CAPÍTULO VI

PROBLEMA DE CAMINHO MÍNIMO

Seja um grafo $G = (N, A)$ onde para cada arco $(i, j) \in A$ existe um custo ou comprimento c_{ij} . Determinar o caminho mínimo de um nó inicial $f \in N$ a um nó final $s \in N$, ou a um conjunto de nós finais $S \subset N$, ou a todos os demais nós do grafo. O custo ou comprimento de um caminho é a soma dos custos dos arcos pertencentes a esse caminho.

É um problema de busca de caminho mínimo que pode ser visto como um caso particular do problema de fluxo de custo mínimo, com um único centro de oferta unitária, $f \in N$, e um único centro de demanda unitária, $s \in N$. Esse fluxo unitário percorrerá o caminho mínimo de f a s.

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij} x_{ij} \quad (1)$$

(M1)

$$\sum_{(i,j)\in S(i)} x_{ij} - \sum_{(i,j)\in E(i)} x_{ij} = 1, \quad i \in O = \{f\} \quad (2)$$

$$\sum_{(i,j)\in S(i)} x_{ij} - \sum_{(i,j)\in E(i)} x_{ij} = 0, \quad i \in (N - \{f\} - \{s\}) \quad (3)$$

$$\sum_{(i,j)\in S(i)} x_{ij} - \sum_{(i,j)\in E(i)} x_{ij} = -1, \quad i \in D = \{s\} \quad (4)$$

$$x_{ij} \geq 0, \quad \forall (i, j) \in A \quad (5)$$

As restrições (5) são uma simplificação para $x_{ij} \in \{0, 1\}$, $\forall\ (i, j) \in A$. Ou seja, $x_{ij} = 1$ se o arco (i, j) pertence ao caminho e $x_{ij} = 0$ caso contrário. Essa simplificação é possível uma vez que a matriz de coeficientes das restrições é unimodular.

O modelo acima pode também ser transformado em um modelo de circulação, acrescentando um arco interligando o nó s ao nó f e estabelecendo limites próprios para todos os arcos.

É um modelo linear que pode ser resolvido pelo algoritmo primal Simples em sua forma geral ou particularizado para o modelo de fluxo de custo mínimo. No entanto, é um problema ainda mais simples em que os fluxos assumem valores nulos ou unitários, possibilitando o desenvolvimento de algoritmos especiais, introduzidos de formas variadas por autores diversos (Dreyfus [1969]).

A justificativa teórica dos algoritmos se baseia na teoria da dualidade. Para tal, seja o problema dual de (M1):

(M2)

$$\text{maximizar } u_f - u_s$$

$$u_i - u_j \leq c_{ij}, \quad \forall (i, j) \in A$$

$$u_i \text{ livre}, \quad \forall i \in N$$

Fazendo $u_i' = -u_i$, $\forall i \in N$, resulta

(M3)

$$\text{maximizar } u_s' - u_f'$$

$$u_j' - u_i' \leq c_{ij}, \quad \forall (i, j) \in A$$

$$u'_i \text{ livre}, \quad \forall i \in N$$

Com essa transformação a variável dual u'_i pode representar o custo mínimo associado ao caminho de f a i. Com isso, se $u'_f = 0$ então a função dual $(u'_s - u'_f) = u'_s$ é o custo mínimo de f a s.

Obtida uma solução dual viável e a correspondente solução primal tal que $x_{ij} = 1$ se (i, j) pertence ao caminho de f a s então temos uma solução ótima para o problema. O algoritmo primal Simplex gera a cada iteração soluções duais inviáveis, correspondentes a soluções primais viáveis, mas com um valor da função objetivo dual acima do seu máximo. Este procedimento será repetido aqui. Iniciando com $u'_f = 0$ e $u'_i = \infty$, $\forall\, i \in (N - f)$, para todo arco (f, i) tem-se $u'_i > c_{ij} + u'_f$, violando as restrições duais. O algoritmo procura a cada iteração viabilizar um arco ou restrição dual.

<u>Algoritmo</u> <u>Caminho</u> <u>Mínimo</u> (1)

Seja $L(j)$ - vetor contendo para cada nó j o seu predecessor no caminho

u'_j - custo do caminho do nó inicial ao nó j

{Inicialização}

$L(j) \leftarrow f$; $\quad u'_f \leftarrow 0$;

<u>para</u> $j \in (N - f)$ <u>faça</u>

$L(j) \leftarrow f$;

$u'_j \leftarrow \infty$;

<u>fimpara</u>;

{Processo iterativo}

<u>enquanto</u> $u'_j > u'_i + c_{ij}$ para algum $(i, j) \in A$ <u>faça</u>

$L(j) \leftarrow i$;

$u'_j \leftarrow u'_i + c_{ij}$;

<u>fimenquanto</u>;

<u>se</u> $u'_s = \infty$ <u>então</u> "não existe caminho de f a s";

<u>senão</u> "recuperar o caminho";

<u>fimse</u>;

<u>fimalgoritmo</u>.

A convergência do algoritmo é garantida desde que o grafo não contenha circuitos negativos, ou seja, a soma dos custos nos arcos pertencentes a um circuito é não negativa. Se o circuito negativo existe, a função objetivo pode decrescer infinitamente à medida que o fluxo unitário circula por esse circuito. Caso contrário a função objetivo dual é limitada inferiormente. Como, a cada iteração, uma variável dual u'_j é decrescida, o processo iterativo é finito.

A identificação, a priori, da existência de um circuito de custo negativo pode não ser evidente. Essa condição é facilmente verificada. Seja $c_m = \sum_{c_{ij} < 0} c_{ij}$. Ao atualizar a variável dual u'_j no algoritmo acima, basta acrescentar o teste se $u'_j < c_m$. Se é verdade então existe o circuito de custo negativo. Sua identificação é facilmente obtida pela recuperação do caminho.

Se o grafo não apresenta qualquer circuito (positivo ou negativo) o algoritmo (1) torna-se mais simples, possibilitando o cálculo das variáveis duais de maneira sequencial.

Pelo algoritmo, a viabilidade dual é garantida uma vez que

$$u'_j = \text{mínimo}\ \{u'_i + c_{ij}\} \text{ para todo } j \in (N - \{f\})$$
$$(i, j) \in A$$

Como $u'_f = 0$, então $u'_s = c_{is} + c_{ki} + c_{pk} + \mathrm{K} + c_{fq}$, é a soma dos custos nos arcos pertencentes ao caminho de f a s. Portanto, fazendo $x_{ij} = 1$ para os arcos do caminho e $x_{ij} = 0$ caso contrário, são atendidas as condições de viabilidade primal e dual, complementaridade de folga, $(c_{ij} + u'_i - u'_j)x_{ij} = 0$, e

$$\text{máx } u'_s = \min \sum_{(i,j)\in C} c_{ij}x_{ij}\text{, onde } C \text{ é o caminho de } f \text{ a } s.$$

EXEMPLO 1. Seja o grafo abaixo. Determinar o caminho mínimo entre os nós 1 e 9.

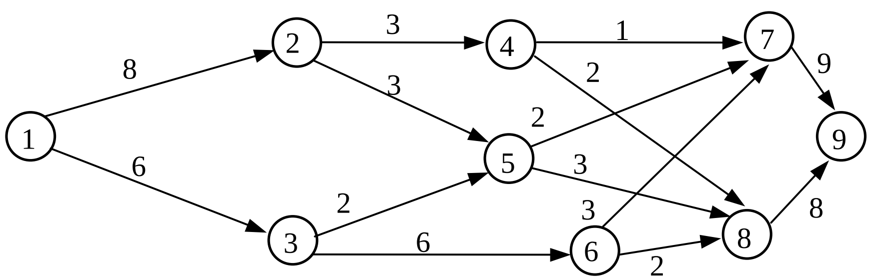

Figura 1. O grafo do exemplo (1).

Aplicando o algoritmo proposto resulta

$u'_1 = 0, u'_2 = u'_3 = \mathrm{K} = u'_9 = \infty, L(1) = L(2) = \mathrm{K} = L(9) = 1$
$u'_2 = \infty > 0 + 8 = 8 = u'_1 + c_{12} \Rightarrow L(2) = 1, u'_2 = 8$
$u'_3 = \infty > 0 + 6 = 6 = u'_3 + c_{13} \Rightarrow L(3) = 1, u'_3 = 6$
$u'_4 = \infty > 8 + 3 = 11 = u'_2 + c_{24} \Rightarrow L(4) = 2, u'_4 = 11$
$u'_5 = \infty > 8 + 3 = 11 = u'_2 + c_{25} \Rightarrow L(5) = 2, u'_5 = 11$
$u'_5 = 11 > 6 + 2 = 8 = u'_3 + c_{35} \Rightarrow L(5) = 3, u'_5 = 8$
$u'_6 = \infty > 6 + 6 = 12 = u'_3 + c_{36} \Rightarrow L(6) = 3, u'_6 = 12$
$u'_7 = \infty > 11 + 1 = 12 = u'_4 + c_{47} \Rightarrow L(7) = 4, u'_7 = 12$
$u'_7 = 12 > 8 + 2 = 10 = u'_5 + c_{57} \Rightarrow L(7) = 5, u'_8 = 10$
$u'_8 = \infty > 11 + 2 = 13 = u'_4 + c_{48} \Rightarrow L(8) = 4, u'_8 = 13$
$u'_8 = 13 > 8 + 3 = 11 = u'_5 + c_{58} \Rightarrow L(8) = 5, u'_8 = 11$
$u'_9 = \infty > 10 + 9 = 19 = u'_7 + c_{79} \Rightarrow L(9) = 7, u'_9 = 19$

O caminho mínimo de 1 a 9 tem um custo 19. Lançando os valores das variáveis duais (ou custos dos caminhos mínimos do nó inicial a cada um dos nós) no grafo da figura (1), e salientando os arcos pertencentes aos caminhos, resulta a figura (2).

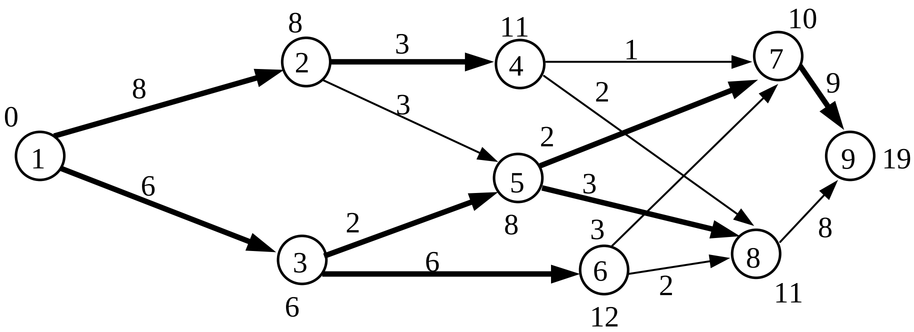

Figura 2. Grafo solução do exemplo (1).

O algoritmo de caminho mínimo (1) atende a redes com custos quaisquer. No entanto, na prática, a maioria dos problemas apresentam custos não negativos ($c_{ij} > 0$). Considerando essa condição, Dijkstra (1959) propôs um procedimento que determina o caminho mínimo de *f* a todos os nós da rede em (n - 1) iterações. Trata-se de adaptar a nova condição ao algoritmo anterior, resolvendo o modelo dual (M3) de uma maneira mais orientada. O resultado ainda é uma árvore geradora do grafo (veja exemplo (1)), ou mais especificamente, uma arborescência com raiz em *f*, contendo um caminho de *f* a todo nó do grafo. Dessa forma, o procedimento é semelhante ao algoritmo de PRIM, para determinação de uma árvore geradora mínima, iniciando em *f*. É um algoritmo guloso para determinação do caminho mínimo. Existem várias formas de apresentação, aqui está uma delas.

Algoritmo de Caminho Mínimo (2) ($c_{ij} > 0$)

Seja P o conjunto dos nós cujo caminho mínimo foi encontrado.

$\overline{P} = N - P$

$L(j)$ - vetor contendo para cada $j \in \overline{P}$ o seu predecessor em P do arco minimal de j.

u'_j - comprimento do caminho do nó inicial ao nó j

{Inicialização}

para $j \in (N - \{f\})$ faça

$L(j) \leftarrow f$;

$$u'_j \leftarrow \begin{cases} c_{ij} & \text{se existe o arco } (f, j) \in A; \\ \infty, & \text{caso contrário;} \end{cases}$$

fimpara;

$P \leftarrow \{f\}$; $\overline{P} \leftarrow (N - \{f\})$; $(s \in \overline{P})$

{Processo iterativo}

repita

selecione $k \in \overline{P}$ tal que $u'_k = \min_{j \in \overline{P}} \{u'_j\}$;

faça $P \leftarrow P \cup \{k\}$;

$\overline{P} \leftarrow \overline{P} - \{k\}$;

para todo $j \in \overline{P}$ faça

se $u'_k + c_{kj} < u'_j$ então faça $\quad u'_j \leftarrow u'_k + c_{kj}$;

$L(j) \leftarrow k$;

fimse;

fimpara;

até $\overline{P} = \varnothing$;

fimalgoritmo.

Como $|\overline{P}| = (n - 1)$ no início do processo, e a cada iteração é retirado um elemento então o algoritmo pára em $(n - 1)$ iterações.

EXEMPLO 2. Seja o grafo do exemplo (1). Determinar o caminho mínimo de 1 a 9 pelo algoritmo (2).

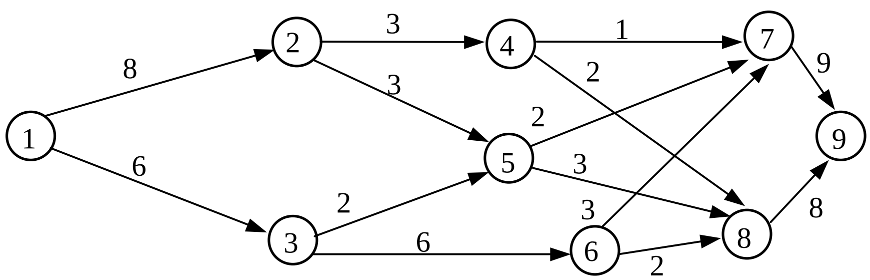

Figura 3. Grafo do exemplo (2).

$P = \{1\}, \overline{P} = \{2, \ldots, 9\}, L(2) = \ldots = L(9) = 1, u'_2 = 8, u'_3 = 6, u'_4 = \ldots = u'_9 = \infty$

$u'_k = \min\{8,6,\infty,\ldots,\infty\} = 6 \Rightarrow k = 3, P = \{1,3\}, \overline{P} = \{2,4,5,\ldots 9\}$

$u'_5 = \infty > 6 + 2 = u'_3 + c_{35} \Rightarrow u'_5 = 8, L(5) = 3$

$u'_5 = \infty > 6 + 6 = u'_3 + c_{36} \Rightarrow u'_6 = 12, L(6) = 3$

$u'_k = \min\{8,\infty,8,12,\infty,\infty,\infty\} = 8 \Rightarrow k = 2, P = \{1,3,2\}, \overline{P} = \{4,5,\ldots 9\}$

$u'_4 = \infty > 8 + 3 = u'_2 + c_{24} \Rightarrow u'_4 = 11, L(4) = 2$

$u'_k = \min\{11,8,12,\infty,\infty,\infty\} = 8 \Rightarrow k = 5, P = \{1,3,2,5\}, \overline{P} = \{4,6,7,8,9\}$

$u'_7 = \infty > 8 + 2 = u'_5 + c_{57} \Rightarrow u'_7 = 10, L(7) = 5$

$u'_8 = \infty > 8 + 3 = u'_5 + c_{58} \Rightarrow u'_8 = 11, L(8) = 5$

$u'_k = \min\{11,12,10,11,\infty\} = 10 \Rightarrow k = 7, P = \{1,3,2,5,7\}, \overline{P} = \{4,6,8,9\}$

$u'_9 = \infty > 10 + 9 = u'_7 + c_{79} \Rightarrow u'_9 = 19, L(9) = 7$

$u'_k = \min\{11,12,11,19\} = 11 \Rightarrow k = 4, P = \{1,3,2,5,7,4\}, \overline{P} = \{6,8,9\}$

$u'_k = \min\{12,11,19\} = 11 \Rightarrow k = 8, P = \{1,3,2,5,7,4,8\}, \overline{P} = \{6,9\}$

$u'_k = \min\{12,19\} = 12 \Rightarrow k = 6, P = \{1,3,2,5,7,4,8,6\}, \overline{P} = \{9\}$

$u'_k = \min\{19\} = 19 \Rightarrow k = 9, P = \{1,3,2,5,7,4,8,6,9\}, \overline{P} = \varnothing$

Os caminhos mínimos e respectivos valores são os mesmos apresentados na figura (2).

Analise a figura (2), resultado da aplicação dos algoritmos (1) e (2) ao grafo da figura (1). Todos os caminhos iniciam no nó 1, nó inicial *f*, e atingem um nó da rede, de forma que, existe em caminho mínimo de *f* a todos os nós da rede, inclusive a *s*. Isto foi possível resolvendo o modelo dual (M3), fazendo $u'_i = -u_i$, para todo $i \in N$. Todo desenvolvimento elaborado anteriormente pode ser refeito para o modelo (M2), porém fazendo $u_s = 0$, e não $u_f = 0$. Nesse caso, u_f será o valor do caminho mínimo de *f* a *s*. Ainda mais, agora todos os caminhos terminam em *s* e iniciam em um nó da rede. Portanto serão determinados todos os caminhos mínimos de todos os nós da rede até o nó final *s*. Por (M3), o caminhamento é feito da frente para o fundo (forward) enquanto, por (M2), é feito do fundo para frente (backward).

Alguns resultados podem ser demonstrados ao aplicar o algoritmo (2):

(i) Quando um nó *j* é lançado em *P* então foi encontrado o caminho mínimo de *f* a *j*.

(ii) Quando um nó *j* é lançado em *P* então todos os nós *i* do grafo tais que $u'_i < u'_j$ já estarão em *P*.

(iii) De (i) e (ii), os caminhos mínimos de *f* a todos os nós da rede são encontrados em ordem crescente dos u'_i.

De (iii), se o objetivo é determinar o caminho mínimo de *f* a *s*, assim que *s* for lançado em *P* o algoritmo pode ser interrompido.

Se o algoritmo (2) não se preocupar em selecionar o elemento u'_k mínimo, duas opções podem ser exploradas. À medida que os nós são rotulados, ou seja, a correspondente variável dual u'_j é atualizada, procura-se selecionar sempre o elemento atualizado mais antigo. Esse processo de busca é denominado busca horizontal. Se selecionar sempre o elemento atualizado por último então resulta a busca em profundidade. No primeiro caso são varridos primeiro os nós ligados a *f*, depois os nós que estão a dois arcos de *f*, três arcos, etc. No segundo caso a busca se aprofunda rapidamente no grafo.