10 exibe-se o ciclo  $C_8$ , o dígrafo  $D$  associado a  $C_8$  e os caminhos fonte-sumidouro de  $D$ .



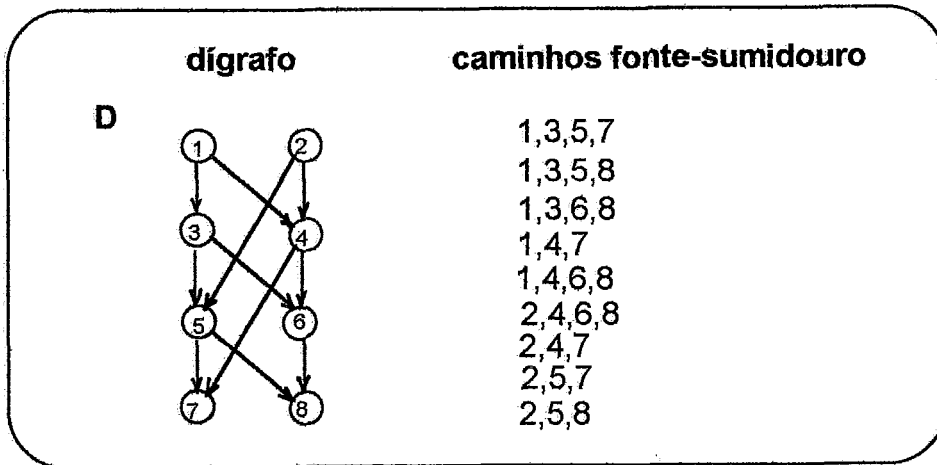


Fig. 4.10: c.i.m. de  $C_8$

Com isto

$$F = \{\{1,3,5,7\}, \{1,3,5,8\}, \{1,3,6,8\}, \{1,4,6,8\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}\};$$

aplicando o passo 3 do algoritmo 4.3 tem-se:

$$\{1,3,5,7\} \Rightarrow$$

$$F = \{\{1,3,5,7\}, \{1,3,5,8\}, \{1,3,6,8\}, \{1,4,6,8\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}\}$$

$$\{1,3,5,8\} \Rightarrow$$

$$F = \{\{1,3,5,7\}, \{1,3,6,8\}, \{1,4,6,8\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}\}$$

$$\{1,3,6,8\} \Rightarrow F =$$

$$\{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6,8\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}$$

$$\{1,4,6,8\} \Rightarrow F =$$

$$\{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}$$

$$\{1,4,7\} \Rightarrow F =$$

$$\{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}$$

$$\{2,4,6,8\} \Rightarrow F =$$

$$\{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}$$

$$\{2,4,7\} \Rightarrow F =$$

$$\{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}$$

$$\{2,5,7\} \Rightarrow F =$$

$$\{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}$$

$$\{2,5,8\} \Rightarrow F =$$

$$\{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\}, \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}$$

logo:  $F(C_8) = \{\{1,3,5,7\}, \{1,3,6\}, \{1,4,6\}, \{1,4,7\},$   
 $\{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\}, \{3,5,8\}, \{3,6,8\}\}.$

## Número de c.i.m. de $C_n$

### Lema 4.6:

O número de c.i.m. de um ciclo sem cordas  $C_n$  é dado pela relação:

$$\begin{aligned} i(C_n) &= i(C_{n-2}) + i(C_{n-3}) && \text{para } n \geq 6, \\ i(C_3) &= 3, \\ i(C_4) &= 2, \\ i(C_5) &= 5. \end{aligned}$$

### **Prova:**

Seja  $I \in F(C_{n-2})$ ,  $I = \{v_1, v_2, \dots, v_k\}$ , existem três possibilidades:

(a)  $v_k = n-2$ ,  $n-2 \in I$  ( $1 \notin I$ ) então  $I \cup \{n\} \in F(C_n)$

(b)  $v_k = n-3$ ,  $n-3 \in I$

(b1)  $1 \in I$  então  $I \cup \{n-1\} \in F(C_n)$

(b2)  $1 \notin I$  então  $I \cup \{n\} \in F(C_n)$

(c)  $v_k = n-4$ ,  $n-4 \in I$  ( $1 \in I$ ) então  $I \cup \{n-1\} \in F(C_n)$

Seja  $I \in F(C_{n-3})$ ,  $I = \{v_1, v_2, \dots, v_k\}$ , há três casos:

(d)  $v_k = n-3$ ,  $n-3 \in I$  ( $1 \notin I$ )

(d1)  $3 \notin I$  então  $I \cup \{n-1\} \in F(C_n)$

(d2)  $3 \in I$  então  $I \cup \{n\} \in F(C_n)$

(e)  $v_k = n-4$ ,  $n-4 \in I$ .

(e1)  $1 \notin I$  então  $I \cup \{n-1\} \in F(C_n)$

(e2)  $1 \in I$  então  $I \cup \{n-2\} \in F(C_n)$

(f)  $v_k = n-5$ ,  $n-5 \in I$  ( $1 \in I$ ) então  $I \cup \{n-2\} \in F(C_n)$

O c.i.m.  $I \in F(C_{n-2})$  do caso (c) é igual ao c.i.m.  $I \in F(C_{n-3})$  do caso (e2), depois do vértice ( $n-4$ ) em  $I$  pode seguir o ( $n-2$ ) ou o ( $n-1$ ). Para construir  $F(C_n)$  considera-se o c.i.m.  $I$  do caso (c), acrescenta-se o vértice ( $n-1$ ) e ao c.i.m.  $I$  do caso (e2) é acrescentado o vértice ( $n-2$ ).



O c.i.m.  $I \in F(C_{n-2})$  do caso (b2) é igual ao c.i.m.  $I \in F(C_{n-3})$  do caso (d1), depois do vértice  $(n-3)$  em  $I$  pode seguir o  $(n)$  ou o  $(n-1)$ , pois  $1 \notin I$ . Para construir  $F(C_n)$  considera-se o c.i.m.  $I$  do caso (b2) acrescenta-se o vértice  $(n)$  e ao c.i.m.  $I$  do caso (d1) é acrescentado o vértice  $(n-1)$ .

O c.i.m.  $I \in F(C_{n-3})$  do caso (f) está contido no c.i.m.  $I \in F(C_{n-2})$  do caso (b1), considerando o c.i.m.  $I$  de (f),  $I \cup \{n-3\}$  corresponde ao c.i.m.  $I$  do caso (b1), depois do vértice  $(n-3)$  pode seguir o  $(n-1)$  mas não o vértice  $n$  pois  $1 \in I$ . Para gerar  $F(C_n)$  considera-se o c.i.m.  $I$  do (b1) união  $\{n-1\}$  e ao c.i.m.  $I$  do (f) acrescenta-se o  $(n-2)$ .

O c.i.m.  $I \in F(C_{n-3})$  do caso (d2) está contido no c.i.m.  $I \in F(C_{n-2})$  do caso (b1), considerando o c.i.m.  $I$  de (d2),  $I \cup \{1\}$  corresponde ao c.i.m.  $I$  do caso (b1), depois do vértice  $(n-3)$  pode seguir o  $(n-1)$  e o vértice  $n$  se  $1 \notin I$ . Para gerar  $F(C_n)$  considera-se o c.i.m.  $I$  do (b1) união  $\{n-1\}$  e ao c.i.m.  $I$  do (d2) acrescenta-se o  $(n)$ .

O c.i.m.  $I \in F(C_{n-3})$  do caso (e1) está contido no c.i.m.  $I \in F(C_{n-2})$  do caso (a), considerando o c.i.m.  $I$  de (e1),  $I \cup \{n-2\}$  corresponde ao c.i.m.  $I$  do caso (a), depois do vértice  $(n-2)$  pode seguir só o  $(n)$  se  $1 \notin I$ , depois do vértice  $(n-4)$  pode seguir o  $(n-1)$  ou o  $(n-2)$ . Para gerar  $F(C_n)$  considera-se o c.i.m.  $I$  do (a) união  $\{n\}$  e ao c.i.m.  $I$  do (e1) acrescenta-se o  $(n-1)$ .



**Observação 4.10:**

A prova do lema 4.4 produz um algoritmo recursivo para gerar a família  $F(C_n)$  a partir das famílias  $F(C_{n-2})$  e  $F(C_{n-3})$ .

Por exemplo, para obter  $F(C_9)$  a partir de  $F(C_6)$  e de  $F(C_7)$ , tem-se:

$$F(C_7) = \{ \{1,3,5\}, \{1,3,6\}, \{1,4,6\}, \{2,4,6\}, \{2,4,7\}, \{2,5,7\}, \{3,5,7\} \}$$

$$F(C_6) = \{ \{1,3,5\}, \{1,4\}, \{2,4,6\}, \{2,5\}, \{3,6\} \}$$

$$(a) \{2,4,7\} \in F(C_7) \Rightarrow \{2,4,7,9\} \in F(C_9)$$

$$\{2,5,7\} \in F(C_7) \Rightarrow \{2,5,7,9\} \in F(C_9)$$

$$\{3,5,7\} \in F(C_7) \Rightarrow \{3,5,7,9\} \in F(C_9)$$

$$(b1) \{1,3,6\} \in F(C_7) \Rightarrow \{1,3,6,8\} \in F(C_9)$$

$$\{1,4,6\} \in F(C_7) \Rightarrow \{1,4,6,8\} \in F(C_9)$$

$$(b2) \{2,4,6\} \in F(C_7) \Rightarrow \{2,4,6,9\} \in F(C_9)$$

$$(c) \{1,3,5\} \in F(C_7) \Rightarrow \{1,3,5,8\} \in F(C_9)$$

$$(d1) \{2,4,6\} \in F(C_6) \Rightarrow \{2,4,6,8\} \in F(C_9)$$

$$(d2) \{3,6\} \in F(C_6) \Rightarrow \{3,6,9\} \in F(C_9)$$

$$(e1) \{2,5\} \in F(C_6) \Rightarrow \{2,5,8\} \in F(C_9)$$

$$(e2) \{1,3,5\} \in F(C_6) \Rightarrow \{1,3,5,7\} \in F(C_9)$$

$$(f) \{1,4\} \in F(C_6) \Rightarrow \{1,4,7\} \in F(C_9).$$

**Observação 4.11:**

A equação de recorrências

$$i(C_n) = i(C_{n-2}) + i(C_{n-3})$$

é equivalente à equação

$$1 + x = x^3$$

apresentada por Füredi (1987) cujas soluções são:

$$\begin{aligned} x^1 &= 1,32472 \\ x^2 &= -0,662359 + 0,56228i \\ x^3 &= -0,662359 - 0,56228i. \end{aligned}$$

Com isto tem-se o seguinte limite para o número de c.i.m. de  $C_n$ :

$$i(C_n) \leq 3 \alpha^{n-3}$$

onde  $\alpha = x^1$  é a única solução real da equação  $1 + x = x^3$ .

### 4.3. Grafos grade completas de duas linhas

Um grafo grade completa  $G_{q,p}$  é aquele cujo conjunto de vértices  $V(G_{q,p})$  é o conjunto de pares ordenados  $(i,j)$ , com  $1 \leq i \leq q$ ,  $1 \leq j \leq p$ . Existe uma aresta entre os vértices  $v = (v_i, v_j)$  e  $w = (w_i, w_j)$  se  $|v_i - w_i| = 1$  e  $|v_j - w_j| = 0$  ou  $|v_i - w_i| = 0$  e  $|v_j - w_j| = 1$ . A dimensão da grade é o número de vértices  $qp$  do grafo.

A grade  $G_{q,p}$  representa-se graficamente em  $q$  linhas e  $p$  colunas, numerando as linhas de cima para baixo e as colunas da esquerda para a direita. Se considera  $q \leq p$  pois as grades  $G_{q,p}$  e  $G_{p,q}$  são isomorfas.

Na figura 4.11 apresenta-se a grade completa  $G_{3,7}$  de 3 linhas e 7 colunas.

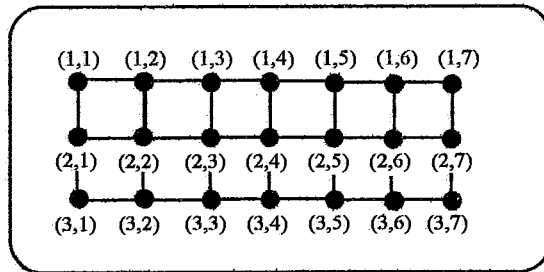


Fig. 4.11: Grade  $G_{3,7}$

Para simplificar o desenho da grade se escreve o número de ordem da linha e o da coluna, segundo a regra especificada acima, como é exibido na figura 4.12.

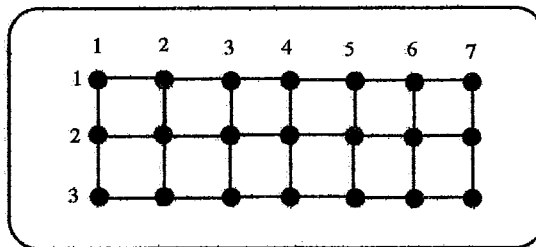


Fig. 4.12: Exemplo de notação, grade  $G_{3,7}$

#### Casos particulares:

$q = p = 1$ : corresponde a um vértice isolado  $V = \{v\}$

$q = k, p = 1$ : corresponde ao caminho induzido  $P_k$

$q = 1, p = k$ : corresponde ao caminho induzido  $P_k$

$q = p = 2$ : corresponde ao ciclo induzido  $C_4$

Um conjunto independente maximal (c.i.m.) de  $G_{q,p}$  é descrito pelo conjunto de pares ordenados que representam aos vértices que o compõem. Por exemplo para  $G_{3,7}$ ,  $I = \{(1,1), (2,2), (1,3), (2,4), (1,5), (2,6), (3,1), (3,3), (3,5), (3,7)\}$  é um c.i.m.

Seja:  $G_{q,p}$  uma grade de  $qp$  vértices,

$F(G_{q,p})$  a família de conjuntos independentes maximais de  $G_{q,p}$ ,

$i(G_{q,p}) = \text{card}(F(G_{q,p})) = \text{número de c.i.m. de } G_{q,p}$ .

Considere por exemplo  $G_{2,3}$ , a família de c.i.m. de  $G_{2,3}$  é

$F(G_{2,3}) = \{ \{(1,1), (2,2), (1,3)\}, \{(1,1), (2,3)\}, \{(1,2), (2,1), (2,3)\}, \{(2,1), (1,3)\} \}$ ,  
o número de c.i.m. de  $G_{2,3}$  é  $i(G_{2,3}) = 4$ .

### Grades completas $G_{2,p}$

Na figura 4.13 apresenta-se  $G_{2,p}$  para  $p = 1, \dots, 5$ .

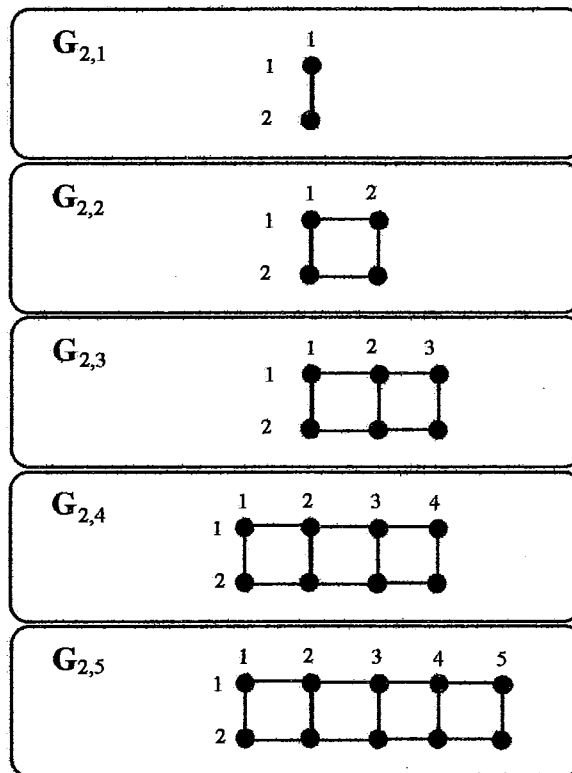


Fig. 4.13: Grades  $G_{2,1}, \dots, G_{2,5}$

Na tabela 4.3 exibe-se a família  $F(G_{2,p})$  e o número  $i(G_{2,p})$  de c.i.m. dos grafos grade completas  $G_{2,p}$  para  $1 \leq p \leq 5$ .

**Tabela 4.3:** C.i.m. de  $G_{2,1}, \dots, G_{2,5}$

$p$	$F(G_{2,p})$	$i(G_{2,p})$
1	$\{(1,1)\}, \{(2,1)\}$	2
2	$\{(1,1),(2,2)\}, \{(2,1),(1,2)\}$	2
3	$\{(1,1),(2,2),(1,3)\}, \{(1,1),(2,3)\}, \{(2,1),(1,2),(2,3)\}, \{(2,1),(1,3)\}$	4
4	$\{(1,1),(2,2),(1,3),(2,4)\}, \{(1,1),(2,2),(1,4)\}, \{(1,1),(2,3),(1,4)\},$ $\{(2,1),(1,2),(2,3),(1,4)\}, \{(2,1),(1,2),(2,4)\}, \{(2,1),(1,3),(2,4)\}$	6
5	$\{(1,1),(2,2),(1,3),(2,4),(1,5)\}, \{(1,1),(2,2),(1,3),(2,5)\},$ $\{(1,1),(2,2),(1,4),(2,5)\}, \{(1,1),(2,3),(1,4),(2,5)\}, \{(1,1),(2,3),(1,5)\},$ $\{(2,1),(1,2),(2,3),(1,4),(2,5)\}, \{(2,1),(1,2),(2,3),(1,5)\},$ $\{(2,1),(1,2),(2,4),(1,5)\}, \{(2,1),(1,3),(2,4),(1,5)\}, \{(2,1),(1,3),(2,5)\}$	10

**Observação 4.12:**

- Considerando cada c.i.m  $I$  como uma seqüência de pares ordenados de vértices  $I = (i_1, j_1), (i_2, j_2), \dots, (i_k, j_k)$  observa-se:
- o primeiro elemento da seqüência é  $(1,1)$  ou  $(2,1)$
  - o último par da seqüência é  $(1,p)$  ou  $(2,p)$
  - depois do par  $(1,j)$  pode seguir o vértice  $(2,j+1)$  ou o  $(2,j+2)$
  - depois do par  $(2,j)$  pode seguir o vértice  $(1,j+1)$  ou o  $(1,j+2)$ .

**Algoritmo para gerar a família  $F(G_{2,p})$  de c.i.m de  $G_{2,p}$**

**Lema 4.7:**

A família  $F(G_{2,p})$  de c.i.m. de  $G_{2,p}$  corresponde ao conjunto de todos os caminhos das raízes até as folhas das duas árvores binárias  $T_1$  e  $T_2$ , mostradas na figura 4.14, onde  $T_1$  está enraizada em  $(1,1)$ ,  $T_2$  está enraizada em  $(2,1)$ ; cada vértice  $(1,j)$  tem dois filhos  $(2,j+1)$  e  $(2,j+2)$ , para  $1 \leq j \leq p-2$ , o vértice  $(1,p-1)$  tem um só filho marcado  $(2,p)$ , cada vértice  $(2,j)$  tem dois filhos  $(1,j+1)$  e  $(1,j+2)$ , para  $1 \leq j \leq p-2$ , o vértice  $(2,p-1)$  tem um só filho marcado  $(1,p)$ ; as folhas das árvores são os vértices marcados  $(1,p)$  ou  $(2,p)$ .

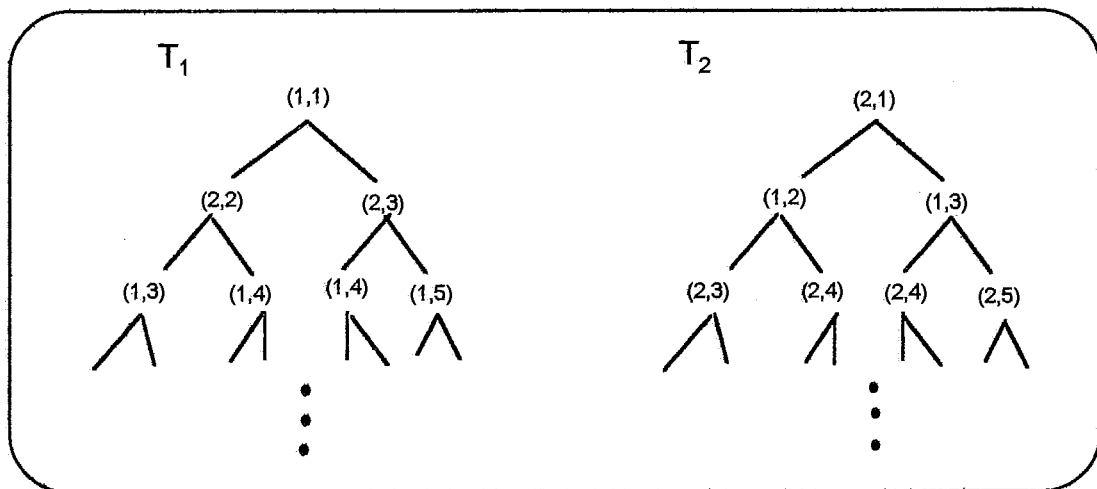


Fig. 4.14: Construção das árvores  $T_1$  e  $T_2$

**Prova:**

Dadas as árvores binárias  $T_1$  e  $T_2$  construídas segundo o lema 4.7, tem-se que cada caminho  $I = (v_1, w_1), (v_2, w_2), \dots, (v_k, w_k)$  em  $T_j$  desde a raiz  $(v_1, w_1)$  até uma folha  $(v_k, w_k)$  corresponde a um c.i.m de  $G_{2,p}$  pois, em  $T_j$  existe um arco de  $(v_p, w_p)$  a  $(v_q, w_q)$  se e somente se tem-se um dos dois casos seguintes:

$$v_p = 1, v_q = 2, w_q = w_p + 1 \text{ ou } w_q = w_p + 2$$

$$v_p = 2, v_q = 1, w_q = w_p + 1 \text{ ou } w_q = w_p + 2,$$

logo é conjunto independente e  $I$  é maximal pois não é possível acrescentar mais vértices a  $I$ , dado que na raiz  $(v_1, w_1)$   $v_1$  é 1 ou 2,  $((1,1), (1,2)) \in E(G_{2,p})$ , se  $(v_p, w_p) \in I$  então  $(v_q, w_{p+1}) \in I$  ou  $(v_q, w_{p+2}) \in I$ ,  $((v_p, w_p), (v_q, w_p)) \in E(G_{2,p})$  com  $|v_q - v_p| = 1$ , e a folha é  $(1, n)$  ou  $(2, n)$ . Os caminhos têm que ser raiz-folha, caso contrário o c.i.m. não é maximal.

Dado um c.i.m.  $I = \{ (v_1, w_1), (v_2, w_2), \dots, (v_k, w_k) \}$  de  $G_{2,p}$ , a seqüência  $I = (v_1, w_1), (v_2, w_2), \dots, (v_k, w_k)$  corresponde a um caminho raiz-folha em  $T_j$ , pois como  $I$  é maximal então  $v_1$  é 1 ou 2, se  $(v_p, w_p) \in I$  então  $(v_q, w_{p+1}) \in I$  ou  $(v_q, w_{p+2}) \in I$  com  $v_q = v_p + 1 \pmod{2}$ , e  $w_k = n$ .



Por exemplo, na figura 4.15 apresentam-se as árvores  $T_1$  e  $T_2$  no caso da grade completa  $G_{2,6}$



- |                                            |                                                  |
|--------------------------------------------|--------------------------------------------------|
| $\{(1,1),(2,2),(1,3),(2,5),(1,7)\},$       | $\{(1,1),(2,2),(1,4),(2,5),(1,7)\},$             |
| $\{(1,1),(2,2),(1,4),(2,5),(1,6),(2,7)\},$ | $\{(1,1),(2,2),(1,4),(2,6),(1,7)\},$             |
| $\{(1,1),(2,3),(1,4),(2,5),(1,7)\},$       | $\{(1,1),(2,3),(1,4),(2,5),(1,6),(2,7)\},$       |
| $\{(1,1),(2,3),(1,4),(2,6),(1,7)\},$       | $\{(1,1),(2,3),(1,5),(2,6),(1,7)\},$             |
| $\{(1,1),(2,3),(1,5),(2,7)\},$             | $\{(2,1),(1,2),(2,3),(1,4),(2,5),(1,6),(2,7)\},$ |
| $\{(2,1),(1,2),(2,3),(1,4),(2,5),(1,7)\},$ | $\{(2,1),(1,2),(2,3),(1,4),(2,6),(1,7)\},$       |
| $\{(2,1),(1,2),(2,3),(1,5),(2,6),(1,7)\},$ | $\{(2,1),(1,2),(2,3),(1,5),(2,7)\},$             |
| $\{(2,1),(1,2),(2,4),(1,5),(2,7)\},$       | $\{(2,1),(1,2),(2,4),(1,5),(2,6),(1,7)\},$       |
| $\{(2,1),(1,2),(2,4),(1,6),(2,7)\},$       | $\{(2,1),(1,3),(2,4),(1,5),(2,6),(1,7)\},$       |
| $\{(2,1),(1,3),(2,4),(1,5),(2,7)\},$       | $\{(2,1),(1,3),(2,4),(1,6),(2,7)\},$             |
| $\{(2,1),(1,3),(2,5),(1,6),(2,7)\},$       | $\{(2,1),(1,3),(2,5),(1,7)\}.$                   |

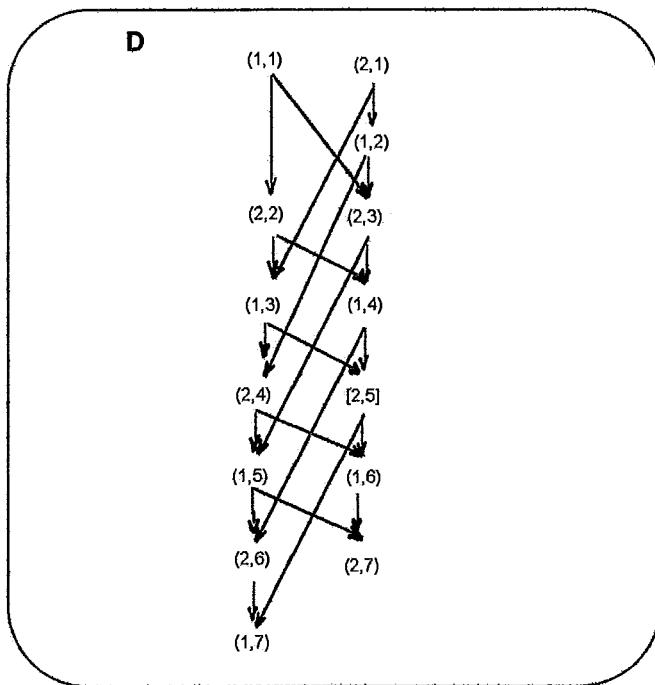
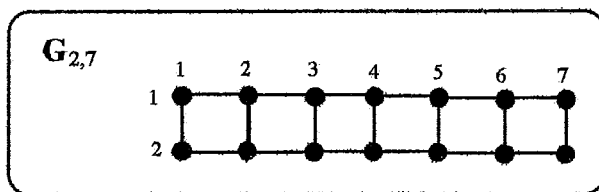


Fig. 4.16: Dígrafo associado a  $G_{2,7}$

**Lema 4.8:**

A família  $F(G_{2,p})$  de c.i.m. de  $G_{2,p}$  corresponde ao conjunto de caminhos maximais fonte-sumidouro do dígrafo acíclico  $D$  com conjunto de vértices  $V(D) = \{(1,1),(1,2),\dots,(1,p),(2,1),(2,2),\dots,(2,p)\}$ ; tal que existe o arco  $((1,v),(2,w))$  em  $E(D)$  se e somente se  $w = v + 1$  ou  $w = v + 2$ , e existe o arco  $((2,v),(1,w))$  em  $E(D)$  se e só se  $w = v + 1$  ou  $w = v + 2$ . As fontes de  $D$  são  $(1,1)$  e  $(1,2)$ , os sumidouros são  $(1, p)$  ou  $(2, p)$ .



**Algoritmo:**

Dado  $p$ , desenvolve-se um algoritmo para construir a família  $F$  de c.i.m de  $G_{2,p}$ , utilizando a caracterização dada pelo dígrafo  $D$  descrito no lema 4.8. O algoritmo 4.4 implementa este procedimento.

**Algoritmo 4.4: Família de c.i.m. de uma grade completa  $G_{2,p}$**

Dado  $p$

1. Construir o dígrafo  $D$  com  $V(D) = V(G_{2,p})$   
 $((1,v),(2,w)) \in E(D) \Leftrightarrow w = v + 1$  ou  $w = v + 2$   
 $((2,v),(1,w)) \in E(D) \Leftrightarrow w = v + 1$  ou  $w = v + 2$
2. Gerar todos os caminhos fonte-sumidouro de  $D$ .

**Complexidade:**

Passo 1:  $O(p)$

Passo 2:  $O(p)$  por caminho, ao fazer busca em profundidade irrestrita (ver observação 4.3).

A complexidade total do algoritmo 4.4 é  $O(p \cdot i(G_{2,p}))$ , possui tempo polinomial no tamanho do grafo e no número de c.i.m. do grafo. No entanto, o tempo por c.i.m. é  $O(p)$ . O espaço requerido é  $O(p)$ .

Por exemplo, na figura 4.17 apresenta-se o dígrafo associado a  $G_{2,8}$ .

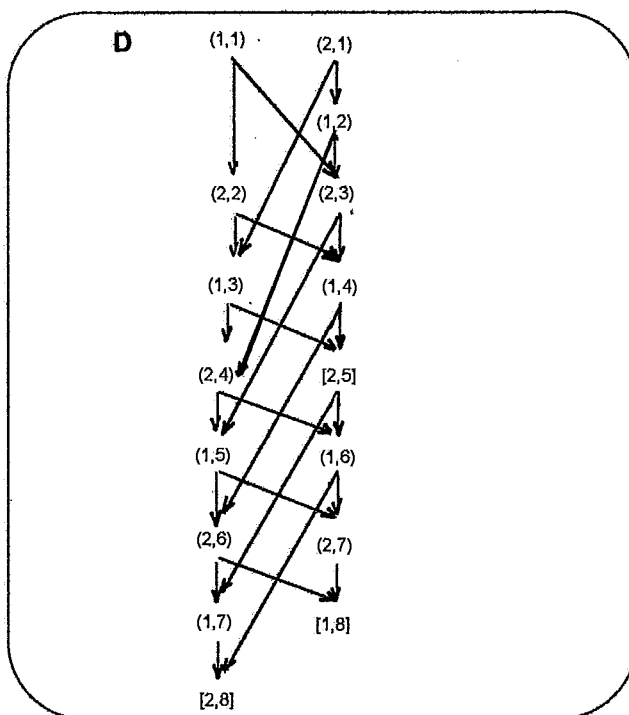


Fig. 4.17: Dígrafo associado a  $G_{2,8}$

Logo  $F(G_{2,8}) =$

$\{(1,1),(2,2),(1,3),(2,4),(1,5),(2,6),(1,7),(2,8)\}, \{(1,1),(2,2),(1,3),(2,4),(1,5),(2,6),(1,8)\},$   
 $\{(1,1),(2,2),(1,3),(2,4),(1,5),(2,7),(1,8)\}, \{(1,1),(2,2),(1,3),(2,4),(1,6),(2,7),(1,8)\},$   
 $\{(1,1),(2,2),(1,3),(2,4),(1,6),(2,8)\}, \{(1,1),(2,2),(1,3),(2,5),(1,6),(2,7),(1,8)\}$   
 $\{(1,1),(2,2),(1,3),(2,5),(1,6),(2,8)\}, \{(1,1),(2,2),(1,3),(2,5),(1,7),(2,8)\},$   
 $\{(1,1),(2,2),(1,4),(2,5),(1,7),(2,8)\}, \{(1,1),(2,2),(1,4),(2,5),(1,6),(2,7),(1,8)\}$   
 $\{(1,1),(2,2),(1,4),(2,5),(1,6),(2,8)\}, \{(1,1),(2,2),(1,4),(2,6),(1,8)\},$   
 $\{(1,1),(2,2),(1,4),(2,6),(1,7),(2,8)\}, \{(1,1),(2,3),(1,4),(2,5),(1,7),(2,8)\},$   
 $\{(1,1),(2,3),(1,4),(2,5),(1,6),(2,8)\}, \{(1,1),(2,3),(1,4),(2,5),(1,6),(2,7),(1,8)\},$   
 $\{(1,1),(2,3),(1,4),(2,6),(1,7),(2,8)\}, \{(1,1),(2,3),(1,4),(2,6),(1,8)\},$   
 $\{(1,1),(2,3),(1,5),(2,6),(1,8)\}, \{(1,1),(2,3),(1,5),(2,6),(1,7),(2,8)\},$   
 $\{(1,1),(2,3),(1,5),(2,7),(1,8)\},$   
 $\{(2,1),(1,2),(2,3),(1,4),(2,5),(1,6),(2,7),(1,8)\}, \{(2,1),(1,2),(2,3),(1,4),(2,5),(1,6),(2,8)\},$   
 $\{(2,1),(1,2),(2,3),(1,4),(2,5),(1,7),(2,8)\}, \{(2,1),(1,2),(2,3),(1,4),(2,6),(1,7),(2,8)\},$   
 $\{(2,1),(1,2),(2,3),(1,4),(2,6),(1,8)\}, \{(2,1),(1,2),(2,3),(1,5),(2,6),(1,8)\},$   
 $\{(2,1),(1,2),(2,3),(1,5),(2,6),(1,7),(2,8)\}, \{(2,1),(1,2),(2,3),(1,5),(2,7),(1,8)\},$   
 $\{(2,1),(1,2),(2,4),(1,5),(2,7),(1,8)\}, \{(2,1),(1,2),(2,4),(1,5),(2,6),(1,7),(2,8)\},$   
 $\{(2,1),(1,2),(2,4),(1,5),(2,6),(1,8)\}, \{(2,1),(1,2),(2,4),(1,6),(2,8)\},$   
 $\{(2,1),(1,2),(2,4),(1,6),(2,7),(1,8)\}, \{(2,1),(1,3),(2,4),(1,5),(2,6),(1,7),(2,8)\},$   
 $\{(2,1),(1,3),(2,4),(1,5),(2,6),(1,8)\}, \{(2,1),(1,3),(2,4),(1,5),(2,7),(2,8)\},$   
 $\{(2,1),(1,3),(2,4),(1,6),(2,7),(1,8)\}, \{(2,1),(1,3),(2,4),(1,6),(2,8)\},$   
 $\{(2,1),(1,3),(2,5),(1,6),(2,8)\}, \{(2,1),(1,3),(2,5),(1,6),(2,7),(1,8)\},$   
 $\{(2,1),(1,3),(2,5),(1,6),(2,7),(1,8)\}.$

### Número de c.i.m. de $G_{2,p}$

#### Lema 4.9:

O número de c.i.m. de uma grade completa  $G_{2,p}$  é dado pela relação:

$$i(G_{2,p}) = i(G_{2,p-1}) + i(G_{2,p-2}) \quad p \geq 3,$$

$$i(G_{2,1}) = 2,$$

$$i(G_{2,2}) = 2.$$

#### Observação 4.14:

O lema 4.9 fornece um algoritmo recursivo para construir a família  $F(G_{2,p})$  a partir de  $F(G_{2,p-1})$  e de  $F(G_{2,p-2})$ .

Seja  $I \in F(G_{2,p-1})$ :

$$(1,p-1) \in I \text{ então } I \cup \{(2,p)\} \in F(G_{2,p})$$

$$(2,p-1) \in I \text{ então } I \cup \{(1,p)\} \in F(G_{2,p})$$

Seja  $I \in F(G_{2,p-2})$ :

$$(1,p-2) \in I \text{ então } I \cup \{(2,p)\} \in F(G_{2,p})$$

$$(2,p-2) \in I \text{ então } I \cup \{(1,p)\} \in F(G_{2,p}).$$

Por exemplo, para obter  $F(G_{2,6})$  a partir de  $F(G_{2,4})$  e de  $F(G_{2,5})$ , tem-se:

$$F(G_{2,5}) = \{ \{(1,1),(2,2),(1,3),(2,4),(1,5)\}, \{(1,1),(2,2),(1,3),(2,5)\}, \\ \{(1,1),(2,2),(1,4),(2,5)\}, \{(1,1),(2,3),(1,4),(2,5)\}, \{(1,1),(2,3),(1,5)\}, \\ \{(2,1),(1,2),(2,3),(1,4),(2,5)\}, \{(2,1),(1,2),(2,3),(1,5)\}, \{(2,1),(1,2),(2,4),(1,5)\}, \\ \{(2,1),(1,3),(2,4),(1,5)\}, \{(2,1),(1,3),(2,5)\} \}$$

$$F(G_{2,4}) = \{ \{(1,1),(2,2),(1,3),(2,4)\}, \{(1,1),(2,2),(1,4)\}, \{(1,1),(2,3),(1,4)\}, \\ \{(2,1),(1,2),(2,3),(1,4)\}, \{(2,1),(1,2),(2,4)\}, \{(2,1),(1,3),(2,4)\} \}$$

$$\begin{aligned} \{(1,1),(2,2),(1,3),(2,4),(1,5)\} \in F(G_{2,5}) &\Rightarrow \{(1,1),(2,2),(1,3),(2,4),(1,5),(2,6)\} \in F(G_{2,6}) \\ \{(1,1),(2,2),(1,3),(2,5)\} &\in F(G_{2,5}) \Rightarrow \{(1,1),(2,2),(1,3),(2,5),(1,6)\} \in F(G_{2,6}) \\ \{(1,1),(2,2),(1,4),(2,5)\} &\in F(G_{2,5}) \Rightarrow \{(1,1),(2,2),(1,4),(2,5),(1,6)\} \in F(G_{2,6}) \\ \{(1,1),(2,3),(1,4),(2,5)\} &\in F(G_{2,5}) \Rightarrow \{(1,1),(2,3),(1,4),(2,5),(1,6)\} \in F(G_{2,6}) \\ \{(1,1),(2,3),(1,5)\} &\in F(G_{2,5}) \Rightarrow \{(1,1),(2,3),(1,5),(2,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,2),(2,3),(1,4),(2,5)\} \in F(G_{2,5}) &\Rightarrow \{(2,1),(1,2),(2,3),(1,4),(2,5),(1,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,2),(2,3),(1,5)\} &\in F(G_{2,5}) \Rightarrow \{(2,1),(1,2),(2,3),(1,5),(2,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,2),(2,4),(1,5)\} &\in F(G_{2,5}) \Rightarrow \{(2,1),(1,2),(2,4),(1,5),(2,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,3),(2,4),(1,5)\} &\in F(G_{2,5}) \Rightarrow \{(2,1),(1,3),(2,4),(1,5),(2,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,3),(2,5)\} &\in F(G_{2,5}) \Rightarrow \{(2,1),(1,3),(2,5),(1,6)\} \in F(G_{2,6}) \\ \{(1,1),(2,2),(1,3),(2,4)\} &\in F(G_{2,4}) \Rightarrow \{(1,1),(2,2),(1,3),(2,4),(1,6)\} \in F(G_{2,6}) \\ \{(1,1),(2,2),(1,4)\} &\in F(G_{2,4}) \Rightarrow \{(1,1),(2,2),(1,4),(2,6)\} \in F(G_{2,6}) \\ \{(1,1),(2,3),(1,4)\} &\in F(G_{2,4}) \Rightarrow \{(1,1),(2,3),(1,4),(2,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,2),(2,3),(1,4)\} &\in F(G_{2,4}) \Rightarrow \{(2,1),(1,2),(2,3),(1,4),(2,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,2),(2,4)\} &\in F(G_{2,4}) \Rightarrow \{(2,1),(1,2),(2,4),(1,6)\} \in F(G_{2,6}) \\ \{(2,1),(1,3),(2,4)\} &\in F(G_{2,4}) \Rightarrow \{(2,1),(1,3),(2,4),(1,6)\} \in F(G_{2,6}) \end{aligned}$$

**Observação 4.15:**

A equação de recorrências

$$i(G_{2,p}) = i(G_{2,p-1}) + i(G_{2,p-2})$$

que caracteriza o número de conjuntos independentes maximais de um grafo grade completa de  $2 \times p$ , é do tipo da equação de recorrências que define os números de Fibonacci mas as condições de contorno não são as mesmas. Em este caso, tem-se:

$$\begin{aligned} i(G_{2,1}) &= 2 \\ i(G_{2,2}) &= 2. \end{aligned}$$

A equação de recorrências

$$i(\mathbf{G}_{2,p}) = i(\mathbf{G}_{2,p-1}) + i(\mathbf{G}_{2,p-2})$$

é equivalente à equação  $1 + x = x^2$   
 cujas soluções são:  $x_1 = (1 + \sqrt{5})/2$   
 $x_2 = (1 - \sqrt{5})/2$ .

Logo, tem-se o seguinte resultado.

**Lema 4.10:**

O número de c.i.m. de uma grade completa  $\mathbf{G}_{2,p}$  é dado por:

$$i(\mathbf{G}_{2,p}) = x_1^p + x_2^p \quad p \geq 3$$

isto é:

$$i(\mathbf{G}_{2,p}) = (2/\sqrt{5})[(1 + \sqrt{5})/2]^p - (2/\sqrt{5})[(1 - \sqrt{5})/2]^p \quad p \geq 3$$

$$i(\mathbf{G}_{2,1}) = 2$$

$$i(\mathbf{G}_{2,2}) = 2.$$

Tem-se:

$$i(\mathbf{G}_{2,p}) = 2 F_p \quad p \geq 3$$

com  $F_p$  o número de Fibonacci de ordem  $p$ .

Considere por exemplo  $\mathbf{G}_{2,7}$ :

pelo lema 4.9:

$$i(\mathbf{G}_{2,7}) = i(\mathbf{G}_{2,6}) + i(\mathbf{G}_{2,5})$$

pelo lema 4.10:

$$i(\mathbf{G}_{2,7}) = (2/\sqrt{5})[(1 + \sqrt{5})/2]^7 - (2/\sqrt{5})[(1 - \sqrt{5})/2]^7 = 2 F_7$$

logo

$$i(\mathbf{G}_{2,7}) = 26.$$

**Observação 4.16:**

Considerando as relações que satisfaz  $i(\mathbf{G}_{2,p})$ , então o número de c.i.m. de um grafo grade completo  $\mathbf{G}_{2,p}$  é limitado por:

$$i(\mathbf{G}_{2,p}) \leq (1 + \sqrt{5})^p \quad p \geq 3.$$

### 4.4. Grafos de intervalo

Um grafo  $G(V,E)$  é de intervalo se existe uma correspondência 1-1 entre o seu conjunto de vértices  $V$  e uma família de intervalos reais  $\{I_v\}_{v \in V}$ , de forma que, para todo par de vértices distintos  $u, v$  tem-se:  $(u,v) \in E$  se e somente se  $I_u \cap I_v \neq \emptyset$ ; isto é, os vértices estão unidos se os intervalos correspondentes se interceptam. A família  $\{I_v\}_{v \in V}$  é chamada representação de  $G$  por intervalos.

Na figura 4.18 exibe-se o caminho  $P_5$  e sua representação por intervalos. Em geral todo caminho  $P_n$  é um grafo de intervalo, não acontece o mesmo com os ciclos  $C_n$  pois não é possível encontrar uma representação de modo que  $I_j \cap I_{j+1} \neq \emptyset, \forall j = 1, \dots, n-1$ , mas  $I_n \cap I_1 = \emptyset$ , se  $n > 3$ .

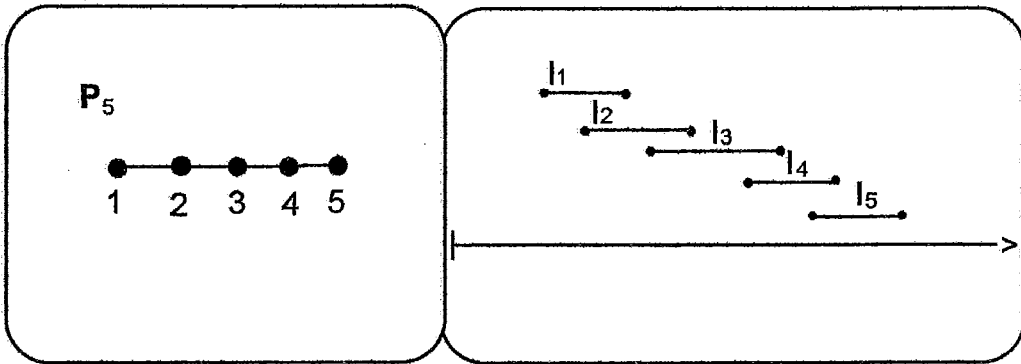


Fig. 4.18:  $P_5$  e  $\{I_j\}_{j=1,5}$

### Algoritmo de Gupta, Lee e Leung

Gupta, Lee e Leung (1982) desenvolvem um algoritmo para determinar um conjunto independente máximo em tempo  $O(n \log n)$ , para um grafo dado através da representação por intervalos.

Cada intervalo  $I_j$  é representado pelo seu limite esquerdo  $e_j$  e pelo seu limite direito  $d_j$ ,  $I_j = [e_j, d_j]$ . Supõe-se que estes limites são todos diferentes.

A idéia do algoritmo é construir um conjunto independente máximo através da seqüência de intervalos  $J = I_{j_1}, I_{j_2}, \dots$ ; onde  $I_{j_1}$  corresponde ao intervalo com menor limite direito,  $I_{j_i}$  ao primeiro intervalo posterior a  $I_{j_{i-1}}$  na reta real e que não o intercepta.

O algoritmo 4.5 a seguir implementa este método.

**Algoritmo 4.5: Conjunto Independente Máximo**

Dado  $G$  grafo de intervalo por  $\{I_j\}_{j=1,n}$ ,  $I_j = [e_j, d_j]$ ,  $j = 1, \dots, n$ ,  $|V| = n$

1. Ordenar os limites dos intervalos em forma crescente na seqüência  $L$
2.  $J = \emptyset$
3. Enquanto  $L \neq \emptyset$  efetuar
  - determinar  $d_k = \min \{d_j / 1 \leq j \leq n\}$
  - $J = J \cup \{k\}$
  - retirar de  $L$ :  $e_k$ ,  $d_k$  e todos os limites correspondentes a intervalos que contém  $d_k$   
( $I_j$  tais que  $e_j < d_k < d_j$ )
4. Devolver  $J$ .

**Complexidade:**

A complexidade é dada pela ordenação dos limites dos intervalos, isto é  $O(n \log n)$ .

Gupta, Lee e Leung provam que o limite inferior para determinar um conjunto independente máximo é  $\Omega(n \log n)$  devido à ordenação dos intervalos, logo, esse é ótimo.

**Observação 4.17:**

Dada a família de intervalos na reta real, que define o grafo de intervalo, considera-se o conjunto independente maximal que contém o vértice correspondente ao intervalo mais à esquerda.

Na figura 4.19 apresenta-se o grafo  $G_1$  e sua representação pela família de intervalos  $\{I_j\}_{j=1,7}$ . Aplicando o algoritmo de Gupta, Lee e Leung para gerar um conjunto independente máximo, tem-se  $L = e_1, e_2, e_3, d_1, e_4, d_2, e_5, d_3, e_6, d_4, e_7, d_5, d_6, d_7$ ; com isto gera-se o conjunto  $\{1,4,7\}$ .

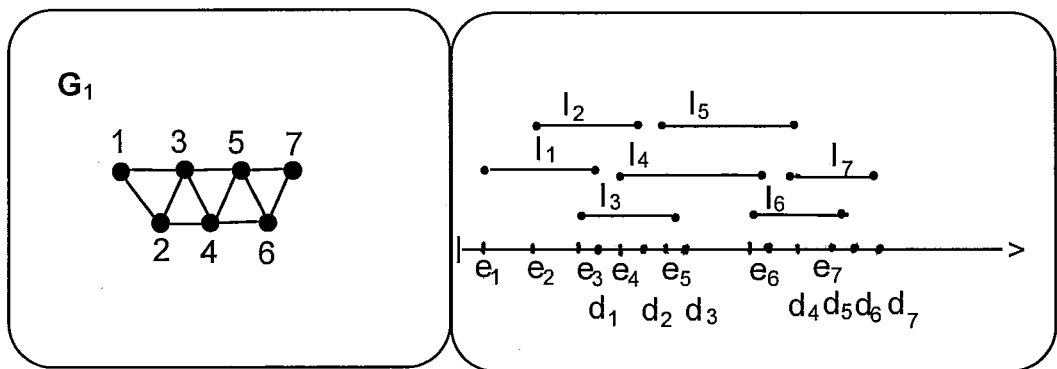


Fig. 4.19: Exemplo 1 do algoritmo de Gupta, Lee e Leung

Para o grafo  $G_2$  e sua família de intervalos  $\{I_j\}_{j=1,6}$  da figura 4.20, aplicando o algoritmo de Gupta, Lee e Leung tem-se  $L = e_1, e_2, d_1, e_4, d_2, e_3, d_3, e_5, d_4, e_6, d_5, d_6$ ; com isto gera-se o conjunto independente máximo  $\{1,3,5\}$ .

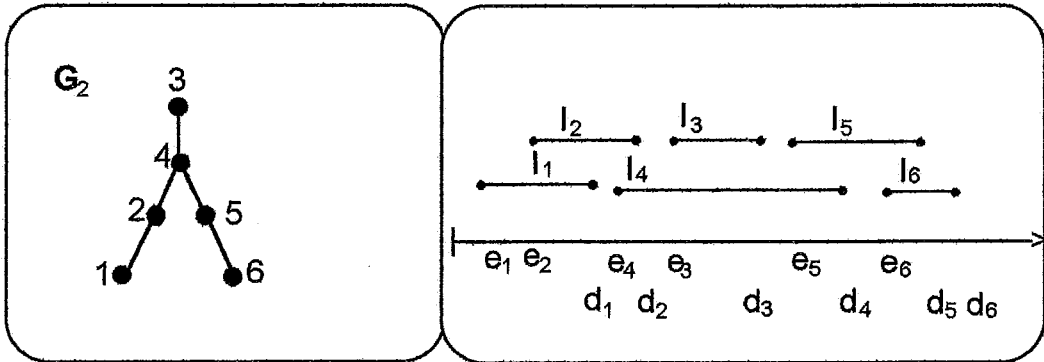


Fig. 4.20: Exemplo 2 do algoritmo de Gupta, Lee e Leung

### Algoritmo para gerar a família $F(G)$ de c.i.m de $G$

#### Algoritmo de Leung

Leung (1984) considera  $G$  dado por uma representação por intervalos  $\{I_j\}_{j=1,n}$  onde  $I_j = [e_j, d_j], j = 1, \dots, n$ . Define:

$$NEXT(I_j) = \begin{cases} k & \text{tal que } d_j < e_k, d_k \leq d_i \quad \forall I_i \text{ tal que } d_j < e_i \\ \lambda & \text{se } I_k \text{ não existe} \end{cases}$$

$$NEXTGROUP(I_j) = \begin{cases} \{i \mid d_j < e_i, d_{NEXT(I_j)} \in I_i\} \\ \emptyset & \text{se } NEXT(I_j) = \lambda \end{cases}$$

$NEXT(I_j)$  corresponde ao primeiro intervalo posterior a  $I_j$  na reta real que não intercepta a  $I_j$ .

$NEXTGROUP(I_j)$  corresponde ao conjunto de intervalos, incluindo  $NEXT(I_j)$ , posteriores a  $I_j$  na reta real, que não interceptam a  $I_j$  mas interceptam  $NEXT(I_j)$ .

Com estas definições o algoritmo de Gupta, Lee e Leung para gerar um conjunto independente máximo, pode ser descrito da seguinte forma:

**Algoritmo 4.6: Conjunto Independente Máximo**

Dado  $G$  grafo de intervalo por  $\{I_j\}_{j=1,n}$ ,  $I_j = [e_j, d_j]$ ,  $j = 1, \dots, n$ ,  $|V| = n$

1. Ordenar os limites dos intervalos em forma crescente
2. Para todo  $j = 1$  até  $n$  determinar  $NEXT(I_j)$
3. Determinar  $I_t$  tal que  $d_t = \min \{d_j / 1 \leq j \leq n\}$
4.  $J = \{t\}$
5. Enquanto  $NEXT(I_t) \neq \lambda$  efetuar  
 $J = J \cup \{NEXT(I_t)\}$   
 $t = NEXT(I_t)$
6. Devolver  $J$ .

Para o grafo  $G_1$  da figura 4.19 tem-se:

j	NEXT(I <sub>j</sub> )
1	I <sub>4</sub>
2	I <sub>5</sub>
3	I <sub>6</sub>
4	I <sub>7</sub>
5	λ
6	λ
7	λ

logo, gera-se o conjunto independente máximo  $J = \{1,4,7\}$ .

Leung gera todos os conjuntos independentes maximais de  $G$  grafo de intervalo. Ele determina, primeiro, um conjunto independente máximo  $J$  com o algoritmo de Gupta, Lee e Leung e efetua backtracking desde o último elemento  $s$  de  $J$  para atrás, examinando todas as possibilidades. Para isto, ele utiliza uma lista para cada posição da seqüência  $J$ .

Seja  $J$  uma seqüência de vértices que pertencem a um conjunto independente maximal de  $G$ .

Seja  $I_t$  o intervalo tal que  $d_t = \min \{d_j / 1 \leq j \leq n\}$ .

O primeiro elemento de  $J$  é  $t$  ou algum outro índice  $r$  tal que  $I_r$  intercepta  $I_t$ .

Se  $j$  é o último elemento da seqüência  $J$  construída até o momento, então não se pode acrescentar nenhum intervalo que intercepte a  $I_j$ ; pode acrescentar-se um intervalo  $I_k$  que não o intercepte, isto é tal que  $d_j < e_k$ . Para garantir a maximalidade do  $J$  obtido ao final do processo,  $k$  deve pertencer a  $NEXTGROUP(I_j)$ .



Este método é implementado no algoritmo 4.7, cada vez que um c.i.m. é gerado ele é listado e realiza-se o backtracking.

**Algoritmo 4.7: Família de Conjuntos Independentes Maximais**

Dado  $G$  grafo de intervalo por  $\{I_j\}_{j=1,n}$ ,  $I_j = [e_j, d_j]$ ,  $j = 1, \dots, n$ ,  $|V| = n$

1. Ordenar os limites dos intervalos em forma crescente
2. Para todo  $j = 1$  até  $n$  efetuar
  - determinar  $NEXT(I_j)$
  - determinar  $NEXTGROUP(I_j)$
3. Determinar  $I_t$  tal que  $d_t = \min \{d_j / 1 \leq j \leq n\}$
4.  $L_1 = \{t\} \cup \{j / e_j < d_t < d_j\}$
5.  $I' = I_t$   
 $k = 1$
6. Enquanto  $NEXTGROUP(I') \neq \emptyset$  efetuar
  - $k = k + 1$
  - $L_k = NEXTGROUP(I')$
  - $I' = NEXT(I')$
7. Listar  $|L_k|$  c.i.m. escolhendo o primeiro elemento das  $(k-1)$  primeiras listas e um elemento de  $L_k$
8.  $k = k - 1$
9. Enquanto  $|L_k| = 1$  e  $k \geq 1$  efetuar  
 $k = k - 1$
10. Se  $k = 0$  então PARAR
  - caso contrário efetuar
    - retirar o primeiro elemento de  $L_k$
    - $I' =$  primeiro elemento de  $L_k$
    - se  $NEXTGROUP(I') \neq \emptyset$  então ir a 6
    - caso contrário efetuar
      - listar um c.i.m. considerando o primeiro elemento de  $L_1, \dots, L_k$
      - ir a 9.

**Complexidade:**

A complexidade total do algoritmo 4.7 é  $O(n^2 + \beta)$ , onde  $\beta$  é a soma do número de vértices de todos os c.i.m..

Para o grafo  $G_1$  da figura 4.19 tem-se:

j	NEXT(I <sub>j</sub> )	NEXTGROUP(I <sub>j</sub> )
1	4	4,5,6
2	5	5,6,7
3	6	6,7
4	7	7
5	λ	∅
6	λ	∅
7	λ	∅

ITERAÇÃO (passo 6)	LISTAS			c.i.m.
	L <sub>1</sub>	L <sub>2</sub>	L <sub>3</sub>	
1	1	4	7	1,4,7
	2	5		1,5
	3	6		1,6
2	2	5		2,5
	3	6		2,6
		7		2,7
3	3	6		3,6
		7		3,7

Para o grafo  $G_2$  da figura 4.20 tem-se:

j	NEXT(I <sub>j</sub> )	NEXTGROUP(I <sub>j</sub> )
1	3	3,4
2	3	3
3	5	5,6
4	6	6
5	$\lambda$	$\emptyset$
6	$\lambda$	$\emptyset$

ITERAÇÃO	LISTAS			c.i.m.
	L <sub>1</sub>	L <sub>2</sub>	L <sub>3</sub>	
1	1	3	5	1,3,5
	2	4	6	1,3,6
2	1	4	6	1,4,6
	2			
3	2	3	5	2,3,5
			6	2,3,6

### Algoritmo proposto

Seja:  $G$  um grafo de intervalo,

$F(G)$  a família de conjuntos independentes maximais de  $G$ ,

$i(G) = \text{card}(F(G)) =$  número de c.i.m de  $G$ .

Para o grafo  $G_1$  da figura 4.19 tem-se  $F(G_1) = \{ \{ 1,4,7 \}, \{ 1,5 \}, \{ 1,6 \}, \{ 2,5 \}, \{ 2,6 \}, \{ 2,7 \}, \{ 3,6 \}, \{ 3,7 \} \}$ ,  $i(G_1) = 8$ .

**Propriedade 4.1 (Gilmore e Hoffman, 1964):**

$G$  é um grafo de intervalo se e somente se suas cliques maximais podem ser ordenadas de forma que, para todo vértice as cliques maximais que o contêm são consecutivas na ordem. Esta ordem linear das cliques maximais se denomina ordem canônica.

Para o caminho  $P_5$  tem-se a ordem canônica com as cliques maximais definidas por  $C_1 = \{1,2\}$ ,  $C_2 = \{2,3\}$ ,  $C_3 = \{3,4\}$  e  $C_4 = \{4,5\}$ . No caso do grafo  $G_1$  da figura 4.19, a ordem canônica das cliques maximais é dada pelas cliques  $C_1 = \{1,2,3\}$ ,  $C_2 = \{2,3,4\}$ ,  $C_3 = \{3,4,5\}$ ,  $C_4 = \{4,5,6\}$  e  $C_5 = \{5,6,7\}$ . Para o grafo  $G_2$  da figura 4.20,  $C_1 = \{1,2\}$ ,  $C_2 = \{2,4\}$ ,  $C_3 = \{3,4\}$ ,  $C_4 = \{4,5\}$  e  $C_5 = \{5,6\}$ . Os grafos podem desenhar-se como exibidos na figura 4.21, cada aresta  $(v_i, v_j)$  tal que  $(v_k, v_{k+1}) \in E$  para todo  $k = i, \dots, j-1$ , corresponde a uma aresta maximal que representa à clique maximal  $\{v_i, v_{i+1}, \dots, v_j\}$ . Tem-se  $F(G_2) = \{\{1,4,6\}, \{1,3,5\}, \{1,3,6\}, \{2,3,5\}, \{2,3,6\}\}$ ,  $i(G_2) = 5$ .

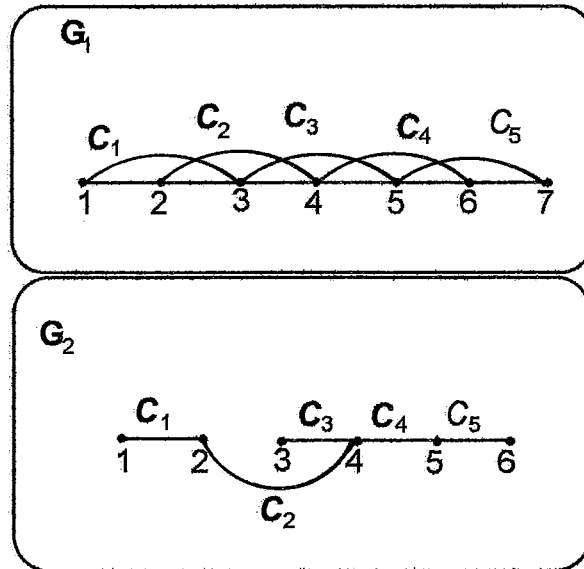


Fig. 4.21: Cliques maximais de  $G_1$  e  $G_2$

Considerando a ordem canônica  $C_1, C_2, \dots, C_t$  das cliques maximais de  $G$ , cada vértice  $v$  é rotulado com o par  $r_v = (a_v, b_v)$ , onde:

$a_v =$  menor índice  $i$  tal que  $C_i$  contem  $v$

$b_v =$  maior índice  $i$  tal que  $C_i$  contem  $v$ .

Pela propriedade 4.1 tem-se  $v \notin C_j, \forall j < a_v$  e  $v \notin C_j, \forall j > b_v$ .

**Observação 4.18:**

Considerando um conjunto independente maximal  $I$  como a seqüência de vértices  $I = v_1, v_2, \dots, v_k$ , com  $k \leq n$ , observa-se:

- $v_1$  pertence a  $C_1$ , pela maximalidade de  $I$ ,
- se  $I$  foi construído até o seu  $s$ -ésimo elemento com  $r_s = (a_s, b_s)$ , então o próximo elemento da seqüência será um vértice  $w$  com  $r_w = (a_w, b_w)$  tal que  $b_s < a_w$ .

Dado que, para cada  $v_s$  da seqüência  $I$  existem diferentes possibilidades para  $v_{s+1}$ , estas possibilidades correspondem num grafo direcionado  $D$  ao conjunto de vértices  $V(D)$  igual ao de  $G$  e tal que existe a aresta  $(v, w)$  em  $E(D)$  se e somente se  $b_v < a_w$ . Desta forma, cada vértice de  $C_1$  é uma fonte e cada vértice de  $C_t$  é um sumidouro, onde  $t$  é o número de cliques maximais de  $G$ . Cada caminho maximal fonte-sumidouro em  $D$  corresponde a um conjunto independente maximal de  $G$ .

**Lema 4.11:**

A família  $F(G)$  de conjuntos independentes maximais de  $G$  grafo de intervalo corresponde ao conjunto de caminhos maximais fonte-sumidouro do dígrafo acíclico  $D$  com conjunto de vértices  $V(D) = \{v_1, v_2, \dots, v_n\}$  e tal que existe a aresta  $(v, w)$  em  $E(D)$  se e somente se  $b_v < a_w$ .

**Prova:**

Dado o dígrafo  $D$ , construído segundo o lema 4.11, tem-se que cada caminho maximal  $I = v_1, v_2, \dots, v_k$  em  $D$  desde uma fonte  $v_1$  até um sumidouro  $v_k$  corresponde a um c.i.m de  $G$  pois, em  $D$  existe o arco  $(v, w)$  se e somente se  $(v, w) \in E(G)$ , isto é sse  $b_v < a_w$ , logo é conjunto independente e  $I$  é maximal pois não é possível acrescentar mais vértices a  $I$ , dado que: a fonte  $v_1 \in C_1$ , o sumidouro  $v_k \in C_t$ , se  $v \in I$  então algum  $w \in C_s$  com  $s > b_v$  deve pertencer a  $I$ . Os caminhos tem que ser fonte-sumidouro e tem que ser maximais (s igual ao menor  $a_w$  tal que  $b_v < a_w$ ) não incluindo arestas induzidas por transitividade, caso contrário o c.i.m. não é maximal.

Dado um c.i.m.  $I = \{v_1, v_2, \dots, v_k\}$  de  $G$  com  $v_1 < v_2 < \dots < v_k$ , a seqüência  $I$  dada por  $v_1, v_2, \dots, v_k$  corresponde a um caminho fonte-sumidouro em  $D$ , pois como  $I$  é maximal então  $v_1 \in C_1$ ,  $v_k \in C_t$  e se  $v_p \in I$  então  $v_{p+1}$  é igual a algum  $w \in C_s$  com  $s$  igual ao menor  $a_w$  tal que  $b_v < a_w$ .



Por exemplo, para o grafo  $G_1$  da figura 4.19 tem-se:

$v$	$a_v$	$b_v$
1	1	1
2	1	2
3	1	3
4	2	4
5	3	5
6	4	5
7	5	5

Na figura 4.22 tem-se o digrafo  $D_1$  associado a  $G_1$  e a redução transitiva  $D_{R_1}$  de  $D_1$ , a família de c.i.m. de  $G_1$  é  $F(G_1) = \{\{1,4,7\}, \{1,5\}, \{1,6\}, \{2,5\}, \{2,7\}, \{2,6\}, \{3,6\}, \{3,7\}\}$ .

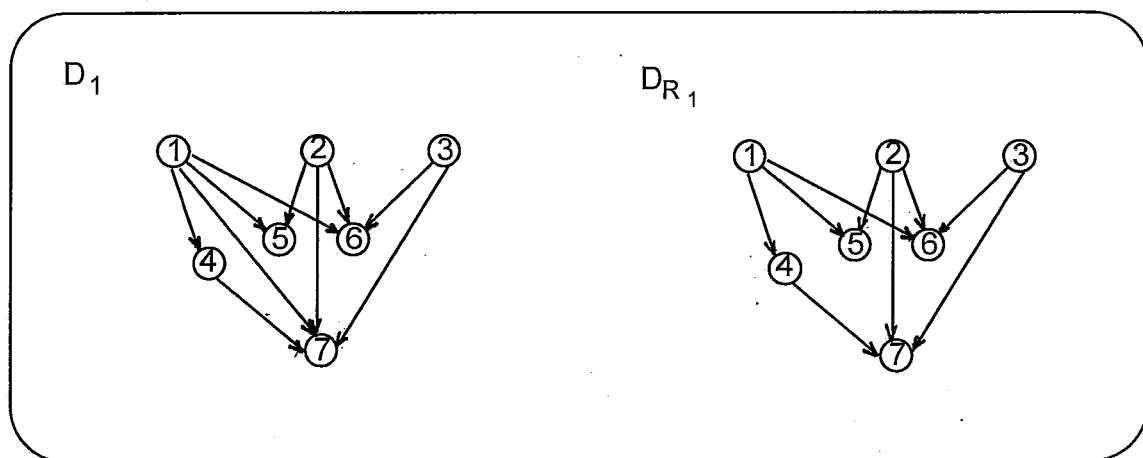


Fig. 4.22: Digrafo e redução transitiva associados a  $G_1$

**Algoritmo:**

Dado  $G$ , desenvolve-se um algoritmo para construir a família  $F$  de c.i.m de  $G$ , utilizando a caracterização dada pelo digrafo  $D$  descrito no lema 4.11, para garantir que os caminhos sejam maximais trabalha-se na redução transitiva de  $D$ . O algoritmo 4.8 implementa este procedimento.

**Algoritmo 4.8: Família de c.i.m. de um grafo de intervalo**

Dado  $G$  grafo de intervalo

1. Determinar a ordem canônica das cliques maximais  $C_1, C_2, \dots, C_t$
2. Para todo  $v \in V$  rotular  $v$  com  $r_v = (a_v, b_v)$  onde:
  - $a_v =$  menor índice  $i$  tal que  $v \in C_i$
  - $b_v =$  maior índice  $i$  tal que  $v \in C_i$
3. Construir o digrafo  $D$  com  $V(D) = V(G)$  e  $(v,w) \in E(D) \Leftrightarrow b_v < a_w$
4. Construir a redução transitiva  $D_R$  de  $D$
5. Gerar todos os caminhos fonte-sumidouro de  $D_R$ .

**Complexidade:**

Passo 1:  $O(n+m)$  utilizando o algoritmo de reconhecimento de Booth e Leuker( 1976)

Passo 2:  $O(n+m)$  usando o algoritmo de Olariu (1991) para determinar uma ordem linear dos vértices.

Passo 3:  $O(n^2)$

Para o dígrafo  $D$  tem-se  $V(D) = V(G)$  e  $(v,w) \in E(D) \Leftrightarrow b_v < a_w$ , isto é:  $v$  e  $w$  podem pertencer ao mesmo c.i.m. se  $I_v \cap I_w = \emptyset$  mas  $I_v \cap I_w = \emptyset \Leftrightarrow b_v < a_w$ . Basta comparar os rótulos de todos os pares de vértices.

Passo 4:  $O(nm)$

Redução transitiva de um dígrafo acíclico

Passo 5:  $O(n)$  por caminho, ao fazer busca em profundidade irrestrita (observação 4.3).

A complexidade total do algoritmo 4.8 é  $O(n(m+ i(G)))$ . No entanto, o tempo por c.i.m. é  $O(nm)$ , dado que para gerar um c.i.m. é preciso os passos 1 a 4 com tempo  $O(nm)$  e apenas uma vez o passo 5. O espaço requerido é  $O(n + m)$ .

Para o caminho induzido  $P_n$  com  $V(P_n) = \{1,2,\dots,n\}$ , a ordem canónica é dada pelas cliques maximais  $C_i = \{ i, i+1\}$  para  $i = 1, \dots, n-1$ . O rótulo do vértice  $i$  é  $r_i = (i-1, i)$ . Na figura 4.23 apresenta-se a redução transitiva  $D_R$  do dígrafo  $D$  associado ao caminho  $P_{10}$ . Em geral, para  $P_n$  as fontes são 1 e 2, os sumidouros são  $n-1$  e  $n$ , cada vértice  $i$  está ligado ao  $(i+2)$  e  $(i+3)$ . A família de c.i.m. de  $P_{10}$  é  $\{\{1,3,5,7,9\}, \{1,3,5,7,10\}, \{1,3,5,8,10\}, \{1,3,6,9\}, \{1,3,6,8,10\}, \{1,4,7,9\}, \{1,4,7,10\}, \{1,4,6,9\}, \{1,4,6,8,10\}, \{2,4,6,8,10\}, \{2,4,6,9\}, \{2,4,7,9\}, \{2,4,7,10\}, \{2,5,7,9\}, \{2,5,8,10\}, \{2,5,8,10\}\}$  com  $i(P_{10}) = 16$ . A redução transitiva  $D_R$  do dígrafo  $D$  coincide com o dígrafo  $D$  construído segundo o lema 4.2.

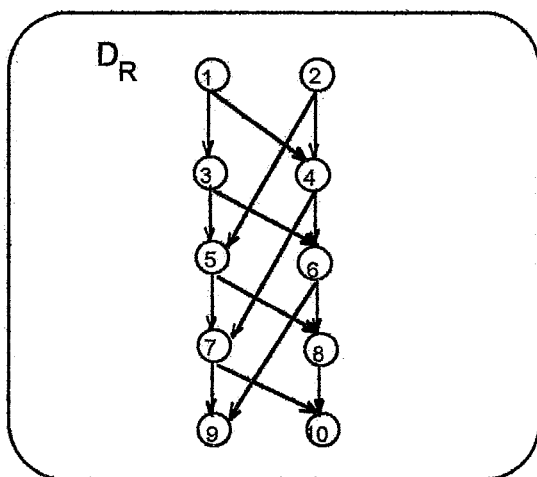


Fig. 4.23: Redução transitiva de  $P_{10}$

Para o grafo  $G_2$  da figura 4.20 tem-se:

$v$	$r_v$
1	(1,1)
2	(1,2)
3	(3,3)
4	(2,4)
5	(4,5)
6	(5,5)

Na figura 4.24 apresenta-se o dígrafo  $D_2$  associado a  $G_2$  e a redução transitiva  $D_{R_2}$  de  $D_2$ , a família de c.i.m. de  $G_2$  é  $F(G_2) = \{\{1,4,6\}, \{1,3,5\}, \{1,3,6\}, \{2,3,5\}, \{2,3,6\}\}$ .

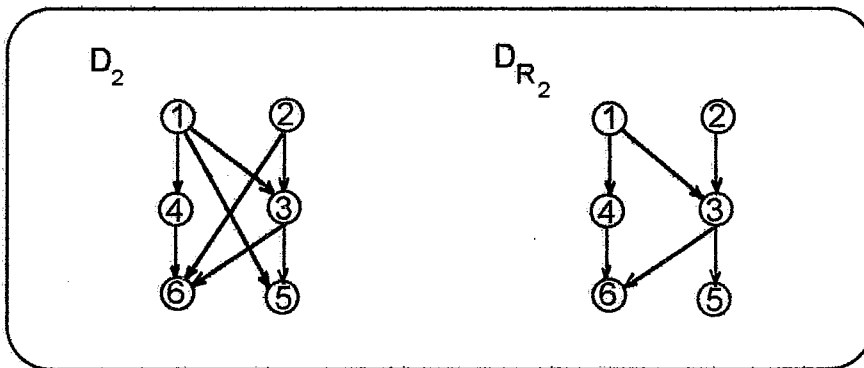


Fig. 4.24: Dígrafo e redução transitiva associados a  $G_2$

**Observação 4.19:**

O grafo subjacente ao dígrafo  $D$  construído é  $\bar{G}$ . O algoritmo dado é um caso particular do algoritmo de Szwarcfiter e Barroso (1988) para determinar as cliques maximais de um grafo dada uma orientação localmente transitiva de suas arestas. Este método pode ser descrito da seguinte forma geral:

**Algoritmo 4.9: Cliques Maximais de um Grafo de Comparabilidade Local**

Dado  $G$  e uma orientação localmente transitiva de  $G$

1. Determinar a redução transitiva  $G_R$
2. Encontrar todas as arestas maximais de  $\bar{G}$
3. Para cada aresta maximal  $(v,w) \in E(\bar{G})$   
 encontrar todos os caminhos  $v-w$  em  $G_R$ .

**Complexidade:**

A complexidade total do algoritmo 4.9 é  $O(n(m + t))$  onde  $t$  é o número de cliques maximais de  $G$ .

## Número de c.i.m. de G

Utilizando o mesmo processo anterior do algoritmo 4.8, é possível contar o número de c.i.m. de um grafo de intervalo, atribuindo um rótulo  $l(v)$  a cada vértice  $v$  de  $\mathbf{D}_R$ , rótulo que depende do rótulo dos seus descendentes. Primeiro, faz-se a atribuição aos sumidouros e logo, considerando  $S(v)$  o conjunto de sucessores de  $v$ , sobe-se no dígrafo atribuindo o seguinte rótulo:

$$l(v) = \begin{cases} 1 & \text{se } v \text{ é sumidouro} \\ \sum_{w \in S(v)} l(w) & \text{se não} \end{cases}$$

com isto

$$i(G) = \sum_{v \text{ fonte}} l(v).$$

Seja:

$$\text{FONTE} = \{ s \in \mathbf{V}(\mathbf{D}_R) / s \text{ é fonte de } \mathbf{D}_R \},$$

$$\text{FOLHA} = \{ f \in \mathbf{V}(\mathbf{D}_R) / f \text{ é sumidouro de } \mathbf{D}_R \},$$

$\text{dist}(v,s)$  a distância entre os vértices  $v$  e  $s$  em  $\mathbf{D}_R$ , que é igual ao comprimento do menor caminho dirigido entre  $v$  e  $s$  no grafo  $\mathbf{D}_R$ . Se não existe caminho dirigido de  $v$  a  $s$  então  $\text{dist}(v,s) = 0$ .

O procedimento para contar o número de c.i.m. de um grafo de intervalo pode ser implementado segundo o algoritmo 4.10 apresentado abaixo:



**Algoritmo 4.10: Contagem de c.i.m. de um grafo de intervalo**

Dado  $G$  grafo de intervalo

1. Determinar a ordem canônica das cliques maximais  $C_1, C_2, \dots, C_t$
2. Para todo  $v \in V$  rotular  $v$  com  $r_v = (a_v, b_v)$  onde:
  - $a_v =$  menor índice  $i$  tal que  $v \in C_i$
  - $b_v =$  maior índice  $i$  tal que  $v \in C_i$
3. Construir o dígrafo  $D$  com  $V(D) = V(G)$  e  $(v, w) \in E(D) \Leftrightarrow b_v < a_w$
4. Construir a redução transitiva  $D_R$  de  $D$
5. Para todo  $v \in V(D_R)$ 
  - para todo  $s \in \text{FONTE}$ 
    - se  $v$  é fonte então  $\text{dist}(v, s) = 0$
    - caso contrário calcular  $\text{dist}(v, s)$
  - $d(v) = \max \{ \text{dist}(v, s) / s \in \text{FONTE} \}$
  - $h = \max \{ d(v) / v \in V(D_R) \}$
  - para todo  $v \in \text{FOLHA}$ 
    - $l(v) = 1$
  - enquanto  $h \geq 0$  efetuar
    - $h = h - 1$
    - para todo  $v$  tal que  $d(v) = h$ 
      - se  $v$  não é sumidouro então
        - $l(v) = 0$
        - para todo  $w \in S(v)$ 
          - $l(v) = l(v) + l(w)$
- $i(G) = 0$ 
  - para todo  $v \in \text{FONTE}$ 
    - $i(G) = i(G) + l(v)$ .

**Complexidade:**

Os passos 1 a 4 são os mesmos do algoritmo 4.8 para enumerar todos os c.i.m. de  $G$ , com complexidade  $O(nm)$ .

Passo 5:  $O(nm)$

A complexidade total do algoritmo 4.10 é  $O(nm)$ .

Aplicando os algoritmos 4.8 e 4.10 ao exemplo apresentado por Leung, grafo da figura 4.25, tem-se:

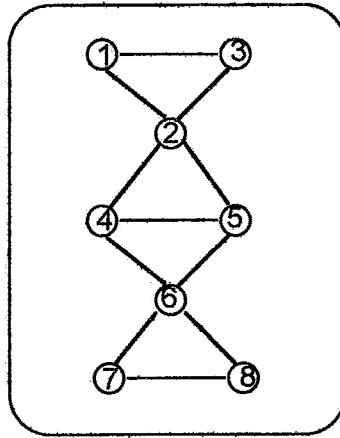


Fig. 4.25: Exemplo de Leung

A ordem canónica das cliques maximais é  $C_1 = \{1,2,3\}$ ,  $C_2 = \{2,4,5\}$ ,  $C_3 = \{4,5,6\}$ ,  $C_4 = \{6,7,8\}$ , os rótulos dos vértices apresentam-se na tabela seguinte:

v	$r_v$
1	(1,1)
2	(1,2)
3	(1,1)
4	(2,3)
5	(2,3)
6	(3,4)
7	(4,4)
8	(4,4)

Na figura 4.26 apresenta-se o dígrafo  $D$  associado ao grafo  $G$  da figura 4.25 e a redução transitiva  $D_R$  de  $D$ ; o conjunto de fontes é FONTE =  $\{1,2,3\}$ , o conjunto de sumidouros é FOLHA =  $\{6,7,8\}$ .

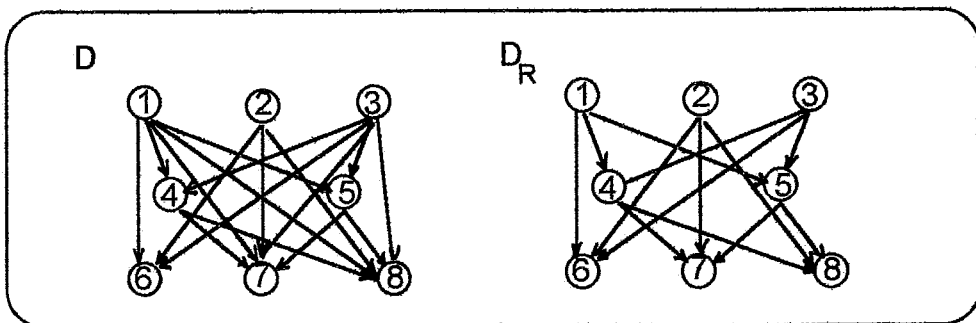


Fig. 4.26: Dígrafo e redução transitiva associados ao grafo de Leung

A família de c.i.m. de  $G$  é  $\{\{1,6\}, \{1,4,7\}, \{1,4,8\}, \{1,5,7\}, \{1,5,8\}, \{2,6\}, \{2,7\}, \{2,8\}, \{3,6\}, \{3,4,7\}, \{3,4,8\}, \{3,5,7\}, \{3,5,8\}\}$ . As distâncias são:

v	dist(v,1)	dist(v,2)	dist(v,3)
1	0	0	0
2	0	0	0
3	0	0	0
4	1	0	1
5	1	0	1
6	1	1	1
7	2	1	2
8	2	1	2

A distância máxima para cada vértice é:

v	d(v)
1	0
2	0
3	0
4	1
5	1
6	1
7	2
8	2

com isto:

v	S(v)
1	4,5,6
2	6,7,8
3	4,5,6
4	7,8
5	7,8
6	∅
7	∅
8	∅

logo  $h = 2$ , os rótulos dos vértices são:

v	l(v)
1	5
2	3
3	5
4	2
5	2
6	1
7	1
8	1

e portanto  $i(G) = 13$ .

**Observação 4.20:**

Leung (1984) desenvolve um algoritmo para gerar todos os conjuntos independentes maximais de  $G$  grafo arco-circular, em tempo  $O(n^2 + \beta)$  onde  $\beta$  é a soma do número de vértices de todos os c.i.m.. Para isto, ele identifica um ponto no círculo, abre o círculo e aproveita o algoritmo 4.7 para gerar todos os conjuntos independentes maximais de um grafo de intervalo. O procedimento é descrito a seguir.

**Grafos arco-circulares**

Um grafo  $G(V,E)$  é arco-circular se existe uma correspondência 1-1 entre o seu conjunto de vértices  $V$  e uma família  $A = \{A_v\}_{v \in V}$  de arcos de circunferência de um círculo, de forma que, para todo par de vértices distintos  $u, v$  tem-se:  $(u,v) \in E$  se e somente se  $A_u$  e  $A_v$  se interceptam; isto é, os vértices estão unidos se os arcos de circunferência correspondentes se interceptam.  $A$  é denominado modelo arco-circular de  $G$ .

Na figura 4.27 exhibe-se o ciclo induzido  $C_5$  e sua representação por arcos de circunferência. Em geral, todo ciclo induzido  $C_n$  é um grafo arco-circular.

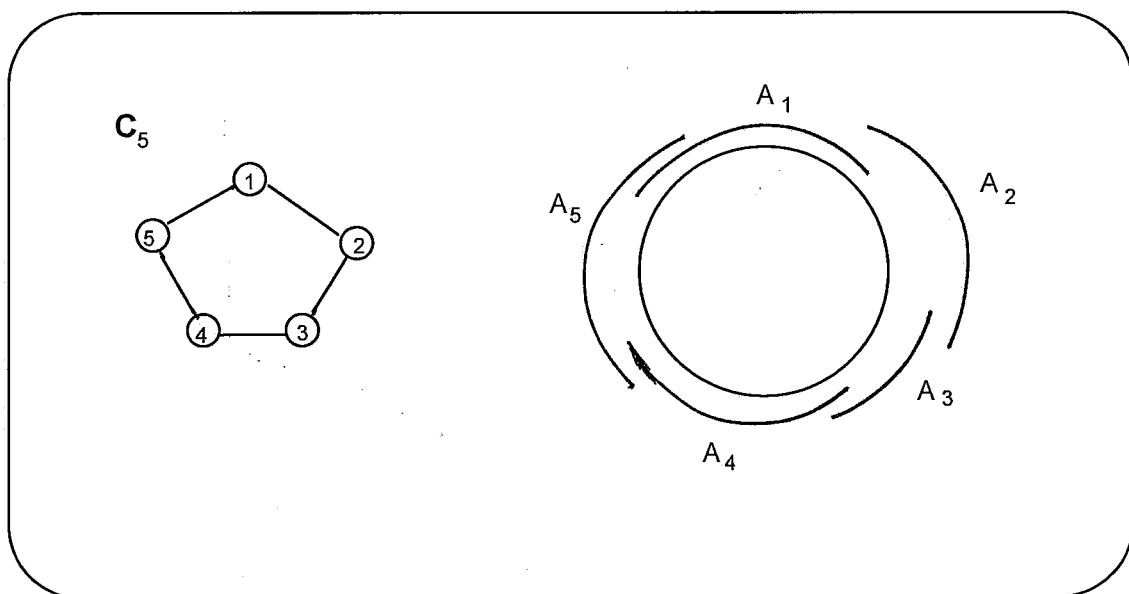


Fig. 4.27:  $C_5$  e  $\{A_j\}_{j=1,5}$

Leung considera  $G$  dado por uma representação por arcos de circunferência  $A = \{A_j\}_{j=1,n}$ . Supõe-se que os  $n$  arcos cobrem todo o círculo e que os pontos finais dos arcos são todos diferentes. Seja  $x$  um ponto qualquer no círculo, que não coincide com nenhum ponto final dos arcos de  $A$ . Caminhando sobre o círculo na direção dos ponteiros de um relógio, a partir do ponto  $x$ , denota-se os pontos finais do arco  $A_j$  como limite esquerdo  $e_j$  e limite direito  $d_j, j = 1, \dots, n$ .

Seja  $x$  um ponto no círculo tal que a região  $\widehat{xy}$  não contem nenhum ponto final dos arcos de  $A$ .

Seja  $A' = \{A_{j_1}, A_{j_2}, \dots, A_{j_p}\}$  o conjunto de arcos que contém  $\widehat{xy}$ , logo  $A - A'$  é um grafo de intervalo dado por sua representação por intervalos, considerando cada arco  $A_j$  como o intervalo  $[e_j, d_j]$  na reta real.

A idéia do algoritmo de Leung para gerar a família de c.i.m. do grafo arco-circular  $G$  é:

1º Gerar todos os c.i.m. de  $A - A'$ .

Para cada c.i.m.  $I$  gerado: se todos os arcos de  $A'$  interceptam com algum elemento de  $I$  então  $I$  é c.i.m. de  $G$ , pois nenhum arco de  $A'$  pode ser agregado a  $I$ .

2º Para cada arco  $A_{j_i}$  em  $A'$ : enumerar todos os c.i.m. que contém  $A_{j_i}$ , modificando o algoritmo de Leung para grafos de intervalo.

Define-se  $\forall A_j \in A$ :

$$NEXT(A_j) = \begin{cases} A_k \in A \text{ tal que } A_k \text{ não intercepta com } A_j \text{ e } d_k \text{ é o mais próximo a } d_j \\ \text{(no sentido do relógio começando do lado direito de } A_j \text{)} \\ \text{entre todos os arcos que não interceptam } A_j \\ \lambda \quad \text{caso contrário} \end{cases}$$

$$NEXTGROUP(A_j) = \begin{cases} \left\{ A_t \in A / A_t \text{ não intercepta } A_j \text{ e } d_{NEXT(A_j)} \in A_t \right\} \\ \text{(ordenados em forma crescente da distância dos lados direitos)} \\ \emptyset \quad \text{se } NEXT(A_j) = \lambda \end{cases}$$

$NEXT(A_j)$  corresponde ao primeiro arco posterior a  $A_j$ , segundo a ordem dada a partir do ponto  $x$ , que não intercepta com  $A_j$ .

$NEXTGROUP(A_j)$  corresponde ao conjunto de arcos posteriores a  $A_j$  no círculo, que não interceptam com  $A_j$  mas interceptam com  $NEXT(A_j)$ , inclui a  $NEXT(A_j)$ .

O algoritmo 4.11 a seguir, implementa o método de Leung.

**Algoritmo 4.11: Família de Conjuntos Independentes Maximais**

Dado  $G(V,E)$  grafo arco-circular por  $\{A_j\}_{j=1,n}$ ,  $A_j$  descrito por  $e_j, d_j$ ,  $|V|=n$

1.  $F = \emptyset$

Para todo  $j=1$  até  $n$  efetuar

determinar  $NEXT(A_j)$

determinar  $NEXTGROUP(A_j)$

2. Escolher  $x$  um ponto qualquer no círculo

determinar  $y$  tal que  $xy$  não contém nenhum ponto final dos arcos  $A_j$ ,  $j = 1, \dots, n$

$A' = \{ A_{j_i} / xy \subset A_{j_i} \}$

$p = |A'|$

3. Construir a família  $F'$  de c.i.m. do grafo representado pelos intervalos reais de  $A - A'$

4. Para todo  $I \in F'$  efetuar

para todo  $B \in A'$  efetuar

se  $\forall D \in I$  tem-se  $D \cap B = \emptyset$  então ir a 4

$F = F \cup \{I\}$

$L_1 = A'$

$k = 1$

seja  $B$  o primeiro arco de  $L_1$

5. enquanto  $NEXTGROUP(B) \neq \emptyset$  e

$\exists K \in NEXTGROUP(B)$  tal que  $K$  não intercepte  $L_1(1), \dots, L_{k-1}(1)$  efetuar

$k = k + 1$

para todo  $D \in NEXTGROUP(B)$

se  $D$  não intercepta  $L_1(1), \dots, L_{k-1}(1)$  então  $L_k = L_k \cup D$

$B = NEXT(B)$

6. Listar  $|L_k|$  c.i.m. escolhendo o primeiro elemento das  $(k-1)$  primeiras listas e um elemento de  $L_k$

7.  $k = k - 1$

8. Enquanto  $|L_k| = 1$  e  $k \geq 1$  efetuar

$k = k - 1$

9. Se  $k = 0$  então PARAR

caso contrário efetuar

retirar o primeiro elemento de  $L_k$

$B =$  primeiro elemento de  $L_k$

se  $NEXTGROUP(B) \neq \emptyset$  e  $\exists K \in NEXTGROUP(B)$

tal que  $K$  não intercepta  $L_1(1), \dots, L_{k-1}(1)$  então ir a 5

caso contrário efetuar

listar um c.i.m. pegando o primeiro elemento de  $L_1, \dots, L_k$

ir a 8.

**Complexidade:**

A complexidade do algoritmo 4.11 é determinada pelo passo 3 para gerar a família de c.i.m de um grafo de intervalo, utilizando o algoritmo 4.7 de Leung é  $O(n^2 + \beta)$ , onde  $\beta$  é a soma do número de vértices de todos os c.i.m..

Para o ciclo induzido  $C_5$  com a representação  $A = \{A_1, A_2, \dots, A_5\}$  da figura 4.27, sejam  $x$  e  $y$  como exibidos na figura 4.28.

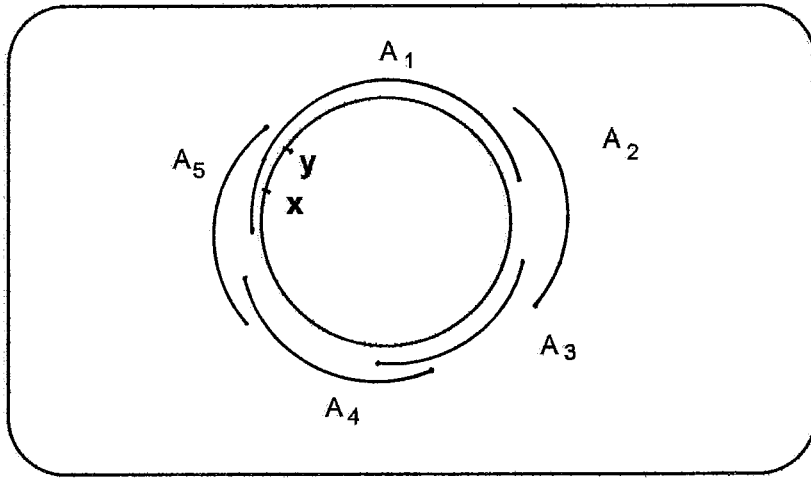


Fig. 4.28: Exemplo para  $C_5$

Com isto:

$$A' = \{A_1, A_5\}$$

$$p = 2$$

$j$	$NEXT(A_j)$	$NEXTGROUP(A_j)$
1	$A_3$	$A_3, A_4$
2	$A_4$	$A_4, A_5$
3	$A_5$	$A_5, A_1$
4	$A_1$	$A_1, A_2$
5	$A_2$	$A_2, A_3$

Na figura 4.29 exibe-se a representação  $A - A'$  por intervalos da reta real e o grafo de intervalo  $G - G'$  representado por esta família, para o caso do ciclo induzido  $C_5$ .

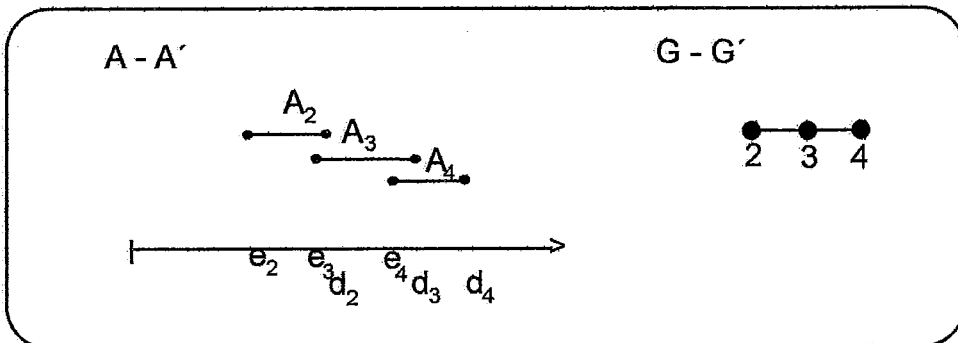


Fig. 4.29:  $A - A'$  associado a  $C_5$

A família de c.i.m. de  $G - G'$  é  $F(G - G') = \{\{2, 4\}, \{3\}\}$ .

Para  $I = \{2,4\}$  tem-se:  $A_1 \cap A_2 \neq \emptyset$ ,  $A_5 \cap A_4 \neq \emptyset$ , logo  $\{2,4\}$  é c.i.m. de  $G$ .  
 Para  $I = \{3\}$  tem-se:  $A_1 \cap A_3 = \emptyset$ , logo  $\{3\}$  não é c.i.m. de  $G$ .

ITERAÇÃO (passo 5)	LISTAS			c.i.m.
	$L_1$	$L_2$	$L_3$	
1	$A_1$ $A_5$	$A_3$ $A_4$	$A_5$	1,3,5
2	$A_1$ $A_5$	$A_4$		1,4
3	$A_5$	$A_2$		5,2

Logo, a família de c.i.m. de  $C_5$  é  $F(C_5) = \{ \{2,4\}, \{1,3,5\}, \{1,4\}, \{2,5\} \}$ .

Leung apresenta como exemplo o grafo de intervalo  $G_L$  da figura 4.30.

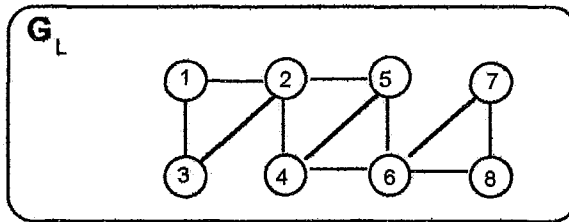


Fig. 4.30: Grafo de Leung

A partir desse grafo constrói-se o grafo arco-circular com o modelo arco-circular dado na figura 4.31, agregando os vértices 9 e 10 e as arestas  $(7,9)$ ,  $(8,9)$ ,  $(3,10)$ ,  $(8,10)$  e  $(9,10)$ .

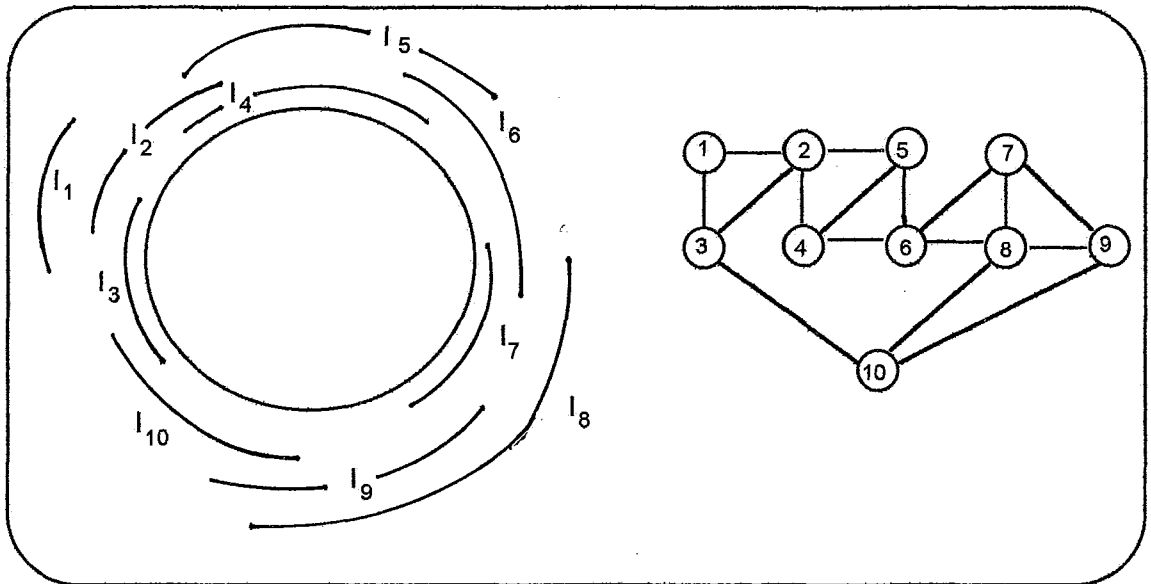


Fig. 4.31: Exemplo de grafo arco-circular



Tem-se:

j	NEXT(A <sub>j</sub> )	NEXTGROUP(A <sub>j</sub> )
1	A <sub>4</sub>	A <sub>4</sub> , A <sub>5</sub> , A <sub>6</sub>
2	A <sub>6</sub>	A <sub>6</sub> , A <sub>7</sub> , A <sub>8</sub>
3	A <sub>4</sub>	A <sub>4</sub> , A <sub>5</sub> , A <sub>6</sub>
4	A <sub>7</sub>	A <sub>7</sub> , A <sub>8</sub> , A <sub>9</sub>
5	A <sub>7</sub>	A <sub>7</sub> , A <sub>8</sub> , A <sub>9</sub>
6	A <sub>9</sub>	A <sub>9</sub> , A <sub>10</sub>
7	A <sub>10</sub>	A <sub>10</sub> , A <sub>3</sub>
8	A <sub>3</sub>	A <sub>3</sub> , A <sub>1</sub> , A <sub>2</sub>
9	A <sub>3</sub>	A <sub>3</sub> , A <sub>1</sub> , A <sub>2</sub>
10	A <sub>1</sub>	A <sub>1</sub> , A <sub>2</sub>

Sejam x e y tais que  $x \in A_9 \cap A_{10}$ ,  $y \in A_9 \cap A_{10}$ ,

$$A = \{A_1, \dots, A_{10}\}$$

$$A' = \{A_9, A_{10}\}$$

$G - G'$  coincide com o grafo dado por Leung, com a representação por intervalos  $A-A'$  apresentada na figura 4.32.

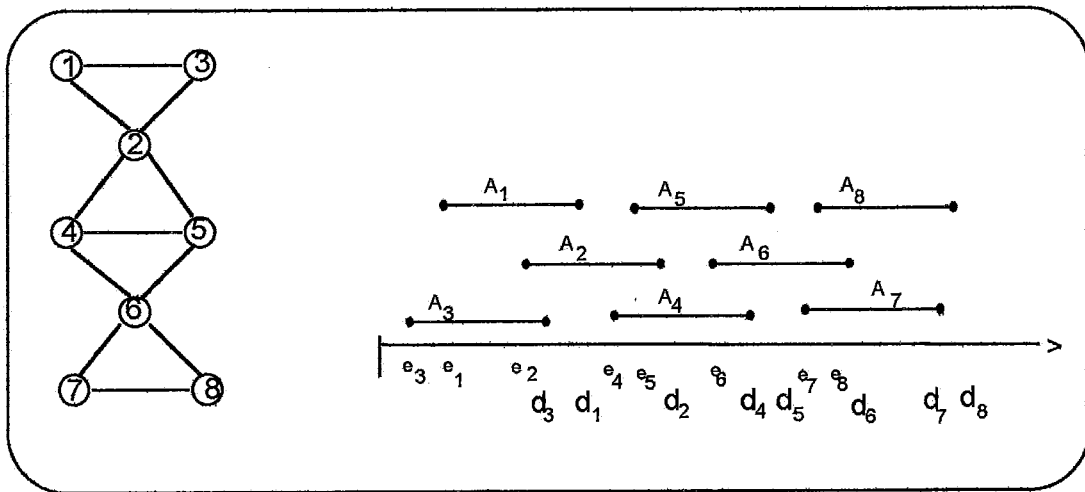


Fig. 4.32: Exemplo de Leung

A família de c.i.m. de  $G - G'$  é  $\{\{1,6\}, \{1,4,7\}, \{1,4,8\}, \{1,5,7\}, \{1,5,8\}, \{2,6\}, \{2,7\}, \{2,8\}, \{3,6\}, \{3,4,7\}, \{3,4,8\}, \{3,5,7\}, \{3,5,8\}\}$ .

Para  $I = \{1,4,7\}$  tem-se:  $A_9 \cap A_7 \neq \emptyset$ ,

$A_{10} \cap A_1 = \emptyset$ ,  $A_{10} \cap A_4 = \emptyset$ ,  $A_{10} \cap A_7 = \emptyset$ , logo  $\{1,4,7\}$  não é c.i.m. de  $G$ .

Para  $I = \{1,4,8\}$  tem-se:

$A_9 \cap A_8 \neq \emptyset$ ,  $A_{10} \cap A_8 \neq \emptyset$ , logo  $\{1,4,8\}$  é c.i.m. de  $G$ .

Para  $I = \{1,5,7\}$  tem-se:  $A_9 \cap A_7 \neq \emptyset$

$A_{10} \cap A_1 = \emptyset$ ,  $A_{10} \cap A_5 = \emptyset$ ,  $A_{10} \cap A_7 = \emptyset$ , logo  $\{1,5,7\}$  não é c.i.m. de  $G$ .

Para  $I = \{1,5,8\}$  tem-se:

$$A_9 \cap A_8 \neq \emptyset, A_{10} \cap A_8 \neq \emptyset, \text{ logo } \{1,5,8\} \text{ é c.i.m. de } G.$$

Para  $I = \{1,6\}$  tem-se:

$$A_9 \cap A_1 = \emptyset, A_9 \cap A_6 = \emptyset, \text{ logo } \{1,6\} \text{ não é c.i.m. de } G.$$

Para  $I = \{2,6\}$  tem-se:

$$A_9 \cap A_2 = \emptyset, A_9 \cap A_6 = \emptyset, \text{ logo } \{2,6\} \text{ não é c.i.m. de } G.$$

Para  $I = \{2,7\}$  tem-se:  $A_9 \cap A_7 \neq \emptyset$

$$A_{10} \cap A_2 = \emptyset, A_{10} \cap A_7 = \emptyset, \text{ logo } \{2,7\} \text{ não é c.i.m. de } G.$$

Para  $I = \{2,8\}$  tem-se:

$$A_9 \cap A_8 \neq \emptyset, A_{10} \cap A_8 \neq \emptyset, \text{ logo } \{2,8\} \text{ é c.i.m. de } G.$$

Para  $I = \{3,4,7\}$  tem-se:

$$A_9 \cap A_7 \neq \emptyset, A_{10} \cap A_3 \neq \emptyset, \text{ logo } \{3,4,7\} \text{ é c.i.m. de } G.$$

Para  $I = \{3,4,8\}$  tem-se:

$$A_9 \cap A_8 \neq \emptyset, A_{10} \cap A_8 \neq \emptyset, \text{ logo } \{3,4,8\} \text{ é c.i.m. de } G.$$

Para  $I = \{3,5,7\}$  tem-se:

$$A_9 \cap A_7 \neq \emptyset, A_{10} \cap A_3 \neq \emptyset, \text{ logo } \{3,5,7\} \text{ é c.i.m. de } G.$$

Para  $I = \{3,5,8\}$  tem-se:

$$A_9 \cap A_8 \neq \emptyset, A_{10} \cap A_8 \neq \emptyset, \text{ logo } \{3,5,8\} \text{ é c.i.m. de } G.$$

Para  $I = \{3,6\}$  tem-se:

$$A_9 \cap A_3 = \emptyset, A_9 \cap A_6 = \emptyset, \text{ logo } \{3,6\} \text{ não é c.i.m. de } G.$$

ITERAÇÃO (passo 5)	LISTAS				c.i.m.
	$L_1$	$L_2$	$L_3$	$L_4$	
1	$A_9$	$A_3$	$A_4$		9,3,4
	$A_{10}$	$A_1$	$A_5$		9,3,5
		$A_2$	$A_6$		9,3,6
2	$A_9$	$A_1$	$A_4$		9,1,4
	$A_{10}$	$A_2$	$A_5$		9,1,5
			$A_6$		9,1,6
3	$A_9$	$A_2$	$A_6$		9,2,6
	$A_{10}$				
4	$A_{10}$	$A_1$	$A_4$	$A_7$	10,1,4,7
		$A_2$	$A_5$		
			$A_6$		
5	$A_{10}$	$A_1$	$A_5$	$A_7$	10,1,5,7
		$A_2$	$A_6$		
6	$A_{10}$	$A_1$	$A_6$		10,1,6
		$A_2$			
7	$A_{10}$	$A_2$	$A_6$		10,2,6
			$A_7$		10,2,7

A família de c.i.m. de  $G$  é  $\{\{1,6,10\}, \{1,4,7,10\}, \{1,4,8\}, \{1,5,7,10\}, \{1,5,8\}, \{2,6,10\}, \{2,7,10\}, \{2,8\}, \{3,6,9\}, \{3,4,7\}, \{3,4,8\}, \{3,5,7\}, \{3,5,8\}, \{3,4,9\}, \{3,5,9\}, \{1,4,9\}, \{1,5,9\}, \{1,6,9\}, \{2,6,9\}\}$ .

### 4.5. Grafos Triangularizados

Um grafo  $G(V,E)$  é triangularizado ou cordal se cada ciclo de comprimento maior do que três possui uma corda.

O grafo  $G$  da figura 4.33 é triangularizado.

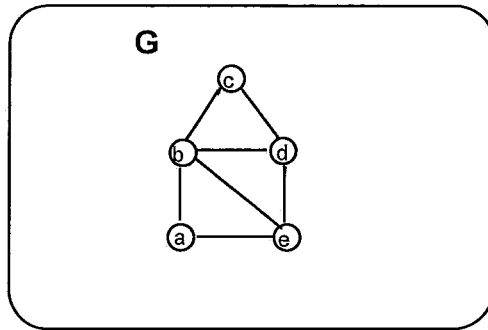


Fig. 4.33: Exemplo de grafo triangularizado

Todo grafo de intervalo é triangularizado.

Um vértice  $v \in V$  é simplicial se o subgrafo induzido por  $Adj(v)$  for completo. No exemplo da figura 4.33,  $a$  é simplicial e  $d$  não o é.

Seja  $S = u_1, u_2, \dots, u_n$  uma seqüência de ordenação dos vértices de  $G$ .  $S$  é um esquema de eliminação perfeita (E.E.P.) se  $\forall u_i \in V$  tem-se que  $u_i$  é um vértice simplicial de  $G - \{u_1, \dots, u_{i-1}\}$ ,  $S^{-1}(v)$  corresponde à ordem de  $v$  segundo a seqüência  $S$ . Para o grafo da figura 4.33,  $S = c, a, b, d, e$  é um esquema de eliminação perfeita de  $G$ . Isto dá uma nova ordenação dos vértices exibida na tabela seguinte:

$v$	$S^{-1}(v)$
$a$	2
$b$	3
$c$	1
$d$	4
$e$	5

Dada uma ordenação  $S$  dos vértices de  $G$ , para  $v \in V$  define-se o conjunto dos vértices monotonamente adjacentes a ele:

$$MAdj(v) = \{ w \in Adj(v) / w \text{ está à direita de } v \text{ em } S ( S^{-1}(v) < S^{-1}(w) ) \}.$$

Para o exemplo da figura 4.33, com  $S = c, a, b, d, e$ , tem-se:  $MAdj(a) = \{b,e\}$   
 $MAdj(b) = \{d,e\}$      $MAdj(c) = \{b,d\}$      $MAdj(d) = \{e\}$      $MAdj(e) = \{ \}$ .

**Propriedade 4.2 (Fulkerson e Gross, 1965):**

Se  $G(V,E)$  é triangularizado com  $\text{card}(V) > 1$  então  $\forall v \in V$  o grafo  $G-v$  é triangularizado.

**Propriedade 4.3 (Rose, 1970):**

$G$  é triangularizado se e somente se  $G$  possuir um esquema de eliminação perfeita.

**Algoritmo de Gavril**

Gavril (1972) desenvolve um algoritmo para determinar um conjunto independente máximo de um grafo triangularizado, em tempo  $O(n + m)$ , dado um E.E.P. do grafo. Esse procedimento é descrito no algoritmo 4.12 a seguir.

**Algoritmo 4.12: Conjunto Independente Máximo**

Dado  $G(V,E)$  grafo triangularizado,  $S$  um E.E.P. de  $G$  e  $MAdj(v) \forall v \in V$

1. Desmarcar todos os vértices
2.  $J = \emptyset$
3. Repetir até que todos os vértices estejam marcados  
 determinar o menor  $j$  tal que  $u_j$  está desmarcado  
 $J = J \cup \{u_j\}$   
 marcar  $u_j$  e todo  $w \in MAdj(u_j)$
4. Devolver  $J$ .

Para o grafo  $G$  da figura 4.33, com  $S = c, a, b, d, e$ , obtém-se  $J = \{a,c\}$ .

Para o grafo de intervalo  $G_L$ , exemplo do Leung (1984), exibido na figura 4.34 tem-se  $S = a, b, c, d, e, f, g, h$ .

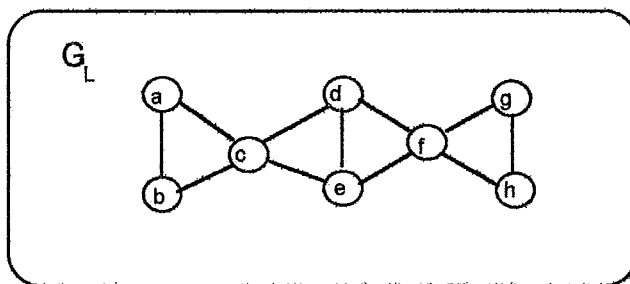


Fig. 4.34: Grafo de Leung

Com este E.E.P. tem-se  $MAdj(a) = \{b, c\}$   $MAdj(b) = \{c\}$   $MAdj(c) = \{d, e\}$   
 $MAdj(d) = \{e, f\}$   $MAdj(e) = \{f\}$   $MAdj(f) = \{g, h\}$   $MAdj(g) = \{h\}$   
 $MAdj(h) = \{ \}$  e gera-se o conjunto independente máximo  $J = \{a,d,g\}$ .

## Algoritmo para gerar a família $F(G)$ de c.i.m de $G$

### Algoritmo de Leung

Leung (1984) gera todos os c.i.m. de um grafo triangularizado  $G$ . Ele determina, primeiro, um conjunto independente máximo  $J$  com o algoritmo de Gavril e efetua backtracking segundo a ordem dos vértices dada pelo E.E.P..

Inicialmente todos os vértices estão desmarcados. Marcar  $u_1$  e todo  $w$  em  $MAdj(u_1)$  com o valor 1, unir  $u_1$  e  $MAdj(u_1)$  na lista  $L_1$ . Os vértices de  $L_1$  induzem uma clique e todo vértice não marcado é independente com  $u_1$ . Para identificar  $u_j$  que possa estar no mesmo c.i.m. que  $u_1$ , determinar o menor  $j$  tal que  $u_j$  não está marcado (o menor pela maximalidade do c.i.m.). Marcar  $u_j$  e todo  $w \in MAdj(u_j)$  que não esteja marcado com o valor 2, unir  $u_j$  e  $MAdj(u_j)$  na lista  $L_2$  com  $L_2(1) = u_j$ . Os vértices de  $L_2$  formam uma clique e são independentes com  $u_1$ .

Repete-se o processo até que todos os vértices estejam marcados, criando-se  $k$  listas. Cada vértice aparece em uma lista  $L_j$  de comprimento  $l(L_j)$  e está marcado com o número de ordem da lista a qual pertence. Cada lista induz uma clique. Cada vértice de  $L_i$  é independente com o primeiro elemento de  $L_1, L_2, \dots, L_{i-1}$ . A partição dos vértices nas listas corresponde a uma cobertura de  $V$  por cliques.

Ao enumerar  $L_1(1), L_2(1), \dots, L_{k-1}(1), L_k(j)$  para todo  $j = 1, \dots, l(L_k)$  obtém-se todos os c.i.m. que contém  $L_1(1), L_2(1), \dots, L_{k-1}(1)$ . Depois precisa-se gerar todos os c.i.m. que contém  $L_1(1), L_2(1), \dots, L_{k-1}(2)$ ; para isto: desmarcar todos os vértices de  $L_k$ , examinar os vértices adjacentes a  $L_{k-1}(2)$ . Logo com  $L_{k-1}(3)$  e assim por diante fazendo o backtracking. Utiliza-se um ponteiro  $PTR(j)$  para indicar o vértice da lista  $L_j$  que está sendo examinado.

Esse método é codificado no algoritmo 4.13 seguinte:

**Algoritmo 4.13: Família de Conjuntos Independentes Maximais**

Dado  $G(V,E)$  grafo triangularizado,  $S$  um E.E.P. de  $G$  e  $MAdj(v) \forall v \in V$

1. Para  $j = 1, 2, \dots, n$   
 $MARCA(u_j) = 0$   
 $k = 0$
2. Para  $j = 1, 2, \dots, n$  efetuar  
 se  $MARCA(u_j) = 0$  então  
 $k = k + 1$   
 $MARCA(u_j) = k$   
 $PTR(k) = 1$   
 $L_k(1) = u_j$   
 $I(k) = 1$   
 para todo  $w \in MAdj(u_j)$  efetuar  
 se  $MARCA(w) = 0$  então  
 $MARCA(w) = k$   
 $I(k) = I(k) + 1$   
 $L_k(I(k)) = w$
3. Para  $j = 1, 2, \dots, I(k)$   
 listar o c.i.m. dado por  $L_1(PTR(1)), L_2(PTR(2)), \dots, L_{k-1}(PTR(k-1)), L_k(j)$   
 $MARCA(L_k(j)) = 0$
4.  $k = k - 1$
5. Para todo  $w \in Adj(L_k(PTR(k)))$  efetuar  
 se  $MARCA(w) = -k$  então  $MARCA(w) = 0$
6.  $PTR(k) = PTR(k) + 1$
7. Se  $PTR(k) > I(k)$  então efetuar  
 para  $j = 1, 2, \dots, I(k)$  efetuar  
 $MARCA(L_k(j)) = 0$   
 $k = k - 1$   
 se  $k = 0$  então PARAR.  
 caso contrário ir a 5  
 caso contrário efetuar  
 para todo  $w \in Adj(L_k(PTR(k)))$  efetuar  
 se  $MARCA(w) = 0$  então  $MARCA(w) = -k$   
 se existe  $v \in V$  com  $MARCA(v) = 0$  então ir a 2  
 caso contrário efetuar  
 listar o c.i.m. dado por  $L_k(PTR(j))$  para  $j = 1, \dots, k-1$  e  $L_k(PTR(k))$   
 ir a 5.

**Complexidade:**

A complexidade por c.i.m. é  $O(m)$ , sendo  $G$  conexo; a complexidade total do algoritmo 4.13 é  $O((n + m) i(G))$ .

Para o grafo  $G$  da figura 4.33, com  $S = c, a, b, d, e$ , tem-se:

ITERAÇÃO (passo 2)	LISTAS		c.i.m.
	$L_1$	$L_2$	
1 PTR(1) = 1	c b d	a e	c,a c,e
2 PTR(1) = 2	c b d		b
3 PTR(1) = 3	c b d	a	d,a

Para o exemplo do Leung, exibido na figura 4.34, com  $S = a, b, c, d, e, f, g, h$  tem-se:

ITERAÇÃO (passo 2)	LISTAS			c.i.m.
	$L_1$	$L_2$	$L_3$	
1 PTR(1) = 1 PTR(2) = 1	a b c	d e f	g h	a,d,g a,d,h
2 PTR(1) = 1 PTR(2) = 2	a b c	d e f	g h	a,e,g a,e,h
3 PTR(1) = 1	a b c	f		a,f
4 PTR(1) = 2 PTR(2) = 1	a b c	d e f	g h	b,d,g b,d,h
5 PTR(1) = 2 PTR(2) = 2	a b c	d e f	g h	b,e,g b,e,h
6 PTR(1) = 2	a b c	d e f		b,f
7 PTR(1) = 3	a b c	f g h		c,f c,g c,h

logo, a família de c.i.m. do grafo do Leung é:  $\{ \{a,d,g\}, \{a,d,h\}, \{a,e,g\}, \{a,e,h\}, \{a,f\}, \{b,d,g\}, \{b,d,h\}, \{b,e,g\}, \{b,e,h\}, \{b,f\}, \{c,f\}, \{c,g\}, \{c,h\} \}$ .

## Algoritmo proposto

Seja:  $G$  um grafo triangularizado,

$S = u_1, u_2, \dots, u_n$  um esquema de eliminação perfeita de  $G$ ,

$F(G)$  a família de conjuntos independentes maximais de  $G$ ,

$i(G) = \text{card}(F(G))$  o número de c.i.m de  $G$ .

Para o grafo  $G$  da figura 4.33 tem-se  $F(G) = \{ \{c,a\}, \{c,e\}, \{b\}, \{d,a\} \}$ ,  $i(G) = 4$ .

### Observação 4.21:

Considerando um conjunto independente maximal de  $G$  como a seqüência de vértices  $I = v_1, v_2, \dots, v_k$  com  $k \leq n$ , observa-se:

- o primeiro vértice  $v_1$  de  $I$  corresponde ao primeiro elemento  $u_1$  do esquema de eliminação perfeita  $S$  ou a um vértice de  $MAdj(u_1)$ ,
- se até um passo dado, tem-se construído  $I$  até o seu  $s$ -ésimo elemento então o elemento seguinte  $v_{s+1}$  será um vértice que não pertença ao conjunto  $MAdj(v_s)$ .

Dado que, para cada  $v_s$  da seqüência existem diferentes possibilidades para escolher o elemento  $v_{s+1}$ , produzindo cada uma delas um c.i.m diferente, estas possibilidades correspondem em um grafo direcionado  $D$  ao conjunto de vértices  $V(D) = V(G)$ . O conjunto FONTE de fontes de  $D$  é dado por  $u_1$  e todos os vértices de  $MAdj(u_1)$ , o conjunto SUMID de sumidouros de  $D$  é formado pelo último elemento da seqüência  $S$  e todos os  $u_j$  não fonte tais que  $\{w / S^{-1}(w) > S^{-1}(u_j)\} - \text{FONTE} - MAdj(u_j) = \emptyset$ , isto é  $\{u_j / \forall w \text{ não fonte com } S^{-1}(w) > S^{-1}(u_j), w \in Adj(u_j)\}$ . Existe o arco  $(v,w)$  em  $D$  se  $v$  não é sumidouro,  $w$  não é fonte,  $[S^{-1}(v) < S^{-1}(w)$  e  $w$  não pertence a  $MAdj(v)$ ] ou  $[S^{-1}(v) > S^{-1}(w)$  e  $v$  não está em  $MAdj(w)$ ].

Um vértice universal em  $G$  é um vértice isolado em  $D$ .

O tamanho de um conjunto independente máximo  $\alpha(G)$  corresponde ao comprimento de um caminho mais longo em  $D$ .

### Lema 4.12:

Dado  $S$  um E.E.P. de  $G$  grafo triangularizado:

A família  $F(G)$  de c.i.m. de  $G$  corresponde ao conjunto de caminhos maximais fonte-sumidouro em um digrafo  $D$  com  $V(D) = V(G)$  tal que existe o arco  $(v,w)$  em  $E(D)$  se  $v$  não é sumidouro,  $w$  não é fonte,  $[S^{-1}(v) < S^{-1}(w)$  e  $w$  não pertence a  $MAdj(v)$ ] ou  $[S^{-1}(v) > S^{-1}(w)$  e  $v$  não está em  $MAdj(w)$ ]. O conjunto de fontes de  $D$  é dado por  $u_1$  e todos os vértices de  $MAdj(u_1)$ , o conjunto de sumidouros de  $D$  é formado pelo último elemento da seqüência  $S$  e todos os  $u_j$  não fonte tais que

$$\{w / S^{-1}(w) > S^{-1}(u_j)\} - \text{FONTE} - MAdj(u_j) = \emptyset,$$

isto é  $\{u_j / \forall w \text{ não fonte com } S^{-1}(w) > S^{-1}(u_j), w \in Adj(u_j)\}$ .



**Prova:**

Dado o digrafo  $D$ , construído segundo o lema 4.12, tem-se que cada caminho maximal  $I = v_1, v_2, \dots, v_k$  em  $D$  desde uma fonte  $v_1$  até um sumidouro  $v_k$  corresponde a um c.i.m de  $G$  pois, em  $D$  existe o arco  $(v,w)$  se e somente se  $(v,w) \notin E(G)$ , isto é sse  $v \notin Adj(w)$  e  $w \notin Adj(v)$ , logo é conjunto independente e  $I$  é maximal pois não é possível acrescentar mais vértices a  $I$ , dado que: a fonte  $v_1 \in [\{u_1\} \cup MAdj(u_1)]$ , o sumidouro  $v_k \in [\{u_n\} \cup \{u_j / \forall w \text{ não fonte com } S^{-1}(w) > S^{-1}(u_j), w \in Adj(u_j)\}]$ , se  $v \in I$  então algum  $w \notin MAdj(v)$  com  $S^{-1}(v) < S^{-1}(w)$  deve pertencer a  $I$ . Os caminhos têm que ser fonte-sumidouro e têm que ser maximais não incluindo arestas induzidas por transitividade, caso contrário o c.i.m. não é maximal.

Dado um c.i.m.  $I = \{v_1, v_2, \dots, v_k\}$  de  $G$ , a seqüência  $I$  dada por  $v_1, v_2, \dots, v_k$  corresponde a um caminho fonte-sumidouro em  $D$ , pois como  $I$  é maximal então:

$v_1$  deve pertencer a  $[\{u_1\} \cup MAdj(u_1)]$ ,

$v_k \in [\{u_n\} \cup \{u_j / \forall w \text{ não fonte com } S^{-1}(w) > S^{-1}(u_j), w \in Adj(u_j)\}]$ ,

se  $v_p \in I$  então  $v_{p+1}$  é igual a algum  $w \in MAdj(v)$  com  $S^{-1}(v) < S^{-1}(w)$ .

◆

Por exemplo, para o grafo  $G$  da figura 4.33, com  $S = c, a, b, d, e$ , tem-se:

$$\begin{aligned} \text{FONTE} &= \{c,b,d\} \\ \text{SUMID} &= \{a,e\} \end{aligned}$$

Na figura 4.35 apresenta-se o digrafo  $D$  associado a  $G$  e a redução transitiva  $D_R$  de  $D$ . Com isto gera-se a família de c.i.m  $F(G) = \{ \{c,a\}, \{c,e\}, \{b\}, \{d,a\} \}$ .

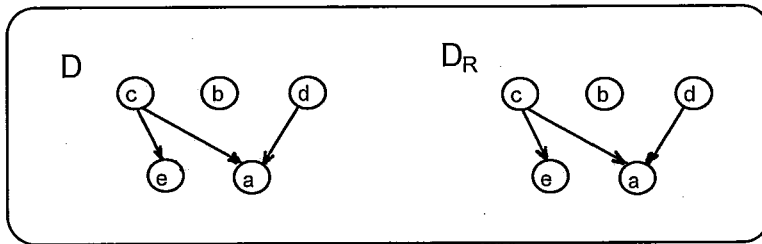


Fig. 4.35: Digrafo  $D$  associado a  $G$  e redução transitiva  $D_R$

Para o mesmo grafo  $G$  da figura 4.33, considerando o E.E.P.  $S = a, c, b, d, e$ , tem-se:  $\text{FONTE} = \{a,b,e\}$

$$\text{SUMID} = \{c,d\}$$

$v$	$S^{-1}(v)$	$MAdj(v)$
<b>a</b>	1	<b>b,e</b>
<b>b</b>	3	<b>d,e</b>
<b>c</b>	2	<b>b,d</b>
<b>d</b>	4	<b>e</b>
<b>e</b>	5	

**Observação 4.21:**

A família  $F(G)$  construída independe do E.E.P. considerado.

Na figura 4.36 apresenta-se o digrafo  $D$  associado a  $G$  e a redução transitiva  $D_R$  de  $D$ , obtidos com este novo E.E.P. Com isto gera-se a família de c.i.m  $F(G) = \{ \{a,c\}, \{e,c\}, \{b\}, \{a,d\} \}$ .

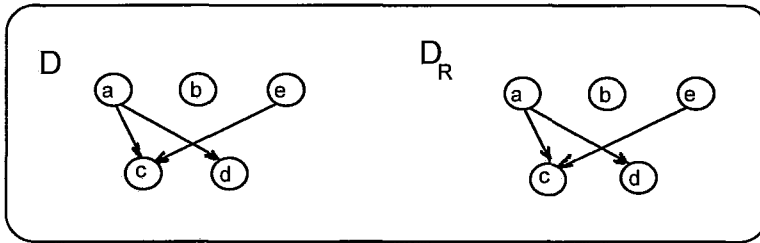


Fig. 4.36: Digrafo  $D$  associado a  $G$  e redução transitiva  $D_R$

**Algoritmo:**

Dado  $G$  e  $S$  um E.E.P. de  $G$ , desenvolve-se um algoritmo para construir a família  $F$  de c.i.m de  $G$ , utilizando a caracterização pelo digrafo  $D$  descrito no lema 4.12. O algoritmo 4.14 implementa este procedimento.

**Algoritmo 4.14: Família de c.i.m. de um grafo triangularizado**

Dado  $G(V,E)$  grafo triangularizado,  $S = u_1, u_2, \dots, u_n$  um esquema de eliminação perfeita de  $G$  e os conjuntos monotonaamente adjacentes  $MAdj(u_j), j = 1, \dots, n$

1. Construir o digrafo  $D$  com  $V(D) = V(G)$   
 $FONTE = \{u_1\} \cup MAdj(u_1)$   
 $SUMID = \{u_n\} \cup \{ u_j / \{ w / S^{-1}(w) > S^{-1}(u_j) \} - Adj(u_j) - FONTE = \emptyset \} - FONTE$   
 $(v,w) \in E(D) \Leftrightarrow v \notin SUMID, w \notin FONTE,$   
 $[S^{-1}(v) < S^{-1}(w) \text{ e } w \notin MAdj(v)]$   
 ou  $[S^{-1}(v) > S^{-1}(w), v \notin MAdj(w) \text{ e } (w,v) \notin E(D)]$
2. Construir a redução transitiva  $D_R$  de  $D$ .
3. Determinar todos os caminhos fonte-sumidouro de  $D_R$ .

**Complexidade:**

- Passo 1:  $O(n^2)$
- Passo 2:  $O(nm)$
- Passo 3:  $O(n)$  por caminho

A complexidade total do algoritmo 4.14 é  $O(n(m + i(G)))$ . No entanto, para  $i(G) > m$ , a complexidade por c.i.m. é  $O(n)$ , isto representa uma melhora em relação ao algoritmo de Leung (1984). O espaço requerido é  $O(n + m)$ .

**Observação 4.22:**

Dado um grafo triangularizado, construir um esquema de eliminação perfeita pode ser feito em tempo  $O(n + m)$  utilizando busca em largura lexicográfica, que permite o reconhecimento de um grafo de esta classe (Rose et alli, 1976).

Para o grafo de intervalo do Leung, da figura 4.25, com  $S = a, b, c, d, e, f, g, h$  tem-se:

$$\begin{aligned} \text{FONTE} &= \{a, b, c\} \\ \text{SUMID} &= \{g, h\} \end{aligned}$$

Na figura 4.37 exhibe-se o digrafo  $D_L$  associado ao grafo de Leung e sua redução transitiva  $D_{RL}$ .

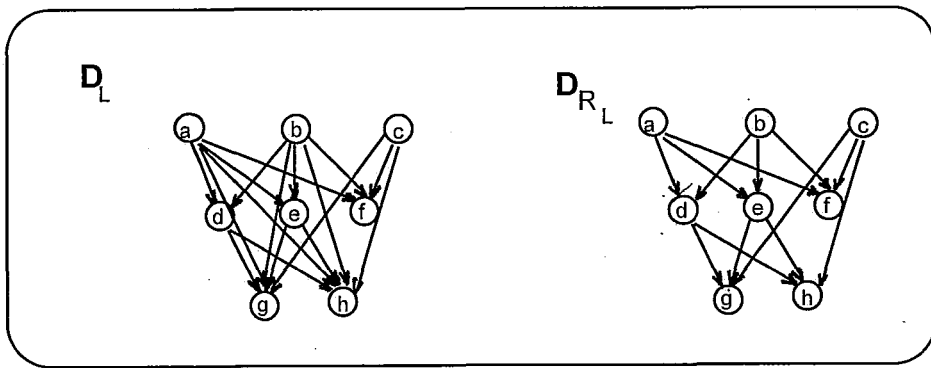


Fig. 4.37: Digrafo  $D_L$  associado ao grafo de Leung e redução transitiva  $D_{RL}$

Com isto gera-se a família de c.i.m  $F(G_L) = \{\{a, d, g\}, \{a, d, h\}, \{a, e, g\}, \{a, e, h\}, \{a, f\}, \{b, d, g\}, \{b, d, h\}, \{b, e, g\}, \{b, e, h\}, \{b, f\}, \{c, f\}, \{c, g\}, \{c, h\}\}$ .

## Capítulo 5

### SOLUÇÃO POR OPERAÇÕES EM GRAFOS

Neste capítulo apresenta-se os algoritmos desenvolvidos para construir a família de conjuntos independentes maximais no caso particular de grafos separáveis por cliques e grafos série-paralelo a dois terminais. Estes algoritmos baseiam-se na idéia de decompor o grafo de interesse em dois ou mais grafos menores, construir as famílias de conjuntos independentes maximais desses grafos, reconstruir o grafo original operando os grafos menores e, ao mesmo tempo, construir a família de conjuntos independentes maximais do grafo de interesse a partir das famílias correspondentes aos grafos menores. As operações consideradas são identificação de um vértice comum em dois grafos, o qual é uma articulação do grafo resultante da operação (conexão série); identificação de uma clique comum a dois grafos, o qual é um conjunto articulação do grafo resultante; e conexão paralela de dois grafos com dois vértices comuns a ambos.

#### 5.1. Grafos com Articulações

Seja  $G(V,E)$  com  $a \in V$ , onde  $a$  é uma articulação de  $G$ , isto é:  $G$  é conexo e  $G - a$  é desconexo. Logo,  $G$  pode ser decomposto em dois grafos  $G_1(V_1,E_1)$  e  $G_2(V_2,E_2)$  tais que:  $V_1 \cup V_2 = V$ ,  $V_1 \cap V_2 = \{a\}$ ,  $E_1 \cup E_2 = E$  com a forma geral exibida na figura 5.1.

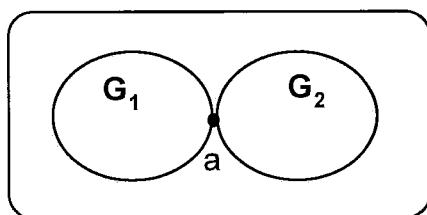


Fig. 5.1: Grafo com uma articulação

O grafo  $G$  é obtido por conexão série dos grafos  $G_1$  e  $G_2$  ( $G = G_1 * G_2$ ).

### Algoritmo de Boulala e Uhry

Boulala e Uhry (1979) desenvolvem um algoritmo, de tempo linear no número de vértices do grafo, para construir um conjunto independente de peso máximo. Para isto eles definem:

$S_a^1$  = conjunto independente de peso máximo do grafo  $G_1$  dentre os que contém  $a$

$s_a^1 = \text{card}(S_a^1) = \text{peso de } S_a^1$

$\bar{S}_a^1$  = conjunto independente de peso máximo do grafo  $G_1$  dentre os que não contém  $a$

$\bar{s}_a^1 = \text{card}(\bar{S}_a^1) = \text{peso de } \bar{S}_a^1$

$S^2$  = conjunto independente de peso máximo do grafo  $G_2$  considerando  $a$  com peso  $\alpha_a = s_a^1 - \bar{s}_a^1$

$s^2 = \text{card}(S^2) = \text{peso de } S^2$ .

O conjunto independente  $S$  de peso máximo do grafo  $G$  é dado por:

$$\text{se } a \in S^2 \text{ então } S = S^2 \cup S_a^1$$

$$\text{se } a \notin S^2 \text{ então } S = S^2 \cup \bar{S}_a^1$$

$$\text{com peso } s = s^2 + \bar{s}_a^1.$$

Para o grafo  $G$  da figura 5.2, o vértice  $a$  é uma articulação, considerando os grafos  $G_1$  induzido por  $\{a,b,c\}$  e  $G_2$  induzido por  $\{a,d,e\}$ , e todos os vértices com peso 1, então:

$$F(G_1) = \{ \{a\}, \{b\}, \{c\} \} \quad F(G_2) = \{ \{a\}, \{d\}, \{e\} \}$$

$$S_a^1 = \{a\} \quad s_a^1 = 1 \quad \bar{S}_a^1 = \{b\} \quad \bar{s}_a^1 = 1$$

peso de  $a$  em  $G_2$  muda a 0,  $S^2 = \{d\}$ ,  $a \notin S^2$ ,

logo um conjunto independente máximo é  $S = \{b,d\}$  com cardinal  $s = 2$ .

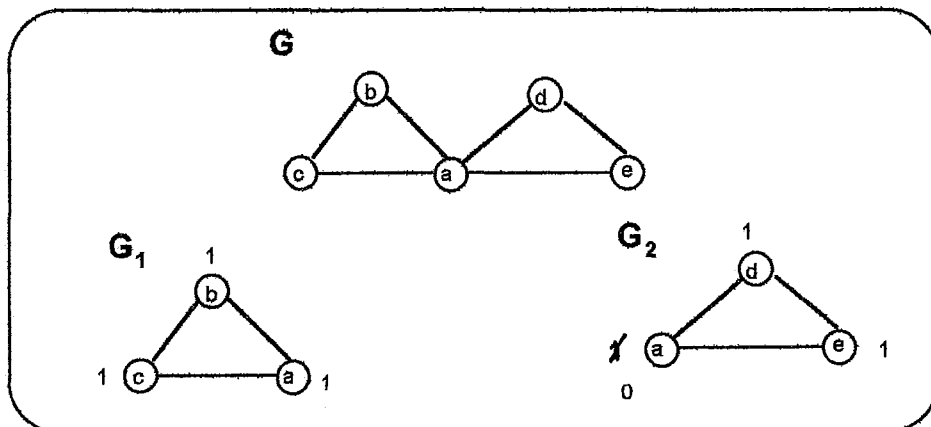


Fig. 5.2: Exemplo do algoritmo de Boulala e Uhry

### Algoritmo proposto

Seja:  $F(G)$  a família de conjuntos independentes maximais de  $G$ ,  
 $i(G) = \text{card}(F(G)) =$  número de c.i.m. de  $G$ ,  
 $a$  uma articulação de  $G$  que o separa em  $G_1$  e  $G_2$ .

A idéia é construir a família  $F(G)$  de conjuntos independentes maximais de  $G$  a partir das famílias de c.i.m. de  $G_1$  e de  $G_2$ .

#### Observação 5.1:

Dados  $I_1$  um conjunto independente maximal qualquer de  $G_1$  e  $I_2$  um conjunto independente maximal qualquer de  $G_2$ , tem-se as seguintes possibilidades:

(1)  $I_1 \cap I_2 = \{a\}$  (figura 5.3)  $\Rightarrow I = I_1 \cup I_2$  é c.i.m. de  $G$

$I$  é conjunto independente porque  $\nexists (u,v) \in E$  com  $u \in V_1 - \{a\}$  e  $v \in V_2 - \{a\}$ ,  
 $I$  é maximal porque  $I_1$  e  $I_2$  são maximais.

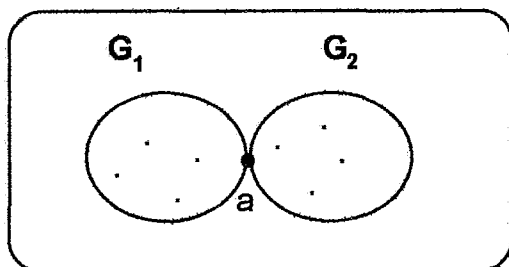


Fig. 5.3: Caso  $I_1 \cap I_2 = \{a\}$

(2)  $a \notin I_1, a \notin I_2$  (figura 5.4)  $\Rightarrow I = I_1 \cup I_2$  é c.i.m. de  $G$

$I$  é conjunto independente porque  $\nexists (u,v) \in E$  com  $u \in V_1 - \{a\}$  e  $v \in V_2 - \{a\}$ ,  
 por outra parte, dado que:  $a \notin I_1$  então  $\exists u \in I_1$  com  $(a,u) \in E_1$ ,  
 $a \notin I_2$  então  $\exists v \in I_2$  com  $(a,v) \in E_2$ ,  
 $I_1$  e  $I_2$  são maximais,

logo não é possível acrescentar mais vértices a  $I_1 \cup I_2$  e portanto  $I$  é maximal.

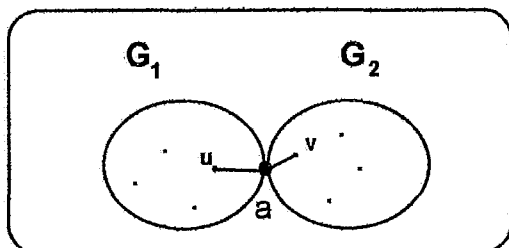


Fig. 5.4: Caso  $a \notin I_1, a \notin I_2$

(3)  $a \in I_1, a \notin I_2$  (figura 5.5)  $\Rightarrow I = [I_1 - \{a\}] \cup I_2$  é c.i.m. de  $G$

$a \notin I_2$  então  $\exists z \in I_2$  com  $(a,z) \in E_2$  caso contrário  $I_2$  não seria maximal, logo para  $I$  ser conjunto independente  $a$  não pode pertencer a  $I$ ,

$I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ ,

para garantir a maximalidade de  $I$  deve  $\exists v \in Adj_{G_1}(a)$  com  $(v,w) \in E(G_1), w \in I_1$ .

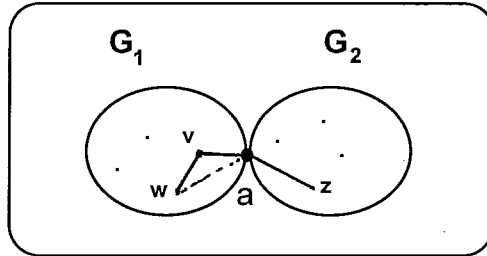


Fig. 5.5: Caso  $a \in I_1, a \notin I_2$

A observação 5.1 é a justificativa da propriedade 5.1 enunciada abaixo, a qual por sua vez é a base do procedimento descrito no algoritmo 5.1. Este procedimento independe da escolha da articulação.

**Propriedade 5.1:**

Dadas as famílias  $F(G_1)$  e  $F(G_2)$  de c.i.m. de  $G_1$  e de  $G_2$ , para  $i = 1, 2$  sejam:

$F_a^i$  = família de c.i.m. de  $G_i$  que contém  $a$

$\bar{F}_a^i$  = família de c.i.m. de  $G_i$  que não contém  $a$

então  $F(G)$  é construída segundo o seguinte procedimento:

1.- Para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in F_a^2$ :  $I_1 \cup I_2 \in F(G)$   
corresponde aos c.i.m. de  $G$  que contém  $a$

2.- Para todo  $I_1 \in \bar{F}_a^1$ , para todo  $I_2 \in \bar{F}_a^2$ :  $I_1 \cup I_2 \in F(G)$

3.- Para todo  $I_1 \in F_a^1$ :  
se  $\exists w \in I_1$  tal que  $\exists v \in Adj_{G_1}(a)$  com  $(v,w) \in E_1$  então

$$\forall I_2 \in \bar{F}_a^2: [I_1 - \{a\}] \cup I_2 \in F(G)$$

4.- Para todo  $I_2 \in F_a^2$ :  
se  $\exists w \in I_2$  tal que  $\exists v \in Adj_{G_2}(a)$  com  $(v,w) \in E_2$  então

$$\forall I_1 \in \bar{F}_a^1: I_1 \cup [I_2 - \{a\}] \in F(G).$$

Por exemplo, para o caminho  $P_8$  da figura 5.6, considerando a articulação  $a = 4$ , tem-se:

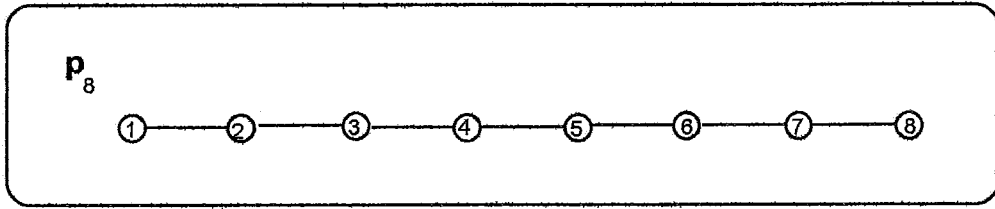


Fig. 5.6: Caminho  $P_8$

$$F(G_1) = \{ \{1,3\}, \{1,4\}, \{2,4\} \} \quad i(G_1) = 3$$

$$F(G_2) = \{ \{4,6,8\}, \{4,7\}, \{5,7\}, \{5,8\} \} \quad i(G_2) = 4$$

$$F_4^1 = \{ \{1,4\}, \{2,4\} \} \quad \bar{F}_4^1 = \{ \{1,3\} \}$$

$$F_4^2 = \{ \{4,6,8\}, \{4,7\} \} \quad \bar{F}_4^2 = \{ \{5,7\}, \{5,8\} \}$$

PASSO	$I_1$	$I_2$	c.i.m.
1	{1,4}	{4,6,8}	{1,4,6,8}
		{4,7}	{1,4,7}
	{2,4}	{4,6,8}	{2,4,6,8}
		{4,7}	{2,4,7}
2	{1,3}	{5,7}	{1,3,5,7}
		{5,8}	{1,3,5,8}
			-
3	{1,4}	{5,7}	-
			{2,4}
	{5,8}		
			-
4	{4,6,8}	{1,3}	{1,3,6,8}
	{4,7}	-	-

$$\text{logo } F(P_8) = \{ \{1,3,5,7\}, \{1,3,5,8\}, \{1,3,6,8\}, \{1,4,6,8\}, \{1,4,7\}, \\ \{2,4,6,8\}, \{2,4,7\}, \{2,5,7\}, \{2,5,8\} \}$$

$$i(P_8) = 9.$$

O algoritmo 5.1 implementa o procedimento para construir a família  $F$  de c.i.m. de um grafo  $G$  com uma articulação, dadas as famílias de c.i.m. dos subgrafos  $G_1$  e  $G_2$ .



**Algoritmo 5.1: Família de c.i.m de um grafo com uma articulação**

Dados  $G, G_1, G_2$ , a articulação de  $G, F(G_1), F(G_2), F_a^1, F_a^2, \bar{F}_a^1, \bar{F}_a^2$   
 $F = \emptyset$

1. Se  $F_a^1 \neq \emptyset$  e  $F_a^2 \neq \emptyset$  então  
     para todo  $I_1 \in F_a^1$   
         para todo  $I_2 \in F_a^2$   
              $F = F \cup \{ I_1 \cup I_2 \}$
2. Se  $\bar{F}_a^1 \neq \emptyset$  e  $\bar{F}_a^2 \neq \emptyset$  então  
     para todo  $I_1 \in \bar{F}_a^1$   
         para todo  $I_2 \in \bar{F}_a^2$   
              $F = F \cup \{ I_1 \cup I_2 \}$
3. Se  $\bar{F}_a^2 \neq \emptyset$  então  
     para todo  $I_1 \in F_a^1$   
         se  $\exists w \in I_1$  tal que  $\exists v \in \text{Adj}_{G_1}(a)$  com  $(v,w) \in E_1$  então  
             para todo  $I_2 \in \bar{F}_a^2$   
                  $F = F \cup \{ [I_1 - \{a\}] \cup I_2 \}$
4. Se  $\bar{F}_a^1 \neq \emptyset$  então  
     para todo  $I_2 \in F_a^2$   
         se  $\exists w \in I_2$  tal que  $\exists v \in \text{Adj}_{G_2}(a)$  com  $(v,w) \in E_2$  então  
             para todo  $I_1 \in \bar{F}_a^1$   
                  $F = F \cup \{ I_1 \cup [I_2 - \{a\}] \}$ .

**Complexidade:**

Passo 1:  $i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Passo 2:  $i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Passo 3:  $n^2 i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Passo 4:  $n^2 i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Logo a complexidade do algoritmo 5.1 é da ordem  $O( n^2 i(G_1) i(G_2)$  uniões de conjuntos disjuntos), isto é  $O(n^2 i(G_1) i(G_2))$ .

Para o grafo  $G$  da figura 5.2, tem-se:

$$F(G_1) = \{ \{a\}, \{b\}, \{c\} \} \quad i(G_1) = 3$$

$$F(G_2) = \{ \{a\}, \{d\}, \{e\} \} \quad i(G_2) = 3$$

$$F_a^1 = \{ \{a\} \} \quad \bar{F}_a^1 = \{ \{b\}, \{c\} \}$$

$$F_a^2 = \{ \{a\} \} \quad \bar{F}_a^2 = \{ \{d\}, \{e\} \}$$

PASSO	$I_1$	$I_2$	c.i.m.
1	{a}	{a}	{a}
2	{b}	{d}	{b,d}
		{e}	{b,e}
	{c}	{d}	{c,d}
		{e}	{c,e}

logo  $F(G) = \{ \{a\}, \{b,d\}, \{b,e\}, \{c,d\}, \{c,e\} \}$ .

### Número de c.i.m. de G

#### Propriedade 5.2:

Sejam :

$i'(G_1)$  = número de c.i.m. de  $G_1$  que contém a

$i'(G_2)$  = número de c.i.m. de  $G_2$  que contém a

$i''(G_1)$  = número de c.i.m. de  $G_1$  que contém a e tal que a se encontra à distância maior que dois de qualquer outro vértice do c.i.m.

$i''(G_2)$  = número de c.i.m. de  $G_2$  que contém a e tal que a se encontra à distância maior que dois de qualquer outro vértice do c.i.m.

então:

$i(G_1) - i'(G_1)$  = número de c.i.m. de  $G_1$  que não contém a

$i(G_2) - i'(G_2)$  = número de c.i.m. de  $G_2$  que não contém a

logo:

O número de c.i.m. de um grafo  $G$  com uma articulação  $a$  que o separa em  $G_1$  e  $G_2$  é dado pela relação:

$$i(G) = i'(G_1) * i'(G_2) + [i(G_1) - i'(G_1)] * [i(G_2) - i'(G_2)] + i''(G_1) * [i(G_2) - i'(G_2)] + i''(G_2) * [i(G_1) - i'(G_1)].$$

Tem-se:

$$i(G) \leq i(G_1) i(G_2).$$

## 5.2. Grafos com um Conjunto Articulação Clique

Seja  $G(V,E)$  um grafo tal que  $V = V_1 \cup V_2$ ,  $E = E_1 \cup E_2$ ,  $V_1 \cap V_2 = K$  onde  $K$  induz uma clique, isto é:  $G$  obtém-se a partir dos grafos  $G_1(V_1, E_1)$  e  $G_2(V_2, E_2)$  fazendo a identificação dos vértices da clique  $K$ , que é subgrafo de  $G_1$  e de  $G_2$ .  $G$  tem a forma geral exibida na figura 5.7.

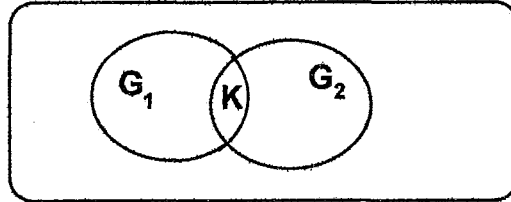


Fig. 5.7: Grafo com um conjunto articulação clique

### Algoritmo de Boulala e Uhry

Boulala e Uhry (1979) generalizam o algoritmo para construir um conjunto independente de peso máximo, em tempo linear no número de vértices. Eles definem:

Para todo  $a \in K$ :  $S_a$  = conjunto independente de peso máximo de  $G_1$  que contém  $a$

$$s_a = \text{card}(S_a) = \text{peso de } S_a$$

$\bar{S}_K$  = conjunto independente de peso máximo do grafo  $G_1$  dentre os que não contém nenhum vértice de  $K$

$$\bar{s}_K = \text{card}(\bar{S}_K) = \text{peso de } \bar{S}_K$$

$S^2$  = conjunto independente de peso máximo de  $G_2$ , considerando em  $G_2$  os vértices de  $K$  com peso  $\alpha_a = s_a - \bar{s}_K \forall a \in K$

$$s^2 = \text{card}(S^2) = \text{peso de } S^2$$

O conjunto independente  $S$  de peso máximo do grafo  $G$  é dado por:

$$\text{se } S^2 \cap K = \{a\} \text{ então } S = S^2 \cup S_a$$

$$\text{se } S^2 \cap K = \emptyset \text{ então } S = S^2 \cup \bar{S}_K$$

$$\text{com peso } s = s^2 + \bar{s}_K.$$

Considere, por exemplo, os grafos  $G_1$  e  $G_2$  da figura 5.8 com  $K = \{b,d,e\}$ . Ao identificar os vértices da clique  $K$  obtém-se o grafo  $G$ .

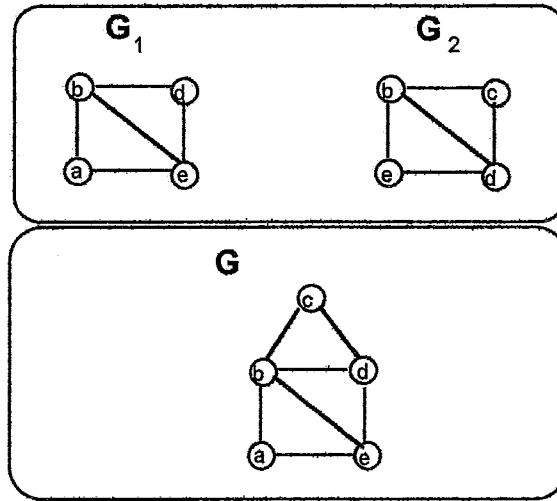


Fig. 5.8: Exemplo de grafo com um conjunto articulação clique

Para o grafo  $G$  da figura 5.8 considerando todos os vértices com peso unitário, tem-se

$$\begin{array}{llll}
 S_b = \{ b \} & s_b = 1 & S_d = \{ d \} & s_d = 1 \\
 S_e = \{ e \} & s_e = 1 & \bar{S}_K = \{ a \} & \bar{s}_K = 1 \\
 S^2 = \{ c, e \} & s^2 = 1 & & 
 \end{array}$$

$S^2 \cap K = \{ e \} \Rightarrow$  um conjunto independente máximo é  $S = \{ c, e \}$  com cardinal  $s = 2$ .

**Algoritmo proposto**

A idéia é construir a família  $F(G)$  de conjuntos independentes maximais de  $G$  a partir das famílias de c.i.m. de  $G_1$  e de  $G_2$ .

**Observação 5.2:**

Dados  $I_1$  um conjunto independente maximal qualquer de  $G_1$  e  $I_2$  um conjunto independente maximal qualquer de  $G_2$ , tem-se as seguintes possibilidades:

- (1)  $I_1 \cap K = I_2 \cap K = \emptyset$  (figura 5.9)  
 logo  $I_1 \cap I_2 = \emptyset$ , e dado que  $Z(u,v) \in E$  para  $u \in V_1 - K$  e  $v \in V_2 - K$  então  $I_1 \cup I_2 \in F(G)$

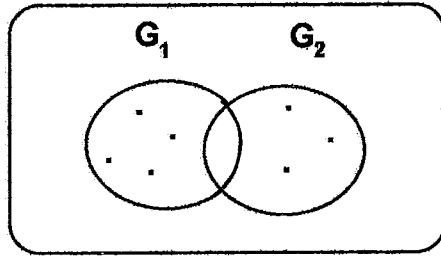


Fig. 5.9: Caso  $I_1 \cap K = I_2 \cap K = \emptyset$

- (2)  $I_1 \cap K = I_2 \cap K = \{a\}$  (figura 5.10)  
 logo  $I_1 \cap I_2 = \{a\}$ , com  $a \in K$  então  $I_1 \cup I_2 \in F(G)$

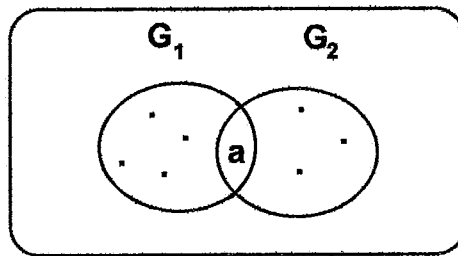


Fig. 5.10: Caso  $I_1 \cap K = I_2 \cap K = \{a\}$

- (3)  $I_1 \cap K = \{a\}$ ,  $I_2 \cap K = \emptyset$  (figura 5.11)  
 logo  $I_1 \cap I_2 = \emptyset$ ,  $\exists (a,v) \in E_2$  com  $v \in V_2 - K$ , caso contrário  $I_2$  não seria maximal  
 então  $I = [I_1 - \{a\}] \cup I_2$  é conjunto independente,  
 para garantir a maximalidade de  $I$  deve  $\exists u \in \text{Adj}_{G_1}(a)$  com  $(u,w) \in E(G_1)$ ,  $w \in I_1$ ;  
 nesse caso  $I = [I_1 - \{a\}] \cup I_2 \in F(G)$

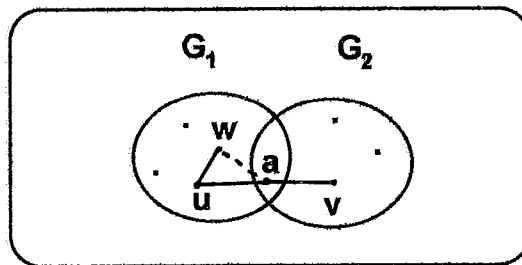


Fig. 5.11: Caso  $I_1 \cap K = \{a\}$ ,  $I_2 \cap K = \emptyset$

- (4)  $I_1 \cap K = \{a\}$ ,  $I_2 \cap K = \{b\}$  com  $a \neq b$   
 logo  $I_1 \cap I_2 = \emptyset$ ,  
 $\exists u \in I_1 - K$  com  $(a,u) \notin E_1$   
 $\exists v \in I_2 - K$  com  $(b,v) \notin E_2$

(4a) se  $(u,b) \in E_1$ ,  $(v,a) \in E_2$  (figura 5.12) então

$I = [I_1 - \{a\}] \cup [I_2 - \{b\}]$  é conjunto independente, para garantir a maximalidade de  $I$  deve  $\exists u \in I_1 - \{a\}$  com  $(u,w) \in E(G_1)$  e  $\exists v \in I_2 - \{b\}$  com  $(v,a) \in E(G_2)$ ; nesse caso  $I = [I_1 - \{a\}] \cup [I_2 - \{b\}] \in F(G)$

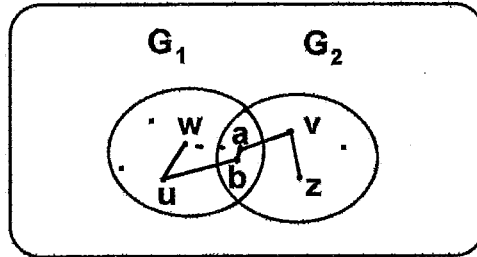


Fig. 5.12: Caso  $I_1 \cap K = \{a\}$ ,  $I_2 \cap K = \{b\}$ ,  $a \neq b$ ,  $(u,b) \in E_1$ ,  $(v,a) \in E_2$

(4b) se  $(u,b) \in E_1$ ,  $(v,a) \notin E_2$  (figura 5.13)

se  $\nexists v' \in I_2$  com  $(v',a) \in E(G_2)$  então  $I_1 \cup [I_2 - \{b\}]$  é c.i.m. de  $G$  mas já apareceu no caso (2) com  $I_1$  e  $\{[I_2 - \{b\}] \cup \{a\}\}$

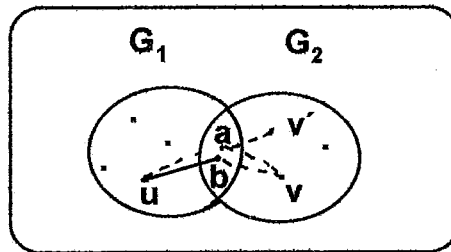


Fig. 5.13: Caso  $I_1 \cap K = \{a\}$ ,  $I_2 \cap K = \{b\}$ ,  $a \neq b$ ,  $(u,b) \in E_1$ ,  $(v,a) \notin E_2$

(4c) se  $(u,b) \notin E_1$ ,  $(v,a) \notin E_2$  (figura 5.14)

se  $\nexists v^* \in I_1$  com  $(v^*,b) \in E(G_1)$  e  $\exists v' \in I_2$  com  $(v',a) \in E(G_2)$  então  $I_1 \cup [I_2 - \{b\}]$  e  $[I_1 - \{a\}] \cup I_2$  são c.i.m. de  $G$  mas já apareceram no caso (2) com  $I_1 \cup I_2'$  e  $I_1' \cup I_2$ , onde  $I_2' = [I_2 - \{b\}] \cup \{a\}$  e  $I_1' = [I_1 - \{a\}] \cup \{b\}$ .

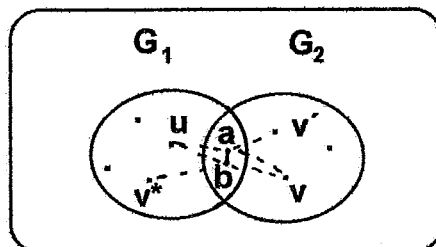


Fig. 5.14: Caso  $I_1 \cap K = \{a\}$ ,  $I_2 \cap K = \{b\}$ ,  $a \neq b$ ,  $(u,b) \notin E_1$ ,  $(v,a) \notin E_2$ .

A observação 5.2 é a justificativa da propriedade 5.3 enunciada abaixo, a qual a su vez é a base do procedimento descrito no algoritmo 5.2.

**Propriedade 5.3:**

Dadas as famílias  $F(G_1)$  e  $F(G_2)$  de c.i.m. de  $G_1$  e  $G_2$ , para  $i= 1, 2$  sejam:

$\bar{F}_K^i$  = família de c.i.m. de  $G_i$  que não contém nenhum elemento de  $K$

$\forall a \in K: F_a^i$  = família de c.i.m. de  $G_i$  que contém o vértice  $a$  de  $K$

então  $F(G)$  é construída segundo o seguinte procedimento:

1.- Para todo  $I_1 \in \bar{F}_K^1$ , para todo  $I_2 \in \bar{F}_K^2$ :  $I_1 \cup I_2 \in F(G)$

2.- Para todo  $a \in K$ , para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in F_a^2$ :  $I_1 \cup I_2 \in F(G)$

3.- Para todo  $a \in K$ , para todo  $I_1 \in F_a^1$ ,  
se  $\exists w \in I_1$  tal que  $\exists v \in \text{Adj}_{G_1}(a)$  com  $(v,w) \in E(G_1)$  então  
para todo  $I_2 \in \bar{F}_K^2$ :  $[I_1 - \{a\}] \cup I_2 \in F(G)$

Para todo  $a \in K$ , para todo  $I_2 \in F_a^2$ ,  
se  $\exists w \in I_2$  tal que  $\exists v \in \text{Adj}_{G_2}(a)$  com  $(v,w) \in E(G_2)$  então  
para todo  $I_1 \in \bar{F}_K^1$ :  $I_1 \cup [I_2 - \{a\}] \in F(G)$

4.- Para todo  $a \in K$ , para todo  $b \in K - \{a\}$   
para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in F_b^2$   
se  $\exists u \in I_1 - \{a\}$  com  $(u,w) \in E(G_1)$  e  $\exists v \in I_2 - \{b\}$  com  $(v,a) \in E(G_2)$  então  
 $I = [I_1 - \{a\}] \cup [I_2 - \{b\}] \in F(G)$ .

**Observação 5.3:**

Um c.i.m.  $I \in F(G)$  gerado no passo 4 pode ser obtido a partir de um c.i.m.  $I_1 \in F_a^1$  e dois c.i.m. diferentes  $I_2 \in F_b^2$  e  $I_2' \in F_c^2$  com  $b, c \in K - \{a\}$ , tais que

$$I_2' - \{b\} = I_2 - \{c\}.$$

Para evitar incorporar c.i.m. com duplicatas em  $F(G)$ , no passo 4, pode-se comparar  $I$  com todos os elementos já incorporados a  $F(G)$ . Um método mais eficiente para evitar duplicatas em vez de comparar  $I$  com todos os elementos já incorporados a  $F(G)$  é,  $I$  é introduzido em  $F(G)$  se e somente se ele é obtido do menor lexicográfico  $I_2 \in F_b^2$  dentre todos os c.i.m.  $I_2 \in F_b^2$ ,  $I_2' \in F_c^2$  tais que  $I_2 - \{b\} = I_2' - \{c\} \forall b, c \in K - \{a\}$ , para cada  $a \in K$ . Para isto basta considerar os elementos  $b, c \in K - \{a\}$  em ordem lexicográfica e verificar que para todo  $c <_L b$  tem-se  $I_2 - \{b\} \neq I_2' - \{c\}$ .

O algoritmo 5.2 implementa o procedimento para construir a família de c.i.m. de um grafo  $G$  com um conjunto articulação clique, dadas as famílias de c.i.m. dos subgrafos  $G_1$  e  $G_2$ .

**Algoritmo 5.2: Família de c.i.m. de um grafo com conjunto articulação clique**

Dados  $G_1, G_2, F(G_1), F(G_2), \bar{F}_K^1, \bar{F}_K^2, F_a^1, F_a^2 \forall a \in K$

$F = \emptyset$

1. Se  $\bar{F}_K^1 \neq \emptyset$  e  $\bar{F}_K^2 \neq \emptyset$  então

para todo  $I_1 \in \bar{F}_K^1$

para todo  $I_2 \in \bar{F}_K^2$

$F = F \cup \{ I_1 \cup I_2 \}$

2. Para todo  $a \in K$

se  $F_a^1 \neq \emptyset$  y  $F_a^2 \neq \emptyset$  então

para todo  $I_1 \in F_a^1$

para todo  $I_2 \in F_a^2$

$F = F \cup \{ I_1 \cup I_2 \}$

3. Se  $\bar{F}_K^2 \neq \emptyset$  então

para todo  $a \in K$

para todo  $I_1 \in F_a^1$

se  $\exists w \in I_1$  tal que  $\exists v \in \text{Adj}_{G_1}(a)$  com  $(v,w) \in E(G_1)$  então

para todo  $I_2 \in \bar{F}_K^2$

$F = F \cup \{ [I_1 - \{a\}] \cup I_2 \}$

Se  $\bar{F}_K^1 \neq \emptyset$  então

para todo  $a \in K$

para todo  $I_2 \in F_a^2$

se  $\exists w \in I_2$  tal que  $\exists v \in \text{Adj}_{G_2}(a)$  com  $(v,w) \in E(G_2)$  então

para todo  $I_1 \in \bar{F}_K^1$

$F = F \cup \{ I_1 \cup [I_2 - \{a\}] \}$

4. Para todo  $a \in K$

para todo  $b \in K - \{a\}$  escolhidos em ordem lexicográfica

para todo  $I_1 \in F_a^1$

para todo  $I_2 \in F_b^2$

se para todo  $c <_L b$  tem-se  $I_2 - \{b\} \neq I_2' - \{c\}$  então

se  $\exists u \in I_1 - \{a\}$  com  $(u,b) \in E_1$  e  $\exists v \in I_2 - \{b\}$  com  $(v,a) \in E_2$

então  $F = F \cup \{ [I_1 - \{a\}] \cup [I_2 - \{b\}] \}$ .

**Complexidade:**

Passo 1:  $i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Passo 2:  $i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Passo 3:  $n^3 i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Passo 4:  $n^4 i(G_1) i(G_2)$  uniões de conjuntos disjuntos

Logo a complexidade do algoritmo 5.2 é da ordem  $O(n^4 i(G_1) i(G_2))$ .



Por exemplo, para o grafo  $G$  da figura 5.8 tem-se:

$$K = \{b, d, e\} \quad \bar{F}_K^1 = \bar{F}_K^2 = \emptyset$$

$$F_b^1 = \{ \{b\} \}, \quad F_d^1 = \{ \{a, d\} \}, \quad F_e^1 = \{ \{e\} \}$$

$$F_b^2 = \{ \{b\} \}, \quad F_d^2 = \{ \{d\} \}, \quad F_e^2 = \{ \{c, e\} \}$$

aplicando o algoritmo 5.2:

Passo 2:

a	$I_1$	$I_2$	c.i.m.
b	{b}	{b}	{b}
d	{a,d}	{d}	{a,d}
e	{e}	{c,e}	{c,e}

Passo 4:

a	b	$I_1$	$I_2$	c.i.m.
d	e	{a,d}	{c,e}	{a,c}

logo:

$$F(G) = \{ \{b\}, \{a,d\}, \{c,e\}, \{a,c\} \}$$

$$i(G) = 4.$$

Para o grafo  $G_G$  da figura 5.15 com  $K = \{b, c, g\}$ , tem-se a decomposição nos grafos  $G_1$  e  $G_2$ .

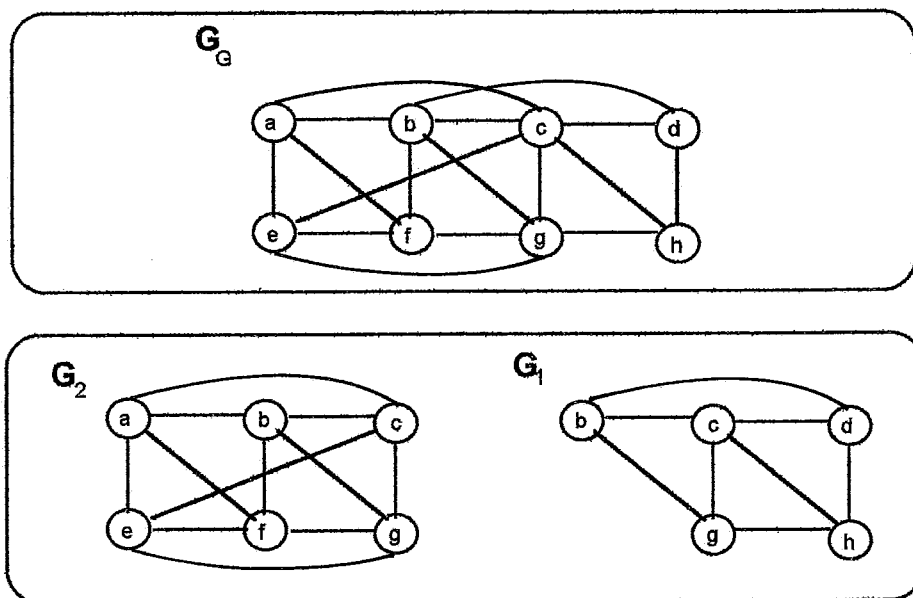


Fig. 5.15: Exemplo do Gavril

Tem-se:

$$F(G_1) = \{ \{c\}, \{g,d\}, \{b,h\} \},$$

$$F(G_2) = \{ \{a,g\}, \{b,e\}, \{c,f\} \},$$

$$\bar{F}_K^1 = \bar{F}_K^2 = \emptyset$$

$$F_b^1 = \{ \{b,h\} \}, \quad F_c^1 = \{ \{c\} \}, \quad F_g^1 = \{ \{g,d\} \}$$

$$F_b^2 = \{ \{b,e\} \}, \quad F_c^2 = \{ \{c,f\} \}, \quad F_g^2 = \{ \{a,g\} \}$$

aplicando o algoritmo 5.2:

Passo 2:

a	I <sub>1</sub>	I <sub>2</sub>	c.i.m.
b	{b,h}	{b,e}	{b,e,h}
c	{c}	{c,f}	{c,f}
g	{g,d}	{a,g}	{a,d,g}

Passo 4:

a	b	I <sub>1</sub>	I <sub>2</sub>	c.i.m.
b	c	{b,h}	{c,f}	{f,h}
b	g	{b,h}	{a,g}	{a,h}
g	b	{g,d}	{b,e}	{d,e}
g	c	{g,d}	{c,f}	{d,f}

logo:

$$F(G_G) = \{ \{b,e,h\}, \{c,f\}, \{a,d,g\}, \{f,h\}, \{a,h\}, \{d,e\}, \{d,f\} \}$$

$$i(G_G) = 7.$$

### Número de c.i.m. de G

#### Propriedade 5.4:

O número de c.i.m. de um grafo G com um conjunto articulação clique K que o separa em G<sub>1</sub> e G<sub>2</sub> está limitado segundo a relação:

$$i(G) \leq i(G_1) + i(G_2).$$

### 5.3. Grafos Separáveis por Cliques

Um grafo  $G(V,E)$  é separável por cliques se satisfaz uma das três condições descritas abaixo:

- 1)  $G$  é tipo 1:  $V$  pode ser particionado em dois conjuntos  
 $V = V_1 + V_2$  com  $V_1 \cup V_2 = V$ ,  $V_1 \cap V_2 = \emptyset$   
 tal que:  $G_{V_1}$  é um grafo bipartido conexo com  $\text{card}(V_1) \geq 3$ ,  
 $G_{V_2}$  é uma clique,  
 em  $G$ , todo vértice de  $V_1$  é adjacente a todo vértice de  $V_2$ .
- 2)  $G$  é tipo 2:  $G$  é um grafo  $k$ -partido completo, para algum  $k$ , isto é:  
 $V = V_1 + V_2 + \dots + V_k$  com  $V_i \cap V_j = \emptyset \quad \forall i \neq j$ , todo  $V_i$  é um conjunto independente e  $\forall i \neq j$  todo vértice de  $V_i$  é adjacente a todo vértice de  $V_j$ .
- 3)  $G$  tem um corte de vértices  $K$  que é uma clique, isto é:  $G - K$  é desconexo e os subgrafos  $G_i$  induzidos por  $C_i \cup K$  são separáveis por cliques, onde  $C_i$  com  $i = 1, \dots, k$  são as componentes conexas de  $G - K$ .

Por exemplo, na figura 5.16 apresenta-se o grafo  $G$  que é separável por cliques.

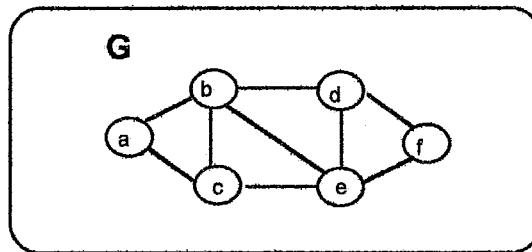


Fig. 5.16: Grafo separável por cliques

#### Árvore de decomposição por cliques

Dado  $G(V,E)$  conexo, a árvore de decomposição por cliques  $T_G$  é uma árvore com raiz  $G$ . Cada nó  $H$  de  $T_G$ , representante do grafo  $H$  com um corte de cliques  $K$  (isto é:  $K$  é clique de  $H$  e  $H - K$  é desconexo) tem como filhos nós associados com os subgrafos  $H_i$  induzidos por  $C_i \cup K$ , onde os  $C_i$  são as componentes conexas de  $H - K$ . As folhas de  $T_G$  são grafos primitivos tipo 1 ou tipo 2.

Na figura 5.17 apresentam-se: os subgrafos  $G_1$ ,  $G_2$ ,  $G_3$  e  $G_4$  obtidos ao decompor o grafo  $G$  da figura 5.16; com a clique  $K = \{b,c,e\}$  de  $G$ , tem-se o subgrafo  $G_1$  induzido por  $\{a,b,c,e\}$  é tipo 1, o subgrafo  $G_2$  induzido por  $\{b,c,d,e,f\}$  é separável por cliques com  $K_2 = \{b,d,e\}$  clique maximal de  $G_2$ ,  $G_3$  induzido por  $\{b,c,d,e\}$  é tipo 1,  $G_4$  induzido por  $\{b,d,e,f\}$  é tipo 2, e a árvore de decomposição por cliques  $T_G$  do grafo  $G$ .

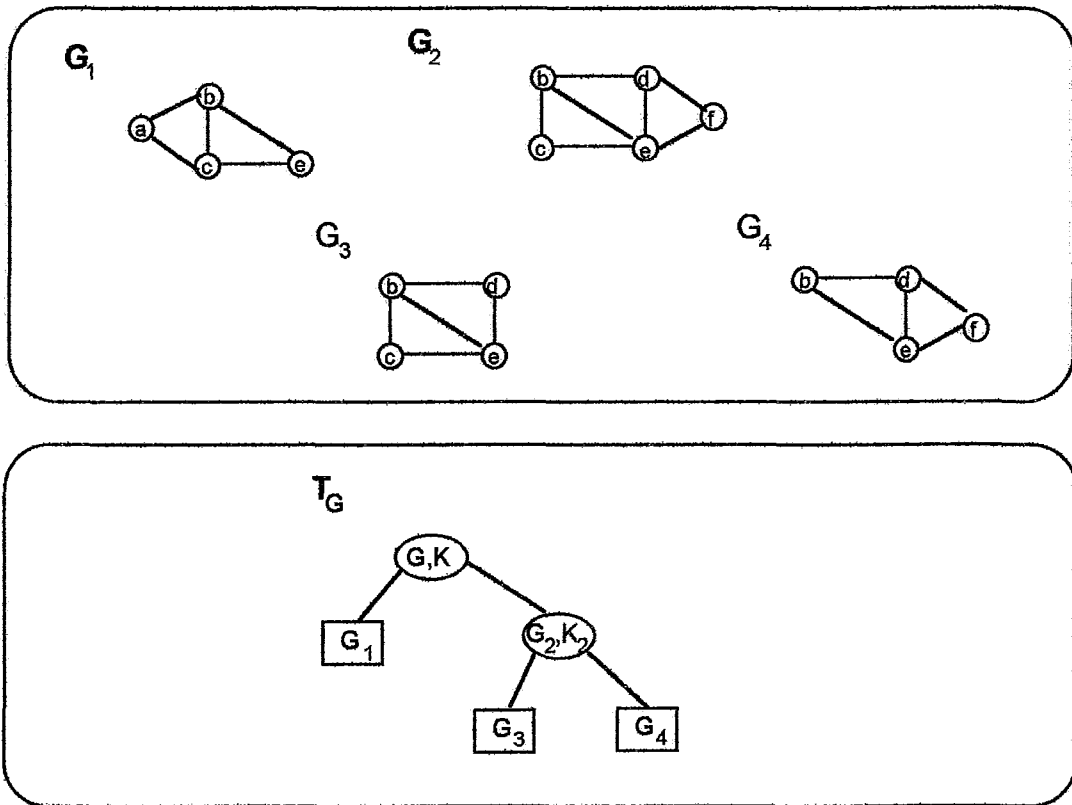


Fig. 5.17: Árvore de decomposição por cliques

Gavril (1977) desenvolve um algoritmo de reconhecimento de um grafo separável por cliques, no qual constrói a árvore de decomposição por cliques, em tempo  $O(n^5)$ , e prova que esta árvore tem no máximo  $n^2$  vértices.

Whitesides (1984) apresenta um algoritmo de reconhecimento utilizando um algoritmo para determinar se um grafo dado possui um corte clique (Whitesides, 1983) e construindo a árvore de decomposição por cliques, em tempo  $O(n^5)$ . Se um grafo não possui um corte clique então deve ser um grafo primitivo tipo 1 ou tipo 2. Ela, também, determina um conjunto independente de peso máximo. Para um grafo tipo 1 faz referência ao algoritmo de Burlet e Uhry (1984) para os grafos de paridade (grafos tais que todo ciclo ímpar tem duas cordas que se cruzam) família à qual pertencem os grafos bipartidos. Para um grafo com um corte clique utiliza o algoritmo de Boulala e Uhry (1979).

### Algoritmo proposto

Para construir a família  $F(G)$  de conjuntos independentes maximais de um grafo  $G$  separável por cliques utiliza-se a árvore de decomposição por cliques  $T_G$  do grafo  $G$ .

Cada nó  $H$  da árvore  $T_G$  tem filhos  $H_1, H_2, \dots, H_k$  com  $H_1$  induzido por  $C_1 \cup K$ ,  $k$

o número de componentes conexas de  $G$  e  $K$  um corte clique de  $G$ . Se opera com o algoritmo 5.2 para construir a família de c.i.m. de um grafo  $H$  com um conjunto articulação  $K$  que é clique.

O processo começa considerando os grafos primitivos localizados nas folhas da árvore  $T_G$ , como os grafos  $H_i$ . Sube-se nesta árvore, aplicando o algoritmo 5.2, até a raiz dela que corresponde ao grafo de interesse.

O algoritmo 5.3 implementa o procedimento para construir a família de c.i.m. de um grafo  $G$  separável por cliques, dadas a árvore de decomposição por cliques  $T_G$  e as famílias de c.i.m. dos subgrafos primitivos gerados na decomposição utilizando o algoritmo 5.2 para gerar a família de c.i.m. de um grafo construído pela identificação de uma clique comum em dois grafos

**Algoritmo 5.3: Família de c.i.m de um grafo separável por cliques**

Dados  $G$  e a árvore de decomposição por cliques  $T_G$

1. Determinar a altura  $h$  da árvore  $T_G$
2. Para todo grafo primitivo  $G_p$  associado às folhas de  $T_G$   
construir a família  $F(G_p)$  de c.i.m. de  $G_p$
3. Para  $j = 1$  até  $h-1$  efetuar  
para todo nó  $G_j$  no nível  $h - j$  de  $T_G$  efetuar  
determinar a família  $F(G_j)$  do grafo associado ao nó  $G_j$ .

**Complexidade:**

Passo 1:  $O(n)$

Passo 2: é executado  $O(n)$  com tempo  $O(n)$

Passo 3: é executado  $O(n^2)$  vezes pois  $T_G$  tem no máximo  $n^2$  vértices, pelo algoritmo 5.2 com tempo  $O(n^4 i(G_1) i(G_2))$ , dado que  $G_1$  e  $G_2$  são os filhos de  $G_j$  em  $T_G$ .

A complexidade total do algoritmo 5.3 é  $O(n^6 i(G_1) i(G_2))$ .

Por exemplo, para o grafo  $G$  da figura 5.16 aplicando o algoritmo 5.3 tem-se:

Passo 1:

altura  $h = 3$

Passo 2:

grafos primitivos  $G_1, G_3$  e  $G_4$

$F(G_3) = \{ \{b\}, \{c,d\}, \{e\} \}$      $F(G_4) = \{ \{b,f\}, \{d\}, \{e\} \}$

Passo 3:

usando o algoritmo 5.2 obtém-se  $F(G_2) = \{ \{b,f\}, \{c,d\}, \{c,f\}, \{e\} \}$

por outra parte  $F(G_1) = \{ \{a,e\}, \{b\}, \{c\} \}$

novamente com o algoritmo 5.2,  $F(G) = \{\{a,d\},\{a,e\},\{a,f\},\{b,f\},\{c,d\},\{c,f\}\}$ .

Para o grafo  $G_G$  da figura 5.15, exemplo do Gavril, com  $K = \{b,c,g\}$ , tem-se a decomposição nos grafos  $G_1$  que é tipo 1 e  $G_2$  que é tipo 2, exibidos na mesma figura. A árvore  $T_G$  é exibida na figura 5.18.

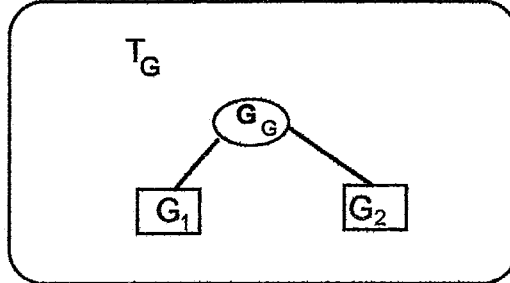


Fig. 5.18: Árvore de decomposição por cliques do exemplo do Gavril

Passo 1:

altura  $h = 2$

Passo 2:

grafos primitivos  $G_1$  e  $G_2$

$$F(G_1) = \{\{c\},\{g,d\},\{b,h}\} \quad F(G_2) = \{\{a,g\},\{b,e\},\{c,f\}\}$$

Passo 3:

usando o algoritmo 5.2 obtém-se

$$F(G_G) = \{\{a,d,g\},\{a,h\},\{b,e,h\},\{d,e\},\{c,f\},\{d,f\},\{f,h}\}.$$

Aplicando os algoritmos 5.2 e 5.3 ao exemplo apresentado por Leung, grafo  $G_L$  da figura 4.25 que é de intervalo e portanto é separável por cliques, com a árvore de decomposição por cliques da figura 5.19,  $K = \{4,5,6\}$  decompoe  $G_L$  nos grafos  $G_1$  induzido por  $\{1,2,3,4,5,6\}$  e  $G_2$  induzido por  $\{4,5,6,7,8\}$  que é tipo 1,  $K_1 = \{2,4,5\}$  decompoe  $G_1$  nos grafos  $G_3$  induzido por  $\{1,2,3,4,5\}$  que é tipo 1 e  $G_4$  induzido por  $\{2,4,5,6\}$  que é tipo 2, exibidos na figura 5.20; tem-se:

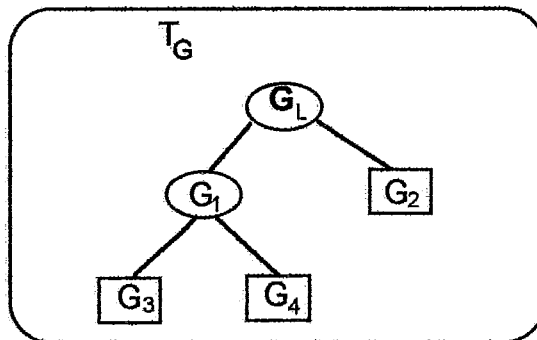


Fig. 5.19: Árvore de decomposição por cliques do exemplo do Leung

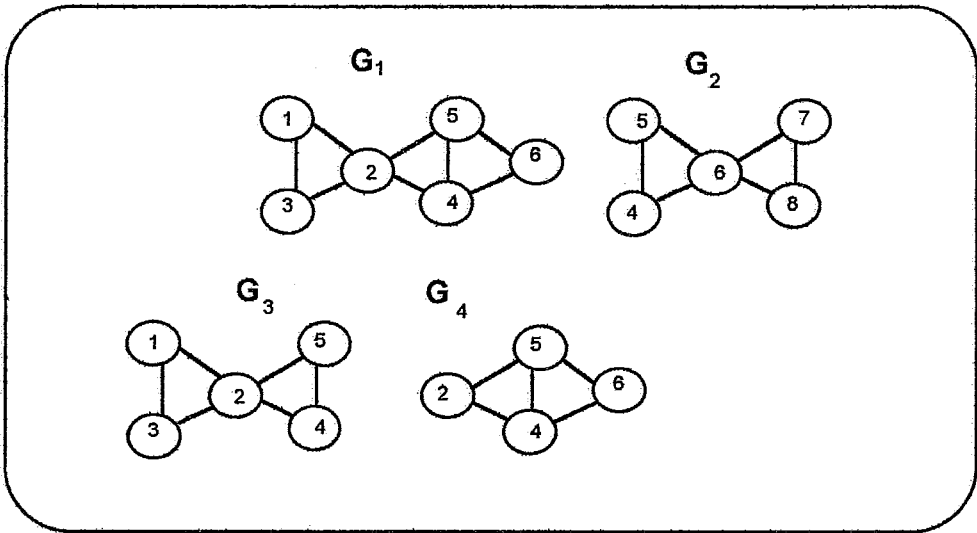


Fig. 5.20: Grafos parciais no exemplo de Leung

Passo 1:  
altura  $h = 3$

Passo 2:  
grafos primitivos  $G_2, G_3$  e  $G_4$   
 $F(G_2) = \{ \{6\}, \{4,7\}, \{4,8\}, \{5,7\}, \{5,8\} \}$ ,  
 $F(G_3) = \{ \{2\}, \{1,4\}, \{1,5\}, \{3,4\}, \{3,5\} \}$ ,  
 $F(G_4) = \{ \{2,6\}, \{4\}, \{5\} \}$

Passo 3:  
para  $G_1$  dadas  $F(G_3)$  e  $F(G_4)$ ,  $K_1 = \{2,4,5\}$ , usando o algoritmo 5.2 obtém-se

$$\bar{F}_K^3 = \emptyset, \quad F_2^3 = \{ \{2\} \}, \quad F_4^3 = \{ \{1,4\}, \{3,4\} \}, \quad F_5^3 = \{ \{1,5\}, \{3,5\} \}$$

$$\bar{F}_K^4 = \emptyset, \quad F_2^4 = \{ \{2,6\} \}, \quad F_4^4 = \{ \{4\} \}, \quad F_5^4 = \{ \{5\} \}$$

passo 2:

a	$I_1$	$I_2$	c.i.m.
2	{2}	{2,6}	{2,6}
4	{1,4}	{4}	{1,4}
	{3,4}	{4}	{3,4}
5	{1,5}	{5}	{1,5}
	{3,5}	{5}	{3,5}

passo 4:

a	b	$I_1$	$I_2$	c.i.m.
2	4	{2}	{4}	-
	5	{2}	{5}	-
4	2	{1,4}	{2,6}	{1,6}
		{3,4}	{2,6}	{3,6}
	5	{1,4}	{5}	-
		{3,4}	{5}	-
5	2	{1,5}	{2,6}	{1,6} *
		{3,5}	{2,6}	{3,6} *
	4	{1,5}	{4}	-
		{3,5}	{4}	-

(\*) estes c.i.m. não são considerados pois são gerados no caso  $a = 4$ .

logo:

$$F(G_1) = \{ \{2,6\}, \{1,4\}, \{3,4\}, \{1,5\}, \{3,5\}, \{1,6\}, \{3,6\} \}$$

Por outra parte:  $F(G_2) = \{ \{6\}, \{4,7\}, \{4,8\}, \{5,7\}, \{5,8\} \}$ ,

Passo 3:

para  $G_L$  dadas  $F(G_1)$  e  $F(G_2)$ ,  $K = \{4,5,6\}$ , usando o algoritmo 5.2 obtém-se

$$\bar{F}_K^1 = \emptyset, F_4^1 = \{ \{1,4\}, \{3,4\} \}, F_5^1 = \{ \{1,5\}, \{3,5\} \}, F_6^1 = \{ \{1,6\}, \{2,6\}, \{3,6\} \}$$

$$\bar{F}_K^2 = \emptyset, F_4^2 = \{ \{4,7\}, \{4,8\} \}, F_5^2 = \{ \{5,7\}, \{5,8\} \}, F_6^2 = \{ \{6\} \}$$

passo 2:

a	$I_1$	$I_2$	c.i.m.
4	{1,4}	{4,7}	{1,4,7}
		{4,8}	{1,4,8}
	{3,4}	{4,7}	{3,4,7}
		{4,8}	{3,4,8}
5	{1,5}	{5,7}	{1,5,7}
		{5,8}	{1,5,8}
	{3,5}	{5,7}	{3,5,7}
		{5,8}	{3,5,8}
6	{1,6}	{6}	{1,6}
	{2,6}	{6}	{2,6}
	{3,6}	{6}	{3,6}



passo 4:

a	b	$I_1$	$I_2$	c.i.m.
4	5	{1,4}	{5,7}	-
			{5,8}	-
		{3,4}	{5,7}	-
		{5,8}	-	
	6	{1,4}	{6}	-
		{3,4}	{6}	-
5	4	{1,5}	{4,7}	-
			{4,8}	-
		{3,5}	{4,7}	-
		{4,8}	-	
	6	{1,5}	{6}	-
		{3,5}	{6}	-
6	4	{1,6}	{4,7}	{1,7}
			{4,8}	{1,8}
		{2,6}	{4,7}	{2,7}
			{4,8}	{2,8}
		{3,6}	{4,7}	-
			{4,8}	-
	5	{1,6}	{5,7}	- *
			{5,8}	- *
		{2,6}	{5,7}	- *
			{5,8}	- *
		{3,6}	{5,7}	-
			{5,8}	-

(\*) estes c.i.m. não são considerados pois são gerados no caso  $a = 5$ .

logo:

$$F(G_L) = \{ \{1,6\}, \{1,4,7\}, \{1,4,8\}, \{1,5,7\}, \{1,5,8\}, \{2,6\}, \{2,7\}, \{2,8\}, \{3,6\}, \{3,4,7\}, \{3,4,8\}, \{3,5,7\}, \{3,5,8\} \}.$$

### 5.4. Grafos com conexão paralela

O grafo  $G = G_1 \parallel G_2$  obtido por conexão paralela dos grafos  $G_1(V_1, E_1, v_{11}, v_{12})$  e  $G_2(V_2, E_2, v_{21}, v_{22})$  com  $V_1 \cap V_2 = \emptyset$ , onde  $v_{11}$  e  $v_{12}$  são os vértices terminais de  $G_1$ ,  $v_{21}$  e  $v_{22}$  são os vértices terminais de  $G_2$ , identificando os vértices  $v_{11}$  com  $v_{21}$  e  $v_{12}$  com  $v_{22}$ , é dado por:  $V_G = V_1 \cup V_2$ ,  $E_G = E_1 \cup E_2$ , com vértices terminais  $v_1$  e  $v_2$  ( $v_1 = v_{11} = v_{21}$ ,  $v_2 = v_{12} = v_{22}$ ), exibido na figura 5.21.

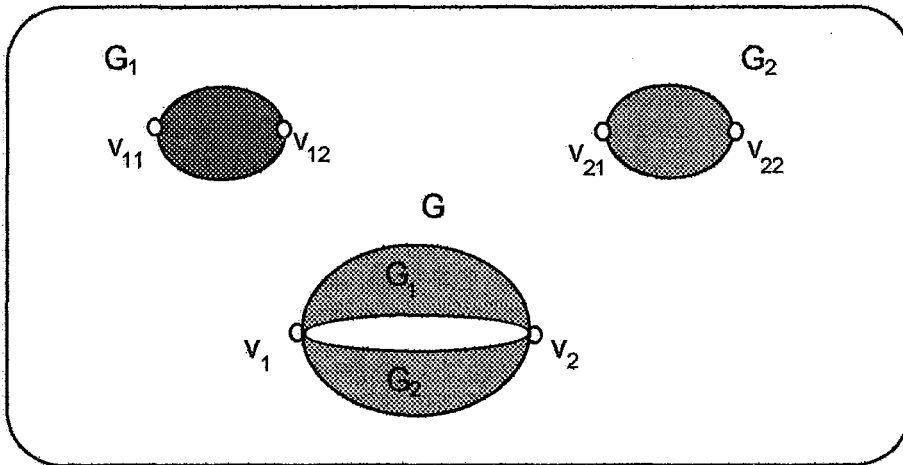


Fig. 5.21: Conexão paralela

### Algoritmo de Boulala e Uhry

Boulala e Uhry (1979) desenvolvem um algoritmo, de tempo linear no número de vértices, para construir um conjunto independente de peso máximo de um grafo  $G$  obtido por conexão paralela de outros dois grafos  $G_1$  e  $G_2$ .

Dados  $G_1(V_1, E_1, v_{11}, v_{12})$  e  $G_2(V_2, E_2, v_{21}, v_{22})$  com  $V_1 \cap V_2 = \emptyset$ , onde  $v_{11}$  e  $v_{12}$ , são os vértices terminais de  $G_1$ ,  $v_{21}$  e  $v_{22}$  são os vértices terminais de  $G_2$ , com pesos  $\text{peso}_1(v)$  e  $\text{peso}_2(v)$  do vértice  $v$  em  $G_1$  e  $G_2$ , respectivamente.

$G = G_1 \parallel G_2$  é dado por:  $V_G = V_1 \cup V_2$ ,  $E_G = E_1 \cup E_2$ , com vértices terminais  $a$  e  $b$  ( $a = v_{11} = v_{21}$ ,  $b = v_{12} = v_{22}$ ).

Eles definem:

$S_{ab}^1 =$  conjunto independente de peso máximo do grafo  $G_1$  dentre os que contém  $a$  e  $b$

$s_{ab}^1 = \text{card}(S_{ab}^1) =$  peso de  $S_{ab}^1$

$\bar{S}_{ab}^1$  = conjunto independente de peso máximo do grafo  $G_1$  dentre os que não contém **a** nem **b**

$\bar{s}_{ab}^1 = \text{card}(\bar{S}_{ab}^1) = \text{peso de } \bar{S}_{ab}^1$

$S_a^1$  = conjunto independente de peso máximo do grafo  $G_1$  dentre os que contém **a** mas não contém **b**

$s_a^1 = \text{card}(S_a^1) = \text{peso de } S_a^1$

$S_b^1$  = conjunto independente de peso máximo do grafo  $G_1$  dentre os que contém **b** mas não contém **a**

$s_b^1 = \text{card}(S_b^1) = \text{peso de } S_b^1$

$S^*$  = conjunto independente de peso máximo do grafo  $G^*$

$s^* = \text{card}(S^*) = \text{peso de } S^*$ .

$S^2$  = restrição de  $S^*$  a  $G_2$

$s^2 = \text{card}(S^2) = \text{peso de } S^2$ .

O conjunto independente  $S$  de peso máximo do grafo  $G$  é dado por:

se  $a \in S^2$  e  $b \in S^2$  então  $S = S^2 \cup S_{ab}^1$

se  $a \notin S^2$  e  $b \notin S^2$  então  $S = S^2 \cup \bar{S}_{ab}^1$

se  $a \in S^2$  e  $b \notin S^2$  então  $S = S^2 \cup S_a^1$

se  $a \notin S^2$  e  $b \in S^2$  então  $S = S^2 \cup S_b^1$

com peso  $s = s^* + \sigma$ .

O grafo  $G^*$  é construído dependendo das conexões existentes em  $G_1$  e  $G_2$ , segundo os casos seguintes, considerando  $\alpha_0 = s_{ab}^1 + \bar{s}_{ab}^1 - (s_a^1 + s_b^1)$ :

1) Se  $(a,b) \in E_1$  (figura 5.22) então

$G^*(V^*, E^*)$  com  $V^* = V_2$

$E^* = E_2 \cup \{(a,b)\}$

$\alpha_v^* = \text{peso}_*(v) = \text{peso}_2(v) \quad \forall v \neq a, \forall v \neq b$

$\alpha_a^* = \text{peso}_*(a) = s_a^1 - \bar{s}_{ab}^1$

$\alpha_b^* = \text{peso}_*(b) = s_b^1 - \bar{s}_{ab}^1$

$$\sigma = \overline{s_{ab}}^1$$

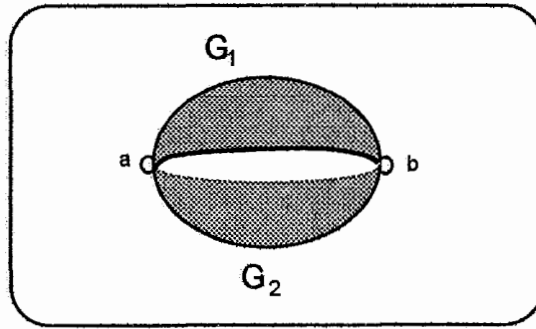


Fig. 5.22: Caso  $(a,b) \in E_1$

2) Se  $\alpha_0 > 0$  (figura 5.23) então

$$\mathbf{G}^*(\mathbf{V}^*, \mathbf{E}^*) \text{ com } \mathbf{V}^* = \mathbf{V}_2 \cup \{c\}, c \in \mathbf{V}_1, c \notin \mathbf{V}_2$$

$$\mathbf{E}^* = \mathbf{E}_2 \cup \{(a,c), (c,b)\}$$

$$\alpha_v^* = \text{peso}_*(v) = \text{peso}_2(v) \quad \forall v \in \mathbf{V}_2 - \{a, b\}$$

$$\alpha_a^* = \text{peso}_*(a) = s_{ab}^1 - s_b^1$$

$$\alpha_b^* = \text{peso}_*(b) = s_{ab}^1 - s_a^1$$

$$\alpha_c^* = \text{peso}_*(c) = \alpha_0$$

$$\sigma = \overline{s_{ab}}^1 - \alpha_0 = s_a^1 + s_b^1 - s_{ab}^1$$

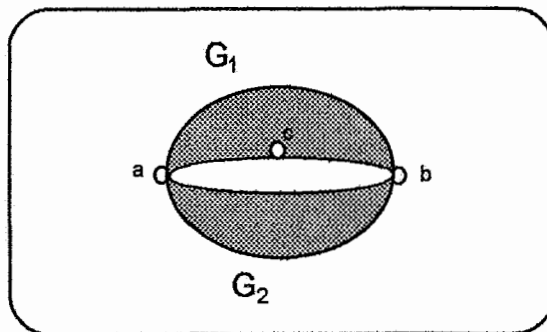


Fig. 5.23: Caso  $\alpha_0 > 0$

3) Se  $\alpha_0 < 0$  (figura 5.24) então

$$\mathbf{G}^*(\mathbf{V}^*, \mathbf{E}^*) \text{ com } \mathbf{V}^* = \mathbf{V}_2 \cup \{c,d\}, c,d \in \mathbf{V}_1, c,d \notin \mathbf{V}_2$$

$$\mathbf{E}^* = \mathbf{E}_2 \cup \{(a,c), (c,d), (d,b)\}$$

$$\alpha_v^* = \text{peso}_*(v) = \text{peso}_2(v) \quad \forall v \in \mathbf{V}_2 - \{a, b\}$$

$$\alpha_a^* = \text{peso}^*(\mathbf{a}) = \mathbf{s}_a^1 - \overline{\mathbf{s}}_{ab}^1$$

$$\alpha_b^* = \text{peso}^*(\mathbf{b}) = \mathbf{s}_b^1 - \overline{\mathbf{s}}_{ab}^1$$

$$\alpha_c^* = \text{peso}^*(\mathbf{c}) = -\alpha_0$$

$$\alpha_d^* = \text{peso}^*(\mathbf{d}) = -\alpha_0$$

$$\sigma = \overline{\mathbf{s}}_{ab}^1 + \alpha_0 = 2\overline{\mathbf{s}}_{ab}^1 + \mathbf{s}_{ab}^1 - (\mathbf{s}_a^1 + \mathbf{s}_b^1)$$

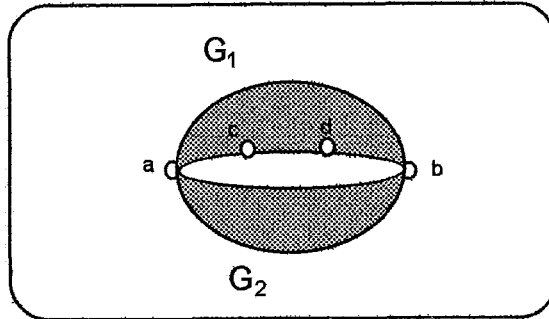


Fig. 5.24: Caso  $\alpha_0 < 0$

4) Se  $\alpha_0 = 0$  então pode assimila-se ao caso 2 ou ao 3.

Por exemplo, para o grafo  $G$  da figura 5.25 com os grafos  $G_1$  e  $G_2$  da mesma figura, tem-se:

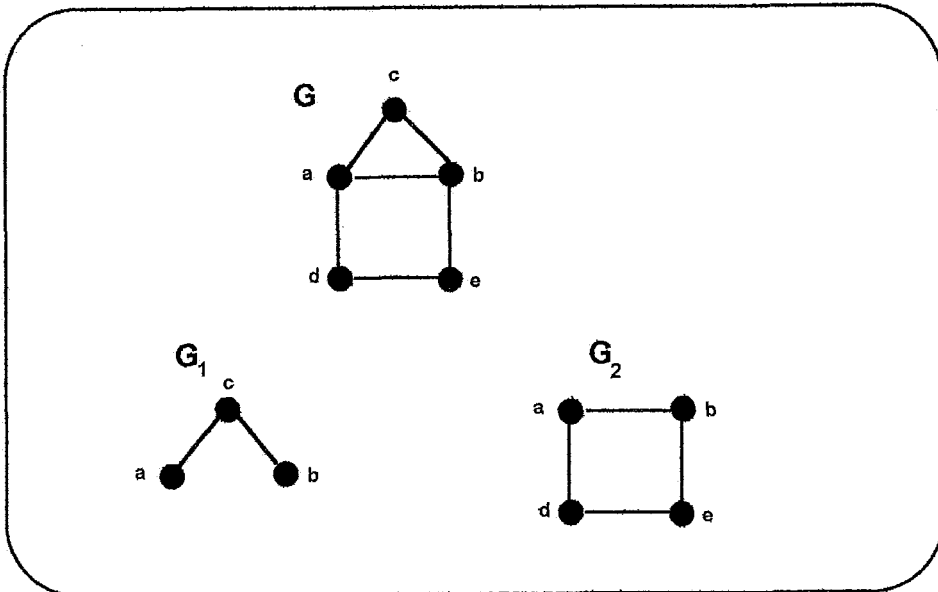


Fig. 5.25: Exemplo do algoritmo de Boulala e Uhry

$$S_{ab}^1 = \{ \{a,b\} \} \quad s_{ab}^1 = 2$$

$$\overline{S}_{ab}^1 = \{ \{c\} \} \quad \overline{s}_{ab}^1 = 1$$

$$S_a^1 = \emptyset \quad s_a^1 = 0$$

$$S_b^1 = \emptyset \quad s_b^1 = 0$$

$$(a,b) \notin E(G_1)$$

$\alpha_0 = 3$ , o grafo  $G^*$  é construído segundo o caso 2 e é exibido na figura 5.26, com os pesos dos vértices indicados do lado deles.

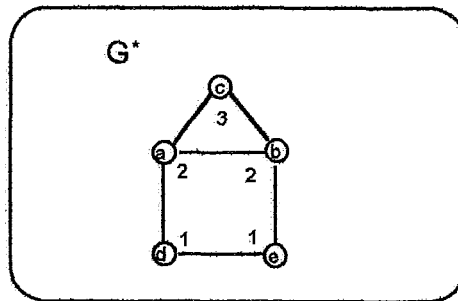


Fig. 5.26: Grafo  $G^*$  do exemplo do algoritmo de Boulala e Uhry

com isto:

$$\sigma = -2$$

$$S^* = \{c,d\} \quad s^* = 4$$

logo:

$$S^2 = \{d\} \quad s^2 = 1$$

portanto  $S = \{d,c\}$  é um conjunto independente de peso máximo  $s = 2$ .

### Algoritmo proposto

A idéia é construir a família  $F(G)$  de conjuntos independentes maximais de um grafo  $G$  obtido por conexão paralela de dois grafos  $G_1$  e  $G_2$ , a partir das famílias de c.i.m. dos grafos  $G_1$  e  $G_2$  que o compõem.

**Observação 5.4:**

Seja  $G = G_1 \parallel G_2$  com vértices terminais  $a$  e  $b$ , tem-se duas possibilidades:

$$(a,b) \notin E_G$$

$$(a,b) \in E_G$$

Dados  $I_1$  um conjunto independente maximal qualquer de  $G_1$  e  $I_2$  um conjunto independente maximal qualquer de  $G_2$ , existem vários casos:

1.-  $(a,b) \notin E_G$  (figura 5.27)

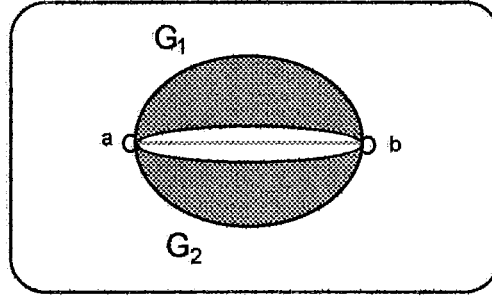


Fig. 5.27: Caso  $(a,b) \notin E_G$

(1a)  $a \in I_1 \cap I_2, b \in I_1 \cap I_2 \Rightarrow I = I_1 \cup I_2$  é c.i.m. de  $G$

$I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ , e são conjuntos independentes, logo  $I$  é conjunto independente maximal

(1b)  $a \in I_1 \cap I_2, b \in I_1, b \notin I_2 \Rightarrow I = [I_1 - \{b\}] \cup I_2$  é c.i.m. de  $G$

$b \notin I_2$  então  $\exists z \in I_2$  com  $(b,z) \in E_2$  caso contrário  $I_2$  não seria maximal, logo para  $I$  ser conjunto independente  $b$  não pode pertencer a  $I$ ,

$I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ , para garantir a maximalidade de  $I$  deve  $\exists v \in Adj_{G_1}(b)$  com  $(v,w) \in E(G_1), w \in I_1$

(1c)  $a \in I_1 \cap I_2, b \notin I_1 \cup I_2 \Rightarrow I = I_1 \cup I_2$  é c.i.m. de  $G$

para  $I$  ser conjunto independente  $b$  não pode pertencer a  $I$ ,

$I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$

(1d)  $a \in I_1, a \notin I_2, b \in I_2, b \notin I_1 \Rightarrow I = [I_1 - \{a\}] \cup [I_2 - \{b\}]$  é c.i.m. de  $G$

$b \notin I_1$  então  $\exists u \in I_1$  com  $(b,u) \in E_1$  caso contrário  $I_1$  não seria maximal,

$a \notin I_2$  então  $\exists z \in I_2$  com  $(a,z) \in E_2$  caso contrário  $I_2$  não seria maximal,

$I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ , para garantir a maximalidade de  $I$  deve  $\exists v \in \text{Adj}_{G_2}(b)$  com  $(v,w) \in E(G_2)$ ,  $w \in I_2$  e  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  $x \in I_1$

(1e)  $a \in I_1, a \notin I_2, b \in I_1, b \notin I_2 \Rightarrow I = [I_1 - \{a,b\}] \cup I_2$  é c.i.m. de  $G$

$b \notin I_2$  então  $\exists u \in I_2$  com  $(b,u) \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $a \notin I_2$  então  $\exists z \in I_2$  com  $(a,z) \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ , para garantir a maximalidade de  $I$  deve  $\exists v \in \text{Adj}_{G_1}(b)$  com  $(v,w) \in E(G_1)$ ,  $w \in I_1$  e  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  $x \in I_1$

(1f)  $a \in I_1, a \notin I_2, b \notin I_1 \cup I_2 \Rightarrow I = [I_1 - \{a\}] \cup I_2$  é c.i.m. de  $G$

$b \notin I_1$  então  $\exists u \in I_1$  com  $(b,u) \in E_1$  caso contrário  $I_1$  não seria maximal,  
 $b \notin I_2$  então  $\exists u' \in I_2$  com  $(b,u') \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $a \notin I_2$  então  $\exists z \in I_2$  com  $(a,z) \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ , para garantir a maximalidade de  $I$  deve  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  $x \in I_1$

(1g)  $a \notin I_1 \cup I_2, b \notin I_1 \cup I_2 \Rightarrow I = I_1 \cup I_2$  é c.i.m. de  $G$

$I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ .

2.-  $(a,b) \in E_G$  (figura 5.28)

supoe-se  $(a,b) \in E_1$ , então  $a$  e  $b$  não pertencer simultaneamente a  $I_1$

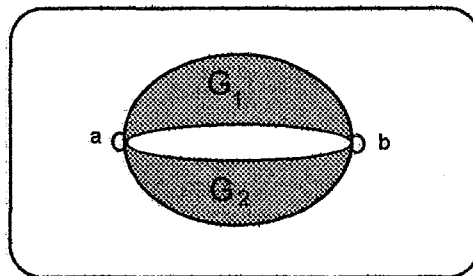


Fig. 5.28: Caso  $(a,b) \in E_1$

(2a)  $a \in I_1 \cap I_2, b \notin I_1 \cup I_2 \Rightarrow I = I_1 \cup I_2$  é c.i.m. de  $G$

para  $I$  ser conjunto independente  $b$  não pode pertencer a  $I$ ,  
 $I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$



(2b)  $a \in I_1, a \notin I_2, b \in I_2, b \notin I_1 \Rightarrow I = [I_1 - \{a\}] \cup [I_2 - \{b\}]$  é c.i.m. de  $G$

$b \notin I_1$  então  $\exists u \in I_1$  com  $(b,u) \in E_1$  caso contrário  $I_1$  não seria maximal,  
 $a \notin I_2$  então  $\exists z \in I_2$  com  $(a,z) \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ ,  
 para garantir a maximalidade de  $I$  deve  $\exists v \in \text{Adj}_{G_2}(b)$  com  $(v,w) \in E(G_2), w \in I_2$   
 e  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1), x \in I_1$

(2c)  $a \notin I_1 \cup I_2, b \notin I_1 \cup I_2 \Rightarrow I = I_1 \cup I_2$  é c.i.m. de  $G$

$I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$

(2d)  $a \in I_2, a \notin I_1, b \notin I_1 \cup I_2 \Rightarrow I = I_1 \cup [I_2 - \{a\}]$  é c.i.m. de  $G$

$b \notin I_1$  então  $\exists u \in I_1$  com  $(b,u) \in E_1$  caso contrário  $I_1$  não seria maximal,  
 $b \notin I_2$  então  $\exists u' \in I_2$  com  $(b,u') \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $a \notin I_1$  então  $\exists z \in I_1$  com  $(a,z) \in E_1$  caso contrário  $I_1$  não seria maximal,  
 $I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ ,  
 para garantir a maximalidade de  $I$  deve  $\exists t \in \text{Adj}_{G_2}(a)$  com  $(t,x) \in E(G_2), x \in I_2$

(2e)  $a \in I_2, a \notin I_1, b \in I_2, b \notin I_1 \Rightarrow I = I_1 \cup [I_2 - \{a,b\}]$  é c.i.m. de  $G$

$b \notin I_1$  então  $\exists u \in I_1$  com  $(b,u) \in E_1$  caso contrário  $I_1$  não seria maximal,  
 $a \notin I_1$  então  $\exists z \in I_1$  com  $(a,z) \in E_1$  caso contrário  $I_1$  não seria maximal,  
 $I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ ,  
 para garantir a maximalidade de  $I$  deve  $\exists v \in \text{Adj}_{G_2}(b)$  com  $(v,w) \in E(G_2), w \in I_2$   
 e  $\exists t \in \text{Adj}_{G_2}(a)$  com  $(t,x) \in E(G_2), x \in I_2$

(2f)  $a \in I_1, a \notin I_2, b \notin I_1 \cup I_2 \Rightarrow I = [I_1 - \{a\}] \cup I_2$  é c.i.m. de  $G$

$b \notin I_1$  então  $\exists u \in I_1$  com  $(b,u) \in E_1$  caso contrário  $I_1$  não seria maximal,  
 $b \notin I_2$  então  $\exists u' \in I_2$  com  $(b,u') \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $a \notin I_2$  então  $\exists z \in I_2$  com  $(a,z) \in E_2$  caso contrário  $I_2$  não seria maximal,  
 $I_1$  e  $I_2$  são maximais então não é possível acrescentar mais vértices a  $I$ ,  
 para garantir a maximalidade de  $I$  deve  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1), x \in I_1$

(2g)  $a \in I_1 \cap I_2, b \in I_2, b \notin I_1$  então  $I = [I_1 - \{a\}] \cup I_2$  é conjunto independente de  $G$  mas não é maximal, está considerado no caso (2b).

Por exemplo, para o ciclo  $C_5$  da figura 5.29, considerando a decomposição nos grafos  $G_1$  e  $G_2$  da mesma figura, tem-se:

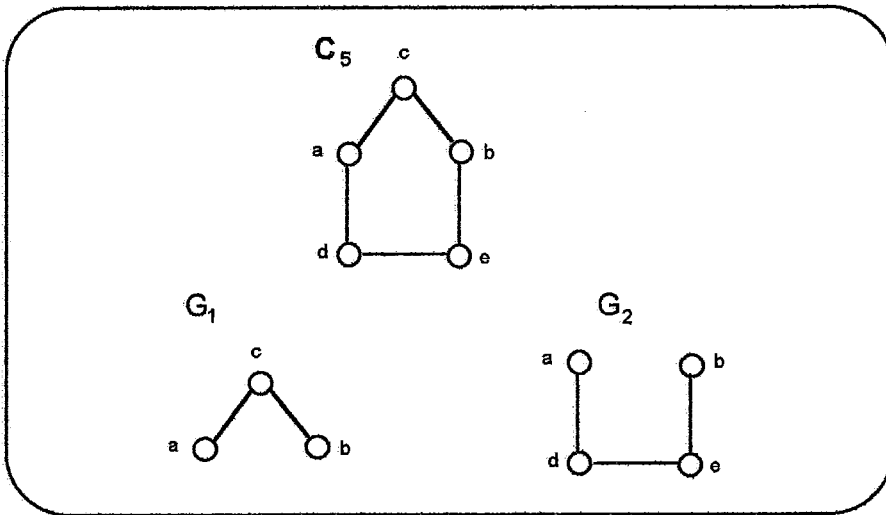


Fig. 5.29: Exemplo de conexão paralela

$$(a,b) \notin E_1$$

$$F(G_1) = \{ \{a,b\}, \{c\} \}$$

$$F(G_2) = \{ \{a,b\}, \{a,e\}, \{b,d\} \}$$

$$F_{ab}^1 = \{ \{a,b\} \}$$

$$\bar{F}_{ab}^1 = \{ \{c\} \}$$

$$F_a^1 = \emptyset$$

$$F_b^1 = \emptyset$$

$$F_{ab}^2 = \{ \{a,b\} \}$$

$$\bar{F}_{ab}^2 = \emptyset$$

$$F_a^2 = \{ \{a,e\} \}$$

$$F_b^2 = \{ \{b,d\} \}$$

caso (1a):  $I_1 = \{a,b\}, I_2 = \{a,b\} \Rightarrow I = \{a,b\} \in F(C_5)$

caso (1b):  $I_1 = \{a,b\}, I_2 = \{a,e\} \Rightarrow I = \{a,e\} \in F(C_5)$

caso (1b):  $I_1 = \{a,b\}, I_2 = \{b,d\} \Rightarrow I = \{b,d\} \in F(C_5)$

caso (1e):  $I_1 = \{c\}, I_2 = \{a,b\} \Rightarrow I = \{c\}$  não maximal

caso (1f):  $I_1 = \{c\}, I_2 = \{a,e\} \Rightarrow I = \{c,e\} \in F(C_5)$

caso (1f):  $I_1 = \{c\}, I_2 = \{b,d\} \Rightarrow I = \{c,d\} \in F(C_5)$

$$\text{logo } F(C_5) = \{ \{a,b\}, \{a,e\}, \{b,d\}, \{c,d\}, \{c,e\} \}.$$

A observação 5.4 é a justificativa da propriedade 5.5 enunciada abaixo, a qual a sua vez é a base do procedimento descrito no algoritmo 5.4.

**Propriedade 5.5:**

Dadas as famílias  $F(G_1)$  e  $F(G_2)$  de c.i.m. de  $G_1$  e de  $G_2$ , sejam para  $i=1,2$ :

$F_{ab}^i$  = família de c.i.m. de  $G_i$  que contém **a** e **b**,

$\bar{F}_{ab}^i$  = família de c.i.m. de  $G_i$  que não contém **a** nem **b**,

$F_a^i$  = família de c.i.m. de  $G_i$  que contém **a** mas não contém **b**,

$F_b^i$  = família de c.i.m. de  $G_1$  que contém  $b$  mas não contém  $a$ ,  
então  $F(G)$  é construída segundo o seguinte procedimento:

1.- Se  $(a,b) \notin E_G$  então

- a) para todo  $I_1 \in F_{ab}^1$ , para todo  $I_2 \in F_{ab}^2$ :  $I_1 \cup I_2 \in F(G)$
- b) para todo  $I_1 \in F_{ab}^1$ ,  
se  $\exists v \in \text{Adj}_{G_1}(b)$  com  $(v,w) \in E(G_1)$ ,  $w \in I_1$  então  
para todo  $I_2 \in F_a^2$ :  $[I_1 - \{b\}] \cup I_2 \in F(G)$   
para todo  $I_1 \in F_{ab}^1$ ,  
se  $\exists v \in \text{Adj}_{G_1}(a)$  com  $(v,w) \in E(G_1)$ ,  $w \in I_1$  então  
para todo  $I_2 \in F_b^2$ :  $[I_1 - \{a\}] \cup I_2 \in F(G)$
- c) para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in F_a^2$ :  $I_1 \cup I_2 \in F(G)$   
para todo  $I_1 \in F_b^1$ , para todo  $I_2 \in F_b^2$ :  $I_1 \cup I_2 \in F(G)$
- d) se  $\exists v \in \text{Adj}_{G_2}(b)$  com  $(v,w) \in E(G_2)$ ,  $w \in I_2$  e  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  
 $x \in I_1$  então:  
para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in F_b^2$ :  $[I_1 - \{a\}] \cup [I_2 - \{b\}] \in F(G)$   
se  $\exists v \in \text{Adj}_{G_2}(a)$  com  $(v,w) \in E(G_2)$ ,  $w \in I_2$  e  $\exists t \in \text{Adj}_{G_1}(b)$  com  $(t,x) \in E(G_1)$ ,  
 $x \in I_1$  então:  
para todo  $I_1 \in F_b^1$ , para todo  $I_2 \in F_a^2$ :  $[I_1 - \{b\}] \cup [I_2 - \{a\}] \in F(G)$
- e) se  $\exists v \in \text{Adj}_{G_1}(b)$  com  $(v,w) \in E(G_1)$ ,  $w \in I_1$  e  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  
 $x \in I_1$  então:  
para todo  $I_1 \in F_{ab}^1$ , para todo  $I_2 \in F_{ab}^2$ :  $[I_1 - \{a,b\}] \cup I_2 \in F(G)$
- f) se  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  $x \in I_1$  então:  
para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in \overline{F}_{ab}^2$ :  $[I_1 - \{a\}] \cup I_2 \in F(G)$   
se  $\exists t \in \text{Adj}_{G_1}(b)$  com  $(t,x) \in E(G_1)$ ,  $x \in I_1$  então:  
para todo  $I_1 \in F_b^1$ , para todo  $I_2 \in \overline{F}_{ab}^2$ :  $[I_1 - \{b\}] \cup I_2 \in F(G)$
- g) para todo  $I_1 \in \overline{F}_{ab}^1$ , para todo  $I_2 \in \overline{F}_{ab}^2$ :  $I_1 \cup I_2 \in F(G)$ .

2.- Se  $(a,b) \in E_1$  então

- a) para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in F_a^2$ :  $I_1 \cup I_2 \in F(G)$   
para todo  $I_1 \in F_b^1$ , para todo  $I_2 \in F_b^2$ :  $I_1 \cup I_2 \in F(G)$
- b) se  $\exists v \in \text{Adj}_{G_2}(b)$  com  $(v,w) \in E(G_2)$ ,  $w \in I_2$  e  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  
 $x \in I_1$  então  
para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in F_b^2$ :  $[I_1 - \{a\}] \cup [I_2 - \{b\}] \in F(G)$   
se  $\exists v \in \text{Adj}_{G_2}(a)$  com  $(v,w) \in E(G_2)$ ,  $w \in I_2$  e  $\exists t \in \text{Adj}_{G_1}(b)$  com  $(t,x) \in E(G_1)$ ,  
 $x \in I_1$  então  
para todo  $I_1 \in F_b^1$ , para todo  $I_2 \in F_a^2$ :  $[I_1 - \{b\}] \cup [I_2 - \{a\}] \in F(G)$
- c) para todo  $I_1 \in \overline{F}_{ab}^1$ , para todo  $I_2 \in \overline{F}_{ab}^2$ :  $I_1 \cup I_2 \in F(G)$

- d) se  $\exists t \in \text{Adj}_{G_2}(\mathbf{a})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_2)$ ,  $\mathbf{x} \in I_2$  então  
 para todo  $I_1 \in \overline{F}_{ab}^1$ , para todo  $I_2 \in F_a^2$ :  $I_1 \cup [I_2 - \{\mathbf{a}\}] \in F(G)$   
 se  $\exists t \in \text{Adj}_{G_2}(\mathbf{b})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_2)$ ,  $\mathbf{x} \in I_2$  então  
 para todo  $I_1 \in \overline{F}_{ab}^1$ , para todo  $I_2 \in F_b^2$ :  $I_1 \cup [I_2 - \{\mathbf{b}\}] \in F(G)$
- e) se  $\exists v \in \text{Adj}_{G_2}(\mathbf{b})$  com  $(v, \mathbf{w}) \in \mathbf{E}(G_2)$ ,  $\mathbf{w} \in I_2$  e  $\exists t \in \text{Adj}_{G_2}(\mathbf{a})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_2)$ ,  $\mathbf{x} \in I_2$  então  
 para todo  $I_1 \in \overline{F}_{ab}^1$ , para todo  $I_2 \in F_{ab}^2$ :  $I_1 \cup [I_2 - \{\mathbf{a}, \mathbf{b}\}] \in F(G)$
- f) se  $\exists t \in \text{Adj}_{G_1}(\mathbf{a})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_1)$ ,  $\mathbf{x} \in I_1$  então  
 para todo  $I_1 \in F_a^1$ , para todo  $I_2 \in \overline{F}_{ab}^2$ :  $[I_1 - \{\mathbf{a}\}] \cup I_2 \in F(G)$   
 se  $\exists t \in \text{Adj}_{G_1}(\mathbf{b})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_1)$ ,  $\mathbf{x} \in I_1$  então  
 para todo  $I_1 \in F_b^1$ , para todo  $I_2 \in \overline{F}_{ab}^2$ :  $[I_1 - \{\mathbf{b}\}] \cup I_2 \in F(G)$ .

O algoritmo 5.4 implementa o procedimento para construir a família de c.i.m. de um grafo  $G$  obtido pela conexão paralela de dois grafos identificando seus dois vértices terminais, dadas as famílias de c.i.m. desses dois grafos.

**Algoritmo 5.4: Família de c.i.m de um grafo conexão paralela de dois grafos**

Dados  $G, G_1, G_2, \mathbf{a}$  e  $\mathbf{b}$  vértices terminais de  $G, F(G_1), F(G_2), F_{ab}^1, F_{ab}^2, \overline{F}_{ab}^1, \overline{F}_{ab}^2, F_a^1, F_a^2, F_b^1, F_b^2$   
 $F = \emptyset$

1. Se  $(\mathbf{a}, \mathbf{b}) \notin \mathbf{E}_G$  então efetuar
  - 1.1 se  $F_{ab}^1 \neq \emptyset$  e  $F_{ab}^2 \neq \emptyset$  então efetuar  
 para todo  $I_1 \in F_{ab}^1$   
 para todo  $I_2 \in F_{ab}^2$   
 $F = F \cup \{ I_1 \cup I_2 \}$
  - 1.2 se  $F_{ab}^1 \neq \emptyset$  e  $F_a^2 \neq \emptyset$  então efetuar  
 para todo  $I_1 \in F_{ab}^1$   
 se  $\exists v \in \text{Adj}_{G_1}(\mathbf{b})$  com  $(v, \mathbf{w}) \in \mathbf{E}(G_1)$ ,  $\mathbf{w} \in I_1$  então efetuar  
 para todo  $I_2 \in F_a^2$   
 $F = F \cup \{ [I_1 - \{\mathbf{b}\}] \cup I_2 \}$
  - se  $F_{ab}^1 \neq \emptyset$  e  $F_b^2 \neq \emptyset$  então efetuar  
 para todo  $I_1 \in F_{ab}^1$   
 se  $\exists v \in \text{Adj}_{G_1}(\mathbf{a})$  com  $(v, \mathbf{w}) \in \mathbf{E}(G_1)$ ,  $\mathbf{w} \in I_1$  então efetuar  
 para todo  $I_2 \in F_b^2$   
 $F = F \cup \{ [I_1 - \{\mathbf{a}\}] \cup I_2 \}$

- 1.3 se  $F_a^1 \neq \emptyset$  e  $F_a^2 \neq \emptyset$  então efetuar  
para todo  $I_1 \in F_a^1$   
para todo  $I_2 \in F_a^2$   
 $F = F \cup \{ I_1 \cup I_2 \}$
- se  $F_b^1 \neq \emptyset$  e  $F_b^2 \neq \emptyset$  então efetuar  
para todo  $I_1 \in F_b^1$   
para todo  $I_2 \in F_b^2$   
 $F = F \cup \{ I_1 \cup I_2 \}$
- 1.4 se  $F_a^1 \neq \emptyset$  e  $F_b^2 \neq \emptyset$  então efetuar  
se  $\exists v \in \text{Adj}_{G_2}(\mathbf{b})$  com  $(v, w) \in E(G_2)$ ,  $w \in I_2$   
e  $\exists t \in \text{Adj}_{G_1}(\mathbf{a})$  com  $(t, x) \in E(G_1)$ ,  $x \in I_1$  então efetuar  
para todo  $I_1 \in F_a^1$   
para todo  $I_2 \in F_b^2$   
 $F = F \cup \{ [I_1 - \{\mathbf{a}\}] \cup [I_2 - \{\mathbf{b}\}] \}$
- se  $F_b^1 \neq \emptyset$  e  $F_a^2 \neq \emptyset$  então efetuar  
se  $\exists v \in \text{Adj}_{G_2}(\mathbf{a})$  com  $(v, w) \in E(G_2)$ ,  $w \in I_2$   
e  $\exists t \in \text{Adj}_{G_1}(\mathbf{b})$  com  $(t, x) \in E(G_1)$ ,  $x \in I_1$  então efetuar  
para todo  $I_1 \in F_b^1$   
para todo  $I_2 \in F_a^2$   
 $F = F \cup \{ [I_1 - \{\mathbf{b}\}] \cup [I_2 - \{\mathbf{a}\}] \}$
- 1.5 se  $F_{ab}^1 \neq \emptyset$  e  $F_{ab}^2 \neq \emptyset$  então efetuar  
se  $\exists v \in \text{Adj}_{G_1}(\mathbf{b})$  com  $(v, w) \in E(G_1)$ ,  $w \in I_1$   
e  $\exists t \in \text{Adj}_{G_1}(\mathbf{a})$  com  $(t, x) \in E(G_1)$ ,  $x \in I_1$  então efetuar  
para todo  $I_1 \in F_{ab}^1$   
para todo  $I_2 \in F_{ab}^2$   
 $F = F \cup \{ [I_1 - \{\mathbf{a}, \mathbf{b}\}] \cup I_2 \}$
- 1.6 se  $F_a^1 \neq \emptyset$  e  $\overline{F}_{ab}^2 \neq \emptyset$  então efetuar  
se  $\exists t \in \text{Adj}_{G_1}(\mathbf{a})$  com  $(t, x) \in E(G_1)$ ,  $x \in I_1$  então efetuar  
para todo  $I_1 \in F_a^1$   
para todo  $I_2 \in \overline{F}_{ab}^2$   
 $F = F \cup \{ [I_1 - \{\mathbf{a}\}] \cup I_2 \}$
- se  $F_b^1 \neq \emptyset$  e  $\overline{F}_{ab}^2 \neq \emptyset$  então efetuar  
se  $\exists t \in \text{Adj}_{G_1}(\mathbf{b})$  com  $(t, x) \in E(G_1)$ ,  $x \in I_1$  então efetuar  
para todo  $I_1 \in F_b^1$   
para todo  $I_2 \in \overline{F}_{ab}^2$   
 $F = F \cup \{ [I_1 - \{\mathbf{b}\}] \cup I_2 \}$

- 1.7 se  $F_{ab}^1 \neq \emptyset$  e  $F_{ab}^2 \neq \emptyset$  então efetuar  
para todo  $I_1 \in F_{ab}^1$   
para todo  $I_2 \in F_{ab}^2$   
 $F = F \cup \{ I_1 \cup I_2 \}$
2. Se  $(a,b) \in E_1$  então efetuar
- 2.1 se  $F_a^1 \neq \emptyset$  e  $F_a^2 \neq \emptyset$  então efetuar  
para todo  $I_1 \in F_a^1$   
para todo  $I_2 \in F_a^2$   
 $F = F \cup \{ I_1 \cup I_2 \}$
- se  $F_b^1 \neq \emptyset$  e  $F_b^2 \neq \emptyset$  então efetuar  
para todo  $I_1 \in F_b^1$   
para todo  $I_2 \in F_b^2$   
 $F = F \cup \{ I_1 \cup I_2 \}$
- 2.2 se  $F_a^1 \neq \emptyset$  e  $F_b^2 \neq \emptyset$  então efetuar  
se  $\exists v \in \text{Adj}_{G_2}(b)$  com  $(v,w) \in E(G_2)$ ,  $w \in I_2$   
e  $\exists t \in \text{Adj}_{G_1}(a)$  com  $(t,x) \in E(G_1)$ ,  $x \in I_1$  então efetuar  
para todo  $I_1 \in F_a^1$   
para todo  $I_2 \in F_b^2$   
 $F = F \cup \{ [I_1 - \{a\}] \cup [I_2 - \{b\}] \}$
- se  $F_b^1 \neq \emptyset$  e  $F_a^2 \neq \emptyset$  então efetuar  
se  $\exists v \in \text{Adj}_{G_2}(a)$  com  $(v,w) \in E(G_2)$ ,  $w \in I_2$   
e  $\exists t \in \text{Adj}_{G_1}(b)$  com  $(t,x) \in E(G_1)$ ,  $x \in I_1$  então efetuar  
para todo  $I_1 \in F_b^1$   
para todo  $I_2 \in F_a^2$   
 $F = F \cup \{ [I_1 - \{b\}] \cup [I_2 - \{a\}] \}$
- 2.3 se  $\overline{F}_{ab}^1 \neq \emptyset$  e  $\overline{F}_{ab}^2 \neq \emptyset$  então efetuar  
para todo  $I_1 \in \overline{F}_{ab}^1$   
para todo  $I_2 \in \overline{F}_{ab}^2$   
 $F = F \cup \{ I_1 \cup I_2 \}$
- 2.4 se  $\overline{F}_{ab}^1 \neq \emptyset$  e  $F_a^2 \neq \emptyset$  então efetuar  
se  $\exists t \in \text{Adj}_{G_2}(a)$  com  $(t,x) \in E(G_2)$ ,  $x \in I_2$  então efetuar  
para todo  $I_1 \in \overline{F}_{ab}^1$   
para todo  $I_2 \in F_a^2$   
 $F = F \cup \{ I_1 \cup [I_2 - \{a\}] \}$

se  $\overline{F}_{ab}^1 \neq \emptyset$  e  $F_b^2 \neq \emptyset$  então efetuar  
 se  $\exists t \in \text{Adj}_{G_2}(\mathbf{b})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_2)$ ,  $\mathbf{x} \in I_2$  então efetuar  
 para todo  $I_1 \in \overline{F}_{ab}^1$   
 para todo  $I_2 \in F_b^2$   

$$F = F \cup \{ I_1 \cup [I_2 - \{\mathbf{b}\}] \}$$

2.5 se  $\overline{F}_{ab}^1 \neq \emptyset$  e  $F_{ab}^2 \neq \emptyset$  então efetuar  
 se  $\exists v \in \text{Adj}_{G_2}(\mathbf{b})$  com  $(v, \mathbf{w}) \in \mathbf{E}(G_2)$ ,  $\mathbf{w} \in I_2$   
 e  $\exists t \in \text{Adj}_{G_2}(\mathbf{a})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_2)$ ,  $\mathbf{x} \in I_2$  então efetuar  
 para todo  $I_1 \in \overline{F}_{ab}^1$   
 para todo  $I_2 \in F_{ab}^2$   

$$F = F \cup \{ I_1 \cup [I_2 - \{\mathbf{a}, \mathbf{b}\}] \}$$

2.6 se  $F_a^1 \neq \emptyset$  e  $\overline{F}_{ab}^2 \neq \emptyset$  então efetuar  
 se  $\exists t \in \text{Adj}_{G_1}(\mathbf{a})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_1)$ ,  $\mathbf{x} \in I_1$  então efetuar  
 para todo  $I_1 \in F_a^1$   
 para todo  $I_2 \in \overline{F}_{ab}^2$   

$$F = F \cup \{ [I_1 - \{\mathbf{a}\}] \cup I_2 \}$$

se  $F_b^1 \neq \emptyset$  e  $\overline{F}_{ab}^2 \neq \emptyset$  então efetuar  
 se  $\exists t \in \text{Adj}_{G_1}(\mathbf{b})$  com  $(t, \mathbf{x}) \in \mathbf{E}(G_1)$ ,  $\mathbf{x} \in I_1$  então efetuar  
 para todo  $I_1 \in F_b^1$   
 para todo  $I_2 \in \overline{F}_{ab}^2$   

$$F = F \cup \{ [I_1 - \{\mathbf{b}\}] \cup I_2 \}.$$

### Complexidade:

Passo 1:

- 1.1:  $i(G_1) i(G_2)$
- 1.2:  $n^2 i(G_1) i(G_2)$
- 1.3:  $i(G_1) i(G_2)$
- 1.4:  $n^2 i(G_1) i(G_2)$
- 1.5:  $n^2 i(G_1) i(G_2)$
- 1.6:  $n^2 i(G_1) i(G_2)$
- 1.7:  $i(G_1) i(G_2)$

total do passo 1:  $O(n^2 i(G_1) i(G_2))$

Passo 2:

- 2.1:  $i(G_1) i(G_2)$
- 2.2:  $n^2 i(G_1) i(G_2)$
- 2.3:  $i(G_1) i(G_2)$
- 2.2:  $n^2 i(G_1) i(G_2)$
- 2.2:  $n^2 i(G_1) i(G_2)$
- 2.2:  $n^2 i(G_1) i(G_2)$

total do passo 2:  $O(n^2 i(G_1) i(G_2))$

Logo, a complexidade do algoritmo 5.4 é  $O(n^2 i(G_1) i(G_2))$ .

Para o grafo  $G$  da figura 5.25, considerando a decomposição nos grafos  $G_1$  e  $G_2$  segundo a figura 5.30, tem-se:

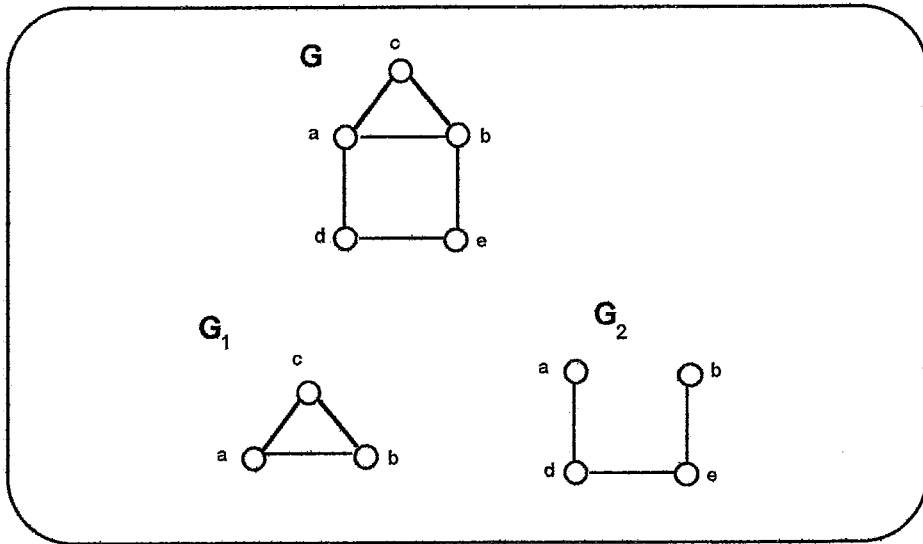


Fig. 5.30: Exemplo do algoritmo conexão paralela

$$(a,b) \in E_1$$

$$F(G_1) = \{ \{a\}, \{b\}, \{c\} \}$$

$$F(G_2) = \{ \{a,b\}, \{a,e\}, \{b,d\} \}$$

$$F_{ab}^1 = \emptyset \quad \bar{F}_{ab}^1 = \{ \{c\} \} \quad F_a^1 = \{ \{a\} \} \quad F_b^1 = \{ \{b\} \}$$

$$F_{ab}^2 = \{ \{a,b\} \} \quad \bar{F}_{ab}^2 = \emptyset \quad F_a^2 = \{ \{a,e\} \} \quad F_b^2 = \{ \{b,d\} \}$$

PASSO	$I_1$	$I_2$	c.i.m.
2.1	{a}	{a,e}	{a,e}
	{b}	{b,d}	{b,d}
2.2	{a}	{b,d}	-
	{b}	{a,e}	-
2.4	{a}	{a,b}	-
	{b}	{a,b}	-
2.6	{c}	{a,b}	-
2.7	{c}	{a,e}	{c,e}
	{c}	{b,d}	{c,d}

$$\log F(G) = \{ \{a,e\}, \{b,d\}, \{c,d\}, \{c,e\} \}.$$



### 5.5. Grafos série-paralelo a dois terminais

Um grafo  $G$  é série-paralelo a dois terminais se ele pode ser obtido por um número finito de aplicações sucessivas de conexões série e/ou conexões paralelas, a partir do grafo série-paralelo básico ou mínimo  $G_b$ , exibido na figura 5.31, que consiste de dois vértices unidos por uma aresta simples, os dois vértices são os terminais de  $G_b$ .

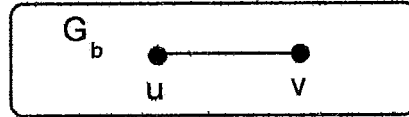


Fig. 5.31: Grafo série-paralelo mínimo

O grafo  $G = G_1 * G_2$  obtido por conexão série dos grafos  $G_1(V_1, E_1, v_{11}, v_{12})$  e  $G_2(V_2, E_2, v_{21}, v_{22})$  com  $V_1 \cap V_2 = \emptyset$ , onde  $v_{11}$  e  $v_{12}$  são os vértices terminais de  $G_1$ ,  $v_{21}$  e  $v_{22}$  são os vértices terminais de  $G_2$ , identificando os vértices  $v_{12}$  e  $v_{21}$ , é dado por:  $V_G = V_1 \cup V_2$ ,  $E_G = E_1 \cup E_2$ , com vértices terminais  $v_{11}$  e  $v_{22}$ , e é exibido na figura 5.32. A conexão série dos grafos  $G_1$  e  $G_2$  que gera o grafo  $G$  é equivalente a considerar que  $G$  tem uma articulação correspondente ao vértice  $v_{12} = v_{21}$  que o separa em  $G_1$  e  $G_2$ . Esta operação é descrita na seção 5.1.

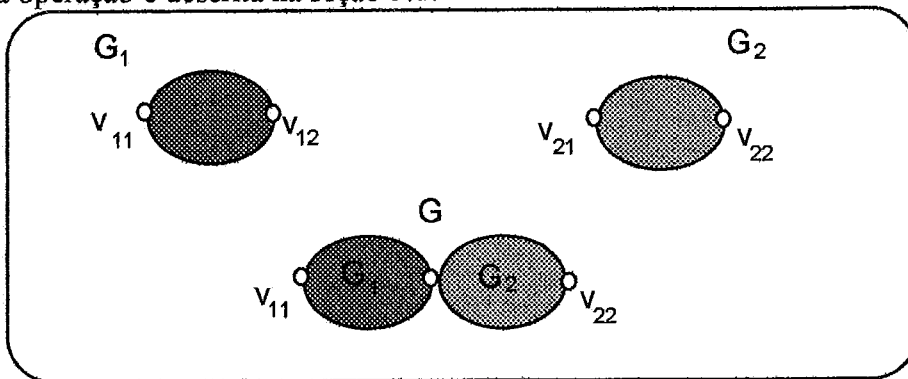


Fig. 5.32: Conexão série

O grafo  $G = G_1 || G_2$  obtido por conexão paralela dos grafos  $G_1(V_1, E_1, v_{11}, v_{12})$  e  $G_2(V_2, E_2, v_{21}, v_{22})$  com  $V_1 \cap V_2 = \emptyset$ , onde  $v_{11}$  e  $v_{12}$  são os vértices terminais de  $G_1$ ,  $v_{21}$  e  $v_{22}$  são os vértices terminais de  $G_2$ , identificando os vértices  $v_{11}$  com  $v_{21}$  e  $v_{12}$  com  $v_{22}$ , é dado por:  $V_G = V_1 \cup V_2$ ,  $E_G = E_1 \cup E_2$ , com vértices terminais  $v_1$  e  $v_2$  ( $v_1 = v_{11} = v_{21}$ ,  $v_2 = v_{12} = v_{22}$ ), exibido na figura 5.33. Esta operação é descrita na seção 5.4.

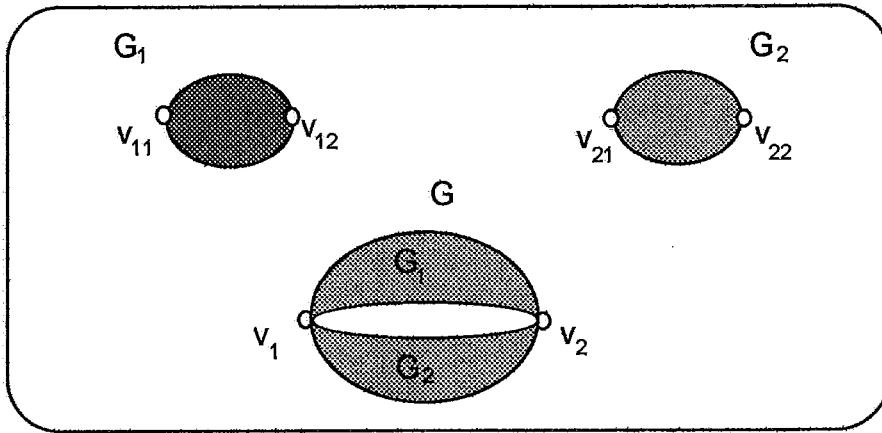


Fig. 5.33: Conexão paralela

### Árvore de decomposição

Dado um grafo  $G$  série-paralelo constrói-se uma árvore binária de decomposição, onde cada nó  $v$  é rotulado com **S** ou **P** para indicar se é realizada uma conexão série ou uma conexão paralela nos grafos correspondentes aos filhos do nó  $v$ , e é rotulado com a aresta  $(u,v)$  se for o grafo série-paralelo mínimo. Na figura 5.34 exhibe-se o grafo  $G$ , os grafos usados na construção e a árvore de decomposição  $T_G$  de  $G$ .

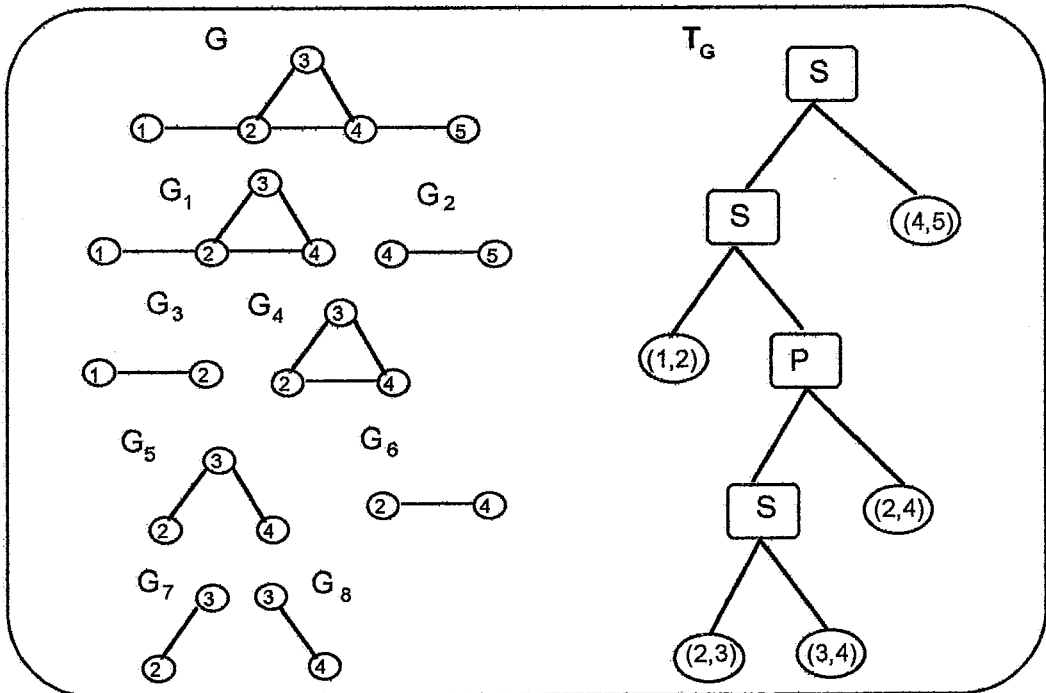


Fig. 5.34: Árvore de decomposição

Boulala e Uhry (1979) provam que a árvore de decomposição pode ser construída em tempo linear e que tem no máximo  $n$  vértices.

### Algoritmo proposto

Para construir a família  $F(G)$  de conjuntos independentes maximais de um grafo  $G$  série-paralelo a dois terminais, utiliza-se a árvore binária de decomposição. Subindo nesta árvore  $T_G$  desde as folhas até a raiz, opera-se com os algoritmos para construir a família de c.i.m. de um grafo obtido por conexão série (algoritmo 5.1) ou por conexão paralela (algoritmo 5.4) de dois grafos  $G_1$  e  $G_2$ , a partir das famílias de c.i.m. dos grafos  $G_1$  e  $G_2$  que o compõem, começando com as famílias dos grafos mínimos correspondentes às folhas da árvore.

O algoritmo 5.5 implementa o procedimento para construir a família de c.i.m. de um grafo  $G$  série-paralelo, dadas a árvore de decomposição por cliques  $T_G$  e as famílias de c.i.m. dos subgrafos primitivos gerados na decomposição utilizando o algoritmo 5.1 para gerar a família de c.i.m. de um grafo construído por conexão série de dois grafos e o algoritmo 5.4 para gerar a família de c.i.m. de um grafo construído por conexão paralela de dois grafos.

#### ***Algoritmo 5.5:*** Família de c.i.m de um grafo série-paralelo a dois terminais

Dados  $G$  e a árvore de decomposição  $T_G$

1. Determinar a altura  $h$  da árvore  $T_G$
2. Para todo grafo primitivo  $G_p$  associado às folhas de  $T_G$   
construir a família  $F(G_p)$  de c.i.m. de  $G_p$
3. Para  $j = 1$  até  $h-1$  efetuar  
para todo nó  $G_j$  no nível  $h - j$  de  $T_G$  efetuar  
determinar a família  $F(G_j)$  do grafo associado ao nó  $G_j$ .

#### ***Complexidade:***

Passo 1:  $O(n)$

Passo 2: é executado  $O(n)$  vezes com tempo  $O(1)$

Passo 3: é executado  $O(s)$  vezes onde  $s$  é o número máximo de vértices de  $T_G$ ,  $s \leq n$ , pelos algoritmos 5.1 e 5.4, tempo  $O(n^2 i(G_1) i(G_2))$ , dado que  $G_1$  e  $G_2$  são os filhos de  $G_j$  em  $T_G$ .

A complexidade total do algoritmo 5.5 é  $O(n^3 i(G_1) i(G_2))$ .

Por exemplo, para o grafo  $G$  da figura 5.34 aplicando o algoritmo 5.5 tem-se:

Passo 1:

$$\text{altura } h = 4$$

Passo 2:

grafos primitivos  $G_2, G_3, G_6, G_7$  e  $G_8$

$$F(G_7) = \{\{2\}, \{3\}\}, \quad F(G_8) = \{\{3\}, \{4\}\}, \quad F(G_6) = \{\{2\}, \{4\}\}, \quad F(G_3) = \{\{1\}, \{2\}\}, \\ F(G_2) = \{\{4\}, \{5\}\}$$

Passo 3:

$$G_5 = G_7 * G_8$$

$$\text{aplicando o algoritmo 5.1} \Rightarrow F(G_5) = \{\{2,4\}, \{3\}\}$$

$$G_4 = G_5 \parallel G_6$$

aplicando o algoritmo 5.4, vértices terminais **2** e **4**:

$$F_{24}^1 = \emptyset \quad \bar{F}_{24}^1 = \{\{c\}\} \quad F_2^1 = \{\{a\}\} \quad F_4^1 = \{\{b\}\}$$

$$F_{24}^2 = \{\{a,b\}\} \quad \bar{F}_{24}^2 = \emptyset \quad F_2^2 = \{\{a,e\}\} \quad F_4^2 = \{\{b,d\}\}$$

$$\Rightarrow F(G_4) = \{\{2\}, \{3\}, \{4\}\}$$

$$G_1 = G_3 * G_4$$

$$\text{aplicando o algoritmo 5.1} \Rightarrow F(G_1) = \{\{1,3\}, \{1,4\}, \{2\}\}$$

$$G = G_1 * G_2$$

$$\text{aplicando o algoritmo 5.1} \Rightarrow F(G) = \{\{1,3,5\}, \{1,4\}, \{2,5\}\}.$$

## Capítulo 6

### CONCLUSÕES

Este trabalho versa sobre dois problemas de combinatória: *Quantos conjuntos independentes maximais existem em um grafo dado* (contagem) e *Qual é a família de conjuntos independentes maximais de um grafo dado* (enumeração).

No capítulo 2 resumem-se os resultados, encontrados na literatura, relacionados com o número  $i(G)$  de conjuntos independentes maximais de um grafo  $G$ . No capítulo 3 descrevem-se quatro algoritmos, da literatura, para enumerar todos os conjuntos independentes maximais de um grafo qualquer.

No capítulo 4 desenvolvem-se algoritmos para construir a família de conjuntos independentes maximais no caso particular de caminhos induzidos, ciclos induzidos, grafos grade completos de duas linhas, grafos de intervalo e grafos triangularizados; transformando o problema de construir a família de c.i.m do grafo dado no problema de determinar todos os caminhos maximais fonte-sumidouro num dígrafo que depende da classe particular considerada. Primeiro, desenvolve-se um algoritmo de tempo linear para cada conjunto independente maximal, para gerar todos os conjuntos independentes maximais de um caminho induzido. Utilizando a mesma idéia dos caminhos induzidos, é possível gerar a família de c.i.m. de um ciclo induzido em tempo linear para cada c.i.m.. Em ambos casos, a complexidade total do algoritmo é polinomial no tamanho do grafo e no número de c.i.m. do grafo.

Da mesma forma, desenvolve-se um algoritmo de tempo linear para cada c.i.m. de grafos grade completas de duas linhas, a complexidade total do algoritmo é polinomial no tamanho do grafo e no número de c.i.m. do grafo. Determina-se, além disso, a equação de recorrências que caracteriza o número de c.i.m. neste caso e a solução dessa equação que dá o número de c.i.m. de uma grade completa de duas linhas. Generalizando a idéia do algoritmo para gerar os c.i.m. de um caminho induzido, obtém-se um algoritmo para um grafo de intervalo, a complexidade por c.i.m. é polinomial no tamanho do grafo e a complexidade total é polinomial no tamanho do grafo e no número de c.i.m. dele. Este algoritmo é mais eficiente que o desenvolvido por Leung. Modificando esse algoritmo é possível contar o número de c.i.m. de um grafo de intervalo dado. Além disso, obtém-se um algoritmo para um grafo triangularizado, com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele, no entanto a complexidade por c.i.m. é linear no número de vértices; isto representa uma melhora em relação ao algoritmo de Leung.

Uma pergunta que surge naturalmente é, em que classes de grafos é possível caracterizar um dígrafo de modo que enumerar todos os conjuntos independentes maximais do grafo seja equivalente a resolver o problema de gerar todos os caminhos maximais fonte-sumidouro do grafo.

Outro problema é generalizar a enumeração e a contagem de conjuntos independentes maximais para grades completas, no caso geral de  $n$  linhas.

No capítulo 5 trabalha-se com a estratégia de operar dois grafos para os quais dispõe-se da família de c.i.m. de cada um deles. Dependendo do tipo de operação nos dois grafos que compõem o grafo de interesse, constrói-se a família de c.i.m. dele. A primeira operação considerada é a *conexão série* de dois grafos com um vértice comum a ambos, o qual é uma articulação do grafo resultante; primeiro descreve-se o algoritmo desenvolvido por Boulala e Uhry (1979) para gerar um conjunto independente de peso máximo. Em seguida, desenvolve-se um algoritmo para gerar a família de c.i.m. de um grafo obtido como conexão série de outros dois grafos, dadas as famílias de c.i.m. desses dois grafos menores, com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele. Generaliza-se então a conexão série, isto é a construção de um grafo com uma articulação no caso de um *grafo com um conjunto articulação que é uma clique*, primeiro descreve-se o algoritmo desenvolvido por Boulala e Uhry (1979) para gerar um conjunto independente de peso máximo; após, obtém-se um algoritmo para gerar a família de c.i.m. de um grafo obtido como conexão série de outros dois grafos identificando uma clique comum em ambos, dadas as famílias de c.i.m. desses dois grafos menores, com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele. Utilizando esse algoritmo, desenvolve-se um algoritmo para gerar a família de c.i.m. de um grafo separável por cliques, aproveitando a árvore de decomposição por cliques e a operação de unir dois grafos por um conjunto comum a ambos.

A segunda operação considerada é a *conexão paralela* de dois grafos com dois vértices comuns a ambos, os quais são denominados vértices terminais do grafo resultante; primeiro descreve-se o algoritmo desenvolvido por Boulala e Uhry (1979) para gerar um conjunto independente de peso máximo; em seguida, desenvolve-se um algoritmo para gerar a família de c.i.m. de um grafo obtido como conexão paralela de outros dois grafos, dadas as famílias de c.i.m. de esses dois grafos menores, com complexidade total polinomial no tamanho do grafo e no número de c.i.m. dele. Finalmente, desenvolve-se um algoritmo para gerar a família de c.i.m. de um *grafo série-paralelo a dois terminais*, aproveitando a árvore de decomposição do grafo e as operações de unir dois grafos por conexão série e por conexão paralela.

Esta decomposição do problema em subproblemas menores da mesma natureza, é uma boa alternativa para continuar a pesquisa em relação aos conjuntos independentes maximais de um grafo. Uma possibilidade é considerar outras operações em grafos e aproveitar suas propriedades, por exemplo a amálgama de dois grafos [Burlat e Fonlupt, 1984] a qual preserva o fato do grafo ser perfeito, propriedade que também satisfaz a união de dois grafos através de uma clique comum. É interessante analisar o que acontece nos grafos série-paralelo em geral, sem identificar os vértices terminais; pois a

propriedade de ser série-paralelo é hereditária mas ao identificar os vértices terminais já não é válida, todo subgrafo de um grafo série-paralelo dois terminais é série-paralelo mas os vértices terminais não são, necessariamente, os mesmos.

Também, falta estudar o problema de determinar o número de c.i.m. de grafos obtidos operando outros grafos menores. Poderia ser possível determinar uma relação entre o número de c.i.m. dos grafos menores e o número de c.i.m. do grafo resultante da operação.

A tabela abaixo resume as classes de grafos para as quais existem algoritmos, na literatura ou desenvolvidos neste trabalho (\* no ano), indicando-se a complexidade total e a referência.

GRAFO	COMPLEXIDADE	AUTOR	ANO
qualquer	$n \ m \ i(G)$	Tsukiyama et alii	1977
caminho induzido	$n \ i(G)$	Villanueva	*
ciclo induzido	$n \ i(G)$	Villanueva	*
grade completa $2 \times p$	$p \ i(G)$	Villanueva	*
intervalo	$n^2 + \beta$	Leung	1984
intervalo	$n \ (m + i(G))$	Villanueva	*
arco-circular	$n^2 + \beta$	Leung	1984
triangularizado	$(n + m) \ i(G)$	Leung	1984
triangularizado	$n \ (m + i(G))$	Villanueva	*
separável por cliques	$n^6 \ i(G_1) \ i(G_2)$	Villanueva	*
série-paralelo	$n^3 \ i(G_1) \ i(G_2)$	Villanueva	*

As possíveis linhas de pesquisa nesse tema são:

- Desenvolver algoritmos para enumerar todos os conjuntos independentes maximais em outras classes de grafos.
- Estudar o número de conjuntos independentes maximais em outras classes de grafos, determiná-lo exatamente ou por uma equação característica ou por limites superiores e inferiores.
- Caracterizar o grafo interseção  $I_G$  de conjuntos independentes maximais de um grafo  $G$  dado; isto é: existe um vértice em  $I_G$  por cada conjunto independente maximal de  $G$ , e existe uma arista unindo dois vértices de  $I_G$  se os c.i.m. correspondentes se interceptam.

- Caracterizar classes de grafos utilizando o grafo interseção de conjuntos independentes maximais do grafo, de forma análoga ao feito com o grafo interseção de cliques [Hedman - 1984, De Mello - 1992, Szwarcfiter - 1991, Szwarcfiter e Bornstein - 1994].

- Contar e enumerar todos os conjuntos independentes maximais de hipergrafos.



## Capítulo 7

### REFERÊNCIAS BIBLIOGRÁFICAS

- Akkoyunlu, E. A. (1973), "The Enumeration of Maximal Cliques of Large Graphs", *SIAM J. Comput.* 2:1, pp. 1-6.
- Augustson, J. G. & Minker, J. (1970), "An Analysis of Some Graph Theoretical Cluster Techniques", *J. Ass. Comput. Mach.* 17:4, pp. 571-588.
- Booth K.S & Leuker G.S. (1976) "Testing for the Consecutive Ones Property, Interval Graphs and Graph Planarity using PQ-tree Algorithms", *J. Comput. Syst. Sci.* 13, pp. 335-379.
- Boulala, M. & Uhry, J.P. (1979), "Polytope des Independants d'un Graphe Serie-parallele", *Discrete Math.* 27, pp. 225-243.
- Bron, C. & Kerbosch, J. (1973), "Finding All Cliques of an Undirected Graph", *Comm ACM* 16:9, pp. 575-577.
- Burlet, M. & Fonlupt, J. (1984), "Polynomial Algorithm to Recognize a Meyniel Graph", *Annals of Discrete Mathematics* 21, North- Holland, pp. 225-252.
- Burlet, M. & Uhry, J. (1984), "Parity Graphs", *Annals of Discrete Mathematics* 21, North-Holland, pp. 253-277.
- Cook, S. A. (1973), "A Hierarchy for Nondeterministic Time Complexity", *J. Comput. System Sci.* 7, pp. 343-353.
- Fulkerson, D. & Gross, O. (1965), "Incidence Matrices and Interval Graphs", *Pacific J. Math.* 15, pp. 835-855.
- Füredi, Z. (1987), "The Number of Maximal Independent Sets in Connected Graphs", *J. Graph Theory* 11:4, pp. 463-470.
- Garey, M. R. & D. S. Johnson (1979), *Computers and Intractability, A Guide to the Theory of NP-Completeness*, Ed. W. H. Freeman and company, Nova Yorque.
- Gavril, F. (1972), "Algorithms for Minimum Coloring, Maximum Clique, Minimum Covering by Cliques and Maximum independent Set of a Chordal Graph", *SIAM J. Comp.* 1, pp. 180-187.

- Gavril, F. (1977), "Algorithms on Clique Separable Graphs", *Discrete Math.* 19, pp. 159-165.
- Gilmore P. C. & Hoffman A. J. (1964) "A Characterization of Comparability Graphs and Interval Graphs", *Canadian J. Math.* 16, pp. 539-548.
- Griggs, J., Grinstead, C. & Guichard, D. (1988), "The Number of Maximal Independent Sets in a Connected Graph", *Discrete Math.* 68, pp. 211-220.
- Gupta, U., Lee, D. & Leung, J. (1982), "Efficient Algorithms for Interval Graphs and Circular-arc Graphs", *Networks* 12, pp. 459-467.
- Hedman, B. (1984), "Clique Graphs of Time Graphs", *J Comb. Th. B* 37, pp.270-278.
- Hujter, M. & Tuza, Z. (1993), "The Number of Maximal Independent Sets in Triangle-free Graphs", *SIAM J. Discrete Math.* 6:2, pp. 284-288.
- Johnson, D. S., Yannakakis, M. & Papadimitriou, C. H. (1988), "On Generating all Maximal Independent Sets", *Inf. Process. Lett.* 27, pp. 119-123.
- Kashiwabara, T., Masuda, S. Nakajima, K & Fujisawa, T. (1992), "Generation of Maximum Independent Sets of a Bipartite Graph and Maximum Cliques of a Circular-Arc Graph", *J. Algorithms* 13, pp. 161-174.
- Lawler, E. L. (1976a), "A Note on the Complexity of the Chromatic Number Problem", *Inf. Process. Lett.* 5:3, pp. 66-67.
- Lawler, E. L. (1976b), "Graphical Algorithms and Their Complexity", *Mathematical Centre Tracts* 81 (ed. K.R. Apt & J. W. de Bakker), Foundations of Computer Science II Part I, Mathematisch Centrum, Amsterdam, pp. 3-32.
- Lawler, E. L., Lenstra, J. K. & Rinnoy Kan, H. G. (1980), "Generating all Maximal Independent Sets: NP-Hardness and Polynomial-time Algorithms", *SIAM J. Comput.* 9, pp. 558-565.
- Leung, J. (1984), "Fast Algorithms for Generating All Maximal Independent Sets of Interval, Circular-arc and Chordal Graphs", *J. Algorithms* 5, pp. 22-35.
- Liu, J. (1993), "Maximal Independent Sets in Bipartite Graphs", *J. Graph Theory* 17:4, pp. 495-507.
- Liu, J. (1994), "Constraints on the Number of Maximal Independent Sets in Graphs", *J. Graph Theory* 18:2, pp. 195-204.

- Loukakis, E. & Tsouros, C. (1981), "A Depth First Search Algorithm to Generate the Family of Maximal Independent Sets of a Graph Lexicographically", *Computing* 27, pp. 349-366.
- Loukakis, E. (1983), "A New Backtracking Algorithm for Generating the Family of Maximal Independent Sets of a Graph", *Comp. and Maths. with Appls.* 9:4, pp. 583-589.
- Marcus, M. P. (1964), "Derivation of Maximal Compatibles Using Boolean Algebra", *IBM J. Res. Dev.* 8, pp. 537-538.
- Meir, A. & Moon, J. W. (1988), "On Maximal Independent Sets of Nodes in Trees", *J. Graph Theory* 12:2, pp. 265-283.
- de Mello, C. P. (1992), "Sobre Grafos Clique-completos", *Tese D.Sc. COPPE/UFRJ*.
- Moon, J. W. & Moser, L. (1965), "On Cliques in Graphs", *Israel J. Math* 3, pp. 23-28.
- Mulligan, G. D. & Corneil, D. G. (1972), "Corrections to Bierstone's Algorithm for Generating Cliques", *J. Ass. Comput. Mach.* 19:2, pp.244-247.
- Olariu S. (1991) "An optimal greedy heuristic to color interval graphs", *Inf. Process. Lett.* 37, pp. 21-25.
- Osteen, R. E. (1974), "Clique Detection Algorithms Based on Line Addition and Line Removal", *SIAM J. Appl. Math.* 26:1, pp.126-135.
- Paull, M. C. & Unger, S. H. (1959), "Minimizing the Number of States in Incompletely Specified Sequential Switching Functions", *IRE Trans. Electronic Comput.* EC-8, pp. 356-367.
- Rose, D. (1979), "Triangulated Graphs and the Elimination Process", *J. Math. An. Appl.* 32, pp. 597-609.
- Rose D., Tarjan R. & Leuker G. (1976), "Algorithmic Aspects of Vertex Elimination on Graphs", *SIAM J. Computing* 5, pp. 266-283.
- Sagan, B. E. (1988), "A Note on Independent Sets in Trees", *SIAM J. Discrete Math.* 1:1, pp. 105-108.
- Simon, J. (1975), "On Some Central Problems in Computational Complexity", *Tesis Ph.D. Universidade de Cornell*.
- Simon, J. (1977), "On the Difference Between the One and the Many", *Lecture Notes in Computer Science* 52, pp. 480-491.

Szwarcfiter, J. L. & M. M. Barroso (1991), "Enumerating the Maximal Cliques of a Circle Graph", em *Graph Theory, Combinatorics, Algorithms and Applications* (ed. Y. Alavi, F. R. K. Chung, R. L. Graham & D. F. Hsu), SIAM publications, pp. 511-517.

Szwarcfiter, J. L. (1991), 18° Colóquio Brasileiro de Matemática, IMPA.

Szwarcfiter, J. L. & Bornstein, C. F. (1994), "Clique Graphs of Chordal and Path Graphs", *SIAM J Disc. Math.*, pp. 331-336.

Tsukiyama, S., M. Ide, H. Ariyoshi & I. Shirakawa (1977), "A New Algorithm for Generating All The Maximal Independent Sets", *SIAM J. Comput.* 6:3, pp. 505-517.

Valiant, L. G. (1979), "The Complexity of Enumeration and Reliability Problems", *SIAM J. Comput.* 8:3, pp. 410-421.

Villanueva, M. (1993), "Conjuntos Independientes Maximales en Grafos de Intervalo", *XIII Congreso de Metodologías e Ingeniería de Sistemas*, Chile.

Villanueva, M. (1994), "Conjuntos Independientes Maximales en Grafos Separables por Cliques", *VII Congreso Latino-Ibero-Americano de Investigación Operativa CLAIO*, Chile.

Villanueva, M. (1995), "Número de Conjuntos Independientes Maximales en Grafos Reticulados", *I Congreso Chileno de Investigación Operativa OPTIMA 95*, Chile.

Whitesides, S. (1983), "An Algorithm for Finding Clique Cut-sets", *Inf. Process. Lett.* 12, pp. 31-32.

Whitesides, S. H. (1984), "A Method for Solving Certain Graph Recognition and Optimization Problems, with Applications to Perfect Graphs", *Annals of Discrete Mathematics* 21, North-Holland, pp. 281-297.

Wilf, H. (1986), "The Number of Maximal Independent Sets in a Tree", *SIAM J. Alg. Disc. Meth.* 7:1, pp. 125-130.

Zito, J. (1991), "The Structure and Maximum Number of Maximum Independent Sets in Trees", *J. Graph Theory* 15:2, pp. 207-221.