3.4 Formulações de problemas clássicos 172
3.4.1 Problemas da mochila 172
3.4.2 Problemas de corte 174
3.4.3 Problemas de designação 178
3.4.4 Problemas de cobertura, partição e empacotamento de conjuntos 181
3.4.5 Problemas de caixeiro-viajante 186

1) Problema da Mochila

Um viajante dispõe de n itens que deve selecionar para acomodar em uma mochila que está sendo preparada para uma viagem. O peso do item j é igual a_j e o "lucro" obtido, caso ele seja selecionado, é igual a c_j, para $j = 1, \dots, n$. Quais itens devem ser selecionados, sabendo-se que o peso máximo que o viajante pode carregar na mochila é igual a b?

Caso (1): os itens podem ser fracionados e não há limite na quantidade selecionada

<u>Variáveis</u>: $x_j \geq 0 \rightarrow$ item j pode ser fracionado

$$\max \quad Z = \sum_{j=1}^{n} c_j x_j$$

$$\text{sa} \quad \sum_{j=1}^{n} a_j x_j \leq b$$

$$x_j \geq 0, \quad j = 1, \dots, n$$

Caso (2): os itens podem ser fracionados e no máximo uma unidade de cada item pode ser selecionada

<u>Variáveis</u>: $0 \leq x_j \leq 1 \rightarrow$ item j pode ser fracionado, limitado a uma unidade

Caso (3): os itens não podem ser fracionados e no máximo uma unidade de cada item pode ser selecionada

<u>Variáveis</u>: $x_j \in \{0, 1\} \rightarrow x_j = 1$ se o item j é acondicionado na mochila

Múltiplas mochilas

Sejam n itens, $j = 1, \dots, n$ com lucro c_j e peso a_j, e m mochilas de capacidade b_i $i = 1, \dots, m$. Queremos carregar as m mochilas de forma a maximizar o lucro total. Cada item pode entrar em uma única das várias mochilas/caminhões/contêineres/... (n»»m).

<u>Variáveis</u>: $x_{ij} \in \{0, 1\} \rightarrow x_{ij} = 1$ se o item j é colocado na mochila i

$$\max \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_j x_{ij}$$

$$\text{sa} \quad \sum_{j=1}^{n} a_j x_{ij} \le b_i, \qquad i = 1, \dots, m$$

$$\sum_{i=1}^{m} x_{ij} \le 1, \qquad j = 1, \dots, n$$

$$x_{ij} \in \{0,1\}, \quad i = 1, \dots, m, \quad j = 1, \dots, n$$

Bin Packing **(Empacotamento em mochilas/caminhões/containers/caixas/...)**

Encontrar o menor número de mochilas (m»»n) de capacidade b_i tal que <u>todos</u> os n itens sejam acondicionados sem exceder a capacidade b_i (podemos fazer n = m).

<u>Variáveis</u>: $x_{ij} \in \{0, 1\} \rightarrow x_{ij} = 1$ se a mochila i é atribuída ao item j

$y_i \in \{0, 1\} \rightarrow y_i = 1$ se a mochila i é utilizada

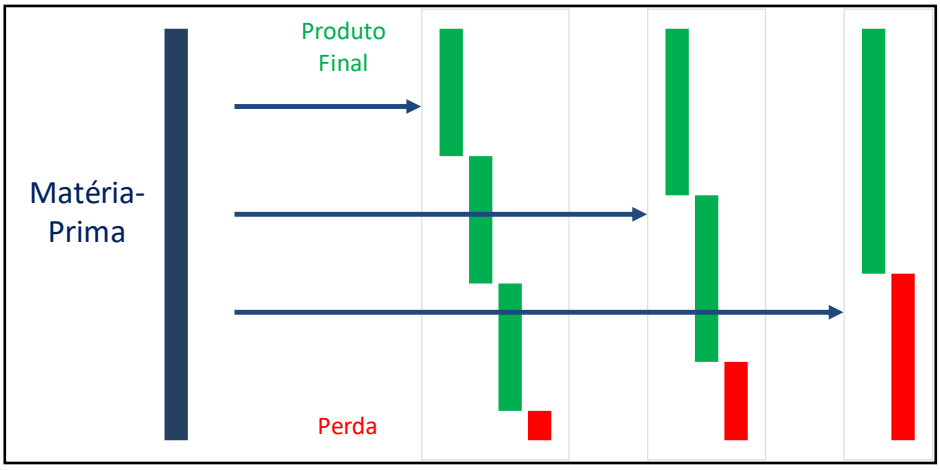
$$\min \quad Z = \sum_{i=1}^{n} y_i$$

$$\text{sa} \quad \sum_{j=1}^{n} a_j x_{ij} \le b_i y_i, \qquad i = 1, \dots, n$$

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1, \dots, n$$

$$x_{ij} \in \{0,1\}, \quad y_i \in \{0,1\}, \quad i = 1, \dots, n, \quad j = 1, \dots, n$$

2) Problema de Corte Unidimensional



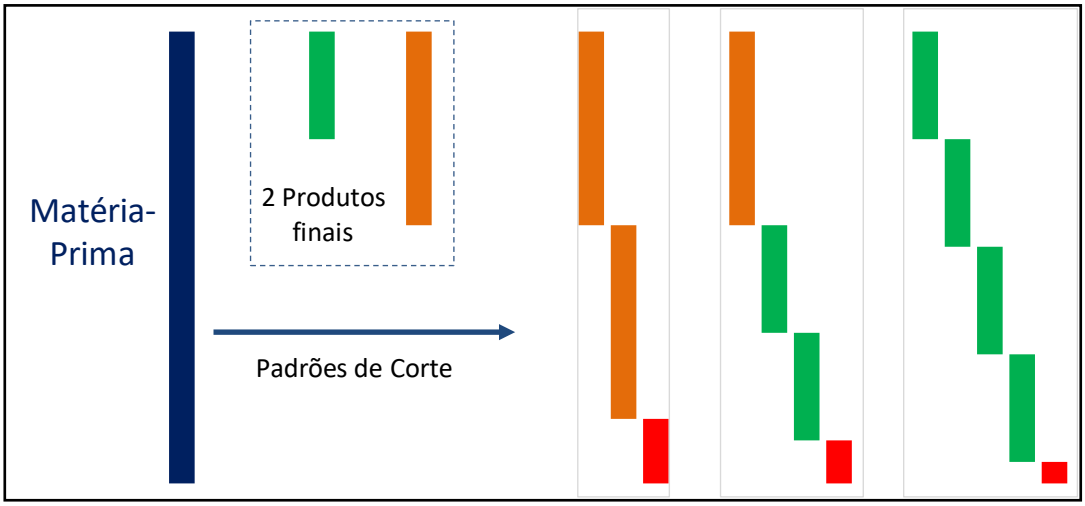
Matéria-prima: a) tubos; b) ... de papel ou têxtil; c) barras; d) varetas ou barras de madeira; e) chapas de aço

Objetivos:

a) minimização da perda total e/ou a quantidade de matéria-prima cortada

b) maximização do número de produtos montados/acabados/fabricados

Uma formulação para o problema de corte (mas não a única) começa com uma lista de n itens, cada uma exigindo D_i unidades ($i = 1, ..., n$). Primeiro construímos o rol de todas as combinações possíveis de cortes (geralmente chamados de "**padrões**").



Seja m o número de todos os possíveis padrões. Associamos a cada padrão uma variável inteira positiva x_j, representando quantas vezes o padrão j deve ser usado, onde $j = 1, \dots, m$. O valor da variável determina o número de unidades de materia-prima que será cortado/fatiado de acordo com o correspondente padrão.

Variáveis: $x_j \in \mathbb{Z}^+ \rightarrow$ quantas vezes o padrão j será utilizado

$$\min \quad Z = \sum_{j=1}^{m} c_j x_j$$

$$\text{sa} \quad \sum_{i=1}^{m} a_{ij} x_j \geq D_i, \quad i = 1, \dots, n$$

$$x_j \in \mathbb{Z}^+, \qquad j = 1, \dots, m$$

onde a_{ij} é o número de vezes que o item i aparece no padrão j e c_j é o custo (geralmente o desperdício) do padrão j. Quando $c_j = 1$, o objetivo minimiza o número de unidades de matéria-prima cortados e, se a restrição para a quantidade a ser produzida for substituída pela igualdade, o problema é o *bin packing*.

3) Problema de Transporte

- Referem-se ao transporte ou distribuição de produtos dos centros de produção (origem) aos mercados consumidores (destino);
- O transporte deve respeitar as limitações de oferta e atender à demanda requisita;
- Em alguns problemas, podem-se usar localidades intermediárias (ou de transbordo): depósitos ou centros de distribuição;
- No problema de transbordo, a quantidade que sai do centro intermediário deve ser igual à quantidade de produto que chega dos centros produtores;
- Modelos de transporte podem representar outras situações: existem n tarefas que precisam ser distribuídas a n pessoas (problema de designação).

Uma empresa produz um produto em m fábricas, para atender a demanda de n locais de demanda. A capacidade de produção da fábrica i é no máximo igual a a_i, i=1,...,m. A demanda da cidade j é igual a b_j, j=1,...,n. Sabendo-se que o custo de envio de uma unidade do produto da fábrica i para o local de demanda j é c_{ij}, determinar a quantidade que deve ser enviada de cada fábrica para cada local de demanda, de modo a minimizar os custos de transporte desta empresa.

<u>Variáveis</u>: $x_{ij} \in \mathbb{Z}^+ \rightarrow$ quantidade transportada do local de oferta i ao consumidor j

$$\max \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{sa} \quad \sum_{j=1}^{n} x_{ij} \leq a_i, \qquad i = 1, \dots, m$$

$$\sum_{i=1}^{m} x_{ij} \geq b_j, \qquad j = 1, \dots, n$$

$$x_{ij} \in \mathbb{Z}^+, \quad i = 1, \dots, m, \quad j = 1, \dots, n$$

4) Problema de Designação

- m agentes (i=1,...,m) e n tarefas (j=1,...,n)
- cada tarefa deve ser realizada por um único agente
- cada agente pode realizar uma única tarefa

$$\max \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{sa} \quad \sum_{j=1}^{n} x_{ij} \leq 1, \qquad i = 1, \dots, m$$

$$\sum_{i=1}^{m} x_{ij} \geq 1, \qquad j = 1, \dots, n$$

$$x_{ij} \in \{0,1\}, \quad i = 1, \dots, m, \quad j = 1, \dots, n$$

Designação generalizada

- m agentes e n tarefas ($m < n$)
- com recursos – por exemplo, o agente i, para executar a tarefa j, utiliza um recurso na quantidade a_{ij}; o recurso, para cada agente é limitado a b_i
- cada agente pode realizar mais de uma tarefa

$$\max \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{sa} \quad \sum_{j=1}^{n} a_{ij}x_{ij} \leq b_i, \qquad i = 1, \dots, m$$

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad j = 1, \dots, n$$

$$x_{ij} \in \{0,1\}, \quad i = 1, \dots, m, \quad j = 1, \dots, n$$

5) Problemas de cobertura, partição e empacotamento de conjuntos

Selecionar subconjuntos de um conjunto inicial de forma a <u>cobrir</u>, <u>particionar</u> ou <u>empacotar</u> o conjunto inicial.

Exemplo[1] de Cobertura: Um hospital de emergências precisa manter médicos de plantão de modo que um médico qualificado esteja disponível para realizar todos os procedimentos que possam ser necessários (há uma lista oficial desses procedimentos). Para cada um dos médicos disponíveis para atuar no plantão, um salário adicional precisa ser pago e os procedimentos que eles podem executar são conhecidos. O objetivo é escolher os médicos para que cada procedimento seja coberto a um custo mínimo.

[1] http://www-personal.umich.edu/~mepelman/teaching/IP/Handouts/Handout2.pdf

	Médico A	Médico B	Médico C	Médico D	Médico E	Médico F
Procedimento 1	√			√		
Procedimento 2	√				√	
Procedimento 3		√	√			
Procedimento 4	√					√
Procedimento 5		√	√			√
Procedimento 6		√				

<u>Representação dos dados</u>: **matriz de incidência**. Com m ($i = 1, \dots, m$) procedimentos e n ($j = 1, \dots, n$) médicos disponíveis, os dados podem ser representados por $A \in \mathbb{R}^{m \times n}$, onde $a_{ij} = 1$ se o médico j pode executar o procedimento i e $a_{ij} = 0$ caso contrário. Além disso, seja c_j, $j = 1, \dots, n$ o salário adicional a ser pago se o médico j atuar no plantão.

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

<u>Variáveis</u>: $x_j \in \{0, 1\} \rightarrow x_j = 1$ se o médico j estiver de plantão

$$\begin{aligned} \min \quad & Z = \sum_{j=A}^{F} c_j x_j \\ \text{sa} \quad & \sum_{j=A}^{F} a_{ij} x_j \geq 1, \quad i = 1, \dots, 6 \\ & x_j \in \{0,1\} \end{aligned}$$

$$\begin{aligned} \min \quad & Z = \sum_{j=1}^{n} c_j x_j \\ \text{sa} \quad & \sum_{j=1}^{n} a_{ij} x_j \geq 1, \quad i = 1, \dots, m \\ & x_j \in \{0,1\} \end{aligned}$$

A restrição impõe a condição de que pelo menos um médico deve executar o procedimento i.

<u>Definições</u>

Sejam dados:

i) um conjunto M;

ii) n subconjuntos M_j de M ($M_j \subseteq M$, $j = 1, \dots, n$); e

iii) pesos dos subconjuntos, c_j, $j = 1, \dots, n$.

a) **Cobertura** (*covering*): uma coleção $T \subseteq \{1, \dots, n\}$ tal que $\cup_{j \in T} M_j = M$.

No exemplo acima, M é o conjunto completo dos procedimentos, M_j é o conjunto de procedimentos que o médico j pode realizar e c_j é o salário do médico j. T é o

conjunto de médicos de plantão; note que deve ser uma cobertura de M. Neste problema estávamos à procura de uma cobertura de custo mínimo.

b) **Empacotamento** (*packing*): uma coleção $T \subseteq \{1, ..., n\}$ tal que $M_j \cap M_k = \emptyset$, $j,k \in T$, $j \neq k$.
Por exemplo, em um dia agitado, pode ser aceitável que alguns pedidos de menor prioridade possam ser adiados para um dia posterior.

c) **Particionamento** (*partition*): uma coleção $T \subseteq \{1, ..., n\}$ que é ao mesmo tempo uma cobertura e um empacotamento.

PLB's

$\min \mathbf{c}^T\mathbf{x}$	$\min \mathbf{c}^T\mathbf{x}$	$\max \mathbf{c}^T\mathbf{x}$
$\mathbf{A}\mathbf{x} \geq \mathbf{1}$	$\mathbf{A}\mathbf{x} = \mathbf{1}$	$\mathbf{A}\mathbf{x} \leq \mathbf{1}$
$\mathbf{x} \in B^n$,	$\mathbf{x} \in B^n$	$\mathbf{x} \in B^n$
Cobertura	**Partição**	**Empacotamento**

Graficamente

6) Problema do Caixeiro Viajante (*Traveling Salesman Problem-TSP*)

Suponhamos que, a qualquer momento em que realizamos uma entrega aos clientes, podemos usar apenas um único veículo e que a capacidade do caminhão não é um problema. Nesse caso, precisamos despachar um único veículo do nosso depósito para n clientes, com o veículo retornando ao depósito após a entrega final. Este é conhecido como problema do caixeiro viajante-TSP.

Portanto, o TSP é o problema de encontrar uma rota de custo/comprimento mínimo através de n locais, de modo que cada local seja visitado exatamente uma vez.

Imagine também, um vendedor que deve visitar n clientes/cidades. Ele começa em uma cidade origem e deve visitar cada uma das outras n cidades exatamente uma vez e depois retornar à cidade origem.

O custo/distância de viajar da cidade i para a cidade j é dado por c_{ij} para todos os pares de cidades. O problema é projetar uma rota/tour de custo/distância mínimo que passa em cada uma das n cidades uma única vez.

Se o custo para viajar da cidade i para a cidade j é igual ao custo para viajar da cidade j para a cidade i ($c_{ij} = c_{ji}$) para todas as cidades, então o problema é simétrico.

Seja o conjunto dessas cidades $N = \{1, \dots, n\}$. Encontre a ordem em que o vendedor deve fazer o *tour* completo em custo/distância/tempo/... mínimo de tal modo que visite todas as cidades.

Então, dado um conjunto de cidades e uma matriz de distâncias entre elas, o TSP consiste em encontrar uma rota que:

- parta de uma cidade origem;
- passe por todas as demais cidades uma única vez;
- retorne à cidade origem ao final do percurso;
- percorra com o menor custo/distância/tempo/... total possível.

Formulação de Dantzig, Fulkerson e Johnson (1954)

<u>Variáveis</u>: $x_{ij} \in \{0, 1\} \rightarrow x_{ij} = 1$ se local j é visitado a partir do local i (x_{ii} não definido)

$$\min Z = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} c_{ij} x_{ij} \quad \text{(minimizar o custo da viagem)}$$

$$\text{sa} \quad \sum_{j=1, j\neq i}^{n} x_{ij} = 1 \quad i = 1, \dots, n \text{ (deixa a cidade i exatamente uma vez)}$$

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1 \quad j = 1, \dots, n \text{ (entra na cidade j exatamente uma vez)}$$

$$x_{ij} \in \{0,1\} \quad \forall i, j$$

Soluções que satisfazem as restrições acima podem levar a subtours/sub-rotas. Para eliminar uma sub-rota S, adicionar, ao PLB original, a restrição

$$\sum_{i\in S}^{n} \sum_{j\in S, j>i}^{n} x_{ij} \leq |S| - 1, \qquad S \subset N, \qquad 3 \leq |S| \leq \left\lfloor \frac{n}{2} \right\rfloor$$

e resolver novamente. Se outras sub-rotas ocorrerem, acrescentar mais restrições. Este processo para quando não ocorrerem sub-rotas.

Formulação de Miller, Tucker e Zemlin-MTZ (1960)

A formulação Miller-Tucker-Zemlin utiliza uma variável extra. A variável é u_i e assume um valor para cada nó, exceto para o depósito. Se um veículo se desloca do nó i para o nó j, o valor de u_j tem que ser maior que o valor de u_i. Portanto, cada vez que um novo nó é visitado, o valor de u_i aumenta.

<u>Variáveis</u>: $x_{ij} \in \{0, 1\} \rightarrow x_{ij} = 1$ se o local j é visitado a partir do local i

$u_i \in \mathbb{Z}^+ \rightarrow$ ordem de visitação do local i

$$\min Z = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} c_{ij} x_{ij}$$

$$\text{sa} \quad \sum_{j=1, j\neq i}^{n} x_{ij} = 1 \qquad i = 1, \dots, n$$

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1 \qquad j = 1, \dots, n$$

$$u_i - u_j + n x_{ij} \leq n - 1 \qquad 2 \leq i, j \leq n, i \neq j$$

$$x_{ij} \in \{0,1\}, \qquad u_i \geq 0, \quad \forall i, j$$

A última restrição também pode ser gerada a partir de $u_j \geq u_i + 1 - n(1 - x_{ij})$. Se $x_{ij} = 1$ então $u_j \geq u_i + 1$. Isto é exatamente o que queremos. Se $x_{ij} = 0$ então temos $u_i - u_j + 1 \leq n - 1$ e, portanto u_j pode assumir qualquer valor acima de u_i.

Por exemplo considere a sub-rota A−B−A, onde A não é o depósito Temos $x_{AB}=x_{BA}=1$. Mas com as restrições MTZ, sem perda de generalidade, se $u_A=2$, então necessariamente, $u_B \geq 3$. E então não podemos ter $u_B+1 \leq u_A$, o que contradiz $x_{BA}=1$.

Lembremos que a última restrição não envolve a origem. Assim, a última restrição irá incluir o nó origem em uma única rota, garantindo que a solução não contenha sub-rotas desconectados da origem.

Exemplo (https://www.dcce.ibilce.unesp.br/~socorro/bienal2006/aula2b_caix_viaj.ppt)

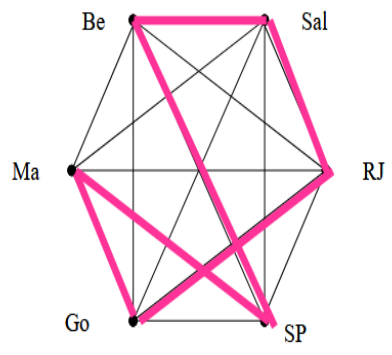


Figura 1 – Reservas e Depósitos a serem visitados

Construção do Modelo:

Elementos conhecidos (dados):

Índices: $i,j = 1,2,3, \ldots 6$ os locais onde as reservas (duas) e os depósitos (quatro) estão situados (RJ,SP,Go,Ma,Be e Sal) respectivamente.

c_{ij} distância entre os locais i e j.

Elementos desconhecidos (variáveis):

$x_{ij} =$ 1 se o local i é visitado imediatamente antes de j
0 caso contrário.

$$\min z = \sum_{i=1}^{6}\sum_{j=1}^{6} c_{ij}x_{ij}$$

sujeito a

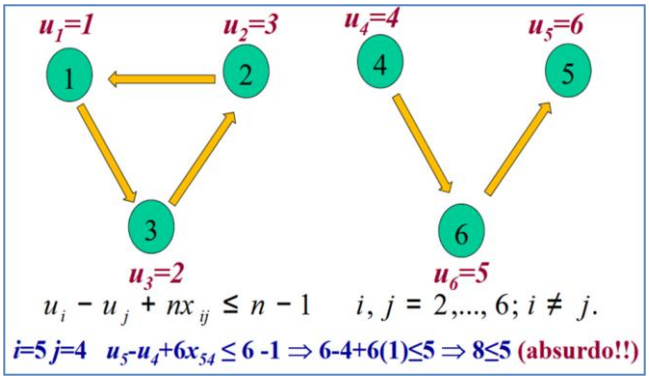
$$x_{i1} + x_{i2} + x_{i3} + x_{i4} + x_{i5} + x_{i6} = 1, \quad i = 1,\ldots,6$$

$$x_{1j} + x_{2j} + x_{3j} + x_{4j} + x_{5j} + x_{6j} = 1, \quad j = 1,\ldots,6$$

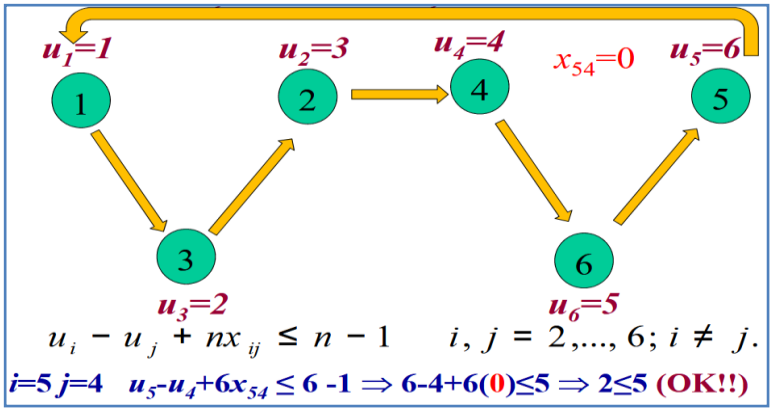
$$u_i - u_j + 6x_{ij} \le 5, \quad i \ne j; i = 2,\ldots,6; \quad j = 2,\ldots,6;$$

$$x_{ij} = 0/1, \forall i, j$$

$$1 \le u_i \le 6 \quad i = 2,\ldots,6$$



Na ilustração da direita, o nó origem já foi visitado ($x_{21} = 1$) e não podemos ter $x_{51} = 1$. O nó que o veículo visitará após o nó 5 deverá ter valor u_j maior que u_5. Não seria possível ir do nó 5 para o nó 4, pois esse nó já possui um valor menor do que u_5.



Mais detalhes interessantes sobre a formulação de MTZ em:

http://user.engineering.uiowa.edu/~dbricker/Stacks_pdf8/TSP_models.pdf

Considerações gerais sobre o TPS

- Para dimensões mais elevadas, a resolução do TSP por métodos de programação matemática é proibitiva em termos de tempos computacionais.

- TSP é da classe NP-hard: não existe algoritmo exato que o resolva em tempo polinomial. À medida que *n* cresce, o tempo cresce exponencialmente.

- TSP geralmente é resolvido por meio de heurísticas:
  - ✓ Procedimentos que seguem uma intuição para resolver o problema (forma humana de resolver o problema, fenômenos naturais, processos biológicos, etc.);
  - ✓ Não garantem a otimalidade da solução final;
  - ✓ Em geral, produzem soluções finais de boa qualidade rapidamente;
  - ✓ A heurística das economias de Clarke e Wright (CW), bastante conhecida e ainda muito utilizada como parte de outros procedimentos, foi originalmente desenvolvida para resolver o problema clássico de roteamento de veículos. Baseia-se na noção de economias/*savings*, que pode ser definido como o custo da combinação, ou união, de duas sub-rotas existentes. Trata-se de uma heurística iterativa de construção baseada numa função gulosa de inserção.