3 Simplex Para Redes

3.1. Noções Iniciais

O algoritmo Simplex para Redes pode ser entendido como uma especialização do método Simplex para aplicação em problemas de programação linear do tipo fluxo de custo mínimo. O Simplex para Redes explora, portanto, as características específicas da rede subjacente ao problema e se mostra extremamente mais eficiente do que o método Simplex, resolvendo problemas, conforme Bazaraa *et al.* (1990), 200 a 300 vezes mais rapidamente do que este. A maior eficiência do Simplex para Redes se dá tanto no menor número de iterações necessárias para se atingir o ótimo, quanto na maior velocidade destas iterações, trata-se, portanto, de um método bastante poderoso na resolução de problemas de fluxo de custo mínimo.

3.2. Solução livre de ciclos e solução como uma árvore geradora

O Simplex para Redes, a exemplo do Simplex padrão, mantém uma solução viável durante a execução do algoritmo que, a cada iteração, "caminha" em direção à solução ótima (salvo os casos de degenerescência que serão vistos mais adiante). Esta solução ótima corresponderá a um vértice da região viável e, no caso de um PFCM, apresentará duas importantes propriedades:

Propriedade 1: Solução Livre de Ciclos

Seja um PFCM dotado de limite inferior em sua região factível, este sempre terá uma solução ótima livre de ciclos.

Prova

É possível classificar os arcos de um grafo em dois tipos de acordo com o seu fluxo. Se um arco (i, j) permite que seja enviada uma unidade de fluxo adicional tanto

no sentido de i para j quanto no sentido inverso, ou seja, se $l_{i,j} < x_{i,j} < u_{i,j}$, este será um arco livre. Caso contrário, se $x_{i,j} = l_{i,j}$ ou se $x_{i,j} = u_{i,j}$ este será um arco restringido.

Diz-se que uma solução é livre de ciclos se esta não contiver ciclos formados apenas por arcos livres. Sendo assim, suponha-se que uma solução ótima contenha um ciclo formado unicamente por arcos livres; pela definição de arco livre isto significa ser possível enviar uma unidade extra de fluxo δ em qualquer direção ao longo do ciclo, o que representará uma oportunidade de se melhorar a solução atual que, por contradição, não poderia ser ótima.

O custo de se enviar uma unidade de fluxo ao longo de um ciclo é a soma dos custos dos arcos que estão na mesma direção do ciclo (aqueles que terão aumento no fluxo) subtraída da soma dos custos dos arcos em direção contrária (aqueles que terão diminuição no fluxo), desta maneira, para o ciclo mostrado na figura 3.1, o custo de se enviar uma unidade adicional de fluxo no sentido horário (figura 3.1a) será de 5, já o custo de se enviar a unidade extra de fluxo no sentido inverso (figura 3.1b) será claramente o mesmo valor com o sinal trocado, ou seja -5.

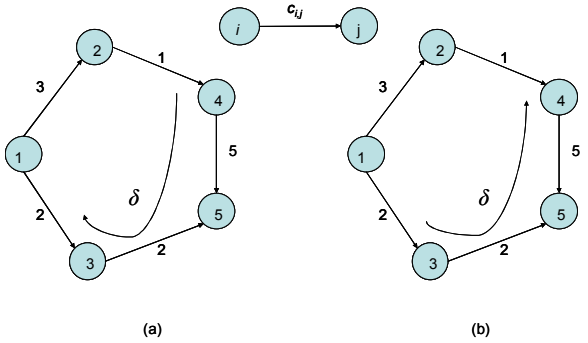


Figura 3.1 - Ciclo composto unicamente por arcos livres.

Desta maneira, se uma solução supostamente ótima não for livre de ciclos, será possível melhorá-la enviando fluxo ao longo do ciclo no sentido em que este torne o custo total da solução menor, até que um de seus arcos não suporte um aumento ou diminuição extra no seu fluxo, tornando-se restringido e desfazendo o ciclo. Ademais,

ainda que o custo do ciclo seja nulo, é possível enviar fluxo em qualquer uma das direções até que um de seus arcos se torne restringido e a solução se torne livre de ciclos. Portanto a conclusão é que há uma solução ótima esta será livre de ciclos.

Propriedade 2: Solução como uma árvore geradora

Seja um problema de fluxo de custo mínimo dotado de limite inferior em sua região factível, este sempre terá como solução uma árvore geradora.

Prova

Uma árvore geradora, por definição, conecta todos os nós do grafo subjacente e não contém ciclos. Já estando provado o fato de ser a solução do PFCM livre de ciclos, suponha-se, portanto, que a solução seja livre de ciclos, mas que não alcance todos os nós do grafo. Neste caso poderiam ser incluídos arcos restringidos à árvore sem gerar ciclos de maneira que esta conectasse todos os nós, obtendo assim, uma árvore geradora como solução.

Uma árvore geradora que contenha arcos restringidos é denominada árvore geradora degenerada, caso contrário esta será uma árvore geradora não degenerada.

3.3. A matriz de incidência nó-arco

Para desenvolver o Simplex para Redes será utilizado, por hora, o desenvolvimento do Simplex padrão uma vez que este último é a base do primeiro; desta maneira nos será possível identificar as semelhanças entre os dois algoritmos, bem como as diferenças que tornam o Simplex para Redes um dos mais bem sucedidos métodos de resolução de PFCM até hoje desenvolvidos.

Inicialmente o Problema de Transbordo será formulado, em seu caso mais genérico (2.1), (2.2) e (2.4), em notação matricial:

$$\left\{ \bar{c}^T \bar{x} \middle| \mathbf{A}\bar{x} = \bar{b} \; e \; 0 \le \bar{l} \le \bar{x} \le \bar{u} \right\} \tag{3.1}$$

Modificando o problema originário, as restrições em (3.1) poderiam ser simplificadas eliminando-se aquelas relativas ao fluxo mínimo nos arcos. Para tal, sempre que um arco apresentar $l_{i,j} > 0$, serão realizadas as seguintes transformações:

$$b_i^* = b_i - l_{i,j} \tag{3.2}$$

$$b_j^* = b_j + l_{i,j} \tag{3.3}$$

$$u_{i,j}^* = u_{i,j} - l_{i,j} \tag{3.4}$$

O procedimento mostrado transforma a rede inicial em uma rede residual com respeito aos fluxos mínimos determinados por (3.1), ou seja, o que está sendo feito neste ponto é equivalente a elevar os fluxos nos arcos aos seus valores mínimos para em seguida otimizar o fluxo restante.

As mudanças nas demandas dos arcos *i* e *j* dadas por (3.2) e (3.3) cuidam para que não seja alterada a condição de equilíbrio entre oferta e demanda.

Ao aplicar as alterações, as restrições em (3.1) ficam:

$$\left\{ \bar{c}^T \bar{x} \middle| \mathbf{A}\bar{x} = \bar{b}^* \; and \; 0 \le \bar{x} \le \bar{u}^* \right\} \tag{3.5}$$

Onde $\bar{u}^* = \bar{u} - \bar{l}$

Posteriormente, tendo sido encontrada a solução ótima para a rede transformada, basta aplicar as transformações inversas para que se tenha a solução ótima para a rede inicial, respeitados os limites inferiores de fluxo nos arcos.

Em (3.5) a matriz **A**, *n* x *m*, associa os *m* arcos (colunas) da rede que a define aos *n* nós (linhas) da mesma, de maneira que cada coluna possui apenas dois elementos não nulos correspondentes aos nós origem (com valor 1) e destino (com valor -1). A matriz **A** é, pelas características apresentadas, chamada de matriz de incidência nó-arco.

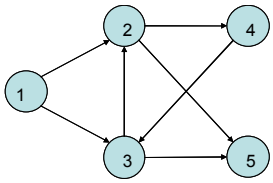


Figura 3.2 – Encontrando a matriz de incidência nó-arco

Como exemplo considere o grafo mostrado na figura 3.2. A matriz de incidência nó-arco referente a este grafo será a seguinte:

$$\mathbf{A} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccccc} (1,2) & (1,3) & (2,4) & (2,5) & (3,2) & (3,5) & (4,3) \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & -1 \\ 0 & 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 \end{bmatrix} \end{array}$$

3.4. Uma revisão de álgebra linear

Antes de prosseguir, é importante que sejam revistos alguns conceitos de álgebra linear que serão utilizados logo em seguida na determinação de uma solução inicial para o problema.

3.4.1.Independência linear

Um conjunto de vetores, $\{\bar{v}_1,...,\bar{v}_n\}$ será linearmente dependente se for possível expressar um dos vetores como uma combinação linear dos demais, ou seja, se for possível encontrar v_i, tal que $\bar{v}_i = \sigma_1\bar{v}_1 + ... + \sigma_{i-1}\bar{v}_{i-1} + \sigma_{i+1}\bar{v}_{i+1} + ... + \sigma_n\bar{v}_n$, sendo os multiplicadores σ_x diferentes de 0. De maneira análoga, um conjunto de vetores $\{\bar{v}_1,...,\bar{v}_n\}$ será linearmente independente caso contrário.

3.4.2. Matriz não singular

Uma matriz **A** é dita não singular se existir uma matriz $\mathbf{A}^{-1}$ tal que $\mathbf{A}^{-1}$ seja a matriz inversa de **A**, ou seja, se existir $\mathbf{A}^{-1}$ tal que $\mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$. De maneira equivalente, a matriz **A** será não singular se o seu determinante for diferente de zero.

3.4.3. Rank de uma matriz

O rank de uma matriz é o número máximo de linhas ou de colunas linearmente independentes desta matriz. O valor correspondente ao número máximo de linhas linearmente independentes é igual ao de colunas linearmente independentes e, portanto, as definições são equivalentes.

3.5. Rank da matriz de incidência nó-arco

Voltando à matriz de incidência nó-arco,. Para encontrar o rank da matriz **A**, pela definição, basta encontrar o número de linhas ou colunas linearmente independentes da matriz **A**.

Cabe, portanto, fazer uma constatação: Em um grafo qualquer, um conjunto de arcos que defina um ciclo, por ser uma corrente fechada, representa um conjunto de vetores linearmente dependentes. A figura 3.3 ajuda a demonstrar este fato.

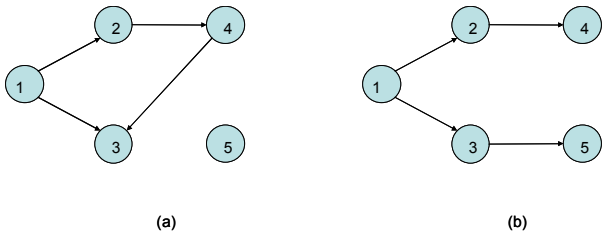


Figura 3.3 - conjuntos de arcos (a), linearmente dependentes, e (b), independentes.

O conjunto definido pela figura 3.3a corresponde à última e às três primeiras colunas da matriz 'A' definida no item 3.3. É possível, por exemplo, dizer que

$(4,3) = (1,3) - (1,2) - (2,4)$ e que, de acordo com a definição apresentada anteriormente, os vetores correspondentes aos arcos em questão são linearmente dependentes.

De maneira análoga é possível demonstrar que não há como definir qualquer dos arcos na figura 3.3b em termos dos demais, sendo assim estes são linearmente independentes.

Esta constatação leva a outra mais interessante. Lembremo-nos da definição de árvore geradora: uma árvore que conecte todos os nós do grafo subjacente e não contenha ciclos. Pode-se, portanto, dizer que, por não conter ciclos, uma árvore geradora pode ser representada por um conjunto de vetores linearmente independentes. Mais ainda, pode-se dizer que uma árvore geradora corresponde a um conjunto de vetores linearmente independentes de cardinalidade máxima.

Conseqüentemente, pode-se dizer que o número de arcos de uma árvore geradora cujo grafo subjacente possua *n* nós será igual ao valor do rank da matriz de incidência nó-arco que define este grafo. E qual seria este valor?

Para chegar à resposta basta imaginar a construção de uma árvore de *n* nós desde o seu primeiro nó. Dado que a árvore é conectada, para cada novo nó acrescentado, a menos do primeiro, deverá ser acrescentado um novo arco não podendo ser acrescentado arco sem um nó correspondente, caso em que seria criado um ciclo, ou seja, desconsiderando-se o primeiro nó, cada nó adicional representa um arco adicional o que, ao final, nos dá o valor de $n-1$ arcos para uma árvore geradora e o mesmo valor para o rank da matriz de incidência nó-arco.

3.6.Variável Artificial

Sendo o número máximo de colunas linearmente independentes da matriz **A** igual a $n-1$ (número máximo de arcos da árvore geradora), não será possível a aplicação do método Simplex, uma vez que uma solução básica será necessariamente composta por *n*

arcos linearmente independentes (são *n* linhas no tableau). A solução neste caso é a criação de uma variável artificial que será a n-ésima na base.

Em outras palavras, para a aplicação do Simplex é necessário que se tenha uma matriz com rank cheio, ou seja, que o rank de **A** seja igual a *n*. Para tal basta que seja acrescentada na matriz uma coluna linearmente independente das demais ou, correspondentemente, que seja acrescentado ao grafo um arco que nunca irá formar um ciclo com os demais, o que só será possível se este possuir uma extremidade em um dos nós do grafo e a outra pendendo no infinito.

O arco em questão recebe o nome de arco-raiz, o nó aonde se encontra uma de suas extremidades será chamado de nó-raiz. Supondo que o arco-raiz se inicia no nó-raiz e se direciona ao infinito tem-se, para o grafo da figura 3.2, uma nova representação dada pelo grafo a seguir.

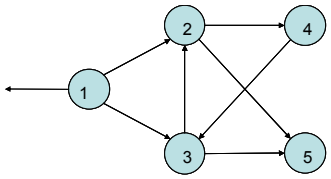


Figura 3.4 - Entrada da Variável Artificial

Qualquer um dos nós do grafo pode ser escolhido com nó-raiz; a única diferença resultante será a maneira como serão representadas as soluções básicas (árvores geradoras) do problema. Como padrão, neste trabalho será utilizado sempre o nó de número "1" para ser a raiz, desta maneira, e de acordo com a figura 3.4, a nova matriz de incidência nó-arco será a seguinte:

$$\begin{array}{r} \\ 1 \\ 2 \\ \mathbf{A} = 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccccc} (1,2) & (1,3) & (2,4) & (2,5) & (3,2) & (3,5) & (4,3) & \textit{raiz} \end{array} \\ \left[\begin{array}{cccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \end{array}\right] \end{array}$$

3.7.Solução inicial

O algoritmo simplex para redes explora a propriedade já abordada anteriormente de que a solução do problema de fluxo de custo mínimo será dada sempre por uma árvore geradora e consiste em, a partir de árvore geradora inicial (solução básica), executar uma série de passos (pivoteamentos) nos quais é mantida uma estrutura de árvore geradora (toda solução básica corresponderá a uma árvore geradora), até que seja encontrada a solução ótima. À árvore correspondente à solução ótima dá-se o nome de árvore geradora ótima.

O primeiro passo necessário à aplicação do simplex para redes é, portanto, determinar uma árvore geradora inicial qualquer para, a partir daí, aplicar o algoritmo.

Uma abordagem prática e de fácil implementação para encontrar esta solução inicial é a criação de um nó artificial com demanda igual a zero, ligado a cada nó original do grafo por arcos de custo proibitivamente altos.

Os arcos que ligam ao nó artificial os nós cujas demandas são maiores ou iguais a zero, ou seja $b_i \leq 0$, terão a origem no nó artificial; já aqueles que ligam ao nó artificial os nós cujas ofertas são maiores que zero, ou seja $b_i > 0$, terão o nó artificial como destino. Os arcos artificiais possuirão ainda capacidade e fluxo inicial iguais ao módulo da demanda do nó original correspondente.

O grafo dado pela figura 3.2 pode ser utilizado para criar um exemplo prático. Para tanto é preciso definir os valores de demanda e oferta dos nós, vetor $\bar{b}$, bem como os custos de cada arco, o vetor $\bar{c}$. Sejam então os valores de oferta/demanda dados por $\bar{b}^T = [8, 0, -2, -1, -5]$ e os de custo dados por $\bar{c}^T = [1, 4, 2, 6, 1, 1, 2]$. A figura 3.5

mostra no grafo os valores de $\bar{b}$ e de $\bar{c}$ (figura 3.5a) e a solução inicial dada pela inclusão de um nó e 5 arcos artificiais (figura 3.5b).

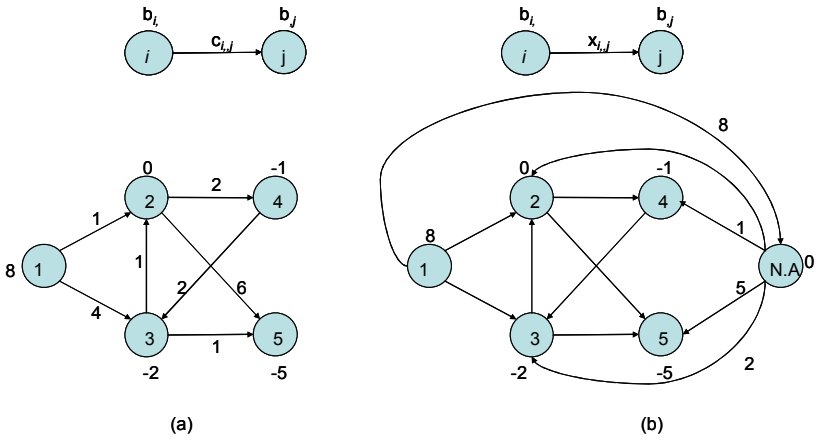


Figura 3.5 - Encontrando uma solução inicial através de um nó artificial.

Para que o algoritmo elimine naturalmente os arcos artificiais da solução será atribuído a eles um custo *M*, representando um valor muito alto, . Alguns autores costumam atribuir a este custo duas vezes o valor da soma dos custos de todos os arcos da rede, Marinho (2003), isto é, $M = 2\sum_{i=1}^{n}\sum_{j=1, j\neq i}^{n} c_{i,j}\eta_{i,j}$, onde $\eta_{i,j}$ será igual à unidade se houver arco entre os nós *i* e *j* e zero caso contrário. O mesmo critério foi utilizado neste trabalho, portanto, para o exemplo ter-se-á que $M = 2(1+4+2+6+1+1+2) = 34$.

A implementação da solução inicial será feita de seguinte maneira:

Função A:

Cria_Solução_Inicial( )

n = número de nós da rede

i = 1

enquanto i < n + 1 fazer:

Se $b_i \leq 0$ *criar arco tal que:*

origem = nó artificial

destino = arco i

fluxo = b_i

custo = *M*

Se $b_i > 0$ *criar arco tal que:*

origem = *arco i*

destino = *nó artificial*

fluxo = b_i

custo = *M*

i = *i* + *1*

fim

Ainda na definição Para o método simplex padrão, a adição de um nó e de *n* arcos artificiais representa a adição de uma linha e *n* colunas ao tableau que, para o exemplo em questão, já na forma canônica, ficará da seguinte maneira:

Para o método simplex padrão, a adição de um nó e de *n* arcos artificiais representa a adição de uma linha e *n* colunas ao tableau que, para o exemplo em questão, já na forma canônica, ficará da seguinte maneira:

	(1,2)	(1,3)	(2,4)	(2,5)	(3,2)	(3,5)	(4,3)	*raiz*	(1,6)	(6,2)	(6,3)	(6,4)	(6,5)	
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
2	1	0	−1	−1	1	0	0	0	0	1	0	0	0	0
3	0	1	0	0	−1	−1	1	0	0	0	1	0	0	2
4	0	0	1	0	0	0	−1	0	0	0	0	1	0	1
5	0	0	0	0	0	1	0	0	0	0	0	0	1	5
6	1	1	0	0	0	0	0	0	1	0	0	0	0	8
−*z*	−67	−64	2	40	1	1	2	0	0	0	0	0	0	−544

O custo total de 544 para a função objetivo é conseqüência do fato de que a demanda e oferta estão sendo transportadas, em sua totalidade, pelos arcos artificiais.

O tableau mostrado corresponde ao tableau inicial para a fase I do método simplex e, na medida em que os arcos artificiais forem saindo da base, estes podem ser eliminados até que se encontre uma solução básica livre destes arcos.

Considerando que o nó "1" foi definido como raiz e omitindo o arco-raiz, a árvore geradora correspondente a esta solução inicial pode ser representada da seguinte maneira:

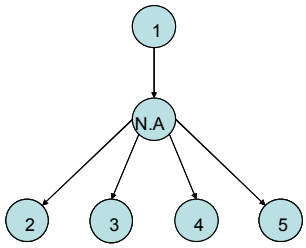


Figura 3.6 - Solução inicial, exemplo.

3.8.Representação da estrutura de árvore geradora

No algoritmo simplex para redes, dada uma árvore geradora que represente uma solução básica, os arcos pertencentes ao grafo subjacente são divididos em três subconjuntos distintos; o primeiro subconjunto, **T**, contém os arcos pertencentes à árvore geradora; o segundo subconjunto, **L**, compreende todos os arcos não pertencentes a **T** que possuam fluxo igual ao seu limite inferior (ou nulo considerando a transformação dada pelas equações (2.13), (2.14) e (2.15)); já o último subconjunto (no caso de o problema ser do tipo capacitado), **U**, possui os arcos não pertencentes a **T** cujos fluxos sejam iguais aos seus limites superiores. Uma solução é dita factível se todos os arcos respeitarem os limites impostos pelas restrições de fluxo.

Sempre que um arco pertencente a **T** não for livre ($x_{ij} = 0$ ou $x_{ij} = u_{ij}$) a árvore geradora correspondente será dita degenerada. O mecanismo que será utilizado para evitar o fenômeno de *cycling*, causado pela ocorrência de degenerescência e que consiste em uma alternância infinita entre bases, será abordado mais adiante,.

3.9.Variáveis Duais e Potenciais dos nós

Para um grafo com n nós e m arcos, o problema dual do problema de minimização de custo será um problema de maximização, onde a função objetivo será composta por n variáveis (variáveis duais) e m equações de restrição, cujos termos

independentes serão formados pelo vetor $\bar{c}$ (vetor com os custos dos *m* arcos do grafo original).

Dada uma solução factível básica do problema primal, para encontrar a solução dual equivalente basta encontrar uma solução para o sistema de equações formado pelas restrições ativas no problema dual (aquelas correspondentes às variáveis básicas do problema primal). Sendo assim tem-se a equação:

$$\bar{w}^T \mathbf{B} = \bar{c}_B^T \tag{3.6}$$

Ou equivalentemente:

$$\bar{w}^T = \bar{c}_B^T \mathbf{B}^{-1} \tag{3.7}$$

Onde **B** representa as colunas da matriz **A** referentes às variáveis básicas no problema primal e o vetor $\bar{c}_B^T$ contém os termos independentes das restrições ativas no problema dual (ou coeficientes tecnológicos associados às variáveis que estão na base do problema primal). As colunas de A referentes às variáveis não básicas são representadas por N sendo, portanto, $\mathbf{A} = [\mathbf{B}|\mathbf{N}]$.

A resolução deste sistema de equações (3.7) se torma mais simples graças à estrutura da matriz **A**, e para demonstrá-la pode ser utilizada como exemplo a base para o grafo mostrado na figura 3.7 dada pela árvore geradora a seguir:

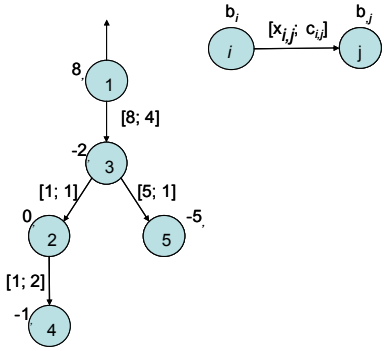


Figura 3.7- Solução factível básica.

O que corresponde, para o cálculo das variáveis duais, ao sistema de equações:

$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 & w_5 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 & 0 \\ -1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 4 & 1 & 1 & 2 & 0 \end{bmatrix} \quad (3.8)$$

Ou ainda:

$$\begin{aligned} w_1 \qquad\quad - w_3 \qquad\quad &= 4 \\ -w_2 \qquad w_3 \qquad\quad &= 1 \\ w_3 \qquad\quad - w_5 \quad &= 1 \\ w_2 \qquad\quad - w_4 \qquad\quad &= 2 \\ w_1 \qquad\quad &= 0 \end{aligned} \quad (3.9)$$

A partir do valor de w_1 dado pela última das equações todas as demais variáveis podem ser encontradas, resultando no vetor $\overline{w}^T = \begin{bmatrix} 0 & -5 & -4 & -7 & -5 \end{bmatrix}$.

Uma constatação muito importante a ser feita com relação à interpretação dada às variáveis duais é a de que, dado um nó *i*, a variável dual correspondente w_i será igual ao custo de se transportar uma unidade de fluxo desde este nó até o nó raiz através dos arcos da árvore geradora.

Pode-se então, definir o vetor $\overline{\pi}^T = \begin{bmatrix} 0 & 5 & 4 & 7 & 5 \end{bmatrix}$ onde $\pi_i = -w_i$ é chamado de potencial do nó *i* e, de maneira simétrica a w_i, pode ser interpretado como o custo de se transportar uma unidade de fluxo desde o nó raiz até este nó. Como conseqüência direta tem-se que, para dois nós adjacentes pertencentes à árvore geradora:

$$\pi_j - \pi_i = c_{ij} \tag{3.10}$$

A expressão (3.10) será utilizada pela demonstrada a seguir, para, recursivamente, computar os potenciais dos nós. Seja o nó 1, para tanto, o nó raiz:

Função B:

Computa_Potenciais(i, j)

Se i =1 fazer:

$\pi_i = 0$

Caso contrário fazer:

$\pi_i = c_{ij} - \pi_j$

Para cada nó k descendente de i através da árvore geradora fazer:

Chama Computa_Potenciais(k, i)

Fim computa_potenciais

Se a função apresentada for iniciada com $i=1$ (nó raiz) e $j=0$, todos os nós do grafo serão visitados (busca em profundidade) e seus potenciais computados de acordo com (3.10).

3.10.Otimalidade

Vale lembrar que a condição de otimalidade do algoritmo simplex é a de haver apenas custos reduzidos positivos. A existência de um custo reduzido negativo significa que a entrada na base da variável relacionada a este representará uma melhoria na solução (salvo o caso de degenerescência que será abordado mais adiante).

Para chegar às condições de otimalidade do simplex para redes serão utilizados também os custos reduzidos que serão dados pela expressão seguinte:

$$\bar{c}_{ij}^{\pi} = \bar{c}^{T} - \bar{c}^{T}\mathbf{B}^{-1}\mathbf{A} \tag{3.11}$$

Que, utilizando-se a definição de potenciais dos nós, pode ser reescrita da seguinte maneira:

$$\bar{c}_{ij}^{\pi} = \bar{c}^{T} + \bar{\pi}^{T}\mathbf{A} \tag{3.12}$$

Onde $\bar{c}_{ij}^{\pi}$ representa o vetor contendo os custos reduzidos associados aos arcos do grafo subjacente.

Desta maneira, para um arco qualquer (i, j) o seu custo reduzido c_{ij}^{π} será dado pela expressão:

$$c_{ij}^{\pi} = c_{ij} + \pi_i - \pi_j \tag{3.13}$$

Para o exemplo dado pelas figuras 3.5 e 3.7, para o qual o vetor $\bar{\pi}^{T}$ já foi calculado, o vetor $\bar{c}_{ij}^{\pi}$ será dado por $\bar{c}_{ij}^{\pi} = [-4,\ 0,\ 0,\ 6,\ 0,\ 0,\ 5]$ onde os valores diferentes de zero correspondem às variáveis não básicas, mais exatamente aos arcos (1,2), $c_{1,2}^{\pi} = -4$, (2,5), $c_{2,5}^{\pi} = 6$ e (4,3), $c_{4,3}^{\pi} = 5$.

Para um arco (i, j) o custo reduzido c_{ij}^{π} pode ser interpretado como o custo de se enviar uma unidade extra de fluxo desde o nó raiz até o nó *i*, enviá-la através do arco (i, j) até o nó *j* para então enviá-la de volta desde o nó *j* até o nó raiz. Intuitivamente já se pode perceber porque os custos reduzidos dos arcos da árvore geradora são iguais a zero já que o caminho de ida até o nó *j* , $c_{ij} + \pi_i$, é o mesmo voltando de *j* para a raiz, π_j. Para enxergar esta propriedade algebricamente basta substituir (3.10) em (3.13).

Já para um arco não básico, isto é, para um arco qualquer do conjunto $\{(i, j) | (i, j) \notin \mathbf{T}\}$, o valor do custo reduzido correspondente será o custo referente ao transporte de uma unidade de fluxo ao longo do ciclo formado pela inclusão do arco (i, j) na árvore geradora, obedecendo-se o sentido do arco (i, j).

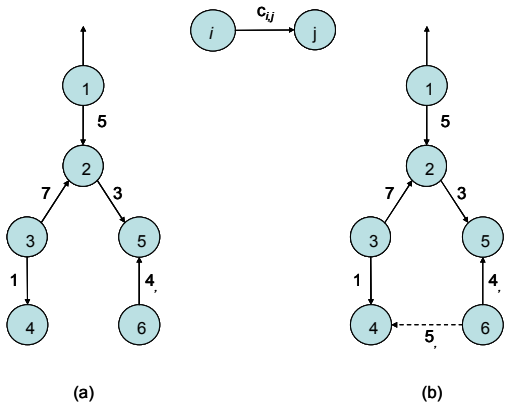


Figura 3.8 - (a) Exemplo de árvore geradora e (b) ciclo formado pela inclusão do arco (6,4).

Na figura 3.8b o arco (6,4) não pertence à solução representada pela árvore geradora da figura 3.8a e, de acordo com a expressão (3.13), o custo reduzido associado a este arco será $c^{\pi}_{6,4} = c_{6,4} + \pi_6 - \pi_4 = 10$, sendo o custo do arco (6,4), $c_{6,4} = 5$, o potencial do nó 6, $\pi_6 = c_{1,2} + c_{2,3} - c_{6,5} = 4$ e o potencial do nó 4, $\pi_4 = c_{1,2} - c_{3,2} + c_{3,4} = -1$. Repare que este valor corresponde, conforme já havia sido exposto anteriormente, ao custo de enviar uma unidade de fluxo ao longo do ciclo formado na árvore geradora pelo arco em questão.

Duas são as possíveis conclusões com relação ao valor encontrado para o custo reduzido do arco (6,4), se $x_{6,4} = 0$, isto é, se $(6,4) \in \mathbf{L}$: o aumento do fluxo no arco e a sua conseqüente entrada na árvore não implicarão em redução do custo total; já se $x_{6,4} = u_{6,4}$, ou seja, se $(6,4) \in \mathbf{U}$: a diminuição do fluxo no arco, que também representará a sua entrada na árvore, significará uma redução no custo total. De maneira análoga, para um arco que apresente custo reduzido negativo, se este pertencer ao conjunto **L,** a sua entrada na árvore geradora significará uma melhoria na solução atual do problema, o mesmo não acontecendo se este pertencer ao conjunto **U**.

Concluí-se, portanto, que para que a solução seja ótima basta que todos os arcos pertencentes a **U** apresentem custos reduzidos negativos ou nulos e que todos os arcos pertencentes a **L** apresentem custos reduzidos positivos ou nulos, ou seja, que a entrada

de qualquer arco não signifique melhoria na solução atual. Se forem incluidos na análise os arcos pertencentes à árvore geradora chegar-se-á às seguintes condições de otimalidade:

$$c_{ij}^{\pi} = 0 \text{ para todo } (i,j) \in \mathbf{T} \quad (3.14)$$

$$c_{ij}^{\pi} \leq 0 \text{ para todo } (i,j) \in \mathbf{U} \quad (3.15)$$

$$c_{ij}^{\pi} \geq 0 \text{ para todo } (i,j) \in \mathbf{L} \quad (3.16)$$

3.11.Índices da árvore

Para facilitar a implementação do simplex para redes são associados aos nós três diferentes índices: predecessor, profundidade e caminho. Em seguida cada um dos três índices será descrito.

3.11.1. Predecessor

Para cada nó de uma árvore geradora existe uma e somente uma corrente ligando este ao nó raiz. Desta maneira ao se enviar uma unidade de fluxo desde o nó raiz até um nó *i*, o predecessor de *i* será o nó imediatamente anterior a *i* neste percurso. De maneira complementar Ahuja e Magnatti (1993) definem como sucessores de *i* aqueles nós que tenham *i* como predecessor; como nós-folhas aqueles nós que não possuam sucessores; e como descendentes de um nó *i*, além do próprio nó *i*, os seus sucessores, os sucessores de seus sucessores e assim por diante. Para o nó raiz o predecessor é sempre definido como zero.

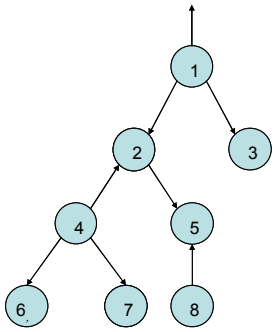
3.11.2. Profundidade

A profundidade de i pode ser entendida como o número de passos dados desde a raiz para se chegar a este nó. A profundidade de um nó será, portanto, a profundidade de seu predecessor acrescida de uma unidade. Já para o nó raiz, a exemplo do índice predecessor, o valor da profundidade será zero.

3.11.3. Caminho

O índice caminho de um nó i será o número do nó que, em uma busca em profundidade, vier na seqüência deste. Repare que, assim como se pode executar uma busca em profundidade de diferentes maneiras (diferentes ordens de visitas aos nós), os índices de caminho podem variar para a mesma árvore. Para o último nó do percurso feito pela busca em profundidade o índice de caminho será igual ao nó raiz.

A figura 3.9a mostra como exemplo uma árvore geradora cujos índices estão relacionados na figura 3.9b.



(a)

i	1	2	3	4	5	6	7	8
predecessor(i)	0	1	1	2	2	4	4	5
profundidade(i)	0	1	1	2	2	3	3	3
caminho(i)	2	4	1	6	8	7	5	3

(b)

Figura 3.9 - Exemplo de árvore geradora e índices associados

Aproveitando a estrutura dada pela solução inicial fica mais fácil computar os valores iniciais destes índices. Repare a solução inicial sempre terá uma estrutura semelhante à mostrada na figura 3.6, com o nó raiz tendo o nó artificial como único descendente que, por sua vez, será predecessor de todos os demais nós.

Função C:

```
Computa_Indices()
    Profundidade do nó_artificial = 1
    Predecessor do nó_artificial = nó_raiz
    Para todos os nós, fazer:
        Se nó = nó_raiz fazer:
            Profundidade do nó = 0
            Predecessor do nó = nulo
            Caminho do nó = nó_artificial
            Caminho do nó_artificial = próximo nó
        Caso contrário fazer:
            Profundidade do nó = 2
            Predecessor do nó = nó_artificial
            Se nó = último nó fazer:
                                Caminho do nó = nó_raiz
            Caso contrário fazer:
                Caminho do nó = próximo nó
Fim
```

No código que foi implementado neste trabalho foi utilizado um quarto índice, o caminho inverso. O caminho inverso de um nó *i* será o nó cujo índice caminho seja *i*, ou seja, é o nó anterior a ele ao se executar a busca em profundidade. Este índice será utilizado para tornar mais rápida a atualização da árvore geradora a cada iteração.

3.12.Funcionamento do algoritmo simplex para redes

Segundo Eppstein (2000), dois são os fatores importantes na eficiência de um algoritmo simplex para redes: o número de pivoteamentos e o tempo por pivoteamento. O pivoteamento é a base do funcionamento do simplex para redes que, explorando o fato de que a solução ótima será necessariamente uma árvore geradora (propriedade 2) consiste em mover a solução de uma árvore para outra melhor repetidas vezes até que as condições de otimalidade sejam verificadas.

Mais detalhadamente, o algoritmo funciona de maneira que para cada árvore sub-ótima encontrada seja selecionado um arco (i, j) que viole as condições de otimalidade dadas por (3.14) e (3.15) a ser acrescentado, através de um pivoteamento, à árvore. O processo é repetido até que todos os arcos, $(i, j) \in \mathbf{U}$ e $(i, j) \in \mathbf{L}$, obedeçam às referidas condições de otimalidade.

3.13.Selecionando o arco entrante.

Suponha-se portanto que, para uma dada solução sub-ótima, existam arcos pertencentes aos conjuntos **U** ou **L** que violem as condições de otimalidade. Neste ponto o problema é definir qual dentre estes arcos será escolhido para integrar a árvore geradora, um processo chamado de precificação, e a resposta parece bem óbvia uma vez que quanto maior for a violação maior tenderá a ser a melhora na solução. Porém, considerando que muitos dos problemas reais de fluxo de custo mínimo chegam a ter milhares ou milhões de arcos, começa a ficar por demais custoso varrer todo o conjunto de arcos à procura daquele com maior violação das condições de otimalidade, ou ainda atualizar os custos reduzidos de todos os nós.

Maros (2003) lembra que, teoricamente, a escolha de qualquer arco dentre os candidatos a entrar na base irá levar à solução em um tempo finito, porém o número de iterações até a solução dependerá profundamente da regra de seleção aplicada. Pensando nisso e a fim de explorar as características particulares de cada problema específico, foram desenvolvidos inúmeros métodos de seleção do arco entrante. Algumas das regras mostradas em Maros (2003) são:

3.13.1.Regra de Dantzig

Esta regra foi proposta por Dantzig (1963) e consiste em, a cada iteração, escolher o arco que apresentar a maior violação, uma vez que a entrada deste representará o maior ganho por unidade de fluxo dentre todos os candidatos, o que resulta em maiores ganhos a cada iteração e, conseqüentemente, em um menor número de iterações. Na

prática, porém, esta regra exige que todos os arcos não básicos sejam verificados a cada iteração, o que torna este critério pouco competitivo.

3.13.2.Primeiro Candidato

Trata-se da regra mais simples e consiste em selecionar o primeiro arco encontrado que apresentar custo reduzido não ótimo sendo, portanto, o oposto da regra de Dantzig. Esta regra não é muito utilizada na prática por gerar um número muito grande de pivoteamentos.

3.13.3.Precifcação Parcial

É um meio termo entre as regras de Dantzig e Primeiro Candidato e refere-se a um tipo de regra que subdivide o conjunto de arcos não básicos e, a cada iteração, escolhe dentro de um destes subgrupos o arco que apresentar a maior violação (Orchard-Hays, 1968). A Precificação Parcial pode ser estática caso os subconjuntos sejam os mesmos durante a execução do algoritmo ou dinâmica caso contrário.

3.13.4.Precificação por Clusters

Em algumas classes de problema pode-se agrupar os arcos de acordo com algum tipo de similaridade ou característica para então aplicar, por exemplo, a precificação múltipla, o que tende a melhorar a eficácia das iterações menores. Conforme dito em Maros (2003), o senso comum e a experiência mostram que, se um arco dentro de um cluster é escolhido e entra na base em uma certa iteração, então será menos provável encontrar bons candidatos neste mesmo cluster na próxima iteração.

3.13.5.Precificação Normalizada

O valor do custo reduzido nos diz o tamanho do ganho que a entrada de determinado arco na base propiciará para cada unidade de fluxo que passar por ele durante o pivoteamento, mas não nos diz nada sobre o total deste fluxo, o que nos deixa apenas com a informação parcial sobre a possível contribuição dada pela entrada de um

determinado arco na base. As regras de precificação normalizada tentam contornar o problema ao normalizar os custos reduzidos multiplicando-os por um fator antes de quaisquer comparações. A regra do limite mais íngrime (Steepest Edge) de Forrest e Goldfarb (1992) e a regra Devex em Harris (1973) são dois exemplos desta classe de regras de precificação.

3.13.6. Precificação Múltipla

Neste trabalho foi utilizada a regra proposta por Ahuja e Magnatti (1993), mais exatamente, a regra de precificação múltipla detalhada a seguir.

A Precificação Múltipla ocorre em duas fases, chamadas de iteração maior e iteração menor. Na iteração maior uma lista de candidatos é construída. A partir daí são executadas as iterações menores que irão varrer toda a lista construída pela iteração maior e eleger o melhor candidato a entrar na base.

Após um número pré-determinado de iterações menores, ou se não houver candidatos restantes na lista devido às atualizações dos custos reduzidos a cada pivoteamento realizado, uma nova iteração maior será realizada partindo do ponto onde a última iteração maior havia parado (percorrendo a lista de arcos de maneira circular). A iteração maior percorre os arcos até atingir o número pré-determinado de candidatos da lista ou até não haver mais candidatos.

Para esta regra tanto o número máximo de candidatos na lista quanto o número de iterações menores podem ser ajustados de maneira a adaptar a estratégia de precificação à realidade de um problema específico. Repare que se não for definido tamanho máximo da lista e o número de iterações menores for apenas um tem-se a regra de Dantzig e ainda, se o tamanho máximo e o tamanho de iterações menores forem ambos unitários tem-se a regra do primeiro candidato.

As iterações maiores serão feitas através da função a seguir:

Função D:

```
Iteração_Maior(arco inicial)
i = 0
otimalidade = verdadeiro
arco = arco inicial
Enquanto i < número máximo de candidatos + 1 fazer
    Se arco não pertence à árvore geradora fazer
        Se arco não satisfizer as condições de otimalidade (3.15) ou (3.16) fazer
            Inclui arco na lista de candidatos
            i = i + 1
            otimalidade = falso
        arco = próximo arco
        se arco = arco inicial fazer
            sair da função
Fim
```

Pode-se perceber que a função buscará candidatos até que seja encontrado o número máximo de candidatos (número definido pelo usuário) ou até que todos os arcos tenham sido testados com relação à otimalidade. Neste último caso, se nenhum candidato tiver sido encontrado a variável booleana, *otimalidade*, terá valor verdadeiro e a solução ótima terá sido encontrada. Em se havendo candidatos serão realizadas iterações menores de maneira em que a cada iteração menor que encontre candidatos a entrar na árvore esta seja atualizada (da maneira que será mostrada mais adiante). Sendo a iteração menor dada simplesmente pela busca, na lista de candidatos gerada pela iteração maior, pelo arco com maior custo reduzido (em módulo).

3.14.Selecionando o arco que sai.

O próximo passo, após ter sido selecionado o arco entrante, é determinar o arco que deixa a árvore geradora. Como já foi repetido algumas vezes, para um problema cujo grafo subjacente contenha n nós, uma árvore geradora deverá conter exatamente n-1 arcos, o que significa que para se manter uma estrutura de árvore geradora a entrada de um nó deve ser acompanhada da saída de outro.

Sendo (i, j) o arco entrante, o primeiro passo para a determinação do arco que irá deixar a árvore geradora (deixar o conjunto **T**) é, portanto, identificar o ciclo formado pela inclusão do arco (i, j) na mesma, chamado de ciclo de pivoteamento, e para isso basta percorrer a rede desde os nós *i* e *j* até o seu primeiro antecessor comum, w, através de seus predecessores, aqueles arcos que fizerem parte do caminho entre *i* e *w* e entre *j* e *w*, bem como o próprio arco (i, j), comporão o ciclo. A função 3 faz a identificação deste ciclo.

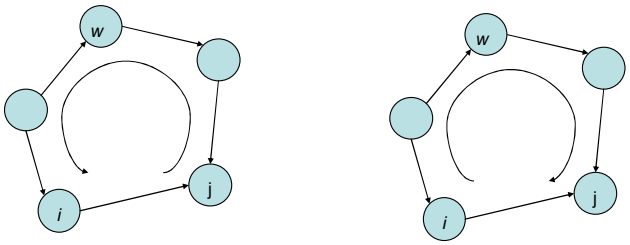
Função E:

```
Identifica_Ciclo(nó i, nó j)
x = i
y = j
Enquanto predecessor(x) <> predecessor(y) fazer:
        Se profundidade(predecessor(x)) > profundidade(predecessor(y))
            x = predecessor(x)
        c.c., se profundidade(predecessor(y))>profundidade(predecessor(x))
            y = predecessor(y)
        c.c, se profundidade(predecessor(y)) = profundidade(predecessor(x))
            x = predecessor(x)
                y = predecessor(y)
Fim identifica_ciclo
```

Ao final da execução da função, w será o predecessor tanto de x quanto de y.

Havendo identificado o ciclo de pivoteamento deve-se determinar qual será o fluxo enviado ao longo do mesmo e, para isso, os arcos que o compõem serão divididos em arcos a favor e contra o fluxo cuja direção dependerá do arco entrante (i, j) de maneira que, se $(i, j) \in \mathbf{U}$, o fluxo no ciclo de pivoteamento terá sentido inverso ao de (i, j), ou seja, um aumento no fluxo do ciclo impõe uma diminuição no fluxo do arco, e que, se $(i, j) \in \mathbf{L}$, o fluxo terá o mesmo sentido de (i, j). A figura 3.10 ilustra esta

diferença: na figura 3.10a, na qual o arco (i, j) não possui fluxo inicialmente, o fluxo no ciclo se dará no sentido anti-horário; já na figura 3.10b, na qual o arco (i, j) possui fluxo igual à sua capacidade máxima, o fluxo no ciclo se dará no sentido horário.



(a) $(i, j) \in \mathbf{L}$ (b) $(i, j) \in \mathbf{U}$

Figura 3.10 - Direção do fluxo ao longo do ciclo de pivoteamento.

Havendo identificado o ciclo de pivoteamento e o seu sentido, deverá ser calculado o montante máximo de fluxo, Δ, que poderá ser enviado através deste, o que equivale a determinar qual o acréscimo ou decréscimo máximo no fluxo suportado por cada arco do ciclo, δ, no sentido determinado por este. Desta maneira, o arco dentro do ciclo que apresentar o menor valor de δ será chamado de arco limitador e determinará o valor de Δ. Formalmente pode-se dizer que, para um arco pertencente ao ciclo (k,l):

$$\delta_{k,l} = x_{k,l} \tag{3.17}$$

se o sentido de (k,l) for contrário ao do fluxo no ciclo e

$$\delta_{k,l} = u_{k,l} - x_{k,l} \tag{3.18}$$

se o sentido de (k,l) for a favor do fluxo no ciclo. Para Δ tem-se então que:

$$\Delta = \min\{\delta_{k,l}\} \tag{3.19}$$

Conforme mostrado em Ahuja e Magnatti (1993), toda a operação de identificação do ciclo e atualização dos fluxos pode ser feita com apenas dois procedimentos. Primeiramente, juntamente com a identificação do ciclo de pivoteamento dado pela função 3, calcula-se para cada arco do ciclo o valor de δ de tal maneira que ao final do procedimento já se tenha definido tanto o valor de Δ, quanto o arco que deixará a base. Em seguida os arcos que compõem o ciclo são novamente percorridos e têm o seu fluxo atualizado com base no valor de Δ. Ainda conforme Ahuja e Magnatti (1993), toda a operação descrita anteriormente é executada com ordem O(n) para o pior caso, muito embora em geral seja necessário examinar apenas um pequeno subconjunto dos nós.

O exemplo dado pela figura 3.11 ilustra bem todo o processo. Seja um arco (5,2), com fluxo $x_{5,2} = u_{5,2} = 4$ pertencente, portanto, ao conjunto **U**. Supondo que o referido arco possui custo reduzido $c_{5,2} \geq 0$, ou seja, que este viole a condição de otimalidade dada por (3.19), conclui-se que a diminuição no seu fluxo acarretará uma diminuição no custo total da solução e que, portanto, se trata de um arco elegível para entrar na árvore. Entrando na árvore o arco (5,2) forma, como se pode observar na figura 3.11a, um ciclo com os arcos (5,4), (1,4) e (2,1). Após identificado o ciclo é calculado o valor de Δ que, neste caso, á igual a 2 (figura 3.11b) sendo o arco limitador o arco (1,4) que, ao final do pivoteamento, deixa a árvore passando a pertencer ao conjunto **L** e dando origem uma nova árvore geradora (figura 3.11c).

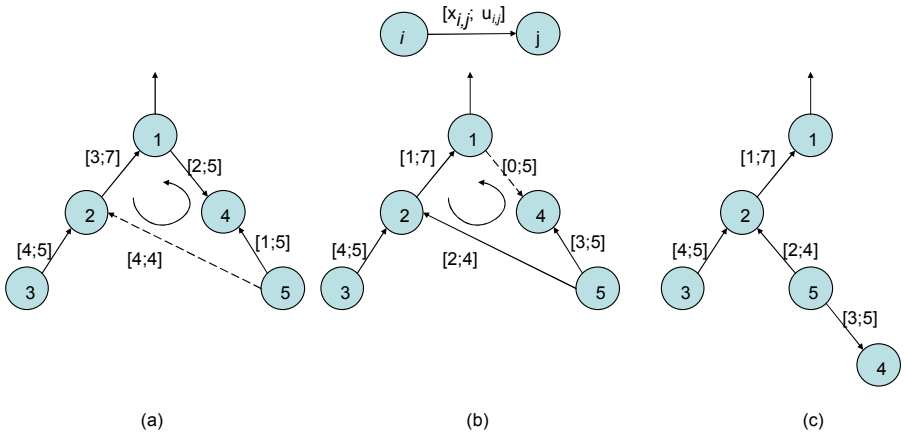


Figura 3.11 - Exemplo de pivoteamento, entrada do arco (5,2) e saída do arco (1,4).

3.15.Atualizando a árvore

Repare que um arco entrante, (i, j), será também um candidato a sair da árvore e, sendo (i, j) o arco limitador, o que ocorrerá na prática será apenas uma transferência entre os conjuntos **L** e **U**, ou seja, (i, j) poderá passar do conjunto **L** para o conjunto **U** ou vice-versa, alterando a solução sem todavia alterar árvore geradora em si.

Na maioria dos casos o arco entrante será diferente do arco que sai, modificando a árvore geradora correspondente e tornando necessária a atualização dos potenciais dos nós, bem como dos índices da árvore.

3.15.1.Atualização dos potenciais dos nós

Sejam (i, j) e (k,l) respectivamente o arco entrante e o arco que sai da base. Ao ser retirado da árvore, o arco (k,l) sempre dividirá o conjunto de nós em duas sub-árvores , T_1 e T_2 onde T_1 contém o nó raiz. De maneira análoga o arco (i, j) conterá um extremo em cada uma das sub-árvores.

Para definir os valores iniciais dos potenciais dos nós foi atribuído ao nó raiz um potencial nulo, $\pi_1 = 0$ (o custo de se transportar uma unidade de fluxo da raiz para ela mesma é zero) e, para os demais, o potencial foi calculado mediante a aplicação da fórmula $\pi_j = c_{ij} + \pi_i$ (3.10). Repetindo os mesmos cálculos para a nova árvore observa-se que os valores para os potenciais dos nós pertencentes a T_1 não se alteram, variando apenas os valores para os nós pertencentes a T_2. Para atualizar os potenciais da árvore basta, portanto, calcular os novos potenciais dos nós de T_2 que, como será visto em seguida, irão variarão apenas por uma constante.

Seja o arco (i, j) tal que *i* esteja em T_1, o potencial do nó *i* não será alterado na nova árvore e o novo potencial de *j* será dado por $\pi_j^* = c_{ij} + \pi_i$ ou ainda, se considerando-

se a expressão $c_{ij}^{\pi} = c_{ij} + \pi_i - \pi_j$ dada por (3.13), $\pi_j^* = c_{ij}^{\pi} + \pi_j$. Estando T_2 "pendurada" pelo nó *j*, para atualizar o potencial dos nós restantes basta que se some c_{ij}^{π} aos seus potenciais atuais. De maneira análoga pode-se verificar que, estando *j* em T_1, o valor da constante a ser somada aos potenciais dos nós será $-c_{ij}^{\pi}$. A função 4 será utilizada para efetuar esta atualização.

Função F:

Se $k \in T_2$ *então y = k, caso contrário y = l*

Se $i \in T_1$ *então constante =* c_{ij}^{π}*, caso contrário constante =* $-c_{ij}^{\pi}$

$\pi(y) = \pi(y)$ *+ constante*

z = y

enquanto profundidade(caminho(z)) > profundidade(y) fazer:

π *(caminho(z)) =* π *(caminho(z)) + constante*

A utilização dos índices profundidade e caminho faz com que a atualização dos potenciais seja rápida e eficiente, mas estes mesmos precisam ser atualizados também e, se as suas atualizações não forem eficientes, a vantagem de utilizá-los será reduzida.

Para atualizar os índices da árvore, seja o arco entrante (i, j) tal que $i \in T_1$ e $j \in T_2$ e seja o arco que sai (k,l) tal que $k \in T_1$ e $l \in T_2$. A figura 3.12 mostra um pivoteamento onde o arco (2,13) é o arco entrante e o arco (1,4) é o arco que sai, ou seja, os nós 2, 13, 1 e 4 correspondem respectivamente a *i*, *j*, *k* e *l*.

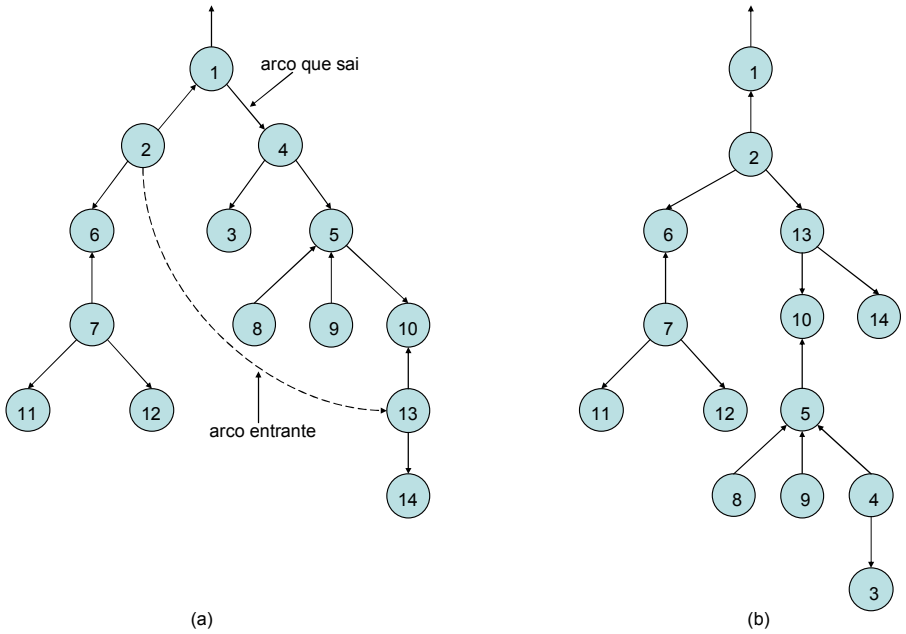


Figura 3.12 - Atualização dos índices da árvore

O caminho entre *j* e *l* será chamado de *pivot stem* ou *backpath* e os nós pertencentes a este caminho de j_h sendo que o índice h nos dá a ordenação destes nós dentro do caminho de maneira que para um nó qualquer, *v*, pertencente a este caminho tem-se:

$$h = profundidade(j) - profundidade(v) + 1 \qquad (3.20)$$

Desta maneira, para o exemplo dado pela figura 3.12, tem-se que $j_1 = 13$, $j_2 = 10$, $j_3 = 5$ e $j_4 = 4$.

Defina-se agora, para cada j_h, um subconjunto S_h de T_2 que ira conter, além do próprio nó, todos aqueles nós em cujos caminhos em direção à raiz o primeiro nó pertencente ao *backpath* seja j_h, de maneira que a ordem dos elementos dentro de cada subconjunto respeite a ordenação dada pelo índice caminho original.

Para o exemplo dado pela figura 3.12, considerando os índices caminho como sendo dados por:

$$\begin{array}{cc} i & \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \end{bmatrix} \\ \textbf{caminho}(i) & \begin{bmatrix} 2 & 6 & 5 & 3 & 8 & 7 & 11 & 9 & 10 & 13 & 12 & 4 & 14 & 1 \end{bmatrix} \end{array}$$

Tem-se os seguintes conjuntos que serão utilizados na atualização dos índices da árvore (este procedimento é mostrado também em Chvatal (1980)):

$S_1 = \{13, 14\}$

$S_2 = \{10\}$

$S_3 = \{5, 8, 9\}$

$S_4 = \{4, 3\}$

3.15.2.Atualização dos índices caminho.

Seja S^* a concatenação dos vetores S_h. No nosso exemplo S^* será tal que $S^* = \{13,14,10,5,8,9,4,3\}$. Para a obtenção dos índices caminho da nova árvore primeiramente retira-se do caminho da árvore original a seqüência referente a T_2 e a substitui por S^*, em seguida restam apenas alguns ajustes, são eles:

1. Quando o bloco referente a T_2 é removido o sucessor do antecessor de l passa a ser o sucessor do último nó do bloco T_2
2. O antecessor do sucessor de i passa a ser o último nó do bloco S^*
3. O sucessor de i passa a ser o primeiro nó do bloco S^*

A figura 3.13 mostra a dinâmica de atualização para o exemplo. Em 14a é retirado o bloco referente a T_2 e são feitos os devidos ajustes, em 14b é colocado o bloco S^*,

também respeitando os ajustes necessários, e finalmente em 14c tem-se a nova ordenação dos nós.

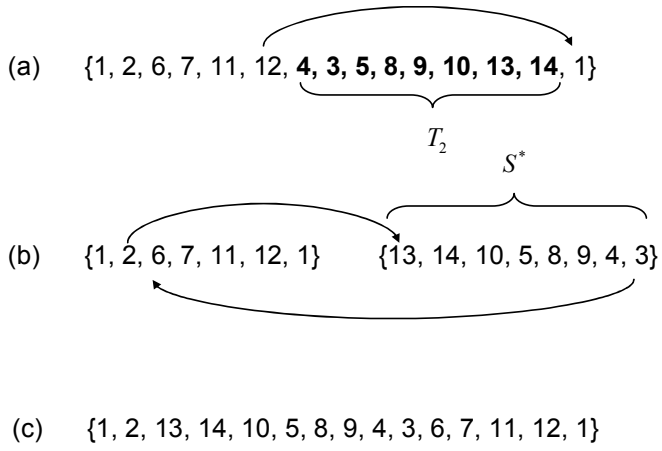


Figura 3.13– Exemplo: dinâmica de atualização dos índices

Ao final os novos índices de caminho serão dados por:

$$\begin{array}{cc} i & \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \end{bmatrix} \\ \textbf{caminho}^*(i) & \begin{bmatrix} 2 & 13 & 6 & 3 & 8 & 7 & 11 & 9 & 4 & 5 & 12 & 1 & 14 & 10 \end{bmatrix} \end{array}$$

3.15.3. Atualização das profundidades

A exemplo do que ocorre para os índices de caminho, os índices de profundidade só precisam ser atualizados para aqueles nós pertencentes a T_2 e, uma vez que $i \in T_1$ e $j \in T_2$ a nova profundidade de j será o valor dado pela expressão $profundidade*(j) = profundidade(i) + 1$. Seja portanto $\theta(j)$ a variação na profundidade de j que será dada por $\theta(j) = profundidade(i) - profundidade(j) + 1$. Já o antecessor de j, se houver, terá $profundidade*(ant(j)) = profundidade*(j) + 1$ o que corresponde à variação $\theta(ant(j)) = \theta(j) + 2$.

Calcula-se a variação para o predecessor do predecessor, e assim por diante, até se encontrar o nó *l* quando terá sido percorrido todo o *backpath* e ter-se-á, para cada j_h, como definido anteriormente, o valor da variação θ_h que, sendo $\theta(j) = \theta_1$, será dada pela expressão:

$$\theta_h = \theta_1 + 2(h-1) \tag{3.21}$$

Por último, para cada conjunto S_h deve-se aplicar a todos os seus elementos a variação dada pelo θ_h correspondente. Os valores para as profundidades dos nós no exemplo dado pelas figuras 3.12a e 3.12b serão os seguintes:

Antes (3.12a):

$$\begin{matrix} i \\ profundidade(i) \end{matrix} \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 0 & 1 & 2 & 1 & 2 & 2 & 3 & 3 & 3 & 3 & 4 & 4 & 4 \end{bmatrix}$$

Depois (3.12b, em negrito as mudanças):

$$\begin{matrix} i \\ profundidade*(i) \end{matrix} \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 0 & 1 & \mathbf{6} & \mathbf{5} & \mathbf{4} & 2 & 3 & \mathbf{5} & \mathbf{5} & \mathbf{3} & 4 & 4 & \mathbf{2} \end{bmatrix}$$

3.15.4.Atualização dos predecessores

A atualização dos predecessores não é complicada, bastando preocupar-se com os nós do *backpath* que serão tratados da seguinte maneira:

$$\begin{cases} predecessor(j_h) = j_{h-1} \quad se \ \ h > 1 \\ predecessor(j_1) = i \end{cases} \tag{3.22}$$

Os valores para os predecessores dos nós no exemplo dado pela figura 3.12 seriam os seguintes:

Antes (3.12a):

$$\begin{array}{c} i \\ predecessor(i) \end{array} \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 0 & 1 & 4 & 1 & 4 & 2 & 6 & 5 & 5 & 5 & 7 & 7 & 10 \end{bmatrix}$$

Depois (13b, em negrito as mudanças):

$$\begin{array}{c} i \\ predecessor^*(i) \end{array} \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 0 & 1 & 4 & \mathbf{5} & \mathbf{10} & 2 & 6 & 5 & 5 & \mathbf{13} & 7 & 7 & \mathbf{2} \end{bmatrix}$$

3.16. Algoritmo

Pode-se agora especificar os passos do algoritmo simplex para redes da seguinte maneira:

Algoritmo Simplex para Redes

Determinar uma solução inicial e os fluxos associados aos arcos (item 3.7)

Calcular os potenciais dos nós associados a esta solução (item 3.9)

Enquanto houver arcos elegíveis para entrar na base fazer: (item 3.10)

Selecionar um arco para entrar na base (item 3.13)

Determinar o arco que sai da base (item 3.14)

Atualizar: os índices e a estrutura da árvore, os fluxos e os potenciais (item 3.15)

Fim

3.17. Degenerescência

Não são raras as ocorrências de árvores geradoras que não são exclusivamente compostas por arcos livres, ou seja, existe $(i, j) \in \mathbf{T}$ tal que $x_{i,j} = 0$ ou $x_{i,j} = u_{i,j}$; esse fato, ao qual se dá o nome de degenerescência, é na verdade extremamente comum e o seu tratamento de grande importância prática pois uma base degenerada pode ocasionar o fenômeno de *cycling* (uma seqüência infinita de pivoteamentos).

Para evitar o fenômeno de *cycling* basta garantir que a árvore geradora seja fortemente factível durante a execução do algoritmo como será mostrado a seguir.

3.18. Árvores Geradoras Fortemente Factíveis

Segundo a definição de Ahuja *et al.* (2002), pode-se dizer que uma árvore geradora é fortemente factível se for possível enviar alguma quantidade positiva de fluxo de qualquer nó *i* para o nó raiz através dos arcos básicos sem violar nenhuma das restrições de fluxo dos mesmos.

Alternativamente pode-se definir uma árvore geradora fortemente factível como uma árvore em que qualquer arco (i, j) com fluxo tal que $x_{ij} = 0$ aponte em direção à raiz (topo da árvore) e qualquer arco (i, j) com fluxo tal que $x_{ij} = u_{ij}$ aponte para a direção oposta à da raiz. As duas definições são equivalentes e uma implica na outra.

Observe que uma árvore não degenerada (que seja composta apenas de arcos livres), será sempre fortemente factível; enquanto uma árvore degenerada poderá ou não sê-lo em virtude da observância ou não das características apresentadas anteriormente.

Partindo do pressuposto que uma árvore é fortemente factível deve-se, então, garantir que a cada iteração a factibilidade forte seja mantida e para tal basta que seja utilizada a seguinte regra de seleção do arco que sai: sempre que houver mais de um arco limitador no ciclo de pivoteamento escolher-se-á para sair da base o último deles, encontrado ao se percorrer o ciclo partindo de *w* (antecessor comum dos nós origem e destino do arco entrante).

A figura 3.14a mostra um exemplo em que todos os arcos de um ciclo de pivoteamento, com exceção do arco entrante, são limitadores, o que irá gerar uma nova árvore geradora degenerada. A figura 3.14b mostra como uma escolha equivocada do arco que sai da árvore não gera uma árvore degenerada fortemente factível, o mesmo não acontecendo quando se utiliza a regra apresentada anteriormente, como mostra a figura 3.14c.

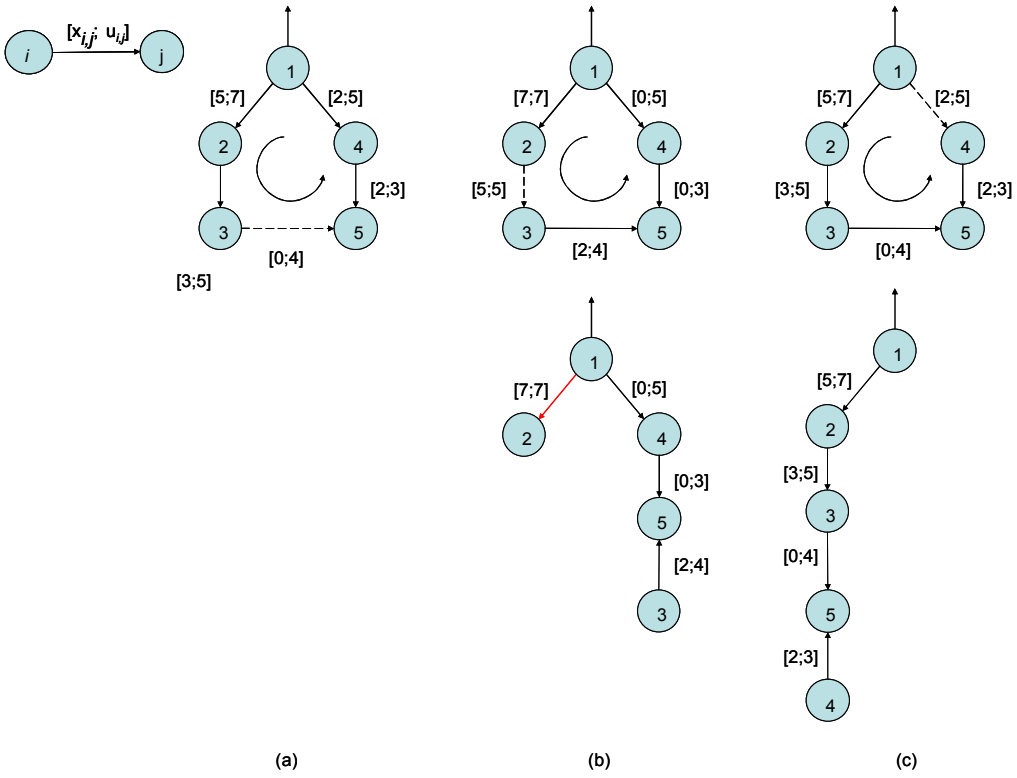


Figura 3.14- Árvores fortemente factíveis, regra de seleção do arco que sai da base.

Note que nenhuma outra escolha que não a dada por 3.14c irá gerar uma árvore fortemente factível.

Ahuja *et al.* (2002) mostram que, mesmo mantendo-se uma árvore geradora fortemente factível, o algoritmo ainda pode executar uma seqüência exponencialmente grande de pivoteamentos degenerados, um fenômeno conhecido como *stalling* e prova que, utilizando um ciclo aumentante de custo negativo para identificar a seqüência de variáveis entrantes, pode-se chegar a um limite de k para o número consecutivo de pivoteamentos degenerados onde k é o número de arcos degenerados na base.