

Susana de Sousa Tavares

Redes: Fluxo Máximo e Corte Mínimo



TESE Nº 215

Departamento de Matemática Pura
Faculdade de Ciências da Universidade do Porto
Dezembro / 2006



FC

Biblioteca
Faculdade de Ciências
Universidade do Porto



0000100727

Reg. 509289
Cota Tese N° 215

Aprovada.
Maia Cavalli

Susana de Sousa Tavares

Redes: Fluxo Máximo e Corte Mínimo



**Tese submetida à Faculdade de Ciências da Universidade do Porto para a obtenção do grau de
Mestre em Engenharia Matemática**

**Faculdade de Ciências do Porto
MATEMÁTICA**

**Departamento de Matemática Pura
Faculdade de Ciências da Universidade do Porto**

Dezembro / 2006

Resumo

A presente tese introduz conceitos relacionados com o problema do fluxo máximo, contendo as definições necessárias para a formalização do problema e algumas abordagens para solucioná-lo, como os algoritmos de Ford – Fulkerson e o de Edmonds – Karp.

Abstract

The present thesis introduces concepts related to the problem of maximum flow. It contains the necessary definitions to formulate the problem and some approaches to solve it, such as the algorithms of Ford - Fulkerson and Edmonds - Karp.

Resumé

Cette thèse introduit des concepts relatifs au problème du flux maximum, contenant les définitions nécessaires à la formalisation du problème et quelques approches pour le résoudre, comme les algorithmes de Ford-Fulkerson et celui de Edmonds-Karp.

Índice

Resumo	3
Abstract	4
Resumé	5
1. Introdução	8
2. Algumas definições e resultados básicos	10
3. Conectividade	12
3.1. Grafos e digrafos conexos	12
Conectividade de arestas	13
Conectividade de vértices	16
3.2. Teorema de Menger para grafos	21
3.3. Algumas formas análogas do Teorema de Menger	24
Teorema de Menger para digrafos (forma de arcos)	24
Teorema de Menger para grafos (forma de vértices)	26
Teorema de Menger para digrafos (forma de vértices)	28
3.4. Redes de telecomunicações fiáveis	28
4. Fluxos em redes básicas e fluxo máximo	32
4.1. Redes básicas e fluxo	32
4.2. Caminhos que aumentam o fluxo	34
4.3. Transformação numa rede básica	41
5. Fluxos máximos e cortes mínimos	51
5.1. Cortes mínimos	51
5.2. Problema do fluxo máximo e corte mínimo	57
5.3. Resolução do problema do fluxo máximo	60
Aumentar o fluxo usando caminhos directos	60
Caminho que aumenta o fluxo f	62
Teorema do Fluxo Máximo e Corte Mínimo	65
5.4. Funções do tempo de execução de um algoritmo, a sua ordem e hierarquia	67
Hierarquia das ordens	71
5.5. Algoritmo do fluxo máximo e corte mínimo	73

5.5.1.	Algoritmo de Ford - Fulkerson	73
5.5.2.	Armazenamento de algoritmos: pilhas e filas	77
	Pilhas ("Stacks")	78
	Filas ("Queues")	80
5.5.3.	Algoritmo de procura <i>breadth first</i> (BFS)	81
5.5.4.	Algoritmo de Edmonds - Karp	86
6.	Demonstração do Teorema de Menger	92
7.	Conclusão	95
	Bibliografia	96

1. Introdução

Esta dissertação desenvolve-se em torno do problema e do Teorema do Fluxo Máximo e Corte Mínimo aplicado a redes.

A Teoria de Grafos é um ramo da matemática que estuda as propriedades dos grafos. Um grafo é um conjunto de pontos, chamados vértices, conectados por linhas, chamadas arestas. Dependendo da aplicação, as arestas podem ou não ser dirigidas, pode ser permitido ou não que as arestas liguem um vértice a ele próprio e vértices e/ou arestas podem ter um peso numérico associado. Se as arestas têm uma direcção associada (indicada por uma seta na representação gráfica) temos um digrafo.

Um problema frequente que usa redes para facilitar a sua resolução é a determinação do fluxo máximo entre dois pontos distintos, a fonte S e o destino T . Por exemplo, numa rede de gás natural na qual ocorre um problema num ponto pode provocar a urgência de fornecer a esse ponto a maior quantidade de gás possível. É necessário encontrar o fluxo máximo de gás que pode ser fornecido entre S e T de modo que a quantidade de gás fornecida seja máxima.

O problema do fluxo máximo pode ser aplicado para maximizar o fluxo de uma rede de distribuição de uma companhia a partir das suas fábricas (fornecedores) para os seus (suas) clientes (fábricas), para maximizar o fluxo de água através de um sistema de aquedutos, para maximizar o fluxo de veículos através de uma rede de transporte e para descobrir quantas ligações numa rede de telecomunicações podem ser quebradas de modo que as comunicações entre todos os subscritores continuem a ser possíveis, entre outras.

Os tipos de redes usadas na vida real são: redes que representam transição de estados, redes físicas de estradas, canais e tubagens e redes com arcos que representam durações no tempo.

O Capítulo 2 é uma recolha de algumas noções e resultados básicos da Teoria de Grafos que serão usados ao longo desta dissertação.

No Capítulo 3 estuda-se a conectividades de grafos e digrafos, passando pela conectividade de arestas e de vértices, bem como a introdução das diversas formas do Teorema de Menger. Finaliza-se este capítulo com uma aplicação de todos estes conceitos nas redes de telecomunicações. É importante saber qual o menor número de ligações entre as centrais telefónicas que podem ser danificadas, fazendo com que a rede fique separada em duas partes, e quais podem ser essas ligações ou qual o menor número de centrais que podem ser fechadas de modo que a comunicação continue a ser possível e quais podem ser essas centrais. Estas questões serão resolvidas usando a conectividade de arestas e a conectividade de vértices.

No Capítulo 4 inicia-se o estudo da procura do fluxo máximo de uma rede, começando com a introdução do conceito de rede básica, de caminhos que aumentam o fluxo e terminando com as transformações necessárias para transformar uma rede numa rede básica.

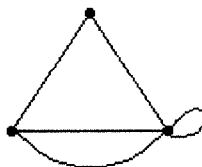
O Capítulo 5 constitui o “grosso” desta dissertação, pois é o capítulo onde se introduzem o conceito de cortes mínimos, o modo de resolução do problema do fluxo máximo e corte mínimo, o Teorema do Fluxo Máximo e Corte Mínimo, os algoritmos do fluxo máximo e corte mínimo de Ford – Fulkerson bem como o de Edmonds – Karp. Neste capítulo, também se estuda a eficiência dos algoritmos através das suas funções de complexidade

de tempo, os modos de armazenamento de algoritmos, através de pilhas ("stacks") e filas ("queues") e o algoritmo de procura *breadth first (BFS)*, usado inicialmente pelo algoritmo de Edmonds – Karp.

Para finalizar, no Capítulo 6 fazem-se as demonstrações das diversas formas do Teorema de Menger usando o Teorema do Fluxo Máximo e Corte Mínimo.

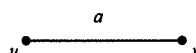
2. Algumas definições e resultados básicos

Grafo: é um conjunto (finito) de vértices (V) unidos (ou não) por um certo número de arestas (E), em que cada aresta une (no máximo) dois vértices.



Arestas múltiplas: são duas ou mais arestas que unem o mesmo par de vértices.

Vértices adjacentes: dois vértices u e v de um grafo são vértices adjacentes se são unidos pela mesma aresta a . Os vértices u e v são **incidentes** com a aresta a e a aresta a é **incidente** com os vértices u e v .



Lacete: é uma aresta que une um vértice a si mesmo.

Passeio: entre dois vértices é uma sucessão de arestas que os ligam.

Atalho: passeio em que todas as arestas são diferentes mas não necessariamente todos os vértices.

Atalho fechado: passeio fechado em que todas as arestas são diferentes e começa e acaba no mesmo vértice.

Caminho: passeio onde as arestas e os vértices são todos diferentes.

Ciclo: é um atalho fechado em que à exceção do primeiro e último vértice, todos os vértices são diferentes.

Grafo simples: é um grafo sem lacetes nem arestas múltiplas.

Subgrafo: de um grafo G é um grafo em que todos os seus vértices são vértices de G e todas as suas arestas são arestas de G .

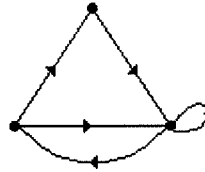
Nota: G é um subgrafo de si mesmo.

Grau de um vértice: é o número de arestas incidentes no vértice. Cada lacete conta duas vezes.

Grafo regular: é um grafo no qual todos os vértices têm o mesmo grau.

Grafo completo: é um grafo em que todos os vértices estão ligados entre si; representa-se por K_n , em que n é o número de vértices.

Digrafo: é um conjunto (finito) de vértices unidos (ou não) por um certo número de arcos dirigidos.



Subdigrafo: de um digrafo D é um digrafo em que todos os seus vértices são vértices de D e todos os seus arcos são arcos de D .

Nota: D é um subdigrafo de si mesmo.

Lema dos Apertos de Mão: Em qualquer grafo a soma dos graus dos vértices é igual ao dobro do número de arestas.

Demonstração:

Em qualquer grafo, como cada aresta tem dois extremos, cada uma contribui exactamente duas vezes para a soma dos graus dos vértices, de onde o resultado segue. ■

Algoritmo: é um procedimento bem definido que toma como parâmetro de entrada um valor (ou um conjunto de valores) – *input* – e produz como resultado um valor (ou um conjunto de valores) – *output*. É uma ferramenta que permite resolver um problema específico.

Árvore: é um grafo conexo sem ciclos.

Árvore geradora: de um grafo conexo G é um subgrafo de G que inclui todos os vértices de G e é, também, uma árvore (árvore maximal).

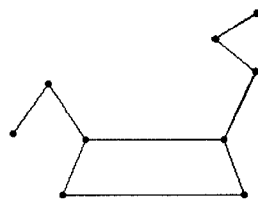
3. Conectividade

3.1. Grafos e digrafos conexos

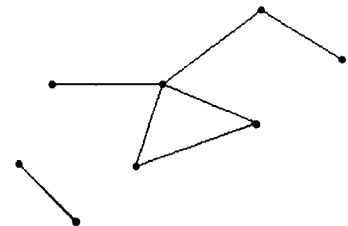
A conectividade é um dos conceitos centrais da Teoria de Grafos, tanto do ponto de vista teórico como prático, uma vez que existem grafos “mais conexos” que outros. Grafos conexos podem ser, às vezes, tornados desconexos removendo apenas um vértice ou uma aresta. A conectividade de arestas e de vértices são dois parâmetros numéricos úteis para estudar a conectividade de um grafo.

Definição 3.1.1: Um grafo é **conexo** se existe um caminho entre cada par de vértices e é **desconexo** caso contrário. Todos os grafos desconexos podem ser separados em subgrafos disjuntos conexos designados por **componentes**.

Por exemplo:



Grafo conexo



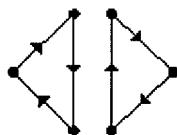
Grafo desconexo com 2 componentes

No caso de digrafos não é sempre verdade que um digrafo constituído apenas por uma componente precisa de ter um caminho entre quaisquer pares de vértices; esta observação leva a definir dois tipos de conectividade em digrafos.

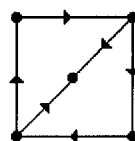
Definição 3.1.2: Um digrafo é **conexo** se todos os grafos subjacentes são conexos e é **desconexo** caso contrário.

Um digrafo é **fortemente conexo** se existe um caminho dirigido do vértice u para v para cada par de vértices u e v .

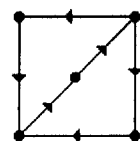
Por exemplo:



Digrafo desconexo



Digrafo conexo
(não fortemente conexo)

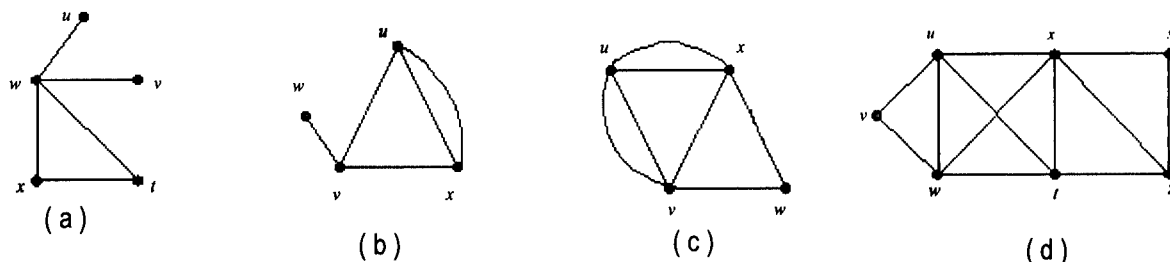


Digrafo fortemente conexo

Conectividade de arestas

Encontrar o número de arestas e vértices que, removidos, desconectam um grafo é aplicado directamente na análise da fiabilidade de redes de telecomunicações, nas quais existem usualmente diferentes caminhos entre um par de subscritores (vértices). Neste tipo de situações é importante saber quantas ligações (arestas) podem ser bloqueadas ou partidas sem que uma chamada entre dois subscritores seja danificada.

Considerem-se os seguintes grafos:



O grafo (a) pode ser separado em duas componentes removendo a aresta wv ou wu , ou seja, a remoção de qualquer uma destas arestas desconecta o grafo.

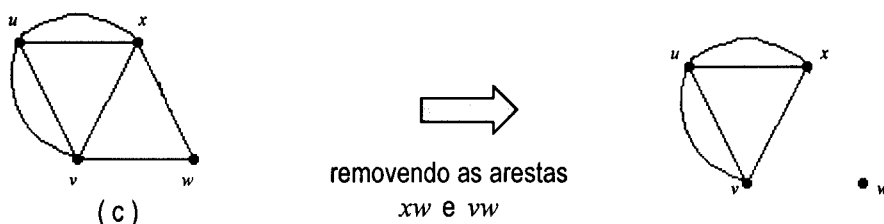


O grafo (b) pode ser desconectado removendo uma única aresta, a aresta vw .



Definição 3.1.3: Uma única aresta que, removida, desconecta o grafo, como a aresta wv ou wu do grafo (a) ou vw do grafo (b), é designada por **ponte**.

O grafo (c) não pode ser desconectado removendo uma única aresta mas pode ser desconectado removendo duas arestas, tais como xw e vw .



Analogamente, o grafo (d) apenas pode ser desconectado removendo duas arestas, como por exemplo, vu e vw .



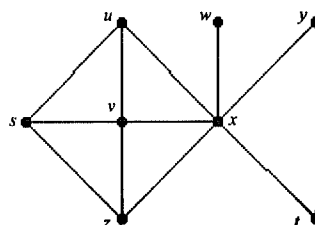
Definição 3.1.4: Seja G um grafo conexo. O menor número de arestas que removidas desconecta G designa-se por **conectividade de arestas** e representa-se por $\lambda(G)$.

Os grafos (a) e (b) têm $\lambda(G) = 1$ e os grafos (c) e (d) têm $\lambda(G) = 2$.

Nota 3.1.5: Qualquer grafo simples não vazio com uma ponte tem $\lambda(G) = 1$. Tem-se que $\lambda(G) = 0$ se e só se G é um grafo desconexo ou um grafo vazio.

Para desconectar um grafo existem vários processos, mas devem ser consideradas formas de desconectar um grafo que não sejam redundantes.

Considere-se o seguinte grafo:



Este grafo pode ser desconectado removendo apenas a aresta $\{xw\}$ ou as arestas $\{xy, yt\}$ mas não apenas uma destas arestas.



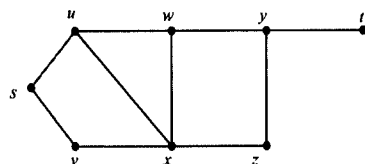
Outra forma de desconectar este grafo é remover estas três arestas, mas a aresta $\{xw\}$ é redundante. Um conjunto de arestas não redundantes designa-se por conjunto de corte.

Definição 3.1.6: Um **conjunto de corte de arestas** de um grafo conexo G é um conjunto de arestas S com as seguintes propriedades:

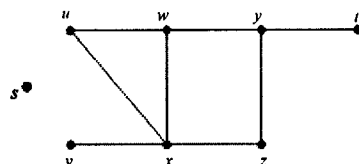
- (i) a remoção de todas as arestas de S desconecta G ;
- (ii) a remoção de algumas (mas não todas) as arestas de S , não desconecta G .

Dois conjuntos de corte não precisam de ter necessariamente o mesmo número de arestas. Por exemplo, no grafo anterior os conjuntos $\{xw\}$ e $\{xy, yt\}$ são ambos conjuntos de corte de arestas. Note-se também que a conectividade de arestas $\lambda(G)$ de um grafo G é o número de elementos do menor conjunto de corte de arestas de G ; no grafo anterior, $\lambda(G)=1$.

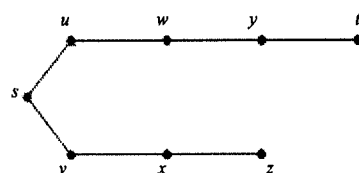
Exemplo 3.1.7: Verifique-se se os seguintes conjuntos são conjuntos de corte de arestas do seguinte grafo G :



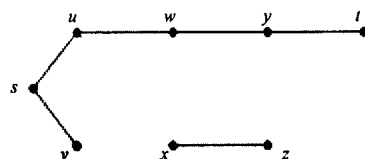
- (a) $\{su, sv\}$ é um conjunto de corte, por definição:



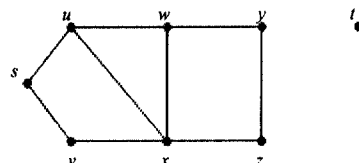
- (b) $\{ux, wx, yz\}$ não é um conjunto de corte porque a remoção destas arestas não desconecta G :



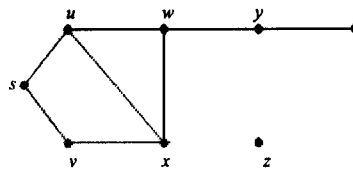
- (c) $\{ux, vx, wx, yz\}$ é um conjunto de corte, por definição:



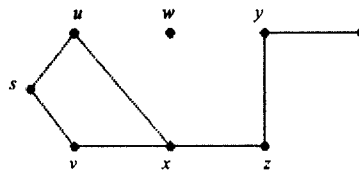
- (d) $\{yt, yz\}$ não é um conjunto de corte porque o grafo pode ser desconectado removendo apenas a aresta yt :



- (e) $\{wx, xz, yz\}$ não é um conjunto de corte porque G pode ser desconectado removendo as arestas xz e yz :

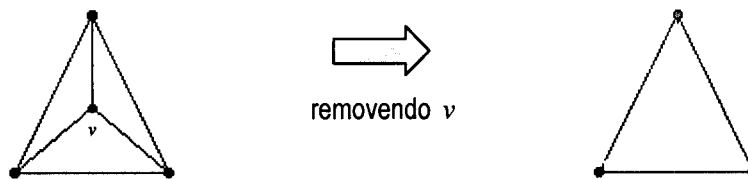


(f) $\{uw, wx, wy\}$ é um conjunto de corte, por definição.

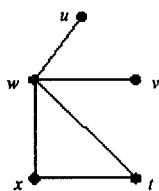


Conectividade de vértices

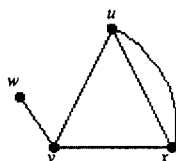
Pode-se definir também a conectividade em termos do número mínimo de vértices que é preciso remover para desconectar um grafo. Quando se remove um vértice retiram-se, também, as arestas nele incidentes:



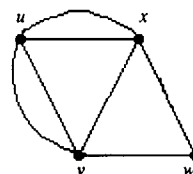
Considerem-se os seguintes grafos:



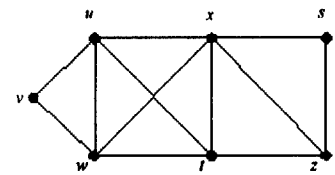
(a)



(b)

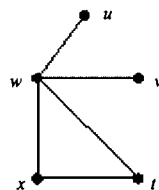


(c)



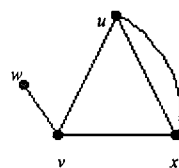
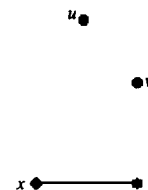
(d)

Os grafos (a) e (b) podem ser desconectados removendo um único vértice.



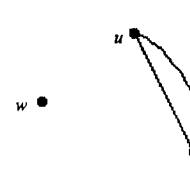
(a)

removendo w

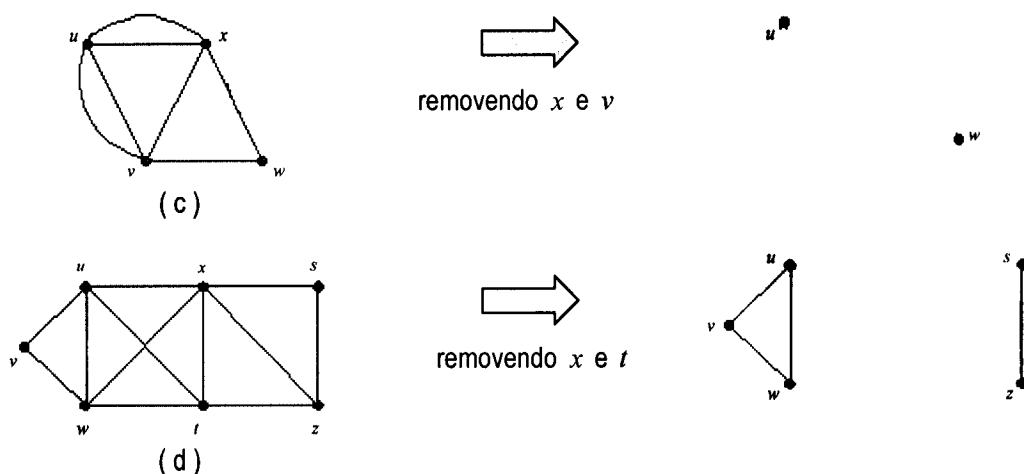


(b)

removendo v



Os grafos (c) e (d) não podem ser desconectados removendo apenas um vértice, mas sim removendo dois vértices.



Definição 3.1.8: Seja G um grafo conexo. O menor número de vértices que removidos desconecta G designa-se por **conectividade de vértices** e representa-se por $K(G)$.

Os grafos (a) e (b) têm $K(G) = 1$ e os grafos (c) e (d) têm $K(G) = 2$.

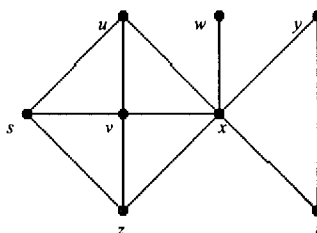
Definição 3.1.9: A **conectividade de vértices de um grafo completo** K_n é $K(K_n) = n - 1$.

Definição 3.1.10: Um **conjunto de corte de vértices** de um grafo conexo G é um conjunto de vértices S (mas não o conjunto de todos os vértices) com as seguintes propriedades:

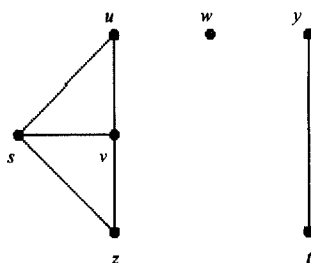
- (i) a remoção de todos os vértices de S desconecta G ;
- (ii) a remoção de alguns (mas não todos) os vértices de S , não desconecta G .

Por outras palavras, um conjunto de vértices é um conjunto de corte de um grafo conexo G se a sua remoção desconecta alguma componente conexa de G , produzindo um subgrafo de G com mais componentes que as que G tem.

Considere-se o seguinte grafo:

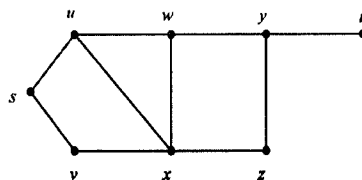


Este grafo pode ser desconectado removendo apenas o vértice x , logo $\{x\}$ é um conjunto de corte de vértices.

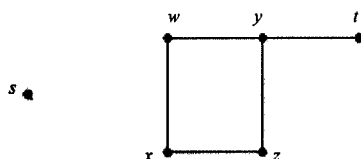


Dois conjuntos de corte de vértices não precisam de ter necessariamente o mesmo número de vértices. Por exemplo, no grafo anterior, os conjuntos $\{u, v, z\}$ e $\{x\}$ são ambos conjuntos de corte de vértices. Note-se, também, que a conectividade de vértices $K(G)$ de um grafo G é o número de elementos do menor conjunto de corte de vértices de G ; no grafo anterior, $K(G) = 1$.

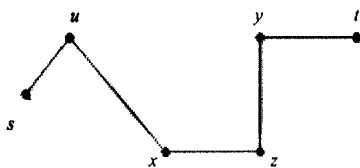
Exemplo 3.1.11: Verifique-se se os seguintes conjuntos são conjuntos de corte de vértices do seguinte grafo G :



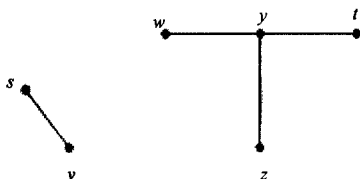
(a) $\{u, v\}$ é um conjunto de corte porque a remoção dos vértices u e v desconecta G e a remoção de apenas um destes vértices não desconecta G :



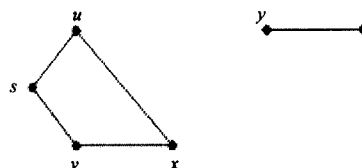
(b) $\{v, w\}$ não é um conjunto de corte porque a sua remoção não desconecta G :



(c) $\{u, x, y\}$ não é um conjunto de corte porque o grafo pode ser desconectado removendo os vértices u e x ou removendo o vértice y :



(d) $\{w, z\}$ é um conjunto de corte porque a remoção dos vértices w e z desconecta G e a remoção de apenas um destes vértices não desconecta G :



Com o exercício anterior verifica-se que a conectividade de vértices $K(G)$ não excede a conectividade de arestas $\lambda(G)$, que por sua vez não excede o menor grau dos vértices de G , $\delta(G)$. Esta desigualdade é válida para grafos conexos.

	(a)	(b)	(c)	(d)
$\lambda(G)$	1	1	2	2
$K(G)$	1	1	2	2
$\delta(G)$	1	1	2	2

Teorema 3.1.12: Para qualquer grafo conexo G , $K(G) \leq \lambda(G) \leq \delta(G)$.

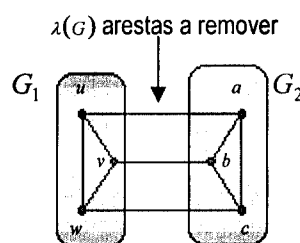
Demonstração:

Se G é um grafo completo K_n , então $K(G) = \lambda(G) = \delta(G) = n - 1$, pela definição de conectividade de um grafo completo.

Se G não é um grafo completo e se v é um vértice com grau $\delta(G)$, então G pode ser desconectado removendo todas as arestas incidentes com v . Decorre que $\lambda(G)$, o menor número de arestas que desconecta G , não pode exceder $\delta(G)$. Assim, $\lambda(G) \leq \delta(G)$.

Falta provar que $K(G) \leq \lambda(G)$ quando G não é um grafo completo. Observe-se que basta considerar G um grafo simples, pois se G for um grafo conexo não simples e se G' for o grafo simples obtido de G retirando os lacetes e trocando as arestas múltiplas por arestas simples, então $K(G) = K(G')$ e $\lambda(G') \leq \lambda(G)$. Logo se $K(G') \leq \lambda(G')$ será $K(G) = K(G') \leq \lambda(G') \leq \lambda(G)$.

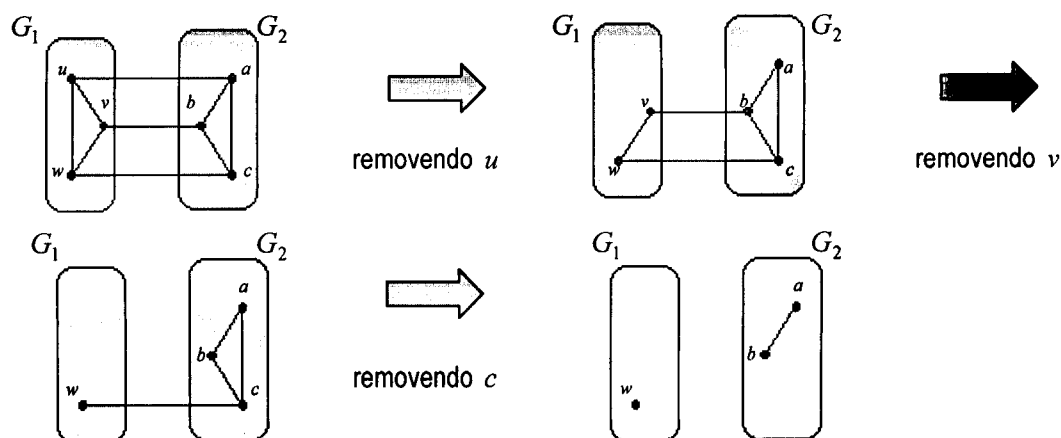
Seja G um grafo simples com n vértices e conectividade de arestas $\lambda(G)$. Como G não é um grafo completo, então existe pelo menos um vértice de grau no máximo $n - 2$, o que prova que $\lambda(G) \leq n - 2$. Existe pelo menos um conjunto $\lambda(G)$ de arestas que removido desconecta G (por definição de $\lambda(G)$) em duas componentes G_1 e G_2 , como se ilustra na figura seguinte:



Faculdade de Ciências do Porto
MATEMÁTICA

Mas podem-se, também, remover estas arestas eliminando, no máximo, $\lambda(G)$ vértices, desde que apenas se retire uma escolha conveniente para o vértice final de cada uma destas $\lambda(G)$ arestas. Removam-se os $\lambda(G)$ vértices um de cada vez. Como há no máximo $n - 2$ arestas para serem eliminadas e como em cada passo se pode retirar um vértice de G_1 ou de G_2 , então esta remoção pode ser feita sem deixar nem G_1 nem G_2 vazios.

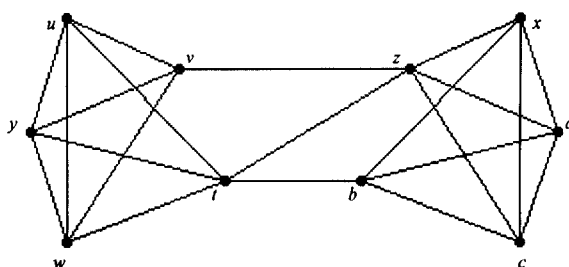
Por exemplo:



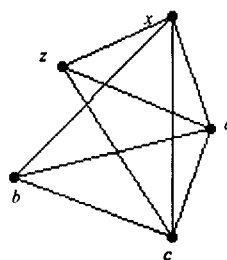
Daqui decorre que o número mínimo de vértices removíveis que desconectam G não pode exceder $\lambda(G)$, isto é, $K(G) \leq \lambda(G)$.

Deste modo, para qualquer grafo conexo G tem-se que: $K(G) \leq \lambda(G) \leq \delta(G)$. ■

As desigualdades do Teorema 3.1.12 podem ser estritas, ou seja, $K(G) < \lambda(G) < \delta(G)$. Por exemplo, no grafo seguinte: $K(G) = 2$, $\lambda(G) = 3$ e $\delta(G) = 4$.

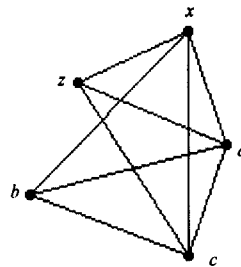
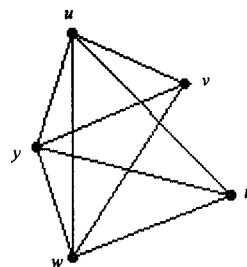


- removendo os vértices v e t :



logo $K(G) = 2$;

- removendo as arestas vz , tz e tb :



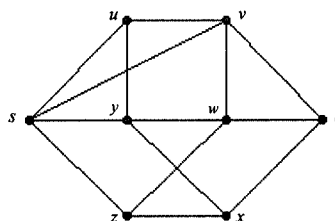
logo $\lambda(G) = 3$.

3.2. Teorema de Menger para grafos

O Teorema de Menger estabelece uma forte dualidade entre vários tipos de problemas de optimização e é usado, como já foi referido anteriormente em redes de telecomunicações. Este teorema vai ser demonstrado utilizando o Teorema do Fluxo Máximo e Corte Mínimo.

Definição 3.2.1: Seja G um grafo conexo e sejam s e t vértices de G . Um caminho entre s e t é designado de **caminho - st** . Dois ou mais caminhos - st são de **arestas disjuntas** se não têm arestas em comum e de **vértices disjuntos** se não têm vértices em comum (além de s e t).

Por exemplo:



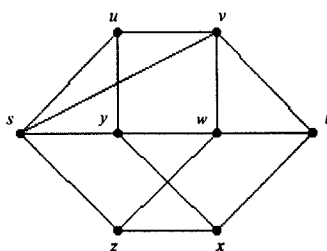
- os caminhos $suv t$, $syw t$ e $szx t$ são caminhos - st de arestas disjuntas e de vértices disjuntos;
- os caminhos $syw t$ e $svw t$ não são nem caminhos - st de arestas disjuntas nem de vértices disjuntos (pois têm a aresta wt e o vértice w em comum);
- os caminhos $suv t$ e $svw t$ são caminhos - st de arestas disjuntas, mas não são de vértices disjuntos (pois têm o vértice v em comum).

Nota 3.2.2: Se dois caminhos - st de um grafo forem de vértices disjuntos então, também, são de arestas disjuntas:

Se dois caminhos - st de vértices disjuntos não fossem de arestas disjuntas, então teriam pelo menos uma aresta em comum, o que significaria que teriam pelo menos um vértice comum (para além dos vértices s e t), contradizendo o facto de serem de vértices disjuntos.

Definição 3.2.3: Seja G um grafo conexo e sejam s e t vértices de G . Diz-se que algumas **arestas separam** s de t se a sua remoção destrói todos os caminhos entre s e t . De modo análogo, alguns **vértices** (excluindo s e t) **separam** s de t se a sua remoção destrói todos os caminhos entre s e t .

Por exemplo:



- a remoção das arestas uv, sv, yw, yx, zw e zx separa s de t , assim como as arestas vt, wt e xt :



- a remoção dos vértices u, v, y e z separa s de t , assim como os vértices v, w e x :



Há de facto uma relação entre caminhos - st de arestas disjuntas e de vértices disjuntos e arestas e vértices que separam s de t , como veremos em seguida:

Considere-se um conjunto de arestas que separam s de t num grafo conexo arbitrário. Como removendo estas arestas são destruídos todos os caminhos entre s e t , cada caminho - st deve incluir pelo menos uma destas arestas. Assim, decorre que o número máximo de caminhos - st de arestas disjuntas não pode exceder o número de arestas desse conjunto. Aplicando esta ideia a qualquer conjunto de arestas que separam s de t :

o número máximo de caminhos - st de arestas disjuntas $\leq$ o número de arestas, em qualquer conjunto, que separam s de t .

Como esta desigualdade é verdadeira para qualquer conjunto de arestas que separam s de t , também é verdadeira para um conjunto com um número mínimo de arestas:

o número máximo de caminhos - st de arestas disjuntas $\leq$ o número mínimo de arestas que separam s de t .

Estas desigualdades são de facto iguais e constituem o Teorema de Menger para grafos na forma de arestas.

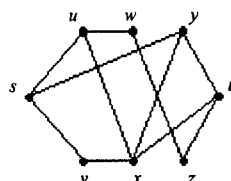
Teorema 3.2.4: TEOREMA DE Menger para grafos (forma de arestas) Seja G um grafo conexo e sejam s e t vértices de G . O número máximo de caminhos - st de arestas disjuntas é igual ao número mínimo de arestas que separam s de t .

(A demonstração do teorema será feita no Capítulo 6)

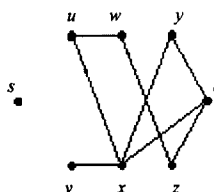
Do Teorema de Menger decorre que se se encontrarem k caminhos - st de arestas disjuntas e k arestas que separam s de t (para o mesmo valor de k), então k é o número máximo de caminhos - st de arestas disjuntas e o número mínimo de arestas que separam s de t . Estas k arestas que separam s de t formam, necessariamente, um conjunto de corte de arestas, sendo assim apenas necessário considerar conjuntos de corte que, removidos, desconectam G em duas componentes, uma contendo s e outra contendo t .

Por exemplo:

Para o seguinte grafo encontre-se k caminhos - st de arestas disjuntas e k arestas que separam s de t (para o mesmo valor de k) e, usando o Teorema de Menger para grafos (forma de arestas), encontre-se o número máximo de caminhos - st de arestas disjuntas.



Três caminhos - st de arestas disjuntas são $suwzt$, syt e $svxt$ e três arestas que separam s de t são su , sv e sy :



Assim $k = 3$, logo o número máximo de caminhos - st de arestas disjuntas e o número mínimo de arestas que separam s de t são ambos iguais a 3.

Nota 3.2.5: O Teorema de Menger pode ser usado para obter a conectividade de arestas. Relembre-se que a conectividade de arestas $\lambda(G)$ de um grafo conexo G é o menor número de arestas que desconecta G . Assim, pelo Teorema de Menger na forma de arestas existem pelo menos $\lambda(G)$ caminhos de arestas disjuntas entre qualquer par de vértices.

Do teorema anterior, segue imediatamente o corolário.

Corolário 3.2.6: TEOREMA DE Menger para grafos (forma de arestas) Um grafo conexo G tem conectividade de arestas l se e só se todo o par de vértices de G é unido por l ou mais caminhos de arestas disjuntas e pelo menos um par de vértices é unido por exactamente l caminhos de arestas disjuntas.

3.3. Algumas formas análogas do Teorema de Menger

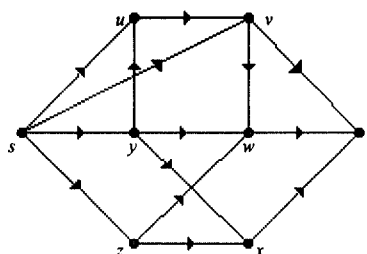
Nesta secção são apresentadas outras formas do Teorema de Menger, começando pelo Teorema de Menger para digrafos (forma de arcos) e continuando com a forma de vértices para grafos e digrafos.

Teorema de Menger para digrafos (forma de arcos)

Muitos conceitos introduzidos anteriormente para grafos são análogos para digrafos. Por exemplo, as seguintes definições são muito idênticas às dadas para grafos.

Definição 3.3.1: Seja D um digrafo conexo e sejam s e t vértices de D . Um caminho de s para t é designado por **caminho - st** . Dois ou mais caminhos - st são de **arcos disjuntos** se não têm arcos em comum e de **vértices disjuntos** se não têm vértices em comum (além de s e t).

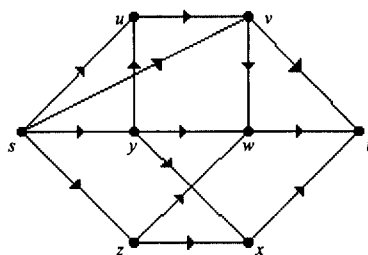
Por exemplo:



- os caminhos $suv t$, $syw t$ e $szx t$ são caminhos - st de arcos disjuntos e de vértices disjuntos;
- os caminhos $syw t$ e $svw t$ não são nem caminhos - st de arcos disjuntos nem de vértices disjuntos (pois têm o arco $w t$ e o vértice w em comum);
- os caminhos $suv t$ e $svw t$ são caminhos - st de arcos disjuntos, mas não são de vértices disjuntos (pois têm o vértice v em comum).

Definição 3.3.2: Seja D um digrafo conexo e sejam s e t vértices de D . Diz-se que alguns **arcos separam s de t** se a sua remoção destrói todos os caminhos de s para t . De modo análogo, alguns **vértices (excluindo s e t) separam s de t** se a sua remoção destrói todos os caminhos de s para t .

Por exemplo:



- a remoção dos arcos uv , sv , yw , yx , zw e zx separa s de t , assim como os arcos vt , wt e xt :



- a remoção dos vértices u , v , y e z separa s de t , assim como os vértices v , w e x :



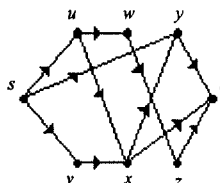
Teorema 3.3.3: TEOREMA DE Menger para digrafos (forma de arcos) Seja D um digrafo conexo e sejam s e t vértices de D . O número máximo de caminhos - st de arcos disjuntos é igual ao número mínimo de arcos que separam s de t .

(A demonstração do teorema será feita no Capítulo 6)

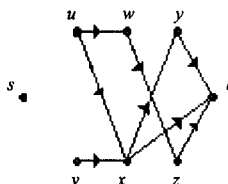
Também neste caso se podem encontrar k caminhos - st de arcos disjuntos e k arcos que separam s de t (para o mesmo valor de k), sendo k o número máximo de caminhos - st de arcos disjuntos e o número mínimo de arcos que separam s de t .

Por exemplo:

Para o seguinte digrafo encontre-se k caminhos - st de arcos disjuntos e k arcos que separam s de t (para o mesmo valor de k) e, usando o Teorema de Menger para digrafos (forma de arcos), encontre-se o número máximo de caminhos - st de arcos disjuntos.



Três caminhos - st de arcos disjuntos são $suwzt$, syt e $svxt$ e três arcos que separam s de t são su , sv e sy :



Assim $k = 3$, logo o número máximo de caminhos - st de arcos disjuntos e o número mínimo de arcos que separam s de t são ambos iguais a 3.

Teorema de Menger para grafos (forma de vértices)

O Teorema de Menger na forma de arestas relaciona o número de caminhos - st de arestas disjuntas num grafo com o menor número de arestas que separam s de t . Este resultado relaciona-se com a conectividade de arestas. Analogamente, o Teorema de Menger na forma de vértices relacionará o número de caminhos - st de vértices disjuntos num grafo com o menor número de vértices que separam s de t .

Mais geralmente, considere-se um conjunto de vértices (excluindo s e t) separando vértices não adjacentes s e t num grafo conexo arbitrário. Desde que a remoção destes vértices destrua todos os caminhos entre s e t , cada caminho - st deve incluir pelo menos um deles, decorrendo que o número máximo de caminhos - st de vértices disjuntos não pode exceder o número de vértices no conjunto.

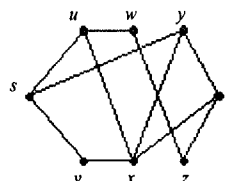
Teorema 3.3.4: TEOREMA DE MENER para grafos (forma de vértices) Seja G um grafo conexo e sejam s e t vértices não adjacentes de G . O número máximo de caminhos - st de vértices disjuntos é igual ao número mínimo de vértices que separam s de t .

(A demonstração do teorema será feita no Capítulo 6)

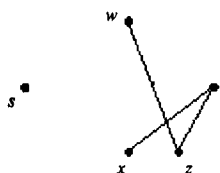
Como anteriormente, decorre que, se se conseguirem encontrar k caminhos - st de vértices disjuntos e k vértices que separam s de t (para o mesmo valor de k), então k é o número máximo de caminhos - st de vértices disjuntos e o número mínimo de vértices que separam s de t . Note-se que estes k vértices que separam s de t formam necessariamente um conjunto de corte de vértices, sendo assim apenas necessário considerar conjuntos de corte de vértices que, removidos, desconectam G em duas ou mais componentes, uma contendo s e outra contendo t .

Por exemplo:

Para o seguinte grafo encontre-se k caminhos - st de vértices disjuntos e k vértices que separam s de t (para o mesmo valor de k) e, usando o Teorema de Menger para grafos (forma de vértices), encontre-se o número máximo de caminhos - st de vértices disjuntos.



Três caminhos - st de vértices disjuntos são $suwzt$, syt e $svxt$ e três vértices que separam s de t são u , v e y :



Assim $k = 3$, logo o número máximo de caminhos - st de vértices disjuntos e o número mínimo de vértices que separam s de t são ambos iguais a 3.

Nota 3.3.5: O Teorema de Menger pode ser usado para obter a conectividade de vértices. Relembre-se que a conectividade de vértices $K(G)$ de um grafo conexo (desde que este não seja completo) é o menor número de vértices que desconecta G . Assim, pelo Teorema de Menger na forma de vértices existem pelo menos $K(G)$ caminhos de vértices disjuntos entre qualquer par de vértices.

Do teorema anterior, segue imediatamente o corolário.

Corolário 3.3.6: TEOREMA DE Menger para grafos (forma de vértices) Um grafo conexo G (desde que este não seja completo) tem conectividade de vértices k se e só se todos os pares de vértices não adjacentes de G são unidos por k ou mais caminhos de vértices disjuntos e pelo menos um par de vértices não adjacentes é unido por exactamente k caminhos de vértices disjuntos.

Teorema de Menger para digrafos (forma de vértices)

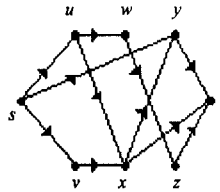
Teorema 3.3.7: TEOREMA DE Menger para digrafos (forma de vértices) Seja D um digrafo conexo e sejam s e t vértices não adjacentes de D . O número máximo de caminhos - st de vértices disjuntos é igual ao número mínimo de vértices que separam s de t .

(A demonstração do teorema será feita no Capítulo 6)

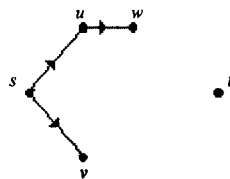
Também neste caso se podem encontrar k caminhos - st de vértices disjuntos e k vértices que separam s de t (para o mesmo valor de k), sendo k o número máximo de caminhos - st de vértices disjuntos e o número mínimo de vértices que separam s de t .

Por exemplo:

Para o seguinte digrafo encontre-se k caminhos - st de vértices disjuntos e k vértices que separam s de t (para o mesmo valor de k) e, usando o Teorema de Menger para digrafos (forma de vértices), encontre-se o número máximo de caminhos - st de vértices disjuntos.



Três caminhos - st de vértices disjuntos são $suwzt$, syt e $svxt$ e três vértices que separam s de t são y , x e z :

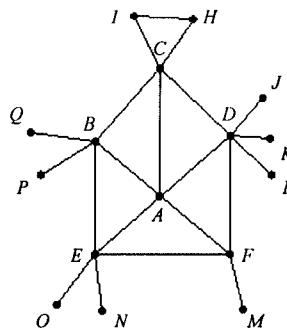


Assim $k = 3$, logo o número máximo de caminhos - st de vértices disjuntos e o número mínimo de vértices que separam s de t são ambos iguais a 3.

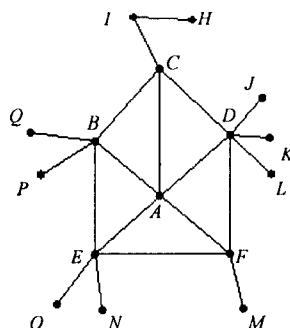
3.4. Redes de telecomunicações fiáveis

Um grafo que represente uma rede de telecomunicações contém um grande número de vértices e de arestas. Os vértices podem representar centrais telefónicas e subscritores ou interruptores nas centrais; em cada caso, as arestas representam as ligações entre estes. Um aspecto importante nestes problemas é que a rede seja fidedigna, ou seja, que as chamadas entre subscritores sejam possíveis se algumas centrais ou ligações forem danificadas.

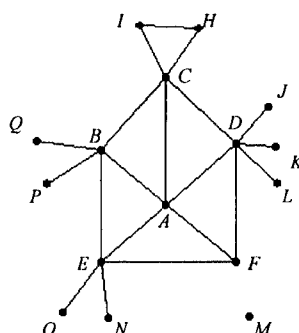
Considere-se o seguinte grafo, o qual representa uma possível ligação de centrais:



Se a ligação CH é danificada:



a comunicação entre as centrais C e H continua a ser possível mas se a ligação FM é danificada:



a central M fica incomunicável. A conectividade de arestas do grafo que representa a rede de telecomunicações representa o número mínimo de ligações que, danificadas, impedem o sistema de funcionar, enquanto que a conectividade de vértices representa o número de centrais que podem ser danificadas sem que haja perda da comunicação entre elas.

Deste modo, existem duas importantes razões para prover caminhos alternativos entre duas centrais; uma é que a comunicação entre estas continue a ser possível, mesmo que algum caminho seja danificado, e outra é que uma particular ligação ou central num caminho entre duas centrais pode também fazer parte de um caminho entre outro par de centrais. A ligação ou central pode ser usada quando a capacidade de uma nova chamada é esforçada, o que previne a nova chamada de ser feita. Diz-se que a nova chamada é *bloqueada*.

Para duas centrais particulares, o número máximo de caminhos alternativos, não passando dois ao mesmo tempo pela mesma central, é (na terminologia da Teoria de Grafos) o número máximo de caminhos de vértices disjuntos entre os vértices correspondentes do grafo. Pelo Corolário 3.3.6, o menor número de tais caminhos entre duas centrais é igual à conectividade de vértices. Analogamente, o número máximo de caminhos alternativos, não usando dois a mesma passagem entre centrais, é o número máximo de caminhos de arestas disjuntas entre os vértices correspondentes de um grafo. Pelo Corolário 3.2.6, o menor número de tais caminhos entre duas quaisquer centrais é igual à conectividade de arestas.

Considerando apenas a fiabilidade de uma rede, um sistema de telecomunicações devia ter muitos caminhos alternativos entre as centrais, ou seja, devia ter a maior conectividade de vértices e de arestas. Para tal ser possível, cada central deveria estar ligada a cada uma das outras centrais e o grafo correspondente seria um grafo completo, o que é impraticável. Na prática, o que se pode fazer é tentar encontrar o valor máximo para

as conectividades de vértices $K(G)$ e de arestas $\lambda(G)$ para o grafo G com o número de vértices e arestas dado.

Suponha-se que G tem n vértices e m arestas. Pelo Lema dos Apertos de Mão decorre que a soma dos graus dos vértices é $2m$. Deste modo, em média, o grau de cada vértice é $\frac{2m}{n}$, logo o grau mínimo dos vértices não pode exceder este valor. Usando o Teorema 3.1.12 e este resultado:

$$K(G) \leq \lambda(G) \leq \delta(G) \leq \frac{2m}{n}.$$

Um grafo G com n vértices e m arestas para o qual estas desigualdades são igualdades tem o valor máximo da conectividade de vértices e da conectividade de arestas.

Definição 3.4.1: Seja G um grafo com n vértices e m arestas; então G tem **conectividade óptima** se:

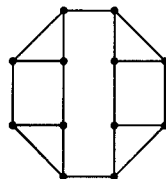
$$K(G) = \frac{2m}{n}.$$

Para verificar que um grafo tem conectividade óptima é suficiente ver que $K(G) = \frac{2m}{n}$, o que garante que

$K(G) = \lambda(G) = \delta(G)$, uma vez pelo que $K(G) \leq \lambda(G) \leq \delta(G) \leq \frac{2m}{n}$, como foi visto anteriormente.

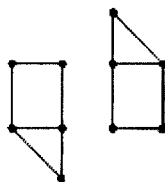
Nota 3.4.2: Todos os grafos com conectividade óptima são grafos regulares dado que o menor grau dos vértices é igual à média dos graus dos vértices.

Nem todos os grafos regulares têm conectividade óptima, por exemplo, o grafo:

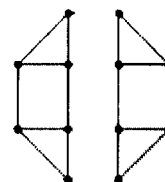


apresentado é um grafo regular de grau 3, $\delta(G) = 3$, mas

$$K(G) = 2$$



$$\lambda(G) = 2$$



e não tem conectividade óptima.

Exemplo 3.4.3: O grafo completo K_n tem conectividade óptima:

Pelo Lema dos Apertos de Mão, o número de arestas é $m = {}^nC_2 = \frac{n!}{2!(n-2)!} = \frac{n(n-1)}{2}$. A conectividade de vértices deste grafo é $K(K_n) = n-1$, logo:

$$\frac{2m}{n} = 2 \times \frac{\frac{n(n-1)}{2}}{n} = n-1,$$

logo K_n tem conectividade óptima.

Os valores de $K(G)$ e $\lambda(G)$ de um grafo G traduzem informação sobre a fiabilidade do sistema de telecomunicações correspondente ou partes desse sistema. No entanto, esses valores não evidenciam toda a história: dois grafos com o mesmo número de vértices, o mesmo número de arestas e os mesmos valores de $K(G)$ e $\lambda(G)$, podem não ter igual segurança dos sistemas.

Considere-se, por exemplo, os grafos seguintes, cada um tem 10 vértices e 12 arestas e que satisfazem $K(G) = \lambda(G) = 2$:



No primeiro grafo todos os caminhos do vértice s para o vértice t podem ser destruídos, removendo as arestas de um dos 4 conjuntos de corte com 2 arestas, separando s de t - $\{su, sv\}$, $\{wt, xt\}$, $\{su, xt\}$ ou $\{sv, wt\}$. No entanto, o segundo grafo tem apenas dois conjuntos de corte com 2 arestas separando s de t - $\{su, sv\}$ e $\{wt, xt\}$. Assim, todas as arestas contendo estes conjuntos de corte têm igual probabilidade de serem bloqueadas ou danificadas; espera-se que o segundo grafo seja mais seguro que o primeiro, uma vez que tem menos conjuntos de corte, o que se verifica neste caso.

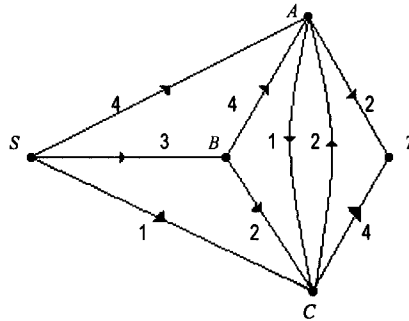
Para ter toda a informação sobre a fiabilidade de uma rede representada por um grafo, é preciso saber não apenas a conectividade de arestas e de vértices mas também todos os conjuntos de corte do grafo.

4. Fluxos em redes básicas e fluxo máximo

4.1. Redes básicas e fluxo

Começemos esta secção com um exemplo prático da aplicação de fluxos em redes básicas:

Um fabricante de uma certa cidade quer exportar diversas caixas de um dos seus produtos para um departamento noutra cidade. Existem diversos canais pelos quais ele as pode enviar, que são representados no digrafo seguinte:



O vértice S representa o fabricante e T o departamento na cidade destino; os números escritos nos arcos representam a capacidade dos canais (arcos).

O fabricante quer encontrar o número máximo de caixas que pode enviar ao longo da rede de modo a que nunca exceda a capacidade de cada canal.

Usando este exemplo define-se o conceito de rede:

Definição 4.1.1: Uma **rede básica** é um digrafo conexo N que satisfaz as condições:

- (i) N tem exactamente uma **fonte** e um **destino** (dois vértices distintos, nos quais a fonte só tem arcos de saída e o destino só tem arcos de entrada);
- (ii) cada arco e de N é assinalado com um número inteiro não negativo $c(e)$ designado de **capacidade de e** , que representa a quantidade máxima de “fluxo” que pode fluir ao longo do arco num dado tempo.

A capacidade de e de um arco pode representar, entre outras coisas:

- custos, distâncias, capacidades, e/ou suprimentos;
- tempo (trânsito, permanência, etc);
- confiabilidade de transmissão;
- probabilidade de ocorrerem falhas;
- capacidade de carga.

A análise de redes na antiguidade era usada para problemas de redes fluviais / marítimas, de abastecimento de água e de esgotos, enquanto que, na actualidade, é usada, principalmente, para redes ferroviárias / rodoviárias, eléctricas e de comunicações.

Cada arco incidente na fonte S vai de S para outro vértice e qualquer arco incidente no destino T vai de algum vértice para T . Deve-se ter em atenção que um arco não recebe mais do que a sua capacidade e nenhuma da capacidade é perdida ao longo do percurso.

Assume-se que cada rede N tem exactamente uma fonte e um destino (sendo estes distintos).

Dado um vértice V de uma rede N , denote-se o conjunto dos arcos que saem de V por $O(V)$ e os que entram em V por $I(V)$.

Definição 4.1.2: O **fluxo** numa rede básica N com fonte S e destino T é uma atribuição a cada arco e de N , de um número não negativo $f(e)$ designado por **fluxo ao longo do arco** e , satisfazendo a:

(i) **condição de praticabilidade:** o fluxo ao longo de cada arco não pode exceder a capacidade do arco, ou seja:

$$\text{fluxo} \leq \text{capacidade};$$

(ii) o fluxo total que sai da fonte S é igual ao fluxo total que entra no destino T ;

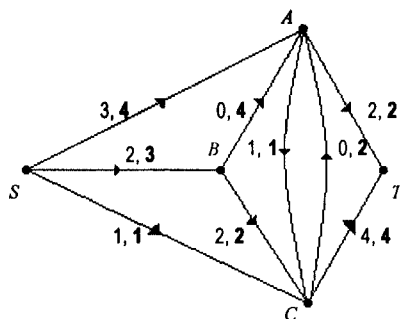
(iii) **regra de conservação do fluxo:** para cada vértice V entre S e T , a soma dos fluxos ao longo dos arcos que entram em V é igual à soma dos fluxos ao longo dos arcos que saem de V .

Tal fluxo é designado de **fluxo de S para T** ou um **fluxo - (S, T)** .

Mais especificamente, (iii) significa que se V é um vértice diferente de S e T , então:

$$\sum_{e \in O(V)} f(e) = \sum_{e \in I(V)} f(e).$$

Por exemplo:



O primeiro número a seguir a cada arco e é o fluxo $f(e)$ e o segundo número (a negrito) é a capacidade $c(e)$.

Neste exemplo:

$$\begin{aligned} \sum_{e \in O(S)} f(e) &= 3 + 2 + 1 = 6 = 2 + 4 = \sum_{e \in I(T)} f(e) \\ \sum_{e \in O(A)} f(e) &= 2 + 1 = 3 = 3 + 0 + 0 = \sum_{e \in I(B)} f(e) \\ \sum_{e \in O(B)} f(e) &= 0 + 2 = 2 = \sum_{e \in I(C)} f(e) \end{aligned}$$

$$\sum_{e \in O(C)} f(e) = 0 + 4 = 4 = 1 + 2 + 1 = \sum_{e \in I(C)} f(e)$$

Definição 4.1.3: Um arco e é **saturado** se $f(e) = c(e)$ e **não saturado** se $f(e) < c(e)$.

Por exemplo, no grafo acima, os arcos AT e AC são saturados e os arcos SA e BA são não saturados.

Definição 4.1.4: O número tal que $val(f) = \sum_{e \in O(S)} f(e) = \sum_{e \in I(T)} f(e)$, onde S e T são fonte e destino de uma rede N é designado por **valor do fluxo** e representa a quantidade de fluxo que flui ao longo da rede.

A representação do valor do fluxo como $val(f) = \sum_{e \in O(S)} f(e) - \sum_{e \in I(S)} f(e)$ ou $val(f) = \sum_{e \in I(T)} f(e) - \sum_{e \in O(T)} f(e)$

será útil mais adiante.

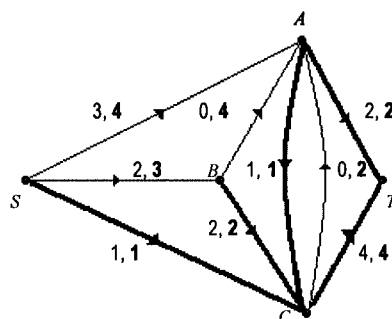
No exemplo acima o fluxo tem valor 6.

Definição 4.1.5: O **fluxo máximo** é o fluxo de maior valor possível.

4.2. Caminhos que aumentam o fluxo

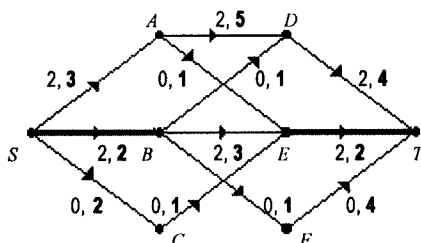
Para encontrar o fluxo máximo, começa-se por encontrar um fluxo inicial por inspecção. Pode-se começar com um em que cada arco tem valor zero e depois aumentá-lo passo a passo.

Usa-se o exemplo anterior para explicar este procedimento. Já se tinha encontrado um fluxo de S para T de valor 6. Para facilitar o procedimento, representam-se os arcos saturados a negro:



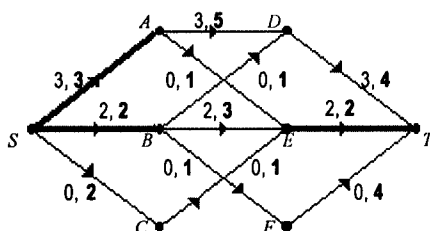
Um fluxo de S para T constituído inteiramente por arcos não saturados é um exemplo de um *caminho que aumenta o fluxo*. Aumentando o fluxo ao longo dos arcos não saturados de tal caminho pode-se aumentar o valor do fluxo. Nesta rede não existe nenhum caminho constituído apenas por arcos não saturados, uma vez que os arcos que entram em T já são saturados, o que significa que não se pode aumentar o fluxo.

Usemos outro exemplo de uma rede básica:

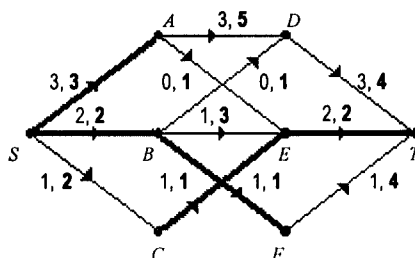


Nesta rede o fluxo que sai de S e o fluxo que entra em T é 4, logo um fluxo de S para T tem valor 4.

O caminho $SADT$ é formado por arcos não saturados e o fluxo pode ser aumentado em 1 ao longo deste, tornando assim SA saturado de fluxo 3, AD e DT com fluxo 3, aumentando o valor do fluxo de 4 para 5:

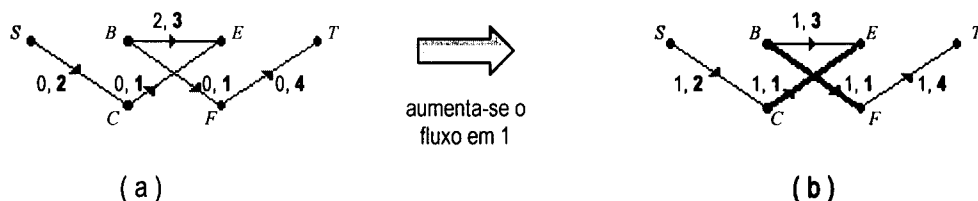


Analisando a rede, verifica-se que não há caminhos formados apenas por arcos não saturados, uma vez que esse caminho deve começar por SCE e não há arcos não saturados a saírem de E , logo o processo não pode ser repetido. De uma análise mais atenta decorre que 5 não é o fluxo máximo, havendo um fluxo de valor 6 exemplificado no diagrama seguinte:



Este método não é muito praticável, por isso, define-se mais cuidadosamente o significado de "caminho que aumenta o fluxo", pois, como vimos no exemplo anterior, não basta ser um caminho que consiste inteiramente em arcos não saturados.

Use-se parte da rede anterior para estudar este caso:



Pode-se obter um valor de fluxo 6 de um de valor 5 aumentando o fluxo em 1 ao longo de SC , CE , BF e FT e diminuindo o fluxo em 1 ao longo de EB . Note-se que se escreveu o arco dirigido para trás como EB e não como BE , uma vez que o fluxo é direccionado de S para T .

Isto é possível, desde que os arcos dirigidos para a frente sejam todos não saturados (pois o fluxo ao longo deste pode ser aumentado) e os arcos dirigidos para trás tenham fluxo não nulo.

Definição 4.2.1: Um caminho que aumenta o fluxo numa rede básica com fonte S e destino T é um caminho de S para T que consiste em:

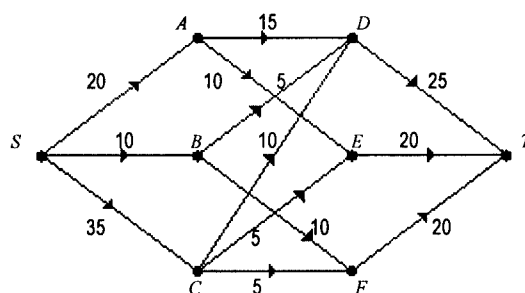
- (i) **arcos dirigidos para a frente:** arcos não saturados dirigidos ao longo do caminho,
- (ii) **arcos dirigidos para trás:** arcos dirigidos contra a direcção do caminho e transportando fluxo não nulo.

Um procedimento que pode ser efectuado para encontrar o número máximo para o qual o fluxo pode ser aumentado, em caminhos que aumentam o fluxo é o seguinte:

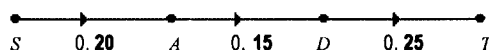
- primeiro, para cada arco dirigido para a frente, calcular o maior número para o qual o fluxo pode ser aumentado sem exceder a capacidade. Seja r o menor desses números;
- em seguida, para cada arco dirigido para trás, calcular o maior número para o qual o fluxo pode ser diminuído sem se tornar negativo. Seja s o menor desses números;

O número pretendido é o menor de r e s .

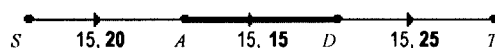
Exemplo 4.2.2: Determinar o fluxo máximo da seguinte rede básica encontrando o caminho que aumenta o fluxo (o número ao lado do arco é a capacidade do arco).



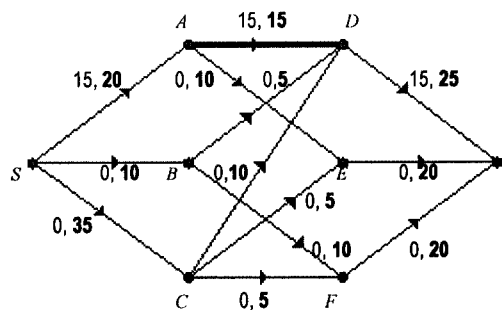
1) Começamos com o caminho que aumenta o fluxo $SADT$ e com $r = 0$.



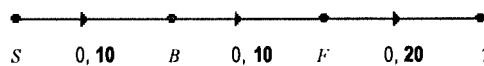
$r = \min(20, 15, 25) = 15$, ou seja, o fluxo pode ser aumentado 15 unidades, ficando o arco AD saturado:



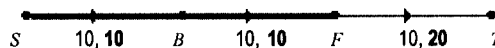
A rede fica:



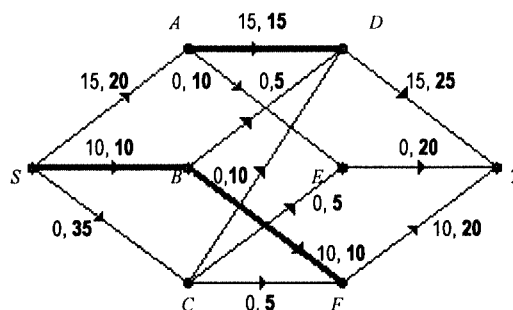
2) Analisemos em seguida o caminho *SBFT* :



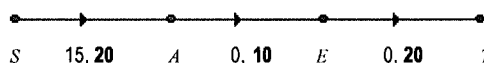
$r = \min(10, 10, 20) = 10$, ou seja, o fluxo pode ser aumentado 10 unidades, ficando os arcos *SB* e *BF* saturados:



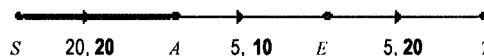
A rede fica:



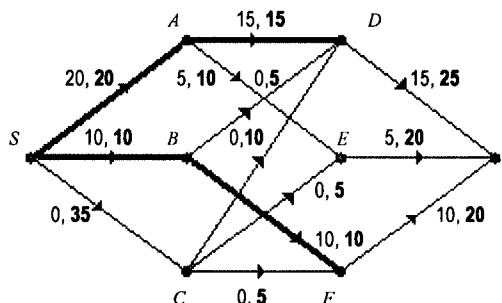
3) Analisemos o caminho *SAET* :



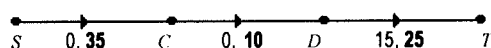
$r = \min(5, 10, 20) = 5$, ou seja, o fluxo pode ser aumentado 5 unidades, ficando o arco *SA* saturado:



A rede fica:

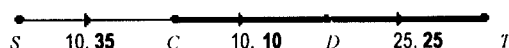


4) Analisemos o caminho *SCDT* :

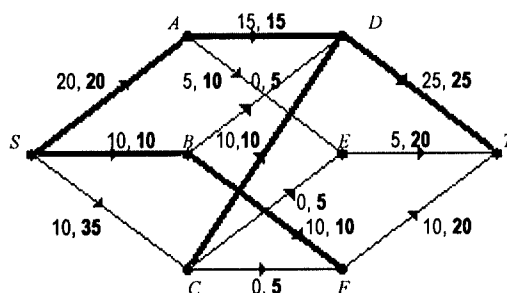


$r = \min(35, 10, 10) = 10$, ou seja, o fluxo pode ser aumentado 10 unidades, ficando os arcos *CD* e *DT*

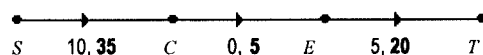
saturados:



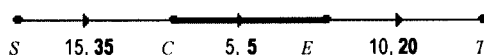
A rede fica:



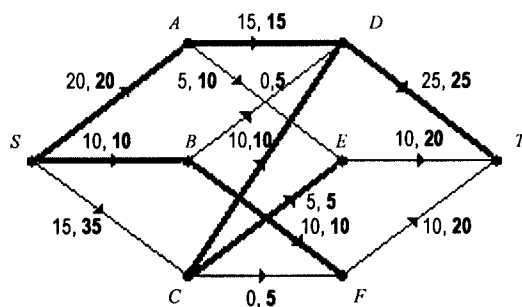
5) Analisemos o caminho *SCET* :



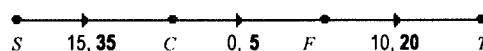
$r = \min(25, 5, 15) = 5$, ou seja, o fluxo pode ser aumentado 5 unidades, ficando o arco *CE* saturado:



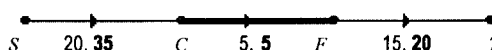
A rede fica:



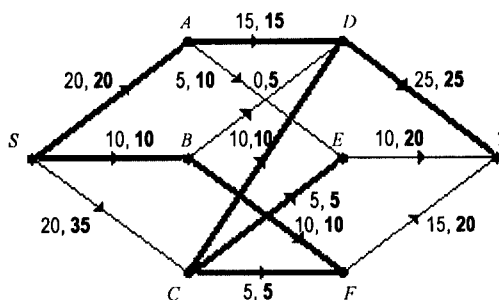
6) Analisemos o caminho *SCFT* :



$r = \min(20, 5, 10) = 5$, ou seja, o fluxo pode ser aumentado 5 unidades, ficando o arco *CF* saturado:

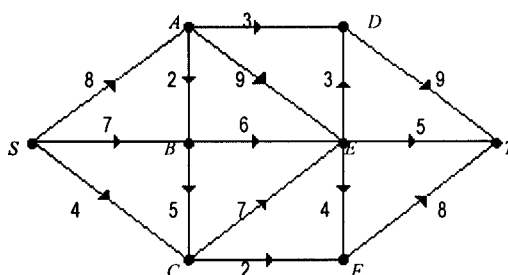


A rede fica:

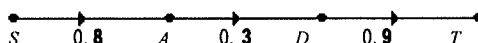


Analisando a rede, verifica-se que não existem mais caminhos que aumentam o fluxo. Os caminhos possíveis teriam de começar com o arco SC e acabar com o arco ET ou FT , e não é possível aumentar o fluxo, logo o fluxo máximo é $25 + 10 + 15 = 50$.

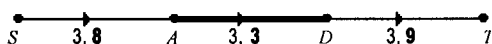
Exemplo 4.2.3: Determinar o fluxo máximo da seguinte rede básica encontrando o caminho que aumenta o fluxo (o número ao lado do arco é a capacidade do arco).



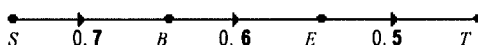
1) Começamos com o caminho que aumenta o fluxo $SADT$ e com $r = 0$.



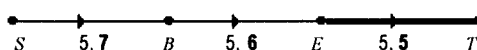
$r = \min(8,3,9) = 3$, ou seja, o fluxo pode ser aumentado 3 unidades, ficando o arco AD saturado:



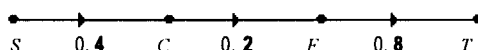
2) Analisemos o caminho $SBET$:



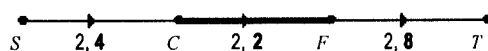
$r = \min(7,6,5) = 5$, ou seja, o fluxo pode ser aumentado 5 unidades, ficando o arco ET saturado:



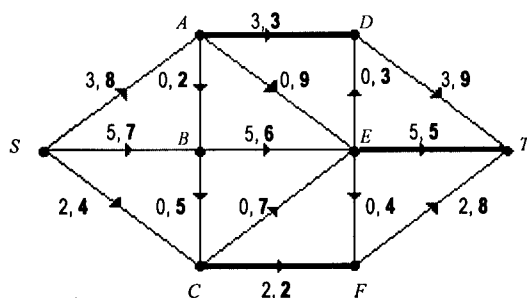
3) Analisemos o caminho $SCFT$:



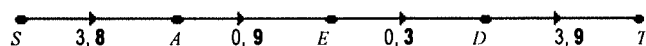
$r = \min(4, 2, 8) = 2$, ou seja, o fluxo pode ser aumentado 2 unidades, ficando o arco CF saturado:



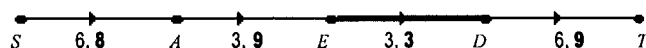
Neste momento a rede está do seguinte modo:



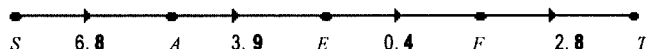
4) Analisemos o caminho $SAEDT$:



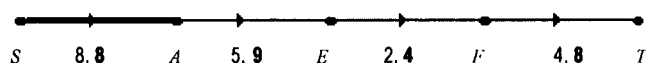
$r = \min(5, 9, 3, 6) = 3$, ou seja, o fluxo pode ser aumentado 3 unidades, ficando o arco ED saturado:



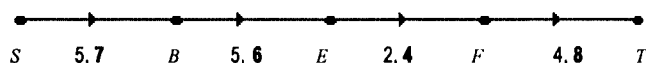
5) Analisemos o caminho $SAEFT$:



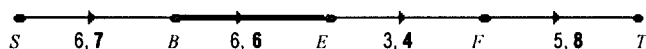
$r = \min(2, 6, 4, 6) = 2$, ou seja, o fluxo pode ser aumentado 2 unidades, ficando o arco SA saturado:



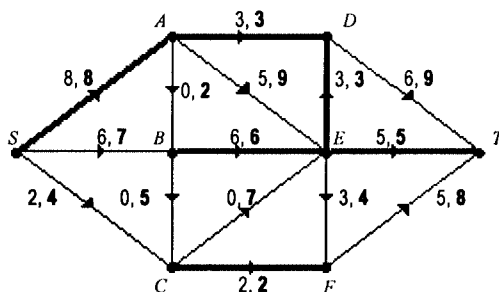
6) Analisemos o caminho $SBEFT$:



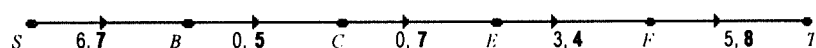
$r = \min(2, 1, 2, 4) = 1$, ou seja, o fluxo pode ser aumentado 1 unidade, ficando o arco BE saturado:



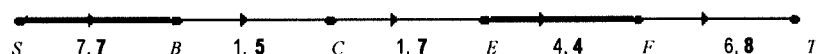
A rede fica:



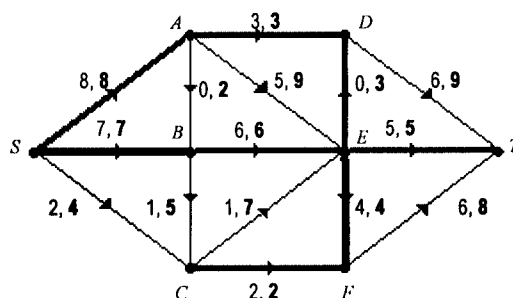
7) Analisemos o caminho *SBCEFT* :



$r = \min(1, 5, 7, 1, 3) = 1$, ou seja, o fluxo pode ser aumentado 1 unidade, ficando os arcos *SB* e *EF* saturados:



A rede fica:



Analisando a rede, verifica-se que não existem mais caminhos que aumentam o fluxo, uma vez que um tal caminho teria de começar com o arco *SC* e acabar com o arco *DT* ou *FT* passando pelo arco *CE* (pois é o único arco não saturado que sai do vértice *C*), o que não é possível, uma vez que todos os arcos que saem de *E* estão saturados, logo o fluxo máximo é $6 + 5 + 6 = 17$.

Claro que este processo é fácil de verificar em redes pequenas. Mas que fazer com redes muito extensas? Precisa-se de certeza de um método mais sistemático para encontrar o fluxo máximo.

Antes de se passar a um método mais eficaz para encontrar o fluxo máximo vejam-se algumas transformações necessárias para transformar uma rede numa rede básica.

4.3. Transformação numa rede básica

Muitas redes que aparecem na prática não satisfazem as condições de uma rede básica:

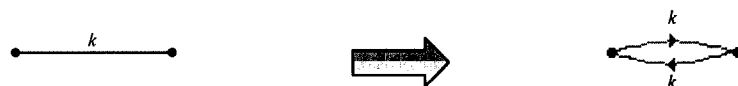
- todas as arestas são dirigidas;
- não há restrições nas capacidades dos vértices;
- há apenas uma fonte e um destino.

Nesta subsecção estuda-se cada um destes casos, desenvolvendo ideias para adaptar a descoberta do fluxo máximo em redes que não satisfazem estas condições.

Primeiro caso: Redes não dirigidas e redes misturadas

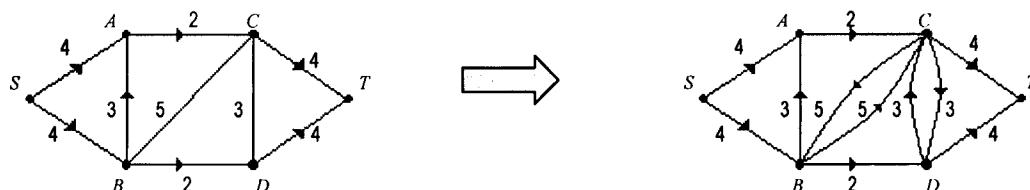
Definição 4.3.1: Uma **rede não dirigida** é uma rede na qual o fluxo pode ir em qualquer direcção ao longo da aresta. Uma rede contendo arestas dirigidas e não dirigidas é designada de **rede misturada**.

Pode-se transformar uma rede misturada ou não dirigida numa rede básica, substituindo uma aresta não dirigida por dois arcos, um em cada direcção, com a mesma capacidade:



Um exemplo deste tipo de redes é uma rede de estradas de uma cidade constituída por ruas com um ou dois sentidos.

Por exemplo, a seguinte rede misturada com duas arestas não dirigidas pode ser transformada numa rede básica:

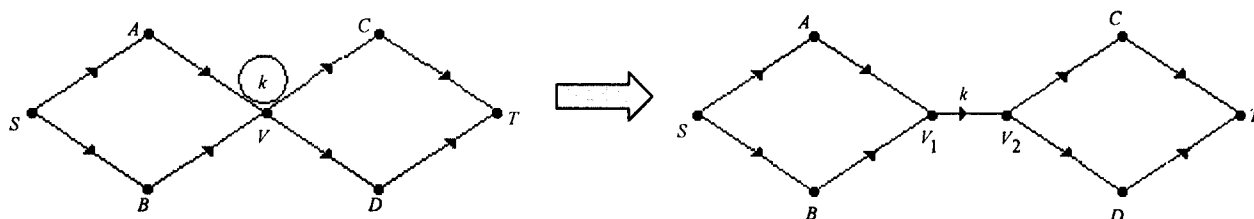


Segundo caso: Redes com restrições de capacidade nos vértices

Se a rede tem um ou mais vértices com restrições de capacidade, pode ser transformada numa rede básica, substituindo esses vértices por dois vértices unidos por um arco.

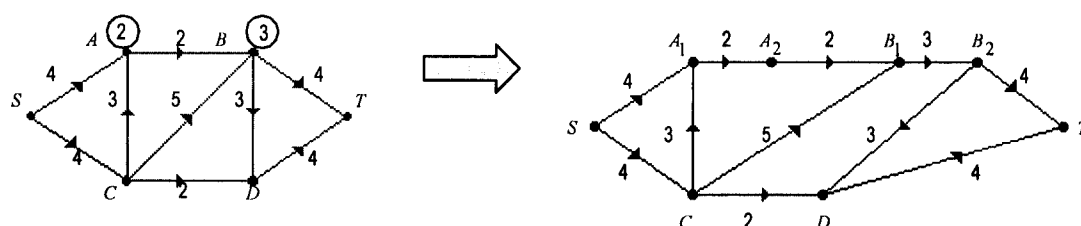
Um exemplo deste tipo de redes é uma rede de estradas de uma cidade na qual os cruzamentos têm fluxo de tráfego restrito devido aos sinais de trânsito.

Neste tipo de redes substitui-se cada vértice V de capacidade k por dois vértices V_1 e V_2 unidos por um arco de V_1 para V_2 de capacidade k .



Repare-se que os arcos que entram em V vão entrar no vértice V_1 e os que saem de V vão sair do vértice V_2 .

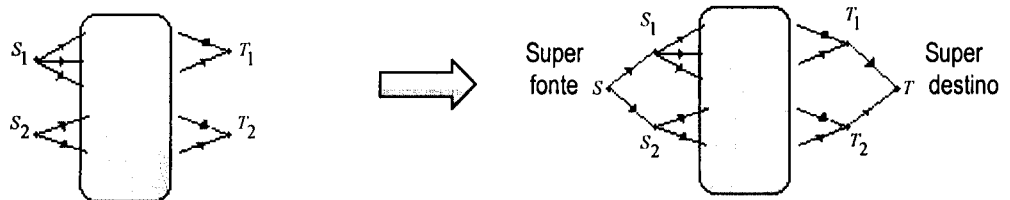
Por exemplo, a seguinte rede com restrições de capacidade nos vértices pode ser transformada numa rede básica:



Terceiro caso: Redes com várias fontes e destinos

Se uma rede tem várias fontes $S_1, S_2, \dots$, e vários destinos $T_1, T_2, \dots$ pode-se transformá-la numa rede básica juntando dois novos vértices – uma *super fonte* S unindo todas as fontes ao novo arco, e um *super destino* T unindo todos os destinos ao novo arco.

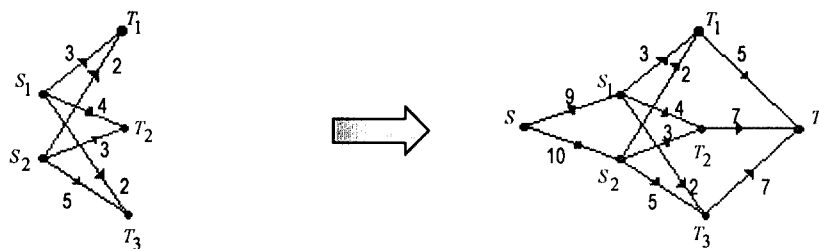
Um exemplo deste tipo de redes é um conjunto de subscritores de telefones de uma rede de telecomunicações.



A cada novo arco SS_i é atribuída uma capacidade igual à soma das capacidades dos arcos que saem de S_i e a cada novo arco T_iT é atribuída uma capacidade igual à soma das capacidades dos arcos que entram em T_i .

Note-se que o valor máximo de fluxo numa rede básica de S para T é igual ao valor do fluxo máximo de todas as fontes originais para os destinos originais.

Por exemplo, na seguinte rede ao arco SS_1 é dada a capacidade $3 + 4 + 2 = 9$, ao SS_2 é dada a capacidade $5 + 3 + 2 = 10$, ao T_1T é dada a capacidade $3 + 2 = 5$, ao T_2T é dada a capacidade $4 + 3 = 7$ e ao T_3T é dada a capacidade $2 + 5 = 7$.



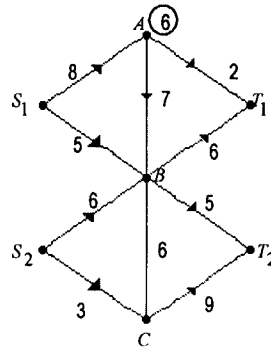
Resumindo: as transformações para converter uma rede numa rede básica são:

1. Substituir as arestas não dirigidas por dois arcos, um em cada direcção, ambos com a mesma capacidade que a aresta original não dirigida;

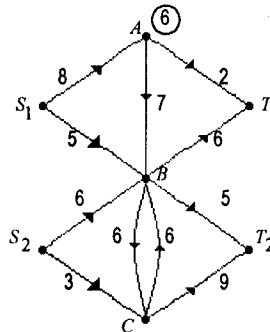
2. Substituir cada vértice V com restrição de capacidade k por dois vértices V_1 e V_2 unidos por um arco de V_1 para V_2 de capacidade k . Todos os arcos que entram em V serão enviados em V_1 e todos os vértices que saem de V sairão de V_2 ;

3. Se há várias fontes $S_1, S_2, \dots, S_i, \dots$ juntam-se numa nova fonte S . Se há vários destinos $T_1, T_2, \dots, T_i, \dots$ juntam-se num novo destino T . Para cada arco SS_i é atribuída uma capacidade igual à soma das capacidades dos arcos que saem de S_i e a cada novo arco T_iT é atribuída uma capacidade igual à soma das capacidades dos arcos que entram em T_i .

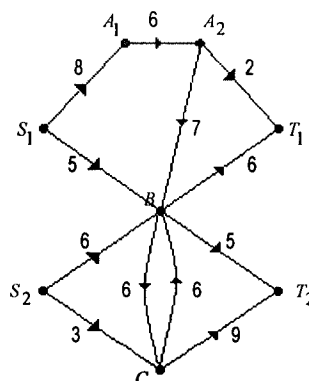
O exemplo seguinte mostra a transformação de uma rede numa rede básica usando estas três transformações:



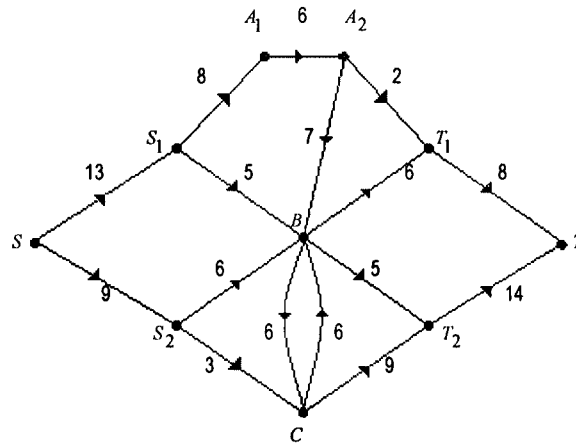
Primeiro passo: substituição da aresta não dirigida BC de capacidade 6, por dois arcos, um em cada direcção, ambos com capacidade 6.



Segundo passo: substituição do vértice A com restrição de capacidade 6 por dois vértices A_1 e A_2 unidos por um arco de A_1 para A_2 de capacidade 6. Note-se que o arco S_1A será enviado no arco S_1A_1 e os arcos AT_1 e AB serão enviados em A_2T_1 e A_2B .



Terceiro passo: juntam-se as duas fontes S_1 e S_2 numa só fonte S e cada arco SS_1 e SS_2 terá capacidade igual à soma das capacidades dos arcos que saem de S_1 e S_2 , respectivamente. Analogamente, os destinos T_1 e T_2 juntam-se num só destino T e cada arco T_1T e T_2T terá capacidade igual à soma das capacidades dos arcos que entram em T_1 e T_2 , respectivamente.



Procedimento geral

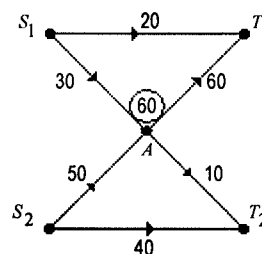
Pode-se resolver uma variedade de problemas de redes de fluxo usando o seguinte procedimento:

Passo 1: usando as transformações descritas anteriormente transforma-se o problema numo envolvendo uma rede básica;

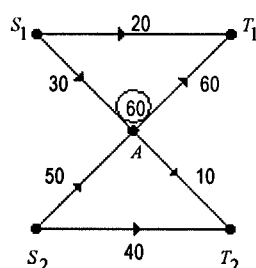
Passo 2: resolve-se o problema de rede básica;

Passo 3: interpreta-se a solução em termos da rede original.

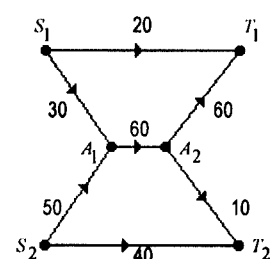
Exemplo 4.3.2: Usando o procedimento anterior encontra-se o fluxo máximo da seguinte rede, com as fontes S_1 e S_2 e os destinos T_1 e T_2 :

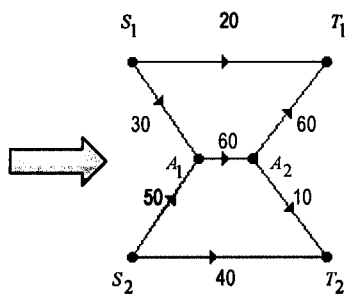


Passo 1: transformação numa rede básica, usando o segundo e terceiro passos desta transformação:

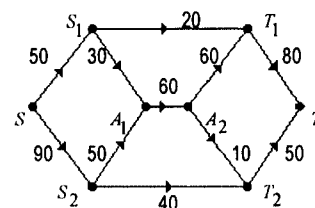


Substitui-se o vértice A com restrição de capacidade de 60 por dois vértices A_1 e A_2 , unidos por um arco de A_1 para A_2 de capacidade 60.



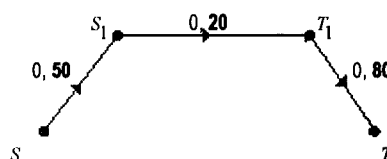


Juntam-se as duas fontes s_1 e s_2 numa fonte s e cada arco ss_1 e ss_2 tem capacidade igual à soma das capacidades dos arcos que saem de s_1 e s_2 . Do mesmo modo juntam-se os dois destinos t_1 e t_2 num destino t e cada arco t_1t e t_2t tem capacidade igual à soma das capacidades dos arcos que entram em t_1 e t_2 .

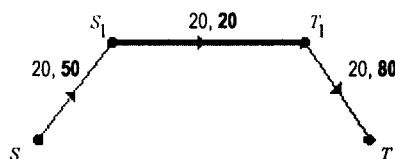


Passo 2: comece-se por analisar alguns caminhos que aumentam o fluxo:

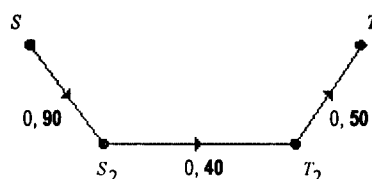
1) O caminho que aumenta o fluxo ss_1t_1t e com $r = 0$.



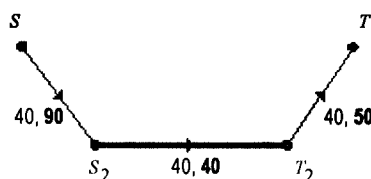
$r = \min(50, 20, 80) = 20$, ou seja, o fluxo pode ser aumentado 20 unidades, ficando o arco s_1t_1 saturado:



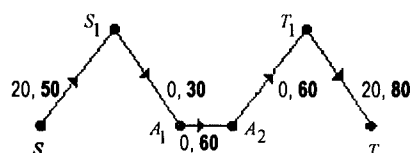
2) O caminho que aumenta o fluxo ss_2t_2t :



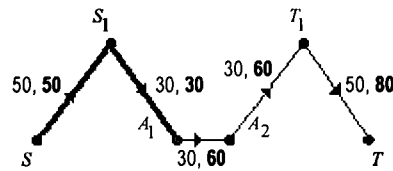
$r = \min(90, 40, 50) = 40$, ou seja, o fluxo pode ser aumentado 40 unidades, ficando o arco s_2t_2 saturado:



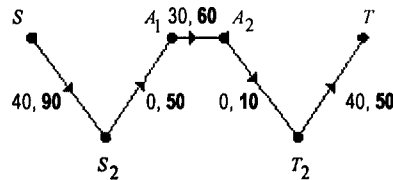
3) O caminho que aumenta o fluxo $ss_1A_1A_2t_1t$:



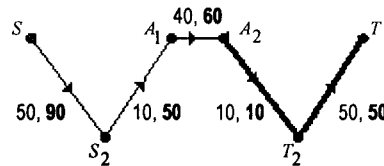
$r = \min(30, 30, 60, 60, 60) = 30$, ou seja, o fluxo pode ser aumentado 30 unidades, ficando os arcos SS_1 e S_1A_1 saturados:



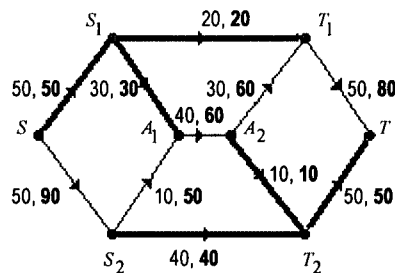
4) O caminho que aumenta o fluxo $SS_2A_1A_2T_2T$:



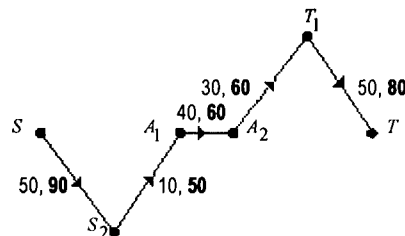
$r = \min(50, 50, 30, 10, 10) = 10$, ou seja, o fluxo pode ser aumentado 10 unidades, ficando os arcos A_2T_2 e T_2T saturados:



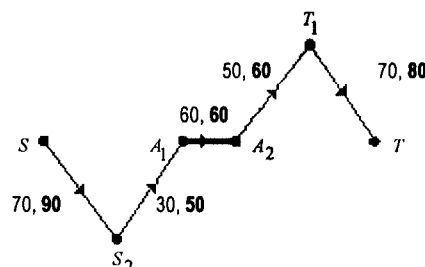
Neste momento, tem-se:



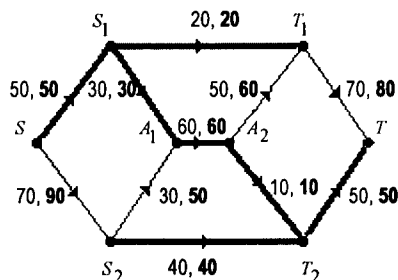
5) Outro caminho que aumenta o fluxo é $SS_2A_1A_2T_1T$:



$r = \min(40, 40, 20, 30, 30) = 20$, ou seja, o fluxo pode ser aumentado 20 unidades, ficando o arco A_1A_2 saturado:

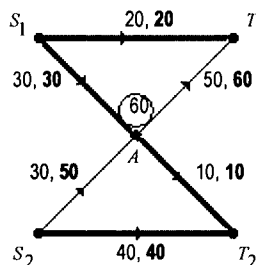


Neste momento, tem-se:



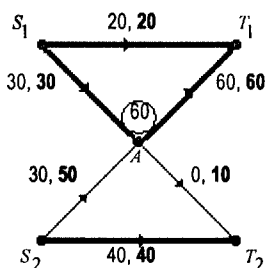
Não existe mais nenhum caminho que aumenta o fluxo porque este teria de passar pelos arcos S_1T_1 ou A_1A_2 ou S_2T_2 , os quais são saturados. Deste modo o valor do fluxo não pode ser aumentado, logo o fluxo máximo é 120.

Passo 3: A interpretação que pode ser feita em termos da rede original é:

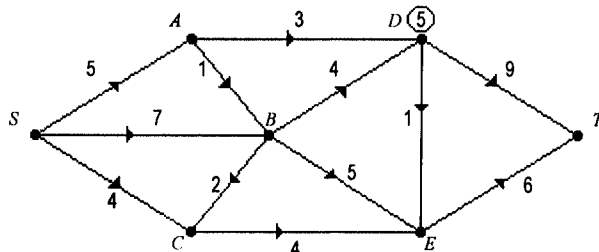


tendo esta 120 como fluxo máximo.

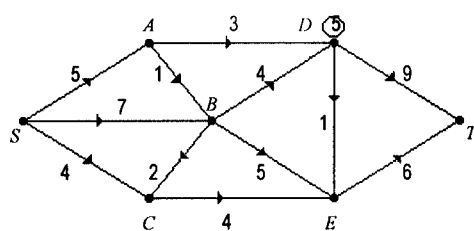
Note-se que outra forma de obter o fluxo máximo é considerando um fluxo de 60 e 0 nos arcos AT_1 e AT_2 , respectivamente:



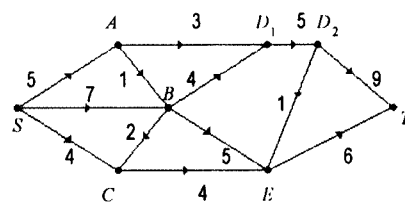
Exemplo 4.3.3: Uma empresa é proprietária de um parque de diversões. Na figura abaixo, os vértices representam pontos de diversão e os arcos as respectivas ligações, onde os visitantes são transportados por pequenos comboios eléctricos. Dado que os caminhos estão um pouco danificados e o material circulante representa sinais de envelhecimento, apenas podem ser realizadas diariamente as viagens assinaladas (a que correspondem as capacidades dos arcos). Quantas viagens se podem fazer diariamente de modo a levar o número máximo de visitantes diários da entrada em S até à montanha russa em T ?



Passo 1: transformação numa rede básica, usando o segundo passo desta transformação:

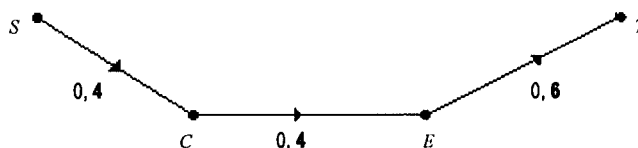


Substitui-se o vértice D com restrição de capacidade de 5 por dois vértices D_1 e D_2 , unidos por um arco de D_1 para D_2 de capacidade 5. Os vértices que entram em D são enviados em D_1 e os que saem de D saíram de D_2 .

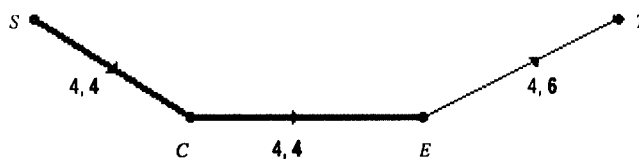


Passo 2: comece-se por analisar alguns caminhos que aumentam o fluxo:

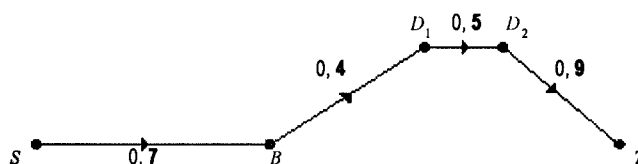
1) O caminho que aumenta o fluxo $SCET$ e com $r = 0$.



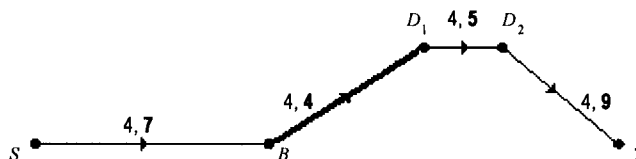
$r = \min(4, 4, 6) = 4$, ou seja, o fluxo pode ser aumentado 4 unidades, ficando os arcos SC e CE saturados:



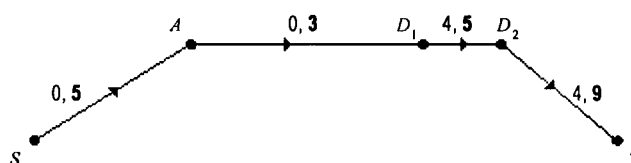
2) O caminho que aumenta o fluxo SBD_1D_2T :



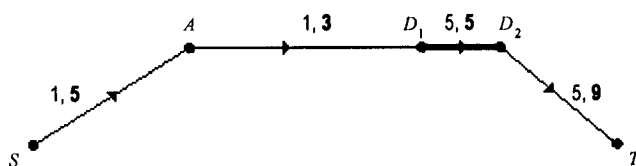
$r = \min(7, 4, 5, 9) = 4$, ou seja, o fluxo pode ser aumentado 4 unidades, ficando o arco BD_1 saturado:



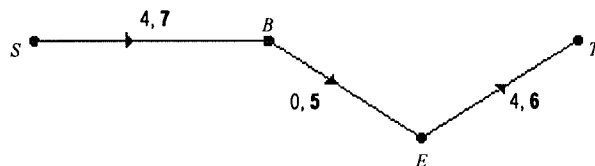
3) O caminho que aumenta o fluxo SAD_1D_2T :



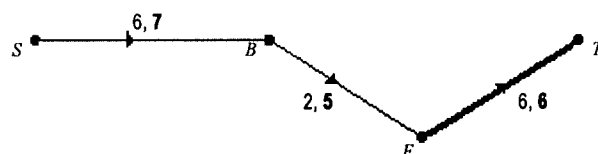
$r = \min(5, 3, 1, 5) = 1$, ou seja, o fluxo pode ser aumentado 1 unidade, ficando o arco D_1D_2 saturado:



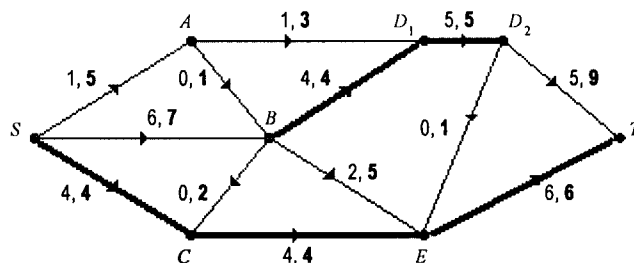
4) O caminho que aumenta o fluxo $SBET$:



$r = \min(3, 5, 2) = 2$, ou seja, o fluxo pode ser aumentado 2 unidades, ficando o arco ET saturado:

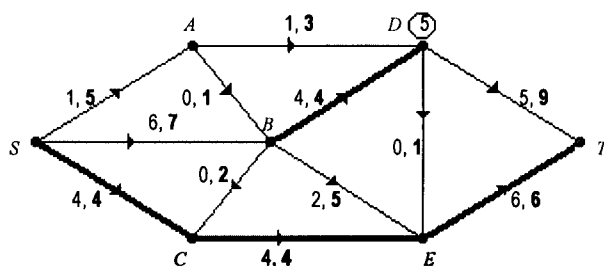


Depois destas alterações, a rede fica do seguinte modo:



Repare-se que não existem mais caminhos que aumentam o fluxo, pois tais caminhos teriam de passar pelo arco DT e consequentemente pelo D_1D_2 , o qual já está saturado.

Passo 3: A interpretação que pode ser feita em termos da rede original é:



Analisando a rede verifica-se que existem caminhos que já não podem ser usados, pois têm fluxo zero e diariamente haverá no máximo 11 viagens de comboio da entrada para a montanha russa.

Mas será que podemos ter a certeza que este é o fluxo máximo? Não existirá um método mais eficiente?

Vejamos alguns conceitos que nos ajudarão a chegar a um método mais eficaz.

5. Fluxos máximos e cortes mínimos

Um conceito importante relacionado com o fluxo máximo é o corte mínimo. No capítulo anterior descreveu-se uma maneira de obter o fluxo máximo numa rede básica pequena. Neste método, primeiro identificam-se os caminhos que aumentam o fluxo por inspecção e, depois, aumenta-se o fluxo passo por passo até se obter o fluxo máximo. Mas este método não é muito satisfatório para redes de grandes dimensões, nas quais os caminhos que aumentam o fluxo podem não ser óbvios e pode não haver maneira de saber quando se encontra o fluxo máximo.

Neste capítulo descreve-se um método alternativo para determinar o fluxo máximo tendo um fluxo e um algoritmo, designado de *algoritmo do fluxo máximo*, que permite identificar sistematicamente os caminhos que aumentam o fluxo e encontrar, assim, o fluxo máximo numa rede. Com este algoritmo obtém-se também o *corte mínimo*.

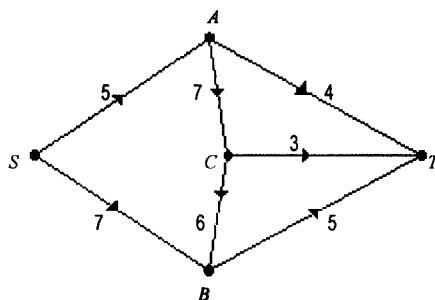
Algumas definições serão feitas de modo alternativo para estarem em coerência com o algoritmo que será apresentado nesta secção.

5.1. Cortes mínimos

Antes de introduzir a definição de corte introduz-se o conceito de corte de partição, para depois relacionar estes conceitos com o de conjunto de vértices que separam s de t .

Para dois quaisquer conjuntos de vértices X e Y da rede N , $\langle X, Y \rangle$ representa todos os arcos de N dirigidos de um vértice de X para um vértice de Y .

Por exemplo:



Se $X = \{A, B\}$ e $Y = \{C, T\}$, os elementos do conjunto dos arcos $\langle X, Y \rangle$ são o arco dirigido do vértice A para o vértice C , o arco dirigido do vértice A para o vértice T e o arco dirigido do vértice B para o vértice T . O único elemento do conjunto de arcos $\langle Y, X \rangle$ é o arco dirigido do vértice C para o vértice B .

Definição 5.1.1: Sejam G um grafo e X_1 e X_2 conjuntos que formam uma partição dos vértices de G , V_G . O conjunto de todas as arestas que têm início em X_1 e fim em X_2 , $\langle X_1, X_2 \rangle$, é designado por **corte de partição** de G .

Por exemplo, usando a rede do exemplo anterior, se $X_1 = \{S, A, C\}$ e $X_2 = \{B, T\}$, então $\langle X_1, X_2 \rangle$ é um corte de partição, pois X_1 e X_2 formam uma partição de V_G .

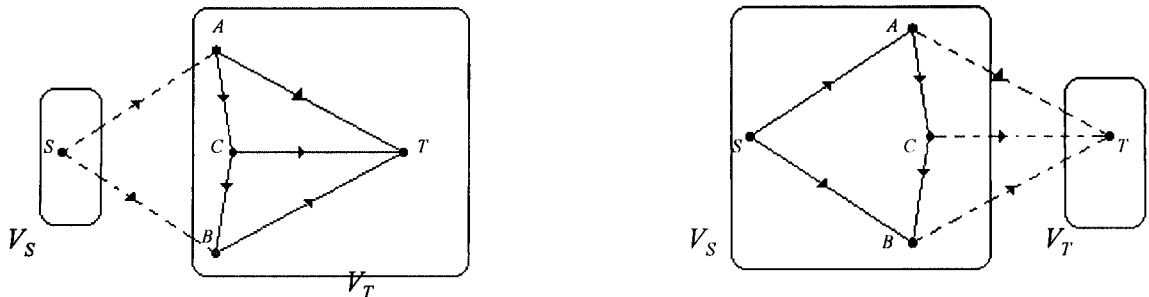
Definição 5.1.2: Sejam N uma rede básica com fonte S e destino T , V_S e V_T conjuntos que formam uma partição de V_N , tais que $S \in V_S$ e $T \in V_T$. O conjunto de todos os arcos dirigidos de um vértice de V_S para um vértice de V_T , $\langle V_S, V_T \rangle$, é designado de **corte da rede** N .

Por outras palavras, **um corte** é um conjunto de arcos que, removidos, separam a rede em duas partes disjuntas: V_S , contendo S , e V_T , contendo T .

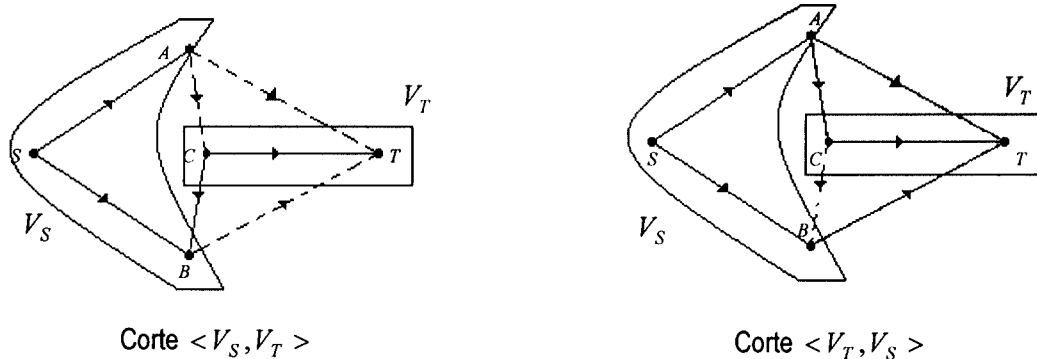
Para qualquer rede N , um arco do conjunto $O(S)$ é um elemento do corte $\langle \{S\}, V_N - \{S\} \rangle$ e um em $I(T)$ é um elemento do corte $\langle V_N - \{T\}, \{T\} \rangle$. Analogamente, um arco de $I(S)$ é um elemento do corte $\langle V_N - \{S\}, \{S\} \rangle$ e um em $O(T)$ é um elemento do corte $\langle \{T\}, V_N - \{T\} \rangle$. Repare-se que estes dois últimos cortes são conjuntos vazios.

Nota 5.1.3: Mais geralmente, se existem n vértices intermédios numa rede N , então N tem 2^n cortes, pois este é o número de subconjuntos de um conjunto de n elementos.

Exemplo 5.1.4: Neste exemplo, usam-se os conjuntos dos arcos de $O(S)$ e $I(T)$ como cortes, onde $O(S) = \langle \{S\}, \{A, B, C, T\} \rangle$ e $I(T) = \langle \{S, A, B, C\}, \{T\} \rangle$:



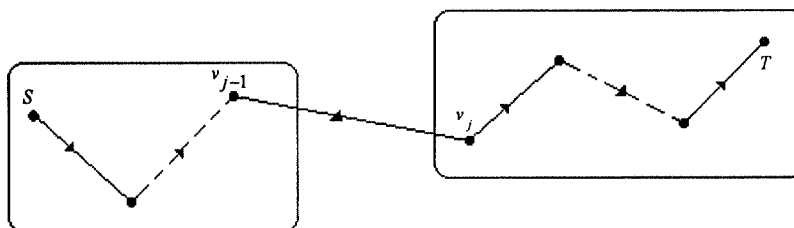
Usando $V_S = \{S, A, B\}$ e $V_T = \{C, T\}$, os cortes $\langle V_S, V_T \rangle$ e $\langle V_T, V_S \rangle$ ficam:



Proposição 5.1.5: Seja $\langle V_S, V_T \rangle$ um corte de uma rede N . Todos os arcos dirigidos de caminhos - st contêm pelo menos um arco em $\langle V_S, V_T \rangle$.

Demonstração:

Seja $P = \langle v_0, v_1, \dots, v_n \rangle$ a sequência de vértices dos caminhos - st dirigidos da rede N . Como $S \in V_S$ e $T \in V_T$, existe um primeiro vértice v_j neste caminho que está no conjunto V_T :



Então, o arco do vértice v_{j-1} para v_j está em $\langle V_S, V_T \rangle$. ■

Observe-se que, usando a definição de corte, o valor do fluxo, $val(f)$, pode ser reescrito da seguinte forma:

$$val(f) = \sum_{e \in \langle S, V_N - \{S\} \rangle} f(e) - \sum_{e \in \langle V_N - \{S\}, \{S\} \rangle} f(e).$$

Por outras palavras, o valor de qualquer fluxo é igual ao fluxo total ao longo dos arcos do corte $\langle \{S\}, V_N - \{S\} \rangle$ menos o fluxo total ao longo dos arcos $\langle V_N - \{S\}, \{S\} \rangle$. Note-se que $\sum_{e \in \langle V_N - \{S\}, \{S\} \rangle} f(e) = 0$.

Observação 5.1.6: Para um corte qualquer, $\langle V_S, V_T \rangle$, de uma rede N tem-se que:

$$\bigcup_{V \in V_S} O(V) = \langle V_S, V_S \rangle \cup \langle V_S, V_T \rangle \quad \text{e} \quad \bigcup_{V \in V_S} I(V) = \langle V_S, V_S \rangle \cup \langle V_T, V_S \rangle.$$

Proposição 5.1.7: Sejam f um fluxo de uma rede N e $\langle V_S, V_T \rangle$ um corte de N , então:

$$val(f) = \sum_{e \in \langle V_S, V_T \rangle} f(e) - \sum_{e \in \langle V_T, V_S \rangle} f(e).$$

Demonstração:

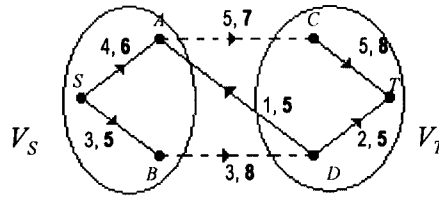
Pela definição, $val(f) = \sum_{e \in O(S)} f(e) - \sum_{e \in I(S)} f(e)$ e pela regra de conservação do fluxo,

$$\sum_{e \in O(V)} f(e) - \sum_{e \in I(V)} f(e) = 0 \quad \text{para todo o vértice } V \in V_S, \text{ para além de } S \text{ e } T.$$

Assim:

$$val(f) = \sum_{V \in V_S} \left(\sum_{e \in O(V)} f(e) - \sum_{e \in I(V)} f(e) \right) = \sum_{V \in V_S} \sum_{e \in O(V)} f(e) - \sum_{V \in V_S} \sum_{e \in I(V)} f(e).$$

Por exemplo, considere-se a seguinte rede e o corte $\langle V_S, V_T \rangle$, onde $V_S = \{S, A, B\}$ e $V_T = \{C, D, T\}$:



Neste caso:

$$\begin{aligned}
 7 = \text{val}(f) &= \sum_{e \in O(S)} f(e) - \underbrace{\sum_{e \in I(S)} f(e)}_{=0} + \sum_{e \in O(A)} f(e) - \underbrace{\sum_{e \in I(A)} f(e)}_{=0} + \sum_{e \in O(B)} f(e) - \underbrace{\sum_{e \in I(B)} f(e)}_{=0} \\
 &= \sum_{e \in O(S)} f(e) + \sum_{e \in O(A)} f(e) + \sum_{e \in O(B)} f(e) - \left(\sum_{e \in I(S)} f(e) + \sum_{e \in I(A)} f(e) + \sum_{e \in I(B)} f(e) \right) = 7 + 5 + 3 - (5 + 3) \\
 &= \sum_{V \in V_S} \sum_{e \in O(V)} f(e) - \sum_{V \in V_S} \sum_{e \in I(V)} f(e) = 15 - 8.
 \end{aligned}$$

Pela observação 5.1.6, tem-se que $\bigcup_{V \in V_S} O(V) = \langle V_S, V_S \rangle \cup \langle V_S, V_T \rangle$, logo:

$$\sum_{V \in V_S} \sum_{e \in O(V)} f(e) = \sum_{e \in \langle V_S, V_S \rangle} f(e) + \sum_{e \in \langle V_S, V_T \rangle} f(e)$$

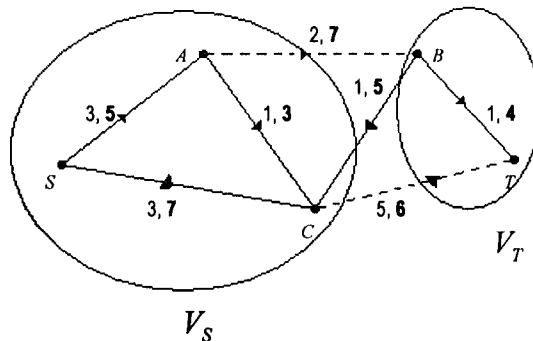
e $\bigcup_{V \in V_S} I(V) = \langle V_S, V_S \rangle \cup \langle V_T, V_S \rangle$, logo:

$$\sum_{V \in V_S} \sum_{e \in I(V)} f(e) = \sum_{e \in \langle V_S, V_S \rangle} f(e) + \sum_{e \in \langle V_T, V_S \rangle} f(e).$$

Assim, substituindo na expressão de $\text{val}(f)$:

$$\text{val}(f) = \sum_{e \in \langle V_S, V_S \rangle} f(e) + \sum_{e \in \langle V_S, V_T \rangle} f(e) - \left(\sum_{e \in \langle V_S, V_S \rangle} f(e) + \sum_{e \in \langle V_T, V_S \rangle} f(e) \right) = \sum_{e \in \langle V_S, V_T \rangle} f(e) - \sum_{e \in \langle V_T, V_S \rangle} f(e). \blacksquare$$

Exemplo 5.1.8: Considere-se a rede seguinte com o corte $\langle V_S, V_T \rangle$, onde $V_S = \{S, A, C\}$ e $V_T = \{B, T\}$:

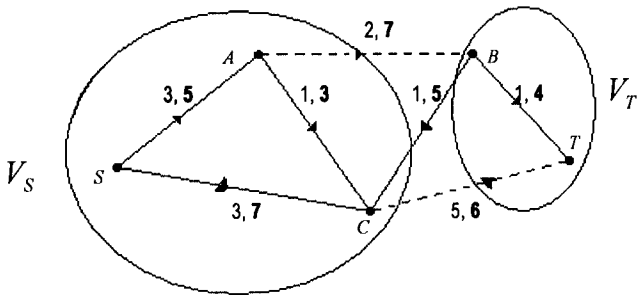


$$6 = 1 + 5 = \text{val}(f) = \sum_{e \in \langle \{S, A, C\}, \{B, T\} \rangle} f(e) - \sum_{e \in \langle \{B, T\}, \{S, A, C\} \rangle} f(e) = 2 + 5 - 1 = 7 - 1 = 6.$$

Definição 5.1.9: A **capacidade de um corte** $\langle V_S, V_T \rangle$, que se denota por $cap \langle V_S, V_T \rangle$, é a soma das capacidades dos arcos do corte $\langle V_S, V_T \rangle$, ou seja:

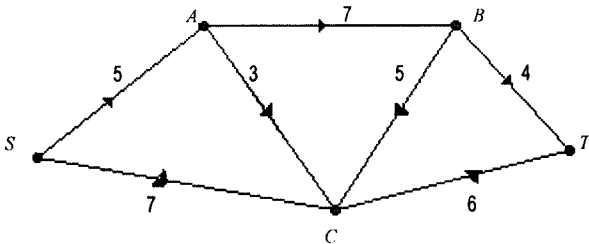
$$cap \langle V_S, V_T \rangle = \sum_{e \in \langle V_S, V_T \rangle} cap(e).$$

Exemplo 5.1.10: A capacidade do corte $\langle V_S, V_T \rangle$, onde $V_S = \{S, A, C\}$ e $V_T = \{B, T\}$ é $cap \langle \{S, A, C\}, \{B, T\} \rangle = 7 + 6 = 13$:



Definição 5.1.11: O **corte mínimo** de uma rede N é o corte com menor capacidade.

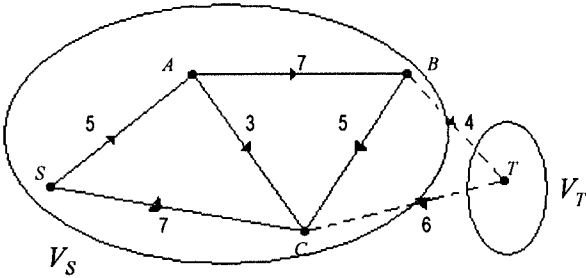
Exemplo 5.1.12: Para a rede do exemplo anterior listam-se os cortes desta, os vértices de V_S e V_T e a capacidade de cada corte. Qual é o corte mínimo?



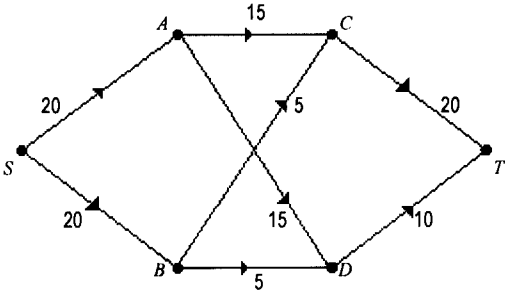
Esta rede tem 3 vértices intermédios, logo terá $2^3 = 8$ cortes:

	Arcos do corte	V_S	V_T	Capacidade do corte
1	SA, SC	S	A, B, C, T	$5 + 7 = 12$
2	SC, AB, AC	S, A	B, C, T	$7 + 7 + 3 = 17$
3	SA, SC, BC, BT	S, B	A, C, T	$5 + 7 + 5 + 4 = 21$
4	SA, CT	S, C	A, B, T	$5 + 6 = 11$
5	SC, AC, BC, BT	S, A, B	C, T	$7 + 3 + 5 + 4 = 19$
6	AB, CT	S, A, C	B, T	$7 + 6 = 13$
7	SA, BT, CT	S, B, C	A, T	$5 + 4 + 6 = 15$
8	BT, CT	S, A, B, C	T	$4 + 6 = 10$

O corte mínimo é o que tem capacidade 10: corte (8), que se encontra representado no seguinte diagrama:



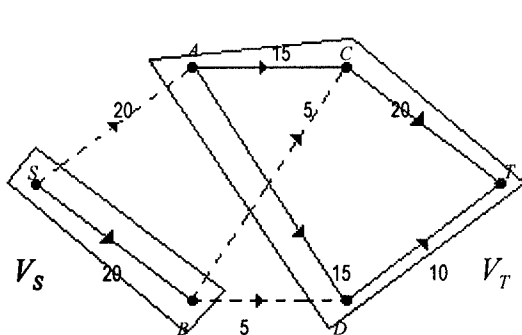
Exemplo 5.1.13: Para a seguinte rede listam-se os cortes, os vértices de V_S e V_T e a capacidade de cada corte. Quais são os cortes mínimos?



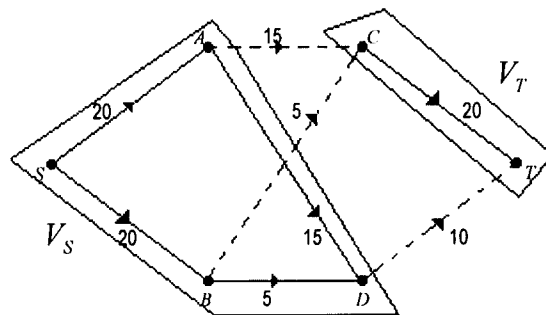
Esta rede tem 4 vértices intermédios, logo terá $2^4 = 16$ cortes:

	Arcos do corte	V_S	V_T	Capacidade do corte
1	SA, SB	S	A, B, C, D, T	$20 + 20 = 40$
2	SB, AC, AD	S, A	B, C, D, T	$20 + 15 + 15 = 50$
3	SA, BC, BD	S, B	A, C, D, T	$20 + 5 + 5 = 30$
4	SA, SB, CT	S, C	A, B, D, T	$20 + 20 + 20 = 60$
5	SA, SB, DT	S, D	A, B, C, T	$20 + 20 + 10 = 50$
6	AC, AD, BC, BD	S, A, B	C, D, T	$15 + 15 + 5 + 5 = 40$
7	SB, AD, CT	S, A, C	B, D, T	$20 + 15 + 20 = 55$
8	SB, AC, DT	S, A, D	B, C, T	$20 + 15 + 10 = 45$
9	SA, BD, CT	S, B, C	A, D, T	$20 + 5 + 20 = 45$
10	SA, BC, DT	S, B, D	A, C, T	$20 + 5 + 10 = 35$
11	SA, SB, CT, DT	S, C, D	A, B, T	$20 + 20 + 20 + 10 = 70$
12	AD, BD, CT	S, A, B, C	D, T	$15 + 5 + 20 = 40$
13	AC, BC, DT	S, A, B, D	C, T	$15 + 5 + 10 = 30$
14	SB, CT, DT	S, A, C, D	B, T	$20 + 20 + 10 = 50$
15	SA, CT, DT	S, B, C, D	A, T	$20 + 20 + 10 = 50$
16	CT, DT	S, A, B, C, D	T	$20 + 10 = 30$

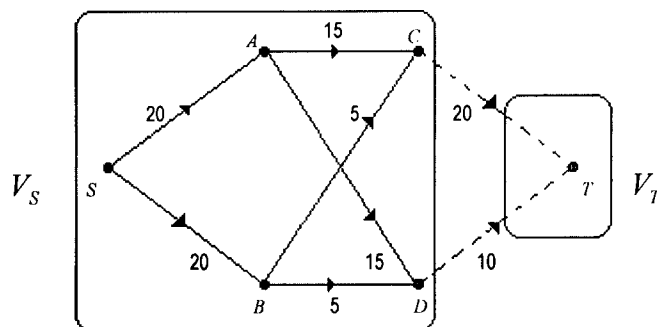
Os cortes mínimos são os que têm capacidade 30: cortes (3), (13) e (16); os quais estão desenhados nos seguintes diagramas:



Corte (3)



Corte (13)



Corte (16)

5.2. Problema do máximo fluxo e corte mínimo

Depois de se ter introduzido o conceito de corte, retoma-se o problema de encontrar o fluxo máximo de uma rede. A relação entre fluxos e cortes surge da observação seguinte:

o valor de qualquer fluxo $\leq$ a capacidade de qualquer corte.

Mas será que esta desigualdade é sempre válida?

Com a proposição seguinte obtém-se um limite superior para o problema do fluxo máximo.

Proposição 5.2.1: Sejam f um fluxo qualquer numa rede N e $\langle V_S, V_T \rangle$ um corte, então:

$$val(f) \leq cap \langle V_S, V_T \rangle.$$

Demonstração:

Usando a Proposição 5.1.7, decorre que:

$$\begin{aligned} val(f) &= \sum_{e \in \langle V_S, V_T \rangle} f(e) - \sum_{e \in \langle V_T, V_S \rangle} f(e) \\ &\downarrow \text{Pela condição de praticabilidade: fluxo} \leq \text{capacidade} \\ &\leq \sum_{e \in \langle V_S, V_T \rangle} cap(e) - \sum_{e \in \langle V_T, V_S \rangle} f(e) \\ &\downarrow \text{Pela definição de } cap \langle V_S, V_T \rangle \end{aligned}$$

$$\begin{aligned}
&= \text{cap} \langle V_S, V_T \rangle - \sum_{e \in \langle V_T, V_S \rangle} f(e) \\
&\downarrow \quad \text{Como cada } f(e) \text{ não é negativo, logo } \sum_{e \in \langle V_T, V_S \rangle} f(e) \geq 0 \text{ e } \text{fluxo} \leq \text{capacidade} \\
&\leq \text{cap} \langle V_S, V_T \rangle. \blacksquare
\end{aligned}$$

O seguinte corolário representa uma fraca dualidade entre fluxo máximo e corte mínimo.

Corolário 5.2.2: Sejam f^* um fluxo máximo de uma rede N e K^* um corte mínimo de N , então:

$$\text{val}(f^*) \leq \text{cap}(K^*).$$

Demonstração:

Este corolário é um caso particular da Proposição 5.2.1. $\blacksquare$

Observação 5.2.3: Dos dois resultados anteriores podemos afirmar que, se se encontrar um fluxo com valor k e um corte com capacidade igual a este valor k , então:

- o fluxo encontrado é o fluxo máximo (uma vez que qualquer fluxo maior infringirá a desigualdade);
- o corte encontrado é o corte mínimo (uma vez que qualquer corte menor infringirá a desigualdade).

Esta observação constitui o corolário seguinte:

Corolário 5.2.4: Sejam f um fluxo, K um corte de uma rede N e suponha-se que $\text{val}(f) = \text{cap}(K)$, então o fluxo f é um fluxo máximo e K é um corte mínimo da rede N .

Demonstração:

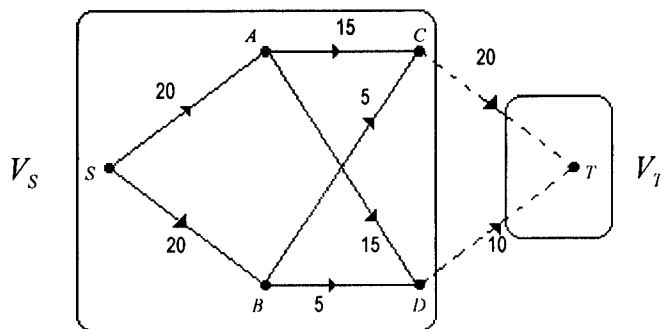
Seja f^* um fluxo máximo de uma rede N e K^* um corte mínimo de N . Pelo Corolário 5.2.2, decorre que:

$$\text{val}(f) \leq \text{val}(f^*) \leq \text{cap}(K^*) \leq \text{cap}(K).$$

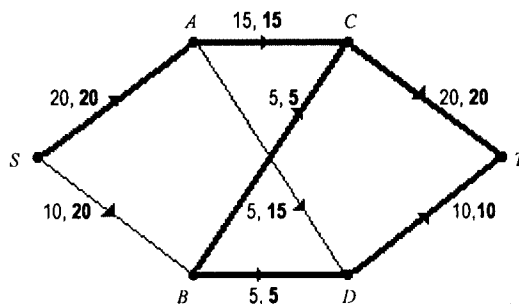
Como, por hipótese, $\text{val}(f) = \text{cap}(K)$, decorre que $\text{val}(f) = \text{val}(f^*)$ e $\text{cap}(K^*) = \text{cap}(K)$.

Deste modo, f é um fluxo máximo e K é um corte mínimo da rede N . $\blacksquare$

Exemplo 5.2.5: Para a seguinte rede foi visto que o corte $\langle \{S, A, B, C, D\}, \{T\} \rangle$ é mínimo de capacidade 30:



Aplicando o método descrito no Capítulo 4, descobre-se que esta rede tem fluxo 30:



Pelo Corolário 5.2.4, como o valor do fluxo é igual à capacidade do corte, logo o fluxo encontrado é o fluxo máximo.

Corolário 5.2.6: Sejam $\langle V_S, V_T \rangle$ um corte de uma rede N e f um fluxo tal que:

$$f(e) = \begin{cases} \text{cap}(e) & \text{se } e \in \langle V_S, V_T \rangle \\ 0 & \text{se } e \in \langle V_T, V_S \rangle \end{cases}$$

Então f é o fluxo máximo da rede N e $\langle V_S, V_T \rangle$ é o corte mínimo.

Demonstração:

Se $e \in \langle V_S, V_T \rangle$, então e é um arco dirigido de V_S para V_T . Como o fluxo deste arco é igual à sua capacidade, tal significa que o fluxo não pode ser aumentado, ou seja, o arco e é saturado.

Se $e \in \langle V_T, V_S \rangle$, então e é um arco dirigido de V_T para V_S . Como o fluxo deste arco é igual a zero, significa que não flui fluxo ao longo destes arcos.

Daqui decorre que o valor do fluxo da rede é igual à capacidade do corte $\langle V_S, V_T \rangle$. Logo, pelo Corolário 5.2.4, f é um fluxo máximo e $\langle V_S, V_T \rangle$ é um corte mínimo. ■

Nota 5.2.7: Todo o corte mínimo numa rede que carregue um fluxo máximo não precisa de ser formado inteiramente por arcos saturados.

5.3. Resolução do problema do fluxo máximo

Nesta secção, ir-se-ão ver as ideias que estão relacionadas com o algoritmo do fluxo máximo e explicar-se-á mais pormenorizadamente, o algoritmo do caminho que aumenta o fluxo. Este último algoritmo é baseado em dois conceitos intuitivos: uma rede residual e um caminho que aumenta o fluxo (propriamente dito).

Apresentar-se-ão definições análogas às apresentadas anteriormente no Capítulo 4 para caminhos que aumentam o fluxo, mas com a diferença que as novas definições serão escritas com uma notação que nos ajudará a introduzir o algoritmo do fluxo máximo.

Aumentar o fluxo usando caminhos directos

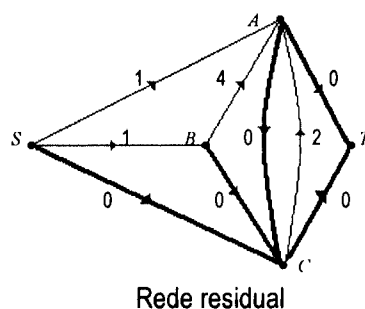
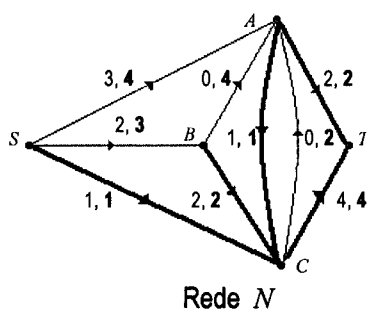
Suponha-se que f é um fluxo numa rede N e que existe um caminho - st dirigido, ou seja, existe P em N tal que:

$$P = \langle s, e_1, v_1, e_2, \dots, e_k, t \rangle,$$

tal que $f(e_i) < \text{cap}(e_i)$ para $i = 1, \dots, k$, ou seja, o caminho é formado por arcos que não são saturados. Assim, considerando apenas as capacidades dos arcos, o fluxo para cada arco e_i pode ser aumentado até ao valor de $\text{cap}(e_i) - f(e_i)$.

Definição 5.3.1: Uma **rede residual** é uma rede na qual os valores dos arcos representam os resíduos, ou seja, a diferença entre a capacidade e o fluxo do arco.

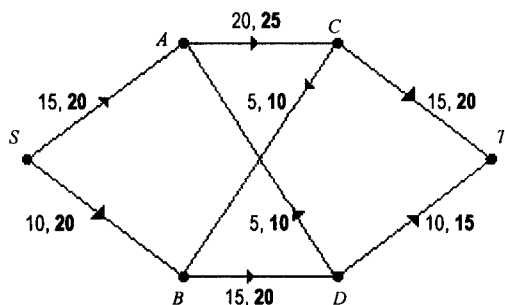
Por exemplo:



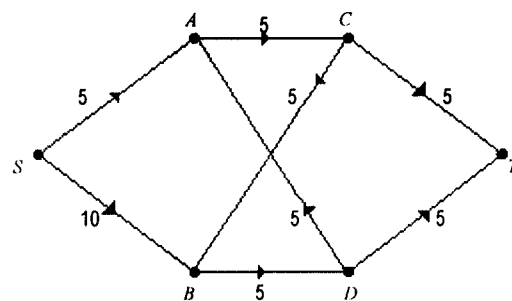
Tendo em conta a regra de conservação do fluxo para cada vértice v_i , o aumento de fluxo ao longo de todos os arcos do caminho P tem de ser igual. Denote-se por Δ_P esse aumento; o maior valor possível para Δ_P é $\min\{\text{cap}(e_i) - f(e_i)\}$. Tendo em conta este aumento, vão existir arcos (dirigidos para a frente) em que se aumenta o fluxo e outros (dirigidos para trás) onde se diminui.

A rede do exemplo anterior tem fluxo máximo 6, uma vez que o corte $\langle \{S, A, B, C\}, \{T\} \rangle$ também tem capacidade 6. Usemos outro exemplo para o qual é possível aumentar o fluxo.

Exemplo 5.3.2:



Rede N

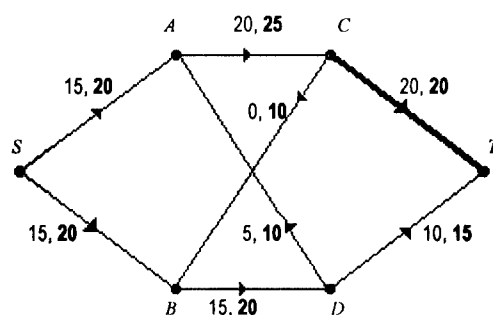
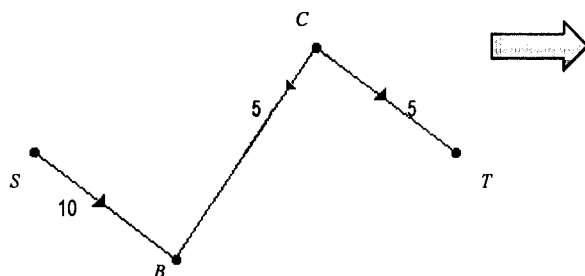


Rede residual

Considerando o caminho $P = \langle S, B, C, T \rangle$,

$$\Delta_P = \min\{10, 5, 5\} = 5$$

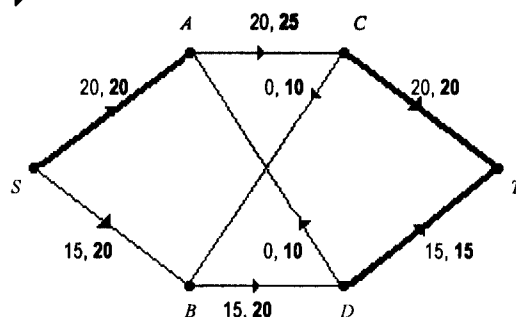
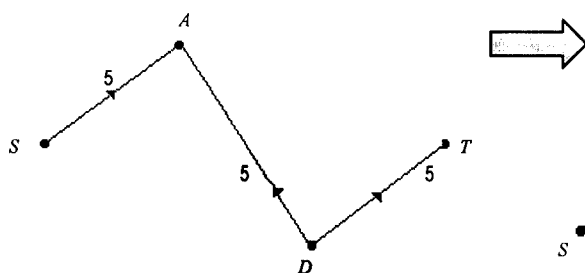
Aumentam-se 5 unidades nos arcos dirigidos para a frente (SB e CT) e diminuem-se 5 unidades no arco dirigido para trás (BC):



Considerando o caminho $P = \langle S, A, D, T \rangle$,

$$\Delta_P = \min\{5, 5, 5\} = 5$$

Aumentam-se 5 unidades nos arcos dirigidos para a frente (SA e DT) e diminuem-se 5 unidades no arco dirigido para trás (AD):



O fluxo máximo é 35, pois o corte $\langle \{S, A, B, C, D\}, \{T\} \rangle$ tem capacidade 35.

Nota 5.3.3: Esta ideia é a que foi apresentada anteriormente, mas será esta ideia de rede residual que fará parte do algoritmo.

Caminho que aumenta o fluxo f

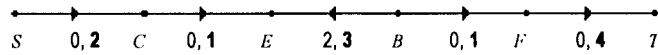
Definição 5.3.4: Um **pseudo-caminho** numa rede N é uma sequência alternada:

$$\langle s = v_0, e_1, v_1, \dots, v_{k-1}, e_k, v_k = t \rangle$$

de vértices e arcos que formam um caminho $-st$ no grafo suporte de N .

Dado um pseudo-caminho $-st$, $Q = \langle s = v_0, e_1, v_1, \dots, v_{k-1}, e_k, v_k = t \rangle$, o arco e_i é um **arco dirigido para a frente** se é dirigido do vértice v_{i-1} para v_i e o arco e_i é um **arco dirigido para a trás** se é dirigido do vértice v_i para v_{i-1} , para qualquer $i \in \{1, \dots, k\}$.

Exemplo 5.3.5: O diagrama seguinte é um pseudo-caminho $-st$ que contém 4 arcos dirigidos para a frente SC, CE, BF e FT e um arco dirigido para a trás EB :



Com a ideia de pseudo-caminho reformulamos a definição de caminho que aumenta o fluxo:

Definição 5.3.6: Seja f um fluxo numa rede N . Um **caminho Q que aumenta o fluxo f** é um pseudo-caminho $-st$ em N , em que o fluxo de cada arco dirigido para a frente pode ser aumentado e o fluxo de cada arco dirigido para trás pode ser diminuído.

Assim, para cada arco e de um caminho Q que aumenta o fluxo f ,

$$\begin{aligned} f(e) < \text{cap}(e), & \text{ se } e \text{ é um arco dirigido para a frente} \\ f(e) > 0, & \text{ se } e \text{ é um arco dirigido para trás.} \end{aligned}$$

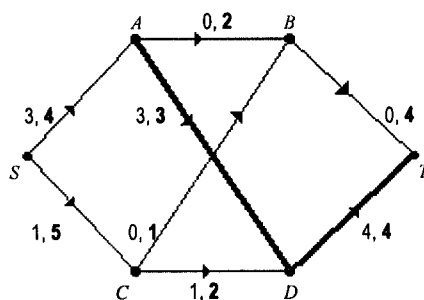
Definição 5.3.7: Para cada arco e de um caminho Q que aumenta o fluxo f , Δ_e é a quantidade dada por:

$$\Delta_e = \begin{cases} \text{cap}(e) - f(e), & \text{se } e \text{ é um arco dirigido para a frente} \\ f(e), & \text{se } e \text{ é um arco dirigido para trás} \end{cases}$$

Para arcos dirigidos para a frente, o valor dado por Δ_e é o maior valor que é possível aumentar o fluxo, enquanto que, para arcos dirigidos para trás, é o maior valor que é possível diminuir o fluxo.

Nota 5.3.8: O maior aumento que é possível fazer num caminho Q que aumenta o fluxo f é $\Delta_Q = \min_{e \in Q} \{\Delta_e\}$, devido à regra de conservação do fluxo, que requer que qualquer mudança no fluxo dos arcos de um caminho que aumenta o fluxo tenha igual "amplitude". Note-se que Δ_Q coincide com Δ_p definido anteriormente, mas Q é um pseudo-caminho.

Exemplo 5.3.9:



O caminho $Q = \langle S, C, D, A, B \rangle$ tem como arcos dirigidos para a frente SC, CD e AB e arco dirigido para trás DA . Assim:

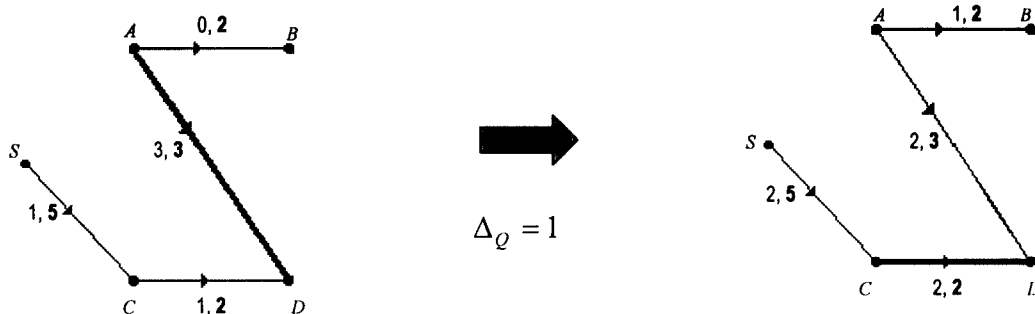
$$\Delta_e(SC) = \text{cap}(SC) - f(SC) = 5 - 1 = 4,$$

$$\Delta_e(CD) = \text{cap}(CD) - f(CD) = 2 - 1 = 1,$$

$$\Delta_e(AD) = f(AD) = 3,$$

$$\Delta_e(AB) = \text{cap}(AB) - f(AB) = 2 - 0 = 2.$$

Neste caso, $\Delta_Q = \min\{4, 1, 3, 2\} = 1$. Deste modo, o fluxo pode ser aumentado em 1 nos arcos SC, CD e AB e diminuído em 1 no arco DA , dando o maior aumento possível ao longo de Q .



A seguinte proposição sumaria como o caminho que aumenta o fluxo f é usado para aumentar o fluxo f na rede N .

Proposição 5.3.10: (Aumento do fluxo) Sejam f um fluxo numa rede N e Q um caminho que aumenta o fluxo f com o mínimo afrouxamento Δ_Q nos seus arcos. O fluxo aumentado dado por:

$$\hat{f}(e) = \begin{cases} f(e) + \Delta_Q, & \text{se } e \text{ é um arco dirigido para a frente e } e \in Q \\ f(e) - \Delta_Q, & \text{se } e \text{ é um arco dirigido para trás e } e \in Q \\ f(e), & \text{caso contrário e } e \notin Q \end{cases}$$

é um fluxo plausível na rede N e $\text{val}(\hat{f}) = \text{val}(f) + \Delta_Q$.

Demonstração:

Pelas definições de $\Delta_Q = \min_{e \in Q} \{\Delta_e\}$ e de Δ_e , verifica-se que os limites, superior e inferior de $\hat{f}$, são:

- se e é um arco dirigido para a frente:

$$\hat{f}(e) = f(e) + \min_{e \in Q} \{cap(e) - f(e)\} \leq f(e) + \min_{e \in Q} \{cap(e)\} \leq cap(e),$$

$\uparrow$
 $\min_{e \in Q} \{f(e)\} \geq 0$

$\uparrow$
 $fluxo \leq capacidade$

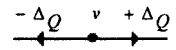
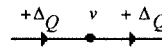
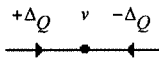
- se e é um arco dirigido para trás:

$$\hat{f}(e) = f(e) - \min_{e \in Q} \{f(e)\} \geq 0;$$

logo, $0 \leq \hat{f}(e) \leq cap(e)$.

Como Q é um caminho que aumenta o fluxo f , obedece à regra de conservação do fluxo, ou seja, o fluxo que sai do vértice inicial de Q é igual ao fluxo que entra no vértice final de Q . Assim, para averiguar se $\hat{f}$ satisfaz a regra da conservação do fluxo, os únicos vértices de Q que precisam de ser verificados são os vértices internos.

Se considerarmos $+$ como o sentido dos arcos dirigidos para a frente e $-$ o sentido dos arcos dirigidos para trás, para um dado vértice v do caminho que aumenta o fluxo Q , os dois arcos de Q que são incidentes em v podem ser ilustrados por uma das quatro formas:



Em cada caso, o fluxo que entra ou sai do vértice v não é alterado, preservando a regra de conservação do fluxo.

Falta ainda mostrar como o fluxo é aumentado por Δ_Q . O único arco incidente na fonte S , no qual o fluxo foi alterado é o primeiro arco, e_1 , do caminho Q que aumenta o fluxo. Se e_1 é um arco dirigido para a frente:

$$\hat{f}(e_1) = f(e_1) + \Delta_Q$$

e se e_1 é um arco dirigido para trás:

$$\hat{f}(e_1) = f(e_1) - \Delta_Q.$$

No outro caso,

$$val(\hat{f}) = \sum_{e \in O(S)} \hat{f}(e) - \overbrace{\sum_{e \in I(S)} \hat{f}(e)}^{=0} = \Delta_Q + val(f)$$

↑

o valor do novo fluxo fica igual à soma do
antigo fluxo com o aumento sofrido

Corolário 5.3.11: Seja f um fluxo numa rede N de valor inteiro na qual as capacidades dos arcos são inteiros. Então o fluxo resultante de sucessivos aumentos de fluxo será um fluxo de valor inteiro.

Demonstração:

Se a capacidade de todos os arcos de uma rede é um número inteiro então o valor do fluxo é também um número inteiro, pois é a soma de inteiros. Como $\Delta_Q = \min_{e \in Q} \{\Delta_e\}$, Δ_Q vai ser um número inteiro, pois resulta da subtração de números inteiros. Logo, o fluxo resultante de sucessivos aumentos de fluxo será um fluxo de valor inteiro. ■

Nota 5.3.12: Até aqui só consideramos valores de fluxos inteiros, mas também podem ser usados valores não inteiros.

Teorema do Fluxo Máximo e Corte Mínimo

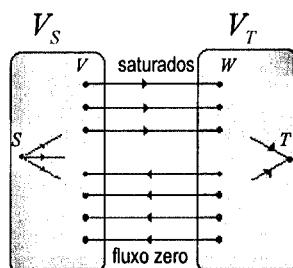
Teorema 5.3.13: (Caracterização do fluxo máximo) Seja f um fluxo numa rede N . f é um fluxo máximo na rede N , se e só se não existe mais nenhum caminho que aumenta o fluxo f em N .

Demonstração:

($\Rightarrow$) Suponha-se que a rede N contém um caminho Q que aumenta o fluxo, então f não pode ser um fluxo máximo, uma vez que o fluxo $\hat{f}$, definido pela Proposição 5.3.10, é o fluxo de maior valor obtido por um caminho Q que aumenta o fluxo.

($\Leftarrow$) Suponha-se que na rede N não há caminhos que aumentam o fluxo. Seja V_S o conjunto de todos os vértices que podem ser ligados a partir de S por um caminho Q que aumenta o fluxo e V_T o conjunto de todos os vértices restantes. Claro que $T \notin V_S$ porque se supôs que não existem caminhos que aumentam o fluxo.

Considere-se o corte $\langle V_S, V_T \rangle$ da rede N e seja e um arco qualquer do corte. Os arcos dirigidos de um vértice V de V_S para um arco W de V_T têm de ser saturados, pois se não o fossem haveria um caminho que aumentava o fluxo de S para W e W deveria ser colocado em V_S em vez de em V_T . Do mesmo modo, um arco dirigido de V_T para V_S tem de ter fluxo 0, pois se não tivesse o fluxo poderia ser diminuído e este arco faria parte de um caminho que aumenta o fluxo, o qual deveria estar em V_S . Esquemmatizando:



Deste modo, pela definição dos conjuntos V_S e V_T o valor do fluxo é dado por:

$$f(e) = \begin{cases} \text{cap}(e) & \text{se } e \in \langle V_S, V_T \rangle \\ 0 & \text{se } e \in \langle V_T, V_S \rangle \end{cases}$$

e, pelo Corolário 5.2.6, um fluxo assim definido é um fluxo máximo. ■

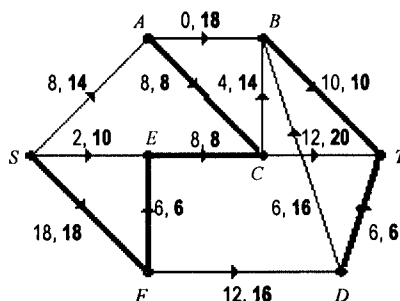
Teorema 5.3.14: (Teorema do Fluxo Máximo e Corte Mínimo) Em qualquer rede, o valor do fluxo máximo é igual à capacidade do corte mínimo.

Demonstração:

O corte construído na demonstração do Teorema 5.3.13 tem capacidade igual ao fluxo total de V_S para V_T , ou seja, igual ao fluxo máximo. ■

Deste teorema decorre que qualquer que seja a ordem pela qual os caminhos que aumentam o fluxo são analisados, o valor do fluxo máximo será sempre o mesmo.

Exemplo 5.3.15: Considere-se a seguinte rede:



Esta rede tem fluxo 28 mas este não é o fluxo máximo. O caminho $Q = \langle S, A, B, C, T \rangle$ tem como arcos não saturados dirigidos para a frente SA , AB e CT e arco dirigido para trás BC . Assim:

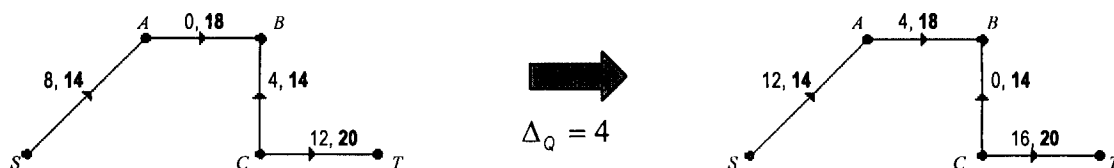
$$\Delta_e(SA) = \text{cap}(SA) - f(SA) = 14 - 8 = 6,$$

$$\Delta_e(AB) = \text{cap}(AB) - f(AB) = 18 - 0 = 18,$$

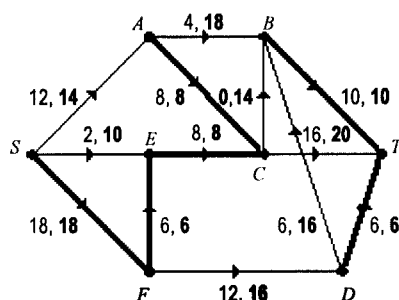
$$\Delta_e(BC) = f(BC) = 12,$$

$$\Delta_e(CT) = \text{cap}(CT) - f(CT) = 20 - 12 = 8.$$

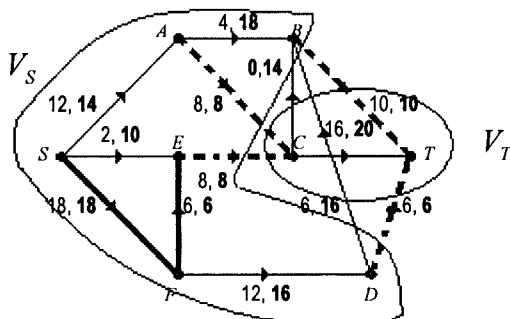
Neste caso, $\Delta_Q = \min\{6, 18, 4, 8\} = 4$. Deste modo, o fluxo pode ser aumentado em 4 nos arcos SA , AB e CT e diminuído em 4 no arco BC , dando o maior aumento possível ao longo de Q .



A rede fica com fluxo 32:



Usando o corte $\langle \{S, A, B, D, E, F\}, \{C, T\} \rangle$, obtém-se um corte de capacidade $8 + 8 + 10 + 6 = 32$:



Pelo Teorema do Fluxo Máximo e Corte Mínimo 32 é o fluxo máximo e o corte $\langle \{S, A, B, D, E, F\}, \{C, T\} \rangle$ é o corte mínimo.

5.4. Funções do tempo de execução de um algoritmo, a sua ordem e hierarquia

Alguns algoritmos criados para resolver o mesmo problema diferem muitas vezes de forma drástica na sua eficiência. Essas diferenças podem ser muito mais significativas do que as diferenças relativas ao *hardware* e ao *software*. Assim, para sabermos se um algoritmo é eficiente ou não, devemos primeiro analisá-lo. Analisar um algoritmo é prever os recursos que o algoritmo precisará. Em geral, fazendo uma análise aos diversos algoritmos candidatos para a resolução de um problema, pode-se identificar facilmente um algoritmo mais eficiente. Essa análise pode indicar mais de um candidato viável, mas vários algoritmos de qualidade inferior serão rejeitados nesse processo.

Para calcular a eficiência de um algoritmo temos de analisar cada uma das suas instruções e devemos ter em atenção que umas instruções demoram mais tempo que outras. Geralmente, o tempo de execução de um algoritmo pode crescer com o tamanho do *input* (entrada), fazendo com que muitas vezes se descreva o tempo de duração de um algoritmo como uma função do tamanho das suas entradas.

O **tamanho do input** depende do problema que está a ser estudado. Por exemplo, num problema de ordenação, o tamanho pode ser dado pelo número de elementos a ordenar, enquanto que num problema de encontrar o fluxo máximo de uma rede, o tamanho pode ser dado até por duas entradas, uma que representa o número de vértices e outra o número de arestas.

O **tempo de execução** (também designado por complexidade de tempo) de um algoritmo é o número de operações executadas. Consideramos que a instrução que demora mais tempo a ser executada é a que compara dois elementos do *input* para ver quando é que eles são iguais e dizemos que uma comparação demora uma unidade de tempo.

Deste modo devemos estudar sempre o tempo de execução do pior dos casos, ou seja, o tempo de execução mais longo para qualquer entrada de tamanho n . Esse estudo é necessário porque o tempo de execução do pior caso de um algoritmo é um limite superior para o tempo de execução de qualquer entrada e conhecê-lo dá-nos a garantia de o algoritmo nunca demorar mais tempo do que este e para alguns algoritmos o pior caso ocorre muitas vezes. Por vezes estima-se a complexidade para o "melhor caso" (pouco útil), o "pior caso" (mais útil) e o "caso médio" (igualmente útil).

Definição 5.4.1: A **função do tempo de execução de um algoritmo** é denotada por $T(n)$ e expressa o tempo máximo necessário para processar qualquer *input* de tamanho n , comparando-o com uma unidade de tempo. De modo análogo, define-se uma **função para o tamanho do input**, $S(n)$, que dá o espaço máximo requerido na memória de um computador para armazenar a informação usada por um algoritmo.

É a **ordem de crescimento** do tempo de execução de um algoritmo que interessa estudar para comparar as funções do tempo de execução de um algoritmo. Assim, considera-se apenas o termo inicial, isto é, o termo de grau mais alto de uma fórmula, pois os termos de ordem mais baixa são relativamente insignificantes para grandes valores de n . Também se ignora o coeficiente do termo inicial, uma vez que factores constantes são menos significativos do que a ordem de crescimento na determinação da eficiência computacional para entradas grandes.

A notação usada é a de **big-oh** (O - grande) e permite suprimir detalhes na análise de algoritmos. Esta notação é usada com três objectivos: limitar o erro que é feito ao ignorar os termos menores nas fórmulas matemáticas, limitar o erro feito na análise ao desprezar parte de um programa que contribui de forma mínima para a complexidade total e permitir classificar os algoritmos de acordo com limites superiores no seu tempo de execução.

Em geral, considera-se um algoritmo mais eficiente do que outro se o tempo de execução do seu pior caso apresenta uma ordem de crescimento mais baixa. Deve-se ter em conta que esta avaliação pode ser incorrecta para entradas pequenas.

Quando se observam entradas com tamanhos suficientemente grandes para tornar relevante apenas a ordem de crescimento do tempo de execução, estuda-se a **eficiência assintótica** dos algoritmos, ou seja, analisa-se a maneira como o tempo de execução de um algoritmo aumenta com o tamanho da entrada no pior caso, ou seja, no limite.

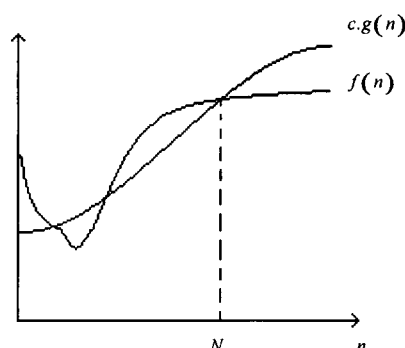
Definição 5.4.2: Sejam $T(n)$ uma função do tempo de execução de um algoritmo e $g(n)$ uma função para a qual existe uma constante positiva c e um número N (suficientemente grande) tal que:

$$T(n) \leq c \cdot g(n) \text{ para todo } n \geq N.$$

Diz-se que $g(n)$ **domina (assintoticamente)** $T(n)$ e $T(n)$ é **dominado (assintoticamente)** por $g(n)$. A função g actua como um limite superior para valores assintóticos da função T .

$O(g(n))$ denota o conjunto de todas as funções que $g(n)$ domina. Se $g(n)$ domina $T(n)$ diz-se que $T(n)$ está em $O(g(n))$, o que significa que $T(n)$ está no conjunto $O(g(n))$.

Graficamente, esta noção traduz-se do seguinte modo:



A partir de um certo N o gráfico de $c \cdot g(n)$ está acima do de $f(n)$.

Muitos algoritmos têm diferentes tipos de ordens de crescimento. Algumas das ordens mais comuns são:

- $O(1)$, muitas instruções são executadas apenas uma vez ou poucas vezes; se tal acontecer, diz-se que o algoritmo tem tempo de execução **constante**;
- $O(n)$, o tempo de execução é **linear**, o que significa que o algoritmo é "bom" quando é necessário processar n dados de entrada ou produzir n dados de saída;
- $O(n^2)$, o tempo de execução é **quadrático**;
- $O(\log_2 n)$, o tempo de execução é **logarítmico**, cresce ligeiramente à medida que n cresce, quando n duplica $\log n$ aumenta mas muito pouco, apenas duplica quando n aumenta para n^2 ;
- $O(n \log_2 n)$, é típico quando se divide um problema em sub-problemas, resolvendo-os separadamente e depois combinando as soluções;
- $O(2^n)$, o tempo de execução é **exponencial**, de pouca utilidade prática.

Exemplo 5.4.3: Mostre-se que:

1. 2^{200} é $O(1)$.

Usando a definição da notação O temos de encontrar uma constante positiva c e um número N tal que:

$$2^{200} \leq c \cdot 1 \text{ para todo } n \geq N.$$

Neste caso, podemos escolher qualquer $c \geq 2^{200}$ e qualquer $N \geq 1$, uma vez que a desigualdade anterior não depende de nenhuma variável n .

2. $5n + 5$ é $O(n)$.

Tendo em conta a definição temos de encontrar uma constante positiva c e um número N tal que:

$$5n + 5 \leq c.n \text{ para todo } n \geq N.$$

Para mostrar que $5n + 5$ é $O(n)$ repare-se que para todo o inteiro $n \geq 1$:

$$5n + 5 \leq 5n + 5n = 10n.$$

Logo, podemos tomar, por exemplo, $c = 10$ e $N = 1$.

3. $30n^3 + 15n \log n + 3$ é $O(n^3)$.

Tendo em conta a definição anterior temos de encontrar uma constante positiva c e um número N tal que:

$$30n^3 + 15n \log n + 3 \leq c.n^3 \text{ para todo } n \geq N.$$

Como $\log n \leq n$, para todo o inteiro $n \geq 1$, tem-se que:

$$\begin{aligned} 30n^3 + 15n \log n + 3 &\leq 30n^3 + 15n.n + 3 \\ &= 30n^3 + 15n^2 + 3 \\ &\leq 30n^3 + 15n^3 + 3n^3 \\ &= 48n^3. \end{aligned}$$

Logo, podemos tomar, por exemplo, $c = 48$ e $N = 1$.

4. $4 \log n + \log(\log n)$ é $O(\log n)$.

Usando a definição da notação O temos de encontrar uma constante positiva c e um número N tal que:

$$4 \log n + \log(\log n) \leq c.\log n \text{ para todo } n \geq N.$$

Como $\log n \leq n$, para todo o inteiro $n \geq 1$, tem-se que:

$$\log(\log n) \leq \log n$$

para todo o inteiro $n \geq 2$ (uma vez que $\log(\log n)$ não está definida para $n = 1$).

Assim,

$$4 \log n + \log(\log n) \leq 4 \log n + \log n = 5 \log n \text{ para todo o inteiro } n \geq 2.$$

Logo, podemos tomar, por exemplo, $c = 5$ e $N = 2$.

Usando o mesmo raciocínio exposto nos exemplos anteriores, podemos generalizar o **tempo de execução de uma função polinomial** $T(n) = pn^m + qn^{m-1} + \dots + r$ ($p > 0$):

- n^k ($k < m$) não domina $T(n)$;
- n^k ($k \geq m$) domina $T(n)$;

- $T(n)$ domina n^k ($k \leq m$);
- $T(n)$ não domina n^k ($k > m$).

Assim, n^m é a única função da forma n^k , para um inteiro não negativo k , que domina e é dominado por $T(n) = pn^m + qn^{m-1} + \dots + r$ ($p > 0$), o que sugere a seguinte definição:

Definição 5.4.4: Se a função $g(n)$ domina a função de tempo de execução $T(n)$ e $T(n)$ também domina $g(n)$, então $T(n)$ tem **ordem** $O(g(n))$ e diz-se que $T(n)$ e $g(n)$ têm a **mesma ordem de grandeza**.

Desta definição decorre que $T(n) = pn^m + qn^{m-1} + \dots + r$ ($p > 0$) tem ordem $O(n^m)$. Geralmente, pode dizer-se que duas funções têm a mesma ordem de grandeza se o comportamento dos seus gráficos é "grosseiramente" o mesmo.

Hierarquia das ordens

As seguintes inclusões chama-se **hierarquia das ordens**:

$$O(1) \subset O(n) \subset O(n^2) \subset \dots \subset O(n^{20}) \subset \dots \subset O(n^k) \subset \dots$$

Para chegar à hierarquia das ordens começa-se com a desigualdade $n \geq 1$ e multipliquem-se ambos os membros por n , obtendo-se $n^2 \geq n$. Continuando este processo de multiplicação de ambos os membros por n , conclui-se que $1 \leq n \leq n^2 \leq \dots \leq n^{20} \leq \dots \leq n^k \leq \dots$ para todo $n \geq 1$. Decorre que qualquer função $T(n)$ dominada por 1 é dominada por $n, n^2, \dots, n^{20}, \dots, n^k, \dots$, uma vez que se $T(n) \leq c \cdot 1$, então $T(n) \leq c \cdot n^k$ para todo $n \geq 1$ e $k \geq 1$. De modo semelhante, se qualquer função $T(n)$ é dominada por n é dominada por $n^2, \dots, n^{20}, \dots, n^k, \dots$, uma vez que, se $T(n) \leq c \cdot n$, então $T(n) \leq c \cdot n^k$ para todo $n \geq 1$ e $k \geq 2$.

Analisando a hierarquia e como esta foi construída por ordem crescente, verifica-se que se a ordem de crescimento de um algoritmo é $O(n^i)$ para um n suficientemente grande, então a eficiência do algoritmo é tanto maior quanto menor for i , ou seja, quanto mais à esquerda estiver a ordem do algoritmo.

Note-se que estas inclusões são estritas, uma vez que não existe igualdade de conjuntos.

Tendo em conta os exemplos dados anteriormente de diferentes tipos de ordens de crescimento, verifica-se que esta hierarquia ainda não está completa, faltando ainda, por exemplo, as ordens logarítmica e exponencial.

Relativamente à ordem logarítmica, a mais usual é a $\log_2 n$. Todas as funções logarítmicas têm ordem $O(\log_2 n)$, uma vez que estas funções apresentam a seguinte propriedade de mudança de base:

$$\log_b n = \frac{\log_a n}{\log_a b}.$$

Para todo o inteiro $n > 2$ tem-se que $1 < \log_2 n < n$, logo:

$$O(1) \subset O(\log_2 n) \subset O(n) \subset O(n^2) \subset \dots \subset O(n^{20}) \subset \dots \subset O(n^k) \subset \dots$$

e $n < n \log_2 n < n^2$, logo:

$$O(1) \subset O(\log_2 n) \subset O(n) \subset O(n \log_2 n) \subset O(n^2) \subset \dots \subset O(n^{20}) \subset \dots \subset O(n^k) \subset \dots$$

À hierarquia das ordens ainda falta acrescentar, de entre outras possíveis, as funções exponenciais; 2^n é dominado por 3^n , 3^n é dominado por 4^n e assim sucessivamente, as quais são dominadas por $n!$, o que se prova do seguinte modo:

- comece-se por mostrar que 2^n é dominado por 3^n :

Repare-se que $\lim_{n \rightarrow \infty} \frac{2^n}{3^n} = \lim_{n \rightarrow \infty} \left(\frac{2}{3}\right)^n = 0$ (basta ver que $\sum \left(\frac{2}{3}\right)^n$ converge). Quer-se mostrar que

$2^n \leq c \cdot 3^n$ para todo $n \geq N$, c uma constante positiva e N um número suficientemente grande. Assim,

$\frac{2^n}{3^n} \leq 1$. Considerando $c = 1$, decorre que: $2^n \leq 1 \cdot 3^n$, ou seja, 2^n é dominado por 3^n ;

- em seguida mostre-se que as funções 2^n , 3^n , 4^n e assim sucessivamente são dominadas por $n!$, ou seja, $a^n \leq c \cdot n!$ para todo $n \geq N$, para $a > 0$, c uma constante positiva e um número N (suficientemente grande).

Comece-se por notar que $\lim_{n \rightarrow \infty} \frac{a^n}{n!} = 0$ para todo número real $a > 0$, uma vez que $\sum \frac{a^n}{n!}$ converge, logo

$\lim_{n \rightarrow \infty} \frac{a^n}{n!} = 0$, o que significa que o factorial de n cresce mais depressa do que a^n . Seja $n_0 \in \mathbb{N}$ tal que

$\frac{n_0}{a} > 2$. Escreva-se $k = \frac{n_0!}{a^{n_0}}$. Para todo $n > n_0$, tem-se que: $\frac{n!}{a^n} = k \cdot \frac{n_0+1}{a} \cdot \frac{n_0+2}{a} \cdot \dots \cdot \frac{n}{a} > k \cdot 2^{n-n_0}$. De

onde, $\lim_{n \rightarrow \infty} \frac{n!}{a^n} = \infty$, ou seja, $\lim_{n \rightarrow \infty} \frac{a^n}{n!} = 0$.

Deste modo, $\exists N \in \mathbb{N} : n \geq N$ se tem que $\frac{a^n}{n!} \leq 1$. Considerando $c = 1$, vem que: $a^n \leq 1 \cdot n!$, ou seja, as

funções a^n são dominadas por $n!$. ■

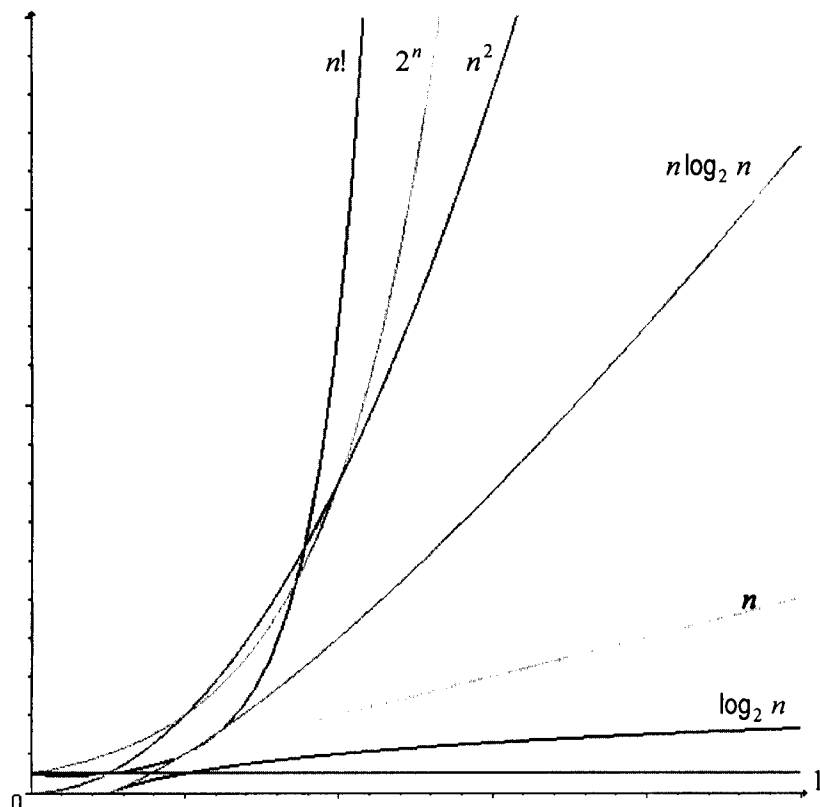
2^n não domina 3^n , 3^n não domina 4^n e assim sucessivamente, logo as inclusões dos conjuntos das ordens são:

$$O(2^n) \subset O(3^n) \subset O(4^n) \subset \dots \subset O(k^n) \subset \dots \subset O(n!).$$

Assim, acrescentando estas hierarquias às anteriores obtém-se:

$$O(1) \subset O(\log_2 n) \subset O(n) \subset O(n \log_2 n) \subset O(n^2) \subset \dots \subset O(n^k) \subset \dots \subset O(2^n) \subset \dots \subset O(k^n) \subset \dots \subset O(n!).$$

Graficamente:



Analisando os gráficos verifica-se que os algoritmos mais rápidos são os que têm tempo de execução de ordem polinomial e os mais lentos são os que têm tempo de execução de ordem exponencial.

5.5. Algoritmo do fluxo máximo e corte mínimo

5.5.1. Algoritmo de Ford - Fulkerson

Um dos métodos para a resolução do problema do fluxo máximo é usar o algoritmo de Ford – Fulkerson, que é o que se fez na secção 5.2. Começa-se o algoritmo com um fluxo inicial de valor 0. Em cada iteração, aumenta-se o valor do fluxo encontrando um caminho que aumenta o fluxo f , ou seja, informalmente encontra-se um caminho ao longo do qual se pode “empurrar” mais fluxo e depois aumentá-lo ao longo desse caminho. Repete-se o processo até não ser possível encontrar mais nenhum caminho que aumenta o fluxo. Aplicando o Teorema do Fluxo Máximo e Corte Mínimo decorre que, no fim deste processo, é produzido um fluxo máximo.

Método de Ford - Fulkerson

Entrada: um fluxo de uma rede $N = (V, E, S, T, cap)$, onde V é o conjunto dos vértices da rede N , E é o conjunto dos arcos de N , S é a fonte, T o destino e cap a capacidade dos arcos

Saída: um fluxo máximo f

Passo 1: **inicia-se** o fluxo f como 0

Passo 2: **enquanto** existir um caminho que aumenta o fluxo f

Passo 3: **aumente-se** o fluxo f ao longo deste caminho

Passo 4: **voltar** a f

O algoritmo seguinte é uma expansão do método anterior. As duas primeiras instruções deste algoritmo iniciam o fluxo f como 0. A instrução seguinte “*enquanto*” encontra repetidamente um caminho que aumenta o fluxo e amplia o fluxo f ao longo desse caminho usando a capacidade residual deste. Quando não existe mais nenhum caminho que aumenta o fluxo f , significa que o fluxo máximo da rede N foi alcançado. Contudo, não há qualquer certeza de que se chegue a esta situação; o que se pode garantir é que a resposta estará correcta se o algoritmo terminar. A função de complexidade de tempo deste algoritmo depende da forma como o caminho que aumenta o fluxo é encontrado. Se este for mal escolhido, o algoritmo poderá não terminar: o valor do fluxo aumentará com ampliações sucessivas, podendo não convergir para o valor do fluxo máximo. Na prática, o problema do fluxo máximo aparece com capacidades inteiras, se as capacidades forem números racionais, pode-se, através de uma transformação apropriada torná-las, inteiras.

Algoritmo 1: Algoritmo de Ford – Fulkerson para (N, S, T) , ou seja, uma rede N com fonte S e destino T :

Entrada: um fluxo de uma rede $N = (V, E, S, T, cap)$

para cada arco e na rede N

faça $f^*(e) = 0$

enquanto existir um caminho que aumenta o fluxo f^* de S para T na rede N

encontre um caminho Q que aumenta o fluxo f^*

 seja $\Delta_Q = \min_{e \in Q} \{\Delta_e\}$. (*)

para cada arco e do caminho Q que aumenta o fluxo

se e é um arco dirigido para a frente

$$f^*(e) = f^*(e) + \Delta_e$$

se não (se e é um arco dirigido para trás)

$$f^*(e) = f^*(e) - \Delta_e$$

voltar o fluxo f^*

Saída: um fluxo máximo f^* da rede N

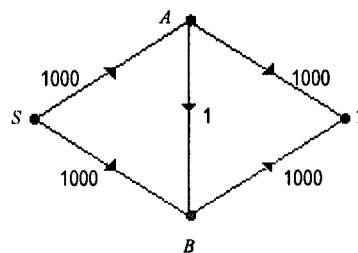
Lema 5.5.1.1: Se as capacidades dos arcos da rede N são inteiras, então o algoritmo de Ford - Fulkerson encontra o fluxo máximo em tempo $O(m|f^*|)$, onde $|f^*|$ é o valor fluxo máximo e $m = |E(N)|$ é o número de arcos da rede N .

Demonstração:

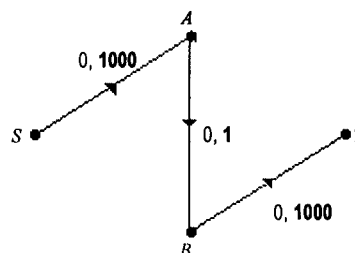
Observe-se que, no algoritmo de Ford - Fulkerson existem, apenas operações de adição, subtração e mínimo por (*). Deste modo, se este encontra um caminho que aumenta o fluxo, então Δ_Q é um número inteiro positivo. Nomeadamente, $\Delta_Q \geq 1$. Assim, pelo Corolário 5.3.11, $|f^*|$ é um número inteiro e a instrução “enquanto” é executada no máximo $|f^*|$ vezes, pois em cada iteração o fluxo é aumentado em pelo menos 1 unidade. Cada caminho que aumenta o fluxo é encontrado em tempo $O(E)$. ■

A eficiência deste algoritmo está em encontrar um bom caminho que aumenta o fluxo. De facto, Ford - Fulkerson verificaram que, se as capacidades dos arcos são números irracionais, então o algoritmo pode não terminar. Mas mesmo quando os fluxos e as capacidades são inteiros, o problema de eficiência continua a existir. Se o fluxo é aumentado apenas numa unidade em cada iteração, o número de aumentos necessários para a maximização será igual à capacidade do corte mínimo. Assim, o algoritmo depende do tamanho das capacidades dos arcos em vez de depender do tamanho da rede.

Por exemplo, analise-se a seguinte rede e aplique-se o algoritmo de Ford - Fulkerson:



1) Comece-se com o caminho que aumenta o fluxo $Q = \langle S, A, B, T \rangle$ e com fluxo inicial 0:

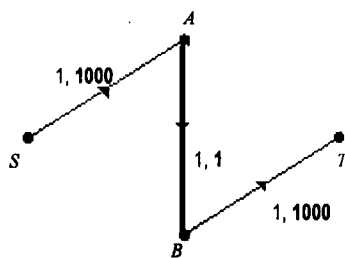


$$\Delta_e(SA) = \text{cap}(SA) - f(SA) = 1000 - 0 = 1000$$

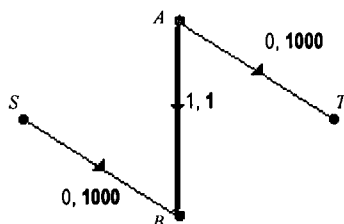
$$\Delta_e(AB) = \text{cap}(AB) - f(AB) = 1 - 0 = 1$$

$$\Delta_e(BT) = \text{cap}(BT) - f(BT) = 1000 - 0 = 1000$$

Neste caso, $\Delta_Q = \min\{1000, 1, 1000\} = 1$. Assim, o fluxo pode ser aumentado em 1 nos arcos SA , AB e BT , dando o maior aumento possível ao longo de Q e ficando o arco AB saturado:



2) Analise-se o caminho que aumenta o fluxo $Q = \langle S, B, A, T \rangle$:

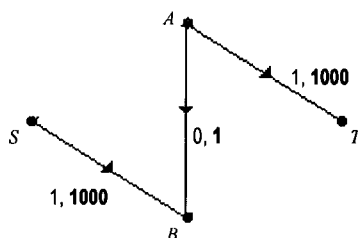


$$\Delta_e(SB) = \text{cap}(SB) - f(SB) = 1000 - 0 = 1000$$

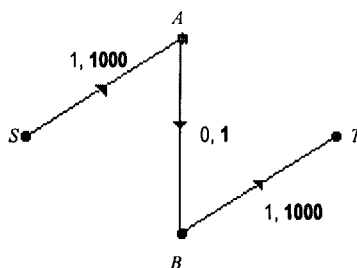
$$\Delta_e(BA) = f(BA) = 1 = 1, \text{ pois o arco } AB \text{ é um arco dirigido para trás}$$

$$\Delta_e(AT) = \text{cap}(AT) - f(AT) = 1000 - 0 = 1000$$

Neste caso, $\Delta_Q = \min\{1000, 1, 1000\} = 1$. Assim, o fluxo pode ser aumentado em 1 nos arcos SB e AT e diminuído em 1 no arco AB , dando o maior aumento possível ao longo de Q :



3) Analise-se novamente o caminho que aumenta o fluxo $Q = \langle S, A, B, T \rangle$:

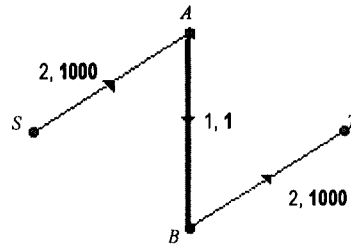


$$\Delta_e(SA) = \text{cap}(SA) - f(SA) = 1000 - 1 = 999$$

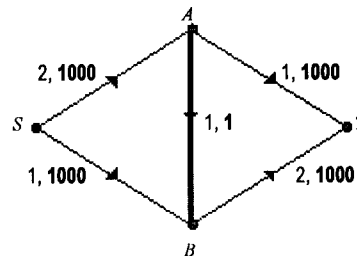
$$\Delta_e(AB) = \text{cap}(AB) - f(AB) = 1 - 0 = 1$$

$$\Delta_e(BT) = \text{cap}(BT) - f(BT) = 1000 - 1 = 999$$

Neste caso, $\Delta_Q = \min\{999, 1, 999\} = 1$. Assim, o fluxo pode ser aumentado em 1 nos arcos SA , AB e BT , dando o maior aumento possível ao longo de Q e ficando o arco AB novamente saturado:



A rede fica:



Continuando este processo e escolhendo alternadamente os caminhos que aumentam o fluxo $SABT$ e $SBAT$, no final executar-se-iam ao todo 2000 ampliações do fluxo, aumentando o valor do fluxo em apenas 1 unidade em cada iteração, fazendo com que o fluxo máximo seja 2000.

Além deste, existe outro algoritmo mais eficiente, o de Edmonds – Karp com complexidade de tempo $O(E^2V)$, mas com final garantido e com um tempo de execução independente do valor do fluxo máximo. Este novo algoritmo usa primeiramente um *algoritmo de procura - breadth first search (BFS)* - para encontrar um caminho que aumenta o fluxo com o menor número de arcos.

Antes de passarmos ao novo algoritmo do fluxo máximo, vejamos como os algoritmos podem ser armazenados (através de pilhas e filas) e em que consta o algoritmo de procura *breadth first search (BFS)*.

5.5.2. Armazenamento de algoritmos: pilhas e filas

As pilhas e filas podem ser vistas como conjuntos dinâmicos, pois, quando manipuladas por algoritmos, podem aumentar, diminuir ou sofrer alterações ao longo do tempo. Tanto as pilhas como as filas são listas lineares, ou seja, são estruturas de dados dinâmicas nas quais os seus elementos estão organizados de maneira sequencial e o elemento a ser removido pela operação *DELETE* (que corresponde a apagar um elemento) é especificado previamente.

Numa pilha, o elemento a ser eliminado é o que foi inserido mais recentemente, é implementada a regra de **LIFO** (*last-in, first-out*), ou seja, último a entrar, primeiro a sair. Enquanto que, numa fila, o elemento a ser eliminado é sempre o que está no conjunto há mais tempo, é implementada a regra de **FIFO** (*first-in, first-out*), ou seja, primeiro a entrar, primeiro a sair.

Pilhas (“Stacks”)

Numa pilha a operação *INSERT* (que corresponde a inserir um elemento) é designada por *PUSH* (empurrar) e a operação *DELETE* por *POP* (eliminar). Podemos associar esta noção às pilhas de pratos de um restaurante, em que a ordem pela qual os pratos são retirados da pilha é a oposta à ordem pela qual são colocados sobre a pilha e o único prato acessível é o prato do topo.

Para esquematizar uma pilha, usemos uma chave *S* do totoloto, na qual os números são colocados pela sua ordem de saída, *S*: 20 – 35 – 10 – 3 – 48 – 15 ,

6	15
5	48
4	3
3	10
2	35
1	20
	<i>S</i>

A instrução *TOP[S]* dá o elemento que se encontra no topo da pilha, neste caso *TOP[S]* = 15 .

A pilha consiste em elementos desde a ordem 1 até ao *TOP[S]*, ou seja, *S*[1...*TOP[S]*], onde *S*[1] é o elemento inferior da pilha e *S*[*TOP[S]*] é o elemento da parte superior da pilha.

Quando *TOP[S]* = 0 a pilha não contém nenhum elemento, é designada por **pilha vazia**.

Usando o exemplo anterior, apliquemos as operações *PUSH* e *POP* para acrescentar e eliminar elementos à pilha. Por exemplo, acrescentemos os elementos 14 e 19, usando *PUSH*[*S*,14] e *PUSH*[*S*,19], ficando a pilha:

8	19	← <i>TOP[S]</i> = 19
7	14	
6	15	
5	48	
4	3	
3	10	
2	35	
1	20	
	<i>S</i>	

Aplicando a instrução *POP* [*S*], a pilha fica sem o último elemento que tinha:

7	14	$\leftarrow TOP[S] = 14$
6	15	
5	48	
4	3	
3	10	
2	35	
1	20	
	<i>S</i>	

Para eliminar um elemento do meio da pilha não se pode fazer esta operação directamente, pois a operação *POP* só permite eliminar o último elemento da pilha; assim, para tal, temos de eliminar todos os elementos da pilha até chegar ao elemento pretendido, depois eliminar o elemento que se pretende e, por último, voltar a acrescentar os elementos que tinham sido retirados da pilha.

Para saber se a pilha está vazia ou não, usa-se a operação *STACK-EMPTY*, que pode ser implementada da seguinte forma:

se $TOP[S] = 0$

então voltar TRUE

se não voltar FALSE

Esta instrução é importante pois, se a pilha estiver vazia, não se podem aplicar as instruções *PUSH* e *POP*.

Outra operação que se pode fazer nas pilhas é a *DEPTH*[*S*] (profundidade), que corresponde à ordem do último elemento, indicando-nos o número de elementos da pilha. Por exemplo, usando a chave do totoloto, tem-se que $DEPTH[S] = 6$.

As operações básicas de uma pilha podem ser conjugadas, de modo a, por exemplo, substituir o último elemento da pilha, saber qual o terceiro elemento da pilha a contar do topo, etc. Por exemplo, para substituir o último elemento da nossa chave de totoloto pelo 28, ou seja, substituir o número 15 pelo número 28 fazem-se as seguintes operações: $PUSH[28, POP[S]]$ obtendo-se a seguinte pilha:

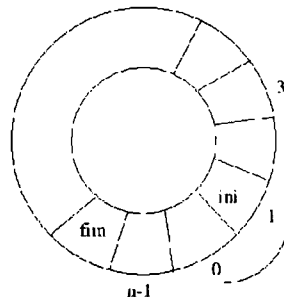
6	28
5	48
4	3
3	10
2	35
1	20
	<i>S</i>

Para sabermos qual o terceiro elemento da pilha a contar do topo, fazem-se as seguintes operações:
 $TOP[POP[POP[S]]] = 3$.

Filas ("Queues")

Numa fila a operação *INSERT* é designada por *ENQUEUE* (enfileirar) e a operação *DELETE* por *DEQUEUE* (desenfileirar). Podemos associar esta noção às filas de qualquer departamento de atendimento, em que a primeira pessoa a chegar é a primeira pessoa a ser atendida. Assim, quando um elemento é colocado na fila, ocupa o último lugar e o primeiro elemento a sair da fila é sempre o que está no início da fila. Deste modo, associado a uma fila, temos dois conceitos: início e fim da fila. Se considerarmos uma sequência de $n - 1$ dados para serem colocados numa fila, por exemplo $Q[1 \dots n]$, o $início[Q]$ representa o início da sequência e $fim[Q]$ representa o seu fim, o local onde o elemento recém-chegado será introduzido. Se $início[Q] = fim[Q]$ diz-se que **a fila está vazia**. Quando se está a formar uma fila tem-se que $início[Q] = fim[Q] = 1$ e quando $início[Q] = fim[Q] + 1$ significa que **a fila está cheia**.

Geometricamente, pode-se pensar numa fila de uma forma circular, no sentido em que à posição 1 segue imediatamente a posição n :



Para enfileirar o elemento x podemos fazê-lo da seguinte forma:

```

ENQUEUE( $Q, x$ )
 $Q[fim[Q]] \leftarrow x$ 
se  $fim[Q] = comprimento[Q]$ 
    então  $fim[Q] \leftarrow 1$ 
    se não  $fim[Q] \leftarrow fim[Q] + 1$ 
    
```

Para retirar um elemento da fila, ou seja, desenfileirar faz-se da seguinte forma:

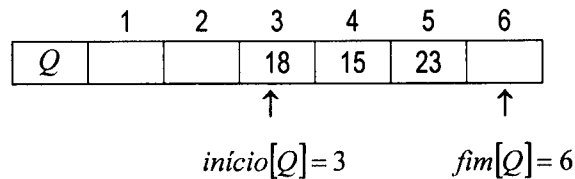
```

DEQUEUE( $Q$ )
 $x \leftarrow Q[início[Q]]$ 
se  $início[Q] = comprimento[Q]$ 
    então  $início[Q] \leftarrow 1$ 
    
```

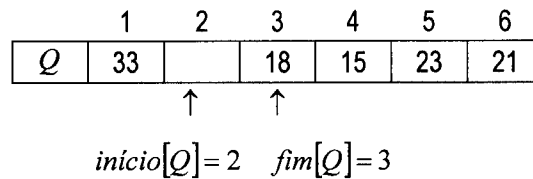
se não $início[Q] \leftarrow início[Q] + 1$

voltar x

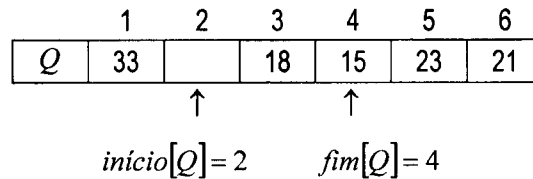
Veja-se um exemplo da aplicação destas operações. Seja Q uma fila com três elementos, 18, 15, 23:



Insiram-se dois elementos na fila, usando $ENQUEUE(Q, 21)$ e $ENQUEUE(Q, 33)$. O resultado é:



Aplique-se agora a instrução $DEQUEUE(Q)$, em que é eliminado o elemento 18 que se encontrava inicialmente no início da fila:



5.5.3. Algoritmo de procura *breadth first* (BFS)

O método de procura *breadth first* pesquisa o grafo, examinando sistematicamente as arestas deste, de modo a alcançar todos os vértices do grafo, procurando uma solução.

Dado um grafo G e um vértice origem S , a busca em largura (BFS) explora sistematicamente as arestas de G até descobrir cada vértice acessível a partir de S . Informalmente, o que este método faz é estudar todos os vértices adjacentes ao vértice onde nos encontramos antes de se passar para outro vértice, ou seja, descobre todos os vértices que estão à distância k (menor número de arestas) a partir de S antes de descobrir quaisquer vértices à distância $k + 1$. O algoritmo calcula a distância desde S até todos os vértices acessíveis, produzindo uma "árvore primeiro na extensão" com raiz S que contém todos os vértices acessíveis. Assim, para um qualquer vértice V acessível a partir de S , o caminho na árvore primeiro em extensão corresponde a um caminho mais curto de S até V em G , ou seja, um caminho que contém um número mínimo de arestas.

Este método usa uma fila para armazenar os vértices que foram descobertos e precisam ser explorados. Este armazenamento é útil para manter a pista da procura BFS. Inserem-se os vértices no fim da fila pela ordem pela qual estes foram visitados e, depois, apagamo-los do início da fila quanto já estiverem servidos.

O algoritmo seguinte executa uma procura BFS usando uma implementação em fila.

Algoritmo 2:

Entrada: um grafo conexo

Passo 1: Escolhe-se um vértice qualquer do grafo, por exemplo, s . Inicia-se a fila com este vértice e escreve-se a etiqueta 0 nele.

Passo 2: Apaga-se s da fila e acrescentam-se a esta os vértices adjacentes a s , etiquetando-os com 1.

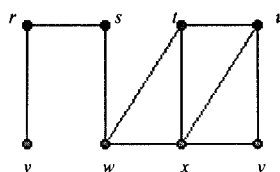
Passo 3: Apaga-se o vértice do início da fila e acrescentam-se a esta os vértices adjacentes ao vértice apagado, etiquetando-os com 2.

Passo 4: O processo continua assim sucessivamente até a fila ficar vazia. Quando tal acontece, encontra-se uma árvore geradora do grafo.

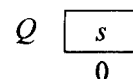
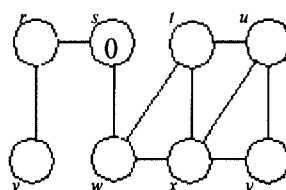
Saída: uma árvore geradora T para a BFS e uma etiquetagem padrão dos vértices de G

Nota 5.5.3.1: Repare-se que os vértices enfileirados no decorrer da procura BFS unem arestas que têm um vértice inicial na árvore e o vértice final que não está na árvore. O processo acaba quando todos os vértices estão etiquetados.

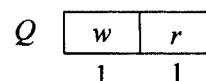
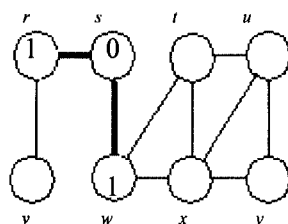
Exemplo 5.5.3.2: Aplique-se o algoritmo BFS ao seguinte grafo:



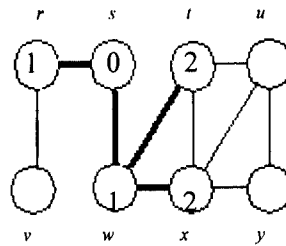
1) Comece-se com o vértice s , etique-se este com 0 e inicie-se a fila.



2) Apague-se s da fila e acrescentem-se a esta os vértices adjacentes a s , ou seja, w e r , etiquetando-os com 1.

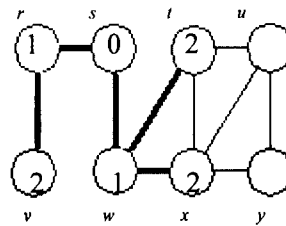


3) Apague-se w da fila e acrescentem-se a esta os vértices adjacentes a w , ou seja, t e x , etiquetando-os com 2.



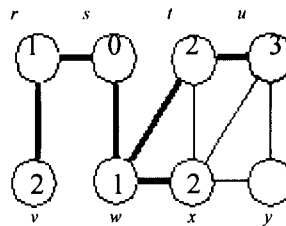
$$Q \begin{array}{|c|c|c|} \hline r & t & x \\ \hline 1 & 2 & 2 \\ \hline \end{array}$$

4) Apague-se r da fila e considerem-se os vértices adjacentes a r : s e v . Como s já esteve na fila, não se acrescenta a esta. Etiqueta-se o vértice v com 2.



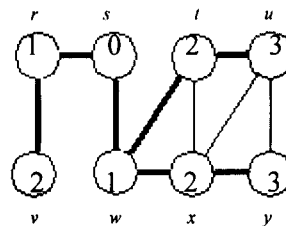
$$Q \begin{array}{|c|c|c|} \hline t & x & v \\ \hline 2 & 2 & 2 \\ \hline \end{array}$$

5) Apague-se t da fila e considerem-se os vértices adjacentes a t : w , x e u . Como os vértices w e x já estiveram ou estão na fila, não se acrescentam a esta. Etiqueta-se o vértice u com 3.



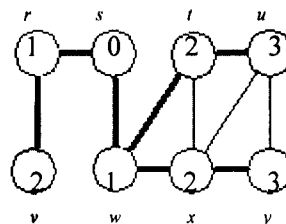
$$Q \begin{array}{|c|c|c|} \hline x & v & u \\ \hline 2 & 2 & 3 \\ \hline \end{array}$$

6) Apague-se x da fila e considerem-se os vértices adjacentes a x : w , t , u e y . Como os vértices w , t e u já estiveram ou estão na fila, não se acrescentam a esta. Etiqueta-se o vértice y com 3.



$$Q \begin{array}{|c|c|c|} \hline v & u & y \\ \hline 2 & 3 & 3 \\ \hline \end{array}$$

7) Apague-se v da fila. Neste caso o vértice adjacente a v é r , que já esteve na fila, logo não se acrescenta a esta.



$$Q \begin{array}{|c|c|} \hline u & y \\ \hline 3 & 3 \\ \hline \end{array}$$

8) Como não existem vértices que ainda não foram visitados, apague-se u da fila.

$$Q \begin{array}{|c|} \hline y \\ \hline 3 \\ \hline \end{array}$$

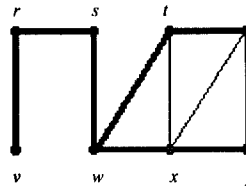
9) Apague-se y da fila, ficando esta vazia.

$$Q \begin{array}{|c|} \hline \phi \\ \hline \end{array}$$

A ordem de visita dos vértices é:

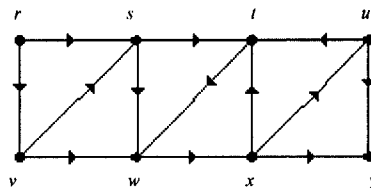
$$s - w - r - t - x - v - u - y.$$

Uma árvore geradora do grafo é:

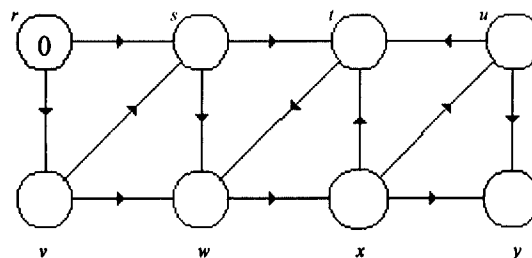


Repare-se que se une o vértice w a x porque x aparece na fila quando w é apagado; une-se o vértice t a u porque u aparece na fila quando t é apagado desta e une-se o vértice x a y , pois o vértice y aparece na fila quando x é apagado desta.

Exemplo 5.5.3.3: Aplique-se o algoritmo *BFS* ao seguinte digrafo:

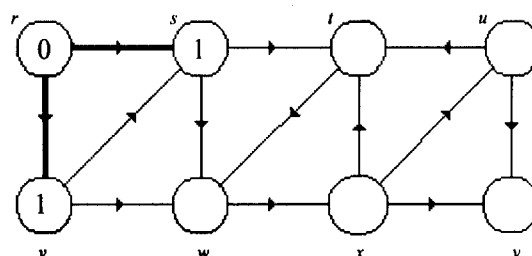


1) Comece-se com o vértice r , etiqüete-se este com 0 e inicie-se a fila.



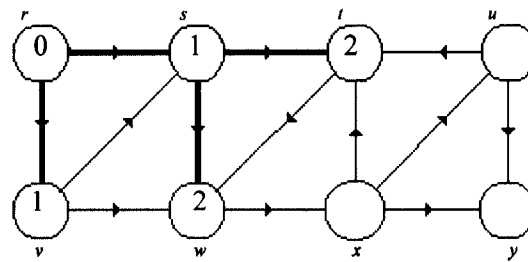
$$Q \begin{array}{|c|} \hline r \\ \hline 0 \\ \hline \end{array}$$

2) Apague-se r da fila e acrescente-se a esta os vértices adjacentes a r , ou seja, s e v , etiqüetando-os com 1.



$$Q \begin{array}{|c|c|} \hline s & v \\ \hline 1 & 1 \\ \hline \end{array}$$

- 3) Apague-se s da fila e acrescente-se a esta os vértices adjacentes a s , ou seja, t e w , etiquetando-os com 2.



$$Q \begin{array}{|c|c|c|} \hline v & t & w \\ \hline 1 & 2 & 2 \\ \hline \end{array}$$

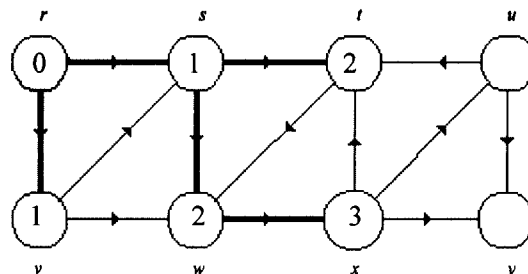
- 4) Apague-se v da fila e analisem-se os vértices adjacentes a v : s e w . Como estes dois últimos vértices já estiveram ou estão na fila, não os acrescentamos a esta.

$$Q \begin{array}{|c|c|} \hline t & w \\ \hline 2 & 2 \\ \hline \end{array}$$

- 5) Apague-se t da fila e analise-se o vértice adjacente a t , que é w . Como este último vértice já está na fila, não o acrescentamos a esta.

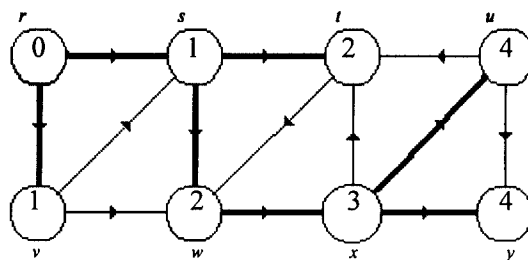
$$Q \begin{array}{|c|} \hline w \\ \hline 2 \\ \hline \end{array}$$

- 6) Apague-se w da fila e acrescente-se a esta o vértice adjacente a w , ou seja, x , etiquetando-o com 3.



$$Q \begin{array}{|c|} \hline x \\ \hline 3 \\ \hline \end{array}$$

- 7) Apague-se x da fila e considerem-se os vértices adjacentes a x : t , y e u . Como t já esteve na fila, não se acrescenta a esta. Etiquetam-se os vértices y e u com 4.



$$Q \begin{array}{|c|c|} \hline y & u \\ \hline 4 & 4 \\ \hline \end{array}$$

- 8) Como não existem vértices que ainda não foram visitados, apague-se y da fila.

$$Q \begin{array}{|c|} \hline u \\ \hline 4 \\ \hline \end{array}$$

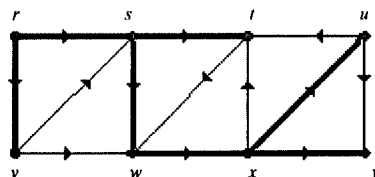
9) Apague-se u da fila.

$$Q \quad \boxed{\phi}$$

A ordem de visita dos vértices é:

$$r - s - v - t - w - x - y - u .$$

Uma árvore geradora do digrafo é:



Repare-se que se une o vértice s a w , pois este aparece na fila quando s é apagado desta. É de notar também que os arcos que vão sendo acrescentados à fila são arcos dirigidos para a frente.

Nota 5.5.3.4: O algoritmo *BFS* tem tempo de execução $O(V)$, sendo V o número de vértices, uma vez que cada vértice é colocado na fila no máximo uma vez e, conseqüentemente, retirado da fila no máximo uma vez. As operações de enfileirar e desenfileirar demoram o tempo $O(1)$ e, deste modo, o tempo de execução deste algoritmo é $O(V)$.

Com esta noção de *BFS* introduzida, podemos agora inserir o algoritmo que encontra um caminho que aumenta o fluxo, no qual o resultado deste algoritmo pode ser este caminho ou um corte mínimo que indica que o fluxo corrente é o fluxo máximo.

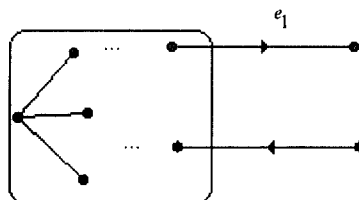
5.5.4. Algoritmo de Edmonds - Karp

Definição 5.5.4.1: Um **arco fronteira** é um arco dirigido de um vértice etiquetado v para um não etiquetado w .

Nota 5.5.4.2: Em qualquer iteração da construção de uma árvore geradora de um caminho que aumenta o fluxo f , seja e um arco fronteira da árvore T com extremidades v e w . O arco e é um **arco usável** se, para o fluxo corrente f , uma das seguintes condições acontece:

- e é dirigido do vértice v para w e $f(e) < \text{cap}(e)$ ou
- e é dirigido do vértice w para v e $f(e) > 0$.

Por exemplo, os arcos e_1 e e_2 são arcos usáveis se $f(e_1) < \text{cap}(e_1)$ e $f(e_2) > 0$:



O algoritmo de Edmonds – Karp usa inicialmente um algoritmo para encontrar um caminho que aumenta o fluxo usando uma procura *BFS*.

Algoritmo 3: Algoritmo para encontrar um caminho que aumenta o fluxo

Entrada: um fluxo de uma rede $N = (V, E, S, T, cap)$

Inicie-se o conjunto de vértices como $V_S := \{S\}$

etiquete-se com 0 o vértice S

Inicie-se a contagem da etiquetagem com $i := 1$

enquanto o conjunto de vértices V_S não contém o destino T (**)

se existem arcos que não foram usados

seja e um arco não usado no qual a extremidade v etiquetada tem a menor etiquetagem possível

seja w a extremidade do arco e .

 o conjunto dos vértices origem $(w) := v$.

escreva-se a etiqueta i no vértice w .

$V_S := V_S \cup \{w\}$

$i := i + 1$

se não

voltar o corte $\langle V_S, V_N - V_S \rangle$.

reconstrua o caminho Q que aumenta o fluxo, seguindo os vértices origem, começando na fonte S .

voltar o caminho Q que aumenta o fluxo.

Saída: um caminho Q que aumenta o fluxo ou um corte mínimo com capacidade $val(f)$

O algoritmo de Edmonds – Karp combina os algoritmos 1 e 3:

Algoritmo 4: Algoritmo de Edmonds – Karp (N, S, T)

Entrada: um fluxo de uma rede $N = (V, E, S, T, cap)$

Iniciação:

para cada arco e na rede N

faça $f^*(e) = 0$

Aumento do fluxo:

Repita-se

aplicando o Algoritmo 3 para encontrar um caminho Q que aumenta o fluxo f^* de S para T na rede N

seja $\Delta_Q = \min_{e \in Q} \{\Delta_e\}$.

para cada arco e do caminho Q que aumenta o fluxo

se e é um arco dirigido para a frente

$$f^*(e) = f^*(e) + \Delta_e$$

se não (se e é um arco dirigido para trás)

$$f^*(e) = f^*(e) - \Delta_e$$

até que um caminho que aumenta o fluxo f^* não possa ser encontrado na rede N

voltar o fluxo f^*

Saída: um fluxo máximo f^* da rede N

Este algoritmo é melhor que o de Ford – Fulkerson pois na instrução “enquanto” (**) deste último algoritmo, o caminho que aumenta o fluxo é procurado usando um método de procura *breadth first*, ou seja, o caminho que aumenta o fluxo passa a ser o caminho mais curto desde a fonte S até ao destino T na rede residual, onde cada aresta tem distância unitária. A análise deste algoritmo depende das distâncias até aos vértices na rede residual.

Em seguida, introduzem-se noções necessárias para mostrar qual o tempo de execução do algoritmo de Edmonds – Karp.

Definição 5.5.4.3: $\delta(u, v)$ é a distância do caminho mais curto desde os vértices u até v .

Antes de mostrar a eficiência do algoritmo de Edmonds – Karp, vejamos uma propriedade importante das distâncias dos caminhos mais curtos.

Lema 5.5.4.4: Sejam G um grafo e $s \in V$ um vértice qualquer, então para qualquer aresta $uv \in E$,

$$\delta(s, v) \leq \delta(s, u) + 1.$$

Demonstração:

Se se puder alcançar u através de s , então o mesmo ocorre com v , pois u está ligado ao vértice v . Neste caso, o caminho mais curto desde s até v não pode ser mais longo do que o caminho mais curto de s

até u passando pela aresta uv e, deste modo, a desigualdade mantém-se válida. Se não se pode chegar a u através de s , então não existe nenhum caminho desde s até u e a desigualdade mantém-se. ■

Notação 5.5.4.5: Seja $\delta_f(u, v)$ a distância do caminho mais curto desde u até v na rede residual da rede N com fluxo f , na qual cada aresta tem distância unitária.

Lema 5.5.4.6: Se o algoritmo de Edmonds – Karp for executado usando um fluxo f numa rede N , com fonte S e destino T , então para todos os vértices $v \in V_N - \{S, T\}$, a distância do caminho mais curto $v \in V_N - \{S, T\}$ na rede residual aumentará monotonamente com cada aumento do fluxo.

Demonstração:

Suponha-se que, para algum vértice $v \in V_N - \{S, T\}$, existe um aumento de fluxo que faz com que a distância do caminho mais curto entre S e v diminua.

Seja f o fluxo imediatamente antes do primeiro aumento que diminui a distância do caminho mais curto e seja f' o fluxo imediatamente após. Seja v o vértice com $\delta_{f'}(S, v)$ mínimo, no qual a distância foi diminuída pelo aumento de tal modo que: $\delta_{f'}(S, v) < \delta_f(S, v)$. Seja $P = \langle S \rightarrow \dots \rightarrow u \rightarrow v \rangle$ um caminho mais curto de S até v na rede residual de modo que $uv \in E_{f'}$ e

$$\delta_{f'}(S, u) = \delta_{f'}(S, v) - 1. \quad (1)$$

Devido ao modo como v foi escolhido, a etiqueta da distância do vértice u não diminuiu, isto é:

$$\delta_{f'}(S, u) \geq \delta_f(S, u). \quad (2)$$

Tem-se que $uv \notin E_f$, uma vez que se $uv \in E_f$ se teria que:

$$\begin{aligned} \delta_f(S, v) &= \delta_f(S, u) + 1 && \text{(usando o Lema 5.5.4.4)} \\ &\leq \delta_{f'}(S, u) + 1 && \text{(pela desigualdade (2) } \Leftrightarrow \delta_f(S, u) \leq \delta_{f'}(S, u)) \\ &= \delta_{f'}(S, v) && \text{(pela igualdade (1) } \Leftrightarrow \delta_{f'}(S, v) = \delta_{f'}(S, u) + 1) \end{aligned}$$

contrariando a hipótese $\delta_{f'}(S, v) < \delta_f(S, v)$.

Repare-se que $uv \notin E_f$ e $uv \in E_{f'}$, uma vez que o aumento efectuado deve ter aumentado o fluxo de v para u . Este algoritmo aumenta sempre o fluxo ao longo de caminhos mais curtos e, deste modo, o caminho mais curto entre s e u na rede residual tem como último arco uv , assim:

$$\begin{aligned} \delta_f(S, v) &= \delta_f(S, u) - 1 \\ &\leq \delta_{f'}(S, u) - 1 && \text{(pela desigualdade (2) } \Leftrightarrow \delta_f(S, u) \leq \delta_{f'}(S, u)) \\ &= \delta_{f'}(S, v) - 1 - 1 && \text{(pela igualdade (1))} \\ &= \delta_{f'}(S, v) - 2, \end{aligned}$$

contrariando a hipótese de $\delta_{f'}(S, v) < \delta_f(S, v)$. Assim, concluiu-se que a suposição de que tal vértice v existe é incorrecta. ■

Teorema 5.5.4.7: Se o algoritmo de Edmonds – Karp for executado usando um fluxo f numa rede N , com fonte S e destino T , então o número total de aumentos de fluxo executados é no máximo $O(VE)$.

Antes de provar este teorema introduza-se a noção de aresta crítica.

Definição 5.5.4.8: Diz-se que uma **aresta é crítica** numa rede residual se num caminho que aumenta o fluxo P , a capacidade residual de P é igual à capacidade residual da aresta.

Demonstração do Teorema 5.5.4.7:

Depois de se ter aumentado o fluxo ao longo de um caminho que aumenta o fluxo, qualquer aresta crítica no caminho desaparece da rede residual. Além disso, pelo menos uma aresta em qualquer caminho que aumenta o fluxo deve ser crítica. Mostra-se seguidamente que cada uma das $|E|$ arestas pode tornar-se crítica no máximo $\frac{|V|}{2} - 1$ vezes.

Sejam u e v dois vértices ligados por uma aresta de E . Sabendo que os caminhos que aumentam o fluxo são os caminhos mais curtos, quando a aresta uv é crítica pela primeira vez tem-se que:

$$\delta_f(S, v) = \delta_f(S, u) + 1. \quad (3)$$

Uma vez que o fluxo é aumentado, a aresta uv desaparece da rede residual, não aparecendo mais até que o fluxo desde u até v seja diminuído, o que acontecerá se vu aparecer num caminho que aumenta o fluxo. Se f' for o fluxo da rede N quando tal acontece (ou seja, quando vu se torna crítica), então:

$$\delta_{f'}(S, u) = \delta_{f'}(S, v) + 1.$$

Tendo em conta que $\delta_f(S, v) \leq \delta_{f'}(S, v)$ pelo Lema 5.5.4.6 tem-se que:

$$\begin{aligned} \delta_{f'}(S, u) &= \delta_{f'}(S, v) + 1 \\ &\geq \delta_f(S, v) + 1 && \text{(pela desigualdade do Lema 5.5.4.6)} \\ &= \delta_f(S, u) + 1 + 1 && \text{(pela igualdade (3))} \\ &= \delta_f(S, u) + 2. \end{aligned}$$

Consequentemente, a partir do momento em que uv se torna crítica até ao momento em que se torna crítica outra vez, a distância de u a partir da origem aumenta pelo menos em 2. A distância de u , desde a origem, é inicialmente pelo menos 0. Os vértices intermédios num caminho mais curto desde S até u não podem conter S, u ou T , uma vez que uv no caminho crítico implica $u \neq T$. Então, até u se tornar inacessível a partir da origem, se isso acontecer a distância será no máximo $|V| - 2$. Deste modo, a aresta uv

tornar-se-á crítica pelo menos $\frac{|V|-2}{2} = \frac{|V|}{2} - 1$ vezes. Como existem $O(E)$ pares de vértices que podem ter entre eles uma aresta numa rede residual, o número total de arestas críticas durante a execução do algoritmo de Edmonds – Karp será $O(VE)$. Cada caminho que aumenta o fluxo terá pelo menos uma aresta crítica e o teorema decorre. ■

Como cada iteração do algoritmo de Ford – Fulkerson pode ser executada em tempo $O(E)$, quando o caminho que aumenta o fluxo é encontrado por uma procura *BFS*, o tempo de execução do algoritmo de Edmonds – Karp será $O(VE^2)$.

A sequência de acontecimentos considerada no algoritmo de Edmonds – Karp é:

fluxo $f \rightarrow$ rede residual $\rightarrow$ procura *BFS* $\rightarrow$ caminho que aumenta o fluxo $P \rightarrow$ novo fluxo $f' \rightarrow$
 $\rightarrow$ nova rede residual $\rightarrow$ nova procura *BFS* $\rightarrow$ novo caminho que aumenta o fluxo P' .

6. Demonstração do Teorema de Menger

Nesta secção demonstram-se as diversas formas do Teorema de Menger, usando o Teorema do Fluxo Máximo e Corte Mínimo. Comece-se por recordar este último teorema.

Teorema 5.3.14: (Teorema do Fluxo Máximo e Corte Mínimo) Em qualquer rede, o valor do fluxo máximo é igual à capacidade do corte mínimo.

Começa-se por demonstrar o Teorema de Menger para digrafos na forma de arcos.

Teorema 3.3.3: TEOREMA DE MENER para digrafos (forma de arcos) Seja D um digrafo conexo e sejam s e t vértices de D . O número máximo de caminhos - st de arcos disjuntos é igual ao número mínimo de arcos que separam s de t .

Demonstração:

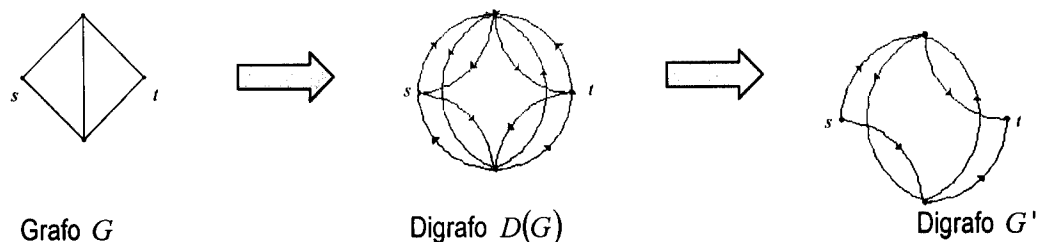
Se existem k caminhos - st de arcos disjuntos, o número mínimo de arcos que podem ser removidos tal que t não fique alcançável através de s é obviamente k . Assuma-se que existe um conjunto de k arcos no qual a sua remoção leva a que não existam caminhos de s para t e que exista um caminho de s para t obtido pela remoção de qualquer conjunto de $k-1$ arcos. Suponha-se que a capacidade de cada arco é 1. Deste modo, a capacidade do corte mínimo é k , logo, pelo Teorema do Fluxo Máximo e Corte Mínimo, o valor do fluxo máximo é também k .

Use-se indução sobre k para mostrar que existem k caminhos de arcos disjuntos da fonte para o destino. Seja I o conjunto de todos os números positivos n tais que, se existe um fluxo de valor n , então existem n caminhos - st de arcos disjuntos. Claro que $1 \in I$, pois tem de existir pelos menos um caminho que una s a t . Suponha-se que $(k-1) \in I$ e que o fluxo tem valor k . Então existe um caminho P de s para t ao longo do qual um fluxo de uma unidade pode ser enviado. Se se removerem todos os arcos pertencentes a P , o valor do fluxo da rede resultante é $(k-1)$. Pela hipótese de indução, existem $(k-1)$ caminhos - st de arcos disjuntos. Juntando esses $(k-1)$ caminhos com o caminho P , forma-se um conjunto de k caminhos de arcos disjuntos. Assim, $k \in I$. Logo, existem k caminhos de arcos disjuntos da fonte para o destino. ■

Teorema 3.2.4: TEOREMA DE MENER para grafos (forma de arestas) Seja G um grafo conexo e sejam s e t vértices de G . O número máximo de caminhos - st de arestas disjuntas é igual ao número mínimo de arestas que separam s de t .

Demonstração:

Seja G um grafo conexo. Construa-se o digrafo associado $D(G)$ substituindo cada aresta do grafo por dois arcos, um em cada direção. Apaguem-se todos os arcos dirigidos que entram em s e todos os arcos dirigidos que saem de t . O resultado será o digrafo G' :



Existe uma correspondência entre o conjunto dos caminhos dirigidos de s para t em G' e o conjunto dos caminhos s e t em G . Deste modo, o teorema decorre como consequência do Teorema de Menger para digrafos (forma de arcos), uma vez que:

$$\begin{array}{l} \text{o número máximo de caminhos } - st \text{ de} \\ \text{arcos disjuntos em } G' \end{array} = \begin{array}{l} \text{o número mínimo de arcos que separam } s \text{ de} \\ t \text{ em } G', \end{array}$$

e então:

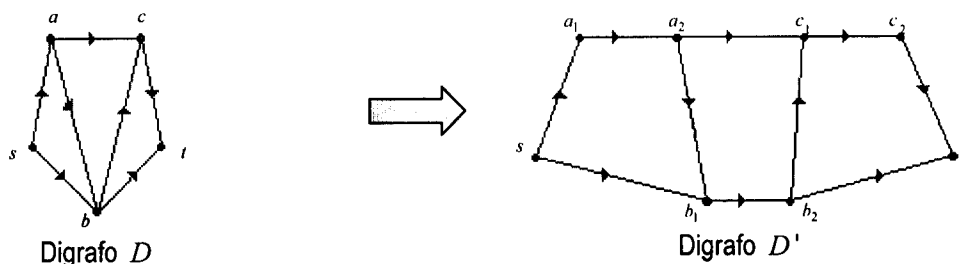
$$\begin{array}{l} \text{o número máximo de caminhos } - st \text{ de} \\ \text{arestas disjuntas em } G \end{array} = \begin{array}{l} \text{o número mínimo de arestas que separam } s \\ \text{de } t \text{ em } G, \end{array}$$

como pretendido. ■

Teorema 3.3.7: TEOREMA DE Menger para digrafos (forma de vértices) Seja D um digrafo conexo e sejam s e t vértices não adjacentes de D . O número máximo de caminhos - st de vértices disjuntos é igual ao número mínimo de vértices que separam s de t .

Demonstração:

Seja D um digrafo conexo. Construa-se um outro digrafo associado D' , substituindo cada vértice v de D (diferentes de s e t) por dois vértices v_1 e v_2 unidos por um arco (repare-se que fazer esta transformação é o mesmo que afirmar que a rede tem restrições de capacidades nos vértices). Esquematizando:



Todos os arcos dirigidos em D que entram num vértice v transformam-se em arcos dirigidos em D' que entram num vértice v_1 e todos os arcos dirigidos de D que saem de v transformam-se em arcos dirigidos de D' que saem de um vértice v_2 .

Não é difícil ver que dois ou mais caminhos $-st$ em D são vértices disjuntos, se e só se os seus correspondentes caminhos $-st$ em D' são arcos disjuntos. Aplicando o Teorema de Menger para digrafos na forma de arcos em D' , obtém-se o Teorema de Menger para digrafos na forma de vértices para D . ■

Teorema 3.3.4: TEOREMA DE MENDER para grafos (forma de vértices) Seja G um grafo conexo e sejam s e t vértices não adjacentes de G . O número máximo de caminhos $-st$ de vértices disjuntos é igual ao número mínimo de vértices que separam s de t .

Demonstração:

Seja G um grafo conexo construa-se o digrafo associado $D(G)$, substituindo cada aresta do grafo por dois arcos, um em cada direcção. Apaguem-se todos os arcos dirigidos que entram em s e todos os arcos dirigidos que saem de t . O resultado será o digrafo G' . Existe uma correspondência entre o conjunto dos caminhos dirigidos de s para t em G' e o conjunto dos caminhos s e t em G . Deste modo, o teorema decorre como consequência do Teorema de Menger para digrafos (forma de vértices), analogamente ao que foi visto anteriormente para o Teorema de Menger para grafos (forma de arestas) que resulta do Teorema de Menger para digrafos (forma de arcos). ■

7. Conclusão

Como vimos existem diversos problemas que podem ser modelados como um problema de fluxo máximo, abstraindo-se dos detalhes. Os métodos de resolução de problemas aqui apresentados têm uma quantidade muito grande de aplicação, podendo ser utilizados em diversas situações, reduzindo o problema inicial ao problema do fluxo máximo.

Bibliografia

- [1] ALDOUS, J. & WILSON, R. (2000). *Graphs and applications: an introductory approach*. London. Springer.
- [2] BONDY, J. & MURTY, U. (1978). *Graph theory with applications*. London. The Macmillian Press LTD.
- [3] BUCKLEY, F. & LEWINTER, M. (2003). *A friendly introduction to graph theory*. New Jersey. Prentice Hall.
- [4] CLARK, J. & HOLTON, D. (1991). *A first look at graph theory*. Singapore. World Scientific.
- [5] CORMEN, T., LEISERSON, C., RIVEST, R. & STEIN, C. (2001). *Introduction to algorithms*. London. McGraw-Hill Book Company.
- [6] GROSS, J. & YELLEN, J. (1999). *Graph theory and its applications*. Boca Raton. CRC Press.
- [7] GROSS, J. & YELLEN, J. (2004). *Handbook of Graph Theory*. Boca Raton. CRC Press.
- [8] JUNGNICHEL, D. (1999). *Graphs, networks and algorithms*. Berlin. Springer.
- [9] THE OPEN UNIVERSITY (2003). *Networks, Vol. 1: Networks Flows*. Milton Keynes. The Open University.
- [10] THE OPEN UNIVERSITY (2005). *Graphs, Vol. 4: Graphs and computing*. Milton Keynes. The Open University.
- [11] WILSON, R. & WATKINS, J. (1990). *Graphs: an introductory approach*. New York. John Wiley.