Emparelhamento em Grafos
Algoritmos e Complexidade

Celina M. Herrera de Figueiredo
Instituto de Matemática e COPPE/UFRJ
celina@cos.ufrj.br

Jayme L. Szwarcfiter
Instituto de Matemática,
Núcleo de Computação Eletrônica e COPPE/UFRJ
jayme@nce.ufrj.br

Resumo

Esta monografia considera o problema de emparelhamento em grafos. Encontrar um emparelhamento máximo em um grafo é um problema clássico no estudo de algoritmos, com muitas aplicações: designação de tarefas, determinação de rotas de veículos, problema do carteiro chinês, determinação de percursos mínimos, e muitos outros. O artigo histórico de Edmonds que descreveu um algoritmo $O(n^4)$ para o problema geral de emparelhamento, na verdade, introduziu a noção de algoritmo de tempo polinomial. Implementações mais eficientes do algoritmo de Edmonds foram apresentadas por vários pesquisadores como Lawler (algoritmo $O(n^3)$), Hopcroft e Karp (algoritmo $O(m\sqrt{n})$ para o caso bipartido), Micali e Vazirani (algoritmo $O(m\sqrt{n})$ para o caso geral, o mais eficiente até o momento). Este texto apresenta uma introdução ao problema de emparelhamento em grafos, e um estudo de seus algoritmos e complexidade. Apresentamos algoritmos eficientes para os quatro problemas relacionados com encontrar um emparelhamento de cardinalidade máxima ou de peso máximo, em grafos bipartidos ou em grafos quaisquer. Estes quatro problemas são todos casos particulares do problema de emparelhamento de peso máximo em grafos quaisquer, mas é interessante considerá-los em ordem crescente de dificuldade por razões de ordem didática.

Abstract

In this text it is considered the problem of matchings in graphs. Finding a maximum matching in a graph is a classical problem in the study of algorithms, with many applications. Among these applications it can be mentioned schedulling of jobs, vehicle routing, chinese postman problem, shortest paths and many others. The seminal article by Edmonds, which described an $O(n^4)$ algorithm for finding a maximum matching in a general graph, in fact introducedthe notion of polynomial time algorithms, for the entire area of computer science. Efficient implementations of Edmond's algorithm were presented by different authors, such as Lawler ($O(n^3)$ algorithm), Hopcroft and Karp ($O(m\sqrt{n})$ algorithm for the bipartite case), Micali and Vazirani ($O(m\sqrt{n})$ algorithm for the general case, so far, the most efficient). This text presents an introduction of matching problem and a study of its algorithms and their complexities. It is described efficient algorithms for the four related problems which are: finding cardinality maximum matchings and weighted maximum mathchings in bipartite and general graphs. Clearly, the problem of finding a maximum weighted matching in a general graph contains the other three. However, for a better understanding of the algorithms, it is interesting to consider them in increasing order of difficulty.

1 O problema de emparelhamento em grafos

Dado um grafo G com conjunto de vértices V e conjunto de arestas E, um emparelhamento M é um subconjunto de E de arestas duas a duas não adjacentes. Dois problemas são de interesse: encontrar um emparelhamento de cardinalidade máxima, isto é, com número máximo de arestas; e encontrar um emparelhamento de peso máximo. Neste segundo caso, é dada, além do grafo, uma função que associa a cada aresta um peso não negativo.

Apresentamos algoritmos eficientes para os quatro problemas, de determinação de emparelhamentos de cardinalidade máxima ou de peso máximo, em grafos bipartidos ou em grafos quaisquer. Observe que estes quatro problemas são todos casos particulares do problema de emparelhamento de peso máximo em grafos quaisquer. Entretanto é interessante considerá-los em ordem crescente de dificuldade porque quanto

mais simples for o problema, sua solução poderá ser mais rápida ou mais simples.

Na Seção 1 apresentamos uma introdução ao problema de emparelhamento em grafos e à notação utilizada neste texto. Na Seção 2 discutimos aplicações e a relação do problema em questão com o de fluxo máximo em redes. Na Seção 3 apresentamos teoremas básicos que constituem o arcabouço teórico. Na Seção 4 consideramos o caso bipartido sem pesos. Na Seção 5 consideramos o caso bipartido com pesos. Na Seção 6 consideramos o caso geral. Na Seção 7 estão as notas bibliográficas. Finalmente, sugerimos na Seção 8, alguns exercícios.

Apresentação da notação e dos quatro problemas

Um *grafo* $G = (V, E)$ consiste de um conjunto finito não vazio V de *vértices* e de um conjunto E de *arestas*, onde cada aresta é um par não ordenado de vértices distintos. Denotaremos o conjunto de *vértices* de um grafo G por $V(G)$, ou simplesmente por V caso não haja ambigüidade.

Dizemos que um grafo é *trivial* quando só possui um vértice.

Dada uma aresta $e = xy \in E$, dizemos que x e y são as *extremidades* de e. Neste caso, x e y são ditos *adjacentes* ou *vizinhos* e dizemos que x *vê* y.

O conjunto de vértices adjacentes a um dado vértice v em um grafo G é chamado a *vizinhança* de v em G e será denotado por $N_G(v)$, ou simplesmente por $N(v)$ caso não haja ambigüidade.

Analogamente, denotaremos por $N_G(T)$ a *vizinhança* de um conjunto T de vértices no grafo G, isto é, o conjunto de vértices de G adjacentes a algum vértice de T.

Um grafo $H = (W, F)$ é dito um *subgrafo* de um grafo $G = (V, E)$ caso $W \subseteq V$ e $F \subseteq E$, e denotaremos por $H \subseteq G$. Um subgrafo $H \subseteq G$ é *gerador* se H contém todos os vértices do grafo G. Dado um conjunto de vértices $W \subseteq V$, dizemos que um subgrafo $H = (W, F)$ de um grafo $G = (V, E)$ é *induzido* por W se toda aresta de G com extremidades em W pertence a F. Denotaremos por $H = G[W]$, o subgrafo $H \subseteq G$ induzido por $W \subseteq V$. Analogamente, dado um conjunto de arestas $F \subseteq E$, definimos o subgrafo $H = G[F]$ *induzido* por F.

O grafo $G \setminus v$, obtido do grafo G pela remoção de um vértice v, é o subgrafo induzido pelo conjunto $V \setminus \{v\}$. Analogamente, o grafo $G \setminus H$, obtido do grafo G pela remoção de um subgrafo H, é o subgrafo induzido pelo conjunto $V(G) \setminus V(H)$.

Dados dois vértices x e y de um grafo $G = (V, E)$, definimos um *caminho* entre x e y como uma seqüência $P = v_0 v_1 \dots v_k$, onde $x = v_0$, $v_k = y$, e onde os vértices v_i são distintos e $v_i v_{i+1} \in E$.

Um *ciclo* é uma seqüência $C = v_0 v_1 \dots v_k$ tal que $v_0 v_1 \dots v_{k-1}$ é um caminho, $v_0 = v_k$ e $k \geq 3$.

O *tamanho* de um caminho é o seu número de arestas. O *tamanho* de um ciclo é o seu número de vértices.

A *distância* entre dois vértices numa mesma componente conexa de um grafo é o tamanho de um menor caminho entre eles.

Dado um grafo $G = (V, E)$, denotaremos por n o número de vértices $|V|$ e por m o número de arestas $|E|$.

Um *grafo direcionado* $G = (V, E)$ consiste de um grafo com uma orientação no seu conjunto de arestas, isto é, cada aresta é um par ordenado de vértices distintos. Denotamos uma aresta orientada num grafo direcionado por (x, y).

Consideramos como entrada para o problema de emparelhamento um grafo não direcionado $G = (V, E)$ com $|V| = n$ e $|E| = m$. Os vértices representam pessoas e cada aresta representa a possibilidade de casamento entre as pessoas correspondentes às suas extremidades. Um *emparelhamento* M é um subconjunto de conjunto de arestas tal que nenhum par de arestas em M tem uma extremidade comum, i.e., não permitimos poligamia.

Num grafo $G = (V, E)$ com um emparelhamento $M \subseteq E$, dizemos que os vértices extremos de uma aresta $e \in M$ estão *emparelhados* por M ou simplesmente *M-emparelhados*. Um emparelhamento M *satura* um vértice v e v é dito *M-saturado* se $e \in M$ é incidente a v; caso contrário, v é *não M-saturado* ou *livre*.

Observe que o conjunto vazio define um emparelhamento. Observe que se $M' \subseteq M$ e M é um emparelhamento, então M' também define um emparelhamento.

Um emparelhamento M é *máximo* em G se M contém o maior número possível de arestas, i.e., G não admite emparelhamento M'

com $|M'| > |M|$. Dizemos também que M é emparelhamento *de cardinalidade máxima*, neste caso. Um emparelhamento é *perfeito*, se cada vértice $v \in V$ é incidente a alguma aresta de M. Observe que todo emparelhamento perfeito é máximo, e que num grafo $G(V, E)$ com emparelhamento perfeito, $|V|$ é par e um emparelhamento perfeito possui exatamente $|V|/2$ arestas. Num emparelhamento perfeito M, todo vértice encontra-se M-saturado.

Considere agora um grafo G junto com um emparelhamento fixo M de G. As arestas em M são ditas *arestas emparelhadas*, enquanto que as arestas restantes são ditas *arestas livres*. Se uv é uma aresta emparelhada, então u é *cônjuge* de v. Vértices incidentes a uma aresta emparelhada são ditos *emparelhados*, enquanto que os vértices restantes são ditos *solteiros*. Um caminho $P = u_1u_2 \dots u_k$ é dito *alternante* caso as arestas u_1u_2, u_3u_4, ..., $u_{2j-1}u_{2j}$, ... sejam livres, enquanto que as arestas restantes u_2u_3, u_4u_5, ..., $u_{2j}u_{2j+1}$, ... são emparelhadas. Um caminho alternante $P = u_1u_2 \dots u_k$ é dito *aumentante* caso tanto u_1 quanto u_k sejam solteiros.

Observe que a idéia de aumentação está presente nos algoritmos clássicos para o problema de fluxo máximo em redes. Na verdade, em alguns casos, mostraremos como usar estes algoritmos para o problema de fluxo máximo em redes, para resolver o problema de emparelhamento. O que ocorre no caso de emparelhamento é que encontrar e efetuar aumentações eficientemente pode ser extremamente sutil. Tanto no caso sem pesos quanto no caso com pesos, estes dois aspectos são simplificados consideravelmente quando o grafo subjacente é bipartido.

Os quatro problemas considerados neste texto são:

Problema 1: Emparelhamento de cardinalidade máxima em grafos bipartidos

Os vértices são particionados em homens e mulheres e uma aresta só pode ligar um homem e uma mulher. Procuramos por um emparelhamento de cardinalidade máxima, i.e., com número máximo de arestas.

Podemos tornar o Problema 1 mais difícil em duas direções diferentes, o que resulta respectivamente nos Problemas 2 e 3.

Problema 2: Emparelhamento de cardinalidade máxima em grafos quaisquer

Este é o caso assexuado onde uma aresta liga duas pessoas. Procuramos por um emparelhamento de cardinalidade máxima, i.e., com número máximo de arestas.

Problema 3: Emparelhamento de peso máximo em grafos bipartidos

Aqui temos vértices representando homens e mulheres, arestas só podem ligar homens e mulheres, mas cada aresta tem ij tem um peso $w(ij)$ associado. O objetivo é encontrar um emparelhamento com peso total máximo, i.e., com soma máxima total dos pesos das arestas escolhidas. Este é o problema conhecido de alocação onde alocamos pessoas a tarefas maximizando o lucro ou a produtividade.

Problema 4: Emparelhamento de peso máximo em grafos quaisquer

Este é o problema obtido do Problema 1, tornando-o mais difícil nas duas direções simultaneamente.

2 Aplicações e relação com problemas em grafos

Apresentamos na Seção 2.1 algumas aplicações do problema de emparelhamento. Na Seção 2.2 consideramos a relação do problema de emparelhamento com o problema de fluxo máximo em redes.

2.1 Aplicações

Encontrar um emparelhamento máximo em um grafo é um problema clássico para o estudo de algoritmos. Vários problemas podem ser modelados como problemas de emparelhamento em grafos, como, por exemplo, os seguintes.

Problema de atribuição de pessoal

O problema de atribuição ou de alocação de pessoal tem como dados n funcionários e n posições numa empresa. Cada funcionário está qualificado para uma ou mais posições. É possível atribuir uma posição a

cada funcionário, de modo que cada funcionário ocupe exatamente uma posição na empresa?

O problema de atribuição ou de alocação ótima de pessoal pode apresentar como dados, adicionalmente, uma função que atribui a cada funcionário um valor numérico, correspondente à sua eficiência para ocupar determinada posição na empresa. O objetivo agora é encontrar uma atribuição ou uma alocação que maximize a eficiência total dos funcionários.

O primeiro caso corresponde ao problema de emparelhamento perfeito em grafos bipartidos, enquanto que o segundo caso é o problema do emparelhamento máximo em grafos bipartidos com pesos.

Problema dos casamentos

Dada uma matriz onde cada entrada é 0 ou 1, um conjunto de entradas é *independente* se não temos duas entradas na mesma linha ou na mesma coluna. Pede-se encontrar um conjunto independente de entradas de valor 1 que tem cardinalidade máxima. Podemos intepretar as linhas da matriz como sendo rapazes e as colunas como sendo moças. Uma entrada i, j com valor 1 seria o caso de rapaz e moça compatíveis. O objetivo é casar um número máximo de casais compatíveis.

Este problema corresponde ao problema de emparelhamento de cardinalidade máxima em grafos bipartidos.

Problema de construção de amostras

Um vendedor de brinquedos educativos possui em estoque brinquedos de várias formas geométricas (cubos, pirâmides, etc.), cada qual fabricado em várias cores. O vendedor quer carregar consigo o menor número de objetos tal que cada cor e cada forma estejam representadas pelo menos uma vez.

O vendedor constrói o seguinte grafo: cada forma e cada cor estão representados individualmente por um vértice e existe uma aresta ligando um vértice-forma a um vértice-cor, caso aquela forma geométrica seja fabricada naquela cor.

O número mínimo de objetos que o vendedor precisa carregar é igual ao número de arestas numa *cobertura de cardinalidade mínima*: um

conjunto de arestas C tal que cada vértice do grafo é extremo de alguma aresta em C, e este conjunto C de arestas é o de menor cardinalidade em relação a esta propriedade.

Consideramos no Exercício 10 uma construção que prova a equivalência entre o problema de cobertura de cardinalidade mínima e o problema de emparelhamento de cardinalidade máxima.

2.2 Relação com o problema de fluxo máximo em redes

O problema de emparelhamento está fortemente relacionado ao de fluxo máximo em redes, conforme detalhado a seguir.

Seguimos as definições e notação de [15]. Uma *rede* é um grafo direcionado $D = (V, E)$ em que a cada aresta direcionada $e \in E$ está associado um número real positivo $c(e)$ denominado *capacidade* da aresta e. Suponha que D possua dois vértices especiais e distintos $s, t \in V$ chamados *origem* e *destino*, respectivamente, com as seguintes propriedades: a origem s é uma fonte que alcança todos os vértices; o destino t é um sumidouro alcançado por todos os vértices. Um *fluxo* f de s a t em D é uma função que a cada aresta $e \in E$ associa um número real não negativo $f(e)$ satisfazendo às condições abaixo:

- $0 \leq f(e) \leq c(e)$, para toda aresta $e \in E$;
- $\sum_{w_1} f(w_1, v) = \sum_{w_2} f(v, w_2)$, para todo vértice $v \neq s, t$, sendo este valor denominado *valor do fluxo no vértice* v.

Por analogia, o somatório dos fluxos das arestas divergentes de s é chamado de *valor do fluxo* em s. O *valor do fluxo* f *na rede* D é definido como o valor do fluxo em s.

Considere um grafo $G(V, E)$ bipartido com bipartição do conjunto de vértices V em conjuntos X, Y. A seguinte construção reduz o problema de emparelhamento de cardinalidade máxima, neste caso, para um problema de fluxo máximo em redes.

Construa uma rede $D(V', E')$ a partir deste grafo bipartido G da seguinte maneira:

1. O conjunto de vértices V' de D é obtido adicionando a V dois vértices auxiliares s, t, que serão respectivamente fonte e sumidouro na rede D.

2. O conjunto de arestas direcionadas E' de D é obtido orientando as arestas de E e definindo arestas direcionadas a partir de s e chegando em t. Para cada $x \in X$, adicione ao conjunto E' a aresta direcionada (s, x) e para cada $y \in Y$, adicione ao conjunto E' a aresta direcionada (y, t);

3. Se $xy \in E$, com $x \in X$, e $y \in Y$, então adicione ao conjunto E' a aresta direcionada (x, y);

4. Todas as arestas desta rede $D(V', E')$ possuem capacidade 1.

Seja f um fluxo em D. Seja M' o conjunto de arestas direcionadas de E' saturadas por f que tem uma extremidade em X e a outra extremidade em Y. Considere um vértice $x \in X$. Como a soma das capacidades das arestas que chegam em x é 1, o máximo valor de fluxo possível em x é 1. Portanto, no máximo uma aresta direcionada (x, y), para algum $y \in Y$, encontra-se saturada por f. Este conjunto M' de arestas direcionadas saturadas corresponde a um conjunto de arestas M de E. Este conjunto M define um emparelhamento no grafo bipartido G, já que não podemos ter em M duas arestas incidentes a um mesmo vértice $x \in X$, e não podemos ter em M duas arestas incidentes a um mesmo vértice $y \in Y$. Observe que a cardinalidade do emparelhamento M é o valor do fluxo f.

Por outro lado, seja M um emparelhamento em G. Este conjunto de arestas define o seguinte fluxo f em D: se a aresta $xy \in M$, então definimos o fluxo nas seguintes três arestas direcionadas por $f(s, x) = 1$, $f(x, y) = 1$, $f(y, t) = 1$. Para todas as outras arestas direcionadas $e \in E'$, definimos $f(e) = 0$. Como M é um emparelhamento, o fluxo f assim definido em D se conserva em cada vértice $v \in X \cup Y$. Como todas as capacidades são 1 em D, o fluxo f em cada aresta assim definido respeita a capacidade da aresta. Observe que o valor deste fluxo f é a cardinalidade do emparelhamento M.

Portanto, existe uma correspondência entre emparelhamentos M em G e fluxos f em D, de forma que um fluxo f define um emparelhamento M cuja cardinalidade é o valor do fluxo f e reciprocamente, um emparelhamento M define um fluxo f cujo valor é a cardinalidade de M.

Logo, um fluxo máximo em D define um emparelhamento máximo em G e reduzimos desta forma o problema do emparelhamento máximo em grafos bipartidos a um problema de fluxo máximo em redes.

Algoritmo $O(m\sqrt{n})$ para redes com capacidades unitárias

Um algoritmo para fluxo máximo em redes resolve o problema reduzindo-o a $O(n)$ problemas de fluxo maximal numa *rede de camadas*, i.e., uma rede auxiliar onde os vértices são dispostos em camadas, segundo a sua distância até a origem s. Cada fluxo maximal numa rede de camadas é encontrado por sua vez em tempo $O(n^2)$. Para detalhes sobre este algoritmo de tempo total $O(n^3)$, veja [15].

Observe que a rede auxiliar construída para a solução do problema de emparelhamento em grafos bipartidos tem características particulares que podem ser exploradas para justificar que este algoritmo $O(n^3)$ tem na verdade limite superior $O(m\sqrt{n})$ neste caso.

A primeira característica é que a rede é de *capacidades unitárias*, i.e., toda aresta direcionada na rede tem capacidade 1. Portanto, cada aresta direcionada ao ser considerada num caminho aumentante fica saturada imeditamente. Este fato fornece um limite superior de $O(m)$ para encontrar um fluxo maximal numa rede de camadas.

A segunda característica é que a rede é *simples*, i.e., todo vértice tem grau de entrada 1 ou 0, ou grau de saída 1 ou 0. Numa rede simples com valor de fluxo máximo F, temos no máximo n/F problemas de redes de camadas. De fato, denote por V_i o conjunto de vértices à distância i da fonte s. Como todo o fluxo F passa por cada camada V_i, e como cada vértice tem valor máximo de fluxo 1, segue que $|V_i| \geq F$. Logo, $n \geq \sum_{i=1} |V_i| \geq \ell F$, onde ℓ é o número de camadas.

Observe que temos neste caso $O(\sqrt{n})$ redes de camadas. Cada estágio onde é encontrado um fluxo maximal numa rede de camadas aumenta o valor de fluxo corrente de pelo menos uma unidade. Logo temos no máximo F estágios. Logo, ou $F \leq \sqrt{n}$, ou $F > \sqrt{n}$. Nesta segunda possibilidade, como o número de estágios é no máximo n/F, temos garantido o limite superior de $\sqrt{n}$ estágios.

Obtemos portanto um algoritmo $O(m\sqrt{n})$, para o problema de fluxo máximo em redes no caso de uma rede simples com capacidades unitárias. Este algoritmo fornece, por sua vez, um algoritmo $O(m\sqrt{n})$ para o pro-

blema de emparelhamento de cardinalidade máxima em grafos bipartidos.

3 Arcabouço teórico

Dois teoremas constituem a base teórica para os algoritmos de emparelhamento. O Teorema de Berge caracteriza a maximalidade de um emparelhamento M em termos da existência de caminhos M-aumentantes. O Teorema de Hall caracteriza e existência de emparelhamentos perfeitos num grafo bipartido.

3.1 Caminhos aumentantes: Teorema de Berge

Dado um emparelhamento M para G, um *caminho M-alternante* em G é um caminho cujas arestas estão alternadamente em $E \setminus M$ e em M. Um *caminho M-aumentante* é um caminho M-alternante cuja origem e término são não M-saturados. Vértices num caminho M-aumentante com posição ímpar são chamados de *externos* enquanto que os com posição par são chamados de *internos*.

Teorema 1 (Berge, 1957) *Um emparelhamento M tem cardinalidade máxima em G se e somente se G não possui caminho M-aumentante.*

Prova: Seja M um emparelhamento em G. Suponha que G possui um caminho M-aumentante P. Observe que P tem, por definição, um número par de vértices, digamos $P = v_0v_1 \dots v_{2m+1}$.
Defina $M' \subseteq E$ por

$$M' = M \setminus \{v_1v_2, v_3v_4, \dots, v_{2m-1}v_{2m}\} \cup \{v_0v_1, v_2v_3, \dots, v_{2m}v_{2m+1}\}.$$

Temos que M' é um emparelhamento em G com $|M'| = |M| + 1$ e portanto M não tem cardinalidade máxima em G. Logo, se M tem cardinalidade máxima em G, então G não possui caminho M-aumentante.

Por outro lado, suponha que M não tem cardinalidade máxima em G. Seja M' um emparelhamento de cardinalidade máxima em G. Logo $|M'| > |M|$. Denote por $M\Delta M'$ a *diferença simétrica* de M e M', i.e., o conjunto das arestas que estão em $M \cup M'$ mas não estão

em $M \cap M'$. Seja $H = G[M \Delta M']$. Cada vértice em H tem grau 1 ou 2, já que pode ser incidente a no máximo uma aresta de M e a uma aresta de M'. Logo cada componente conexa de H é um ciclo par com arestas alternadamente em M e M' ou um caminho com arestas alternadamente em M e M'. Mas H tem mais arestas de M' do que de M e portanto alguma componente que é um caminho Q em H deve começar e terminar com arestas de M'. Como a origem e o término de Q são M'-saturados em H, segue que são vértices não M-saturados em G. Logo Q é o caminho M-aumentante procurado. Portanto, se M não tem cardinalidade máxima em G, então G possui caminho M-aumentante. ■

Um algoritmo natural para encontrar um emparelhamento de cardinalidade máxima que decorre do Teorema 1 é começar com um emparelhamento vazio e, repetidamente, aumentar a cardinalidade do emparelhamento corrente através do uso sucessivo de caminhos aumentantes. Observe que a existência de um caminho M-aumentante P define através da operação diferença simétrica um novo emparelhamento $M' = M \Delta P$, onde $|M'| = |M| + 1$. Este processo de sucessivas aumentações termina com um emparelhamento de cardinalidade máxima porque:

- A cardinalidade do emparelhamento de cardinalidade máxima é finita;
- Cada iteração aumenta de uma unidade a cardinalidade do emparelhamento corrente.

A complexidade do algoritmo acima é função da procura por caminhos aumentantes. Observe que em [15], no capítulo sobre algoritmos para fluxo máximo em redes, foi justamente o estudo da procura por caminhos aumentantes na rede residual que permitiu a descrição de algoritmos cada vez mais eficientes para aquele problema.

3.2 Emparelhamentos perfeitos: Teorema de Hall

Dado um conjunto S de vértices em G, definimos a *vizinhança* de S em G como o conjunto de todos os vértices adjacentes a vértices de S e denotamos por $N(S)$.

Considere o grafo bipartido $G = (V, E)$ com bipartição X, Y. Em muitas aplicações, queremos saber se o grafo bipartido G admite um emparelhamento que satura todo vértice de X. Este problema pode ser utilizado, por exemplo, para modelar o seguinte problema.

Exemplo 1 *Numa empresa, n funcionários $x_1, x_2, \ldots, x_n$ se candidatam para uma ou mais dentre n tarefas $y_1, y_2, \ldots, y_n$. É possível alocar todos os funcionários atribuindo uma tarefa diferente para cada funcionário, respeitando as habilidades dos funcionários?*

O teorema abaixo descreve condições necessárias e suficientes para a existência de emparelhamentos do tipo desejado.

Teorema 2 (Hall, 1956) *Seja G um grafo bipartido com bipartição X, Y. Então G admite emparelhamento que satura todo vértice de X se e somente se $|N(S)| \geq |S|$, para todo $S \subseteq X$.*

Prova: Suponha que G contém um emparelhamento M que satura todo vértice em X e seja S um subconjunto de X. Como os vértices em S estão emparelhados em M com vértices distintos em $N(S)$, nós temos $|N(S)| \geq |S|$.

Por outro lado, suponha que G é um grafo bipartido que satisfaz $|N(S)| \geq |S|$, para todo $S \subseteq X$, mas que G não tem emparelhamento que satura todo vértice de X. Seja M um emparelhamento máximo em G. Por hipótese, M não satura todo vértice de X. Seja u um vértice não M-saturado em X e seja Z o conjunto dos vértices que são atingíveis a partir de u por caminhos M-alternantes. Como M é um emparelhamento máximo, temos que u é o único vértice não M-saturado em Z. Defina $S = Z \cap X$ e $T = Z \cap Y$. Como os vértices em $S \setminus \{u\}$ estão emparelhados por M com vértices de T, temos $|T| = |S| - 1$. Além disso, temos $T \subseteq N(S)$, já que todo vértice de T, estando num caminho M-alternante a partir de u, é vizinho de u ou é vizinho de algum vértice em X que está num caminho M-alternante a partir de u, i.e., é vizinho de algum vértice de S. Na verdade, $N(S) \subseteq T$, já que todo vértice em $N(S)$, ou é vizinho de u ou é vizinho de algum vértice de $S \setminus \{u\}$ e portanto está conectado a u por um caminho M-alternante. Logo $|N(S)| = |T| = |S| - 1 < |S|$, o que contradiz a hipótese. ■

O corolário abaixo segue imeditamente do Teorema 2 e caracteriza a existência de emparelhamentos perfeitos em grafos bipartidos. No Exercício 7, consideramos uma caracterização para a existência de emparelhamentos perfeitos em grafos quaisquer.

Corolário 1 *Seja G um grafo bipartido com bipartição X, Y. Então G admite emparelhamento perfeito se e somente se $|X| = |Y|$ e $|N(S)| \geq |S|$, para todo $S \subseteq X$.*

Uma outra caracterização para a existência de emparelhamentos perfeitos em grafos bipartidos que também segue do Teorema 2 é a seguinte.

Corolário 2 *Um grafo bipartido tem um emparelhamento perfeito se e somente se $|N(S)| \geq |S|$, para todo $S \subseteq V$.*

Prova: É deixada como exercício e sugerida para o leitor no Exercício 4. ■

O caso particular de grafo bipartido *regular*, i.e., onde todos os vértices têm o mesmo grau, pode ser utilizado para modelar o seguinte problema.

Exemplo 2 *Numa cidade, toda moça conhece exatamente k rapazes e todo rapaz conhece exatamente k moças. É possível que cada moça se case com um rapaz que ela conhece e que cada rapaz se case com uma moça que ele conhece?*

É interessante observar que o fato de um grafo bipartido regular sempre admitir emparelhamento perfeito também pode ser obtido como consequência do Teorema 2.

Corolário 3 *Se G é um grafo bipartido k-regular, com $k > 0$, então G tem um emparelhamento perfeito.*

Prova: Seja $G = (V, E)$ um grafo bipartido k-regular com bipartição X, Y. Como G é k-regular, o número de arestas de G é $|E| = k|X| = k|Y|$ e como $k > 0$, segue que $|X| = |Y|$. Agora seja $S \subseteq X$ e sejam E_1 as arestas incidentes a S e E_2 as arestas incidentes a $N(S)$. Por

definição de $N(S)$, toda aresta incidente a S é também incidente a $N(S)$, e temos que $E_1 \subseteq E_2$. Portanto, $k|S| = |E_1| \leq |E_2| = k|N(S)|$, o que dá $|S| \leq |N(S)|$ e garante pelo Teorema 2 que G tem um emparelhamento que satura todo vértice em X. Como $|X| = |Y|$, este emparelhamento é perfeito. ■

4 Grafo bipartido sem pesos: Algoritmo Húngaro

Numa certa empresa, temos n funcionários x_1, x_2, ..., x_n disponíveis para n tarefas y_1, y_2, ..., y_n. Cada funcionário está qualificado para uma ou mais tarefas. O problema é encontrar, caso exista, uma alocação dos funcionários que respeite as suas qualificações e que atribua exatamente uma tarefa para cada funcionário. Em outras palavras, o problema consiste em encontrar, caso exista, um emparelhamento perfeito neste grafo bipartido.

Dado um grafo $G = (V, E)$ com $2n$ vértices, com conjunto de vértices V bipartido em conjuntos X e Y, com $|X| = |Y| = n$, consideramos nesta seção duas variações do problema de emparelhamento de cardinalidade máxima para grafos bipartidos. Descrevemos primeiro o chamado Algoritmo Húngaro que, em tempo $O(nm)$, ou encontra um emparelhamento que satura todo vértice de X ou encontra um subconjunto de X que viola a condição do Teorema de Hall. Em particular, o Algoritmo Húngaro decide em tempo $O(nm)$ se este grafo admite emparelhamento perfeito. Em seguida, descrevemos uma variação do Algoritmo Húngaro que, em tempo $O(nm)$, encontra um emparelhamento de cardinalidade máxima neste caso.

Os Teoremas 1 e 2, discutidos na Seção 3, justificam a correção do seguinte algoritmo: comece com um emparelhamento M. Caso M não sature o conjunto X, procure um caminho M-aumentante P a partir de $u \in X$, um vértice não M-saturado. Caso P exista, defina $M^* = M\Delta E(P)$. Caso P não exista, encontre o conjunto Z dos vértices alcançáveis a partir de u por caminhos M-alternantes e teremos o conjunto $S = Z \cap X$ que dá a condição de parada $|N(S)| < |S|$.

A eficiência deste algoritmo decorre do seu método de procura por caminhos aumentantes. Seja M um emparelhamento em G e seja u

um vértice não M-saturado em X. Uma árvore $H \subseteq G$ é dita *árvore M-alternante com raiz u* se:

- $u \in V(H)$;
- Para todo $v \in V(H)$, o caminho de u a v em H é um caminho M-alternante.

O Algoritmo Húngaro procura um caminho M-aumentante a partir de u, construindo uma árvore M-alternante H com raiz u.

Inicialmente, H consiste de um único vértice u. Naturalmente, uma dentre as seguintes condições ocorre, ao longo da construção da árvore.

(i) Todos os vértices de H, exceto u, são M-saturados; ou

(ii) H contém um vértice não M-saturado diferente de u.

Se temos o caso (i), que de fato é o caso inicial, então definindo os conjuntos de vértices $S = V(H) \cap X$ e $T = V(H) \cap Y$, obtemos $T \subseteq N(S)$; e podemos distinguir dois casos:

(a) Se $N(S) = T$, então como os vértices em $S \setminus \{u\}$ são M-emparelhados com vértices em T, temos $|N(S)| = |S| - 1$, o que indica pela Condição de Hall que G não tem emparelhamento que satura todos os vértices em X.

(b) Se $T \subset N(S)$, então existe $y \in Y \setminus T$ adjacente a um vértice $x \in S$. Como todos os vértices de H a menos de u são M-emparelhados, ou $x = u$ ou x é emparelhado com um vértice de H. Portanto $xy \notin M$. Se y é M-saturado, com $yz \in M$, observe que $z \in X$ e $z \notin S$, porque todos os vértices de S a menos de u encontram-se emparelhados com vértices de T. Aumentamos a árvore H ao adicionar a H os vértices y e z, e as arestas xy e yz. Estamos de novo no caso (i). Se y é não M-saturado, aumentamos H ao adicionar a H o vértice y e a aresta xy, resultando no caso (ii). Observe que, neste último caso, o caminho de u até y em H é um caminho M-aumentante com origem u, como desejado.

O Algoritmo Húngaro pode então ser resumido nos seguintes passos:

1. Comece com um emparelhamento arbitrário M.

2. Se M satura todo vértice de X, então PARE. Caso contrário, seja u um vértice não M-saturado em X. Defina os conjuntos $S = \{u\}$ e $T = \emptyset$.

3. Se $N(S) = T$, então $|N(S)| < |S|$ porque $|T| = |S| - 1$. PARE porque o Teorema 2 diz que não temos emparelhamento que satura todos os vértices em X. Caso contrário, seja $y \in N(S) \setminus T$.

4. Se y é M-saturado, então seja $yz \in M$. Atualize $S \leftarrow S \cup \{z\}$, $T \leftarrow T \cup \{y\}$ e vá para o Passo 3. Caso contrário, seja P um caminho M-aumentante entre u e y. Atualize $M \leftarrow M \Delta P$ e vá para o Passo 2.

Algoritmo Húngaro
Dados: grafo G bipartido com emparelhamento M
Saída: emparelhamento perfeito M ou conjunto S, violando Condição de Hall

enquanto X não M-saturado faça
    $u \leftarrow$ vértice não M-saturado em X
    $S \leftarrow \{u\}$
    $T \leftarrow \emptyset$
    enquanto $N(S) \neq T$ e todo $y \in N(S) \setminus T$ é M-saturado faça
        $yz \leftarrow$ aresta de M
        $S \leftarrow S \cup \{z\}$
        $T \leftarrow T \cup \{y\}$
    se $N(S) = T$ então
        retorne S viola condição do Teorema 2
    senão
        $y \leftarrow$ vértice em $N(S) \setminus T$ não M-saturado
        $P \leftarrow$ caminho aumentante entre u e y
        $M \leftarrow M \Delta P$
retorne M

O Algoritmo Húngaro não constrói explicitamente a árvore aumentante, mas esta pode ser obtida diretamente a partir do algoritmo, se desejado.

Observe que o Algoritmo Húngaro não termina necessariamente com um emparelhamento máximo. O Algoritmo Húngaro termina ou com

um emparelhamento perfeito e, neste caso, termina com um emparelhamento máximo, ou no caso do grafo não admitir um emparelhamento perfeito, o Algoritmo Húngaro termina com um emparelhamento M que é *maximal* no sentido de que o último vértice u não M-saturado considerado não é extremo de caminho M-aumentante.

Uma pequena modificação do Algoritmo Húngaro fornece um emparelhamento de cardinalidade máxima num grafo bipartido. Dado uma grafo bipartido G com emparelhamento M, basta executarmos o Algoritmo Húngaro para cada vértice não M-saturado. Os detalhes são pedidos ao leitor no Exercício 6.

Exemplo

Consideremos um exemplo da aplicação do Algoritmo Húngaro.
Seja $G = (V, E)$ bipartido, onde $V = X \cup Y$, e

$$\begin{aligned} X &= \{x_1, x_2, x_3, x_4, x_5\} \\ Y &= \{y_1, y_2, y_3, y_4, y_5\}, \\ E &= \{x_1y_2, x_1y_3, x_2y_1, x_2y_2, x_2y_4, x_2y_5, x_3y_2, x_3y_3, x_4y_2, x_4y_3, x_5y_5\} \end{aligned}$$

Começamos com o emparelhamento $M = \{x_2y_2, x_3y_3, x_5y_5\}$. Construímos a árvore aumentante H a partir do vértice x_1 da seguinte forma: inicializamos $S = \{x_1\}$ e $T = \emptyset$. Seja $y_2 \in N(S) \setminus T$. Como a aresta $y_2x_2 \in M$, atualizamos $S = \{x_1, x_2\}$ e $T = \{y_2\}$. Em seguida, seja $y_3 \in N(S) \setminus T$. Como a aresta $y_3x_3 \in M$, atualizamos $S = \{x_1, x_2, x_3\}$ e $T = \{y_2, y_3\}$. Seja $y_1 \in N(S) \setminus T$. Como y_1 não é M-saturado, temos o caminho aumentante $P = x_1y_2x_2y_1$ na árvore aumentante H.

Temos o emparelhamento aumentado de uma unidade para: $M = \{x_1y_2, x_2y_1, x_3y_3, x_5y_5\}$. Construímos a árvore aumentante a partir do vértice x_4 da seguinte forma: inicializamos $S = \{x_4\}$ e $T = \emptyset$. Seja $y_2 \in N(S) \setminus T$. Como a aresta $y_2x_1 \in M$, atualizamos $S = \{x_1, x_4\}$ e $T = \{y_2\}$. Seja $y_3 \in N(S) \setminus T$. Como a aresta $y_3x_3 \in M$, temos $S = \{x_1, x_3, x_4\}$ e $T = \{y_2, y_3\}$. Como $N(S) = T$, o Teorema 2 afirma que não há emparelhamento que sature todos os vértices de X.

Observe que o Algoritmo Húngaro terminou neste caso com um emparelhamento máximo.

Comecemos de novo, agora com o emparelhamento $M = \{x_1y_2, x_4y_3\}$. Construímos a árvore aumentante H a partir do vértice x_3 da seguinte forma: inicializamos $S = \{x_3\}$ e $T = \emptyset$. Seja $y_2 \in N(S) \setminus T$. Como a aresta $y_2x_1 \in M$, atualizamos $S = \{x_1, x_3\}$ e $T = \{y_2\}$. Em seguida, seja $y_3 \in N(S) \setminus T$. Como a aresta $y_3x_4 \in M$, atualizamos $S = \{x_1, x_3, x_4\}$ e $T = \{y_2, y_3\}$. Como $N(S) = T$, o Teorema 2 afirma que não há emparelhamento que sature todos os vértices de X.

Observe que o Algoritmo Húngaro não terminou neste caso com um emparelhamento máximo.

Complexidade

Para avaliarmos a complexidade do Algoritmo Húngaro, consideramos as seguintes propriedades:

- Um emparelhamento tem no máximo $O(n)$ arestas;
- Cada aumentação acrescenta uma unidade ao emparelhamento corrente;
- Para cada aumento, em tempo $O(m)$ construímos nova árvore M-alternante onde procuramos novo caminho aumentante;
- Em tempo $O(n)$ aumentamos o emparelhamento corrente.

Logo, temos no máximo $O(n)$ estágios de aumentação, cada estágio com limite superior $O(m)$. Portanto obtemos o limite superior total de $O(nm)$.

5 Grafo bipartido com pesos: Algoritmo de Kuhn

O Algoritmo Húngaro descrito na Seção 4 fornece um algoritmo eficiente para se determinar uma alocação viável de funcionários por tarefas, se tal alocação existe. Entretanto, pode-se também levar em consideração a produtividade dos funcionários em tarefas variadas. Neste caso, estamos interessados numa alocação que maximize a produtividade total dos funcionários. O problema de se encontrar tal alocação é o conhecido

problema de alocação ótima. O problema de alocação ótima é equivalente a encontrar um emparelhamento perfeito de peso máximo neste grafo com pesos. Nos referimos a tal emparelhamento simplesmente como um *emparelhamento ótimo*.

Dado um grafo bipartido completo $G = (V, E)$, com $2n$ vértices e n^2 arestas, com conjunto de vértices bipartido em conjuntos X e Y, o Algoritmo de Kuhn que vamos apresentar a seguir encontra em tempo $O(n^2 m)$ um emparelhamento perfeito de peso máximo.

Sejam $X = \{x_1, x_2, \ldots, x_n\}$, $Y = \{y_1, y_2, \ldots, y_n\}$, $w_{ij} = w(x_i y_j)$, o peso não negativo da aresta $x_i y_j$, o qual representa a produtividade do funcionário x_i em relação à tarefa y_j.

Definimos *rotulação de vértices viável* como uma função ℓ real sobre o conjunto de vértices $X \cup Y$ tal que, para todo $x \in X$, para todo $y \in Y$, temos $\ell(x)+\ell(y) \geq w(xy)$. Dados um grafo G, uma rotulação de vértices viável ℓ e um emparelhamento M, sempre temos a desigualdade: $\sum_{e \in M} w(e) \leq \sum_{v \in V} \ell(v)$.

Observe que sempre temos naturalmente uma rotulação de vértices viável *trivial* dada por:

$$\ell(v) = \begin{cases} \max_{y \in Y} w(vy), & \text{se } v \in X; \\ 0, & \text{se } v \in Y. \end{cases}$$

Dada uma rotulação de vértices viável, selecionamos o seguinte conjunto de arestas: $E_\ell = \{xy \in E : \ell(x) + \ell(y) = w(xy)\}$.

Lembramos que um *subgrafo gerador* é um subgrafo que contém todos os vértices do grafo original e um subconjunto das arestas do grafo original. O subgrafo gerador de G com conjunto de arestas E_ℓ é chamado *subgrafo-igualdade* associado à rotulação de vértices viável ℓ e é denotado por G_ℓ. Observe que todo emparelhamento perfeito M^* no grafo G_ℓ satisfaz $\sum_{e \in M^*} w(e) = \sum_{v \in V} \ell(v)$.

O seguinte teorema estabelece uma relação entre subgrafo-igualdade e emparelhamento ótimo:

Teorema 3 *Seja ℓ uma rotulação de vértices viável de G. Se G_ℓ contém um emparelhamento perfeito M^*, então M^* é um emparelhamento ótimo de G.*

Prova: Seja G_ℓ com um emparelhamento perfeito M^*. Como G_ℓ é um subgrafo gerador de G, temos que M^* também é um emparelhamento perfeito de G. Suponha também M qualquer emparelhamento perfeito de G. Temos: $w(M) = \sum_{e \in M} w(e) \leq \sum_{v \in V} \ell(v) = \sum_{e \in M^*} w(e) = w(M^*)$. ■

O Teorema 3 é base do Algoritmo de Kuhn para encontrar um emparelhamento ótimo num grafo bipartido com pesos. Observe que na Seção 4 descrevemos o Algoritmo Húngaro para emparelhamento perfeito. O Algoritmo de Kuhn aplica sucessivas vezes o Algoritmo Húngaro a sucessivas rotulações viáveis do grafo entrada.

Inicialmente, começamos com a rotulação de vértices viável trivial ℓ, e determinamos o subgrafo-igualdade associado G_ℓ. Em seguida, escolhemos um emparelhamento arbitrário M em G_ℓ e aplicamos o Algoritmo Húngaro. Se encontramos um emparelhamento perfeito em G_ℓ, então o Teorema 3 garante que este emparelhamento é ótimo. Caso contrário, o Algoritmo Húngaro termina com um emparelhamento M' que não é perfeito e uma árvore M'-alternante H que não contém caminho M'-aumentante e não pode ser aumentada em G_ℓ. Modificamos então ℓ para uma rotulação de vértices viável ℓ' tal que ambos M' e H estejam contidos em $G_{\ell'}$ e H possa ser estendida em $G_{\ell'}$. Tais modificações numa rotulação de vértices viável são feitas sempre que necessárias até que um emparelhamento perfeito é encontrado no subgrafo-igualdade corrente.

O Algoritmo de Kuhn pode ser resumido nos seguintes passos:

1. Comece com a rotulação de vértices viável trivial ℓ, determine G_ℓ, e escolha um emparelhamento arbitrário M em G_ℓ.

2. Se X é M-saturado, então já que $|X| = |Y|$ temos um emparelhamento perfeito e portanto ótimo. Neste caso, PARE. Caso contrário, seja u um vértice não M-saturado. Defina $S = \{u\}$ e $T = \emptyset$.

3. Se $T \subset N_{G_\ell}(S)$, então vá para o Passo 4. Caso contrário, $T = N_{G_\ell}(S)$. Calcule $\alpha_\ell = \min_{x \in S, y \notin T}\{\ell(x) + \ell(y) - w(xy)\}$ e defina a rotulação de vértices viável ℓ' como: se $v \in S$, então $\ell'(v) = \ell(v) - \alpha_\ell$; se $v \in T$, então $\ell'(v) = \ell(v) + \alpha_\ell$; caso contrário,

$\ell'(v) = \ell(v)$. Observe que $\alpha_\ell > 0$ porque ℓ é uma rotulação de vértices viável. Observe que $T \subset N_{G_{\ell'}}(S)$ porque G é bipartido completo. Substitua ℓ por ℓ' e G_ℓ por $G_{\ell'}$. Observe que $G_\ell \subset G_{\ell'}$ porque $T = N_{G_\ell}(S)$, logo toda aresta xy de G_ℓ ou tem ambas as extremidades atualizadas com $x \in S$ e $y \in T$, ou tem ambas as extremidades não atualizadas: toda aresta xy de G_ℓ satisfaz: $\ell'(x) + \ell'(y) = \ell(x) + \ell(y)$.

4. Escolha um vértice y em $N_{G_\ell}(S) \setminus T$. Se y é M-saturado, com $yz \in M$, substitua S por $S \cup \{z\}$ e T por $T \cup \{y\}$. Vá para o Passo 3. Caso contrário, seja P um caminho M-aumentante de u até y em G_ℓ. Substitua M por $M' = M\Delta E(P)$. Vá para o Passo 2.

Algoritmo de Kuhn
Dados: grafo G bipartido completo com pesos e rotulação viável trivial ℓ
Saída: emparelhamento ótimo M

encontre G_ℓ, e um emparelhamento M em G_ℓ
enquanto X não M-saturado em G_ℓ faça
    $u \leftarrow$ vértice não M-saturado em X
    $S \leftarrow \{u\}$
    $T \leftarrow \emptyset$
    $OK \leftarrow$ verdadeiro
    repita
        se $N_{G_\ell}(S) = T$ então
            calcule α_ℓ, ℓ', $G_{\ell'}$
            $\ell \leftarrow \ell'$
            $G_\ell \leftarrow G_{\ell'}$
        $y \leftarrow$ vértice em $N_{G_\ell}(S) \setminus T$
        se y é M-saturado então
            $S \leftarrow S \cup \{z\}$
            $T \leftarrow T \cup \{y\}$
        senão
            $OK \leftarrow$ falso
    até não OK
    $P \leftarrow$ caminho aumentante entre u e y
    $M \leftarrow M\Delta P$
retorne M

Exemplo

Consideremos um exemplo da aplicação do Algoritmo de Kuhn. Representamos a entrada do Algoritmo de Kuhn, um grafo bipartido G completo com pesos nas arestas, por uma matriz $W = [w_{ij}]$, onde w_{ij} é o peso da aresta $x_i y_j$ em G.

Comecemos então este exemplo da execução do Algoritmo de Kuhn considerando como entrada a seguinte matriz $W = [w_{ij}]$, onde as linhas correspondem aos vértices $x_i \in X$, as colunas correspondem aos vértices $y_j \in Y$, e cada entrada w_{ij} corresponde ao peso da aresta $x_i y_j$:

$$W = \begin{bmatrix} 3 & 5 & 5 & 4 & 1 \\ 2 & 2 & 0 & 2 & 2 \\ 2 & 4 & 4 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 3 & 3 \end{bmatrix}$$

A rotulação de vértices viável inicial é:

$$\ell(v) = \begin{cases} \max_{y \in Y} w(vy), & \text{se } v \in X; \\ 0, & \text{se } v \in Y. \end{cases}$$

Usaremos a seguinte representação para esta rotulação de vértices viável. Colocamos à direita da i-ésima linha o rótulo $\ell(x_i)$ de x_i e colocamos embaixo da j-ésima coluna o rótulo $\ell(y_j)$ de y_j, obtendo o diagrama a seguir:

$$W = \begin{array}{c} \begin{bmatrix} 3 & 5 & 5 & 4 & 1 \\ 2 & 2 & 0 & 2 & 2 \\ 2 & 4 & 4 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 3 & 3 \end{bmatrix} \\ \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \end{array} \end{array} \begin{array}{c} 5 \\ 2 \\ 4 \\ 1 \\ 3 \end{array}$$

Observe que o subgrafo-igualdade G_ℓ obtido é precisamente o grafo do exemplo da Seção 4. Sabemos pelo exemplo da Seção 4 que o Algoritmo Húngaro aplicado a este grafo G_ℓ mostra que G_ℓ não admite emparelhamento perfeito. Na verdade, exibimos o conjunto $S = \{x_1, x_3, x_4\}$ com $N_{G_\ell}(S) = \{y_2, y_3\}$. Portanto temos que modificar

a rotulação de vértices viável corrente ℓ usando a variável auxiliar $\alpha_\ell = \min_{x \in S, y \notin T}\{\ell(x) + \ell(y) - w(xy)\}$. Neste caso, $\alpha_\ell = 1$, o que fornece como ℓ':

$$W = \begin{bmatrix} 3 & 5 & 5 & 4 & 1 \\ 2 & 2 & 0 & 2 & 2 \\ 2 & 4 & 4 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 3 & 3 \end{bmatrix} \begin{matrix} 4 \\ 2 \\ 3 \\ 0 \\ 3 \end{matrix}$$
$$\begin{matrix} 0 & 1 & 1 & 0 & 0 \end{matrix}$$

Uma aplicação do Algoritmo Húngaro mostra que o subgrafo-igualdade associado $G_{\ell'}$ tem o emparelhamento perfeito $M = \{x_1y_4, x_2y_1, x_3y_3, x_4y_2, x_5y_5\}$. Portanto este emparelhamento é um emparelhamento ótimo para G.

Complexidade

Para avaliarmos a complexidade do algoritmo, observamos que:

1. Um emparelhamento tem no máximo $O(n)$ arestas;
2. Cada aumentação acrescenta uma unidade ao emparelhamento corrente;
3. Para construir cada árvore húngara onde procuramos novo caminho aumentante precisamos de $O(n)$ vezes computar o valor α_ℓ e atualizar a rotulação de vértices viável com o subgrafo-igualdade correspondente em tempo $O(m)$;
4. Em tempo $O(n)$ aumentamos o emparelhamento corrente.

Logo temos $O(n)$ estágios de aumentação, onde o tempo de cada estágio é dominado pelo tempo da construção da árvore húngara em tempo $O(nm)$. Portanto temos um algoritmo eficiente de complexidade $O(n^2m)$.

Para ver que as hipóteses impostas de que a entrada seja um grafo bipartido completo com $|X| = |Y|$ não são restritivas, observamos que

dado um grafo $G = (X \cup Y, E)$ podemos adicionar as arestas que faltam para torná-lo bipartido completo atribuindo peso nulo a cada nova aresta. Caso $|X| \neq |Y|$, podemos adicionar novos vértices e novas arestas a eles incidentes com peso nulo.

6 Caso geral sem pesos: Algoritmo de Edmonds

O Algoritmo de Edmonds para o caso grafo geral sem pesos de complexidade $O(n^3)$ generaliza o algoritmo para o caso bipartido. Encontrar caminhos aumentantes de modo eficiente é simples no caso bipartido. No caso geral, Edmonds mostrou como se podia contrair florações (certos ciclos ímpares) e procurar por caminhos aumentantes de modo eficiente. Esta generalização adiciona apenas dificuldades conceituais, mas não adiciona ineficiência assintótica.

Vale a pena dedicar um parágrafo para comentar a vantagem de se considerar grafos bipartidos ao estudar problemas de emparelhamento. Tanto no caso sem pesos quanto no caso com pesos, o estudo dos problemas de emparelhamento no caso geral também fornece algoritmos eficientes clássicos embora de natureza mais complexa do que no caso bipartido.

Comecemos por analisar um exemplo e assim apreciar a dificuldade de se considerar o caso geral. No grafo $G = (V, E)$ com

$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}\}$$
$$E = \{v_1v_2, v_2v_3, v_3v_4, v_4v_5, v_5v_6, v_6v_7, v_7v_8, v_8v_9, v_9v_{10}, v_1v_9, v_2v_6, v_4v_6\},$$

considere o emparelhamento $M = \{v_2v_3, v_4v_5, v_6v_7, v_8v_9\}$. Pelo Teorema 1, a existência do caminho aumentante $P = v_1v_2v_3v_4v_5v_6v_7v_8v_9v_{10}$ diz que M não é um emparelhamento de cardinalidade máxima.

Uma estratégia natural é tentar aplicar o algoritmo da Seção 4 que, a partir de um vértice não M-saturado, digamos v_1 no caso do nosso exemplo, constrói os conjuntos T e S que definem uma árvore M-alternante onde esperamos encontrar um caminho M-aumentante que ligue v_1 a um outro vértice não M-saturado.

A dificuldade aparece porque existem ramos da árvore M-alternante que não correspondem a caminhos. No caso bipartido sabemos que

$S \subset X$ e $T \subset Y$ e que portanto $S \cap T = \emptyset$. Agora no caso geral, onde permitimos ciclos ímpares, a aplicação do algoritmo da Seção 4 permite considerar uma aresta $v_i v_j$ duas vezes: uma vez com $v_i \in S$ e $v_j \in T$ e outra vez com $v_i \in T$ e $v_j \in S$. No nosso exemplo, um ramo possível de uma árvore M-alternante de raiz v_1 seria $R = v_1 v_9 v_8 v_7 v_6 v_4 v_5 v_6 v_7 v_8 v_9 v_{10}$. De um modo geral, denomina-se *passeio aumentante* à saída obtida pelo Algoritmo Húngaro, quando submetido a um grafo não necessariamente bipartido.

Ao passar do caso bipartido para o caso geral, a única diferença estrutural introduzida foi a presença de ciclos ímpares. Tentamos então controlar através deles o mau comportamento de exemplos como o que acabamos de analisar. O que ocorreu de errado? Foi a presença do triângulo: após a sua travessia o ramo na árvore M-alternante corresponde a um "caminho" no grafo G que começa a se atravessar na contra-mão! Nem todos os ciclos ímpares podem causar tal comportamento: é necessário que eles sejam tão densos em arestas emparelhadas quanto possível.

Definimos então uma *floração* como um ciclo com $2k + 1$ vértices onde k arestas são emparelhadas.

Portanto, se um grafo não tem florações em relação ao emparelhamento corrente, então ele é "efetivamente" bipartido, no sentido de que a procura por caminhos aumentantes pode ser feita como no caso bipartido.

O que se tenta fazer em relação ao algoritmo da Seção 4 é modificá-lo para que, caso existam florações, elas sejam identificadas e para que mesmo na presença de florações estes algoritmos possam continuar encontrando caminhos aumentantes válidos.

Para encontrar uma floração, assim que encontramos pela segunda vez uma aresta $v_i v_j$ emparelhada, procuramos o último vértice externo b que estes caminhos têm em comum: este vértice é o único da floração que não é emparelhado com outro vértice da floração. A floração consiste deste vértice b chamado de *base* da floração e dos dois caminhos de u até v_i e de u até v_j.

Ao procurar por um caminho aumentante a partir de um vértice u, se descobrimos uma floração f, então existe um caminho alternante de u até qualquer vértice v em f que termina com uma aresta emparelhada:

basta compor o caminho alternante de u até b, a base da floração, com o caminho convenientemente escolhido de b até v em f.

A técnica utilizada para se lidar com florações consiste essencialmente num processo de encolhimento onde uma floração é reduzida a um único vértice. Se f é uma floração num grafo $G = (V, E)$ em relação a um emparelhamento M, o grafo resultante de G por encolhimento de f será $G/f = (V/f, E/f)$, onde V/f é o conjunto de vértices de V com todos os vértices de f omitidos e um novo vértice v_f adicionado; E/f é o conjunto de arestas obtidas de E omitindo as arestas com duas extremidades em f e substituindo as arestas vu com $u \in f$ e $v \notin f$ por vv_f. Este novo vértice v_f é chamado de *pseudo-vértice*.

O que é importante garantir é que o encolhimento de uma floração não adiciona nem omite caminhos aumentantes em relação ao grafo original.

Teorema 4 (Edmonds, 1965) *Suponha que enquanto procuramos por um caminho aumentante a partir de um vértice u de um grafo G, em relação a um emparelhamento M, descobrimos uma floração f. Então existe um caminho aumentante a partir de u em G em relação a M se e somente se existe um caminho aumentante a partir de u (ou de v_f caso u seja a base da floração f) em G/f em relação a M/f.*

Prova: Suponha que existe caminho aumentante P a partir de u em G/f com respeito a M/f. Suponha que P passa por v_f, isto é, $P = [uP'wv_fw'P'']$. (Observe que se P não passa por v_f, então P é caminho aumentante em G.) Como P é alternante, ou wv_f ou v_fw' é emparelhada. Suponha que wv_f é emparelhada. Então $wu_0 \in M$ e $w'u_j \in E \setminus M$, onde u_0 é a base da floração f e u_j é outro vértice de f. O caminho aumentante a partir de u será então $[uP'wu_oP'''u_jw'P'']$, onde P''' é o caminho de u_0 até u_j que termina com uma aresta emparelhada (o caminho vazio caso u_0 concida com u_j).

Reciprocamente, suponha que existe caminho aumentante P a partir de u em G com respeito a M e construa um caminho aumentante a partir de u em G/f com respeito a M/f. Caso P não contenha vértice da floração f, nada temos a analisar. Caso contrário, P contém vértice da floração f e uma análise das diferentes possibilidades da interseção

de P com a floração f constrói, em cada caso, um caminho aumentante a partir de u em G/f com respeito a M/f. ■

O caminho M-aumentante em G obtido pelo Teorema 4 correspondente ao caminho P, M/f-aumentante em G/f, é denominado *imagem inversa* de P.

O Teorema 4 sugere uma maneira de resolver o problema do emparelhamento de cardinalidade máxima num grafo geral reduzindo-o a problemas de determinar caminhos aumentantes.

Dado um grafo G e um emparelhamento M, o princípio é procurar um caminho M-aumentante em G, mediante a aplicação do Algoritmo Húngaro da Seção 4. Se G não contém caminho M-aumentante, então M é máximo pelo Teorema 1. Caso contrário, seja C um caminho M-aumentante de G e M é substituído por $M\Delta C$. A aplicação do Algoritmo Húngaro pode fornecer um passeio aumentante P ao invés de caminho. Nesse caso, P contém uma floração f. Constroem-se G/f e M/f, e o problema se reduz a encontrar um caminho M/f-aumentante em G/f, segundo o Teorema 4. O algoritmo pode ser descrito, como segue.

No *passo inicial*, sejam G um grafo, M um emparelhamento de G. Determinar o conjunto $U \subseteq V$ dos vértices não M-saturados de G. Desmarcar todos os vértices de U. No *passo geral*, se U não contém vértices desmarcados, então o algoritmo termina: M é um emparelhamento de cardinalidade máxima de G. Caso contrário, escolher $u \in U$ desmarcado. Marcar u. Determinar um caminho M-aumentante C em G, iniciado em u, se existir. Para construir C, utilizar o algoritmo recursivo, abaixo descrito. Em caso de sucesso, reiniciar o presente algoritmo com M substituído por $M\Delta C$. Se G não contiver tal caminho C, repetir o passo geral.

O algoritmo seguinte determina um caminho M-aumentante, se existir, iniciado em um dado vértice não M-saturado $u \in V$, em um grafo $G = (V, E)$.

Dados um grafo G, um emparelhamento M de G e um vértice $u \in V$ não M-saturado, o algoritmo seguinte determina, de forma recursiva, um caminho M-aumentante em G, iniciado em u, se existir.

Aplicar o Algoritmo Húngaro, com raiz u. Se esse informar que G não possui caminho M-aumentante iniciado em u, o algoritmo ter-

mina, com a resposta de que G não possui o caminho procurado. Caso contrário, o Algoritmo Húngaro produz um passeio M-aumentante P, iniciado em u. Se P for um caminho, o algoritmo termina, informando P. Caso contrário, P contém uma floração f. Construir G/f e M/f. Aplicar o presente método para determinar, se existir, um caminho M/f-aumentante C/f iniciado em u (ou iniciado em v_f, se u for a base de f). Se G/f não possuir tal caminho, então o algoritmo termina, respondendo que G não possui caminho M-aumentante iniciado em u. Caso contrário, seja C o caminho M-aumentante de G, obtido de C/f, a partir do Teorema 4. O algoritmo termina com a informação de que C é o caminho procurado.

Em seguida, encontra-se descrita a implementação do Algoritmo de Edmonds. Dados um grafo G e um emparelhamento M de G, a função $AE(G, M)$ retorna o conjunto vazio $\emptyset$, se M for de cardinalidade máxima. Caso contrário, $AE(G, M)$ é um emparelhamento de cardinalidade uma unidade maior do que M. Esta função emprega a função $AE1(G, M, u)$, que corresponde a um caminho M-aumentante de G, iniciado em u. Se tal caminho não existir, $AE1(G, M, u)$ é o conjunto vazio. Finalmente, $AE1(G, M, u)$ utiliza uma função $AH(G, M, u)$ a qual fornece um passeio M-aumentante, iniciado em u, se existir, obtido pela aplicação do Algoritmo Húngaro. Se tal caminho não existir, então $AH(G, M, u)$ é o conjunto vazio.

A função $AE(G, M)$ pode ser descrita como se segue.

```
AE(G, M) :
    U ← subconjunto dos vértices não M-saturados de G
    desmarque todos os vértices de U
    enquanto ∃u ∈ U desmarcado faça
        marque u
        se AE1(G, M, u) ≠ ∅ então
            retorne MΔAE1(G, M, u)
    retorne ∅
```

Abaixo, encontra-se detalhada a função $AE1(G, M, u)$.

$AE1(G, M, u)$:
    se $AH(G, M, u) = \emptyset$ então
        retorne $\emptyset$
    senão
        se $AH(G, M, u)$ é um caminho então
            retorne $AH(G, M, u)$
        senão
            $f \leftarrow$ uma floracão de base b contida em $AH(G, M, u)$
            $v_f \leftarrow$ pseudo-vértice de G/f, relativo a f
            se $u = b$ então $v \leftarrow v_f$, senão $v \leftarrow u$
            retorne imagem inversa de $AE1(G/f, M/f, v)$

As funções acima são utilizadas como se segue.

Algoritmo de Edmonds
Dados: grafo G
Saída: emparelhamento máximo M

$M \leftarrow \emptyset$
enquanto $AE(G, M) \neq \emptyset$ faça
    $M \leftarrow AE(G, M)$
    $AE(G, M)$
retorne M

Exemplo

Consideremos um exemplo da aplicação do Algoritmo de Edmonds. Seja $G = (V, E)$, com

$$\begin{aligned} V &= \{v_1, v_2, v_3, v_4, v_5\} \\ E &= \{v_1v_2, v_2v_3, v_3v_4, v_4v_5, v_1v_5, v_1v_3, v_2v_4, v_2v_5\}. \end{aligned}$$

Na primeira iteração, todos os vértices são livres e M é vazio. Escolhendo v_1 como raiz, o rotulamos como externo e v_2 como interno. Temos $P = v_1v_2$ e esta aumentação fornece $M = \{v_1v_2\}$.

Na segunda iteração, os vértices livres são v_3, v_4 e v_5 e $M = \{v_1v_2\}$. Geramos T a partir de v_3. Adicionamos a T as arestas v_3v_1 e v_1v_2. A adição da aresta v_2v_3 descobre a floração com vértices v_1, v_2, v_3. Esta floração é encolhida dando origem ao pseudo-vértice v_6 contendo os

vértices v_1, v_2, v_3. A exploração a partir de v_6 da aresta v_6v_4 descobre o vértice livre v_4. Temos assim um caminho aumentante P que a princípio é $P = v_6v_4$. Ao expandirmos o pseudo-vértice v_6 e interpolarmos a porção par da floração $v_3v_1v_2$ em P obtemos $P = v_3v_1v_2v_4$. O emparelhamento fica então aumentado para $M = \{v_3v_1, v_2v_4\}$.

Neste ponto sabemos que M é máximo já que apenas o vértice v_5 está livre. Vejamos como o algoritmo conclui isto.

Na iteração final, o único vértice livre é v_5 e $M = \{v_1v_3, v_2v_4\}$. Geramos T a partir de v_5. Adicionamos a T as arestas v_5v_1 e v_1v_3. Adicionamos a T as arestas v_3v_2 e v_2v_4. A adição da aresta v_4v_3 descobre a floração com vértices v_2, v_3, v_4. Esta floração é encolhida dando origem ao pseudo-vértice v_7 contendo os vértices v_2, v_3, v_4. A adição da aresta v_7v_5 descobre a floração com vértices v_1, v_5, v_7. Esta floração é encolhida dando origem ao pseudo-vértice v_8 contendo os vértices v_1, v_5, v_7.

Com a contração de T a um único vértice o algoritmo termina e retorna $M = \{v_1v_3, v_2v_4\}$ como emparelhamento de cardinalidade máxima para G.

Complexidade

O algoritmo de Edmonds para emparelhamento de cardinalidade máxima no caso geral é claramente polinomial. Sua complexidade é dominada pelo custo de encontrar sucessivos caminhos aumentantes. Como cada novo caminho aumentante traz um acréscimo de uma unidade na cardinalidade do emparelhamento corrente, temos que no máximo $O(n)$ caminhos aumentantes são encontrados. Para encontrar um caminho aumentante, temos que considerar no máximo $O(n)$ vértices livres. Para cada vértice livre, construímos uma árvore de busca T. A construção de T, incluindo o processamento de florações tem custo $O(m)$. Obtemos, com esta análise, um limite superior de $O(n^2m)$.

Podemos obter um limite superior melhor com o seguinte argumento: se um vértice livre não dá origem a caminho aumentante, então na verdade toda aresta presente na árvore de busca encontrada a partir deste vértice não pode estar em caminho aumentante. Passamos então a procurar por um caminho aumentante a partir do próximo vértice livre. Portanto, precisamos de tempo $O(m)$ para encontrar um caminho

aumentante, o que fornece o limite superior total de $O(nm)$ para o algoritmo de Edmonds.

Micali e Vazirani [13] apresentaram um algoritmo $O(m\sqrt{n})$, o mais eficiente que se conhece para o problema.

7 Notas bibliográficas

O Teorema 1 sobre caminhos aumentantes e emparelhamento de cardinalidade máxima foi estabelecido por Berge [1] em 1957. Durante muito tempo acreditou-se que algoritmos eficientes tanto para o caso bipartido quanto para o caso não bipartido seriam consequências imediatas deste teorema. De fato, podemos citar os algoritmos eficientes para o caso bipartido de Hall [8] de 1956 e o método húngaro de Kuhn [10] de 1955. Por outro lado, o caso não bipartido só foi de fato resolvido por Edmonds [4] em 1965.

Em relação à complexidade dos algoritmos para emparelhamento conhecidos para o caso bipartido, temos as descrições de Hopcroft e Karp [9] em 1973 e de Even e Tarjan [5] em 1975 para algoritmos $O(m\sqrt{n})$. Observamos que Even e Tarjan [5] na verdade reconheceram o caso particular de redes simples com capacidades unitárias do algoritmo de Dinic [3] que resolve em tempo $O(n^3)$ o problema de fluxo máximo em redes através de redes de camadas. Para o caso não bipartido, o algoritmo mais eficiente que se conhece tem complexidade também $O(m\sqrt{n})$ tendo sido descrito por Micali e Vazirani [13] em 1980.

A caracterização para existência de emparelhamentos perfeitos discutida no Exercício 7 é de Tutte [16] e uma prova alternativa é a de Lovász [12].

8 Exercícios

1. Descreva um algoritmo que testa se uma dada árvore admite um emparelhamento perfeito. Conclua do seu algoritmo que uma árvore admite no máximo um emparelhamento perfeito. Caso a árvore de entrada não admita um emparelhamento perfeito, o

seu algoritmo pode ser modificado de modo a retornar um emparelhamento máximo nesta árvore?

2. Duas pessoas jogam um jogo num grafo G selecionando vértices distintos v_0, v_1, v_2, ... tal que, para $i > 0$, v_i é adjacente a v_{i-1}. O último jogador que conseguir selecionar um vértice ganha. Mostre que o primeiro jogador tem uma estratégia vencedora se e somente se G não tem um emparelhamento perfeito.

3. Considere um tabuleiro 8×8 de onde foram removidos dois cantos opostos de tamanho 1×1. Prove que é impossível, usando retângulos 1×2, cobrir exatamente este tabuleiro.

4. Prove que um grafo bipartido tem um emparelhamento perfeito se e somente se $|N(S)| \geq |S|$, para todo $S \subseteq V$.

5. A condição de parada do Algoritmo Húngaro exibe um $S \subseteq X$, com $|N(S)| < |S|$. A condição de parada do Algoritmo de Ford-Fulkerson para fluxo máximo em uma rede pode exibir um corte mínimo para esta rede?

6. O Algoritmo Húngaro é descrito originalmente para procurar um emparelhamento perfeito num grafo bipartido, ou mais geralmente, um emparelhamento que sature um dos conjuntos da bipartição. Modifique o Algoritmo Húngaro de modo que encontre um emparelhamento máximo num grafo bipartido.

7. Denote por $\Phi(G \setminus V')$ o número de componentes conexas de $G \setminus V'$ que tem um número ímpar de vértices. Prove a seguinte caracterização: $G = (V, E)$ tem um emparelhamento perfeito se e somente se $\Phi(G \setminus V') \leq |V'|$, para todo $V' \subseteq V$.

8. A caracterização do Exercício 7, assim como a caracterização do Teorema 2, não fornecem diretamente algoritmos polinomiais. Descreva um algoritmo polinomial para testar se um grafo admite um emparelhamento perfeito.

9. Obtenha uma prova alternativa do Teorema 2 usando a caracterização para existência de emparelhamentos perfeitos obtida no Exercício 7.

10. Uma *cobertura* C é um conjunto de arestas tal que todo vértice do grafo é extremo de pelo menos uma aresta de C. Uma *cobertura de cardinalidade mínima* é uma cobertura com o menor número possível de arestas.

    Seja M um emparelhamento de cardinalidade máxima e seja C uma cobertura de cardinalidade mínima de $G = (V, E)$. Construa uma cobertura C' a partir de M adicionando a M, para cada vértice não M-emparelhado v, uma aresta incidente a v. Construa um emparelhamento M' a partir de C removendo de C, para cada vértice v que é extremo de mais de uma aresta em C, todas as arestas incidentes a v que estão em C a menos de uma.

    Prove as seguintes duas iqualdades: $|C'| = |V| - |M|$ e $|M'| = |V| - |C|$. Conclua que C' é uma cobertura de cardinalidade mínima e que M' é um emparelhamento de cardinalidade máxima. Conclua que o mesmo algoritmo apresentado para resolver o problema de emparelhamento de cardinalidade máxima resolve também o problema de cobertura de cardinalidade mínima.

Referências

[1] C. Berge. Two theorems in graph theory. *Proc. National Acad. of Science* 43 (1957) 842–844.

[2] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications.* American Elsevier Publishing Co., Inc., New York, 1976.

[3] E. A. Dinic. Algorithm for solution of a problem of maximal flow in a network with power estimation. *Soviet Math. Dokl.* 11 (1970) 1277–1280.

[4] J. Edmonds. Paths, trees and flowers. *Canad. J. Math.* 17 (1965) 449–467.

[5] S. Even and R. E. Tarjan. Network flow and testing graph connectivity. *SIAM J. Computing* 4 (1975) 507–512.

[6] L. R. Ford and D. R. Fulkerson. *Flows in Networks.* Princeton University Press, New Jersey, 1962.

[7] A. Gibbons. *Algorithmic Graph Theory.* Cambridge University Press, Cambridge, 1985.

[8] M. Hall. An algorithm for distinct representatives. *American Math. Monthly* 63 (1956) 716–717.

[9] J. E. Hopcroft and R. M. Karp. A $n^{5/2}$ algorithm for maximum matching in bipartite graphs. *SIAM J. Computing* 2 (1973) 225–231.

[10] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quaterly* 2 (1955) 83–97.

[11] E. L. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt, Rinehart and Winston, New York, 1976.

[12] L. Lovász. Three short proofs in graph theory. *J. Combinatorial Theory B* 19 (1975) 111–113.

[13] S. Micali and V. V. Vazirani. An $O(\sqrt{|V|}.|E|)$ algorithm for finding maximum matching in general graphs. *Proc. 21st Annual Symposium on the Foundations of Computer Science*, IEEE (1980) 17–27.

[14] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, New Jersey, 1982.

[15] J. L. Szwarcfiter. *Grafos e Algoritmos Computacionais.* Editora Campus, Rio de Janeiro, 1986.

[16] W. T. Tutte. The factorisation of linear graphs. *J. London Math. Soc.* 22 (1947) 107–111.

[17] R. E. Tarjan. *Data Structures and Network Algorithms.* SIAM, Philadelphia, 1983.