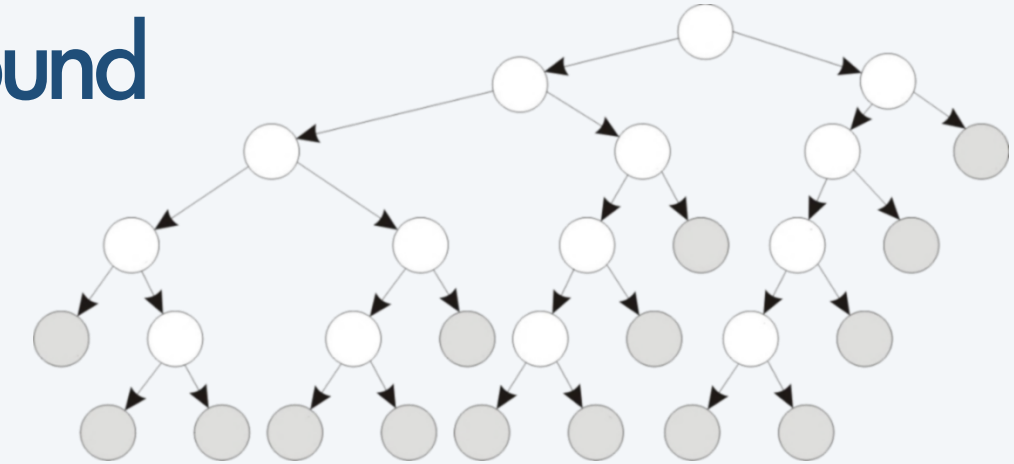
INSTITUTO DE INFORMÁTICA
UFRGS

# INF05010 - OTIMIZAÇÃO COMBINATÓRIA

## Branch and Bound



André Grahl Pereira

Busca Combinatória Heurística e Complexidade:
Problemas de Planejamento de Movimentos (Sokoban)

# Introdução

- **Enumeração Completa:**
  - Através da enumeração exaustiva de todas as soluções admissíveis é possível obter a solução ótima.
  - Proibitivo mesmo para problemas pequenos.

- **Enumeração Implícita:**
  - Usa um método de busca inteligente que cobre todas as possíveis soluções.
  - Avaliando explicitamente apenas um pequeno conjunto de soluções e ignorando (possivelmente) um grande número de soluções de menor qualidade.

# Introdução: Divisão e Conquista

- **Ideia:**
  - **Divisão:**
    - Dividir o problema uma série de problemas mais fáceis.
  - **Conquista:**
    - Resolver os problemas mais fáceis e combina-los para resolver o problema original.

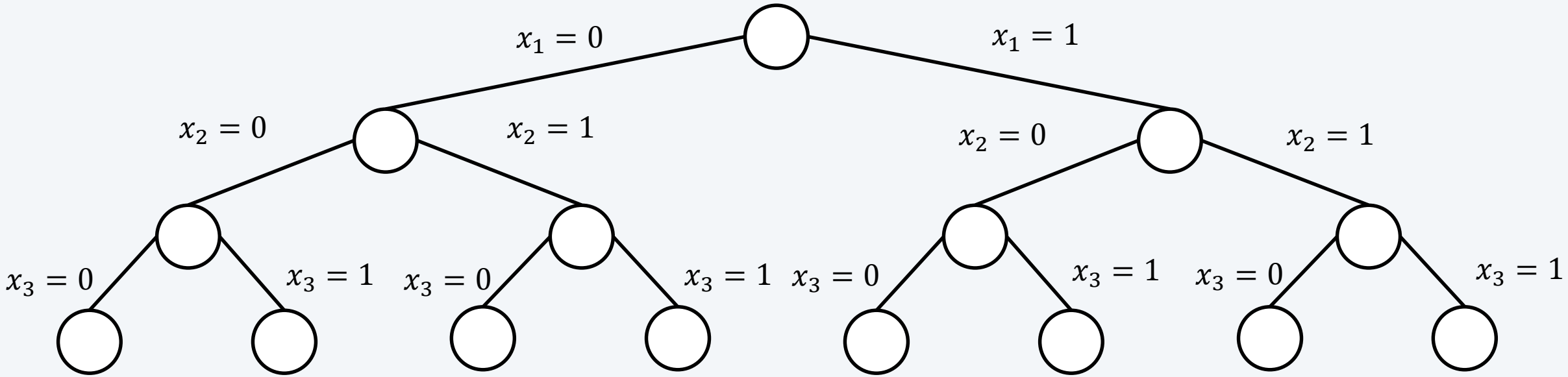
- **Formalmente:**

Seja $\{S_1, S_2, \dots, S_k\}$ uma decomposição de $S$, i.e. $S = S_1 \cup S_2 \cup \dots \cup S_k$.

Se $z^k = \min\{cx : x \in S_k\}, k = 1, \dots, K$, então $z = \min_k z^k$.

# Introdução: Divisão e Conquista

Exemplo: $S = \{0,1\} = \mathbb{B}^3$



# Introdução: Branch and Bound (BnB)

- Técnica geral para problemas de otimização combinatória.

*“Branch and Bound is by far the most widely used tool for solving large scale NP-Hard combinatorial optimization problems.” Clausen, 1999.*

- **Branching:**
  - Divisão do problema.

- **Bounding:**
  - Processo de conquista.

# Podas

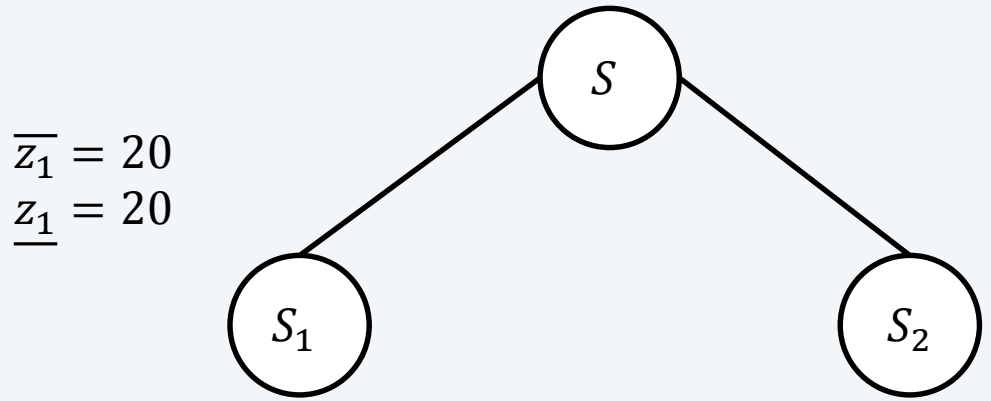
- A eficiência do método vêm da capacidade de podar soluções parciais.
- Para rejeitar uma solução parcial, é preciso ter certeza que o custo da solução parcial excede o custo de uma solução anteriormente encontrada.
- Assim, limites são calculados para cada solução parcial.

- **Tipo de Podas:**
  - Caso 1: poda por otimalidade.
  - Caso 2: poda por limitante.
  - Case 3: poda por inviabilidade.

(Exemplos supondo o calculo de um limite inferior para um problema de minimização)

# Poda por Otimalidade

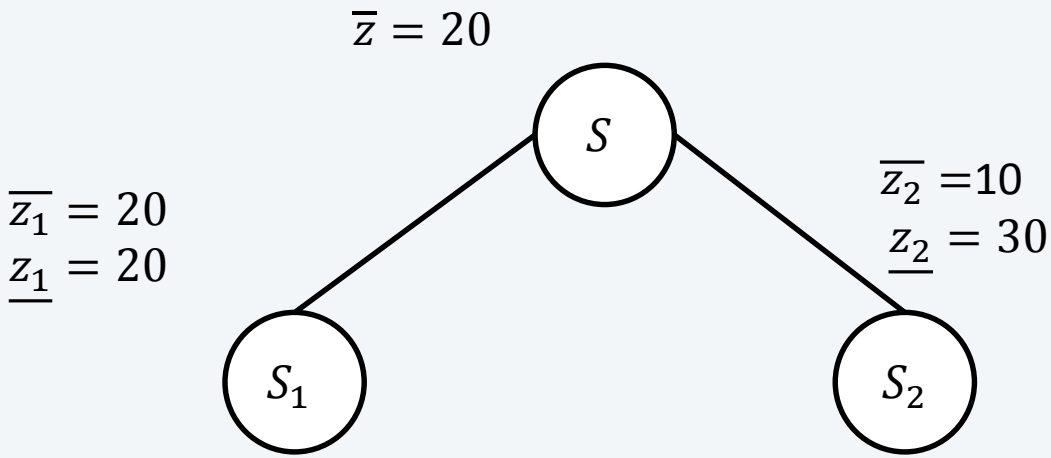
- Solução conhecida é a melhor possível para a subárvore, $\overline{z_i} = \underline{z_i}$.




Em $S_1$ é conhecida uma solução de custo 20 e ela é ótima, pois custo de qualquer solução de $S_1$ é $\geq 20$.

# Poda por Limitante

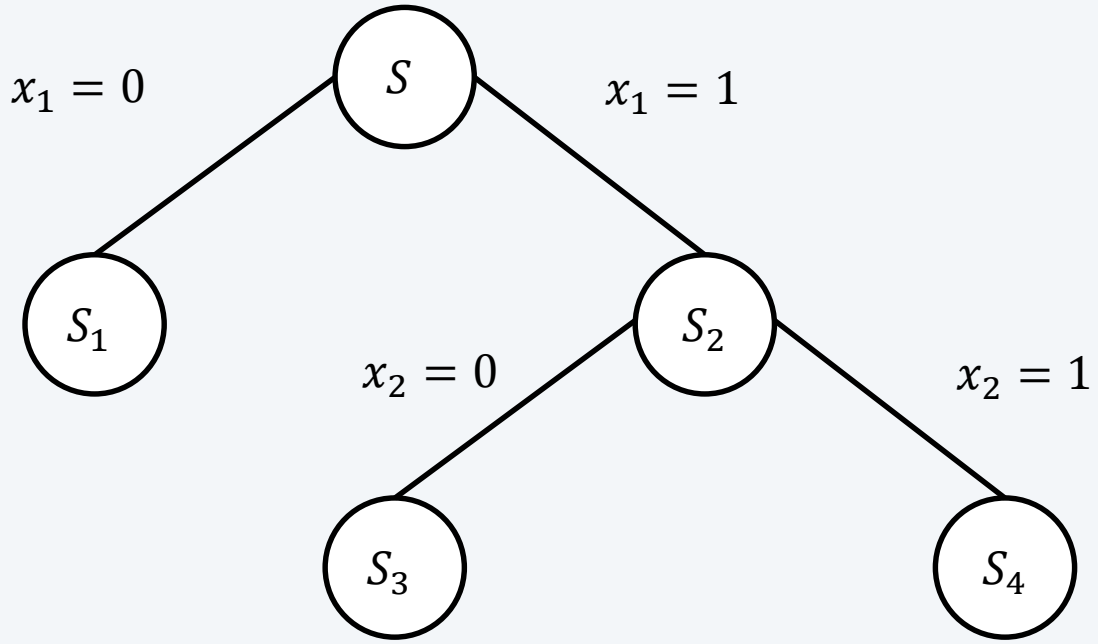
- Limite inferior da subárvore $\underline{z_i}$ é maior que o limite superior $\overline{z}$.



# Poda por Inviabilidade

- Subárvore $\underline{z_i}$ é inviável.

**Mochila 0-1:** $8x_1 + 5x_2 + 2x_3 \leq 10$



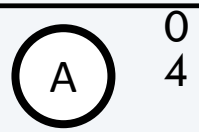

A soma dos pesos em $S_4$ ultrapassa a capacidade da mochila.

# Algoritmo DFBnB

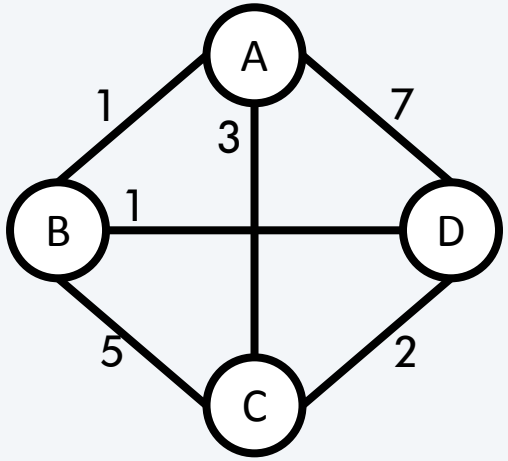
Intância: $\mathrm{P} = \min\{\mathrm{c}^{\mathrm{t}}\mathrm{x} | \mathrm{Ax} \leq \mathrm{b}, \mathrm{x} \in \mathrm{Z}_{+}^{\mathrm{n}}\}$.
Saida: Solução inteira ótima.

1. $\overline{z} := \infty$
2. $S := \{P\}$
3. **while** $S \neq \emptyset$ **do**
4. $N := S.pop()$
5. **for** *each child* $N_i$ *of* $N$ **do**
6. **if** $\underline{z}(N_i) < \overline{z}$ **then**
7. $S := S \cup \{N_i\}$
8. **end if**
9. **if** isFeasible$(N_i)$ **then**
10. $\overline{z} := \min(\overline{z}, \underline{z}(N_i))$
11. **end if**
12. **end for**
13. **end while**

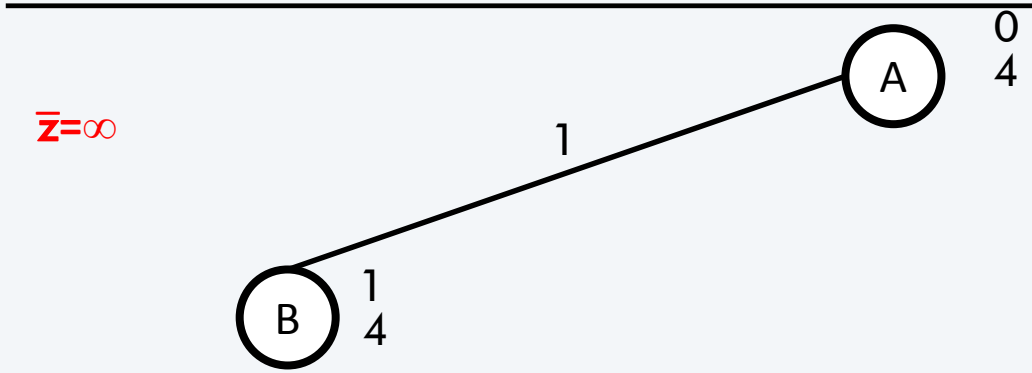
# Exemplo: DFBnB TSP

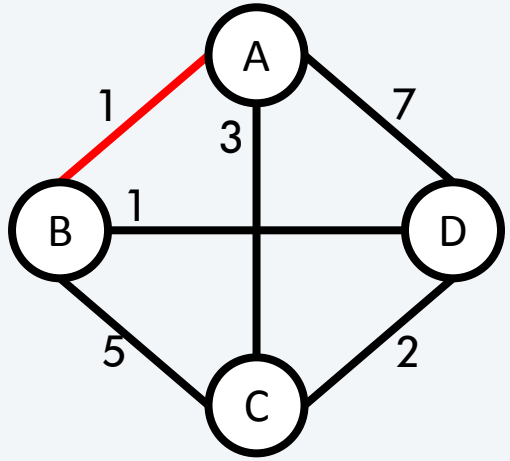



$\bar{z}=\infty$

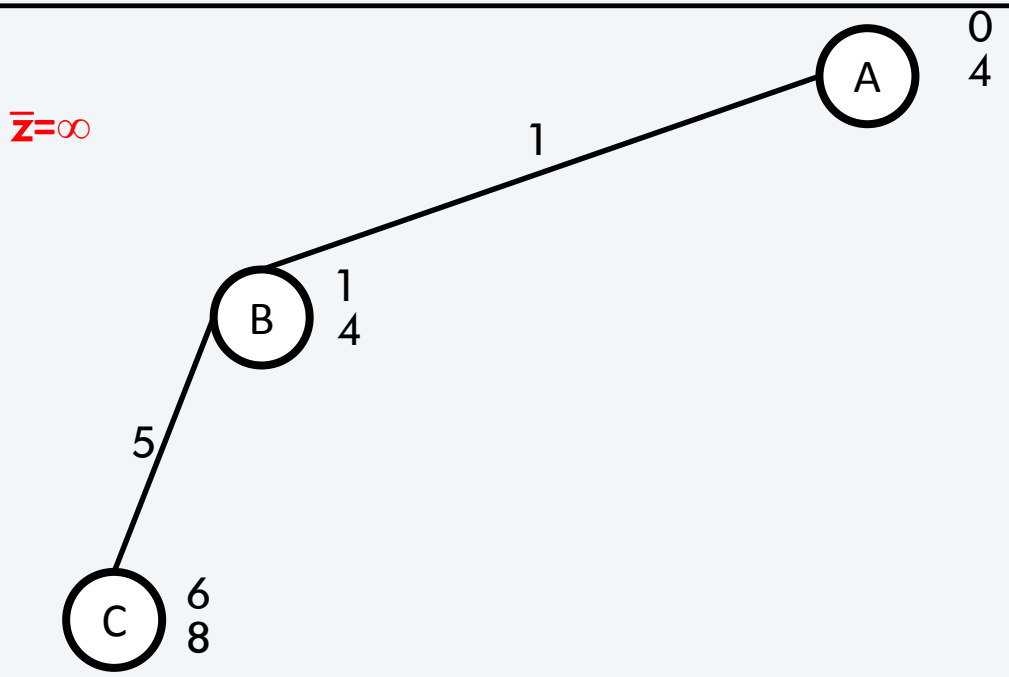


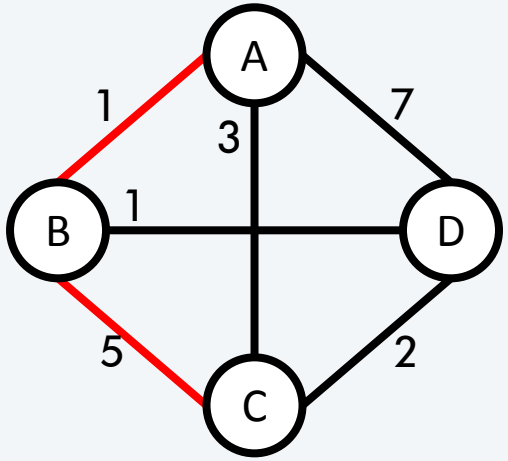
# Exemplo: DFBnB TSP



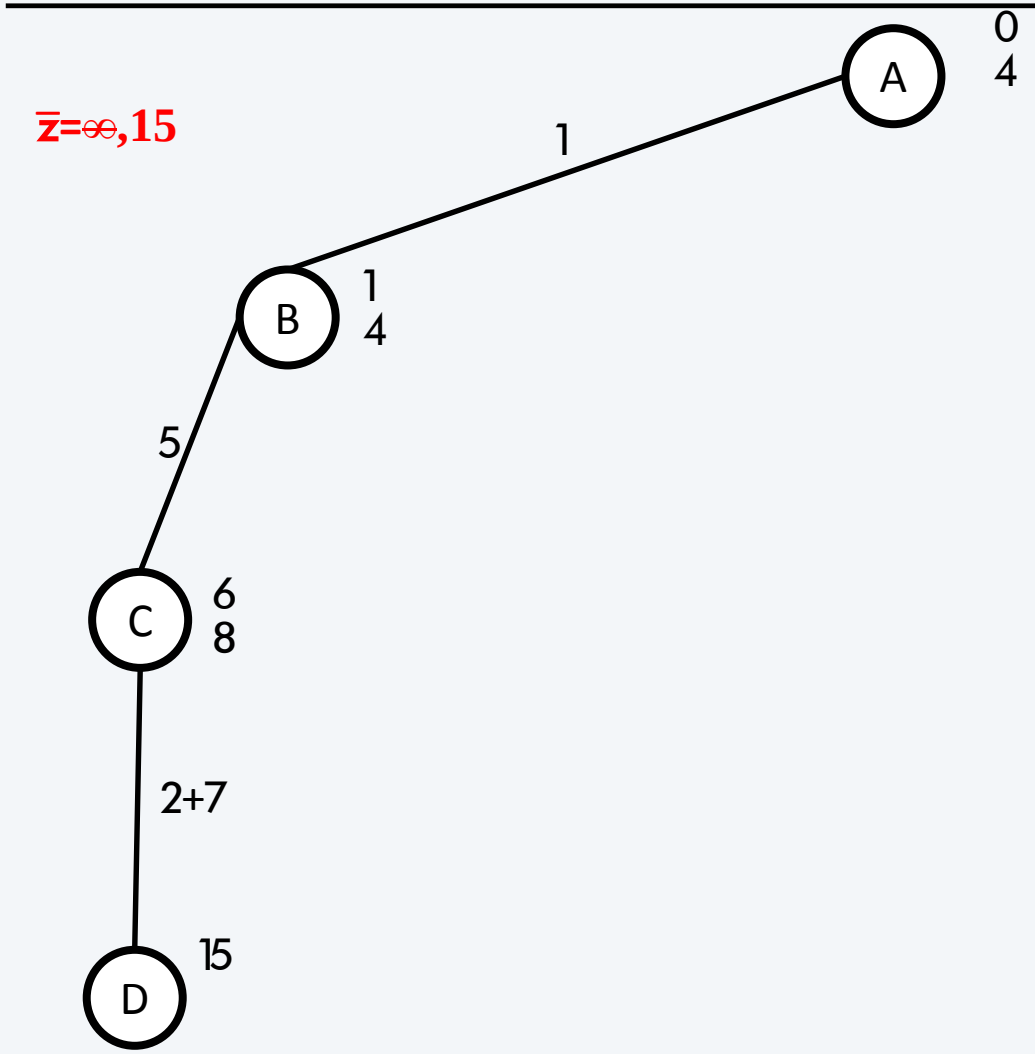



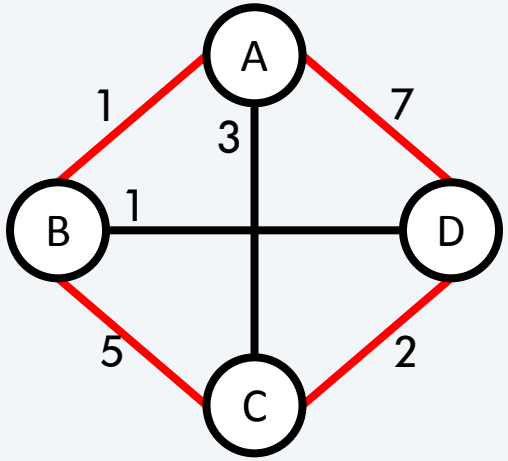
# Exemplo: DFBnB TSP



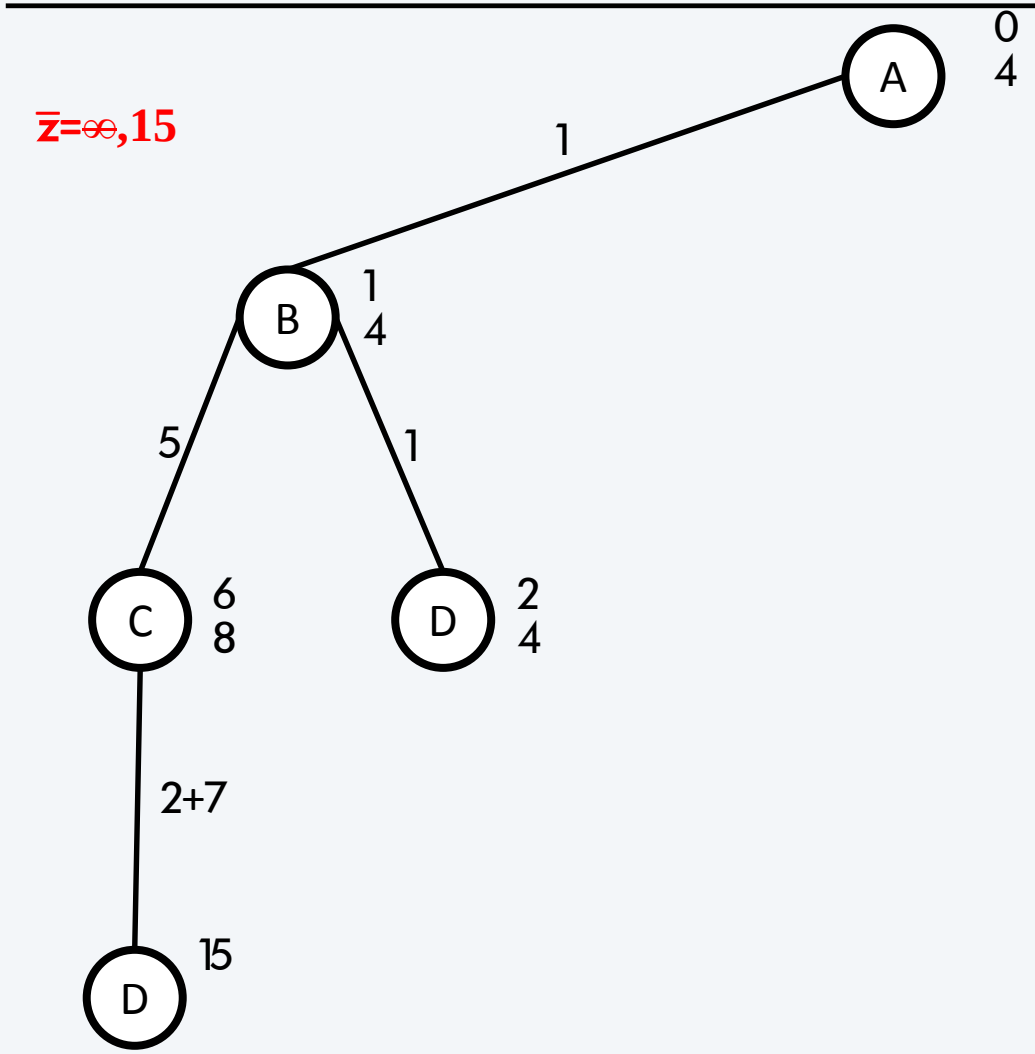



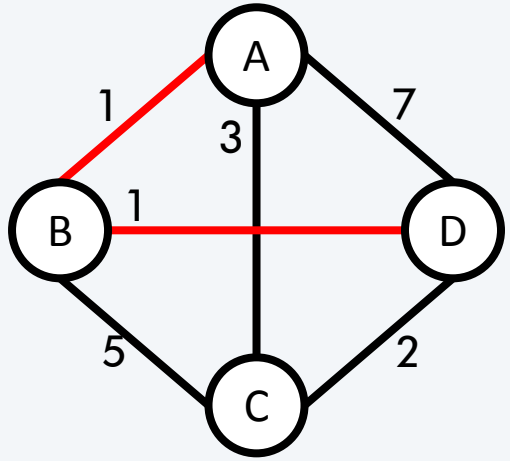
# Exemplo: DFBnB TSP



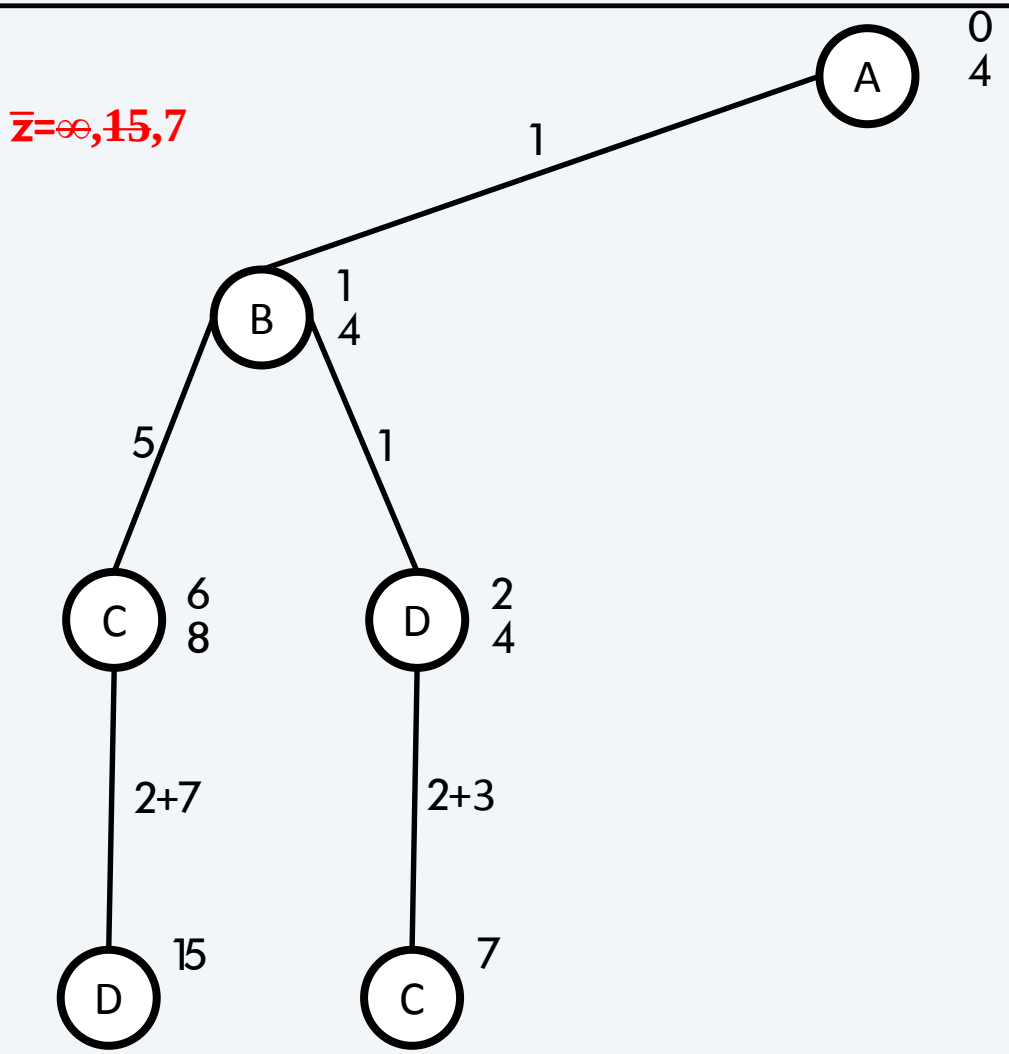



# Exemplo: DFBnB TSP

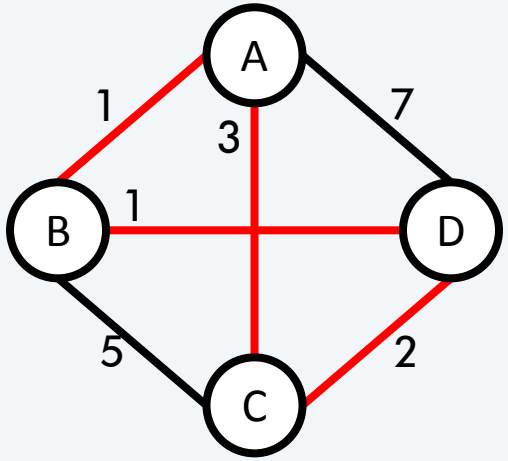


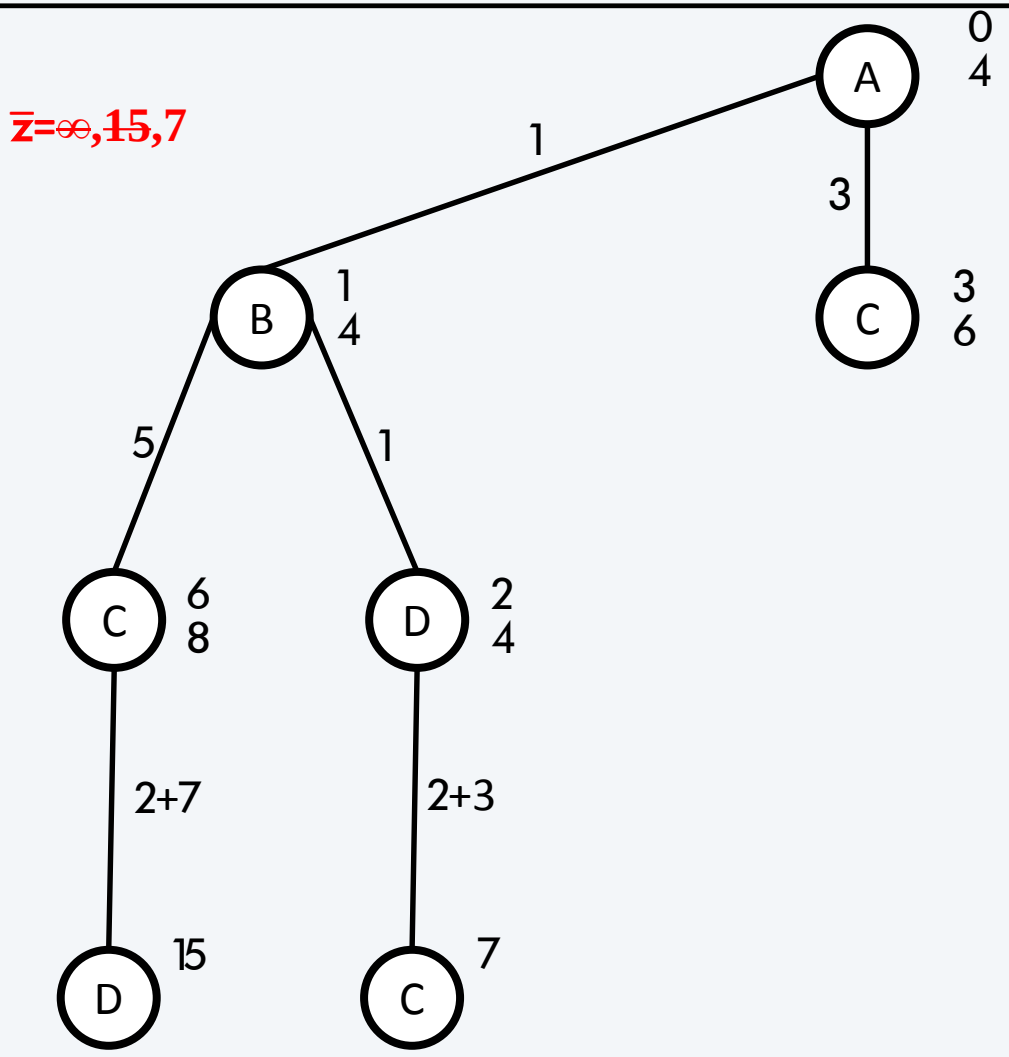
# Exemplo: DFBnB TSP

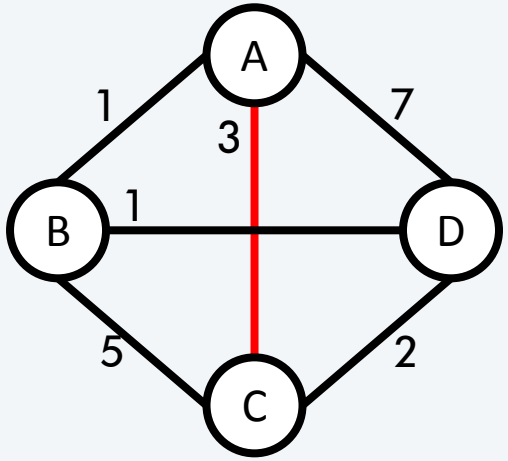



$\bar{z}=\infty,\cancel{15},7$

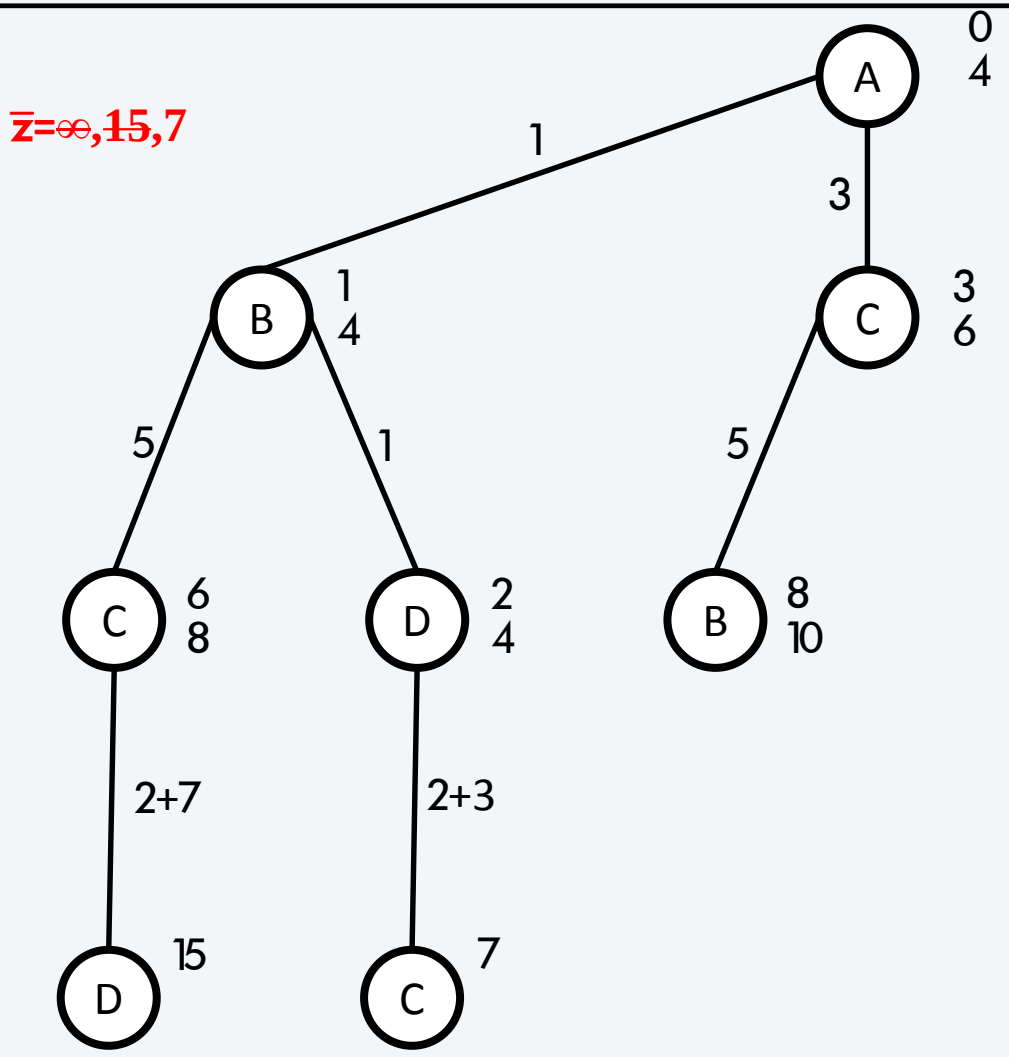


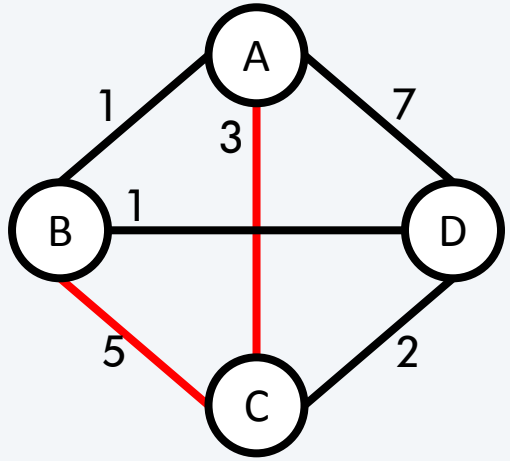
# Exemplo: DFBnB TSP



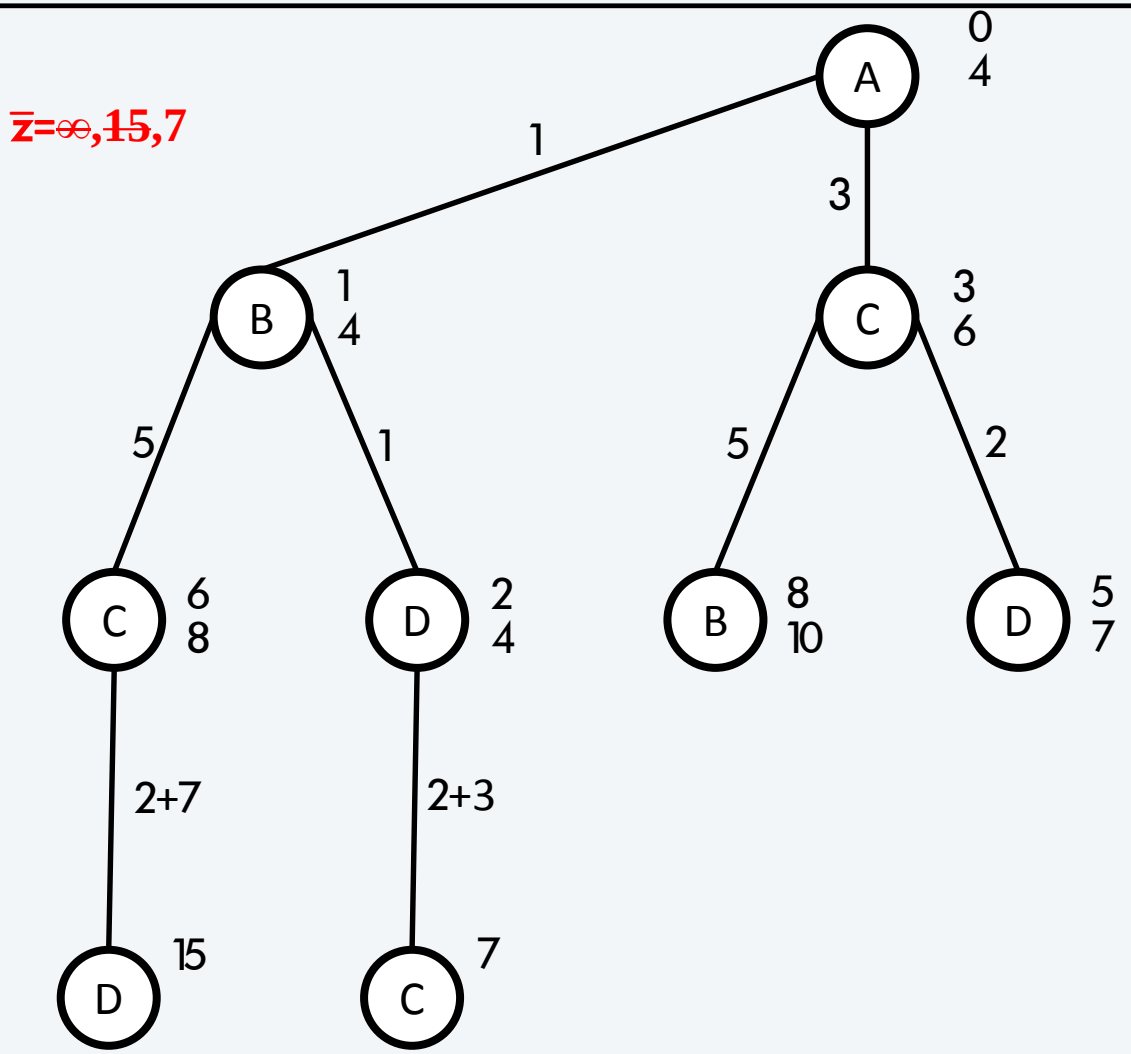



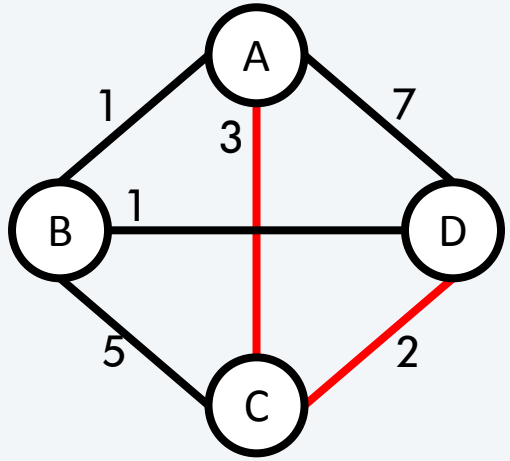
# Exemplo: DFBnB TSP



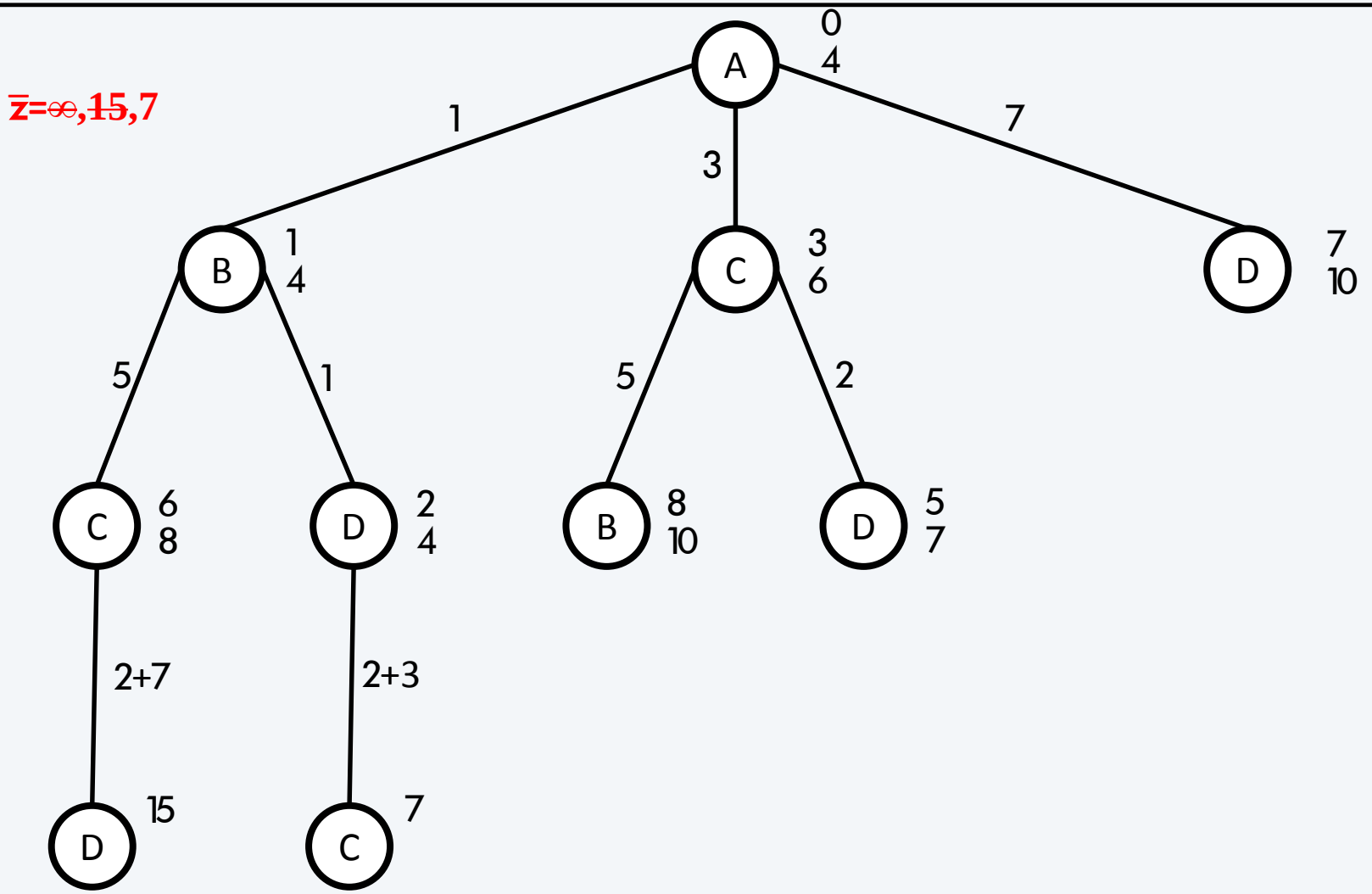



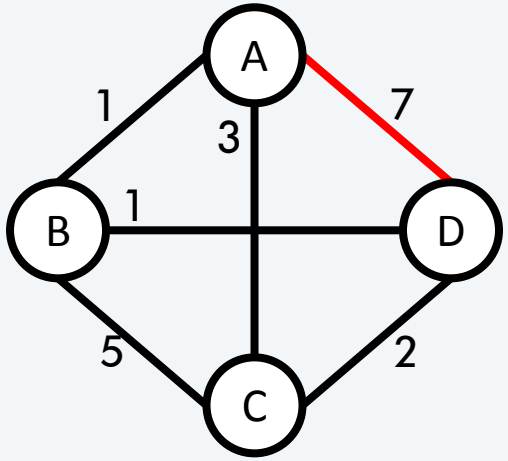
# Exemplo: DFBnB TSP



# Exemplo: DFBnB TSP



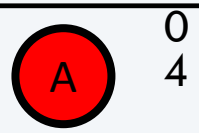



# Algoritmo BFBnB

Intância: $P = \min\{c^t x | Ax \leq b, x \in Z_+^n\}$.
Saida: Solução inteira ótima.

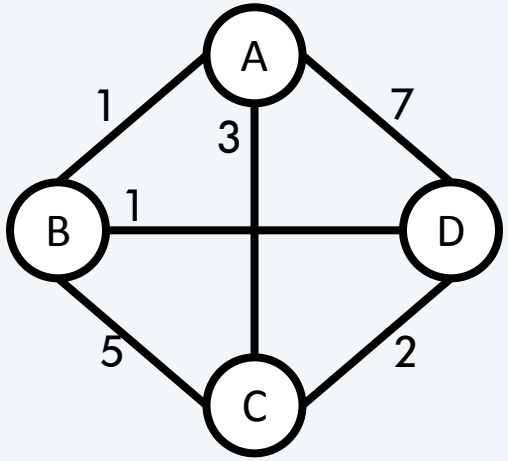
1. $\overline{z} := \infty$
2. $Q := \{P\}$
3. **while** $Q \neq \emptyset$ **do**
4. $N := Q.pop()$
5. **if** isFeasible$(N)$ **then**
6. **return** $N$
7. **end if**
8. **for** *each child* $N_i$ *of* $N$ **do**
9. **if** $\underline{z}(N_i) < \overline{z}$ **then**
10. $Q := Q \cup \{N_i\}$
11. **end if**
12. **if** isFeasible$(N_i)$ **then**
13. $\overline{z} := \min(\overline{z}, \underline{z}(N_i))$
14. **end if**
15. **end for**
16. **end while**

# Exemplo: BFBnB TSP



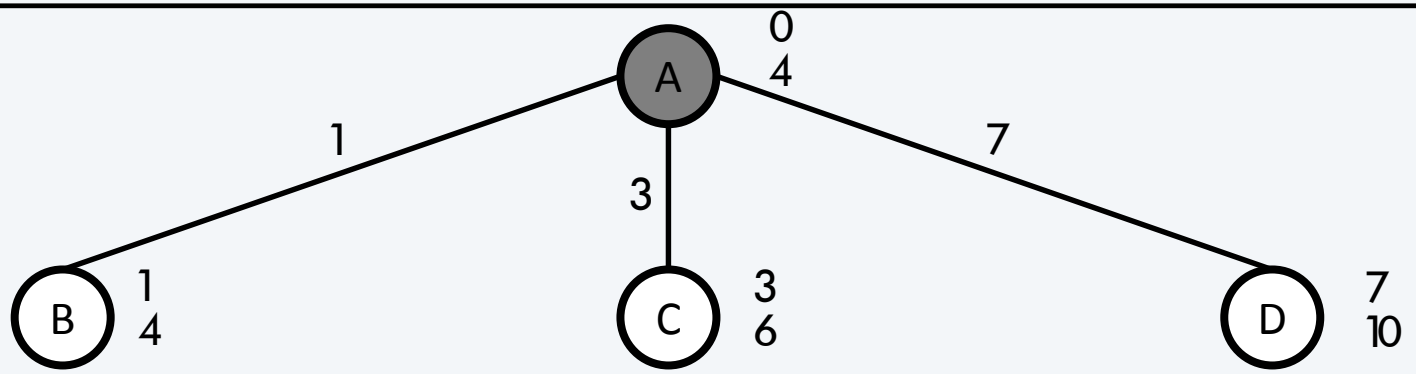

0
4

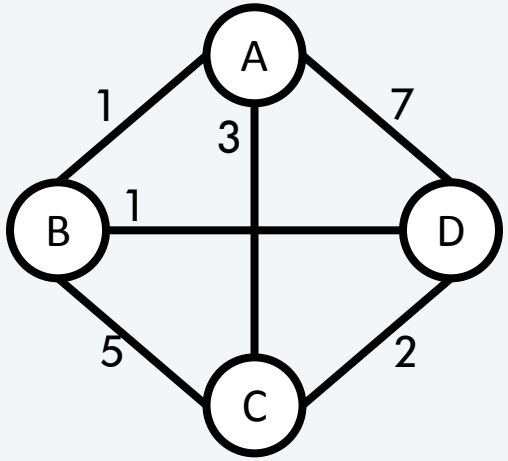
$\bar{z}=\infty$



# Exemplo: BFBnB TSP

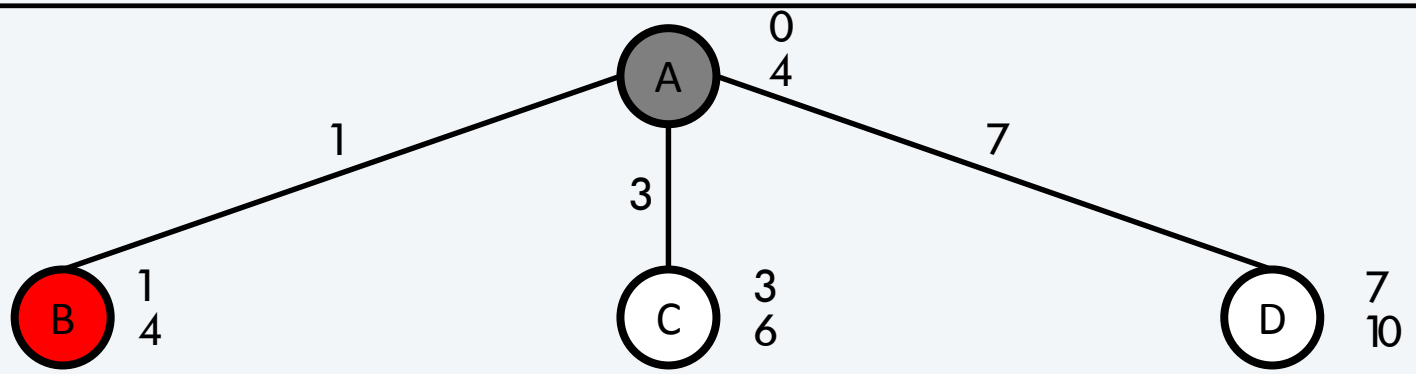
$\bar{z}=\infty$

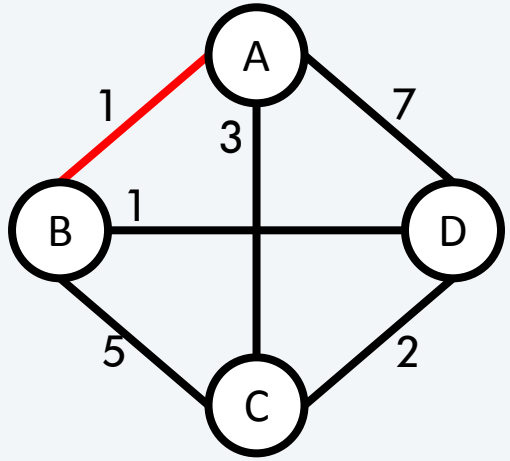


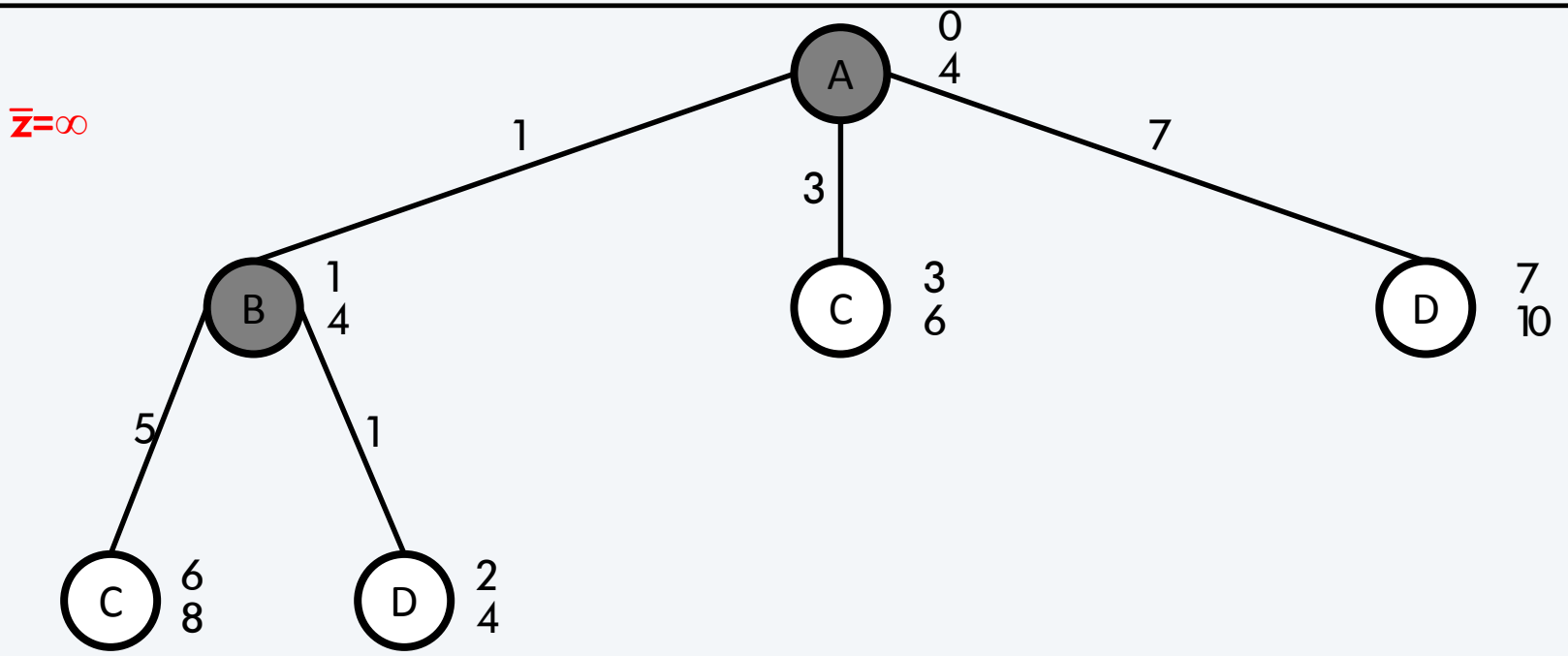
# Exemplo: BFBnB TSP

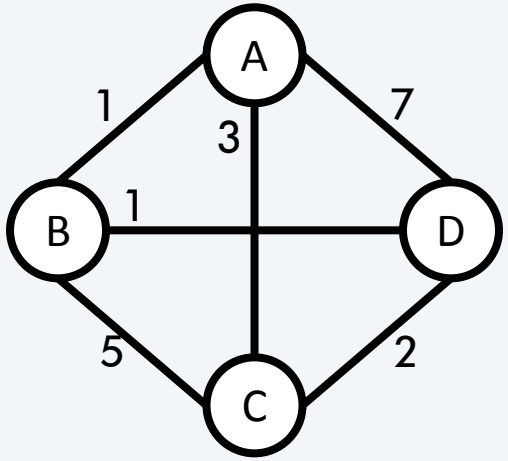
$\bar{z}=\infty$



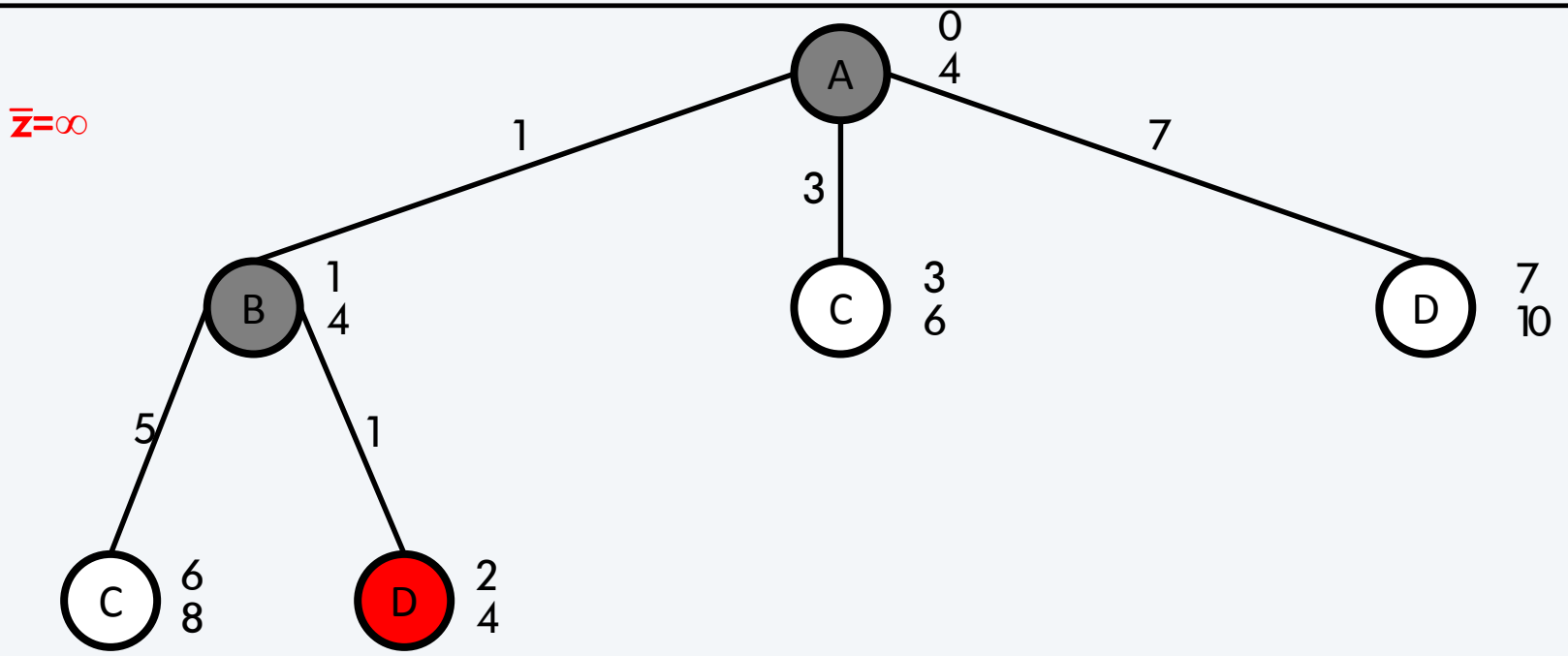



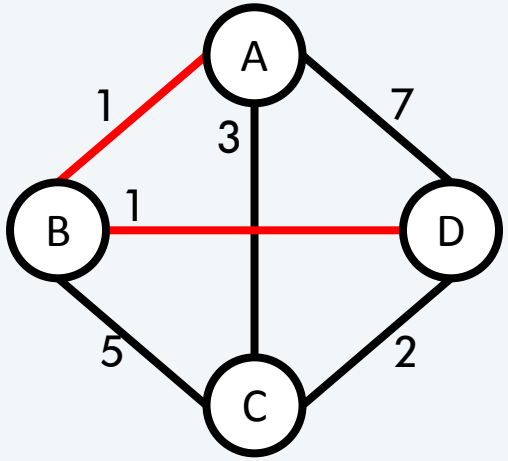
# Exemplo: BFBnB TSP



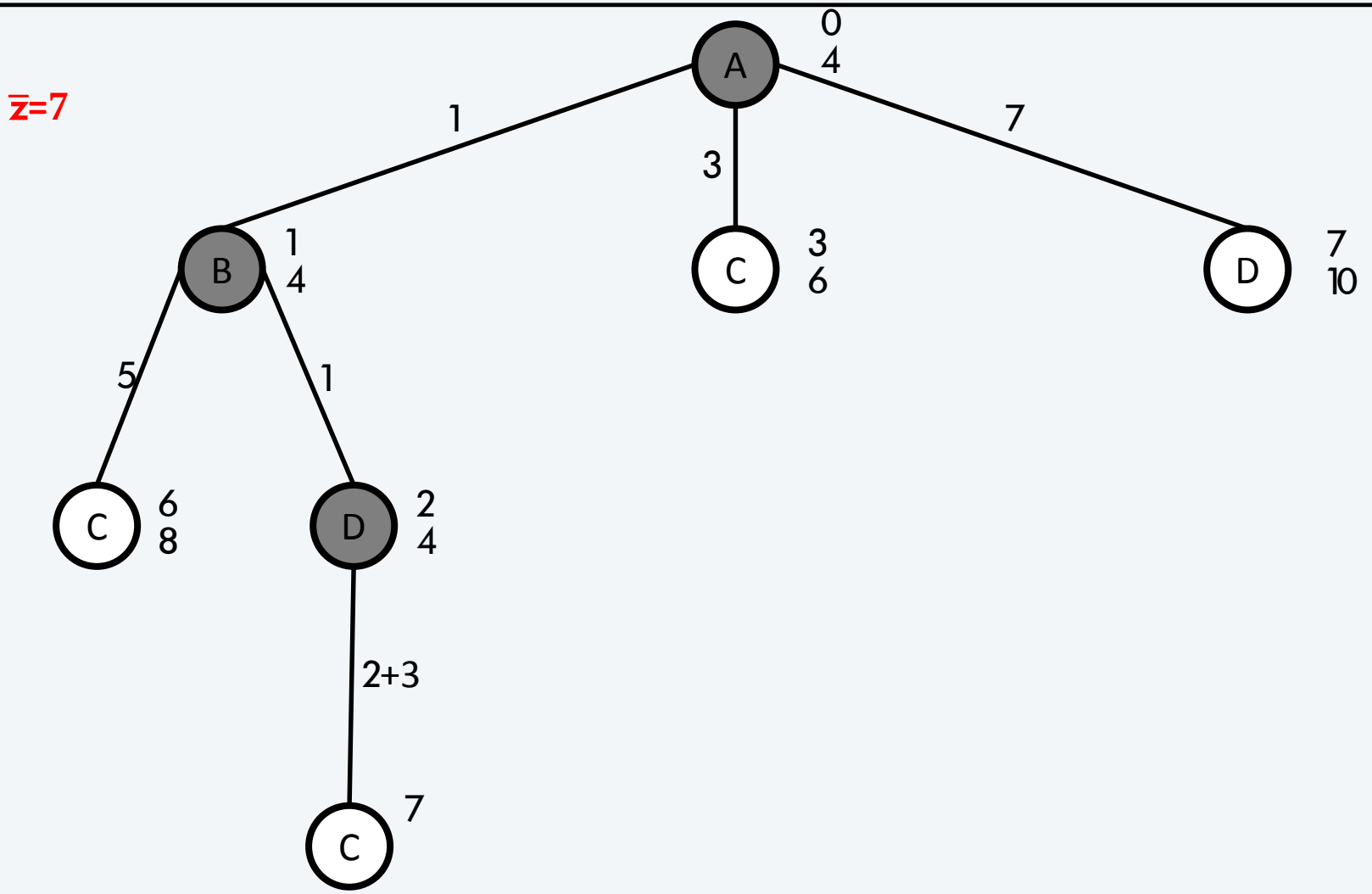



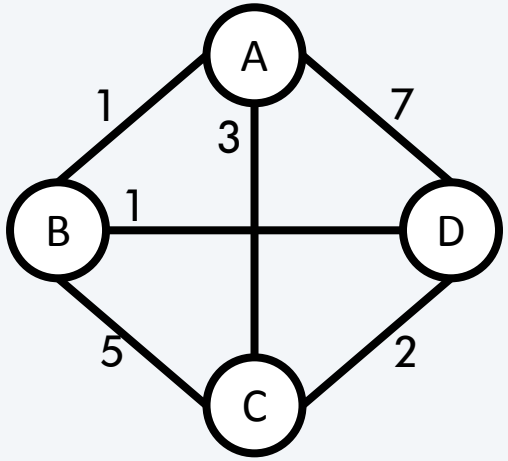
# Exemplo: BFBnB TSP



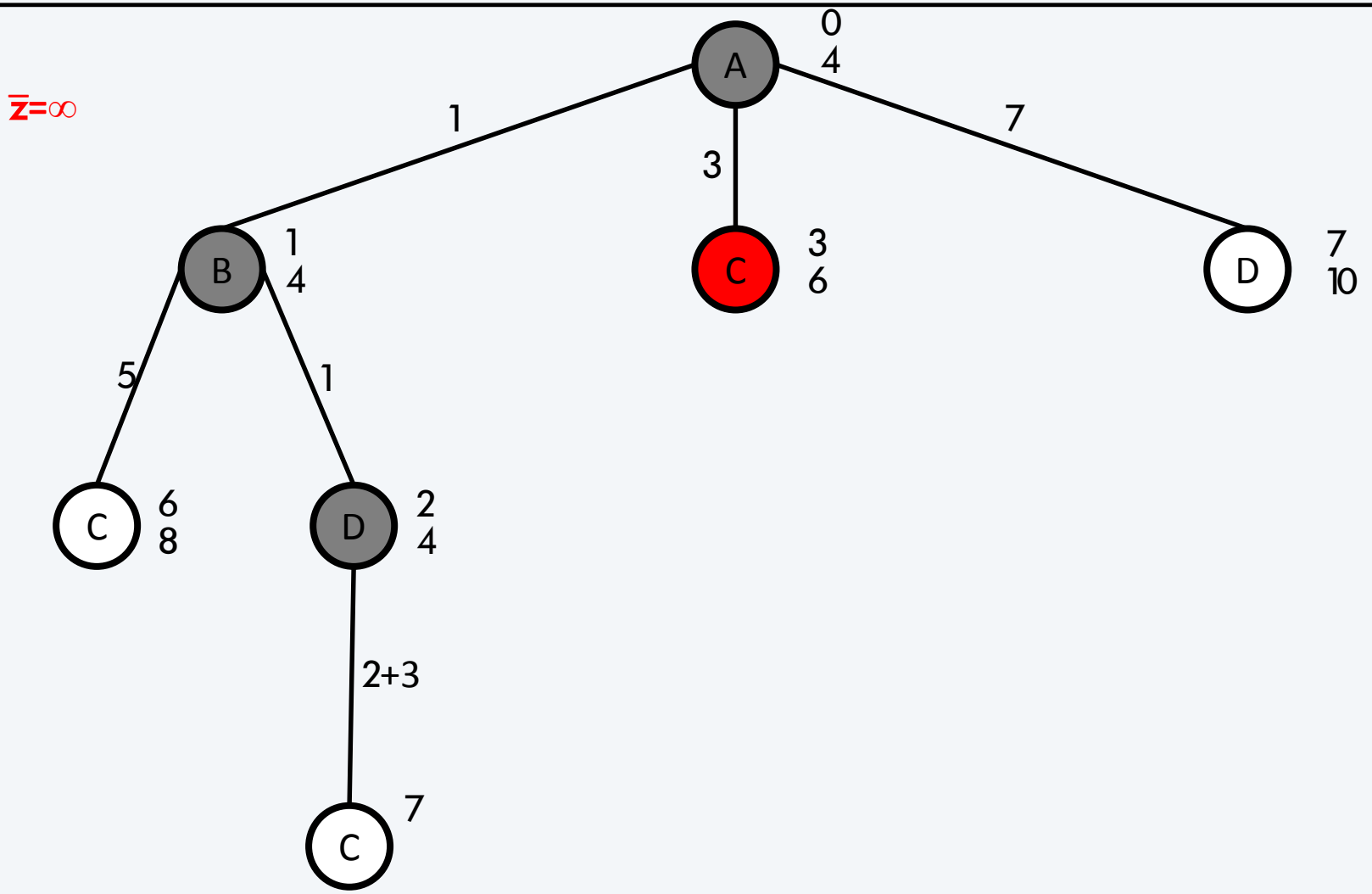



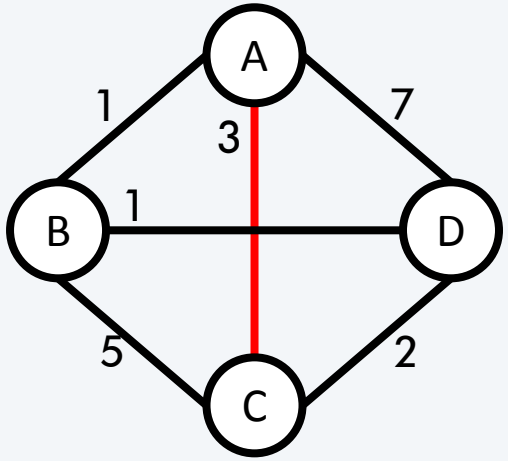
# Exemplo: BFBnB TSP



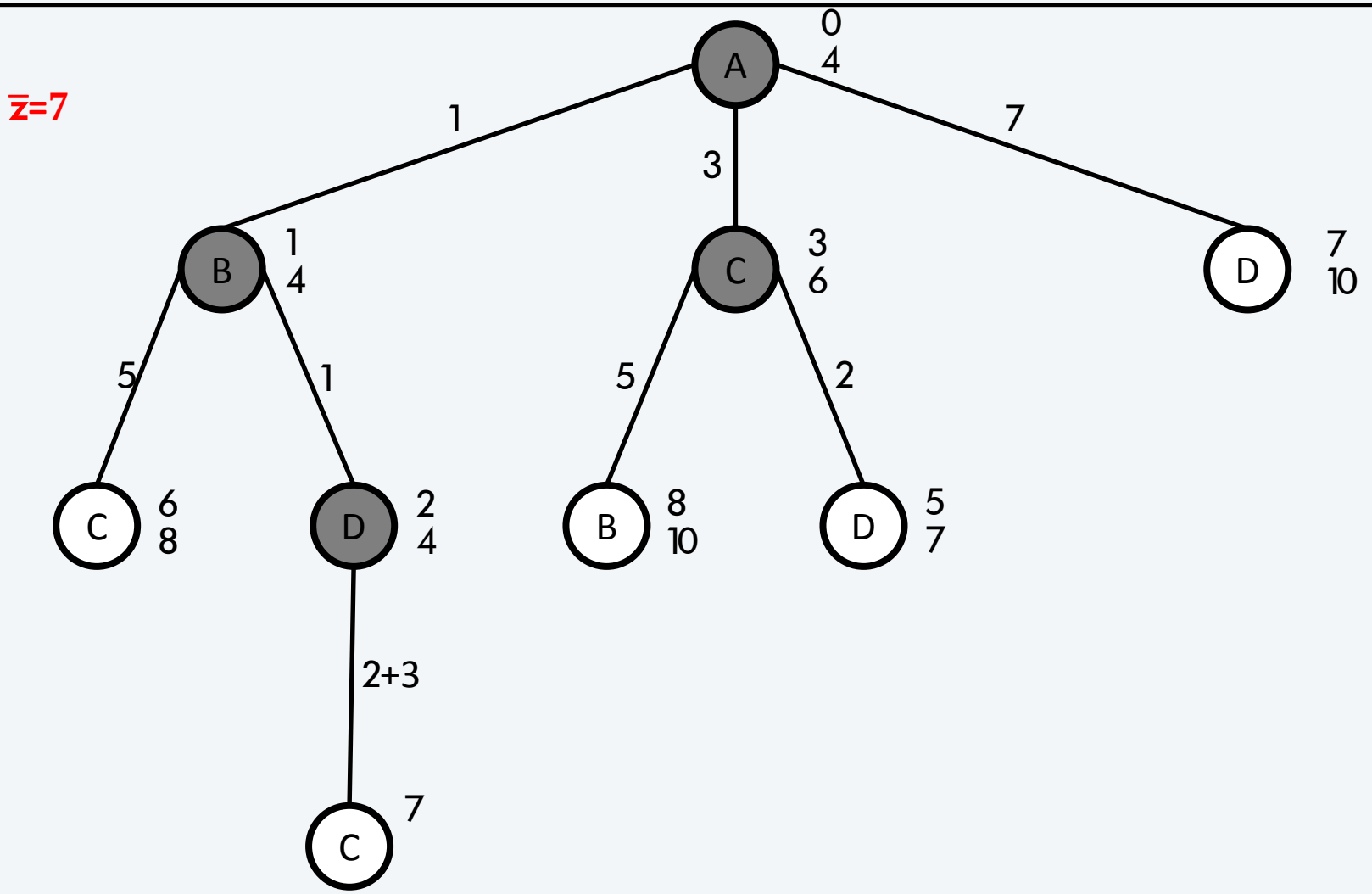



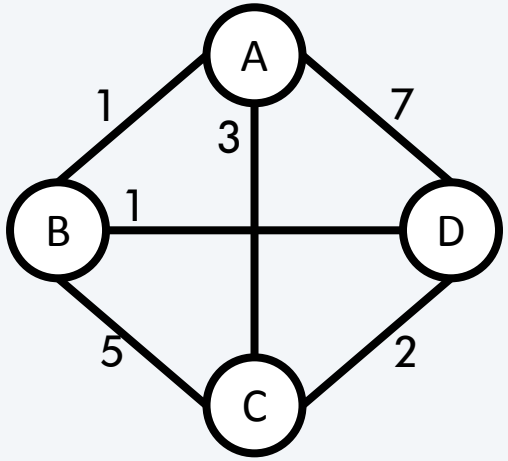
# Exemplo: BFBnB TSP



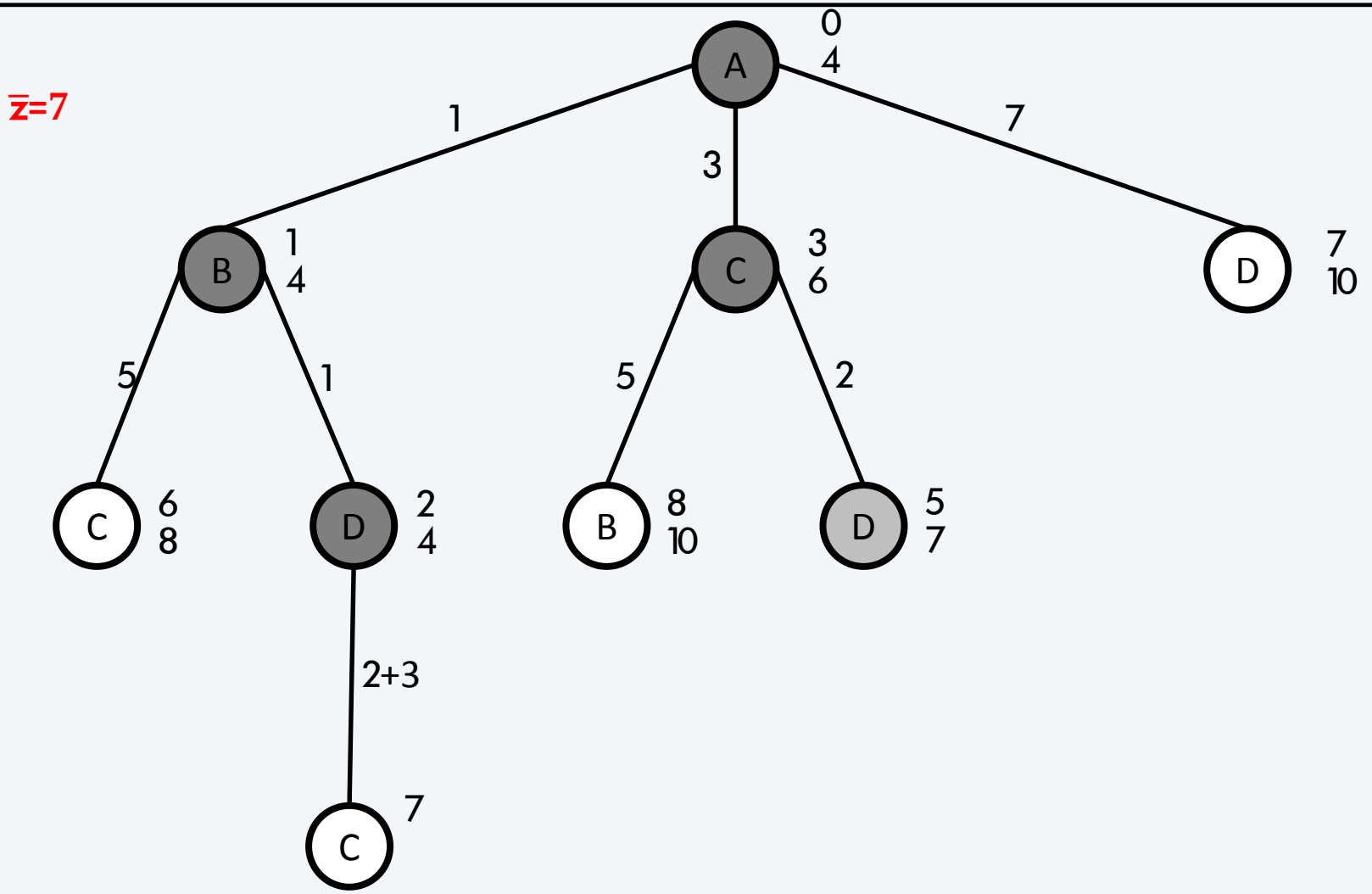



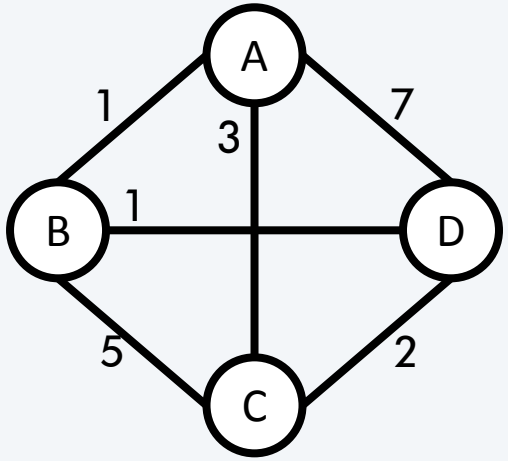
# Exemplo: BFBnB TSP



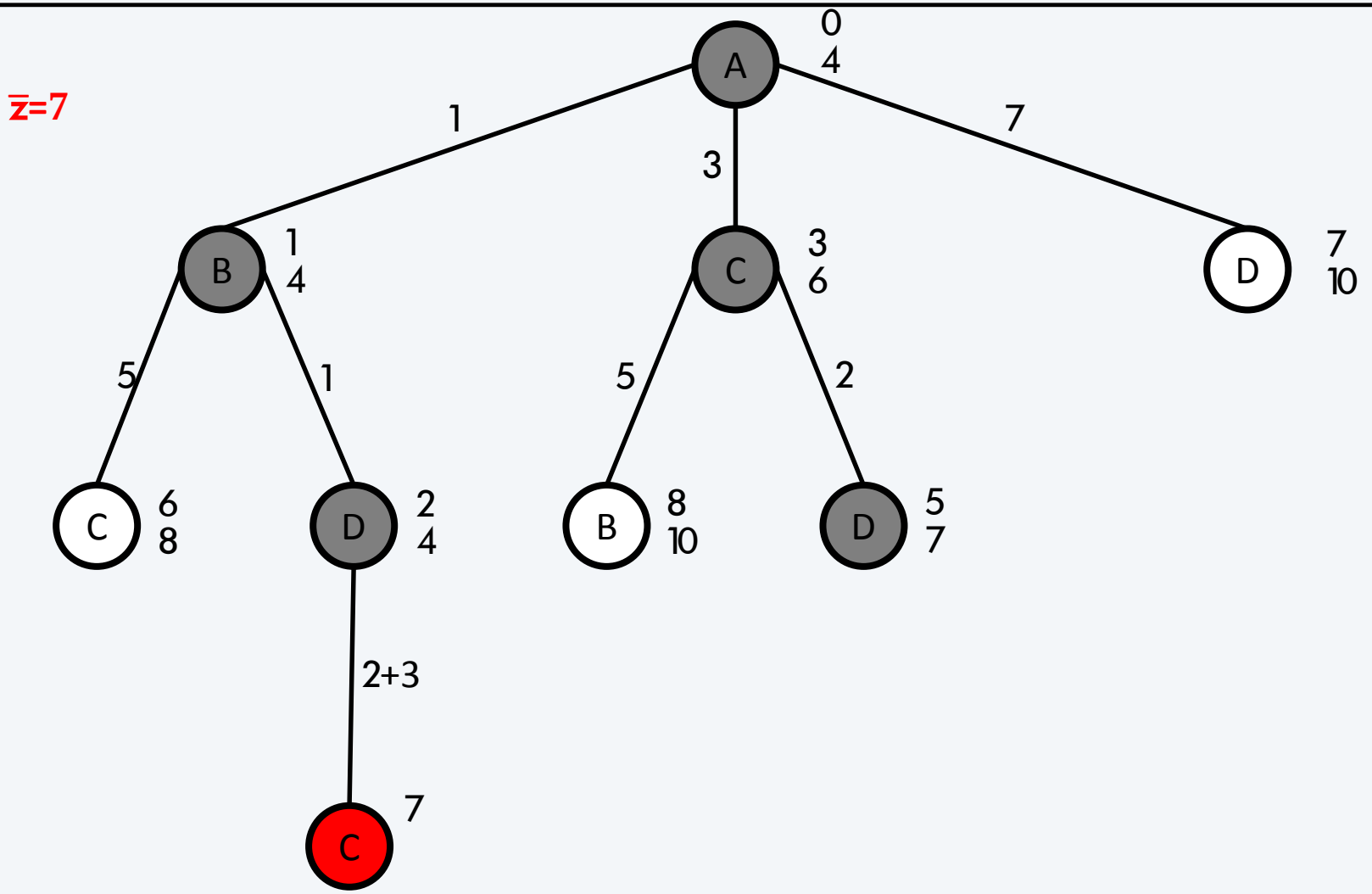



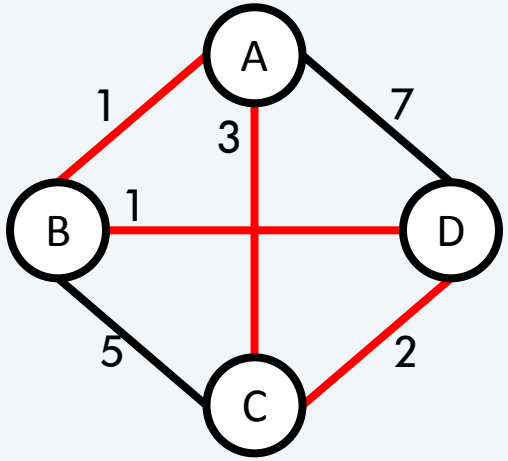
# Exemplo: BFBnB TSP



# Exemplo: BFBnB TSP



# Ordem de Percurso: DFBnB vs BFBnB

- Arvore completa possui 16 vértices.
- No exemplo, a estratégia de DFBnB gerou 10 nós e a estratégia BFBnB gerou 9 nós .

- **Deep First:**
  - Limitantes são encontrados mais rapidamente.
  - Uso de memória linear.
  - Recalculo de limitantes eficiente (Simplex Dual).
  - Custo alto, se solução ótima encontrada tarde.

- **Best First:**
  - Minimiza o número de nós explorados.
  - Nunca explora nós com limites inferiores maiores que o valor da solução ótima.
  - O algoritmo para assim que um nó solução é desempilhado.
  - Uso de memória exponencial.
  - Necessário calculo completo dos limitantes.

# Programação Inteira

- **Branching:**
  - ?
- **Bounding:**
  - ?

# Programação Inteira

- **Branching:**
  - Adição de uma restrição.

- **Bounding:**
  - ?

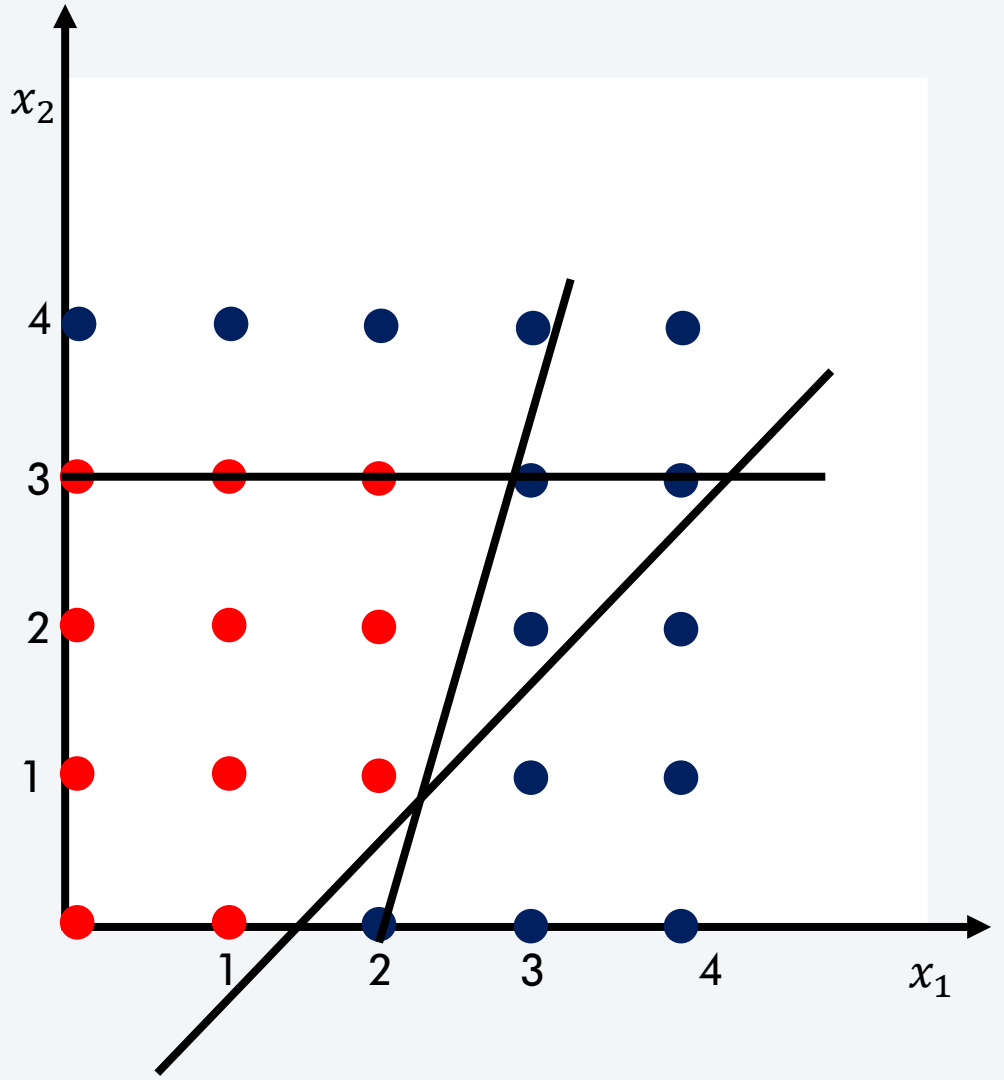
# Programação Inteira

- **Branching:**
  - Adição de uma restrição.

- **Bounding:**
  - Relaxação linear.

# Exemplo: DFBnB usando Programação Linear

$$
\begin{aligned}
\textbf{max} \quad & z = 4x_1 - x_2 \\
\textbf{s.a.} \quad & 7x_1 - 2x_2 \leq 1 \\
& 2x_1 - 2x_2 \leq 3 \\
& x_2 \leq 3 \\
& x_1 e\ x_2 inteiros
\end{aligned}
$$




- Solução do PL: $x_1^* = \frac{20}{7}, x_2^* = 3$
- Limitante Superior: $\bar{z} = \frac{59}{7}$
- Limitante Inferior: $\underline{z} = -\infty$

# Exemplo: DFBnB usando Programação Linear

Branching:

- Escolher variável fracionaria $x_j^*$ e fazer:

$$S_1 = S \cap \{x \in \mathbb{R} : x_j \leq \lfloor x_j^* \rfloor\}$$
$$S_2 = S \cap \{x \in \mathbb{R} : x_j \geq \lceil x_j^* \rceil\}$$

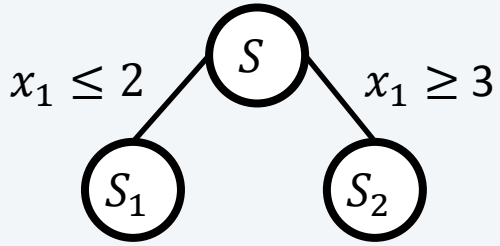
- $x^*$ não é viável nem $S_1$ nem em $S_2$.
- Limitante superior vai diminuir.

No exemplo:

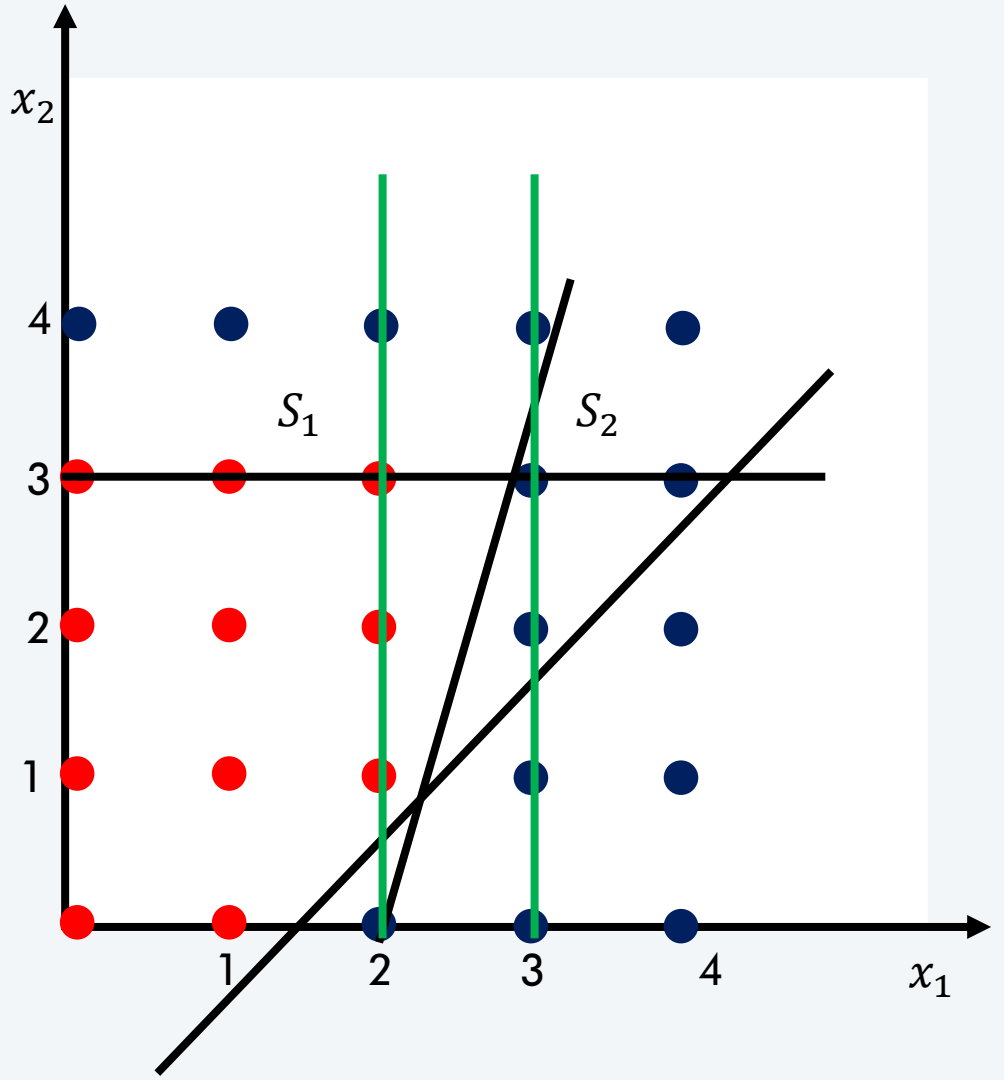
$$S_1 = S \cap \{x \in \mathbb{R} : x_1 \leq 2\}$$
$$S_2 = S \cap \{x \in \mathbb{R} : x_1 \geq 3\}$$

# Exemplo: DFBnB usando Programação Linear

$\underline{z} = -\infty$

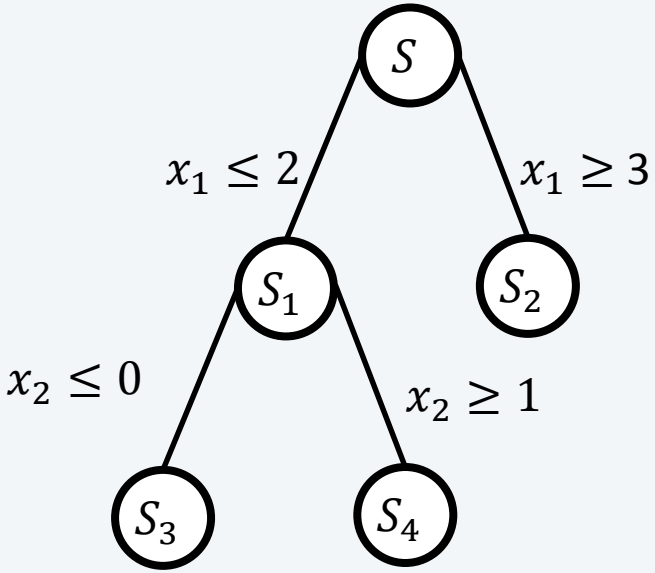


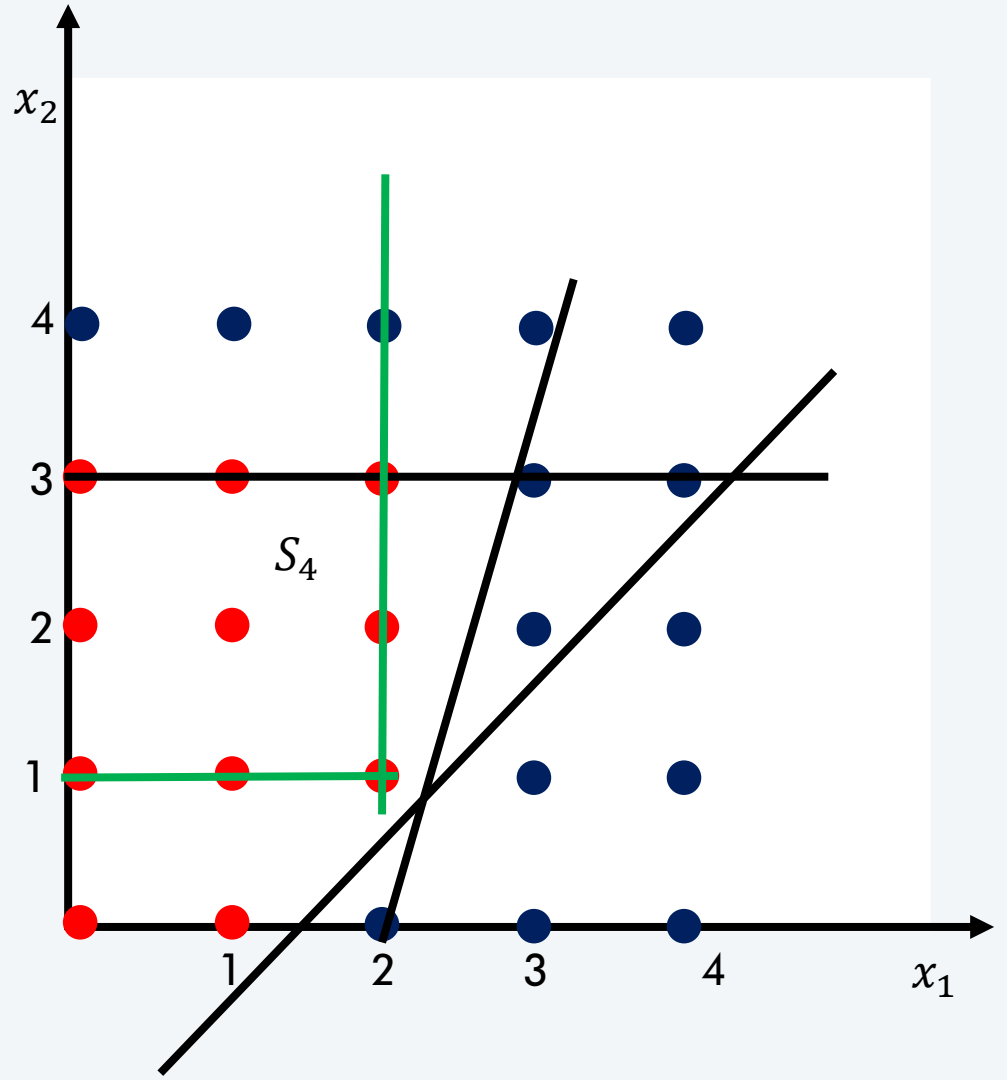

- $S_1$: $x_1^* = 2, x_2^* = \frac{1}{2}, \bar{z} = \frac{15}{2}$
- $S_2$: podado por inviabilidade.



# Exemplo: DFBnB usando Programação Linear

$\underline{z} = 7$

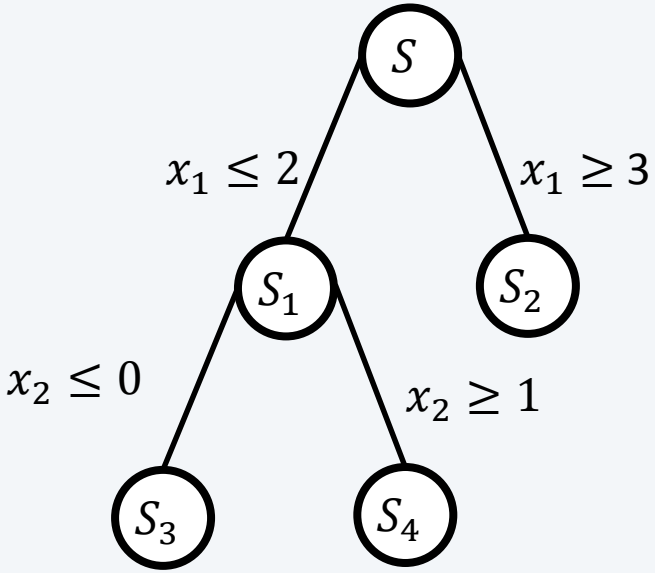


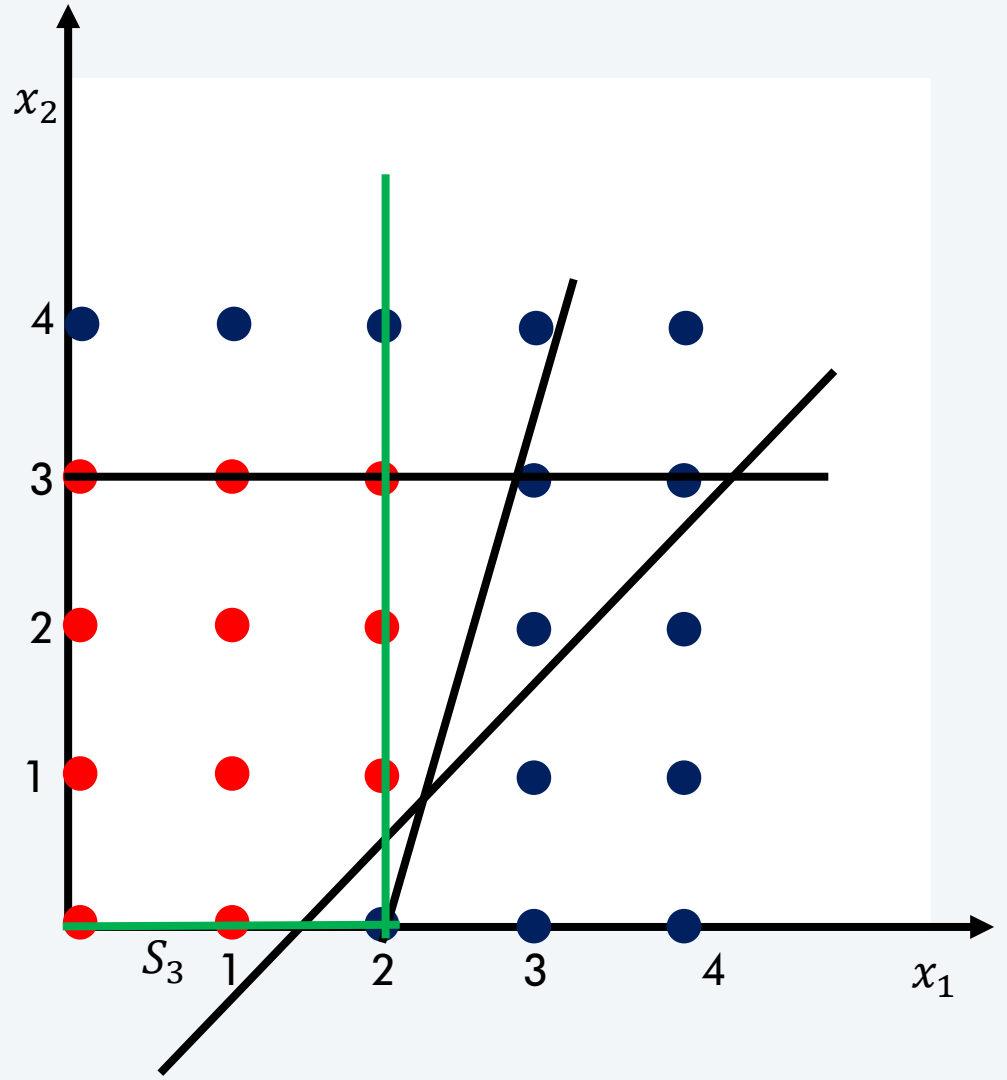




- $S_4$: $x_1^* = 2, x_2^* = 1, \bar{z} = 7$
- Solução inteira viável: poda por otimalidade.

# Exemplo: DFBnB usando Programação Linear

$\underline{z} = 7$




- $S_3$: $x_1^* = 3/2, x_2^* = 0, \bar{z} = 6$
- $S_3$ podado por limitante.

# Programação Linear: Branching

- **Variável Fracionaria:**
  - Variável mais próxima de 0.5. (Mais fracionaria (Usual))
  - Variável com maior impacto na função objetivo.
  - Variável com maior custo para se torna inteira.
  - Variável com menor índice.

- Possivelmente mais complexo e depende-te do problema.

# Implementação: Ordem de Exploração

- **DFBnB:**
  - Escolha do nó filho a ser explorado tem impacto no algoritmo.
  - Preferir nós com melhores limitantes.

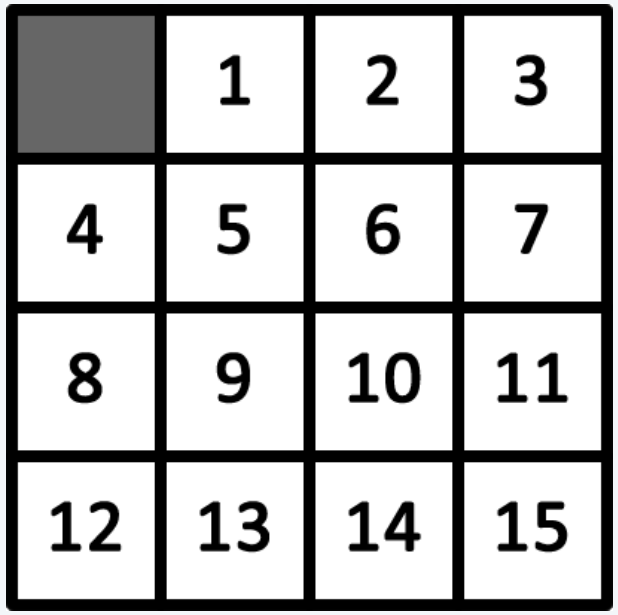
- **BFBnB:**
  - Regra de desempate para nós com limitante igual tem impacto no algoritmo.
  - Exemplo: Preferir soluções primeiro.

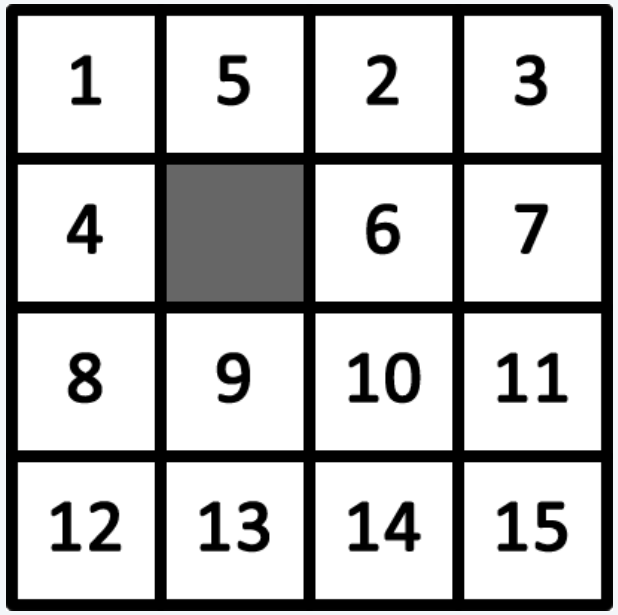
# Implementação: Heurísticas e Limitantes

- **Heurísticas:**
  - Uso de heurísticas para tentar integralizar soluções e assim obter limitantes mais cedo na busca.

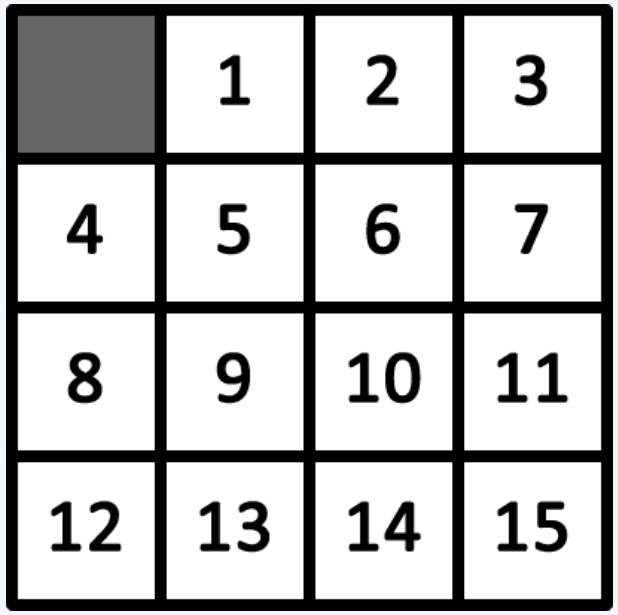
- **Limitantes:**
  - Qualidade dos limitantes tem grande impacto na eficiência do algoritmo.
  - Limitantes mais eficientes vs custo computacional.

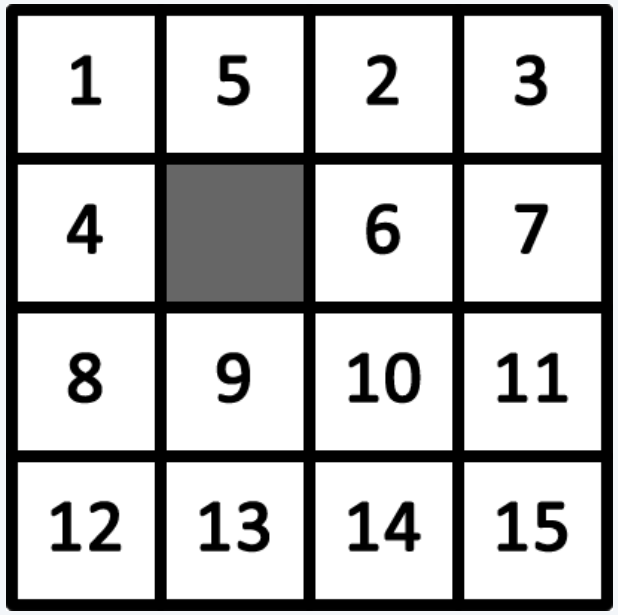
# Implementação: Limitantes



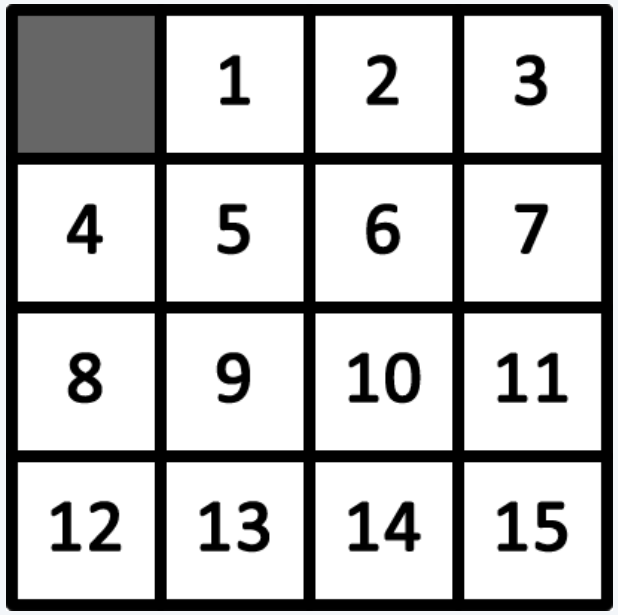
# Implementação: Limitantes



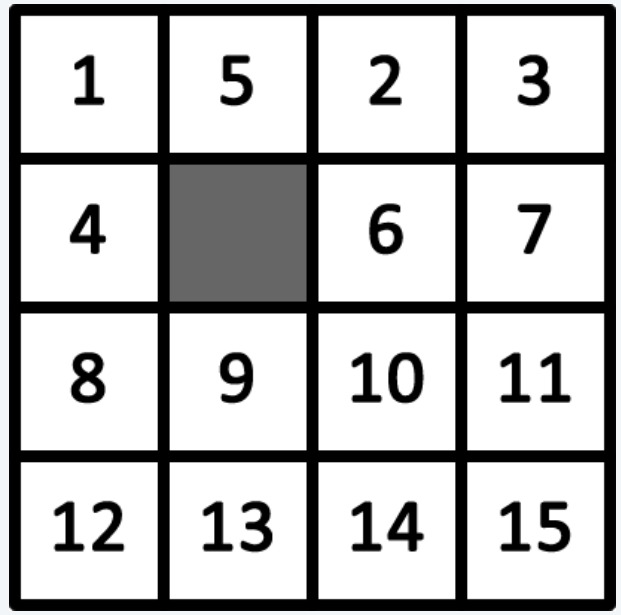




- 24-Puzzle e Distance Manhattan: 65 mil anos.

# Implementação: Limitantes



| | | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 |
| 12 | 13 | 14 | 15 |



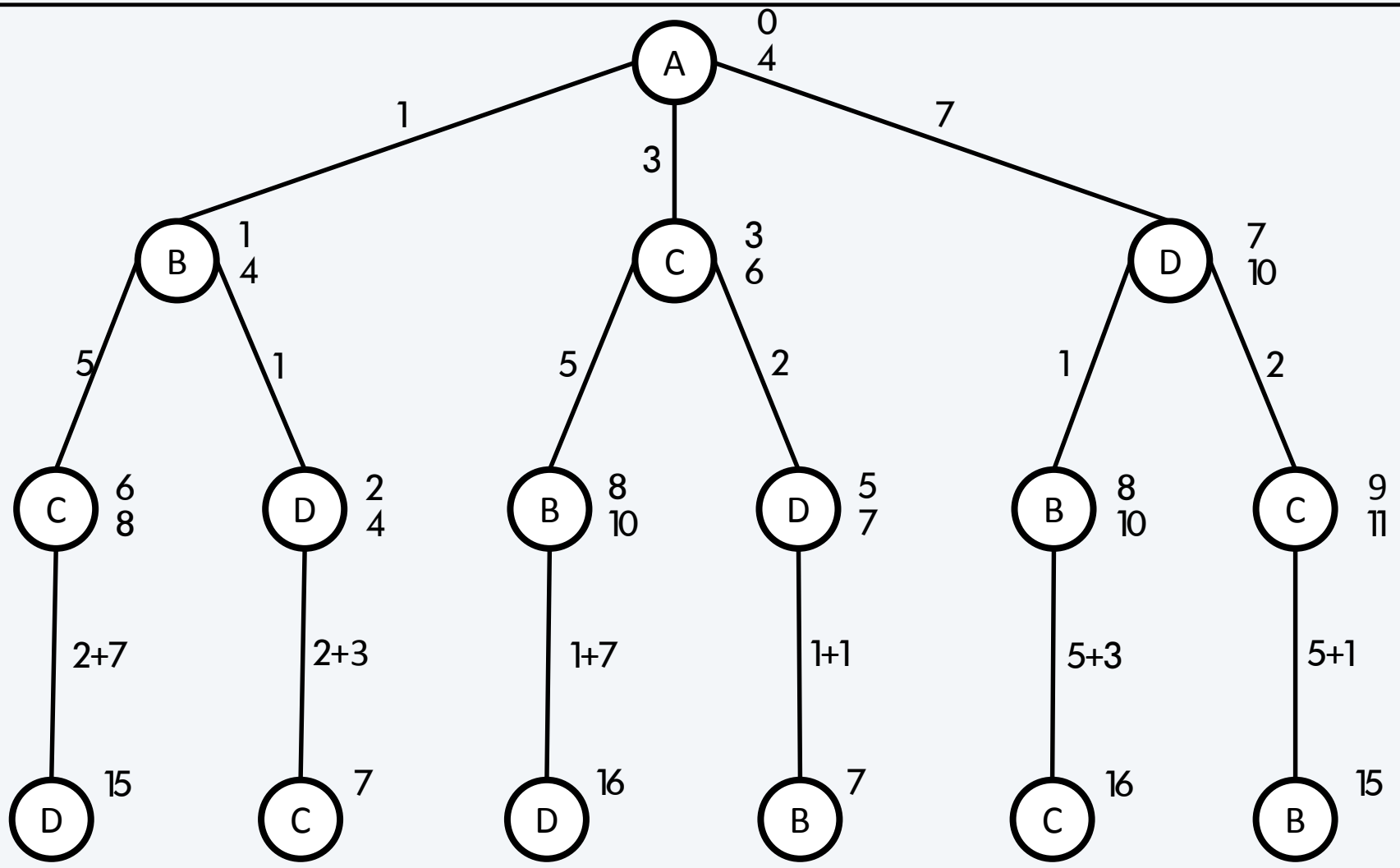
| | | | |
|---|---|---|---|
| 1 | 5 | 2 | 3 |
| 4 | | 6 | 7 |
| 8 | 9 | 10 | 11 |
| 12 | 13 | 14 | 15 |

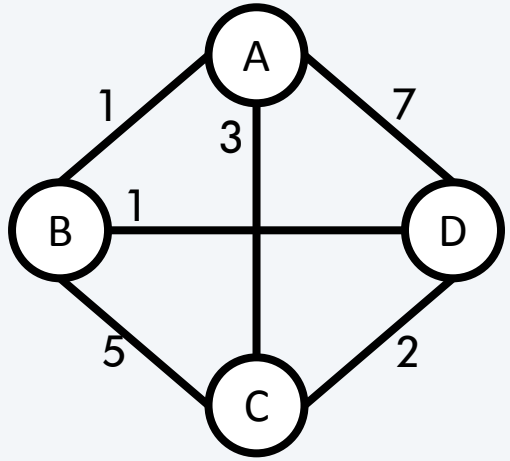
- 24-Puzzle e Distance Manhattan: 65 mil anos.
- 24-Puzzle e Pattern Databases: de segundos a algumas horas.

# Referências

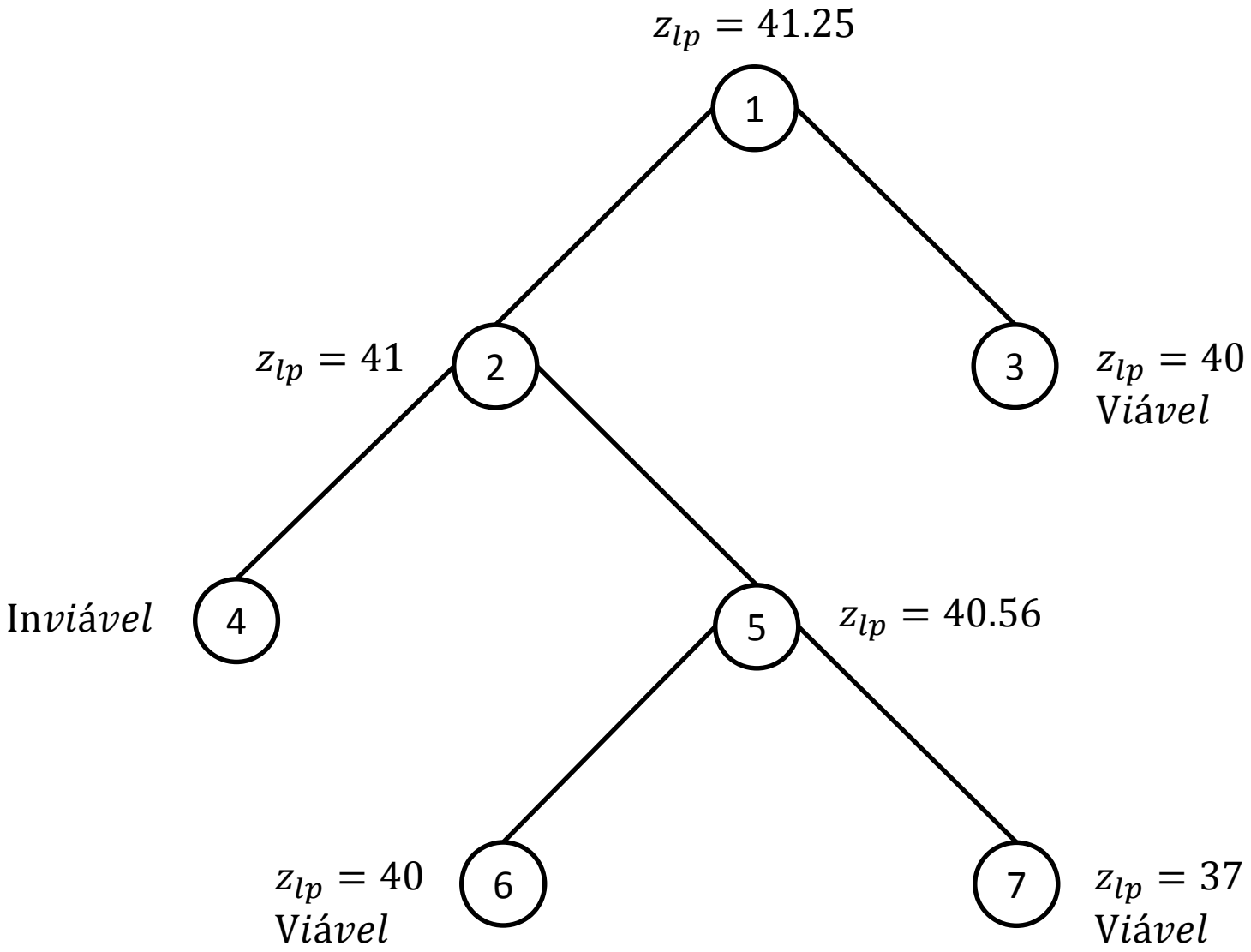

- Algorithms, S. Dasgupta, C.H. Papadimitriou, and U.V. Vazirani.
- Heuristic Search: Theory and Applications, S. Edelkamp, S. Schroedl.
- Notas de Aula, INF05010 - Otimização Combinatoria.
- Applied Integer Programming Modeling and Solution, D.S. Chen, R. G. Batson, Y. Dang.
- Slides, Prof. Cid Carvalho de Souza.

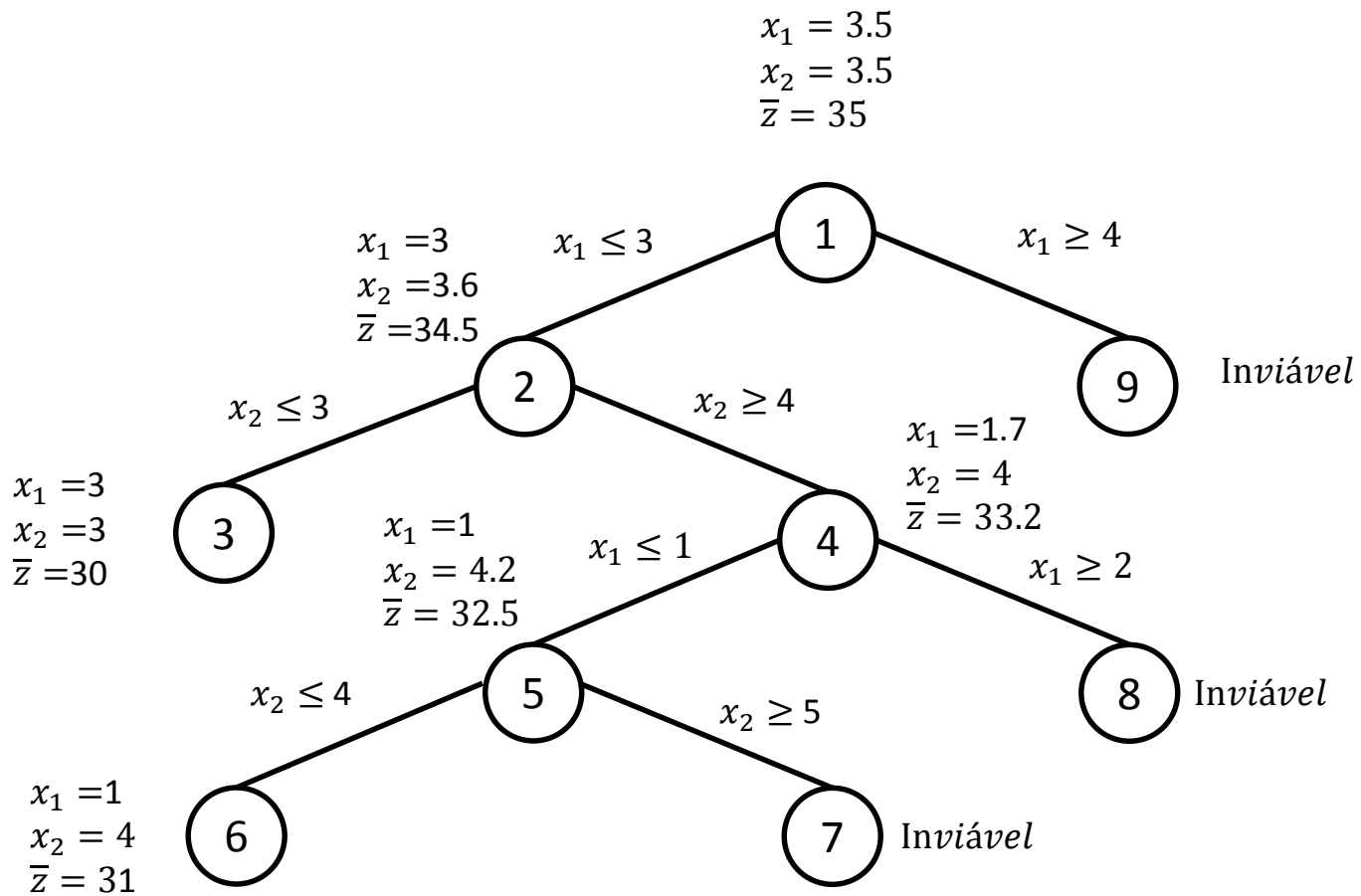
# Exemplo: TSP

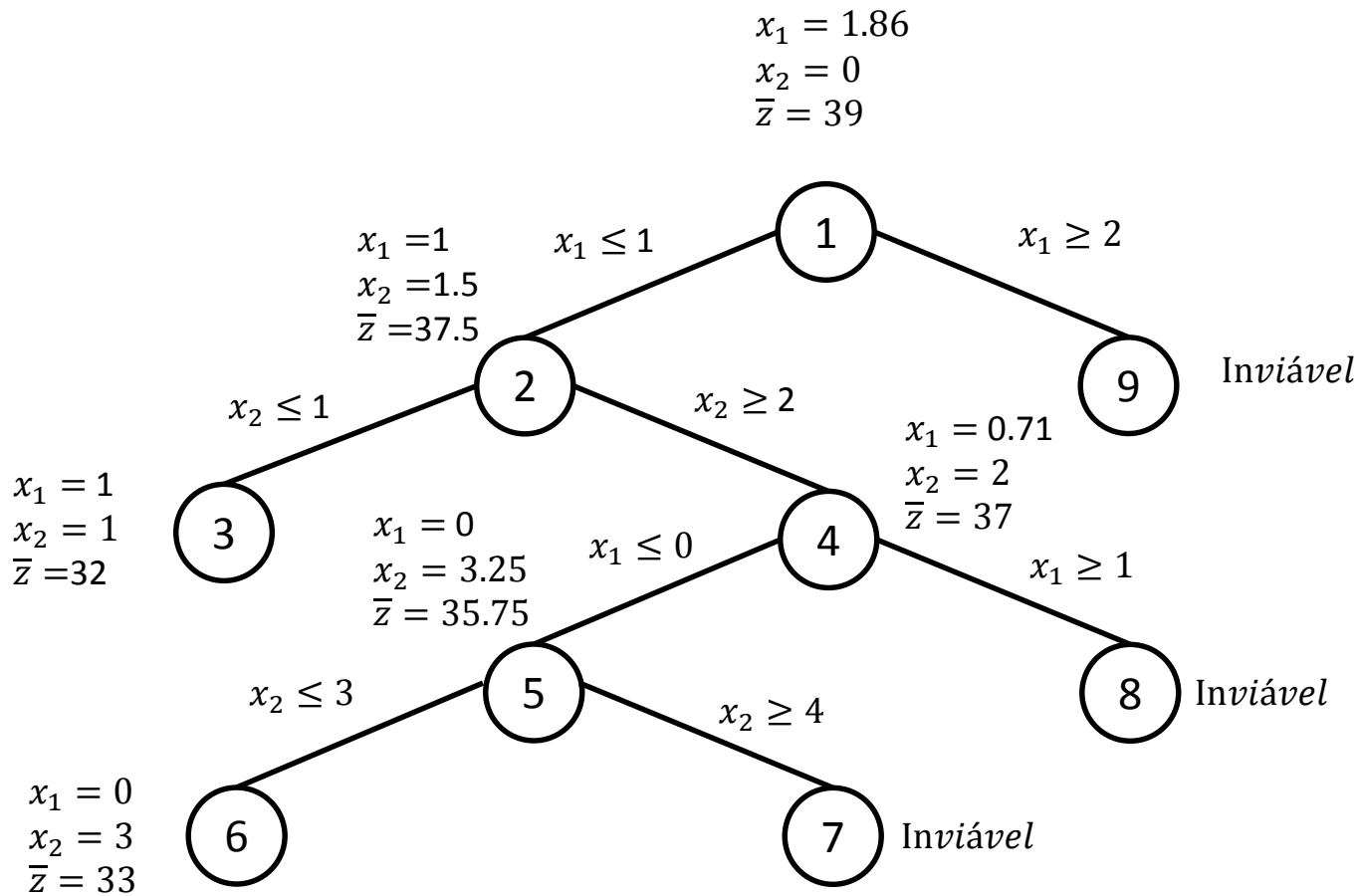

A
0
4
1
3
7
B
1
4
C
3
6
D
7
10
5
1
5
2
1
2
C
6
8
D
2
4
B
8
10
D
5
7
B
8
10
C
9
11
2+7
2+3
1+7
1+1
5+3
5+1
D
15
C
7
D
16
B
7
C
16
B
15

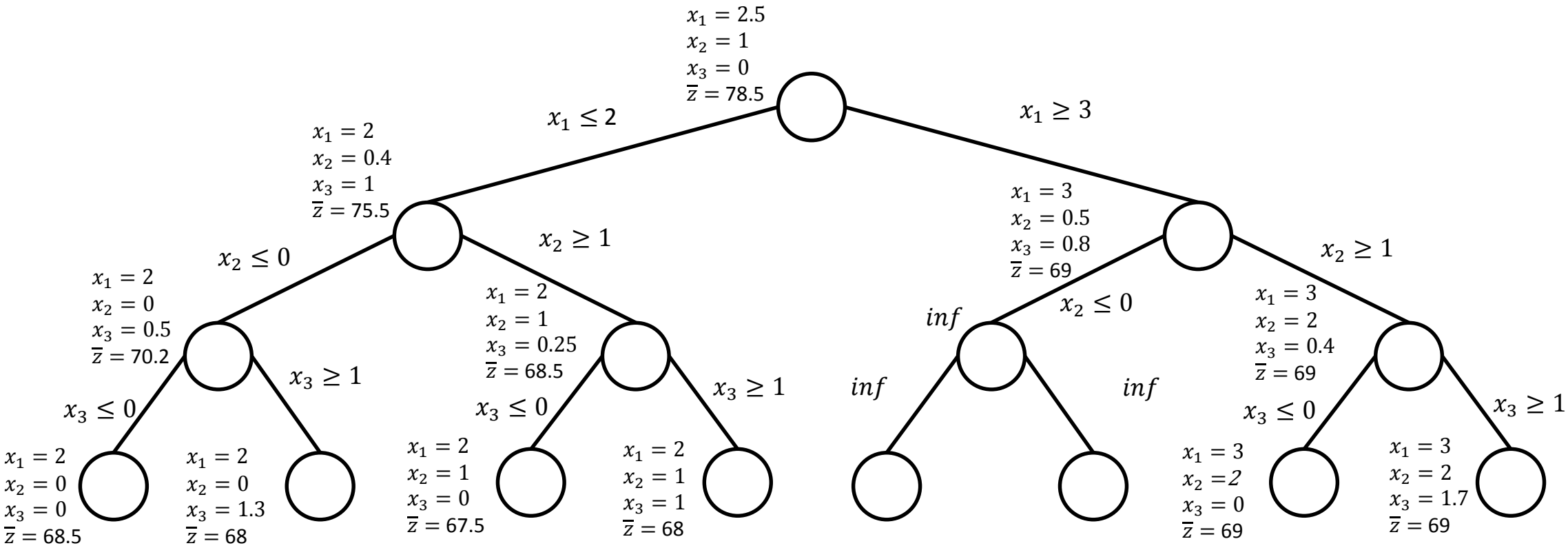

A
1
3
7
B
1
D
5
2
C

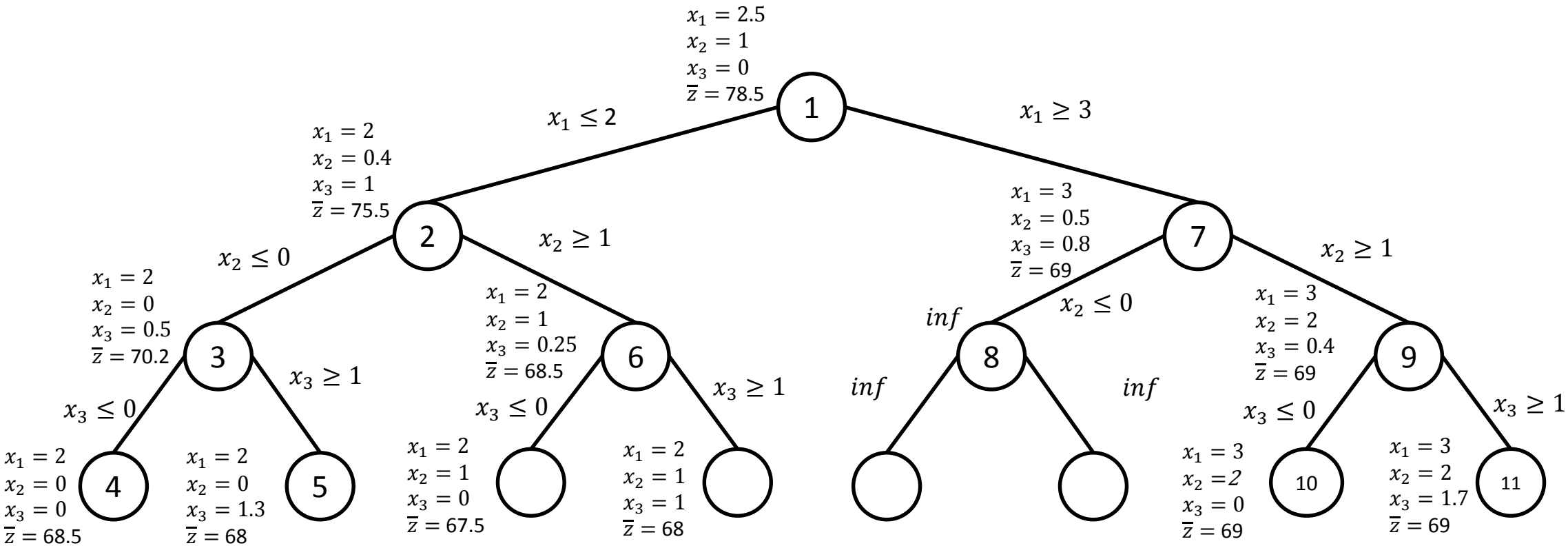
# Figura questão 1:

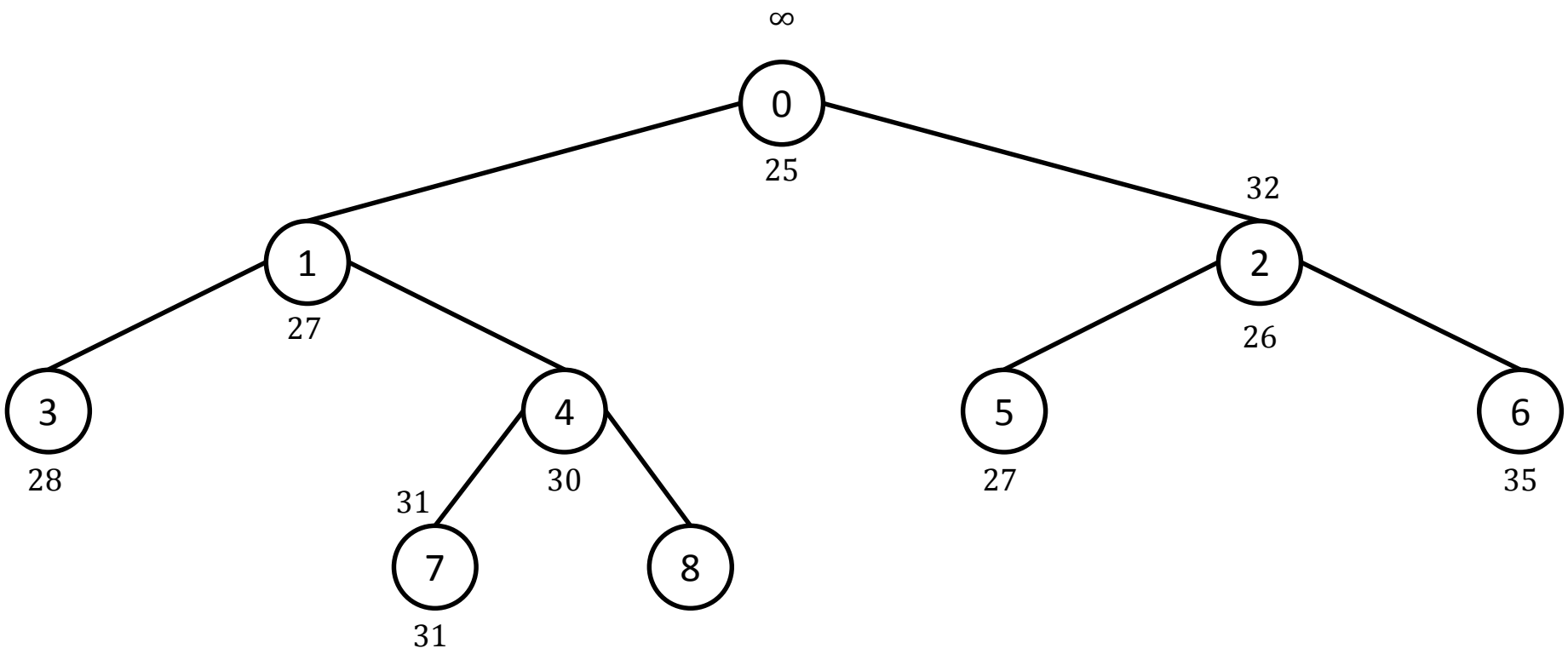


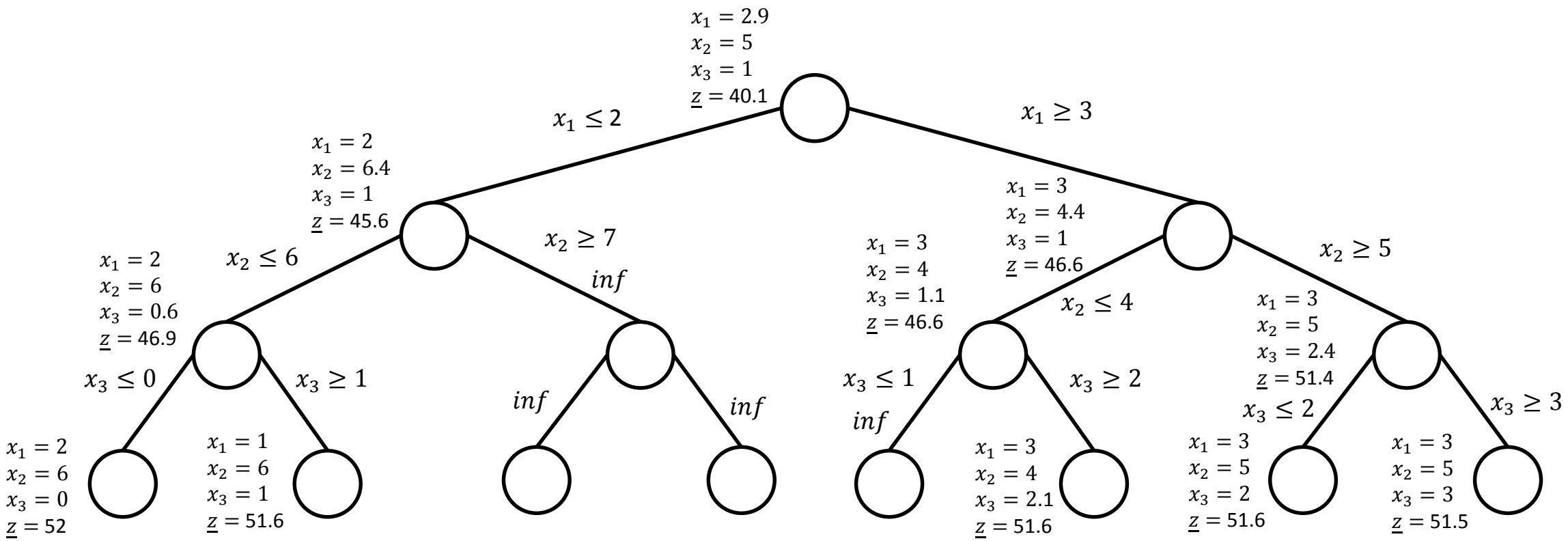

$x_1 = 3.5$
$x_2 = 3.5$
$\overline{z} = 35$
1
$x_1 \leq 3$
$x_1 \geq 4$
$x_1$ =3
$x_2$ =3.6
$\overline{z}$ =34.5
2
9
Inviável
$x_2 \leq 3$
$x_2 \geq 4$
$x_1$ =1.7
$x_2$ = 4
$\overline{z}$ = 33.2
$x_1$ =3
$x_2$ =3
$\overline{z}$ =30
3
$x_1$ =1
$x_2$ = 4.2
$\overline{z}$ = 32.5
$x_1 \leq 1$
4
$x_1 \geq 2$
$x_2 \leq 4$
5
$x_2 \geq 5$
8
Inviável
$x_1$ =1
$x_2$ = 4
$\overline{z}$ = 31
6
7
Inviável

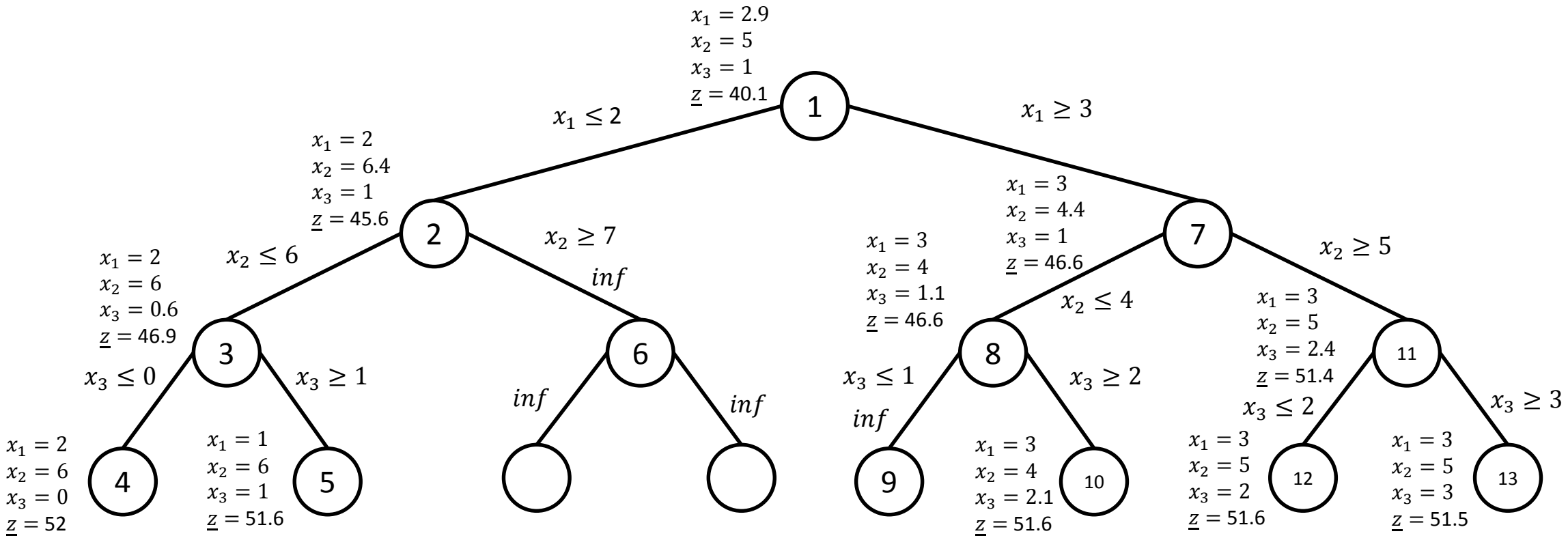

$x_1 = 1.86$
$x_2 = 0$
$\overline{z} = 39$
1
$x_1 \leq 1$
$x_1 \geq 2$
$x_1 = 1$
$x_2 = 1.5$
$\overline{z} = 37.5$
2
9
Inviável
$x_2 \leq 1$
$x_2 \geq 2$
$x_1 = 0.71$
$x_2 = 2$
$\overline{z} = 37$
$x_1 = 1$
$x_2 = 1$
$\overline{z} = 32$
3
$x_1 = 0$
$x_2 = 3.25$
$\overline{z} = 35.75$
$x_1 \leq 0$
4
$x_1 \geq 1$
$x_2 \leq 3$
5
8
Inviável
$x_2 \geq 4$
$x_1 = 0$
$x_2 = 3$
$\overline{z} = 33$
6
7
Inviável


$x_1 = 2.5$
$x_2 = 1$
$x_3 = 0$
$\overline{z} = 78.5$
$x_1 \leq 2$
$x_1 \geq 3$
$x_1 = 2$
$x_2 = 0.4$
$x_3 = 1$
$\overline{z} = 75.5$
$x_1 = 3$
$x_2 = 0.5$
$x_3 = 0.8$
$\overline{z} = 69$
$x_2 \leq 0$
$x_2 \geq 1$
$x_2 \leq 0$
$x_2 \geq 1$
$x_1 = 2$
$x_2 = 0$
$x_3 = 0.5$
$\overline{z} = 70.2$
$x_1 = 2$
$x_2 = 1$
$x_3 = 0.25$
$\overline{z} = 68.5$
$inf$
$inf$
$x_1 = 3$
$x_2 = 2$
$x_3 = 0.4$
$\overline{z} = 69$
$x_3 \leq 0$
$x_3 \geq 1$
$x_3 \leq 0$
$x_3 \geq 1$
$x_3 \leq 0$
$x_3 \geq 1$
$x_1 = 2$
$x_2 = 0$
$x_3 = 0$
$\overline{z} = 68.5$
$x_1 = 2$
$x_2 = 0$
$x_3 = 1.3$
$\overline{z} = 68$
$x_1 = 2$
$x_2 = 1$
$x_3 = 0$
$\overline{z} = 67.5$
$x_1 = 2$
$x_2 = 1$
$x_3 = 1$
$\overline{z} = 68$
$x_1 = 3$
$x_2 = 2$
$x_3 = 0$
$\overline{z} = 69$
$x_1 = 3$
$x_2 = 2$
$x_3 = 1.7$
$\overline{z} = 69$


$x_1 = 2.5$
$x_2 = 1$
$x_3 = 0$
$\overline{z} = 78.5$
1
$x_1 \leq 2$
$x_1 \geq 3$
$x_1 = 2$
$x_2 = 0.4$
$x_3 = 1$
$\overline{z} = 75.5$
2
$x_1 = 3$
$x_2 = 0.5$
$x_3 = 0.8$
$\overline{z} = 69$
7
$x_2 \leq 0$
$x_2 \geq 1$
$x_2 \leq 0$
$x_2 \geq 1$
$x_1 = 2$
$x_2 = 0$
$x_3 = 0.5$
$\overline{z} = 70.2$
3
$x_1 = 2$
$x_2 = 1$
$x_3 = 0.25$
$\overline{z} = 68.5$
6
$inf$
8
$x_1 = 3$
$x_2 = 2$
$x_3 = 0.4$
$\overline{z} = 69$
9
$x_3 \leq 0$
$x_3 \geq 1$
$x_3 \leq 0$
$x_3 \geq 1$
$inf$
$inf$
$x_3 \leq 0$
$x_3 \geq 1$
$x_1 = 2$
$x_2 = 0$
$x_3 = 0$
$\overline{z} = 68.5$
4
$x_1 = 2$
$x_2 = 0$
$x_3 = 1.3$
$\overline{z} = 68$
5
$x_1 = 2$
$x_2 = 1$
$x_3 = 0$
$\overline{z} = 67.5$
$x_1 = 2$
$x_2 = 1$
$x_3 = 1$
$\overline{z} = 68$
$x_1 = 3$
$x_2 = 2$
$x_3 = 0$
$\overline{z} = 69$
10
$x_1 = 3$
$x_2 = 2$
$x_3 = 1.7$
$\overline{z} = 69$
11


∞
0
25
32
1
27
2
26
3
28
4
30
5
27
6
35
31
7
31
8


$x_1 = 2.9$
$x_2 = 5$
$x_3 = 1$
$\underline{z} = 40.1$
$x_1 \leq 2$
$x_1 \geq 3$
$x_1 = 2$
$x_2 = 6.4$
$x_3 = 1$
$\underline{z} = 45.6$
$x_1 = 3$
$x_2 = 4.4$
$x_3 = 1$
$\underline{z} = 46.6$
$x_2 \leq 6$
$x_2 \geq 7$
$inf$
$x_1 = 3$
$x_2 = 4$
$x_3 = 1.1$
$\underline{z} = 46.6$
$x_2 \leq 4$
$x_2 \geq 5$
$x_1 = 2$
$x_2 = 6$
$x_3 = 0.6$
$\underline{z} = 46.9$
$x_3 \leq 0$
$x_3 \geq 1$
$inf$
$inf$
$x_3 \leq 1$
$x_3 \geq 2$
$x_1 = 3$
$x_2 = 5$
$x_3 = 2.4$
$\underline{z} = 51.4$
$x_3 \leq 2$
$x_3 \geq 3$
$x_1 = 2$
$x_2 = 6$
$x_3 = 0$
$\underline{z} = 52$
$x_1 = 1$
$x_2 = 6$
$x_3 = 1$
$\underline{z} = 51.6$
$inf$
$x_1 = 3$
$x_2 = 4$
$x_3 = 2.1$
$\underline{z} = 51.6$
$x_1 = 3$
$x_2 = 5$
$x_3 = 2$
$\underline{z} = 51.6$
$x_1 = 3$
$x_2 = 5$
$x_3 = 3$
$\underline{z} = 51.5$


$x_1 = 2.9$
$x_2 = 5$
$x_3 = 1$
$\underline{z} = 40.1$
1
$x_1 \leq 2$
$x_1 \geq 3$
$x_1 = 2$
$x_2 = 6.4$
$x_3 = 1$
$\underline{z} = 45.6$
2
$x_2 \leq 6$
$x_2 \geq 7$
$inf$
$x_1 = 3$
$x_2 = 4.4$
$x_3 = 1$
$\underline{z} = 46.6$
7
$x_1 = 3$
$x_2 = 4$
$x_3 = 1.1$
$\underline{z} = 46.6$
$x_2 \leq 4$
$x_2 \geq 5$
$x_1 = 2$
$x_2 = 6$
$x_3 = 0.6$
$\underline{z} = 46.9$
3
$x_3 \leq 0$
$x_3 \geq 1$
6
8
11
$x_1 = 3$
$x_2 = 5$
$x_3 = 2.4$
$\underline{z} = 51.4$
$x_3 \leq 1$
$x_3 \geq 2$
$x_3 \leq 2$
$x_3 \geq 3$
$inf$
$inf$
$inf$
$x_1 = 2$
$x_2 = 6$
$x_3 = 0$
$\underline{z} = 52$
4
$x_1 = 1$
$x_2 = 6$
$x_3 = 1$
$\underline{z} = 51.6$
5
9
$x_1 = 3$
$x_2 = 4$
$x_3 = 2.1$
$\underline{z} = 51.6$
10
$x_1 = 3$
$x_2 = 5$
$x_3 = 2$
$\underline{z} = 51.6$
12
$x_1 = 3$
$x_2 = 5$
$x_3 = 3$
$\underline{z} = 51.5$
13