

Métodos de Apoio à Decisão

Pesquisa em árvore

João Pedro Pedroso

2024/2025

- *Aula passada:*
 - Programação inteira pura/mista
 - Relaxação linear
 - Aplicações:
 - seleção de projetos
 - problemas com custos fixos
 - restrições ou não exclusivo
 - restrições se ... então
- *Aula de hoje:*
 - Método da pesquisa em árvore (**branch and bound**)
 - Problemas de localização

Problema de programação inteira

$$\begin{aligned} \max z = & \quad c_1x_1 + c_2x_2 + \dots + c_nx_n \\ \text{sujeito a} & \quad a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \leq b_1 \\ & \quad a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n \leq b_2 \\ & \quad \dots \\ & \quad a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \leq b_m \\ & \quad x_1, x_2, \dots, x_n \in \mathbb{Z}^+ \end{aligned}$$

Recordemos:

- **Programação inteira pura** — todas as variáveis são inteiras.
- **Programação inteira mista** — apenas algumas das variáveis têm de ser inteiras.
- **Relaxação linear** — problema linear em que se omitem todas as restrições de integralidade das variáveis.

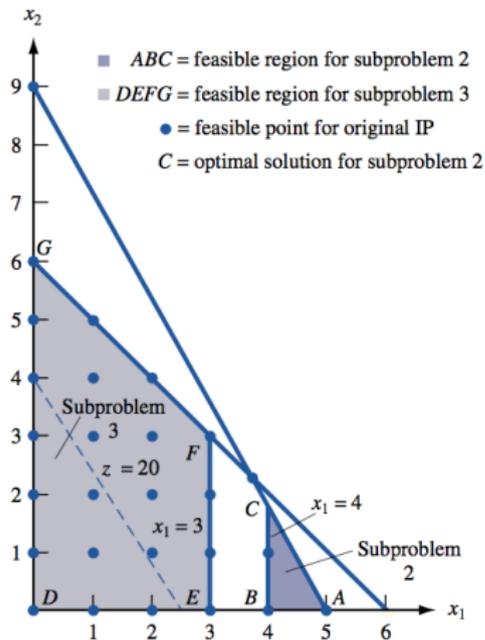
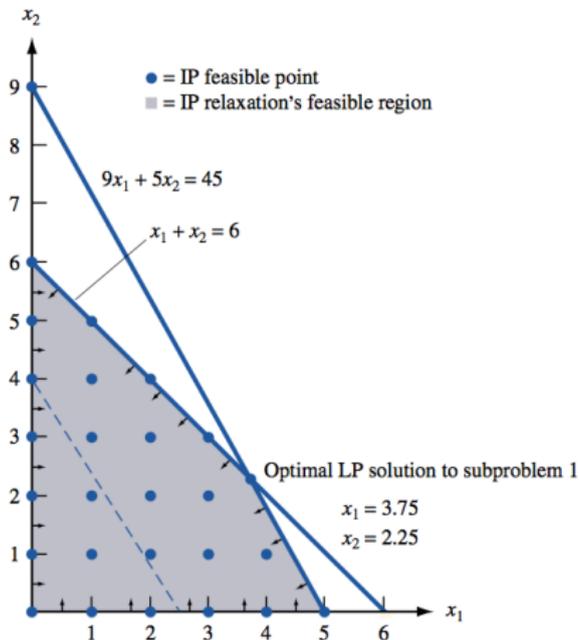
Em inglês: **Branch-and-bound**

- Resolver a relaxação linear (obtendo x^{LP}, z^{LP})
- Repartir a região admissível por forma a remover a solução anterior, e.g.:
 - escolher variável x_i^{LP} com valor fracionário
 - criar dois subproblemas:
 - SP com restrição $x_i \leq \lfloor x_i^{LP} \rfloor$
 - SP com restrição $x_i \geq \lceil x_i^{LP} \rceil$
- Repetir, enquanto houver variáveis fracionárias

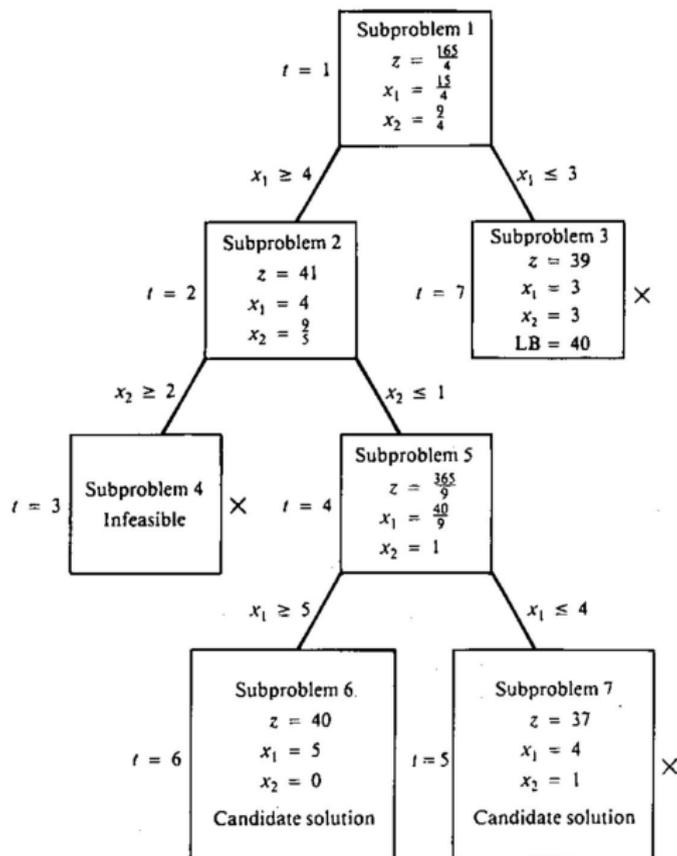
Pesquisa em árvore: exemplo

$$\begin{aligned} \text{maximizar } z &= && 8x_1 + 5x_2 \\ \text{sujeito a : } &&& x_1 + x_2 \leq 6 \\ &&& 9x_1 + 5x_2 \leq 45 \\ &&& x_1, x_2 \in \mathbb{Z}^+ \end{aligned}$$

Pesquisa em árvore: exemplo



Pesquisa em árvore: exemplo



- Pesquisa não-informada:
 - em profundidade (depth-first search, DFS → usada atrás)
 - em largura (breadth-first search, BFS)
- Pesquisa informada:
 - best-first search

- 1 Resolver a relaxação linear (obtendo x^{LP}, z^{LP})
 - 1 se na solução todas as variáveis forem inteiras: a solução é ótima para o problema original
 - 2 caso contrário: z^{LP} fornece um majorante para z^* (em maximização)
 - em minimização é um minorante
- 2 Repartir a região admissível, criando 2 subproblemas:
 - 1 escolher uma variável fracionária
 - 2 adicionar restrições que impeçam a variável de voltar a ter esse valor
- 3 Escolher um dos subproblemas
- 4 Repetir a partir de 2.
 - terminar quando não houver nenhum subproblema por explorar

- Árvore:
 - Nós → subproblemas → resolvidos com relaxação linear
 - Arestas → decisões → reduzir o domínio de variáveis
 - Folhas:
 - se admissíveis → **solução candidata**
 - caso contrário podem ser descartadas
- Melhor candidata encontrada até ao momento → **incumbent solution**
- **Bounding:**
 - descartar nós cujo *bound* (majorante/minorante) é inferior à *incumbent*
- Quando não há mais nós para expandir: → ***incumbent é ótima***

Ordem de visita é importante para tentar manter árvore pequena

Pontos em que são necessárias decisões:

- Escolha do nó da árvore de pesquisa a seleccionar:
 - pesquisa não-informada:
 - em profundidade (depth-first search, DFS)
 - em largura (breadth-first search, BFS)
 - pesquisa informada:
 - best-first search: o nó com melhor majorante/minorante
 - ...
- Escolha da variável para ramificar
 - Frequentemente: a mais fracionária, embora haja muitas heurísticas alternativas.

Problemas de localização

- Problema de otimização clássico para **determinar os locais** de fábricas e armazéns
 - escolher o melhor entre os sites potenciais
 - sujeito a restrições de *satisfação da procura* (encomendas) de clientes
 - objetivo: selecionar locais para instalações, de forma a **minimizar os custos totais**
- Estrutura de custos típica:
 - parte proporcional a **distâncias** dos pontos de procura às instalações de atendimento
 - parte relacionada à **abertura de instalações**
- As instalações podem ter capacidades limitadas
 - FLP capacitado
 - FLP não capacitado
- Veremos várias formulações e seu desempenho

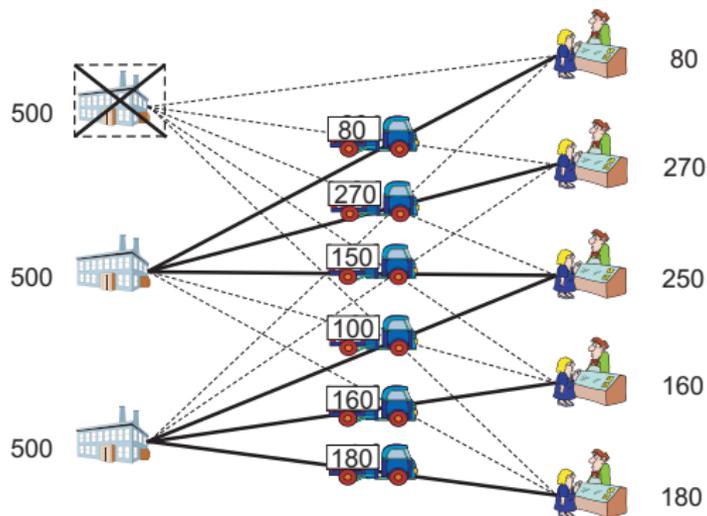
- a procura total que cada instalação pode satisfazer é limitada
- modelagem:
 - satisfação da procura
 - restrições de capacidade

Exemplo

- Uma empresa tem 3 locais onde pode instalar os seus armazéns e 5 pontos de procura
- Cada local j tem um **custo de ativação** anual f_j
 - e.g., custo de *leasing* anual se a empresa o usar
 - independente do volume que serve
- Volume limitado a um máximo anual M_j
- Custo de transporte c_{ij} por unidade servida do local j ao ponto de procura i

Cliente i	1	2	3	4	5		
Procura anual d_i	80	270	250	160	180		
Local j	custo c_{ij}					f_j	M_j
1	4	5	6	8	10	1000	500
2	6	4	3	5	8	1000	500
3	9	7	4	3	4	1000	500

Exemplo



Cliente i	1	2	3	4	5		
Procura anual d_i	80	270	250	160	180		
Local j	custo c_{ij}					f_j	M_j
1	4	5	6	8	10	1000	500
2	6	4	3	5	8	1000	500
3	9	7	4	3	4	1000	500

Formulação

- Clientes $i \in I = \{1, 2, \dots, n\}$
- Locais para instalações $j \in J = \{1, 2, \dots, m\}$
- Variáveis:
 - $x_{ij} \geq 0 \rightarrow$ quantidade enviada da instalação j ao cliente i
 - $y_j = 1$ se se abrir instalação no local j , $y_j = 0$ c.c.

$$\text{minimizar} \quad \sum_{j \in J} f_j y_j + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\text{sujeito a:} \quad \sum_{j \in J} x_{ij} = d_i \quad \forall i \in I$$

$$\sum_{i=1}^n x_{ij} \leq M_j y_j \quad \forall j \in J$$

$$x_{ij} \geq 0 \quad \forall i \in I, j \in J$$

$$y_j \in \{0, 1\} \quad \forall j \in J$$

- Objetivo: minimizar custos de ativação + custos de transporte
- Primeiras restrições: satisfação de procura de clientes
- Segundas restrições: quant. máxima servida por cada local
 - 0 se não for ativado
 - capacidade das instalações se forem ativadas

$$\text{minimizar} \quad \sum_{j \in J} f_j y_j + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\text{sujeito a:} \quad \sum_{j \in J} x_{ij} = d_i \quad \forall i \in I$$

$$\sum_{i=1}^n x_{ij} \leq M_j y_j \quad \forall j \in J$$

$$x_{ij} \geq 0 \quad \forall i \in I, j \in J$$

$$y_j \in \{0, 1\} \quad \forall j \in J$$

Modelo em AMPL

```
set I;
set J;
param f {J};
param c {I, J};
param d {I};
param M {J};

var x {I, J} >=0;
var y {J} binary;

subject to
Demand {i in I}: sum {j in J} x[i,j] = d[i];
Supply {j in J}: sum {i in I} x[i,j] <= M[j] * y[j];

minimize cost: sum {j in J} f[j] * y[j] +
               sum {i in I, j in J} c[i,j] * x[i,j];
```

Dados AMPL

```
data;
param: J: M    f := # defines set "J" and param "M" and "f"
      1 500 1000
      2 500 1000
      3 500 1000 ;
param: I: d := # defines set "I" and param "d"
      1 80
      2 270
      3 250
      4 160
      5 180;
param c (tr) : # (tr) --> transposed
      1 2 3 4 5 :=
1      4 5 6 8 10
2      6 4 3 5 8
3      9 7 4 3 4 ;
```

Formulação para o caso **capacitado**

- Clientes $i \in I = \{1, 2, \dots, n\}$
- Locais para instalações $j \in J = \{1, 2, \dots, m\}$
- Variáveis:
 - $x_{ij} \geq 0 \rightarrow$ quantidade enviada da instalação j ao cliente i
 - $y_j = 1$ se se abrir instalação no local j , $y_j = 0$ c.c.

$$\text{minimizar} \quad \sum_{j \in J} f_j y_j + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\text{sujeito a:} \quad \sum_{j \in J} x_{ij} = d_i \quad \forall i \in I$$

$$\sum_{i=1}^n x_{ij} \leq M_j y_j \quad \forall j \in J$$

$$x_{ij} \geq 0 \quad \forall i \in I, j \in J$$

$$y_j \in \{0, 1\} \quad \forall j \in J$$

Caso sem limite na capacidade

Uncapacitated facility location

Problema: restrição

$$\sum_{i=1}^n x_{ij} \leq My_j \quad \forall j \in J$$

- Como decidir valor de M ?

Caso sem limite na capacidade

Uncapacitated facility location

Problema: restrição

$$\sum_{i=1}^n x_{ij} \leq My_j \quad \forall j \in J$$

- Como decidir valor de M ?
- Formulação "correta": $M = \text{valor da procura total}$

Caso sem limite na capacidade

Uncapacitated facility location

Problema: restrição

$$\sum_{i=1}^n x_{ij} \leq My_j \quad \forall j \in J$$

- Como decidir valor de M ?
- Formulação "correta": $M = \text{valor da procura total}$
- Mas podemos melhorar a formulação adicionando

$$x_{ij} \leq d_i y_j \quad \forall i \in I, j \in J$$

- majorantes nos valores das variáveis (*variable upper bounds*)
- válida também no caso capacitado
 - redundantes, mas tornam a relaxação mais justa (*tight*)

Median problem

*Escolher um dado número de instalações de um conjunto de pontos possíveis num grafo, de tal forma que a **soma** das distâncias de cada cliente para a instalação mais próxima é minimizada*

- Normalmente, o número k de instalações a escolher é predeterminado
- Problema *k*-median:
 - variante do **problema de localização sem capacidades**
 - determina a localização de k instalações sem considerar custos fixos
 - cada ponto de procura (cliente) é servido por exatamente uma instalação
 - **objectivo**: servir todos os pontos de procura com o custo total mínimo

- Notação:

- distancia do cliente i à instalação $j \rightarrow c_{ij}$
- conjunto de clientes $\rightarrow \{1, \dots, n\}$
- conjunto de locais potenciais para instalações $\rightarrow \{1, \dots, m\}$
- (frequentemente, instalações e clientes são o mesmo conjunto de pontos; e.g., cidades num país)

- Variáveis:

$$x_{ij} = \begin{cases} 1 & \text{quando a procura do cliente } i \text{ é satisfeita pela instalação } j \\ 0 & \text{caso contrário} \end{cases}$$

$$y_j = \begin{cases} 1 & \text{se instalação } j \text{ for aberta} \\ 0 & \text{c.c.} \end{cases}$$

The k-Median problem

minimizar
$$\sum_{i=1}^n \sum_{j=1}^m c_{ij} x_{ij}$$

sujeito a:
$$\sum_{j=1}^m x_{ij} = 1 \text{ para } i = 1, \dots, n$$

$$\sum_{j=1}^m y_j = k$$

$$x_{ij} \leq y_j \text{ para } i = 1, \dots, n; j = 1, \dots, m$$

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

$$y_j \in \{0, 1\} \text{ para } j = 1, \dots, m$$

- cada cliente i é atribuído a exatamente uma instalação j

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

- são abertas exatamente k instalações

$$\sum_{j=1}^m y_j = k$$

- obrigar instalação j a abrir se servir algum ponto de procura i

$$x_{ij} \leq y_j \text{ for } i = 1, \dots, n; j = 1, \dots, m$$

- formulação mais fraca se substituirmos estas nm restrições pelas n seguintes

$$\sum_{i=1}^n x_{ij} \leq y_j, \text{ for } j = 1, \dots, m$$

→ piores valores para a relaxação linear

AMPL model

```
param n; # number of customers
param m; # number of facilities
param c {1..n, 1..m};
param k;

var x {1..n, 1..m} binary;
var y {1..m} binary;

minimize cost: sum {i in 1..n, j in 1..m} c[i,j] * x[i,j];

subject to
Serve {i in 1..n}: sum {j in 1..m} x[i,j] = 1;
Kfacil: sum {j in 1..m} y[j] = k;
Activate {i in 1..n, j in 1..m}: x[i,j] <= y[j];
```

```
data;  
param n := 5;  
param m := 3;  
param k := 2;  
param c (tr) : # (tr) --> transposed  
           1 2 3 4 5 :=  
1         4 5 6 8 10  
2         6 4 3 5 8  
3         9 7 4 3 4 ;
```

Center problem

*Escolher um dado número de instalações de conjunto de pontos possíveis num grafo, de tal forma que a **distância máxima** de um cliente para a instalação mais próxima é minimizada.*

- variante do problema *k-median*
- escolher subconjunto de vértices como locais para instalações
 - objetivo: deixar cada cliente “próximo” de uma instalação
- número de instalações *k* predeterminado

- Notação (como para *k-median*):
 - distancia do cliente i à instalação $j \rightarrow c_{ij}$
 - conjunto de clientes $\rightarrow \{1, \dots, n\}$
 - conjunto de locais potenciais para instalações $\rightarrow \{1, \dots, m\}$
- Variáveis:

$$x_{ij} = \begin{cases} 1 & \text{quando a procura do cliente } i \text{ é satisfeita pela instalação } j \\ 0 & \text{caso contrário} \end{cases}$$

$$y_j = \begin{cases} 1 & \text{se instalação } j \text{ for aberta} \\ 0 & \text{c.c.} \end{cases}$$

- Distância/custo para o cliente **mais distante** de uma instalação ativada
 - variável contínua adicional z

minimizar z

sujeito a: $\sum_{j=1}^m x_{ij} = 1$ para $i = 1, \dots, n$

$$\sum_{j=1}^m y_j = k$$

$$x_{ij} \leq y_j \quad \text{para } i = 1, \dots, n; j = 1, \dots, m$$

$$\sum_{j=1}^m c_{ij} x_{ij} \leq z \quad \text{para } i = 1, \dots, n$$

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

$$y_j \in \{0, 1\} \quad \text{para } j = 1, \dots, m$$

- Nova restrição:

$$\sum_{j=1}^m c_{ij}x_{ij} \leq z \quad \text{for } i = 1, \dots, n$$

- Determinar z de forma a ter **peelo menos** o valor c_{ij}
 - para todas as instalações j e todos os clientes i atribuídos a j
 - versão mais fraca (mas talvez mais natural):

$$c_{ij}x_{ij} \leq z, \quad \text{for } i = 1, \dots, n; j = 1, \dots, m$$

- intuição: na formulação forte estamos a adicionar mais termos ao lado esquerdo \rightarrow região admissível mais estreita
- Novo objetivo: z
 - minimizar o valor máximo \rightarrow objetivo **min-max**
 - tipo de problemas em que resolutores para otimização matemática tipicamente têm desempenho inferior
 - vem substituir o objetivo de *k-median*

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

```
param n; # number or customers
param m; # number or facilities
param c {1..n, 1..m};
param k;

var x {1..n, 1..m} binary;
var y {1..m} binary;
var z >= 0;

minimize maxcost: z;

subject to
Serve {i in 1..n}: sum {j in 1..m} x[i,j] = 1;
Kfacil: sum {j in 1..m} y[j] = k;
Activate {i in 1..n, j in 1..m}: x[i,j] <= y[j];
MinZ {i in 1..n}: sum {j in 1..m} c[i,j] * x[i,j] <= z;
```

Programação: semelhante ao k-median

- Observe diferenças no desempenho
- Observe diferenças na solução

- "Minimização do valor máximo" pode ser reduzido a otimização linear *standard*
 - criar nova variável
 - fazer essa variável pelo menos tão grande como cada um dos valores a minimizar
- Assumindo que queremos **minimizar o valor máximo de duas expressões lineares**
 - $3x_1 + 4x_2$
 - $2x_1 + 7x_2$.
 - minimizar nova variável z sujeita a:

$$3x_1 + 4x_2 \leq z$$

$$2x_1 + 7x_2 \leq z$$

Minimização do valor absoluto $|x|$ de uma variável real x :

- Expressão não linear
- Para a linearizar: adicionar variáveis não negativas y e z
 - $x = y - z \rightarrow$ valor de x em termos de y e z
 - $|x|$ pode ser escrito como $y + z$
- Assim:
 - ocorrências de x na formulação \rightarrow substituídas por $y - z$
 - $|x|$ na função objetivo \rightarrow substituído por $y + z$.
- Outra possibilidade:
 - adicionar variável z
 - impor $z \geq x$ e $z \geq -x$
 - z substitui $|x|$ na função objetivo

Se possível, deve-se evitar uma função objetivo que minimiza o máximo de valores.

- Otimização inteira \rightarrow resolução com *branch-and-bound*
- Se o objetivo minimizar o máximo valor de um conjunto de variáveis:
 - tendência a ter valores elevados para a diferença entre o minorante e o majorante do objetivo (**duality gap**)
 - tempo de resolução aumenta
 - se interrompermos *branch-and-bound*, solução *incumbent* costuma ser fraca

Formulações fortes e fracas

- Consideremos o caso **sem capacidades**
 - qualquer quantidade pode ser servida de qualquer instalação
 - **problema de localização sem capacidades**
- Uma forma de modelar: arbitrar **M muito elevado** em

$$\sum_{i \in I} x_{ij} \leq M_j y_j \quad \forall j \in J$$

- Se removermos as restrições redundantes $x_{ij} \leq d_i y_j, \quad \forall i \in I, j \in J$
 - problema fica difícil de resolver
 - em especial quando M aumenta
 - **big M pitfall**

$$\sum_{i \in I} x_{ij} \leq M_j y_j \quad \forall j \in J$$

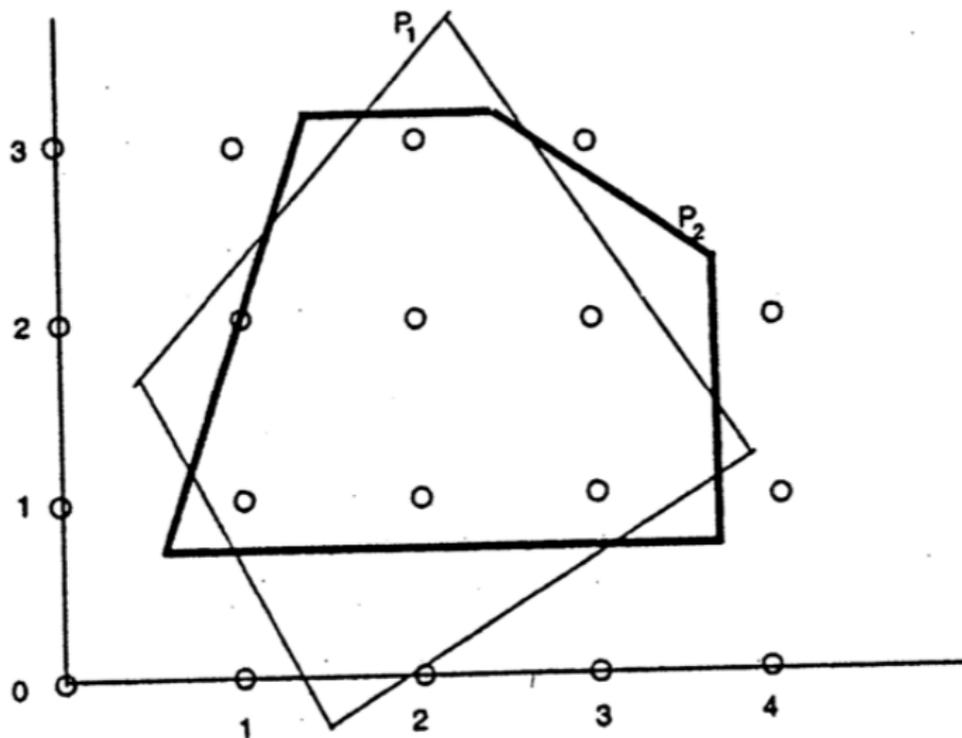
- Ideia: modelar "se não ativarmos instalação, não podemos transportar nada de lá"
- Parametro M representa um valor **suficientemente grande**
 - restrição deve ser limitante se $y_j = 0$
 - não deve ser limitante no caso contrário

Definição: formulações fortes e fracas

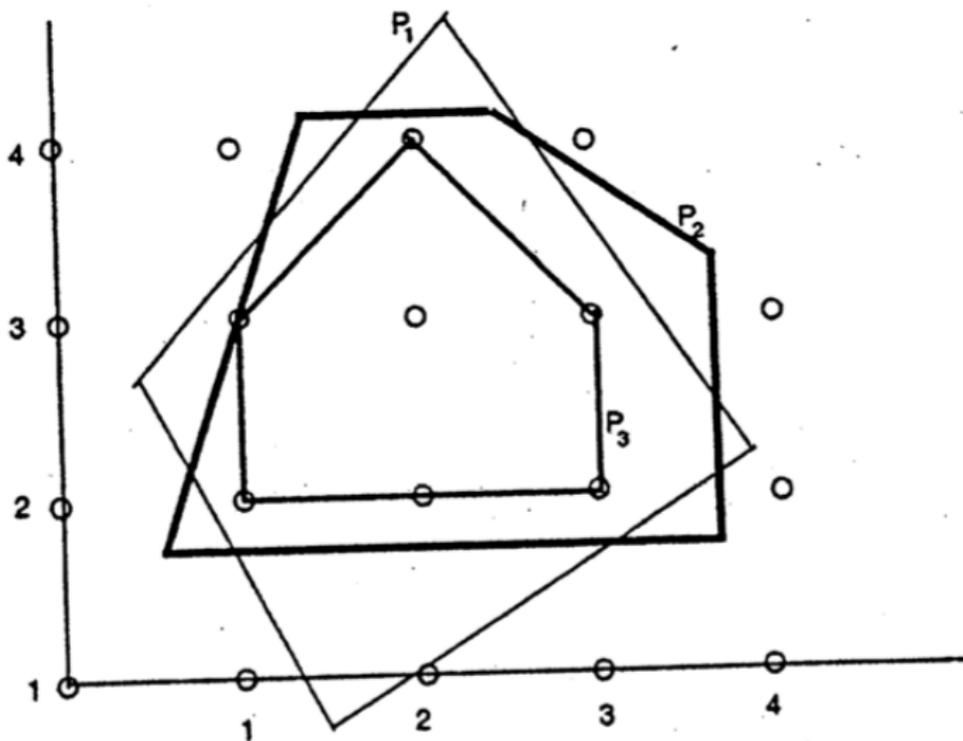
- Para um mesmo problema: duas formulações A e B
- Relaxação linear:
 - excluir restrições de integralidade
- Sejam as regiões admissíveis correspondentes P_A e P_B
- Se $P_A \subset P_B \rightarrow$ formulação A é mais forte do que B
 - B é mais fraca do que A

Intuitivamente, se P_A é mais justa do que P_B , o majorante/minorante dado pela relaxação linear fica mais próximo do ótimo do problema inteiro

Formulações diferentes



Formulação ideal



- Considere as formulações:
 - $A \rightarrow$ com restrições $x_{ij} \leq d_i y_j$
 - $B \rightarrow$ usando apenas $\sum_{i=1}^n x_{ij} \leq (\sum_{i=1}^n d_i) y_j$
- A é mais forte do que B
- Como verificamos:
 - $P_A, P_B \rightarrow$ regiões admissíveis de A e B
 - restrições de B obtêm-se adicionando as de A
 - isso implica que $P_A \subseteq P_B$

- Para mostrarmos que formulação é mais forte:
 - $P_A \subset P_B$, ou
 - verificar que a solução da relaxação linear de B não está incluída em P_A
- "É sempre preferível usar uma formulação mais forte?"
 - não há uma resposta teórica
 - parte da arte de modelação matemática

Percepção dos efeitos de uma formulação mais forte

- Geralmente, formulação mais forte têm **mais restrições e variáveis**
 - exemplo anterior:
 - formulação forte: nm restrições
 - fraca: apenas n restrições
- Tempo para resolver a relaxação linear:
 - depende do número de restrições e variáveis
 - provavelmente, mais longo para a formulação mais forte
- **Trade-off** entre
 - tempos mais curtos para solução das relaxações, em formulações fracas
 - maior número de nós na árvore de pesquisa (branch-and-bound)

Guideline: *como o tamanho da árvore de enumeração aumenta muito rapidamente quando a escala do problema aumenta, mesmo com mais variáveis e restrições as formulações **mais fortes** são geralmente preferíveis*

- Mais exemplos
 - otimização inteira
 - localização sem capacidades
 - qualidade das formulações