UNIVERSIDADE FEDERAL DA PARAÍBA

CENTRO DE CIÊNCIAS E TECNOLOGIA

DEPARTAMENTO DE SISTEMAS E COMPUTAÇÃO

DETERMINAÇÃO DE ROTEIROS ÓTIMOS EM ALGUNS SERVIÇOS PÚBLICOS

POR

HASSAN SHERAFAT

CAMPINA GRANDE - PARAÍBA

AGOSTO - 1978

S551d Sherafat, Hassan.
Determinação de roteiros ótimos em alguns serviços públicos / Hassan Sherafat. - Campina Grande, 1978.
108 f.

Dissertação (Mestrado em Ciências) - Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, 1978.
"Orientação : Prof. Dr. Jean Claude Picard".
Referências.

1. Algoritmos - Resolução de Problemas. 2. Serviços Públicos - Roteiros Ótimos. 3. Serviços Públicos - Distribuição. 4. Dissertação - Ciências. I. Picard, Jean Claude. II. Universidade Federal da Paraíba - Campina Grande (PB). III. Título

CDU 004.021(043)

DETERMINAÇÃO DE ROTEIROS ÓTIMOS EM ALGUNS SERVIÇOS PÚBLICOS

HASSAN SHERAFAT

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS CURSOS DE PÓS-GRADUAÇÃO DE ENGENHARIA DO CENTRO DE CIÊNCIAS E TECNOLOGIA DA UNIVERSIDADE FEDERAL DA PARAÍBA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.).

Aprovado Por:

JEAN CLAUDE PICARD - Ph.D.
- Presidente -

SIMIN JALALI R. RABBANI - M.Sc.
- Examinador -

EDUARDO ANDRADE VELOSO - M.Sc.
- Examinador -

Campina Grande - Paraíba

Agosto de 1978

A meu irmão Abbas pelo incentivo e apoio sempre demonstrado, sem os quais não seria possível a realização deste trabalho.

A minha querida esposa Flora, pela compreensão, carinho, ajuda e incentivo em todos os momentos.

AGRADECIMENTOS

O autor agradece:

Ao seu orientador e amigo Prof. Ph.D. JEAN CLAUDE PICARD, pela atenção, incentivo e amizade, não só na orientação deste trabalho, mas também no decorrer do Curso de mestrado;

Ao Prof. M.Sc EDUARDO ANDRADE VELOSO, Coordenador do curso de Pós-Graduação em Engenharia de Sistemas, pela sua valiosa colaboração sem a qual não seria possível concluir este trabalho;

A todos os colegas de curso pela amizade sempre demonstrada.

ABSTRACT

In this work a class of public service distribuition problems are considered in which the demand is distributed along the streets. Tipical problems of this class are: Garbage collection, postman problem etc.

The principal effort is made in the determination of vehicle routes, to minimize total travel distance. The classic model of Chinese postman problem is used to solve this problem.

In a computational experience, a real problem of garbage collection was solved. The comparison of obtained results and the existing solution demonstrates that the work could be applied to improve the service in practice.

RESUMO

Neste trabalho, é considerada uma classe de problema de distribuição de serviços públicos na qual a demanda é distribuída ao longo das ruas. Problemas típicos de tal classe são: a coleta de lixo, a distribuição de cartas etc.

O esforço principal é concentrado no problema da determinação de roteiros dos veículos, de modo a minimizar as distancias totais percorridas. O modelo clássico do problema do Carteiro Chinês é usado para a solução desse problema.

Numa experiência computacional, um problema real da coleta de lixo foi resolvido. A comparação dos resultados obtidos com a solução existente, mostrou que o trabalho pode ter aplicações práticas e aumentar a produtividade dos serviços.

ÍNDICE

CAPÍTULO I

INTRODUÇÃO

O problema da distribuição de serviços públicos é um dos problemas que envolvem mais pesquisas e trabalhos depois de 1960. Exemplos típicos de tais serviços são: limpeza de ruas, coleta de lixo, distribuição de cartas etc.

Para perceber a importância de considerar o problema de distribuição de serviços públicos, basta dizer que em 1973 nos EUA foram despendidos aproximadamente 4,5 bilhões de dólares para a coleta de lixo. Isto foi 10 por cento do orçamento total da comunidade [12]. Esses fatos podem mostrar a importância de tais problemas para qualquer outro país.

Até anos recentes, o maior esforço no campo da pesquisa operacional era concentrado nos problemas de serviços privativos, i. e., problemas enfrentados pela indústria e comércio. Enquanto isso, poucos esforços foram feitos para a sólução dos problemas enfrentados pelos setores públicos. Talvez a maior razão fosse devido à complexidade e ao porte desses problemas. Felizmente, com o

desenvolvimento das técnicas da pesquisa operacional e a disponibilidade dos computadores de alta velocidade, agora se tornou possível tentar resolver esses problemas, ou pelo menos aumentar a produtividade dos serviços.

Alguns tópicos já considerados no problema da distribuição de serviços públicos são os seguintes:

- Alocação e sequenciamento de facilidades (veículos, mão de obra etc);

- Divisão de uma área em seções de demanda igual;

- Localização de algumas facilidades para a transferência do serviço (estações de transferência);

- Determinação do roteiro dos veículos.

A análise da determinação do roteiro torna-se muito importante para uma classe de problemas de distribuição de serviços públicos. Esse problema pode ser definido das seguintes maneiras:

a) Determine um roteiro para servir a um conjunto de locais distintos, de modo que a distância total do roteiro seja mínima. Esse problema na literatura é conhecido como "o problema do caixeiro viajante".

b) Determine um roteiro de distância mínima para servir a todos os segmentos de uma rede, onde a demanda é distribuída ao longo dos segmentos. Este é conhecido como "o problema do Carteiro Chinês".

c) Determine um roteiro para servir a um subconjunto de locais

é a um subconjunto dos segmentos de uma rede de tal maneira que a distância total percorrida seja mínima. Este é chamado "o problema geral de determinação do roteiro".

Nesta tese, é considerado o problema de determinação do roteiro para a distribuição de serviços públicos ao longo dos segmentos de uma rede (o problema do Carteiro Chinês). Exemplos de tal problema são: a distribuição de cartas, a limpeza de ruas pelas vassouras mecânicas, a coleta de lixo, a inspeção de fios de transmissão de energia, a distribuição de leite, passeios regulares dos policiais nas ruas nos expedientes à noite etc.

Uma definição mais precisa do problema considerado nesta tese pode ser o seguinte: Dada uma área definida por um conjunto de segmentos de rua, e um conjunto de interseções, onde o serviço deve ser distribuído ao longo das ruas, um custo de percurso (também pode ser tempo ou distância) é associado a cada segmento da rua, o objetivo é determinar um roteiro que começa de um ponto, percorre todas as ruas pelo menos uma vez e volta ao ponto inicial, de tal modo que o custo total de percurso seja mínimo.

Nós consideramos esse problema em três casos diferentes:

a) Quando todas as ruas são de sentido duplo;

b) Quando todas as ruas são de sentido único;

c) Quando algumas ruas são de sentido único e algumas outras de sentido duplo.

Nos Capítulos 3 e 4 todos os casos serão considerados e serão apresentados métodos exatos e eficientes para resolver o proble

ma nos casos (a) e (b). No Capítulo 5 a teoria básica para o desenvolvimento dos algoritmos será apresentada.

Muitas vezes, esses algoritmos não podem ser aplicados diretamente para resolver os problemas reais da distribuição de serviços públicos. Por exemplo, na coleta de lixo cada veículo é sujeito à sua capacidade, portanto, um veículo numa só viagem possivelmente não pode fazer o serviço da cidade inteira. Geralmente, mais que um veículo é necessário e cada vez que um veículo está cheio, deve voltar ao depósito de lixo e descarregar. Isto significa que deve ser determinado um conjunto de roteiros ao invés de um só. Mas ainda os algoritmos apresentados podem dar soluções boas (mas não necessariamente ótimas) para esses problemas. Como um exemplo, a implementação dos algoritmos para resolver o problema da coleta de lixo é considerado no Capítulo 6.

Como uma experiência, o problema da coleta de lixo do Bairro dos Estados, da cidade de João Pessoa, foi considerado e foi usado para testar os programas desenvolvidos nesse trabalho. Os resultados dessa experiência são apresentados no Capítulo 7. Finalmente, no último capítulo, conclusões e sugestões serão abordadas.

CAPÍTULO II

DEFINIÇÕES

2.1 Definição de um Grafo

Um grafo $G = (N,A)$ é uma estrutura composta de um conjunto finito N de elementos chamados nós e de um conjunto A de pares de nós chamados ramos. Como exemplo, a figura 2.1(a) mostra um grafo no qual os nós são representados por circulos e os ramos por linhas. Nesse exemplo $N = \{1, 2, 3, 4, 5\}$ e $A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7,\}$.

Se os ramos de conjunto A tiverem uma direção, que é mostrada por uma seta, eles serão chamados arcos e o grafo é chamado um grafo direcionado (Figura 2.1(b)). Se os ramos não têm nenhuma direção, eles são chamados ligações e o grafo é chamado não-direcionado (Figura 2.1(a)). Um grafo composto de alguns ramos direcionados (arcos) e outros não-direcionados (ligações) é chamado misto (Figura 2.1(c)).

Neste trabalho um arco também é denotado pelos seus nós terminais. Por exemplo, se o nó i é o nó inicial do arco a_k e o nó j é

o seu nó final, então a_k pode ser denotado por (i,j). Portanto, na figura 2.1(b) o arco a_2 pode ser representado por (5,2). Nesse trabalho uma ligação sempre pode ser substituída por dois arcos contrariamente direcionados. Então uma ligação a_k cujos terminais são i e j, pode ser denotada por (i,j) ou (j,i).

2.2 Cadeias e Caminhos

Uma cadeia num grafo não-direcionado é qualquer sequência de ligações onde o nó final de uma é o nó inicial da próxima. Portanto, na Figura 2.2 as sequências das ligações:

(2.1) a_5, a_8, a_6, a_2

(2.2) a_9, a_8, a_7, a_6

(2.3) $a_3, a_7, a_{10}, a_9, a_8, a_7 a_2,$

são cadeias.

Os ramos $(i,j)\varepsilon A$ e $(j,k)\varepsilon A$ que têm o nó j em comum, são chamadas incidentes. Então na Figura 2.2 os ramos a_1 e a_4 são incidentes.

Uma cadeia simples é uma cadeia que não usa a mesma ligação mais que uma vez. Portanto, as cadeias (2.1) e (2.2) acima são simples, mas a cadeia (2.3) não é, porque usa a_7 duas vezes.

Uma cadeia elementar é uma cadeia que não usa o mesmo nó mais que uma vez. Então a cadeia (2.1) é elementar, mas as cadeias (2.2) e (2.3) não o são.

Um caminho é uma cadeia que tem uma direção determinada. Por

exemplo, na Figura 2.1(c) a sequência dos ramos a_7, a_3, a_1 forma um caminho do nó 2 ao nó 1. Se existe uma cadeia entre dois nós i e j, então existe um caminho do nó i ao nó j e um outro do nó j para o i (porque cada ligação da cadeia pode ser substituída por dois arcos de direções contrárias).

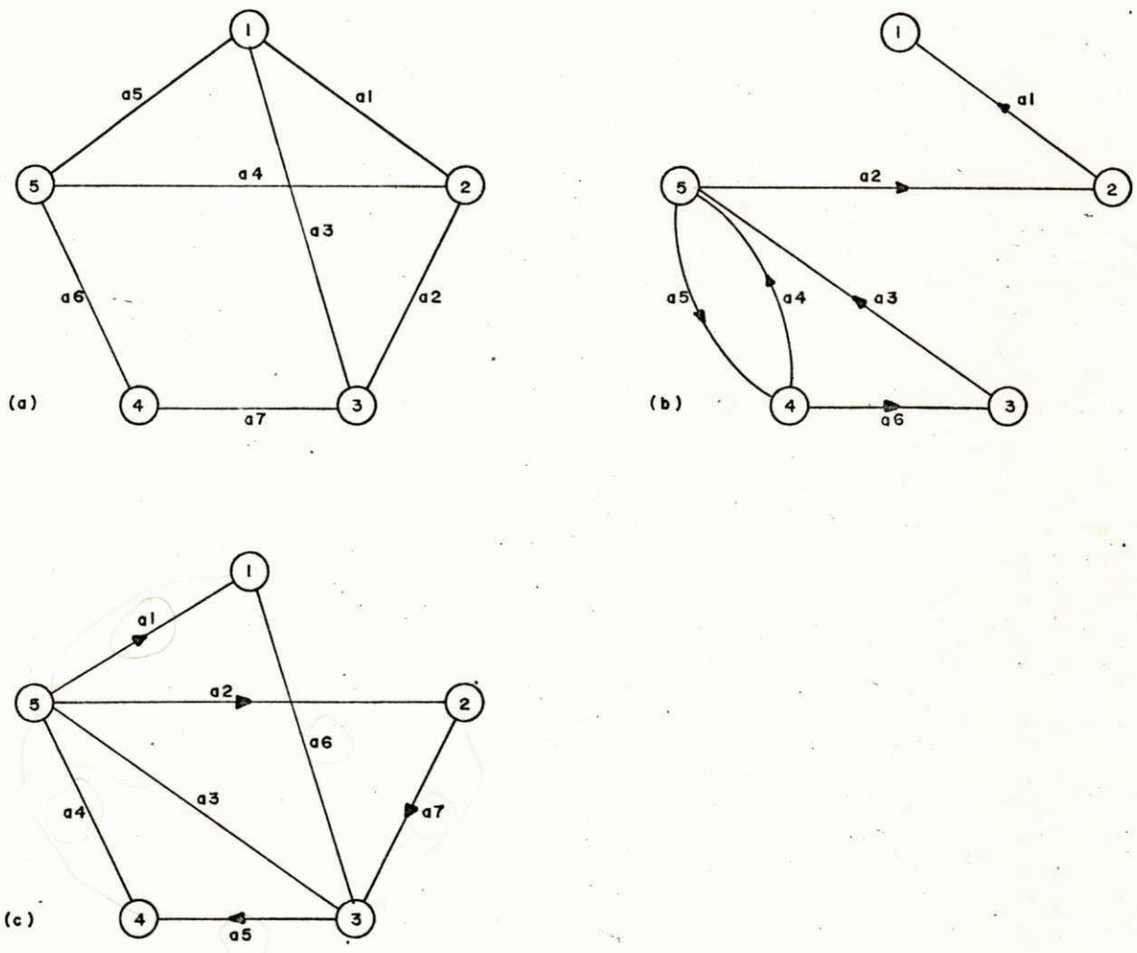


Figura 2.1: Exemplo de um grafo direcionado, não direcionado e misto

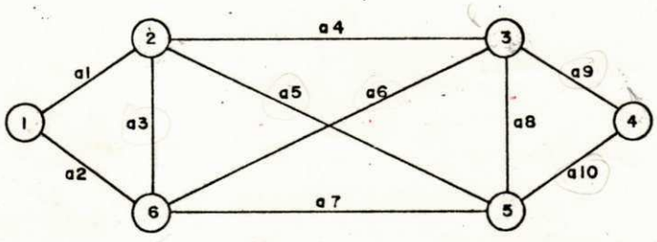


Figura 2.2.

2.3 Pesos

Um número C_{ij} às vezes pode ser associado a um ramo (i,j). Esses números são chamados pesos. Também um peso u_i às vezes pode ser associado a um nó i. O valor de uma cadeia (ou caminho) é definido como a soma dos pesos dos ramos na cadeia (ou caminho).

2.4 Cardinalidade

Dado um grafo $G = (N,A)$, seja B um subconjunto de A. A cardinalidade do subconjunto B, denotada por $|B|$, é definida como o número dos ramos de B. Por exemplo, na Figura 2.1(c) a cardinalidade do subconjunto dos ramos definidos por $\{a_1, a_7, a_4\}$ é três.

2.5 Ciclos e Circuitos

Um Ciclo é uma cadeia na qual o nó inicial e o final são os mesmos. Portanto, na Figura 2.2, a cadeia formada pela sequência $a_6, a_9, a_{10}, a_5, a_3$ é um ciclo.

Um Circuito é um ciclo direcionado. Portanto, um circuito é um caminho que começa e termina no mesmo nó. Na Figura 2.1(c) a sequência a_1, a_6, a_5, a_4 é um circuito.

O valor de um ciclo (ou circuito) é definido como a soma dos pesos dos ramos no ciclo (ou circuito).

Ciclos (ou circuitos) simples e elementares são definidos da mesma forma que cadeias (ou caminhos).

2.6 Grau de um Nó

O grau de um nó i é definido como o número dos ramos inciden

tes no mesmo e é denotado por d_i. Portanto, na Figura 2.2 o grau do nó 6 $d_6 = 4$ e o grau do nó 1 $d_1 = 2$.

O grau de saída de um nó i é o número dos ramos direcionados cujo nó inicial é o próprio i e, semelhantemente, o número dos ramos direcionados cujo nó final é o nó i, é chamado grau de entrada. Por exemplo, na Figura 2.1(b) o grau de entrada do nó 4 é um e o seu grau de saída é dois.

2.7 Grafos Parciais e Subgrafos

Dado um grafo $G = (N,A)$, um grafo parcial G_P de G é o grafo (N,A_P) com $A_P \subset A$. Portanto, um grafo parcial é um grafo com o mesmo número dos nós, mas com somente um Subconjunto dos ramos do grafo original.

Se a Figura 2.3(a) representa o grafo G, a Figura 2.3(b) é um grafo parcial G_P.

Dado um grafo $G = (N,A)$, um subgrafo G_s é o grafo (N_s,A_s) com $N_s \subset N$ e $A_s = \{(i,j) \mid i \in N_s,\ j \in N_s,\ \text{e } (i,j) \in A\}$. Portanto, um subgrafo tem um subconjunto N_s do conjunto dos nós do grafo original,

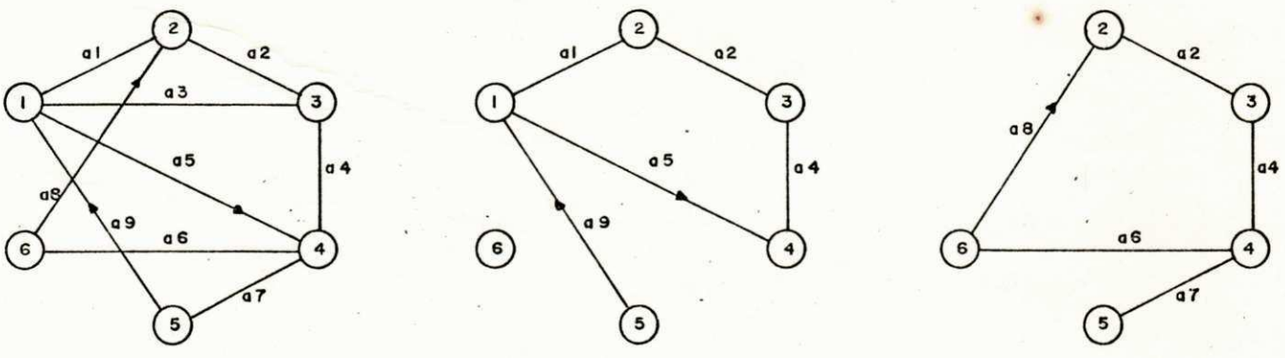


Figura 2.3

mas contêm somente os ramos cujos nós iniciais e finais estão no subconjunto N_s. A Figura 2.3(c) mostra um subgrafo do grafo da Figura 2.3(a).

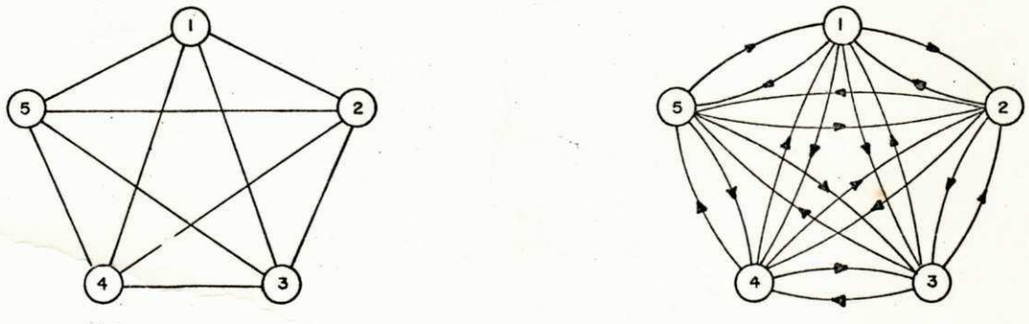
2.8 Tipo dos Grafos

2.8.1 Grafo Conexo

Um grafo $G = (N,A)$ é conexo, se para qualquer dois nós i e j ε N há um caminho em G de i para j. Os grafos das Figuras 2.1(a) e 2.1(c) são conexos, mas o grafo da Figura 2.1(b) não é conexo, porque não existe um caminho nesse grafo do nó 1 para o 5.

2.8.2 Grafo Completo

Um grafo $G = (N,A)$ é chamado Completo, se para dois nós quaisquer i e j ε N $(i \neq j)$, o ramo $(i,j) \varepsilon A$. Por exemplo, o grafo da Figura 2.4(a) é um grafo completo não-direcionado e o grafo da Figura 2.4(b) é um grafo completo direcionado.



(a) Um grafo Completo não-direcionado

(b) Um grafo completo direcionado

Figura 2.4

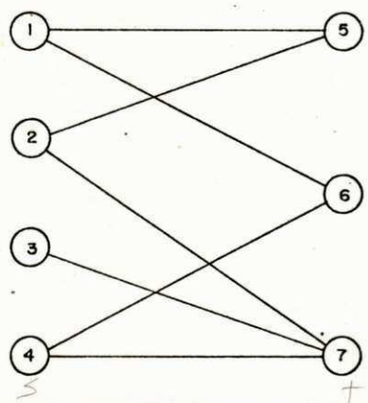
2.8.3 Grafo Bipartido

Um grafo $G = (N,A)$ é chamado um grafo Bipartido, se o conjunto N dos seus nós pode ser dividido em dois subconjuntos S e T, tais que todos os ramos tenham um nó terminal em S e o outro em T. Então, um grafo bipartido pode ser representado por $G = (S,T,A)$. A Figura 2.5(a) mostra um grafo bipartido não direcionado, no qual os nós 1, 2, 3 e 4 formam o conjunto S e os nós 5, 6 e 7 formam o conjunto T..

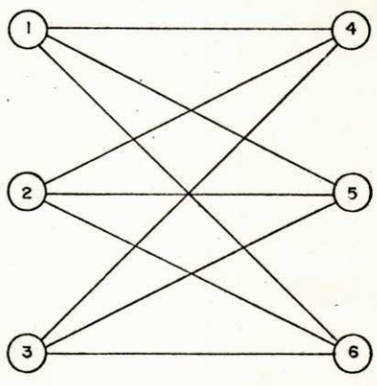
É muito facíl mostrar que:

TEOREMA 2.1 Um grafo não direcionado G é bipartido, se e somente se não contém nenhum Ciclo de cardinalidade ímpar.

Um grafo bipartido $G = (S,T,A)$ é chamado Completo, se para qualquer $i \in S$ e qualquer $j \in T$, o ramo $(i,j) \in A$. A Figura 2.5(b)mostra um grafo bipartido completo.



(a) Grafo bipartido



(b) Grafo bipartido completc

Figura 2.5

2.9 A Representação de Grafo por Matriz

Uma maneira conveniente da representação de um grafo algebricamente é pelo uso de matrizes, como o seguinte.

2.9.1 A Matriz de Adjacência

Dado um grafo G, a sua matriz de Adjacência é denotada por $A = [a_{ij}]$ e é dada por:

a_{ij} = 1 se o arco (i,j) existe em G

a_{ij} = 0 se o arco (i,j) não existe em G.

(Uma ligação pode ser substituída por dois arcos de direções contrárias). Portanto, a matriz de adjacência do grafo mostrado na Figura 2.6 é:

$A =$

	1	2	3	4	5	6
1	0	1	0	0	0	0
2	1	0	1	0	0	0
3	0	1	0	1	1	1
4	0	0	1	0	1	0
5	0	0	0	1	0	1
6	1	1	1	0	0	0

A matriz de adjacência no caso de um grafo bipartido $G = (S,T,A)$ pode ter uma forma mais reduzida. Nesse caso não existem ramos entre nós do conjunto S, então para qualquer $i \varepsilon S$ e $j \varepsilon S$ temos a_{ij} = 0 (o que também ocorre para os nós do conjunto T). Portanto, podemos definir os índices das linhas da matriz como os nós do conjunto S, e os índices das colunas como os do conjunto T. Por exemplo, o grafo bipartido da Figura 2.5(a) pode ser representado pela seguinte matriz:

$$A = \begin{array}{c|ccc} & 5 & 6 & 7 \\ 1 & 1 & 1 & 0 \\ 2 & 1 & 0 & 1 \\ 3 & 0 & 0 & 1 \\ 4 & 0 & 1 & 1 \end{array}$$

2.9.2 A Matriz de Incidência

Dado um Grafo G de n nós e m ramos, a matriz de Incidência de G é denotado por $B = [b_{ij}]$ e é uma matriz de nxm definida como o seguinte:

b_{ij} = + 1 se i é o nó inicial do arco a_k,
ou i é um nó terminal da ligação a_k,

b_{ij} = - 1 se i é o nó final do arco a_k,

b_{ij} = 0 no outro caso.

Para o grafo mostrado na Figura 2.6, a matriz de incidência é:

$$B = \begin{array}{c|ccccccccc} & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 \\ 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 5 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 1 \\ 6 & 0 & 1 & 1 & 0 & 1 & -1 & 0 & 0 & 0 \end{array}$$

Porque cada ramo é incidente exatamente a dois nós, então cada coluna da matriz de incidência contém dois elementos diferentes de ze

ro.

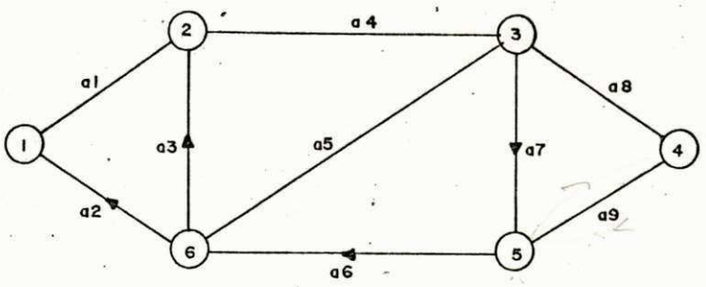
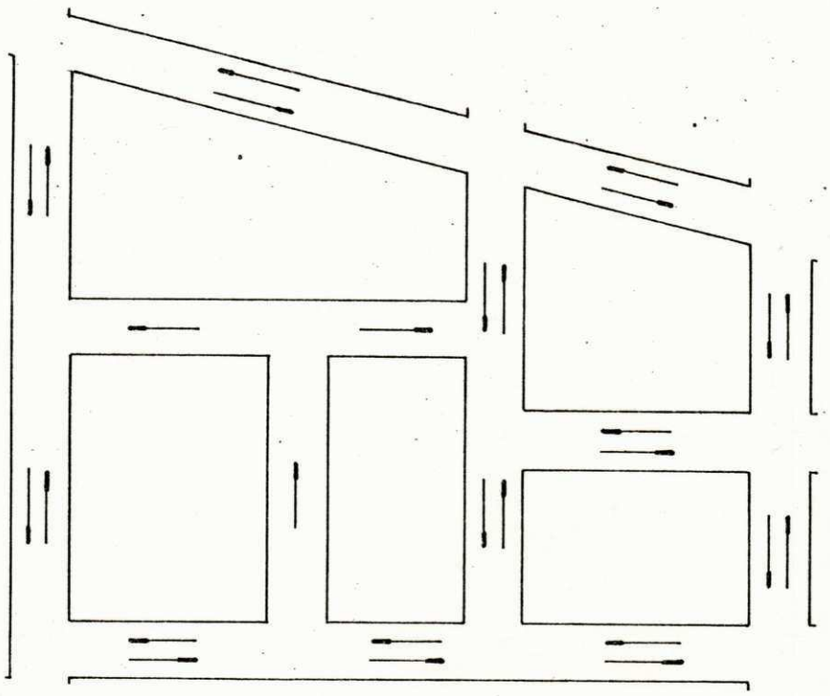


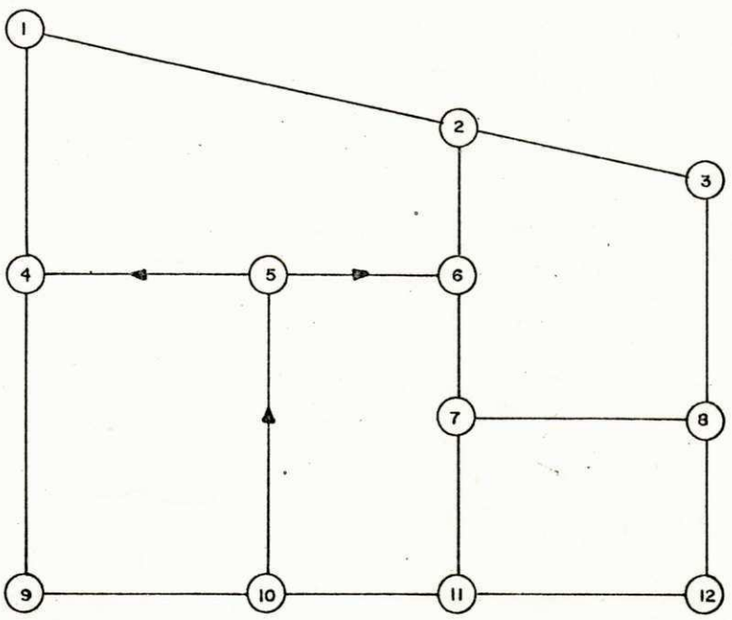
Figura 2.6

2.10 Representação de Rede de Ruas por Grafos

Conforme a definição de grafo, uma rede de ruas pode ser representada pelo mesmo. Para isto representamos cada segmento da rua por um ramo e cada interseção por um nó. Se o segmento da rua não é de direção única, o respectivo ramo é não-direcionado. Se o segmento tem um sentido determinado, então é representado por um arco. Por exemplo, na Figura 2.7(a) é dada uma rede de ruas onde a direção de tráfego de cada segmento de rua é determinada pelas setas. Essa rede pode ser representada pelo grafo misto da Figura 2.7(b). O comprimento (ou custo do percurso) de cada rua pode ser interpretado como um peso para o respectivo ramo.



(a) a rede de ruas



(b) o grafo representante da rede

Figura 2.7

CAPÍTULO III

PROBLEMA DE EULER

3.1 Caso de um Grafo Não-Direcionado

Ciclo Euleriano

Dado um grafo não direcionado G, um Ciclo Euleriano é um Ciclo que percorre todas as ligações de G uma vez e somente uma vez.

Obviamente, nem todos os grafos têm ciclo Euleriano, mas, se existir um ciclo Euleriano, isto significa que ele pode ser desenhado sem levantar o lápis do papel. Como exemplo, o grafo da Figura 3.1 não tem Ciclo Euleriano, mas o grafo da Figura 3.2 tem tal Ciclo dado pela sequência: (Começando no nó n_1)

a_1 a_2 a_3 a_4 a_{15} a_{14} a_{13} a_{12} a_{11} a_{16} a_{17} a_{10} a_9 a_8 a_5 a_7 a_6

A direção a percorrer é mostrada pelas setas na Figura 3.2.

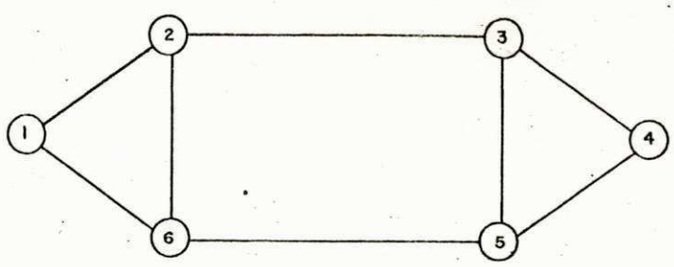


Figura 3.1: grafo sem ciclo Euleriano

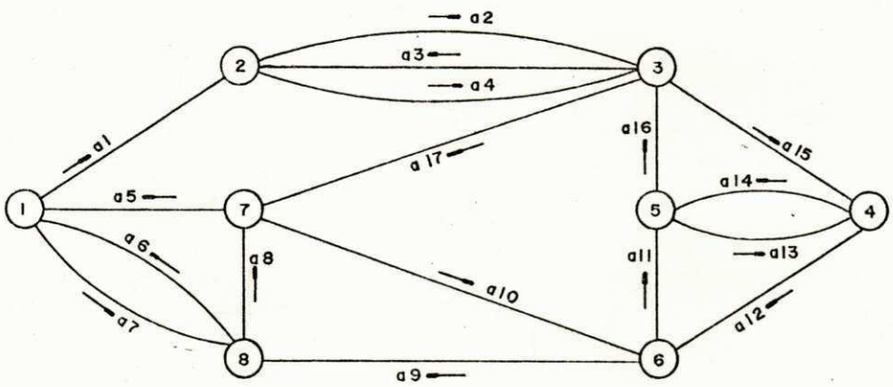
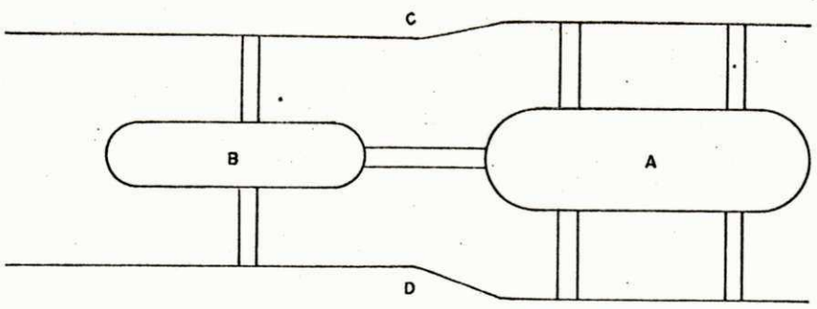
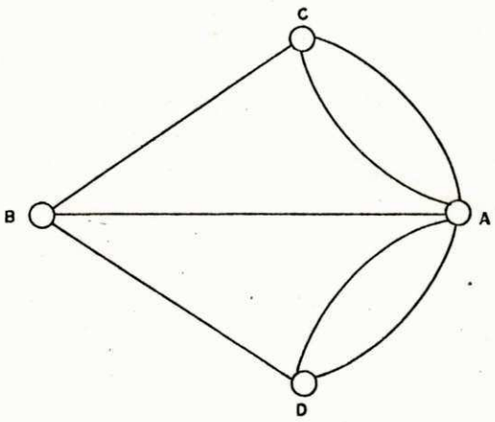


Figura 3.2: grafo com ciclo Euleriano.

Euler, no seu problema célebre da ponte Konigsberg, foi a primeira pessoa que considerou a existência da tais ciclos em grafos. Konigsberg é uma cidade construída em ambas as margens do rio Pregel e sobre duas ilhas no rio. As margens do rio e as duas ilhas são conectadas por sete pontes, como mostrado na Figura 3.3(a). A questão proposta em 1936 era: se possível começar de um ponto qualquer da cidade, cruzar cada ponte exatamente uma vez, e retornar ao ponto inicial. Se cada margem do rio e cada ilha são representadas por um nó e cada ponte por uma ligação, o mapa da Figura 3.3(a) pode ser representado pelo grafo da Figura 3.3(b). O problema agora é determinar se existe um ciclo Euleriano nesse grafo. Euler conclui que o grafo não contém ciclo Euleriano.



(a) Mapa de Konigsberg



(b) Grafo equivalente

Figura 3.3

O teorema básico sobre a existência de um ciclo Euleriano é o seguinte:

TEOREMA 3.1 [8] Um grafo conexo não direcionado G contém um ciclo Euleriano, se e somente se não tiver nós de grau ímpar.

NECESSIDADE. Qualquer ciclo Euleriano deve usar uma ligação para chegar a um nó e uma outra ligação para partir, sendo que todas as ligações devem ser percorridas exatamente uma vez. Então, se G contêm um ciclo Euleriano, o grau de todo nó deve ser par.

SUFICIÊNCIA. Seja G um grafo conexo não direcionado com nós de grau

par. Comece um ciclo em algum nó n_0 arbitrário e prossiga ao longo de uma ligação anteriormente não usada para o próximo nó até que o ciclo retorne a n_0 e se torne completo. Caso todas as ligações tenham sido usadas, o ciclo Euleriano desejado foi estabelecido. Caso algumas ligações não tenham sido usadas, então seja Φ o ciclo já completo. Porque G é conectado, Φ deve ter passado através de algum nó n_i, o qual é o nó terminal de algumas ligações até agora não usadas. Se todas as ligações usadas por Φ são removidas, então o grafo resultante ainda terá grau par em todos os nós, por que Φ deve usar um número par de ligações adjacêntes em cada nó (zero é considerado um número par).

Começando novamente de n_i podemos percorrer um ciclo Φ' começando e terminando em n_i. Mais uma vez, se o resto das ligações são usadas por Φ', concluímos o teorema. Uma parte de Φ de n_o a n_i seguido pelo Ciclo Φ' e seguida pela outra parte de Φ de n_i a n_o será o ciclo Euleriano requerido. Se algumas ligações ainda não são usadas, seja o novo ciclo Φ a união de Φ e Φ' descrita acima. Novamente podemos achar um nó n_j encontrado por Φ, o qual é o terminal de algumas ligações ainda não usadas. Então podemos prosseguir formando um novo ciclo Φ', começando de n_j e assim por diante, até que finalmente dessa maneira todas as ligações sejam usadas e um ciclo Euleriano Φ seja obtido. Isto prova que o ciclo requerido pode ser construído em qualquer grafo conexo não direcionado, no qual todos os nós são de grau par, e provê um método para construir tal ciclo.

No grafo da ponte Konigsberg, todos os nós são de grau ímpar; portanto, não existe um ciclo Euleriano nesse grafo.

Podemos notar também que qualquer grafo deve ter um número par de nós de grau ímpar. Este resultado é baseado no número total de incidências no grafo inteiro. Cada ligação está en contacto com

dois nós; então a soma de grau de todos os nós é:

$$\sum_{i=1}^{n} d_i = 2m$$

onde m é o número total de ramos e d_i é o grau de nó i. No caso em que alguns nós sejam de grau par e alguns outros sejam de grau ímpar, podemos escrever

$$\sum_{i=1}^{n} d_i = \sum_{i\varepsilon P} d_i + \sum_{i\varepsilon I} d_i = 2m$$

onde P é o conjunto dos nós de grau par, e I é o conjunto dos nós de grau ímpar. Obviamente, $\sum_{i\varepsilon P} d_i$ e $2m$ são números pares; portanto, $\sum_{i\varepsilon I} d_i$ também deve ser um número par. Se se somar um grupo de números ímpares e se obter um número par, então o número dos números ímpares deve ser par. Logo, em qualquer grafo há um número par de nós de grau ímpar.

3.2 Caso de um Grafo Direcionado

Circuito Euleriano

Dado um grafo direcionado G, um Circuito Euleriano é um circuito que percorre todos os arcos de G uma vez e somente uma vez.

TEOREMA 3.2 A condição necessária e suficiente para existência de um circuito Euleriano é que o grafo seja conexo e para cada nó o grau de entrada seja igual ao grau de saída.

A prova desse teorema e semelhante à prova do teorema 3.1, e

e também o método de construir um circuito Euleriano de acordo com esse teorema é o mesmo que foi discutido no caso de um grafo não-direcionado. Aqui tentaremos mostrar isto somente com um exemplo. Considere o grafo direcionado da Figura 3.4. De acordo com o teorema 3.2 o grafo contém um circuito Euleriano, porque ele é conexo e para cada nó o grau de entrada é igual ao grau de saída. Um circuito Euleriano nesse grafo pode ser construído na seguinte maneira. A partir de um nó qualquer, por exemplo n_1, começamos a fazer um circuito; se o circuito percorrer todos os arcos, então ele é um circuito Euleriano e terminamos. Se não, seja n_1, n_2, n_6, n_5, n_3, n_1

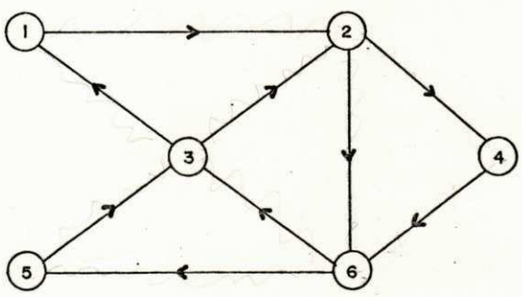


Figura 3.4: Grafo com Circuito Euleriano

o circuito já feito. Novamente a partir de um nó do circuito existente, o qual é o nó terminal de um arco ainda não usado, começamos a fazer um outro circuito usando somente arcos não usados. Como exemplo, a partir de n_2 podemos fazer o circuito n_2, n_4, n_6, n_3, n_2. A parte do circuito anterior de n_1 a n_2 seguido por novo circuito e seguido pela outra parte do circuito anterior de n_2 a n_1 será o circuito Euleriano requerido nesse grafo. Então, um circuito Euleriano nesse grafo pode ser obtido pela sequência de nós n_1, n_2, n_4, n_6, n_3, n_2, n_6, n_5, n_3, n_1. Portanto, em qualquer grafo que satisfaça as condições do teorema 3.2, podemos construir um circuito Euleriano.

3.3 Caso de um Grafo Misto

A existência de um circuito Euleriano num grafo misto é garantida pelo seguinte teorema:

TEOREMA 3.3 [9] A condição necessária e suficiente para existência de um circuito Euleriano num grafo misto $G = (N,A)$ é que:

(a) O grafo seja conexo;

(b) O número total de ramos incidentes em cada nó do grafo seja par;

(c) Para cada nó, e cada subconjunto de nós, o número de ramos não direcionados incidentes no nó (ou subconjunto) seja maior do que, ou igual à diferença em número entre ramos direcionados de entrada e de partida.

As primeiras duas condições são requisitos familiares, a terceira é algo diferente. A necessidade de condição (c) pode ser considerada por um exemplo. Isto é mostrado na Figura 3.5. O grafo nessa figura é conectado e todos os nós são de grau par. Contudo, se considerarmos a região do grafo definido pelos nós 4, 5 e 8, notamos que é exigido entrar 5 vezes nessa região (por cinco ramos direcionados) e só existem 3 ramos (um direcionado e dois não direcionados) disponíveis para sair dessa região. Portanto, é claramente impossível fazer um circuito Euleriano nesse grafo, e concluímos que a condição (c) é necessária.

A prova de suficiência é baseada no teorema de circulação de fluxo em redes [9]. Considere-se um grafo $G = (N,A)$ e supondo que c e ℓ sejam limites superior e inferior de capacidade dos ra

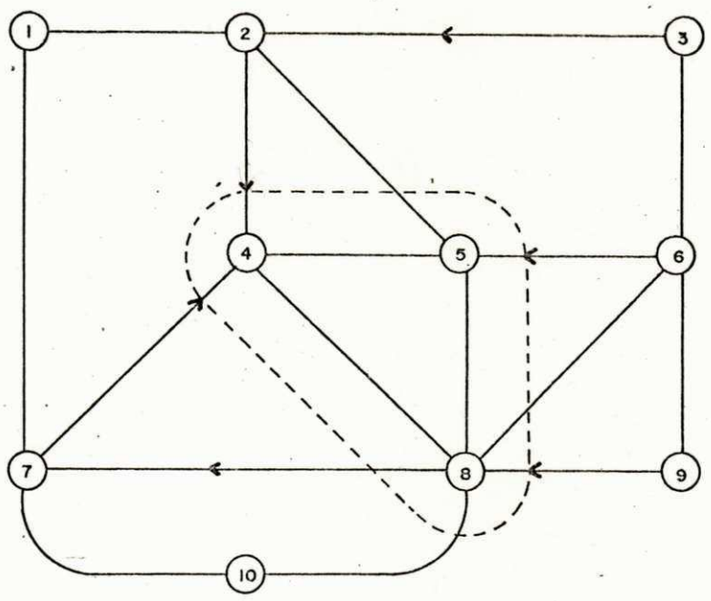


Figura 3.5 Um grafo misto sem circuito Euleriano

mos em A onde $0 \leq \ell \leq c$, uma circulação viável em (N,A) é uma função f em A satisfazendo

$$(3.1)\quad f(i,N) - f(N,i) = 0\ , \qquad i \in N$$

$$(3.2)\quad \ell(i,j) \leq f(i,j) \leq c(i,j) \qquad (i,j) \in A.$$

Isto significa que uma circulação viável satisfaz os limites superior e inferior da capacidade dos ramos e para cada nó o fluxo de entrada é igual ao fluxo de saída. Também definimos X como subconjunto em N e $\bar{X}$ seu complemento.

O teorema de circulação é:

TEOREMA 3.4 A condição necessária e suficiente para existência de uma circulação viável, quando $0 \leq \ell(i,j) \leq c(i,j)$ é que:

$$(3.3)\quad c(X,\bar{X}) \geq \ell(\bar{X},X) \text{ para todos } X \subset N$$

Não provamos esse teoremas, porque é fora do assunto da tese.

A interpretação de (3.3) fica clara com o exemplo da figura 3.6. Nesse exemplo, o fluxo máximo que pode fluir de X para $\bar{X}$, são 4 unidades de fluxo, enquanto o fluxo mínimo que deve sair de $\bar{X}$ para X, são 5 unidades (2+3); logo não pode existir uma circulação viável para esse exemplo

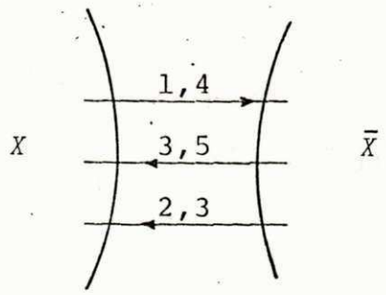


Figura 3.6 A interpretação da relação (3.3)

Voltamos para a prova de suficiência do teorema 3.3. Primeiramente substituimos cada ramo não direcionado de G por um par de arcos contrariamente direcionados, obtendo um grafo direcionado $G_1 = (N, A_1)$. Definimos limites superior e inferior de capacidade para os arcos (i,j) em A_1 por:

$c(i,j) = 1$ para todos $(i,j) \in A_1$

$\ell(i,j) = \begin{cases} 1 & \text{Se } (i,j) \text{ é um arco direcionado de } A; \\ 0 & \text{Caso contrário.} \end{cases}$

Então, a hipótese (c) do teorema 3.3 é equivalente à (3.3) do teorema de circulação; portanto, há uma circulação viável f em G_1. Agora, conforme essa circulação, orientamos alguns ramos não direcionados de G de acordo com o seguinte:

Se (i,j) é um ramo não direcionado de G e se $f(i,j) = 1$, $f(j,i) = 0$, direcionamos o ramo de i para j. Disto resulta um grafo misto $G_2 = (N, A_2)$, o qual tem propriedades (a), (b) do teorema 3.3

e também

(C'): em cada nó o número dos ramos direcionados que entram no nó é igual ao número dos ramos direcionados que saem do mesmo.

Agora, basta estabelecer que em G_2 existe um circuito Euleriano. De acordo com (b) e (c'), o número dos ramos não direcionados incidentes em cada nó deve ser par. Isto significa que os ramos não direcionados de G_2 formam ciclos. Direcionando esses ciclos e direcionando os ramos de acordo com a direção dos ciclos, o grafo G_2 se torna totalmente direcionado. Então, conforme o teorema 3.2 existe um circuito Euleriano no G_2 e concluímos o teorema.

Deve ser observado que, se um grafo misto satisfaz as condições (a) e (b), mas não existe uma circulação viável, isto significa que o grafo não tem circuito Euleriano. Mas, se existir, a prova da suficiência do teorema provê um método para construir tal circuito.

Voltamos para o problema de distribuição de serviços públicos (coleta de lixo, distribuição de cartas etc) discutido no capítulo I. O problema é o seguinte. Dada uma rede de ruas (um bairro), representada por um grafo começando de um nó na rede (depósito de lixo, correio etc) encontrar um roteiro que percorre todas as ruas pelo menos uma vez e retorne para o ponto inicial, de modo que o comprimento total do roteiro seja minimizado. O teorema de Euler garante que num grafo conexo ou rede de ruas que satisfaça as condições necessárias do teorema, existe um roteiro (um ciclo ou circuito Euleriano) que percorre todas as ruas exatamente uma vez. Porque o problema de distribuição de serviços públicos também exige pelo menos uma passagem em cada rua, então, nesse caso, o comprimento

do ciclo (ou circuito) é mínimo. Mas, se o grafo não satisfizer as condições, não existe um ciclo Euleriano e qualquer roteiro que percorre cada rua pelo menos uma vez, percorrerá algumas ruas mais que uma vez. Se desenhamos as ruas repetidas como ramos adicionais no grafo, o grafo resultante vai possuir um ciclo (ou circuito) Euleriano e podemos fazer um roteiro que percorre todos os ramos (incluíndo os ramos adicionais) exatamente uma vez.

Então, nosso problema é começar com um grafo que não tem um ciclo Euleriano e adicionar (duplicar) ramos de modo que o mesmo possua um ciclo (ou circuito) Euleriano, de tal maneira que o comprimento (custo de percurso) dos ramos adicionais seja minimizado. O teorema de Euler garante que poderemos deteminar um roteiro no novo grafo, o qual percorre todos os ramos (incluíndo os duplicados) exatamente uma vez e retorne ao ponto inicial. Porque o custo de percurso dos ramos adicionais é mínimo, o custo de percurso do nosso roteiro também o é. Esse problema é conhecido como "o problema do Carteiro Chinês".

CAPÍTULO IV

O PROBLEMA DO CARTEIRO CHINÊS

Consideramos neste Capítulo o problema: "achar um ciclo (ou circuito) que percorre todos os ramos de um grafo pelo menos uma vez de tal modo que a distância (custo de percurso) total seja minizada", conhecido como "o problema do carteiro Chinês". Esse problema é considerado em três casos diferentes: no caso de um grafo não-direcionado, direcionado e misto.

4.1 Caso de um Grafo Não-Direcionado

Tratamos nesta seção do problema do carteiro Chinês no caso de um grafo não direcionado. O teorema de Euler no Capítulo anterior afirmou que, num grafo conexo não direcionado, que não tem nenhum nó de grau ímpar, existe um ciclo Euleriano que percorre todas as ligações do grafo exatamente uma vez. Porque um ciclo Euleriano não usa nenhuma ligação mais que uma vez, então podemos concluir que o ciclo é uma solução ótima para o problema do carteiro Chinês nesse grafo.

Se alguns nós do grafo são de grau ímpar, então qualquer ciclo que percorre todas as ligações do grafo pelo menos uma vez, per

correrá algumas ligações mais que uma vez. O problema nesse caso é minimizar o comprimento total das ligações duplicadas.

4.1.1 O Algorítmo de KWAN

O primeiro trabalho publicado para resolver este problema de minimização foi por Mei-ko Kwan (1962) que era interessado em determinar roteiros de carteiros. O trabalho de Kwan é baseado na descoberta de ciclos negativos [17]. O algorítmo de Kwan pode ser resumido nos seguintes passos:

1. O grafo não direcionado $G = (N,A)$, que tem peso w_{ij} associado a cada ligação a_{ij} (que representa o custo de percurso) é dado. Se o grafo tiver alguns nós de grau ímpar, acrescentem-se algumas ligações de modo que todos os nós se tornem de grau par e adicionalmente associe-se peso $-w_{ij}$ a cada ligação adicionada a_{ij}, onde, w_{ij} é o peso da ligação a_{ij} no grafo original.

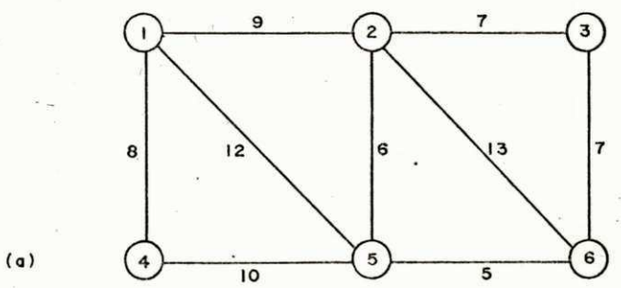
2. Examinemos qualquer ciclo no grafo. Se encontrarmos um ciclo negativo (i.e. um ciclo de valor negativo), removam-se todas as ligações adicionais e dupliquem-se todas as ligações não duplicadas do ciclo; associem-se pesos negativos às novas ligações duplicadas da mesma maneira que foi descrito acima.

3. Voltemos para o passo 2 até que não existam mais ciclos negativos.

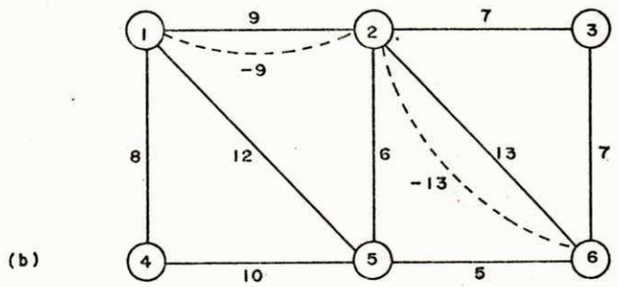
4. Construamos um ciclo Euleriano no grafo resultante de acordo com o teorema de Euler.

Como exemplo, na figura 4.1(a) é mostrado um grafo no qual os nós 1 e 6 são de grau ímpar. Se duplicarmos os ramos (1,2) e (2,6),

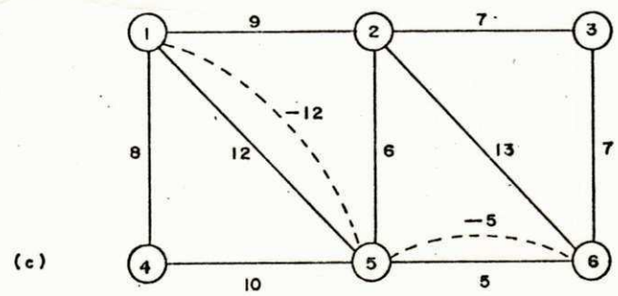
no grafo resultante todos os nós serão de grau par. Esse grafo é mostrado na figura 4.1(b). Agora nesse grafo, se considerarmos o ciclo (1, 2, 6, 5, 1), a soma dos pesos é -5; então, ele é um ciclo negativo. Portanto, nesse ciclo as ligações duplicadas devem ser trocadas; o novo grafo é mostrado na figura 4.1(c) e ele não tem mais um cuclo negativo. Isto significa que não existem mais cadeias de valor menor que a cadeia já duplicada; então, o peso total das ligações duplicadas é mínimo e consequentemente um ciclo Euleriano nesse grafo tem o peso mínimo.



(a)



(b)



(c)

Figura 4.1 O exemplo do algorítmo de KWAN

O algorítmo do KWAN tem um problema sério, porque geralmente o número de ciclos numa rede é extremamente grande; então, a inspeção de todos os ciclos, para assegurar que não há mais ciclos negativos, é praticamente impossível.

Por causa do trabalho de KWAN, o problema de achar um ciclo de comprimento mínimo envolvendo todas os ramos num grafo conexo é chamado "o problema do carteiro Chinês".

4.1.2 O Conceito de Caminho Mais Curto

Uma outra maneira de se considerar o problema do carteiro Chinês é encontrada no trabalho de EDMONDS (1965) [6] e GLOVOR (1967) [11]. Num grafo em que existem alguns nós de grau ímpar, o que é requerido é a duplicação de ligações apropriadas, de modo que o grafo possua um ciclo Euleriano. Isto implica que cada nó de grau ímpar deve converter-se em um de grau par, mantendo o grau dos nós restantes. Claramente, quando duplicamos uma só ligação entre dois nós, trocamos o grau de ambos os nós. Se originalmente ambos os nós são de grau ímpar, os dois torman-se de grau par. Mas, se um nó é de grau ímpar e o outro de grau par, duplicando a ligação entre eles simplesmente se troca a propriedade de par e ímpar. Então, o processo de duplicação deve ser continuado até que termine num nó que era originalmente de grau ímpar. A figura 4.2 mostra tal processo entre dois nós que originalmente eram de grau ímpar.

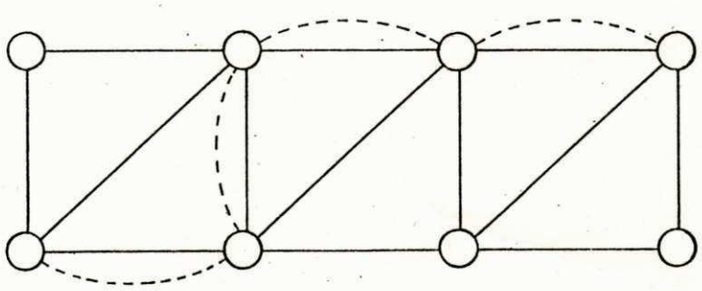


Figura 4.2

Uma propriedade adicional de uma boa solução pode ser descoberta por inspeção da Figura 4.3. Se o caminho duplicado entre dois nós de grau ímpar não é o caminho mais curto diponível entre aqueles nós, podemos melhorar o custo de percurso no grafo por eliminação das ligações duplicadas e duplicar ligações no caminho mais curto. Então, em qualquer solução ótima, isto deve ser verdadeiro: que o caminho onde houve duplicações, seja o caminho mais curto ligando os nós de grau ímpar. No exemplo da Figura 4.3, os nós 1 e 6 eram originalmente de grau ímpar, portanto o caminho (1, 2, 3, 4, 5, 6) entre eles foi duplicado. Mas o caminho do nó 1 ao nó 6 via nó 7 claramente é mais curtò do que o que foi duplicado.

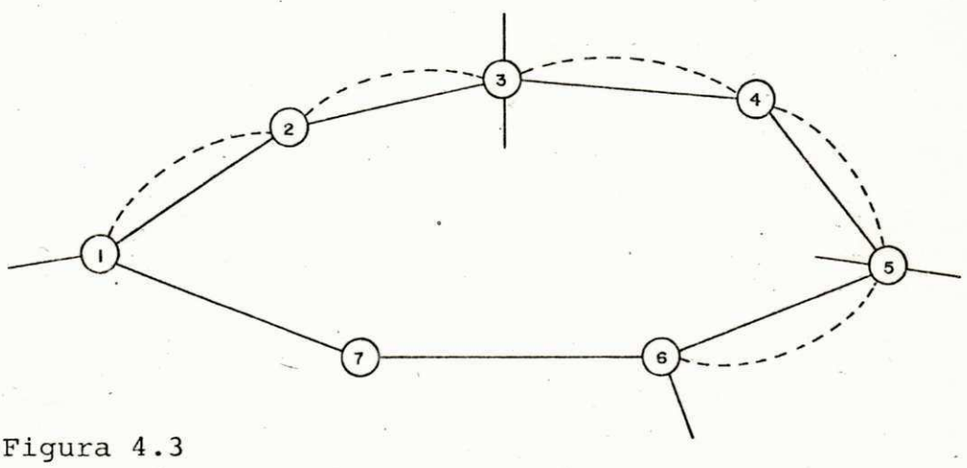
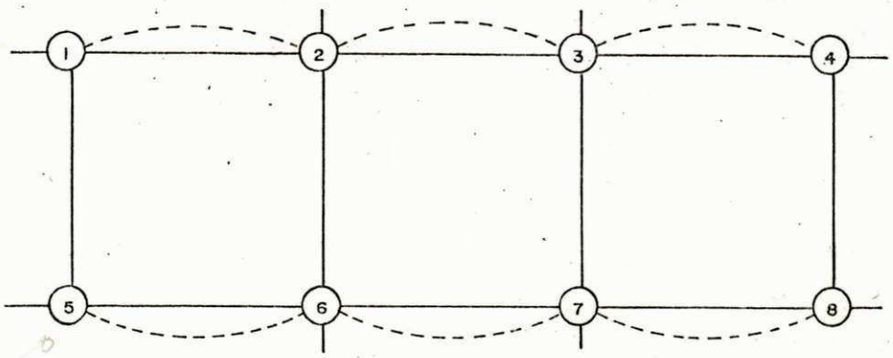


Figura 4.3

Quando se fazem duplicações entre dois nós de grau ímpar, nem sempre essa duplicação é ótima. Depende de quais pares de nós de grau ímpar são escolhidos para serem conectados. Este fato é demonstrado pela Figura 4.4, onde os nós de grau ímpar são os nós 1, 4, 5 e 8. As ligações duplicadas na Figura 4.4(a) são os caminhos mais curtos, respectivamente, conectando o nó 1 com o 4 e o nó 5 com o 8. Claramente as duplicações não são ótimas e as ligações duplicadas na Figura 4.4(b), as quais conectam o nó 1 com o 5 e o nó 4 com o 8,

são melhores. Portanto, não somente devemos conectar os nós de grau ímpar pelos caminhos mais curtos, mas devemos também selecionar os pares corretos de nós de grau ímpar para a conecção.

(a)



(b)

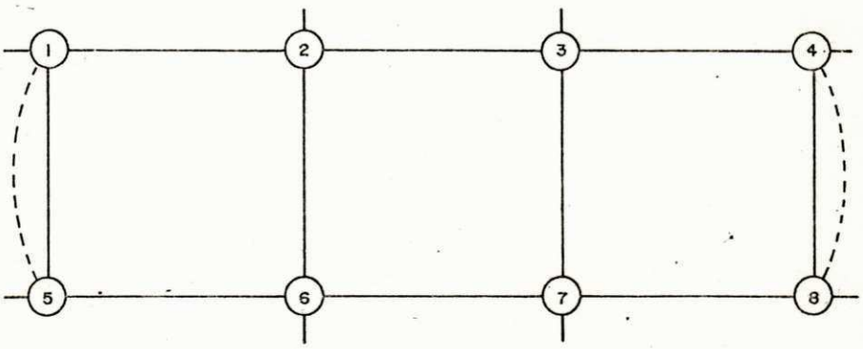


Figura 4.4: Dois conjuntos de caminhos mais curtos.

Este último ponto matemáticamente é o ponto mais difícil do problema. Relativamente é fácil achar o caminho mais curto entre dois nós de um grafo; há um número de algorítmos, já implementados e extremamente eficientes para achar a solução de tal problema [4,13]. O problema de acasalar os nós a serem conectados por caminhos mais curtos, envolve um número grande de permutações. Por exemplo, num grafo que tem 20 nós de grau ímpar o número de conjuntos de caminhos mais curtos entre os nós de grau ímpar, é aproximadamente $6{,}5 \times 10^8$ (isto é 3 x 5 x 7 x .... x 19). O grafo da Figura 4.4 tem 3 de tais conjuntos, dois dos quais são mostrados.

4.1.3 A Solução de Acoplamento

O desenvolvimento de uma técnica que proveu uma solução ótima para este problema é devido a EDMONDS [6]. EDMONDS era interessado no problema de grafo conhecido como acoplamento [5]. O problema do acoplamento é achar em qualquer grafo, se existe um conjunto de ligações tal que cada nó do grafo esteja em contato por exatamente uma das ligações do conjunto e a soma dos comprimentos das ligações seja mínima (ou máxima). Ele desenvolveu um algorítmo muito elegante para a solução deste problema.

Na aplicação direta do algorítmo de EDMONDS à solução do problema do carteiro Chinês requer uma transformação do grafo. O algorítmo de EDMONDS determina casais dentre todos os nós, de tal forma que a soma dos comprimentos das ligações que os conectam é mínima. O problema do carteiro Chinês exige somente casais de nós de grau ímpar, que podem ser conectados por cadeias, e não necessariamente por ligações únicas. A transformação requerida é mostrada na Figura 4.5. A Figura 4.5(a) é o grafo original, com quatro nós de grau ímpar. A Figura 4.5(b) é um grafo mostrando somente os de grau ímpar. As ligações entre os nós na Figura 4.5(b) representam caminhos mais curtos entre os nós de grau ímpar. Então, por exemplo, a ligação entre nós 5 a 10 na Figura 4.5(b) representa o caminho do nó 5 ao nó 10 via nó 6 no grafo original e o seu peso representa o peso desse caminho. A solução do problema do acoplamento no grafo da Figura 4.5(b) são as duas ligações mostradas por linhas grossas, conectando o nó 5 com o 2 e nó 10 com o 8. Isto implica em que as ligações duplicadas requeridas no grafo da Figura 4.5(a) são as seguintes: O caminho de 5 a 2 via 1 e o caminho de 10 a 8 via 11. O problema do acoplamento é discutido no capítulo 5.

Portanto, a solução do problema de acoplamento pode prover

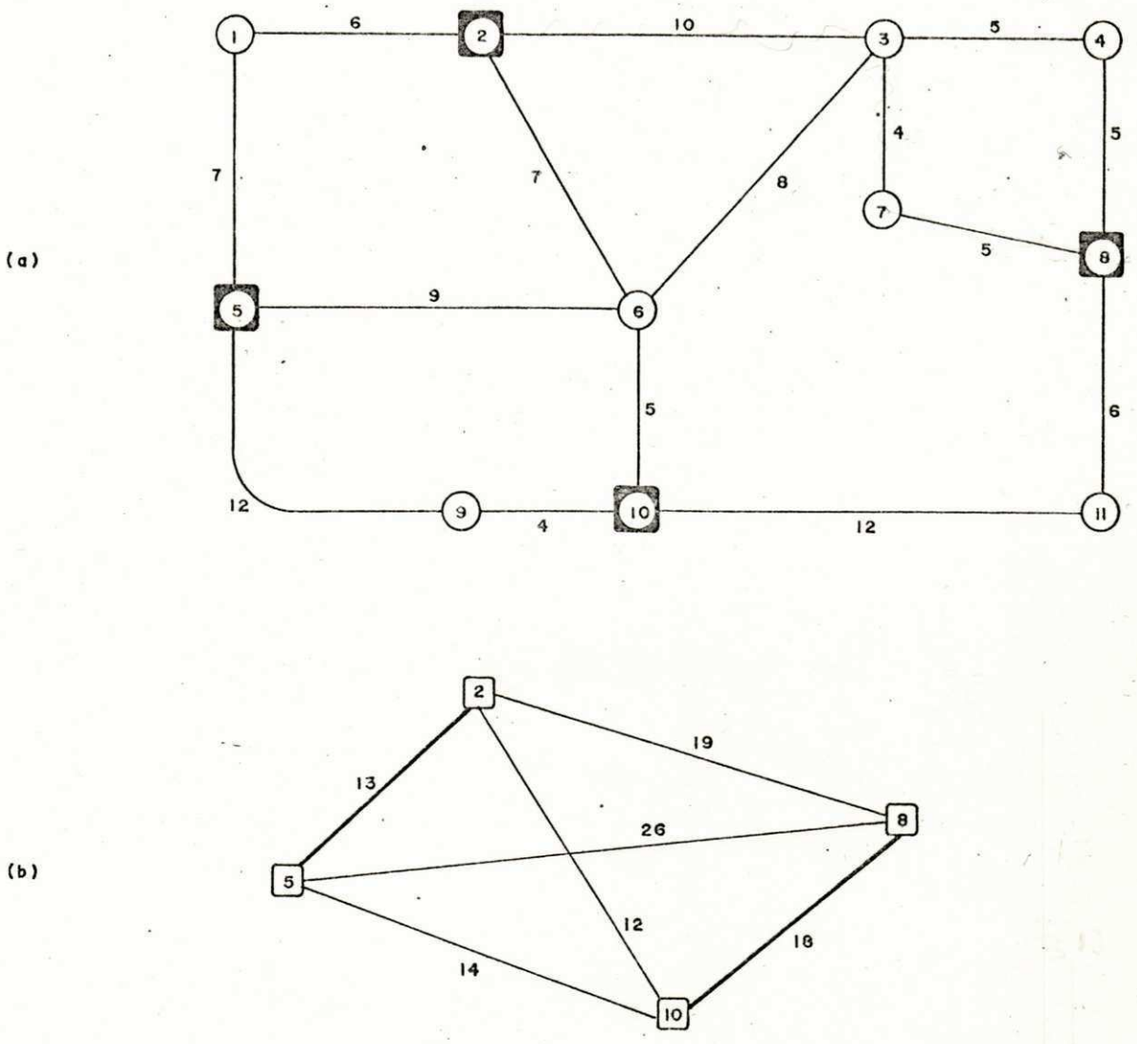


Figura 4.5: Grafo Original e Grafo derivado de nós de Grau Ímpar.

uma solução para o problema de carteiro Chinês, no qual ele determina a construção de um grafo que possui um ciclo Euleriano por duplicação de ligações em caminhos conectando nós de grau ímpar de tal maneira que a soma dos comprimentos das ligações duplicadas é mínima. Do teorema de Euler sabemos que um ciclo Euleriano pode ser construído no grafo resultante. Como uma consequência, esse ciclo será de comprimento mínimo (custo de percurso mínimo). O algorítmo do problema do carteiro Chinês pode ser resumido nos seguintes passos:

PASSO 0 (Início)

O grafo G que representa a rede de ruas, é dado: A cada ligação de

G um peso (comprimento de rua ou custo de percurso) é atribuído.

PASSO 1 (Identificação de Nós de grau Ímpar)
Identifiquem-se os nós de grau ímpar em G; se não houver nenhum, vá-se ao passo 4.

PASSO 2 (Caminhos mais Curtos)
Encontrem-se os caminhos mais curtos entre todos os pares de nós de grau ímpar.

PASSO 3 (Acoplamento com Peso)
Faça-se o grafo transformado G^*, no qual os nós são somente aqueles de grau ímpar do grafo G e as ligações representam os caminhos mais curtos entre eles [4,13]. Usando o algoritmo de EDMONDS, encontre-se o acoplamento com peso mínimo em G^*. Dupliquem-se ligações apropriadas do G de acordo com o acoplamento ótimo obtido. (veja o capítulo 5).

PASSO 4 (Construção do Ciclo)
Conforme o teorema 3.1, faça-se um ciclo Euleriano em G [7].

4.2 Caso de um Grafo Direcionado

O problema do carteiro Chinês no caso de um grafo direcionado torna-se mais simples que no não-direcionado e o mesmo pode ser resolvido como um problema de fluxo de custo mínimo numa rede.

No caso de um grafo direcionado, a condição necessária e suficiente para a existência de um circuito Euleriano é que o grafo seja conexo e para cada nó o grau de entrada seja igual ao grau de saída. Se a rede original de ruas não satisfaz esta condição, então alguns arcos devem ser duplicados de modo que o grafo resultante

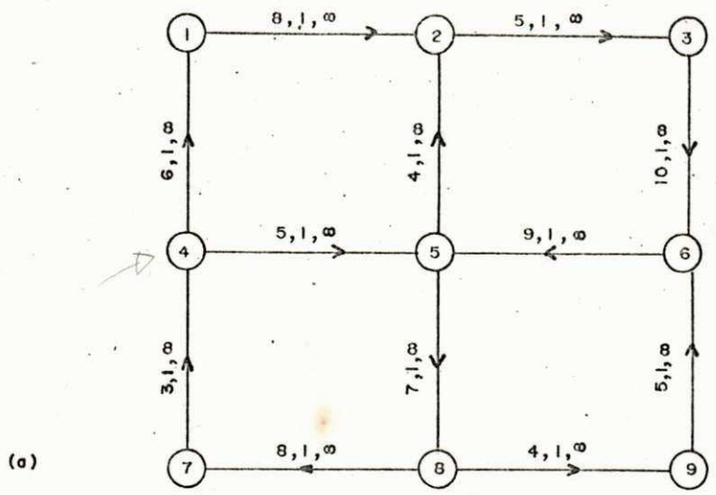
possua um circuito Euleriano. Os arcos duplicados devem formar caminhos mais curtos entre nós de grau ímpar, obdecendo à direção dos arcos, começando de nós com um excesso de entrada e terminando em nós com um excesso de saída.

4.2.1 O Algoritmo de Out-of-Kilter

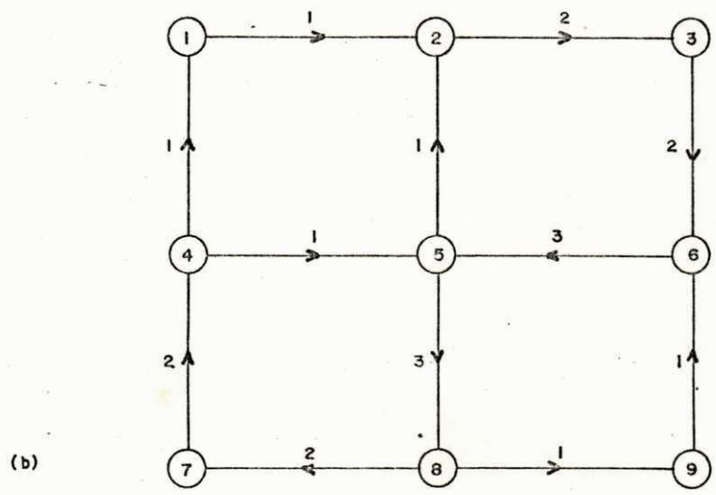
Um algoritmo que pode ser usado para este problema, é o algoritmo de "Out-of-Kilter" [9,14]. Considere-se uma rede direcionada na qual os arcos representam tubos, onde três números são associados a cada arco: um custo associado a cada unidade de fluxo, um limite superior para o fluxo (capacidade), e um limite inferior. O limite inferior pode ser zero e o limite superior pode ser infinito. Dadas essas quantidades, o algoritmo de Out-of-Kilter encontra fluxo em todos os tubos, de tal maneira que os limites superior e inferior em fluxo sejam satisfeitos: o fluxo de entrada seja igual ao fluxo de saída em cada nó, e o custo total do fluxo seja minimizado. É claro que, se o limite inferior em todos tubos for zero, a solução ótima terá fluxo zero em todos os tubos. Mas, se houver qualquer limite inferior positivo em fluxos, esses limites podem forçar fluxos em outros tubos.

Para resolver o problema de carteiro Chinês no caso direcionado, representamos a rede original de ruas por um grafo direcionado. O custo por unidade de fluxo em cada arco é o seu custo de percurso, o limite inferior em todos os arcos sendo a unidade e todos os limites superiores são infinitos. Então o algoritmo de Out-of-Kilter encontrará fluxos em todos os arcos, tais que o fluxo em qualquer arco seja maior que ou igual a um e os fluxos de entrada sejam iguais aos fluxos de saída em cada nó. Interpretamos o fluxo num arco como o número de vezes que o arco deve ser percorrido; fluxo maior que um implica passagem mais que uma vez. Portanto, o

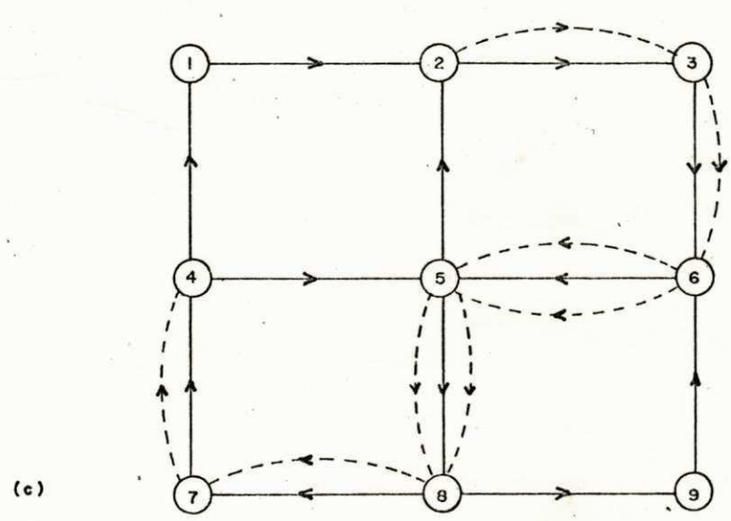
algoritmo de Out-of-Kilter determina duplicações apropriadas dos arcos de modo que o grafo resultante possua um circuito Euleriano e o custo total de percurso seja minimizado. A Figura 4.6(a) mostra um grafo direcionado: três números são associados a cada arco, o primeiro é o custo por unidade de fluxo, o segundo é o limite inferior e o terceiro é o limite superior. O fluxo ótimo que o algoritmo de Out-of-Kilter encontra para esse problema, é mostrado na Figura 4.6(b). A Figura 4.6(c) é a interpretação dos fluxos da Figura 4.6(b), esse grafo possui um circuito Euleriano cujo custo total de percurso é mínimo.



(a)



(b)



(c)

Figura 4.6: Exemplo do algoritmo de Out-of-Kilter

4.2.2 A Solução de Acoplamento Bipartido

Uma outra técnica de solução do problema do Carteiro Chinês no caso direcionado é a seguinte. Considere-se um grafo direcionado, como foi descrito anteriormente. A condição necessária e suficiente para a existência de um circuito Euleriano é que o grafo seja conexo e para cada nó o grau de entrada seja igual ao grau de saída. Se a rede original de ruas não satisfizer esta condição, então alguns arcos devem ser adicionados de modo que o grafo resultante possua um circuito Euleriano. Os arcos adicionados devem formar caminhos mais curtos entre nós com excesso de entrada e nós com excesso de saída. Chamemos S o conjunto de todos os nós que têm excesso de entrada, e T o conjunto de todos nós que têm excesso de daída; então, cada nó do conjunto S deve ser conectado com um nó do conjunto T através do caminho mais curto entre eles. Mas, para uma solução ótima, isto é suficiente. Também casais de nós (um nó do conjunto S e um nó do conjunto T) devem ser selecionados de modo que a soma dos comprimentos dos caminhos mais curtos entre eles seja minimizada. Este último ponto é semelhante ao problema do acoplamento de peso que foi descrito no caso de grafo não-direcionado. Obviamente, no caso direcionado, um nó do conjunto S não pode ser conectado a um outro nó de S (também válido para o conjunto T). Cada nó de conjunto S somente pode ser conectado com um nó de conjunto T. Então o problema de acoplamento nesse caso torna-se como um problema de acoplamento bipartido que é muito conhecido na teoria dos grafos e é denominado "O Problema de Designação" [9]. A solução deste problema provê uma solução ótima para o problema do carteiro Chinês no caso de um grafo direcionado.

O algoritmo do problema do carteiro Chinês no caso de um grafo direcionado pode ser resumido nos seguintes passos:

PASSO 0 (Início)

O grafo direcionado G é dado. A cada arco de G um peso (que representa o custo de percurso) é atribuido.

PASSO 1 (Formar os Conjuntos S e T)

Defina-se o conjunto S de nós com excesso de entradas; se um nó tiver mais que um excesso de entrada, ele deve ser repetido no conjunto S igual ao seu número de excesso para que possamos aplicar acoplamento bipartido. Faça-se o mesmo também para o conjunto T.

PASSO 2 (Caminhos Mais Curtos)

Encontrem-se os caminhos mais curtos entre cada nó do conjunto S e todos os nós do conjunto T.

PASSO 3 (Acoplamento Bipartido com Peso)

Faça-se um grafo bipartido completo $G^* = (S,T,A)$ no qual S e T são os conjuntos identificados no PASSO 1 (incluidos os nós repetidos); os arcos representam caminhos mais curtos entre nós de conjunto S e nós de conjunto T. Usando o algoritmo de EDMONDS (veja o capítulo 5) ou qualquer algoritmo de designação, encontre-se o acoplamento com peso mínimo em G^*. Dupliquem-se arcos apropriados de G de acordo com o acoplamento ótimo obtido.

PASSO 4 (Construção de Circuito)

Conforme o teorema 3.2, faça-se um circuito Euleriano em G.

Como exemplo, considere-se o grafo da Figura 4.7, no qual os nós 2, 5 e 13 têm excesso de entrada, o nó 6 tem um excesso de saída e o 11, dois. O grafo bipartido feito com esses nós é mostrado na Figura 4.8. Como foi descrito no algoritmo, o nó 11 é duplicado, porque no grafo original existem dois excessos de saída nesse nó. Um acoplamento de peso mínimo é mostrado por linhas grossas nesse

grafo, no qual os nós 2 e 13 são acoplados com o 11 (o nó 11 é du

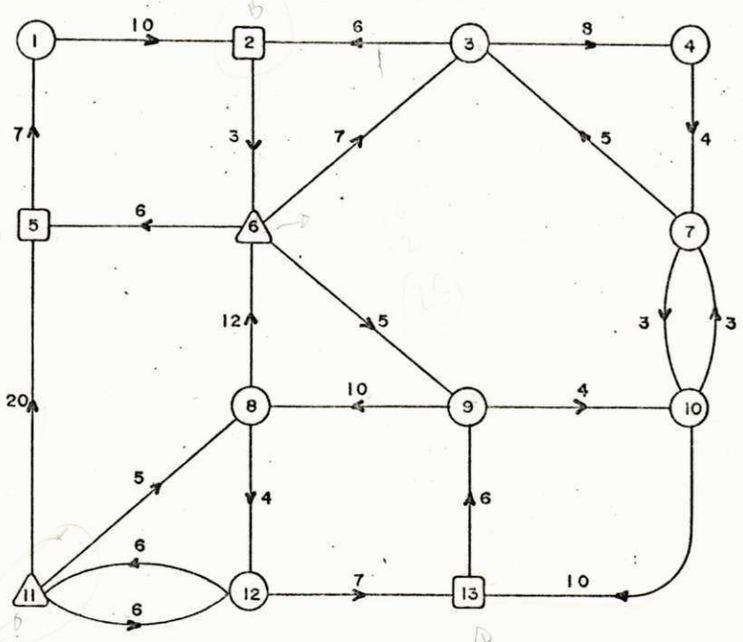


Figura 4.7: O grafo original

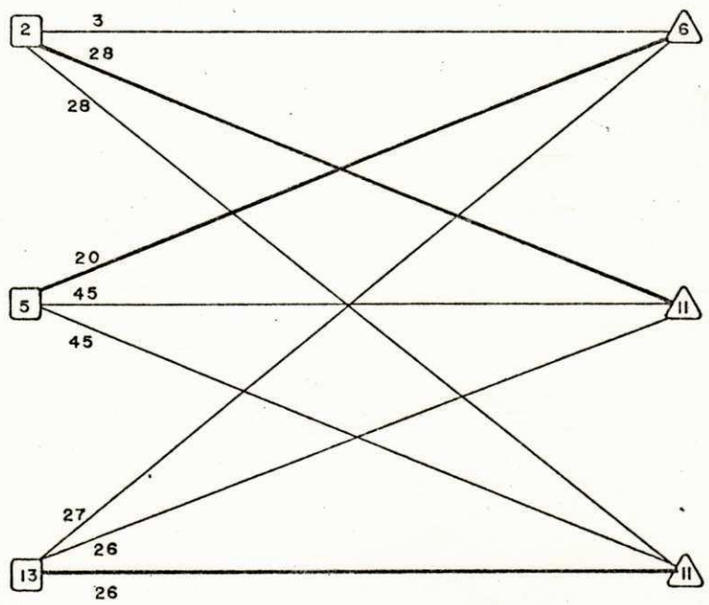


Figura 4.8: Acoplamento com peso mínimo

plicado), e o nó 5 com o 6. Isto implica em que o caminho mais curto entre o nó 2 e 11 no grafo original deve ser duplicado e assim por diante. O grafo com duplicações apropriadas é mostrado na Figura 4.9; esse grafo possui um circuito Euleriano.

O problema de acoplamento bipartido será discutido no capitulo 5 e veremos que esse problema é um caso particular do problema de acoplamento num grafo não-bipartido e que o mesmo algoritmo de acoplamento não bipartido pode resolver esse problema. Portanto, o presente algoritmo do problema no caso direcionado não necessita ser totalmente programado, pois os programas implementados para o caso não-direcionado podem ser usados como sub-rotinas para resolver o problema no caso direcionado.

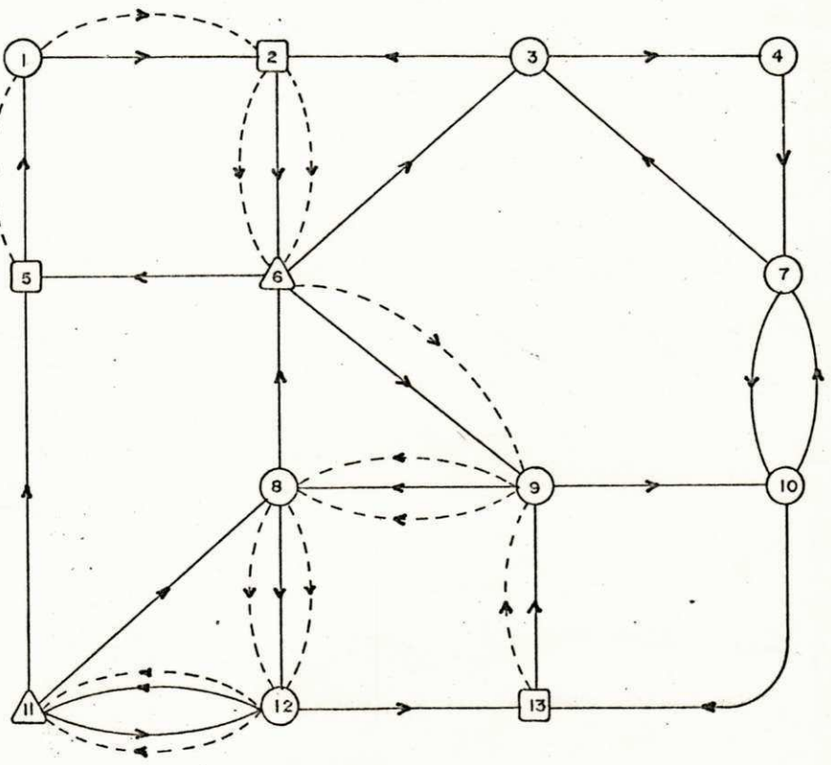


Figura 4.9: O grafo com duplicações apropriadas.

4.3 Caso de um Grafo Misto

O problema do carteiro Chinês no caso de um grafo misto torna-se muito complicado. Isto é principalmente devido às condições necessárias e suficientes para existência de um circuito Euleriano que foram citadas no capítulo 3. Na realidade, só testar para ver se existe um circuito Euleriano num grafo misto, exige resolver um problema de fluxo em redes.

O problema de adicionar alguns ramos a um grafo misto geral de modo que o grafo resultante possua um circuito Euleriano de custo de percurso mínimo ainda não está resolvido. Para grafos mistos pequenos, o problema do carteiro Chinês pode ser resolvido com um problema de programação linear inteira. Dado um grafo misto $G = (N,A)$, podemos formular o problema como o seguinte:

$$(4.1) \quad Min \sum_{(i,j)\in D} X_{ij}\, C_{ij} + \sum_{(i,j)\in U} (X_{ij} + X_{ji})\, C_{ij}$$

$$(4.2) \quad X_{ij} \geq 1 \qquad \forall (i,j)\in D$$

$$(4.3) \quad X_{ij} + X_{ji} \geq 1 \qquad \forall (i,j)\in U$$

$$(4.4) \quad \sum_{(i,j)\in A} X_{ij} - \sum_{(j,k)\in A} X_{jk} = 0 \qquad \forall j\in N$$

$$(4.5) \quad X_{ij} \geq 0 \text{ e inteiro} \qquad \forall (i,j)\in A$$

Onde:

N = O número de nós do grafo

A = O conjunto de todos ramos (i,j)

D = O conjunto de ramos direcionados

U = O conjunto de ramos não direcionados $(A = D \cup U)$

X_{ij} = O número das vezes que o ramo (i,j) é percorrido do nó i ao nó j.

C_{ij} = O custo de percurso do ramo (i,j) do nó i para o nó j.

De (4.1) a (4.5) tem-se um problema linear inteiro. Para resolver esse problema podem ser aplicados algoritmos de programação inteira. Como a maior limitação dos modelos de programação inteira é o porte do problema [10], então só problemas pequenos (aproximadamente até 50 ramos) podem ser resolvidos com esses algoritmos.

No caso especial, quando o grafo misto contém somente um ramo direcionado, o algoritmo para grafos não direcionados pode ser aplicado. Isto sempre é possível, porque o ciclo Euleriano determinado por esse algoritmo pode ser percorrido em ambas as direções. Então, se existir somente um ramo direcionado, ainda o problema pode ser resolvido pelo algoritmo do grafo não direcionado. Ainda, se houver vários ramos direcionados separadamente distribuídos num grafo (i.e. entre qualquer dois ramos direcionado não existam cadeias de cardinalidade menor que dois), é aconselhável usar o algoritmo de grafo não direcionado.

CAPÍTULO V

PROBLEMA DE ACOPLAMENTO

5.1 Introdução

5.1.1 Acoplamento com Peso

Considere-se um grafo não direcionado $G = (N,A)$. Um acoplamento é definido como um subconjunto X de A, escolhido de modo que nenhum dos ramos de X seja adjacente (i.e. tenha um mesmo nó em comum). Então o problema do acoplamento com peso pode ser assim definido:

Ache um acoplamento X^* com peso máximo, sendo o peso de um acoplamento X igual a:

$$W_x = \sum_{a_j \in X} W_j$$

onde W_j é o peso associado com o ramo a_j. X^* será chamado O Acoplamento com Peso Máximo. No exemplo da Figura 5.1 o acoplamento com peso máximo é mostrado com linhas grossas, onde o peso total do acoplamento é $W_x = 18$.

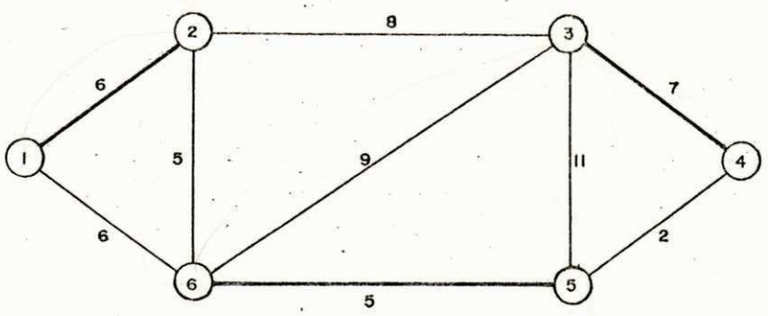


Figura 5.1: Acoplamento com peso máximo

5.1.2 Casos Particulares

Problema de Acoplamento de Cardinalidade Máxima

O problema de acoplamento com peso considerado na seção 5.1.1 é o caso geral. No caso especial, onde todos os pesos dos ramos W_j são unidades, o problema de acoplamento é reduzido ao Problema de Acoplamento de Cardinalidade Máxima. Se um grafo G tiver n nós, obviamente a cardinalidade de um acoplamento em G não pode exceder $n/2$. Ainda este número não é sempre alcançado; por exemplo, o grafo da Figura 5.2 tem um acoplamento de cardinalidade máxima de valor 1.

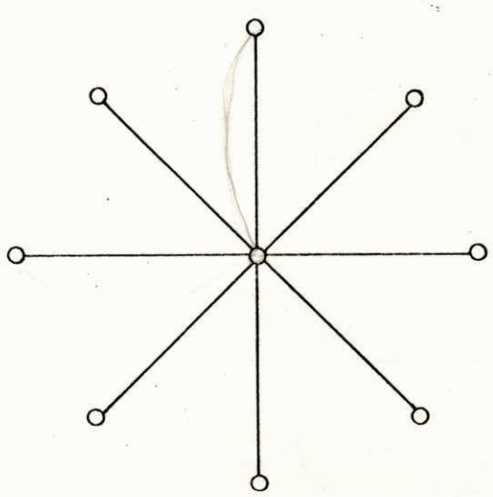


Figura 5.2

Acoplamento num Grafo Bipartido

No caso especial onde o grafo é bipartido, podemos considerar os seguintes problemas:

Acoplamento Bipartido de Cardinalidade Máxima

Dado um grafo bipartido, encontre-se um acoplamento contendo um número máximo de ramos. Um exemplo importante envolvendo acoplamento em grafo bipartido é O Problema de Designação; há m operários disponíveis que devem ser designados a n máquinas ($m \neq n$ no caso geral), cada operário pode trabalhar com algumas máquinas e cada máquina só precisa de um operário. Qual é o maior número de operários que podem ser designados às máquinas?

Considere-se um grafo bipartido $G = (S,T,A)$ onde S corresponde ao conjunto de operários e T ao conjunto de máquinas. Existe uma ligação (i,j) entre S_i e t_j, se o operário S_i pode trabalhar com a máquina t_j. Vê-se que o problema de designação é nada mais que um acoplamento de cardinalidade máxima em G. Considere-se o grafo da Figura 5.3 onde os nós do conjunto S são operários e de T são máquinas; o operário 1 pode trabalhar com a máquina 1 e a máquina 4 e assim por diante. Um acoplamento de cardinalidade máxima (designação ótima dos operários às máquinas) é mostrado com linhas grossas nesse grafo.

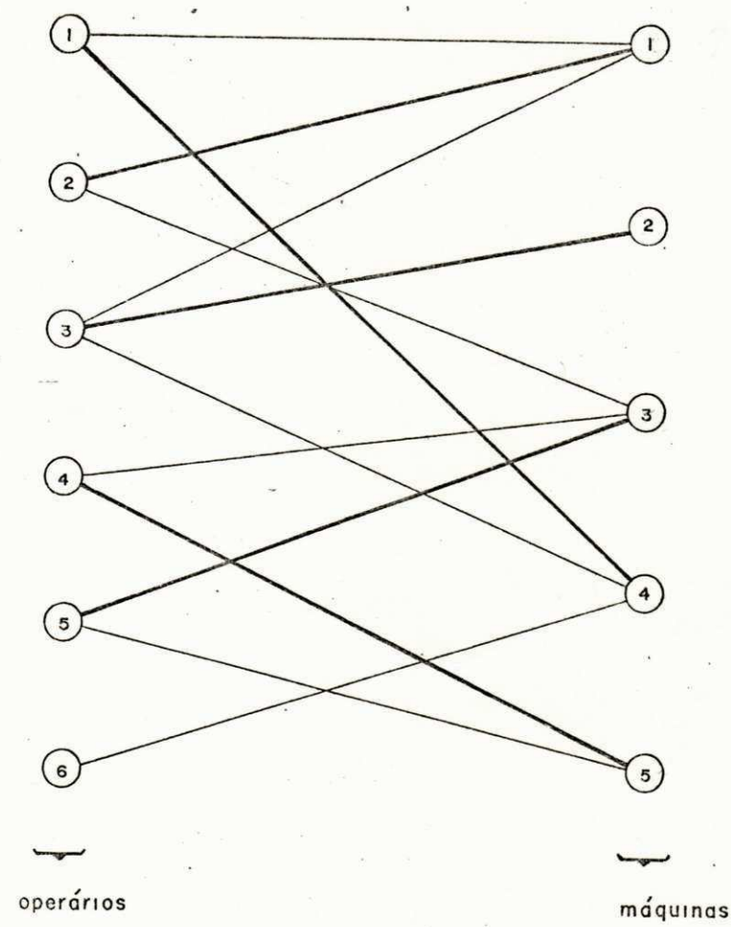


Figura 5.3: Acoplamento de cardinalidade máxima num grafo bipartido.

Problema de Acoplamento Max-Min

Dado um grafo bipartido com pesos associados às ligações, encontre-se um acoplamento de cardinalidade máxima para o qual o menor dos pesos das ligações no acoplamento seja máximo (Este problema também é chamado o problema de "Bottleneck"). Um exemplo comum é o seguinte. Há n operários para serem designados a n tarefas numa linha de produção. Seja W_{ij} a eficiência com a qual o operário i pode realizar a tarefa j. A taxa de produção é limitada pela eficiência do operário mais lento. Que designação dos operários às tarefas maximiza a taxa de produção? Como exemplo, considere-se a matriz da Figura 5.4 que representa um grafo bipartido completo; cada linha da matriz representa um operário e cada coluna uma tarefa. Por exemplo o operário 1 pode realizar as tarefas 1, 2, 3, 4, 5 e 6 respectivamente com taxas de 8, 3, 5, 10, 6 e 7. A designação ótima nesse exemplo é mostrada por círculos.

Operários \ Tarefas	1	2	3	4	5	6
1	8	3	5	(10)	6	7
2	(7)	9	9	5	8	3
3	6	5	2	6	4	(8)
4	5	(7)	10	1	7	5
5	6	5	(8)	6	4	10
6	3	3	12	4	(7)	6

Figura 5.4: Acoplamento Max-Min

Problema de Acoplamento Bipartido com Peso

Dado um grafo bipartido com pesos associados à cada ligação, encontrar o acoplamento para o qual a soma dos pesos das ligações é máximo.

Considere-se um time de futebol de n jogadores que devem ser designados a m posições. Seja W_{ij} a eficiência do jogador i na posição j. Que designação dos jogadores maximiza a eficiência total deles? Este problema pode ser formulado como se segue: Seja $G = (S,T,A)$ um grafo bipartido completo no qual S corresponde ao conjunto de jogadores e T ao conjunto de posições e seja W_{ij} pesos associados à cada ligação que significa a eficiência do jogador i na posição j. Claramente, este problema é nada mais que um problema de acoplamento com peso. A matriz da Figura 5.5 mostra um grafo bipartido; cada linha da matriz representa um jogador e cada coluna, uma posição. O acoplamento com peso máximo nesse grafo é mostrado por círculos.

Jogadores \ Posições	1	2	3	4	5
1	(8)	3	7	2	3
2	4	1	5	6	(7)
3	6	5	7	8	6
4	8	3	(8)	5	2
5	5	9	3	(11)	5
6	5	(7)	8	7	1

Figura 5.5: Acoplamento bipartido com peso

Todos os problemas de acoplamento num grafo bipartido podem ser resolvidos com algoritmos de fluxo em redes. O problema de acoplamento de cardinalidade máxima num grafo bipartido pode ser transformado em um problema de fluxo máximo, e o de acoplamento com peso em um problema de fluxo máximo com custo mínimo. O problema de acoplamento no caso geral não pode ser resolvido pelos algoritmos existentes de fluxo em redes.

No capítulo presente será mostrado um método eficiente de so

lução desse problema no caso geral em que o grafo não é bipartido. A implementação de um programa para este caso é incluído neste trabalho. Esse programa é capaz de resolver o problema de acoplamento em todos os casos particulares já discutidos.

5.2 Acoplamento de Cardinalidade Máxima

5.2.1 Nós Expostos

Dado um acoplamento X, um nó i que não é o nó terminal de qualquer ligação em X, é chamado de exposto. Na Figura 5.6 onde o acoplamento é mostrado por linhas grossas, os nós 6 e 9 são expostos.

5.2.2 Cadeias Alternadas, Cadeias de Aumento

Uma Cadeia Alternada é uma cadeia elementar cujas ligações estão alternadamente em X e não em X. (8, 7, 5, 3, 4, 10, 1) é um exemplo de tal cadeia na Figura 5.6.

Uma Cadeia de Aumento é uma cadeia alternada cujos nós inicial e final são expostos. (9, 10, 4, 1, 2, 3, 5, 6) é um exemplo de tal cadeia no grafo da Figura 5.6.

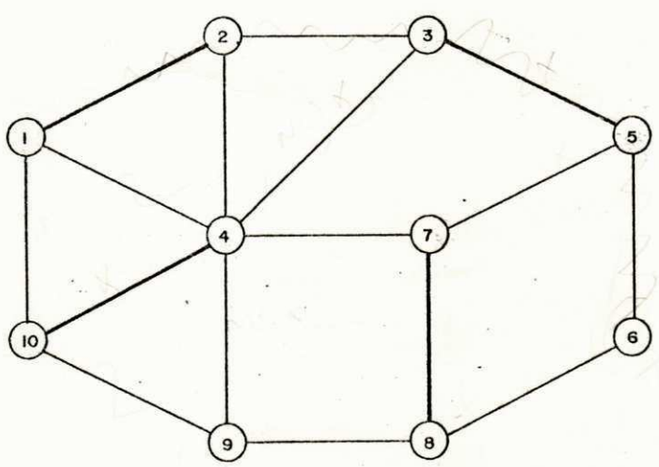


Figura 5.6: Acoplamento X

TEOREMA 5.1 Um acoplamento X é um acoplamento de cardinalidade máxima, se e somente se não existir nenhuma cadeia de aumento em G.

Prova

NECESSIDADE. Suponha-se que existe uma cadeia de aumento P e que P também representa o conjunto de ligações na cadeia. Nenhuma ligação no conjunto $X - P \cap X$ pode ser adjacente com qualquer ligação em P, porque os dois nós terminais de P são expostos (por definição, e todos os outros nós em P são nós terminais de algumas ligações em $P \cap X$. Então, o conjunto de ligações $X' = (X - P \cap X) \cup (P - P \cap X) = X \cup P - P \cap X$ também é um acoplamento. O conjunto X' pode ser formado trocando aquelas ligações em P não pertencentes a X (i.e. $P - P \cap X$) com aquelas ligações em P que estão em X (i. e. $P \cap X$). Porque ambas as ligações extremas de P não estão em X, o conjunto $P - P \cap X$ contém uma ligação a mais que o conjunto $P \cap X$ e, portanto, $|X'| = |X| + 1$, e X não é um acoplamento de cardinalidade máxima.

SUFICIÊNCIA. Seja X um acoplamento que não admite nenhuma cadeia de aumento e seja X^* o acoplamento de cardinalidade máxima. Seja o grafo parcial G_P composto de ligações $X \cup X^* - X \cap X^*$. Nenhum nó de G_P pode ter grau maior que 2, porque isto implicaria em duas ou mais ligações em X (ou X^*) adjacentes, o que viola a definição de um acoplamento. Então G_P é formado de um ou mais componentes conectados, cada um dos quais é, ou um nó isolado, uma cadeia simples, ou um ciclo simples, como é mostrado na Figura 5.7.

Uma cadeia do tipo (b) não pode existir, porque ela é uma cadeia de aumento em relação a X, e é contrário à suposição inicial. Uma cadeia do tipo (c) não pode existir, porque ela é uma cadeia de aumento em relação a X^*, e é contrário à suposição de que X^* é

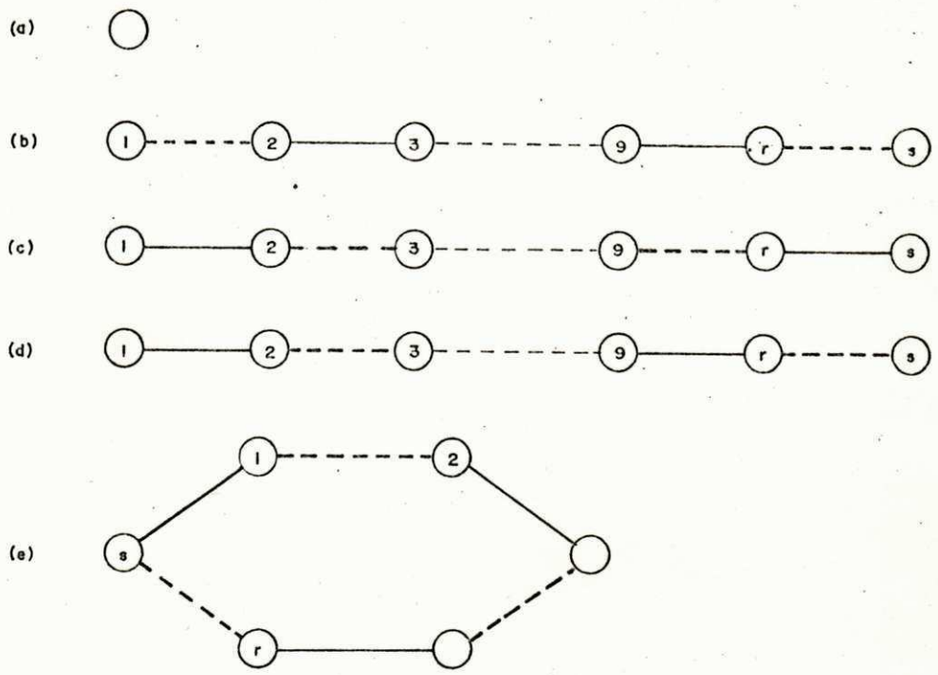


Figura 5.7: _____ Ligações em X - - - - Ligações em X^*

máxima. Um ciclo do tipo (e) com um número ímpar de ligações não pode existir, porque então duas ligações de X ou duas ligações de X^* serão adjacentes em algum nó. Então, os que sobram, são componentes da forma (a), (d) e (e) com um número par de ligações. Para cada um desses componentes, o núemro de ligações $n(X)$ em X é igual ao número de ligações $n(X^*)$ em X^*. Como isso se aplica a cada componente conectado k de G_P, então temos

$$\sum_k n_k(X) = \sum_k n_k(X^*)$$

onde $n_k(X)$ e $n_k(X^*)$ são os números de ligações do componente k pertencendo respectivamente a X e X^*. Então:

$$|X| \equiv |X \cap X^*| + \sum_K n_K(X) = |X \cap X^*| + \sum_K n_K(X^*) \equiv |X^*|$$

portanto X é um acoplamento de cardinalidade máxima.

Como exemplo, considere-se o grafo da Figura 5.6. Trocando

alternadamente as ligações na cadeia de aumento P = (9, 4, 10, 1, 2, 3, 5, 6), as quais não estão em X, com aquelas ligações na cadeia que estão; isto produz o novo acoplamento X' mostrado na Figura 5.8. Nessa figura não existe nenhum nó exposto; então, nenhuma cadeia de aumento pode existir e, portanto, conforme o teorema 5.1, X', é um acoplamento de cardinalidade máxima.

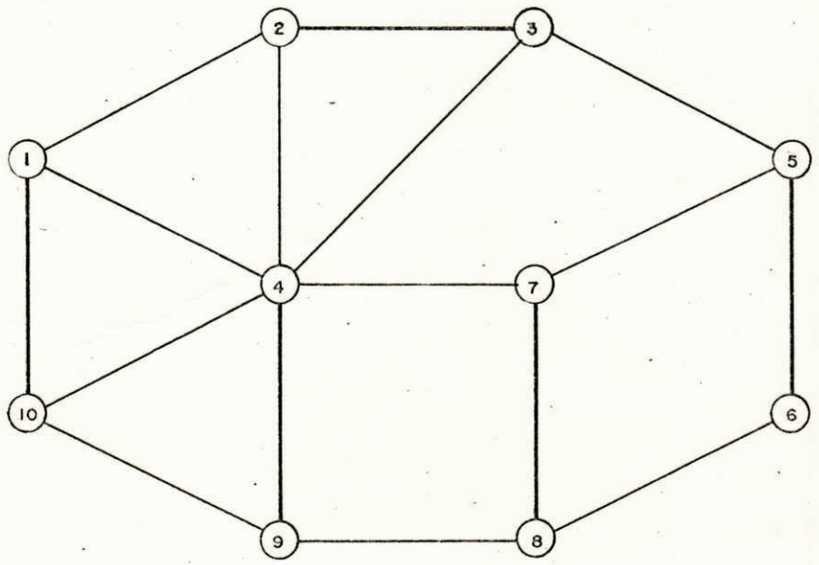


Figura 5.8: Acoplamento de cardinalidade máxima X'

O teorema 5.1 parece que por si mesmo é suficiente para prover uma solução para o problema de acoplamento de cardinalidade máxima; podemos começar com um acoplamento qualquer no grafo, depois aumentar a cardinalidade do acoplamento por descobrimento de cadeias de aumento e continuando este processo até que o grafo não admita nenhuma cadeia de aumento. Conforme o teorema 5.1, o acoplamento obtido é de cardinalidade máxima. Encontrar cadeias de aumento para grafos pequenos não é difícil, mas o número das cadeias possíveis aumenta exponencialmente com o número de nós; então, para grafos grandes é praticamente impossível testar todas as cadeias.

Uma solução elegante para esse problema foi descoberta por EDMONDS, a qual será apresentada neste capítulo. EDMONDS adicionou conceitos de árvores alternadas e flores ao conceito de cadeia alternada.

5.2.3 Árvores Alternadas

Uma Árvore Alternada em relação a um acoplamento X é uma árvore A para a qual:

(a) Um nó de A é exposto e é chamado raiz de A;

(b) Todas as cadeias começando na raiz são cadeias alternadas;

(c) Todas as cadeias partindo da raiz contêm um número par de ligações.

Agora, o encontro de cadeias de aumento pode ser feito construindo árvores alternadas no grafo. Um procedimento sistemático para construir uma árvore alternada e descobrir uma cadeia de aumento pode ser elaborado por designação de rótulos aos nós como se segue [14]:

Dado um acoplamento X, designamos rótulos S e T aos nós da seguinte maneira. primeiramente designamos um rótulo "$S:\phi$" a um nó exposto. Depois, quando um rótulo S em um nó i pesquisado, a cada ligação $(i,j) \notin X$ incidente a i, o rótulo "$T:i$" é dado ao nó j, a menos que o nó j tenha um rótulo-T. Quando um rótulo-T é pesquisado, a única ligação $(i,j) \in X$ é identificada, e rótulo "$S:i$" é dado ao nó j. O procedimento é continuado até que um rótulo-T seja dado a um nó exposto, ou mais rótulos não possam ser aplicados. No primeiro caso, uma cadeia de aumento é encontrada. No segundo caso, a ár

vore de nós rótulados é chamada "Húngara" e uma outra árvore pode ser construída a partir de um outro nó exposto.

Este procedimento pode levar ao descobrimento falso de uma cadeia de aumento, como mostrado na Figura 5.9.

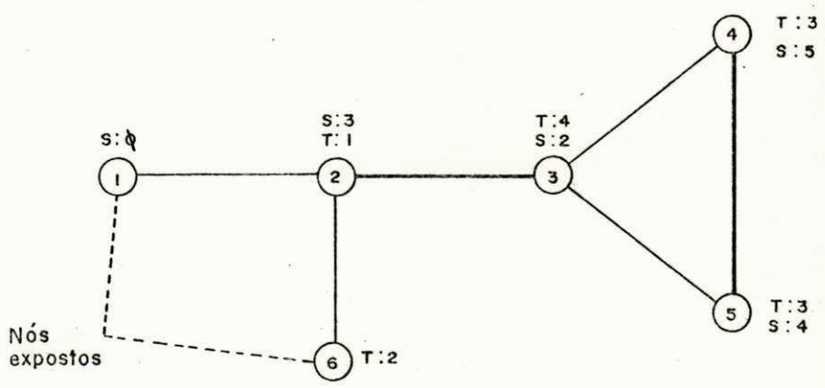


Figura 5.9: Descobrimento falso de uma cadeia de aumento

Podemos suspeitar que o processo de designação de rótulos, às vezes, encontra uma cadeia de aumento não existente, porque é permitido designar ambos os tipos dos rótulos a um nó. Será muito facíl adicionar a restrição que, uma vez que é dado um tipo de rótulo a um nó, não pode ser dado o outro tipo de rótulo ao mesmo. Obviamente, isto elimina a possibilidade de formação de cadeias falsas, mas também pode proibir o descobrimento de cadeias falsas, mas também pode proibir o descobrimento de cadeias de aumento válidas, como mostrado na Figura 5.10.

Estas deficiências serão resolvidas pela definição de flores e pela maneira especial de designação de rótulos aos nós de uma flor.

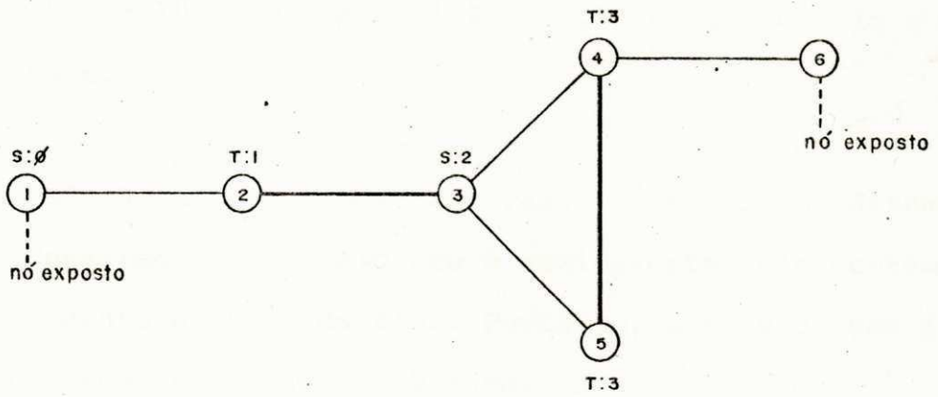


Figura 5.10: A cadeia de aumento não encontrada

5.2.4 Flores

Seja X um acoplamento no grafo $G = (N,A)$. Seja $N_b \subseteq N$ um subconjunto de $2r + 1$ nós, $r \geq 1$, e seja B o conjunto de todos os ramos, que são incidentes ao nós em N_B. B é chamado uma Flor com respeito ao acoplamento X se:

$$|X \cap B| = r.$$

Isto significa que o acoplamento X é máximo dentro de B. O único nó b de N_B deixado exposto por $X \cap B$ é a Base da flor.

Existe uma cadeia alternada T, chamada o Talo da flor, onde $[T]$ é par e $T \cap B = \phi$, estendendo-se da base da flor a um nó exposto por X, chamado a raiz do talo.

Para cada nó $i \in N_B$ há uma cadeia alternada $S_{b,i} \subseteq B$, onde $|S_{b,i}|$ é par, entre o nó i e a base da flor. Isto resulta que há uma cadeia alternada da forma T, $S_{b,i}$ entre a raiz do talo e o nó i.

A forma mais simples de uma flor é aquela que B é um ciclo

é mostrado um grafo no qual os ramos (3,4), (4,5), (5,6), (6,7) e (7.3) formam uma flor com respeito ao acoplamento mostrado no mesmo. O grafo da Figura 5.11(b) é o mesmo grafo no qual a flor é encolhida a um pseudo-nó. Agora nesse grafo encontramos a cadeia de aumento (1, 2, *B*, 8). O teorema 5.2 garante que podemos encontrar uma cadeia de aumento no grafo original. Isto é baseado no fato que entre qualquer nó de uma flor e a base, existe uma cadeia alternada de cardinalidade par. No exemplo, essa cadeia é (5, 4, 3) e a cadeia de aumento nesse grafo é (8, 5, 4, 3, 2, 1).

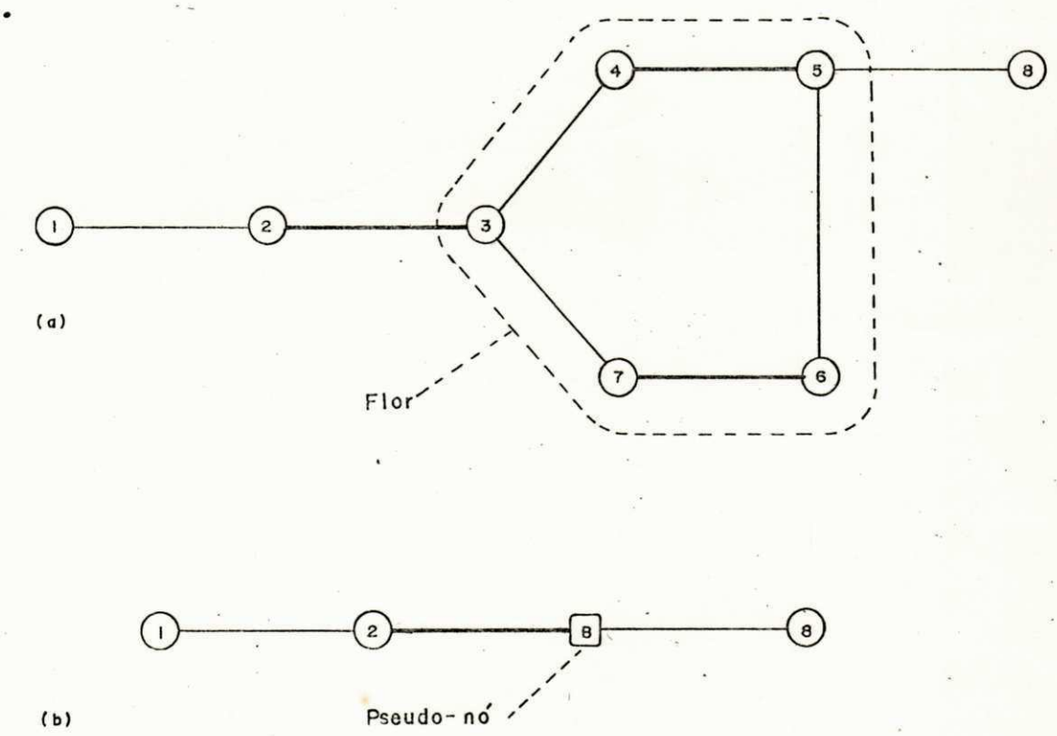


Figura 5.11: Exemplo para o teorema 5.2

Agora podemos fazer o esboço de um algoritmo que será apresentado com um exemplo. Considere-se o acoplamento no grafo mostrado na Figura 5.12. Existe uma cadeia de aumento do nó 1 para o 10. Nossa tarefa é construir essa cadeia sistematicamente.

Inicialmente, estabelecemos o nó 1 como a raiz de uma árvore alternada. O nó 1 recebe o rótulo "$S:\phi$"; os nós 2 e 3 recebem o rótulo "$T:1$" e assim por diante (Figura 5.13(a)). Observamos que aos nós 6 e 7 são dados rótulos-T e um ramo do acoplamento é encontrado entre eles. Portanto, uma flor B_1 é formada e substituída por um pseudo-nó B_1 como indicado na Figura 5.13(b). O pseudo-nó B_1 recebe o mesmo rótulo-S que a base da flor recebeu, e esse rótulo é considerado ser não pesquisado. Continuando o procedimento de designação de rótulos, resulta o descobrimento e encolhimento das flores B_2 e B_3 como mostrado na Figura 5.13(c). Finalmente, uma cadeia de aumento é encontrada na Figura 5.13(d).

A construção de uma cadeia de aumento no grafo original prossegue desta maneira: Primeiro, "retraçando" do nó 10 no grafo final (Figura 5.13(d)), encontra-se a sequência de nós B_3, 10. Então é necessário encontrar uma cadeia alternada dentro de B_3 no grafo

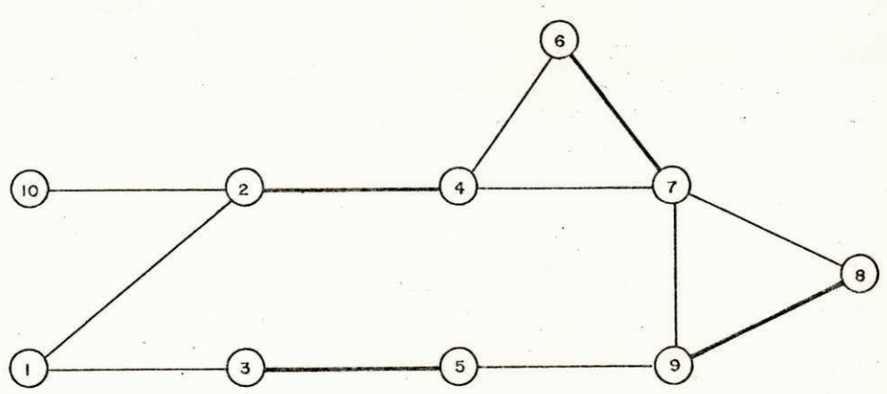


Figura 5.12: Grafo para exemplo

da Figura 5.13(c). A cadeia alternada apropriada os nós 1, 3, 5, B_2, 2. A cadeia desejada dentro B_2 na Figura 5.13(b) é B_1, 8, 9 e a cadeia dentro B_1 na Figura 5.13(a) é 7, 6, 4. Colocando essas cadeias juntas, obtemos a sequência desejada 1, 3, 5, 9, 8, 7, 6, 4, 2, 10.

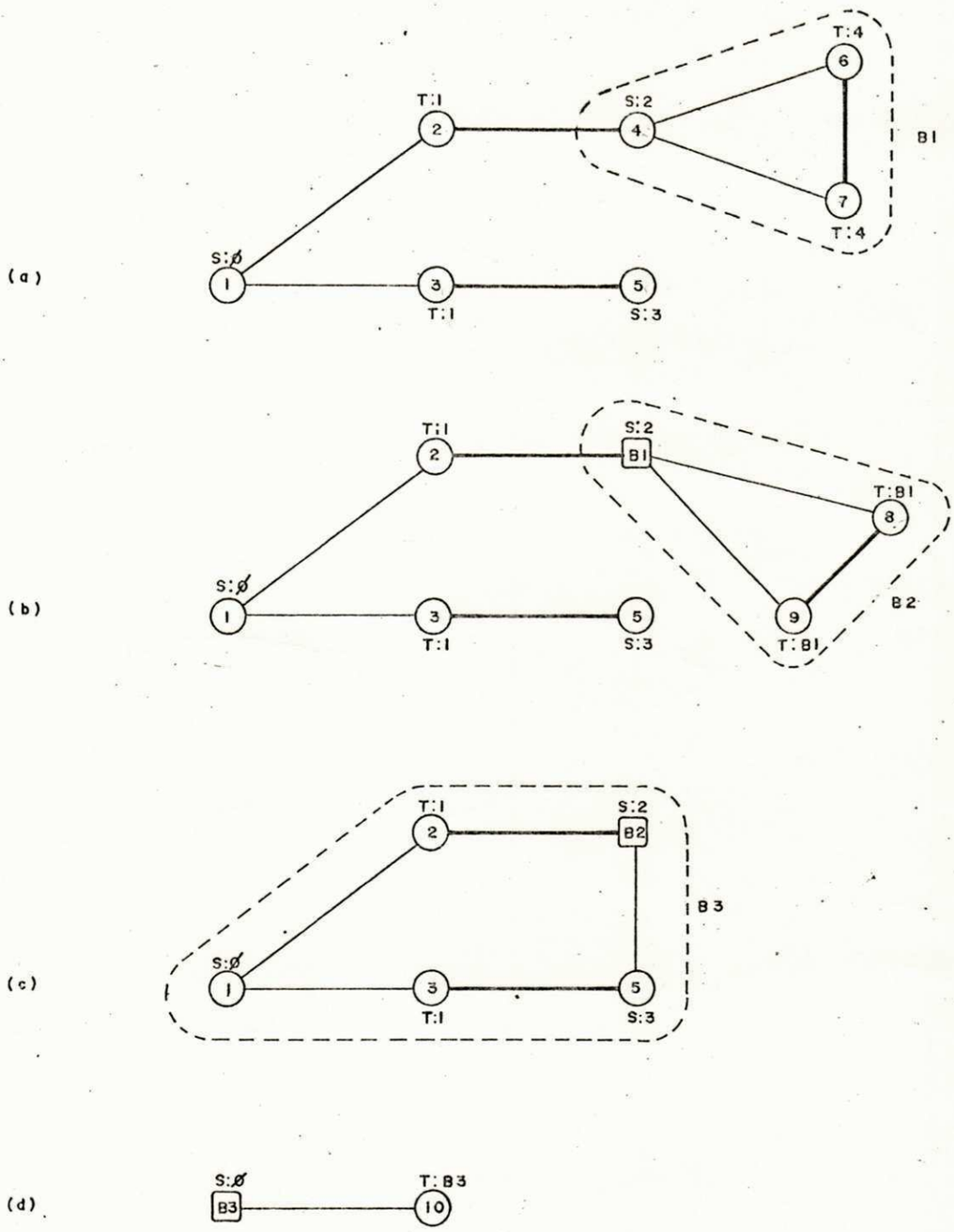


Figura 5.13: Árvore alternada para exemplo

Agora nos interessa a implementação do algoritmo de EDMONDS para o computador. Nós desenvolveremos um procedimento de designação de rótulos, o qual não exigirá encolhimento de flores. O procedimento provê um método sistemático e eficiente para o "retraçamento" dentro das flores [14].

5.2.5 Registro de Flores

Nós precisamos guardar um registro somente das flores mais esternas, e essas flores são identificadas por seu nó base. Associamos um índice $b(i)$ a cada nó i, indicando o nó base da flor mais externa na qual ele é contido. Se um nó i não é contido numa flor, então $b(i) = i$. Portanto, dois nós i,j estão na mesma flor mais externa, se e somente se $b(i) = b(j)$.

Quando uma nova flor é formada, o nó base da nova flor é identificado e $b(i)$ fica igual a b para todos os nós i na flor. Isto significa que é necessário manter uma listagem de todos os nós dentro de uma dada flor.

5.2.6 Descobrimento de Cadeias de Aumento e Flores

É possível começar com uma árvore alternada, e, quando a árvore torna-se húngara, começar uma outra num novo nó exposto. Ou podemos começar enraizando uma árvore alternada em cada nó exposto e simultaneamente ampliá-las. Por razões técnicas motivadas pela modificação das variáveis duais para o problema de acoplamento com peso, esta segunda alternativa é preferida. Portanto, nós adaptamos este plano aqui.

Inicialmente, o rótulo "S:ϕ" é dado a todos os nós expostos. Depois, rótulos-T e rótulos-S são aplicados aos nós. Um rótulo-S indica a existência de uma cadeia alternada de cardinalidade par para o nó raiz, e um rótulo-T indica a existência de uma cadeia alternada de cardinalidade ímpar (Um nó recebe ambos os tipos de rótulos, se e somente se ele é um nó não-base de uma flor mais externa). Cadeias de aumento estendem-se entre nós raizes de duas árvores diferentes como é mostrado na Figura 5.14.

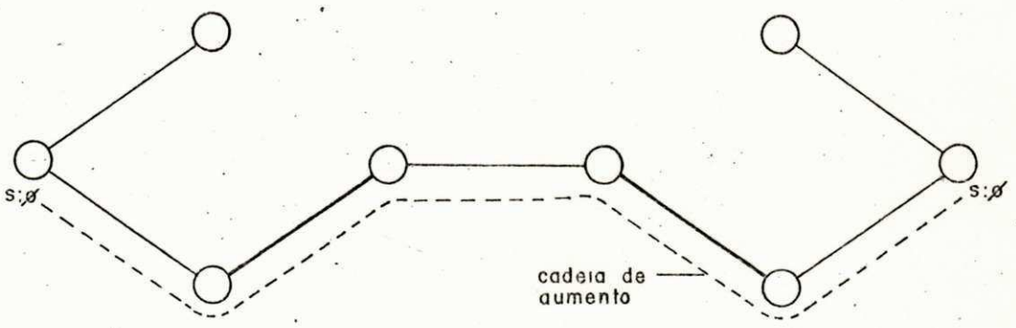


Figura 5.14: Exemplo de cadeia de aumento

Agora, suponha-se que o procedimento de designação de rótulos descobre um ramo $(i,j) \notin X$ onde i e j tem rótulos-S ou um ramo $(i,j) \in X$, onde i e j têm rotulos-T. Suponha-se $b(i) \neq b(j)$, isto significa que os nós i e j não estão contidos dentro da mesma flor. Então, uma cadeia de aumento é encontrada, se i e j estão em árvores alternadas diferentes, e uma nova flor é encontrada, se i e j estão na mesma árvore. A pergunta é: qual dessas situações existe? A resposta é conseguida pelo retraçamento dos rótulos dos nós i e j. Se diferentes nós raizes foram encontrados, então uma cadeia de aumento é descoberta. Se o mesmo nó raiz é encontrado em ambos os retraçamentos, então uma flor é formada.

5.2.7 O Procedimento de Designação de Rótulos

O procedimento de designação de rótulos, que fornece um algoritmo para descobrimento de flores e cadeias de aumento é o seguinte:

Quando um rótulo-S em um nó i é pesquisado, o procedimento seguinte é feito para cada ramo $(i,j) \notin X$ incidente a i. Se $b(i)=b(j)$, então nada é feito, porque i e j estão contidos dentro da mesma flor (No momento que uma flor é formada, todos os rótulos possíveis

são aplicados; Veja abaixo). Senão, se o nó j tem um rótulo-S, o retraçamento é feito do nó i e do nó j para descobrir uma cadeia de aumento ou uma flor. Se o nó j não tem um rótulo-S nem um rótulo-T, então o rótulo "$T:i$" é aplicado ao nó j.

Quando um rótulo-T em um nó i é pesquisado, o único ramo $(i,j)\varepsilon X$ incidente a i é encontrado. Se $b(i) = b(j)$, então nada é feito. Senão, se o nó j tem um rótulo-T, o retraçamento é feito do nó i e do nó j para descobrir uma cadeia de aumento ou uma flor. Se o nó j não tem rótulo-S nem um rótulo-T, então o rótulo "$S:i$" é aplicado ao nó j.

5.2.8 A Construção de Flores

Uma vez que uma flor é formada, é necessário identificar os nós da flor e o seu nó base. Isto é feito da seguinte maneira:

O retraçamento dos nós i e j produz duas sequências de nós:

$$i_1, i_2, \ldots, i_p$$

$$j_1, j_2, \ldots, j_q$$

onde $i_1 = j_1$ (o nó raiz da árvore alternada) e $i_p = i$, $j_q = j$ (onde o retraçamento começa). Como $i_1 = j_1$ e $i_p \neq j_q$, há alguns índices m, tais que $i_1 = j_1$, $i_2 = j_2$, $\ldots i_m = j_m$, ou $i_m = i$ ou $j_m = j$, ou $i_{m+1} \neq j_{m+1}$. A base da nova flor é i_ℓ, $\ell \leq m$, onde $i_\ell = b(i_m)$, e seu talo percorre os nós $i_1, i_2, \ldots i_\ell$. A nova flor contém todos os nós k, de modo que:

$$b(k) \varepsilon \{b(i_m), b(i_{m+1}), \ldots, b(i_p), b(j_{m+1}), b(j_{m+2}), \ldots, p(j_q)\}$$

De acordo com isto, $b(k)$ é feito igual a i_ℓ para todos os nós k na nova flor. Isto, mais a adição de "rótulos ausentes" a nós da flor (a ser descrita abaixo), é tudo que é necessário para encolhimento da flor.

Como exemplo, considere a situação mostrada na Figura 5.15. Há rótulos-T nos nós 8 e 9 e $(8,9) \varepsilon X$. Por isto uma flor é descoberta. Retraçando dos nós 8 e 9, resultam as sequências:

1, 2, 3, 4, 6, 8

e

1, 2, 3, 4, 6, 9

nesse caso $i_m = j_m = 6$ e $b(6) = 3$, porque o nó 6 já faz parte de uma flor, com o nó 3 em sua base. Os nós 3, 4, 5, 6, 7, 8 e 9 estão na flor, e os nós 1, 2 e 3 formam o talo.

5.2.9 Rótulos dos Nós nas Flores

Entre cada nó não-base na nova flor e a raiz da árvore alternada existe uma cadeia alternada de cardinalidade para e uma de cardinalidade ímpar. Esse fato deve ser indicado pela existência de um rótulo-S e um rótulo-T em cada nó da flor.

Suponha-se que a flor foi descoberta por retraçamento de nós $i = i_p$ e $j = j_q$ onde i e j têm rótulo-S e $(i,j) \notin X$. Nós nos interessamos somente pelos nós $i_{m+1}, i_{m+2}, \ldots, i_p$ (naturalmente as regras para os nós $j_{m+1}, j_{m+2}, \ldots, j_q$ são similares). No retraçamento foram usados rótulos-S em $i_p, i_{p-2}, \ldots, i_{m+2}$, e rótulos-$T$ em $i_{p-1}, i_{p-3}, \ldots, i_{m+1}$. Por isto, qualquer rótulo ausente deve ser rótulo-T em $i_p, i_{p-2}, \ldots, i_{m+2}$, ou rótulo-$S$ em $i_{p-1}, i_{p-3}, \ldots, i_{m+1}$. O rótulo designado a qualquer nó i_r será tal que o retraçamen

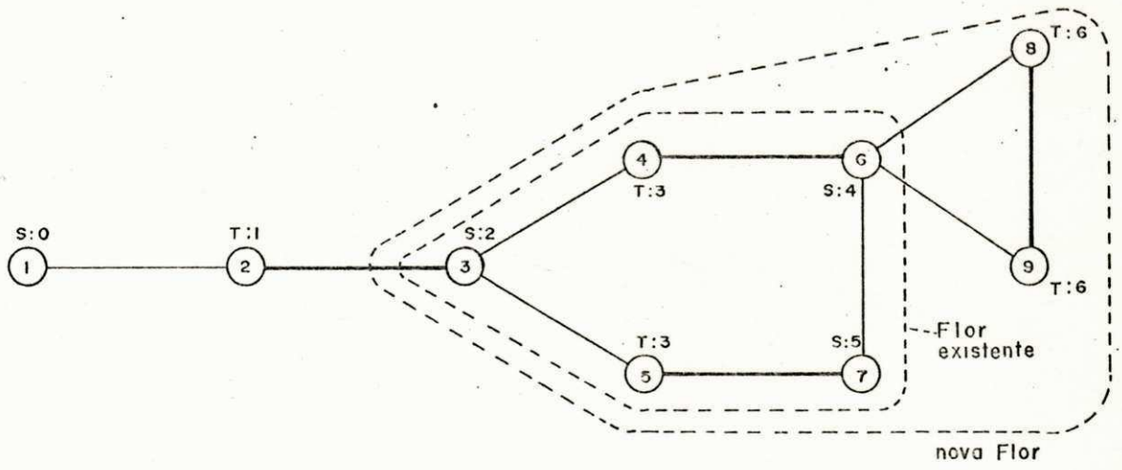


Figura 5.15: Exemplo de construção de flores

to daquele rótulo resulte a sequência de nós i_r, i_{r+1}, ..., i_p, j_{q-1}, ..., j_1.

Sejam designados rótulos ausentes a i_{m+1}, i_{m+2}, ..., i_p. Suponha-se que i_r não tem um rótulo-S. Afirmamos necessariamente que $(i_r, i_{r+1}) \in X$ e que i_{r+1} não tem um rótulo-T. Então, damos a i_r o rótulo "$S:i_{r+1}$".

Agora suponha-se que i_r não tem um rótulo-T. Afirmamos necessariamente que $(i_r, i_{r+1}) \notin X$. Se i_{r+1} também não tem um rótulo-S, então damos o rótulo "$T:i_{r+1}$" a i_r.

Mas agora suponha que i_r não tem rótulo-T e i_{r+1} já tem um rótulo-S. Então i_r deve ser o nó base de uma flor anteriormente existente, contendo i_{r+1}, e o retrançamento do rótulo-S em i_{r+1} voltará a i_r. Portanto, é completamente errado dar o rótulo "$T:i_{r+1}$" ao nó i_r.

O que fazemos para resolver este problema, é encontrar o último nó i_k na sequência i_{r+1}, i_{r+2}, ..., i_p que está contido nessa

flor mais externa anteriormente formada com i_r sendo sua base. Necessariamente $k \geq r + 2$. Então designamos um rótulo especial "$T:i_{k+1}$, i_k" a i_r. Esse rótulo é interpretado como o seguinte: Existe uma cadeia alternada de cardinalidade ímpar entre i_r e o nó raiz. Para achar essa cadeia, retrace-se do rótulo-S no nó i_{k+1} para a raiz, também do nó i_k ao próprio i_r. Os ramos descobertos junto com o ramo (i_k, i_{k+1}), corretamente ordenados, constituem a cadeia alternada desejada.

Um exemplo da designação de rótulos dentro de uma flor é mostrado na Figura 5.16.

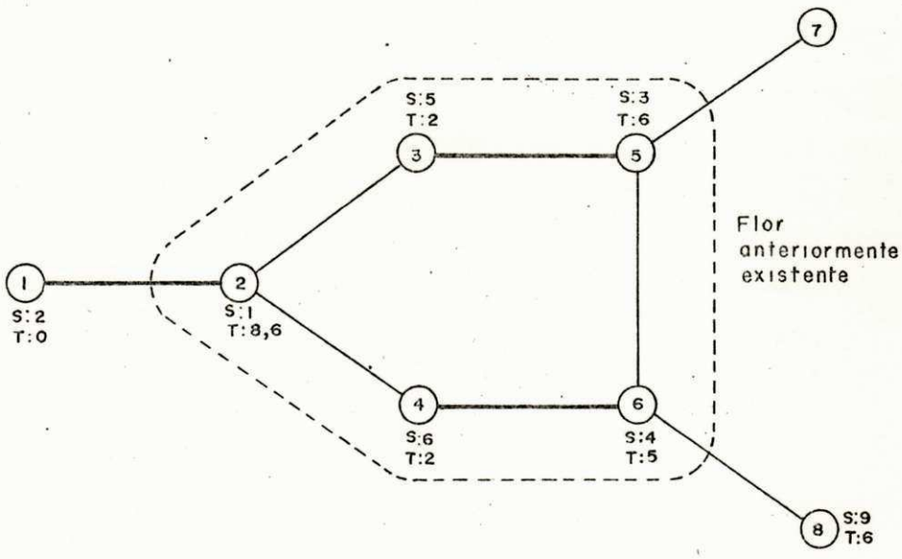


Figura 5.16: Designação de rótulo dentro de uma flor

5.2.10 Rotina de Retraçamento

A introdução do rótulo especial-T com índice duplo complica um pouco o retraçamento. Por exemplo, no retraçamento talvez encontramos o rótulo "$T:i,j$" no nó k. Retraçando de j para k, talvez encontremos "$T:i_1$, j_1" no nó K_1. Retraçando de j_1 para k_1, talvez encontremos "$T:i_2$, j_2" no nó k_2, e assim por diante. Isto pode ser

continuado tantas vezes quantas flores forem aninhadas. Basta dizer que a rotina de retraçamento pode ser implementada eficientemente num computador.

O algoritmo completo de acoplamento de Cardinalidade máxima pode ser assim resumido:

5.2.11 Algoritmo de Acoplamento de Cardinalidade Máxima

PASSO 0 (Início)
O grafo $G = (N,A)$ é dado, seja X qualquer acoplamento, possívelmente o acoplamento vázio. Coloquemos $b(i) = i$, para todos os nós $i \in N$. nenhum nó é rótulado.

PASSO 1 (Designação de Rótulos).
(1.0) Apliquemos o rótulo "$S:\phi$" a cada nó exposto.
(1.1) Se não houver nenhum rótulo não-pesquisado, desviemo-nos ao passo 4. Senão encontremos um nó i com um rótulo não-pesquisado. Se o rótulo é um rótulo-S, desviemo-nos ao passo 1.2: Se ele é um rótulo-T, desviemo-nos ao passo 1.3.
(1.2) Pesquisemos o rótulo-S no nó i pela execução do seguinte procedimento para cada ramo $(i,j) \notin X$ incidente ao nó i. Se $b(i) = b(j)$, não façamos nada. Senão se o nó j tem um rótulo-S, retracemos dos rótulos-S nos nós i e j; se diferentes nós raizes forem encontrados, desviemo-nos ao passo 2; se o mesmo nó raiz for encontrado, desviemo-nos ao passo 3. Se o nó j não tem um rótulo-S nem um rótulo-T, apliquemos o rótulo "$T:i$" ao nó j. Quando o nó i é pesquisado, voltemos ao passo 1.1.
(1.3) Pesquisemos o rótulo-T no nó i como a seguir: Encontremos o único ramo $(i,j) \in X$ incidente ao nó i. Se $b(i) = b(j)$ não façamos nada. Senão, se o nó j tem um rótulo-T, retracemos dos rótulos-T dos rótulos-T nos nós i e j; Se diferentes nós raizes foram encontra

dos, desviemo-nos ao passo 2; Se o mesmo nó raiz for encontrado, desviemo-nos ao passo 3. Se o nó j não tem um rótulo-S nem um rótulo-T, apliquemos o rótulo "$S:i$" ao nó j. Voltemos ao passo 1.1.

PASSO 2 (Aumento)

Uma cadeia de aumento é encontrada no passo 1.2 ou 1.3. Aumente-se o acoplamento X. Retirem-se todos os rótulos dos nós e faça $b(i) = i$ para todos os i. Volte-se ao passo 1.0.

PASSO 3 (Floração)

Uma flor é formada no passo 1.2 ou 1.3. Determinem-se os nós da nova flor e o seu nó base, como descrito no texto. Forneçam-se rótulos ausentes para todos os nós não-base na nova flor. Atualizem-se $b(i)$ para todos os nós i na nova flor. Volte-se ao passo 1.2 ou 1.3., como lhe for apropriado.

PASSO 4 (Árvores Húngaras)

As árvores formadas por designação de rótulos são Húngaras. Não existe nenhuma cadeia de aumento e o acoplamento X é de cardinalidade máxima. Os rótulos e os números das flores podem ser usadas para construir uma solução dual ótima (veja seção 5.3).

5.3 Acoplamento com Peso

5.3.1 Formulação pela Programação Linear do Problema do Acoplamento com Peso

Consideremos agora o problema de acoplamento no caso geral para um grafo $G = (N,A)$, com pesos W_{ij} associados a cada ramo $(i,j) \varepsilon A$

Seja

$X_{ij} = 1$ se o ramo (i,j) é escolhido para o acoplamento;

$X_{ij} = 0$, em caso contrário.

A formulação pela programação linear do problema é

$$\text{Maximizar} \quad W\,x \tag{5.1}$$

$$A\,x \leq 1 \tag{5.2}$$

$$x = \{0.1\}$$

onde A é a matriz de incidência do grafo para o qual o acoplamento deve ser calculado. EDMONDS [5] mostrou que a restrição (5.3) pode ser substituída por um sistema linear de restrições e provou o seguinte teorema:

TEOREMA 5.3 Para qualquer grafo, o poliedro convexo definido por (5.2) e (5.3) é o mesmo definido por (5.2) e em adição:

(i) Para qualquer subconjunto $R_k \varepsilon N$ contendo $2r_k + 1$ nós (i.e. um número par de nós)

$$\sum_{i \varepsilon R_k} \sum_{j \varepsilon R_k} X_{ij} \leq r_k \tag{5.4}$$

e

(ii) $x \geq 0$ (5.5)

Obviamente, qualquer acoplamento em G satisfaz restrições (5.4) e (5.5); o que não é óbvio é que essas restrições também são suficientes. Nós mostramos isto construtivamente pelo fornecimento de um acoplamento em G, o qual é a solução do programa linear definido por (5.1), (5.2), (5.4) e (5.5) para qualquer conjunto dado de pesos.

O dual para o programa linear acima é:

Minimizar $\sum_i u_i + \sum_k r_k Z_k$

$$u_i + u_j + \sum_{R_k \{(i,j)\}} Z_k \geq W_{ij} \quad \forall (i,j) \in A$$

$$u, z \geq 0.$$

Portanto, uma variável u_i é associada a cada nó i de G, e uma variável z_k a cada conjunto ímpar de nós R_k.

De acordo com o teorema de folga complementar de programação linear [14], as condições de ortogonalidade necessárias e suficientes para que as soluções primal e dual sejam ótimas, são:

$$X_{ij} > 0 \rightarrow u_i + u_j + \sum_{R_k \{(i,j)\}} Z_k = W_{ij}, \qquad (5.6)$$

$$u_i > 0 \rightarrow \sum_j X_{ij} = 1 \qquad (5.7)$$

$$Z_k > 0 \rightarrow R_k X = r_k \qquad (5.8)$$

Agora porossigamos provando e teorema 5.3 para descrição de um procedimento que encontra um acoplamento viável X, e também um vetor $[u_i , z_k]$ que satisfaz as condições (5.4) e (5.5) e as condições de ortogonalidade (5.6), (5.7) e (5.8).

O procedimento computacional mantém a viabilidade do primal e do dual em todos os momentos, e além disso mantém todas as condições de ortogonalidade satisfeitas, menos a condição (5.7). O número de tais condições não satisfeitas ou seja, o número de nós expostos i para os quais u_i é positivo, é diminuido monotonicamente durante os cálculos.

Começamos com o acoplamento viável $X = \phi$ e com a solução

dual viável

$$u_i = W, \quad \text{para todos os } i,$$

$$Z_k = 0, \quad \text{para todos os } k,$$

onde W é adquadamente grande, digamos

$$W = \frac{1}{2} \max_{i,j} \{W_{ij}\}$$

Essas soluções iniciais do primal e do dual claramente satisfazem todas as condições (5.6) e (5.8), mas não a condição (5.7).

No passo geral do procedimento, X é viável, todas as condições (5.6) e (5.8) são satisfeitas, mas algumas das condições (5.7) não são. Então, procuramos encontrar uma cadeia de aumento dentro do subgrafo obtido por encolhimento de todas as flores k para as quais $Z_k > 0$, usando somente os ramos (i,j) para as quais $u_i + u_j + \Sigma Z_k = W_{ij}$.

Se uma cadeia de aumento é encontrada, ela estende-se entre dois nós expostos i e j, para os quais $u_i = u_j > 0$. Portanto, depois de aumento do acoplamento mais duas das condições (5.7) se tornam satisfeitas. A alteração do acoplamento dentro de cada uma das flores encolhidas é de tal modo que o acoplamento continua sendo máximo dentro da flor. Portanto, cada uma das condições (5.8) depois do aumento do acoplamento continua sendo satisfeita, porque a cadeia de aumento envolve somente os ramos (i,j) para os quais $u_i + u_j + \Sigma Z_k = W_{ij}$; então todas as condições (5.6) continuam sendo satisfeitas.

Se o aumento não é possível, então um valor apropriado $\delta > 0$

é escolhido, e as seguintes alterações são feitas nas variáveis duais. Para cada nó i com um rótulo-S e cada nó i contido dentro de uma flor mais externa, δ é subtraído de u_i. Para cada nó i com um rótulo-T, δ é adicionado a u_i. Para cada flor mais externa k, 2δ são adicionados a z_k.

Se um ramo (i,j) é contido dentro de uma flor, não há nenhum efeito sobre $u_i + u_j + \Sigma z_k$ causado pelas alterações nos valores das variáveis duais. Mas, se o nó i tem um rótulo-S ou é contido dentro de uma flor mais externa e o nó j não é rótulado, então o efeito líquido é $-\delta$. Outros casos são indicados na Figura 5.17. como antes os quadros representam pseudo-nós.

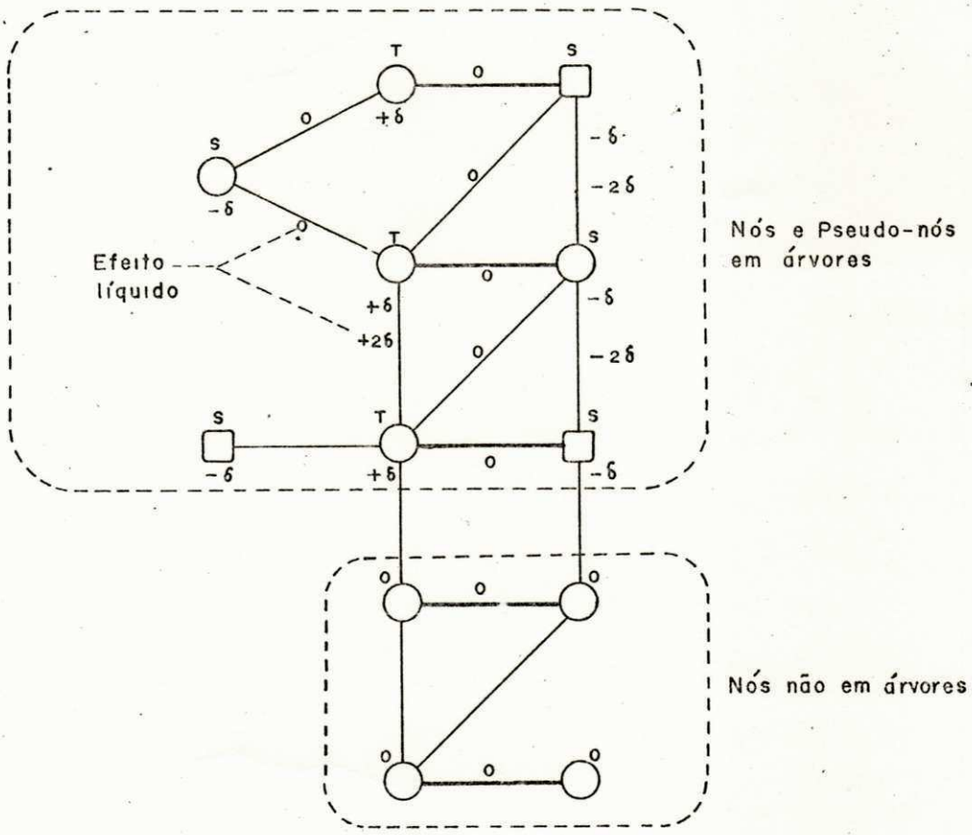


Figura 5.17 Efeito de alteração nas variáveis duais

Há três restrições para o valor máximo de δ que são as se

guintes:

(5.9) Se i é um nó com um rótulo-S ou é contido dentro de uma flor mais externa, é necessário que $u_i - \delta \geq 0$.

(5.10) Se (i,j) é um ramo tal que ambos os nós i e j têm rótulo-S ou estão contidos dentro de flores diferentes, é requerido que

$$(u_i - \delta) + (u_j - \delta) \geq W_{ij}$$

(5.11) Se (i,j) é um ramo tal que i é um nó com um rótulo-S ou é contido dentro de uma flor mais externa, enquanto j é um nó ainda não rótulado, então é necessário que

$$(u_i - \delta) + u_j \geq W_{ij}.$$

Suponha-se que δ é escolhido tão grande quanto possível, sujeito às restrições (5.9) a (5.11). Se a condição (5.9) está controlando o valor de δ, então a nova solução dual é de tal modo que todas as restrições (5.7) são satisfeitas. Ambas as soluções primal e dual são ótimas e um acoplamento de peso máximo é obtido. (Lembramos que valores iniciais uniformes foram escolhidos para as variáveis u_i. Portanto, o mesmo valor mínimo de u_i existe em cada nó exposto i.).

Se a restrição (5.10) está controlando o valor de δ, então ou uma cadeia de aumento pode ser encontrada ou uma nova flor é formada. Quando o mesmo ocorre para (5.11), pelo menos um novo ramo pode ser adicionado a uma das árvores alternadas.

Agora podemos esboçar o algorítmo da seguinte maneira:

5.3.2 Esboço do Algoritmo de Acoplamento com Peso Máximo

PASSO 0 (Início)

Começemos com $X = \phi$ e $u_i = \frac{1}{2} \max \{W_{ij}\}$ como as soluções primal e dual.

PASSO 1 (Designação de Rótulos)

Geremos uma raiz para uma árvore alternada em cada nó exposto e prossigamos a construção da mesma pela designação de rótulos, usando somente os ramos (i,j) para os quais

$$u_i + u_j + \Sigma Z_k = W_{ij}.$$

Se uma cadeia de aumento é encontrada, desviemo-nos ao passo 2. Se uma flor é formada, desviemo-nos ao passo 3. Se as árvores se tornam Húngaras, desviemo-nos ao passo 4.

PASSO 2 (Aumento)

Encontremos a cadeia de aumento, trançando a cadeia dentro das flores encolhidas. Aumentemos o acoplamento, removamos todos os rótulos dos nós e pseudo-nós e voltemos ao passo 1.

PASSO 3 (Floração)

Identifiquemos a flor e encolhamo-la no grafo. O pseudo-nó representando a flor recebe um rótulo-S e a sua variável-Z é feita igual a zero. Voltemos ao passo 1.

PASSO 4 (Alteração nas Variáveis Duais).

Determinemos o valor máximo de δ de acordo com as restrições (5.9) a (5.11) e façamos as alterações apropriadas nas variáveis duais. Se a restrição (5.9) está controlando o valor de δ, paremos; o acoplamento e a solução dual são ótimos. Caso contrário, removamos todos

os rótulos dos nós e voltemos ao passo 1.

5.3.3 Implementação do Algoritmo de Acoplamento com Peso

Agora consideremos a implementação do algoritmo de acoplamento com peso esboçado na seção anterior.

O procedimento de designação, o encontro das cadeias de aumento e a formação de flores são semelhantes aos que foram considerados na implementação do algoritmo do acoplamento de cardinalidade máxima.

Variável - Δ

Uma variável Δ é introduzida e atualizada pelo procedimento de designação de rótulos. Essa variável é para indicar o valor máximo de δ que pode ser escolhido, sujeito às restrições (5.10) e (5.11).

Variável - $\bar{W}_{ij}$

Para abreviar cálculos, uma variável $\bar{W}_{ij}$ é usada para representar $u_i + u_j - W_{ij}$. Cada vez que nos cálculos os valores de u_i, u_j e W_{ij} são encontrados, também será calculado $\bar{W}_{ij} = u_i + u_j - W_{ij}$.

5.3.4 O Algoritmo de Acoplamento com Peso

PASSO 0 (Início)

O grafo $G = (N,A)$ é dado com um peso W_{ij} associado a cada ramo (i,j). Coloquemos $u_j = \frac{1}{2} \max \{W_{ij}\}$, para cada nó $i \in N$ Façamos $\Delta = +\infty$. Coloquemos $X = \phi$. Não há nehuma flor e nenhum nó é rótula

do. Façamos $b(i) = i$ para todos os nós $i \in N$.

PASSO 1 (Designação de Rótulos)

(1.0) Apliquemos o rótulo "$S:\phi$" a cada nós exposto.

(1.1) Se não houver nenhum nó não-pesquisado, desviemo-nos ao passo 4. Caso contrário, encontremos um nó i com um rótulo não-pesquisado. Se o rótulo é um rótulo-S, desviemo-nos ao passo 1.2; se é um rótulo-T, desviemo-nos ao passo 1.3.

(1.2) Pesquisemos o rótulo-S no nó i pela execução do seguinte procedimento para cada ramo $(i,j) \notin X$ incidente ao nó i: Façamos $\bar{W}_{ij} = u_i + u_j - W_{ij}$. Se $b(i) = b(j)$, não façamos nada. Caso contrário, se o nó $b(j)$ tem um rótulo-S e $\bar{W}_{ij} = 0$, retracemos dos rótulos-S nos nós i e j. Se diferentes nós raizes forem encontrados, desviemo-nos ao passo 2; se o mesmo nó raiz é encontrado, desviemo-nos ao passo 3. Se o nó $b(j)$ tem um rótulo-S e $\bar{W}_{ij} > 0$, façamos $\Delta = min\ \{\Delta, \frac{1}{2}\ \bar{W}_{ij}\}$. Se o nó j não é rótulado e $\bar{W}_{ij} = 0$, apliquemos o rótulo "$T:i$" ao nó j. Se o nó j não é rótulado e $\bar{W}_{ij} > 0$, façamos $\Delta = min\ \{\Delta, \bar{W}_{ij}\}$. Quando o nó i é pesquisado, voltemos ao passo 1.1.

(1.3) Pesquisemos o rótulo-T no nó i pela execução do seguinte procedimento para o único ramo $(i,j) \in X$ incidente ao nó i. Se $b(i) = b(j)$, não façamos nada; caso contrário, se o nó j tem um rótulo-T, retracemos dos rótulos-T nos nós i e j. Se diferentes nós raizes forem encontrados, desviemo-nos ao passo 2. Se o mesmo nó raiz é encontrado, desviemo-nos ao passo 1.1.

PASSO 2 (Aumento).

Uma cadeia de aumento é encontrada no passo 1.2 ou 1.3. Aumentemos o acoplamento X. Retiremos todos os rótulos dos nós e façamos $b(i) = i$ para todos os $i \in N$ façamos $\Delta = +\infty$. Todos os nós são não-pesquisados. Desviemo-nos ao passo 1.0.

PASSO 3 (Floração)

Uma flor é formada no passo 1.2 ou 1.3. Determinemos os nós da nova flor e o seu nó base. Forneçamos rótulos ausentes para todos os nós da nova flor, menos o nó base. Atualizemos o vetor $b(i)$ para todos os nós na nova flor. Voltemos ao passo 1.2 ou 1.3, como lhe for apropriado.

PASSO 4 (Revisão das Variáveis Duais)

Encontremos:

$$\delta_1 = min \{u_i\}$$

$$\delta_2 = min \{\delta_1, \Delta\}$$

Coloquemos $u_i = u_i - \delta$ para cada nó i em que o nó $b(i)$ tem um rótulo-S. Coloquemos $u_i = u_i + \delta$ para cada nó i em que o nó i tem um rótulo-T, mas não é contido numa flor. Coloquemos $W_{ij} = W_{ij} - 2\delta$ para todos os ramos (i,j) em qualquer flor. Se $\delta = \delta_1$, paremos; X é o acoplamento de peso máximo. Caso contrário, coloquemos todos os rótulos-S no estado não pesquisado incluindo os nós contidos nas flores. Façamos $\Delta = + \infty$. Voltemos ao passo 1.1.

A codificação de um programa para este algoritmo é incluída neste trabalho. Este programa pode ser usado como uma sub-rotina na solução do problema do carteiro Chinês, nos casos de um grafo direcionado e não-direcionado, como foi discutido no capítulo 4.

Relembramos que o algoritmo de acoplamento considerado acima encontra o acoplamento de peso máximo. e sabemos que o algoritmo do problema do carteiro Chinês precisa de um acoplamento de peso mínimo (Veja o algorítmo do problema do carteiro Chinês no Capítulo 4). Por isso a seguinte transformação deve ser feita para todos os

pesos de todos os ramos no grafo G^*:

$$W_{ij} = M - a_{ij}$$

onde a_{ij} é o custo de percurso do caminho mais curto entre nós i e j, M é uma valor grande e W_{ij} é o peso transformado do ramo (i,j) em G^*. Agora um acoplamento de peso máximo em G^* usando os pesos W_{ij} é quivalente a um acoplamento de peso mínimo em G^* usando os pesos a_{ij}.

CAPÍTULO VI

O PROBLEMA DE COLETA DE LIXO

6.1 Introdução

Consideramos neste Capítulo o problema da coleta de lixo e a implementação do algoritmo do problema do Carteiro Chinês para resolvê-lo.

O problema da coleta de lixo pode ser definido da seguinte forma:

Dada uma rede de ruas (por exemplo, um bairro) na qual o lixo deve ser coletado, alguns recursos limitados (por exemplo caminhões de coleta e o tempo disponível para o serviço), e algumas restrições (como as capacidades dos caminhos), fazer um plano viável para a coleta de lixo em custo mínimo.

Como foi descrito no capítulo 2, uma rede de ruas pode ser representada por um grafo no qual os segmentos das ruas são representados pelos ramos e as interseções pelos nós. O comprimento ou custo de percurso de cada segmento pode ser atribuído como um peso aos ramos (uma definição mais precisa de custo de percurso será a

presentada na seção 6.3). Agora o algoritmo do problema do Carteiro Chinês pode achar no grafo correspondente um ciclo (ou circuito) que percorre todos os ramos pelo menos uma vez, de forma tal que o peso total do ciclo (ou circuito), seja mínimo.

Com a definição do problema da coleta de lixo, observa-se que o mesmo é uma generalização do problema do Carteiro Chinês e que esse algoritmo não pode ser aplicado para resolvê-lo. Mas veremos que, combinando o algoritmo com algumas técnicas heurísticas, podemos obter soluções boas (e não necessariamente ótimas) para o problema.

Existem algumas restrições adicionais no problema da coleta de lixo que não existiam no modelo clássico do problema do Carteiro Chinês. A quantidade de lixo de uma cidade é muito mais que a capacidade de um caminhão. Portanto, um caminhão só não pode fazer o serviço de uma cidade inteira numa viagem. O caminhão deve voltar ao depósito de lixo cada vez que está cheio, e, depois de descarregar, deve novamente continuar o serviço. Chamamos de "roteiro" o ciclo que percorre um caminhão na sua viagem para uma coleta igual à sua capacidade, e de "jornada", o ciclo que um caminhão percorre num dia de trabalho. Obviamente, uma jornada é composta de um ou mais roteiros. Isto significa que devemos achar um conjunto de ciclos na rede tal que a quantidade de lixo de cada ciclo seja igual à capacidade de um caminhão. Normalmente, são utilizados vários caminhões para fazer o serviço da coleta de uma cidade. Então, temos que determinar o plano de coleta de cada caminhão, de modo a minimizar o custo total de coleta.

Nesse capítulo será considerado o problema da coleta de lixo usando m-veículos.

Como foi descrito, o algoritmo do problema do Carteiro Chinês fornece uma solução ótima para um veículo e um roteiro único. Veremos que este algorítmo ainda pode usar uma solução boa (mas não necessariamente ótima) para o caso geral onde temos *m*-veículos e a jornada de cada veículo é composta de vários roteiros.

6.2 O Problema da Coleta de Lixo Usando *m*-Veículos

O objetivo principal é minimizar a distância (ou custo de percurso) total percorrida pelos veículos, e na maioria das vezes, como uma consequência, minimizar o número de veículos necessários para fazer o serviço total da cidade.

Nessa seção consideramos métodos heurísticos que geram soluções aproximadas às ótimas. Todos os algoritmos existentes para resolver este problema podem ser classificados em dois grupos:

Algoritmos de "Divisão Primeiro - Roteiro Depois" e algorítmos de "Roteiro Primeiro - Divisão Depois" [3].

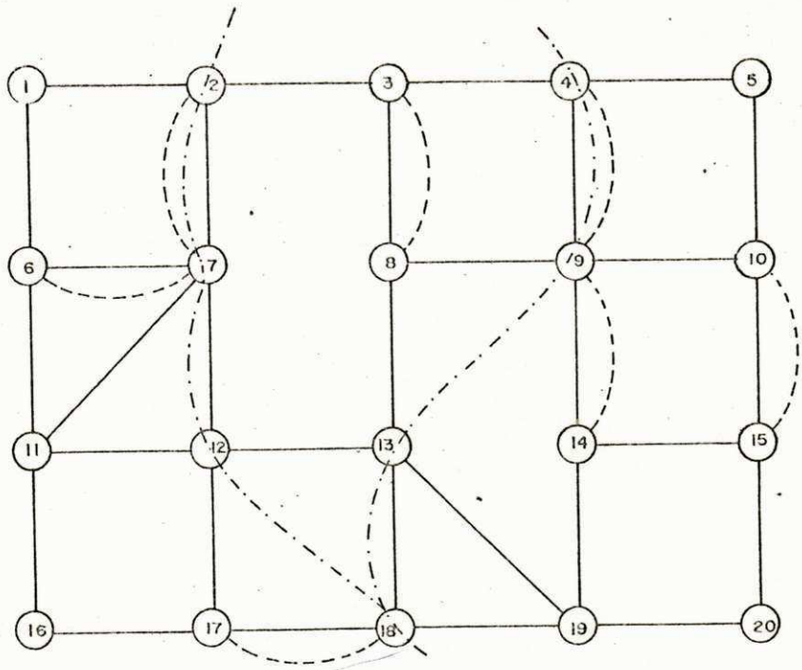
6.2.1 O Procedimento de "Roteiro Primeiro - Divisão Depois"

Neste método, primeiramente, sem levar em consideração o número de veículos, resolvemos o problema para um só roteiro, isto é, construir um ciclo Euleriano envolvendo a rede inteira ("roteiro gigante"). Depois dividimos o ciclo em um conjunto de ciclos menores para *m*-veículos (*m* não é conhecido), tal que cada ciclo seja um roteiro viável. Um roteiro é chamado viável, se a quantidade de lixo nele não for maior que a capacidade de um veículo. Às vezes, uma restrição de tempo de percurso também é considerada.

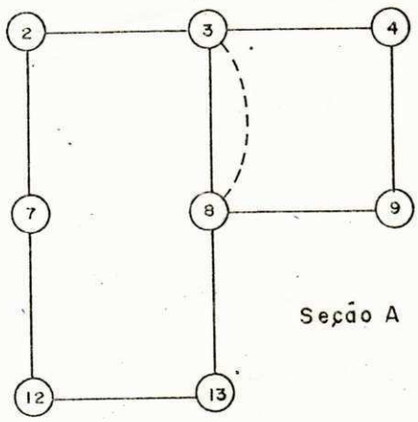
Numa boa divisão, não devem ser criados nós de grau ímpar, por

par, porque isto implicará novas duplicações entre esses nós de grau ímpar. Felizmente, não é difícil dividir um ciclo Euleriano em ciclos menores (como foi descrito na prova da suficiência do teorema de Euler no capítulo 3, um ciclo Euleriano é composto de um conjunto de ciclos menores). O que devemos fazer é dividir o grafo em seções menores, assegurando que cada seção tenha uma fronteira contínua, e que para qualquer nó na fronteira as ligações coincidentes no nó sejam divididas em número par pela fronteira.

Agora consideremos o procedimento com um exemplo. Suponhamos que o grafo da Figura 6.1(a) representa uma rede de ruas, e que o comprimento de cada ligação mostra a distância real, e ainda para facilitar mais, a quantidade de lixo associado a cada ligação seja igual a uma tonelada, e a capacidade dos caminhões seja igual a 12 toneladas. Primeiramente, construimos um "roteiro gigante" nesse grafo; as duplicações apropriadas para construir tal roteiro são mostradas pelas linhas tracejadas nessa figura. Agora dividimos o "roteiro gigante" (que é um ciclo Euleriano) em ciclos menores de modo que a quantidade de lixo de cada ciclo não seja maior que a capacidade de um caminhão. Conforme o procedimento, uma divisão válida pode ser a que é mostrada pelas linhas interrompidas na figura. Na Figura 6.1(b) as seções são mostradas separadamente. Podemos observar que cada seção contém um ciclo Euleriano e conforme a suposição que a quantidade de lixo de cada ligação é uma tonelada, a quantidade do lixo de cada seção seria menor ou igual a 12 toneladas (a capacidade de um caminhão), então cada seção tem um roteiro viável. Na seguinte tabela esses roteiros são construídos pelas sequências de nós.

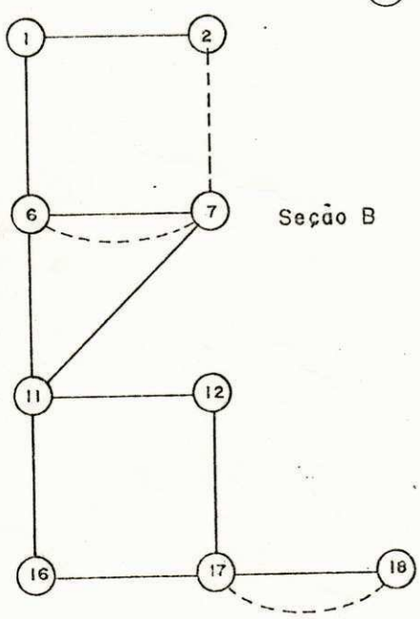


(a)



Seção A

(b)



Seção B

Seção C

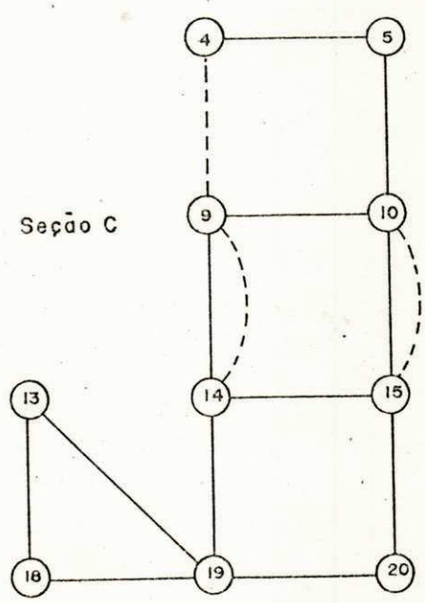


Figura 6.1 O exemplo do método "Roteiro Primeiro Divisão Depois"

Roteiro	Ordem de nós	a quantidade de lixo
A	2, 3, 8, 9, 4, 3, 8, 13, 12, 7, 2	9
B	18,17,16,11,6,7,2,1,6,7,11,12,17,18	10
C	18,19,20,15,10,9,14,15,10,5,4,9,14,19,13,18	12

Se verificarmos que em algumas seções a carga é maior que a capacidade de um veículo e em algumas outras é muito menor, podemos modificar as fronteiras através de desvio de ciclos de uma seção para outra. Por exemplo, se na Figura 6.1(b) achamos que a carga da seção C é muito grande e a da seção A é muito pequena, então podemos desviar o ciclo (18, 19, 13, 18) da seção C para a A. Podemos continuar as modificações dessa maneira até ficarmos satisfeitos com a distribuição de cargas.

Se a rede é dividida em k seções (ou k roteiros) e cada veículo pode fazer o serviço de n roteiros, então o número de veículos necessários m seria.

$$m \geq k/n,$$

onde m é o menor valor inteiro igual ou maior do que o quociente. Obviamente, isto é no caso de que os caminhões tenham igual capacidade.

A solução de "Roteiro Primeiro - Divisão Depois" não é ótima por causa de duas razões: primeira, a técnica de divisão de grafo em seções é heurística; segunda, o depósito de lixo não foi considerado no grafo. No problema real, cada vez que o serviço de um roteiro é completo, o veículo deve partir para o depósito e novamente voltar. Numa solução ótima essas distâncias também devem ser minimizadas.

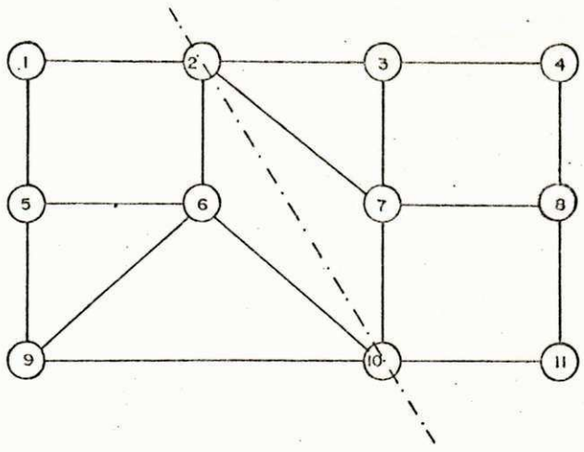
Excluindo o depósito de lixo no grafo, obtemos uma flexibilidade na decisão de onde começar as divisões do "roteiro gigante", proque normalmente temos muitas opções para definir um conjunto de roteiros, enquanto que com a inclusão do depósito isto não acontece.

6.2.2 O Procedimento de "Divisão Primeiro - Roteiro Depois"

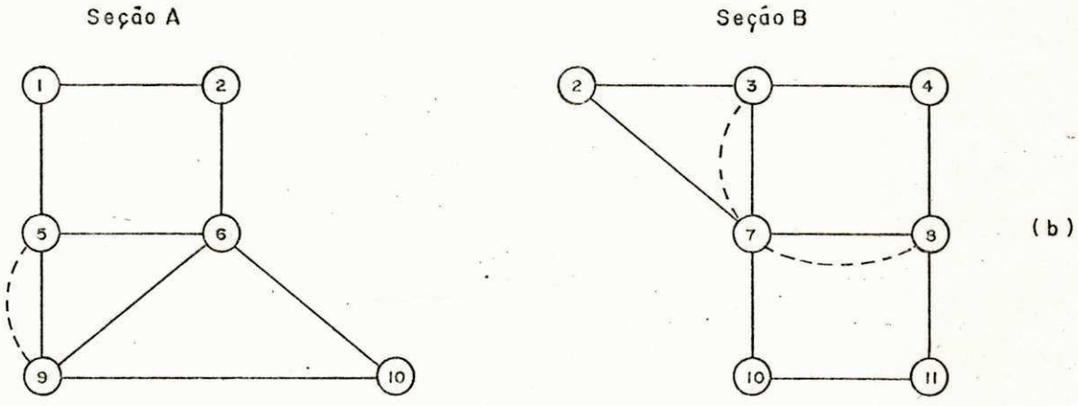
No método "Divisão Primeiro - Roteiro Depois", primeiramente dividimos a rede em seções de modo que a carga de cada seção seja aproximadamente igual à capacidade de um veículo e depois em cada uma das seções aplicamos o algoritmo do problema do carteiro Chinês e encontramos ciclos Eulerianos. Cada ciclo Eulariano seria um roteiro para um veículo. Na divisão da rede, como no procedimento anterior, deve-se evitar que não sejam criados nós de grau ímpar naqueles que eram originalmente do grau par.

Como exemplo consideremos o grafo da Figura 6.2 (a). Suponhamos ainda que o comprimento de cada ligação mostra a distância real e que a quantidade de lixo de cada ligação é uma tonelada. Nessa figura o grafo é dividido em duas seções *A* e *B*. Aplicando o algoritmo do problema do Carteiro Chinês em cada uma das seções, encontramos as duplicações ótimas mostradas na Figura 6.2(b). Agora cada uma das seções dessa figura contém um roteiro viável (como no exemplo anterior, a capacidade de um caminhão são 12 toneladas).

Pelas mesmas razões mencionadas no procedimento anterior, geralmente não consideramos o depósito de lixo na rede.



(a)



(b)

Figura 6.2: O exemplo do método "Divisão Primeiro - Roteiro Depois"

6.2.3 Comparação dos dois Métodos

Agora desejamos comparar os dois métodos discutidos e mostrar qual utilizar: se "Roteiro Primeiro - Divisão Depois" ou "Divisão Primeiro - Roteiro Depois".

Podemos verificar que, quando o depósito não é incluído, o procedimento "Roteiro Primeiro - Divisão Depois" sempre precisa de menos duplicações que o outro, porque as duplicações no primeiro método são ótimas globais, mas no segundo método a solução é ótima local. Apesar desse fato, a escolha de um dos dois métodos é mais uma decisão administrativa que uma decisão algoritmica. Normalmente, ci

dades têm divisões administrativas e, muitas vezes, não é desejável a interferência dessas divisões através de roteiros de coleta de lixo.

Uma combinação dos dois métodos discutidos dá resultados melhores. Nas divisões administrativas, as cidades são divididas em bairros e, muitas vezes, a carga de um bairro é mais que a capacidade de um caminhão. Então um procedimento aconselhável é primeiro dividir a cidade em bairros (de acordo com as divisões administrativas), e depois em cada bairro aplicar o procedimento "Roteiro Primeiro - Divisão Depois".

6.3 Alguns Detalhes na Implementação do Algoritmo do Problema do Carteiro Chinês para resolver o Problema da Coleta de Lixo.

Os algoritmos apresentados no capítulo 4 e as técnicas consideradas no presente capítulo provêem uma base para uma solução do problema real da coleta de lixo. Nessa seção consideramos os detalhes de sua implementação.

Admitimos que as seções são determinadas de acordo com as técnicas discutidas anteriormente, agora deve ser determinado um roteiro, dentro de cada seção, que percorra, pelo menos uma vez, a custo mínimo, as ruas onde se faz a coleta.

A definição de custo é importante. Um veículo pode percorrer um segmento de rua, para coletar o lixo, ou simplesmente percorrer a rua para chegar a um ponto onde faz a coleta. Um princípio importante é que o custo total referente às passagens que têm coleta, é fixo. Então o custo que deve ser minimizado é o referente às passagens que não têm coleta. Os custos associados a essas passagens inativas são de dois tipos: Os custos de mão de obra (que geralmente

são função de tempo), e os custos do veículo (que geralmente são função da distância). Podemos associar a cada segmento de rua um custo de percurso que é a soma desses dois custos e representa o custo da passagem sem coleta. Muitas vezes, podemos usar a distância de cada segmento para determinar o custo de percurso, porque normalmente o tempo de passagem também é função de distância.

Devemos notar que o custo de percurso associado a cada rua não precisa ser necessariamente um custo verdadeiro; é suficiente que o valor seja proporcional ao custo verdadeiro de tal forma que os custos de percursos de todas as ruas sejam relativamente corretos.

Há um outro custo também que é associado ao tipo de pavimentação de rua. Se a pavimentação de todas as ruas da rede é uniforme, podemos dispensar esse custo. Mas se elas não são uniformes, então podemos definir um peso associado a cada tipo de pavimento, de modo que o produto da distância de cada rua e o peso associado à sua pavimentação representem o custo de percurso da rua. Normalmente, para as ruas bem pavimentadas, esse peso pode ser definido igual a um e para as ruas mal pavimentadas ou não pavimentadas terá um valor aumentado, a ser definido pelo serviço público implicado.

Um caso que ocorre frequentemente na coleta de lixo é quando existem alguns segmentos das ruas (ramos) nos quais a coleta não é necessária. Tais ramos serão percorridos somente se eles se encontram num caminho mais curto acoplando dois nós de grau ímpar. Além disso, o grau de cada nó é calculado considerando somente os ramos que precisam do serviço. Em outras palavras, esses ramos não estão incluídos no grafo, quando estamos calculando o grau dos nós, mas os mesmos são usados na determinação dos caminhos mais curtos entre nós de grau ímpar. Assim, o algoritmo de acoplamento determina que

numa solução ótima tais ramos devem ser pecorridos ou não. Como exemplo, suponhamos que o grafo da Figura 6.3(a) representa uma rede de ruas; os pesos representam os custos de percurso. Queremos determinar um ciclo de custo mínimo nesse grafo, que percorre todos os ramos pelo menos uma vez, exceto os ramos (2,3) e (4,8).

Eliminando esses dois ramos do grafo, os nós do grau ímpar serão os nós 4 e 5. As duplicações ótimas para esse exemplo são mostradas na Figura 6.3(b). Embora o ramo (4,8) não precise ser percorrido, ele é usado para ligar os nós de grau ímpar, enquanto que para o ramo (2,3) isto não acontece.

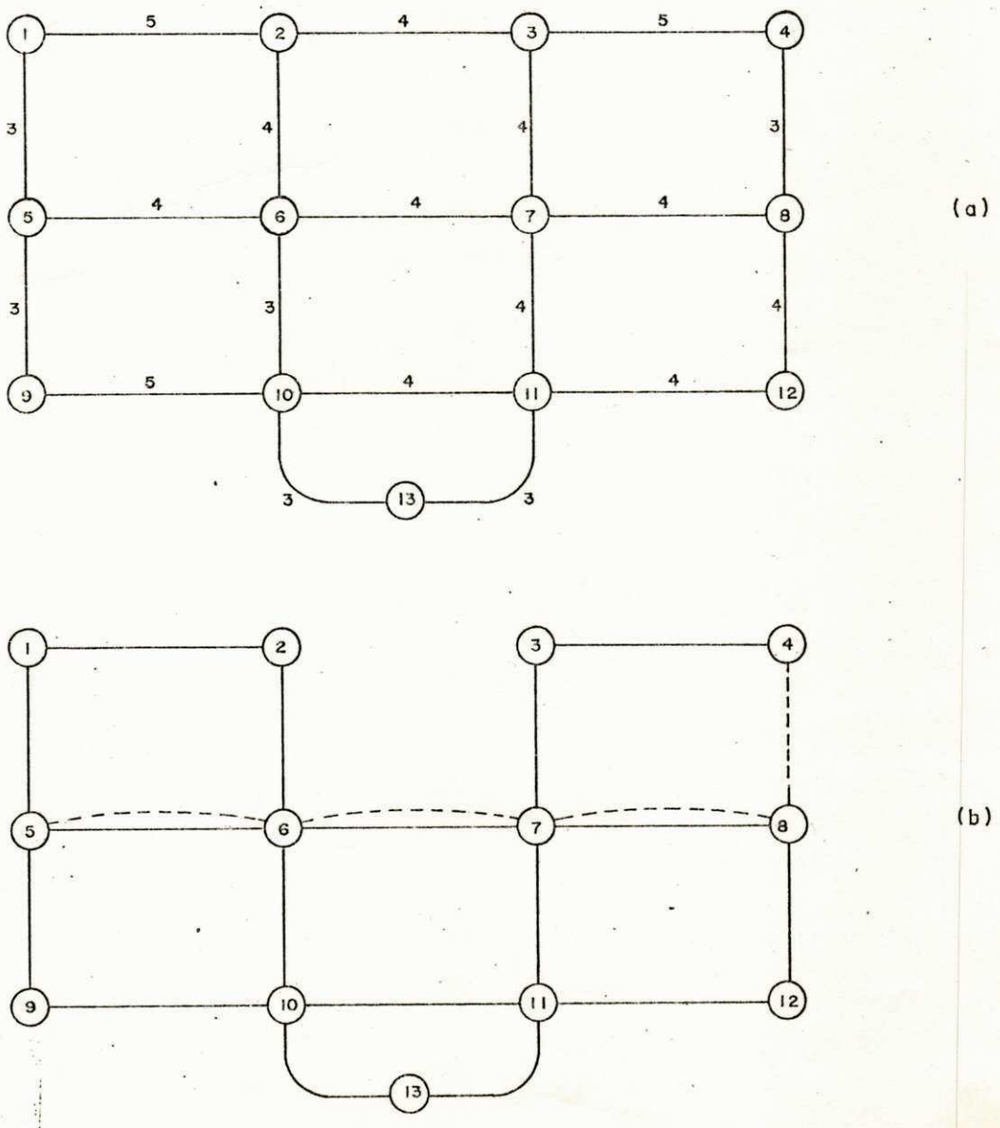


Figura 6.3

Este procedimento pode falhar, se a omissão dos ramos que não precisam ser pecorridos, resulte um grafo não conexo. No exemplo da Figura 6.3, isto acontece se o ramo (3,7) também não precise ser percorrido.

Uma outra complicação no problema da coleta de lixo é o caso de que algumas ruas devem ter coleta em ambos os lados. Normalmente em ruas reisdenciais, numa passagem só ambos os lados podem ser coletados, porém isto não é viável em ruas movimentadas, porque prejudica o tráfego. Consequentemente, essas ruas devem ser percorridas duas vezes.

Felizmente, este caso pode ser levado em consideração pelo algoritmo do problema do Carteiro Chinês simplesmente adicionando um nó e duas ligações, para cada ligação que deve ser percorrido duas vezes. A maneira de combinar essas ligações adicionais com a ligação original é mostrada na Figura 6.4. Suponhamos que na Figura 6.4(a) a ligação (i,j) deve ser percorrida duas vezes; então definimos um nó adicional k e duas ligações adicionais (i,k) e (j,k) (que não existem no grafo original). A Figura 6.4(b) mostra como devem ser combinados o nó e as ligações adicionais com a original. Notamos que a soma dos custos de percurso das ligações adicionais deve ser igual ao custo de percurso da original. Agora o algoritmo do problema do Carteiro Chinês encontra uma solução ótima na qual o segmento entre os nós i e j é percorrido pelo menos duas vezes (uma vez pela ligação original e uma vez através das ligações adicionais). Obviamente, quando desenhamos o roteiro, o segmento (i,j) deve ser percorrido uma vez do nó i para o j e outra vez do nó j para o i. Para atender isto, antes da construção do roteiro, determinamos uma direção para o ciclo $(i, j, k, i,)$; em outras palavras, formamos um circuito. Agora, eliminando esse circuito do grafo, no restante ainda existe um ciclo Euleriano; combinando o ciclo com o cir

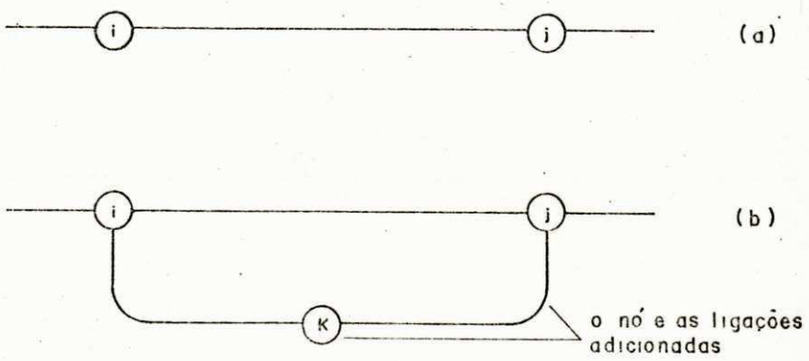


Figura 6.4

cuito, obtemos o roteiro desejado.

A construção de ciclos (também circuitos) Euleriano foi considerada no capítulo 3. É importante notar que, se num grafo existe um ciclo Euleriano, então, geralmente, podemos definir uma grande família de tais ciclos no mesmo, onde cada um deles tem o mesmo custo de percurso. Esta flexibilidade nos permite observar algumas restrições, as quais não conseguimos levar em consideração pelo algoritmo. Por exemplo, podemos construir um roteiro de tal maneira que o serviço das ruas mais movimentadas seja realizado nos horários permitidos. Uma outra restrição que não é consderada no algoritmo, são as interseções nas quais o retorno à esquerda é proíbido. Na maioria dos casos podemos encontrar facilmente um roteiro que observe tais restrições.

Raras vezes, não se consegue achar um ciclo Euleriano que satisfaça as restrições adicionais. Em tais casos, devemos considerar um aumento do custo de percurso do roteiro; em outras palavras, devemos criar algumas outras duplicações dos ramos no grafo, de modo que, no grafo resultante possamos observar as restrições. De

qualquer maneira, o grau de todos os nós do grafo deve continuar sendo par. Isto implica que as novas duplicações também devem formar ciclos completos, ou que os ramos duplicados e não duplicados sejam trocados num ciclo.

CAPÍTULO VII

RESULTADOS COMPUTACIONAIS

Foi incluída neste trabalho a codificação de um programa, em linguagem WATFIV para resolver o problema de acoplamento com peso num grafo qualquer. A codificação é baseado no algoritmo da seção 5.3.3. Como foi citado anteriormente, o programa também pode resolver o problema do acoplamento nos casos particulares mencionados no Capítulo 5. Realmente, nesses casos o volume dos cálculos é reduzido. Por exemplo, no caso do problema do acoplamento de cardinalidade máxima (i. e. quando todos os pesos dos ramos são unitários), o programa não entra na parte da solução dual. Ainda, no problema do acoplamento bipartido, não podem ser formadas flores, porqeu num grafo bipartido, não existem ciclos de cardinalidade ímpar. Consequentemente, nesses casos, o tempo de execução do programa serpa reduzido.

Foi codificado um outro programa, para resolver o problema do Carteiro Chinês, no caso de um grafo não direcionado, no qual o programa do acoplamento com peso é utilizado como um subprograma. Como foi descrito no capítulo 4, esse programa também pode ser utilizado para o problema do Carteiro Chinês no caso de um grafo direcio

nado e em alguns casos particulares de um grafo misto (quando o grafo misto tem poucos ramos direcionados, distribuídos dispersamente no grafo). A eficiência desse programa foi testada pela solução de um problema real de coleta de lixo no Bairro dos Estados da cidade de João Pessoa.

O Bairro dos Estados é composto de 309 segmentos de ruas e 180 interseções. das quais 80 são interseções ímpares. A Figura 7.1 é um mapa desse bairro. O serviço de coleta do lixo, nesse bairro autualmente é feito por um caminhão e através de dois roteiros.

O primeiro passo na solução do problema foi a representação gráfica do mapa. Isto foi feito designando um nó a cada interseção e um ramo a cada segmento de rua. Então, o grafo correspondente tem 180 nós, dos quais 80 são de grau ímpar, e 309 ramos. Como todas as ruas do bairro tem tráfego nos dois sentidos, então todos os ramos são não-direcionados.

O custo de percurso de cada segmento foi calculado em função do comprimento do segmento. Nesse bairro encontrou-se variedade de tipos de pavimentação das ruas. As ruas são sem calçamento ou calçadas, com diferentes qualidades de calçamento. Portanto, elas foram classificadas em categorias diferentes, e a cada uma das categorias foi associada um peso, os quais são dados na tabela seguinte:

	Boa	Razoável	Má
Calçada	1,0	1,2	1,4
Não-Calçada	1,2	1,4	1,7

Normalmente, os valores dos pesos são calculados de acordo com as experiências dos setores responsáveis pela coleta do lixo;

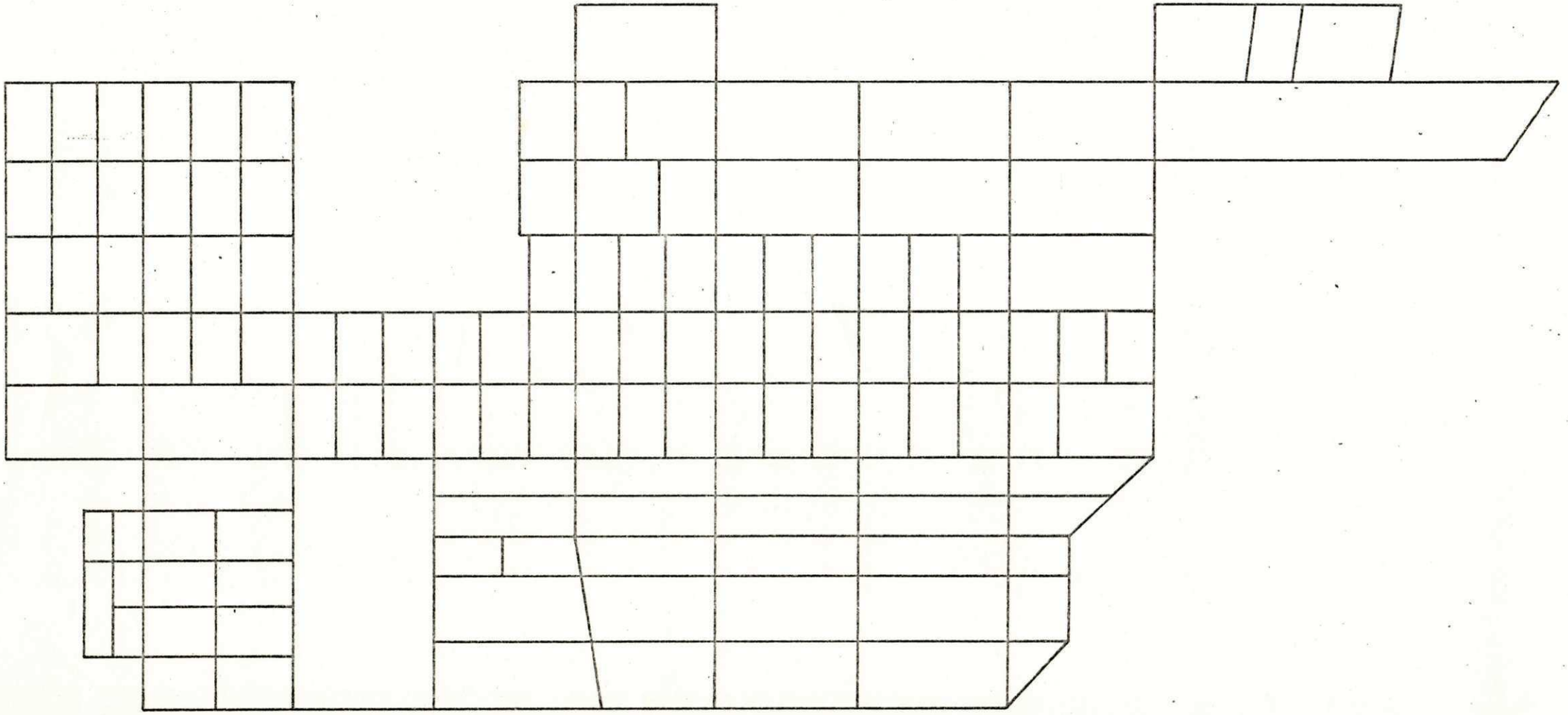


Figura 7.1 O Mapa do Bairro dos Estados da Cidade de João Pessoa

porém, no nosso exemplo são estimativas. O custo de percurso de cada segmento de rua foi calculado, multiplicando o comprimento do segmento pelo peso associado ao seu tipo de pavimento.

A quantidade de lixo nesse bairro atualmente é igual à capacidade de dois caminhões (ou dois roteiros para um caminhão). Portanto, devemos determinar dois roteiros viáveis nesse bairro. Tentamos fazer isto pelo método "roteiro Primeiro - Divisão Depois".

Primeiramente, aplicamos o algoritmo do problema do Carteiro Chinês ao grafo correspondente, para encontrar as duplicações ótimas e depois dividimos o grafo resultante (incluidas as duplicações) em duas seções, de modo que a quantidade de lixo de cada uma seja igual ou menor do que a capacidade de um caminhão.

Os dados de entrada do programa são os seguintes:

(a) Um cartão especificando respectivamente o número dos nós e o número dos ramos e também a palavra 'MINIMAL' (essa palavra especifica que o subprograma de acoplamento deve procurar um acoplamento com peso mínimo). Os dados são perfurados em formato livre.

(b) Cada segmento de rua (ramo) é especificado por três números: os dois primeiros são os números dos nós terminais do ramo e o terceiro é o seu curso de percurso. Deve-se prover dados para todos os ramos existentes no grafo. Cada cartão pode conter dados de mais de um ramo. Esses dados também são em formato livre.

Com esses dados de entrada o programa começa identificando todos os nós de grau ímpar no bairro. Depois determina os caminhos

mais curtos entre todos esses nós e as distâncias correspondentes. Essas informações entram no subprograma de acoplamento com peso e, finalmente, a saída do programa é uma listagem dos pares de nós de grau ímpar, que devem ser conctados através dos caminhos mais curtos entre eles, para obter um ciclo Euleriano de custo mínimo no grafo.

A partir da saída do programa podemos identificar os caminhos que devem ser duplicados no grafo original. Apesar do programa só determina os pares de nós a serem conectados e não identificar quais são os ramos que devem ser duplicados, não é dificil encontrar tais caminhos entre os pares de nós dados pelo programa. Geralmente os caminhos mais curtos numa solução ótima são de cardinalidade igual ou menor qeu 2, e em raras vezes pode ser maior. Então facilmente podemos identificar e duplicar tais caminhos.

A Figura 7.2 mostra as duplicações ótimas para o exemplo do Bairro dos Estados. Observa-se que de 40 caminhos duplicados, 31 são de cardinalidade um e somente 9 são de cardinalidade dois ou maior. A solução desse problema levou 43 minutos no IBM/370-145, dos quais aproximadamente 30 minutos foram para encontrar os caminhos mais curtos entre todos os nós de grau impar, e 13 minutos para a execução do subprograma de acoplamento.

Como a carga de lixo do bairro é igual à capacidade de 2 caminhões, então o grafo da Figura 7.2 deve ser dividido em duas seções, tais que a quantidade do lixo, de cada uma não seja maior que a capacidade de um caminhão. I. e. a segunda etapa do método "Roteiro Primeiro - Divisão Depois". Como nós não tivemos dados sobre a quantidade de lixo de cada rua, então não podemos fazer tal divisão para o exemplo. Mas uma divisão típica pode ser a que é mostrada com a linha interrompida na Figura 7.2.

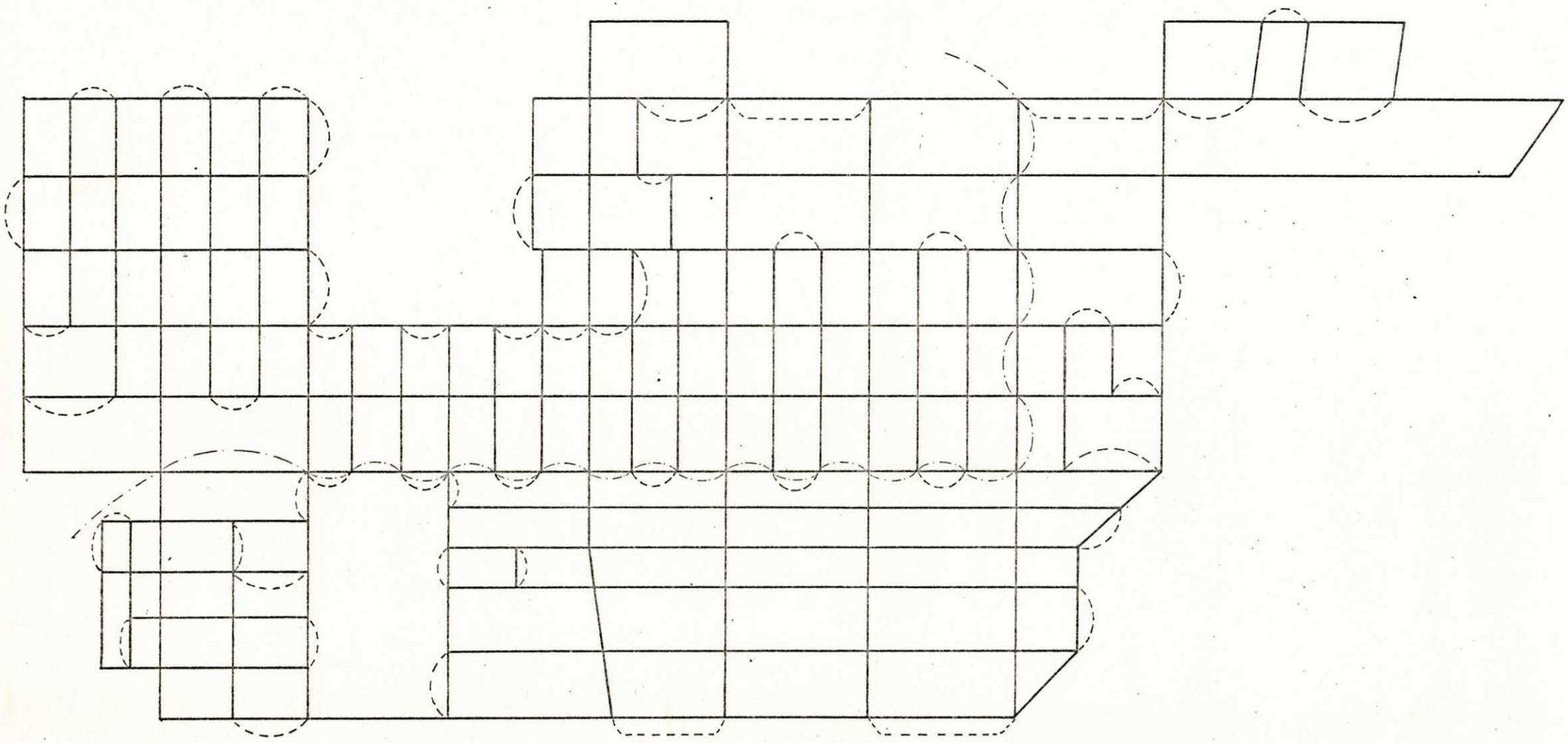


Figura 7.2 : A Solução do Problema do Carteiro Chinês para o Bairro

Linhas - - - - - - representam as passagens adicionais

Linhas -.-.-.-.- divide o mapa em duas seções

O último passo na solução do problema é determinar o roteiro do veículo em cada seção. Simplesmente nós devemos fazer um ciclo Euleriano em cada seção do grafo. A prova da suficiência do teorema 3.1 forneceu um procedimento para fazer tal ciclo. Como foi descrito na seção 6.3, normalmente temos muitas opções para definir um ciclo Euleriano e essa facilidade nos permite observar certas restrições adicionais (Veja o capítulo 6). Por isso, esse passo não foi programado e deve ser feito manualmente.

Certamente, um modo de testar a eficácia do programa é comparar os seus resultados com o roteiro existente para a coleta de lixo nesse bairro. O roteiro do caminhão foi seguida num dia de trabalho nesse bairro e foi traçado no mapa. A Figura 7.3 mostra o roteiro resultante, onde as passagens adicionais são mostradas pelas linhas tracejadas.

Usando a solução mostrada na Figura 7.2, O Bairro dos Estados com 33,92 Km de ruas, precisa de 4,73 Km de duplicações. Porém da Figura 7.3, atualmente 11,31 Km são duplicados. A solução representa uma economia de 58 por cento dos percursos não necessários, e 14,5 por cento da distância total percorrida.

Uma vez que os roteiros de coleta de lixo são escolhidos aleatoriamente em todos os birros da cidade de João Pessoa, sem dúvida nenhuma será possível alguma economia também para eles.

Na implementação corrente do programa, no computador IBM 370/145 do Centro de Ciências e Tecnologia da UFPb, as variáveis foram dimensionadas para solução de problemas constituídos até 800 segmentos de rua e 350 interseções aproximadamente, desde que o número de interseções ímpares não seja maior que 160. Nessa implementação, o programa precisa de aproximadamente 357 k bytes de memó

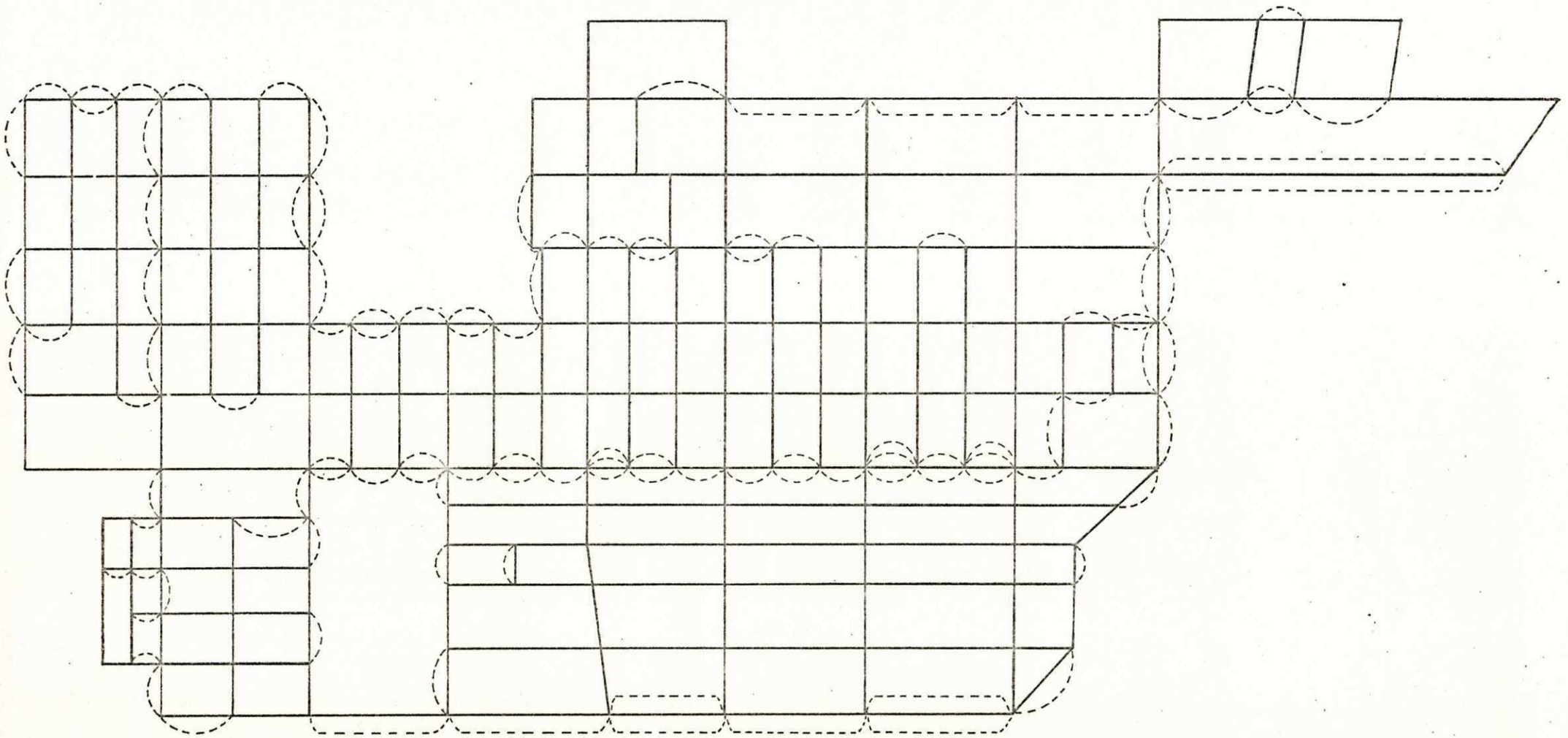


Figura 7.3 : As Passagens Adicionais na Solução Existente para a Coleta de Lixo do Bairro

ria para as variáveis dimensionadas. No caso geral as variáveis po dem ser dimensionadas da seguinte maneira:

```
REAL*4 G(i,j), U(i)
INTEGER*2 C(n,n), W(m,3), DEG(n), B(i), SCAN(i),
          LAB(2,i), MISS(2,i), TRACE(2,i), X(i),
          PATH(i), SPEC(2,i), TCAN(i),
```

onde n é o número dos nós do grafo, i é o número dos nós de grau ím par e m é o número dos ramos.

CAPÍTULO VIII

SUMÁRIO E CONCLUSÕES

Neste trabalho, foi considerado o problema da distribuição de serviços públicos. O esforço principal foi concentrado na solução do problema da determinação do roteiro dos veículos para uma classe de problemas nos quais o serviço deve ser distribuído ao longo das ruas. A coleta de lixo, a limpeza de ruas pelas vassouras mecânicas, a distribuição de carta etc são exemplos de tal classe de problemas.

O modelo do problema do Carteiro Chinês foi usado como solução. Esse modelo foi considerado em três casos diferentes: no caso de que todas as ruas são de sentido duplo; quando todas as ruas são de sentido único e no caso de que algumas ruas são de sentido único e algumas outras não o são. Para o primeiro e o segundo casos foram apresentados algoritmos muito eficientes, enquanto o terceiro é um problema não resolvido até agora.

Devido à simplicidade dos princípios dos algoritmos apresentados, muitos problemas podem ser resolvidos manualmente e podem ser encontrados soluções pelo menos razoáveis.

A codificação de um programa para resolver o problema do acoplamento com peso num grafo qualquer foi também incluída neste trabalho. Ela foi utilizada como um subprograma do programa codificado para o problema do Carteiro Chinês. O programa foi testado por um exemplo real de coleta de lixo; a comparação dos resultados obtidos com a solução existente mostrou que o mesmo pode ser muito bem aplicado em tais problemas.

A versão atual permite a solução de problemas de porte relativamente grande. O tempo de execução do programa é razoável, porém por alguns refinamentos poderia ser diminuído.

É muito importante notar que o algoritmo do problema do Carteiro Chinês fornece uma solução ótima, mas geralmente o problema da distribuição de serviços públicos não tem uma solução ótima; as técnicas apresentadas podem ser utilizadas como instrumentos para encontrar uma solução boa. Esse fato foi considerado explicitamente na implementação do algoritmo do problema do Carteiro Chinês para resolver o problema da coleta de lixo discutido no capítulo 7. Podemos dizer que as técnicas existentes para resolver tais problemas de serviços públicos ainda devem ser usadas em conjunto com procedimentos manuais e julgamentos intuitivos.

Ainda existe uma série de problemas na distribuição de serviços públicos para serem resolvidos, dos quais os mais importantes são:

(a) O problema da determinação de roteiro numa rede na qual algumas ruas são de sentido único e algumas outras não o são.

(b) O problema da determinação de roteiros numa rede para *m*-

veículos.

(c) O problema da localização das facilidades / determinação de roteiros, no qual a interdependência entre os roteiros e a localização das facilidades seja considerada explicitamente.

REFERÊNCIAS

1. Berlin, G.N. "Computerized Districting for Residential Refuse Collection" (1974)

2. Bodin, L.D. "A Computerized Model for Municipal Street Sweeping Operation" Part 1 (1973).

3. Bodin, L.D. "A Taxometric Structure for Vehicle Routing and Scheduling Problems".

4. Christofides, N. "Graph Theory an Algorithmic Approach" Academic Press (1975)

5. Edmonds, J. "Maximum Matching and a Polyhedron with 0,1 Vertices". Research National Bureau of Standard B, 1 and 2 (1965)

6. Edmonds, J. "The Chinese Postman Problem" Operations Research Volume 13, Supllement 1 (1965).

7. Edmonds, J. and Johnson, E.L. "Matching, Euler Tours and the Chinese Postman" (1972).

8. Euler, Leonhard (1736) "Solutio Problematis ad Geometriam Situa Pertinentis" Comment. Academic Sci. 1. Petropolitanae 8.

9. Ford, L.R., Jr. and D.R. Fulkerson "Flows in Networks" Princenton University Press, Princeton (1962).

10. Garfinkel, R.S. and Nemhauser, G.L. "Integer Programming" John Wiley & Sons (1972).

11. Glover, F. "Finding an Optimal Edge Covering Tour of a Connected Graph" ORC 67 - 13, Operations Research Center, University of California Berkeley (1967).

12. Hougland, E.S. and Turner, W.C. and Case, K.E. "Routing of Solid Waste Collection Vehicles Development and Aplication".

13. Hu, T.C. "A Decomposition Algorithm for Shortest Path in a Network" Operations Research, 16 (1968).

14. Lawler, E.L. "Combinatorial Optimization: Networks and Matroids" Holt, Rinehart and Winston (1965).

15. Liebman, J.C. "Routing of Solid Waste Collection Vehicles" (1973).

16. Liebman, J.C. and Marks, D.H. "Location Models: A Solid waste Collection Example"

17. Meio-ko Kwan "Graphic Programming Using odd or Evem Points" Chinese Math.1 (1962).

18. Orloff, C.S. "A Fundamental Problem in Vehicle Routing" Networks, 4 John Wiley & Sons (1974).

19. Orloff, C.S. "Routing a Fleet of M Vehicles to/from a Central Facility" Networks, 4 John Wiley & Sons (1974).

20. Stricker, R. "Public Sector Vehicle Routing: The Chinese Postman Problem" Master Thesis, MIT (1970).