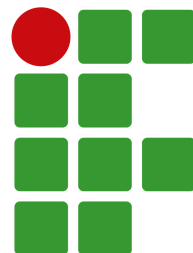


# Inteligência Artificial

## Aula 04 – Busca Heurística



**INSTITUTO FEDERAL**

Espírito Santo  
Campus Serra

# Métodos de Busca

- **Busca Cega ou Exaustiva:**

- Não sabe qual o melhor nó da fronteira a ser expandido. Apenas distingue o estado objetivo dos não objetivos.

- **Busca Heurística:**

- **Estima qual o melhor nó da fronteira a ser expandido com base em funções heurísticas.**

- **Busca Local:**

- Operam em um único estado e movem-se para a vizinhança deste estado.

# Busca Heurística

- **Algoritmos de Busca Heurística:**
  - Busca Gulosa
  - $A^*$
- A busca heurística leva em conta o **objetivo** para decidir qual caminho escolher.
- Conhecimento extra sobre o problema é utilizado para **guiar o processo de busca**.

# Busca Heurística

- Como encontrar um barco perdido?
  - **Busca Cega** -> Procura no oceano inteiro.
  - **Busca Heurística** -> Procura utilizando informações relativas ao problema.
    - Exemplo: correntes marítimas, vento, etc.

# Busca Heurística

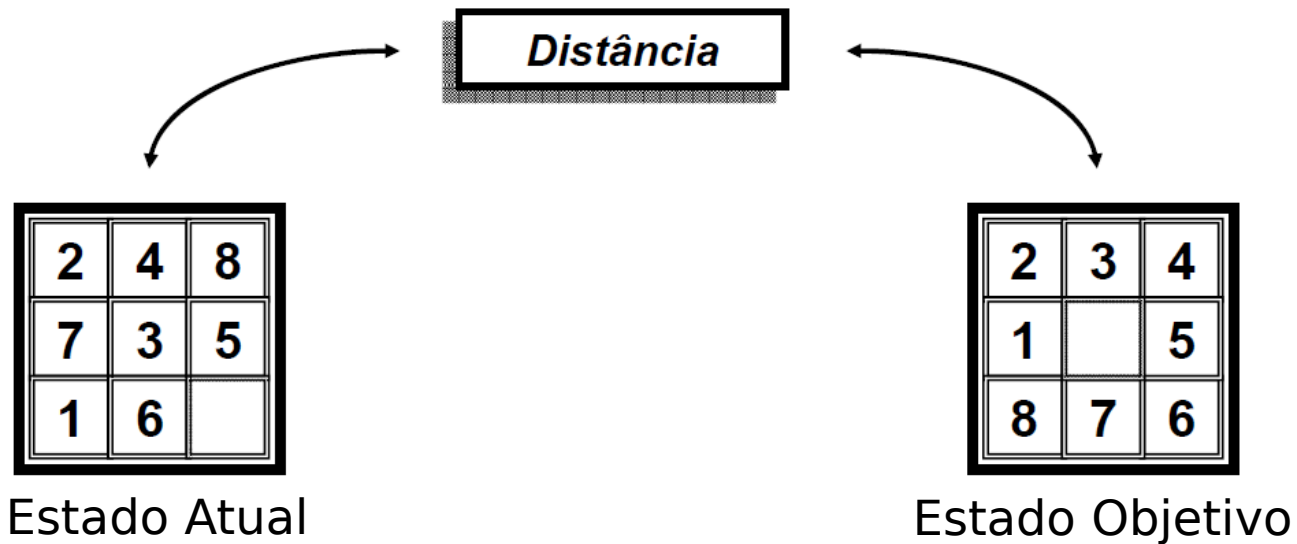
- **Função Heurística ( $h$ )**

- Estima o custo do caminho mais barato do estado atual até o estado final mais próximo.
- São específicas para cada problema.

- **Exemplo:**

- Encontrar a rota mais curta entre duas cidades:
  - $h(n)$  = distância em linha reta direta entre o nó  $n$  e o nó final.

# Função Heurística



$$h_1 \left( \begin{array}{|c|c|c|} \hline 2 & 4 & 8 \\ \hline 7 & 3 & 5 \\ \hline 1 & 6 & \\ \hline \end{array} \right) = ?$$

# Busca Heurística

- **Algoritmos de Busca Heurística:**

- Busca Gulosa

- $A^*$

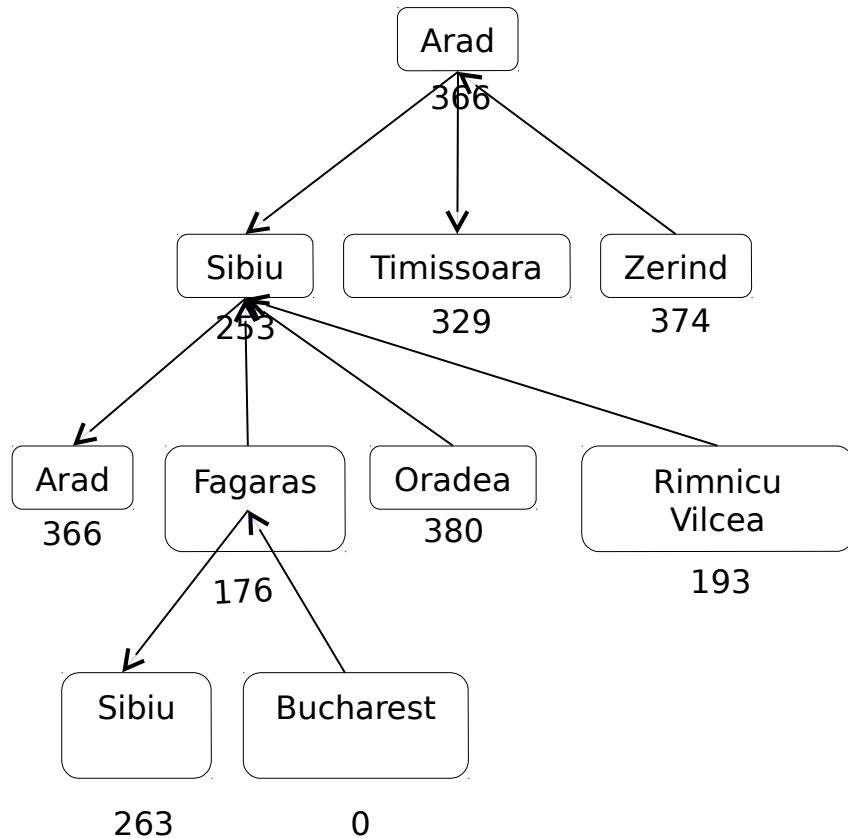
# Busca Gulosa

- **Estratégia:**

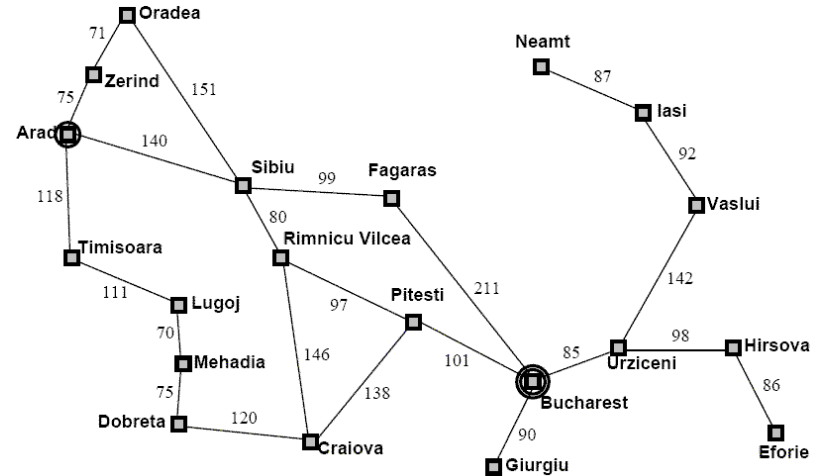
- Expande os nós que se encontram mais próximos do objetivo (uma linha reta conectando os dois pontos no caso de distâncias), desta maneira é provável que a busca encontre uma solução rapidamente.
- A implementação do algoritmo se assemelha ao utilizado na busca cega, entretanto utiliza-se uma função heurística para decidir qual o nó deve ser expandido.



# Busca Gulosa



**Função Heurística (h):** Distancia em linha reta(hDLR)



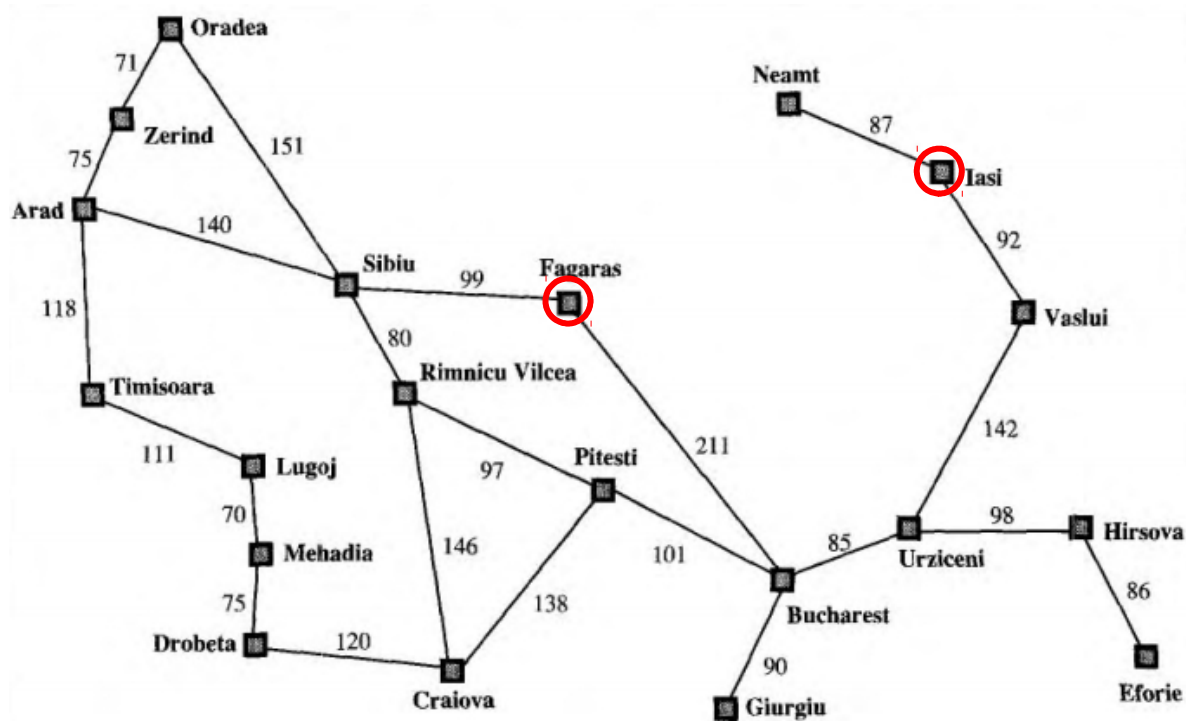
Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80

# Busca Gulosa

- **Custo de busca mínimo:**
  - No exemplo, não expande nós fora do caminho.
- **Não é ótima:**
  - No exemplo, escolhe o caminho que é mais econômico à primeira vista, via Fagaras.
  - Porém, existe um caminho mais curto via Rimnicu Vilcea.
- **Não é completa:**
  - Pode entrar em loop se não detectar a expansão de estados repetidos.
  - Pode tentar desenvolver um caminho infinito.

# Busca Gulosa

- Ir de **Iasi** para **Fagaras**?



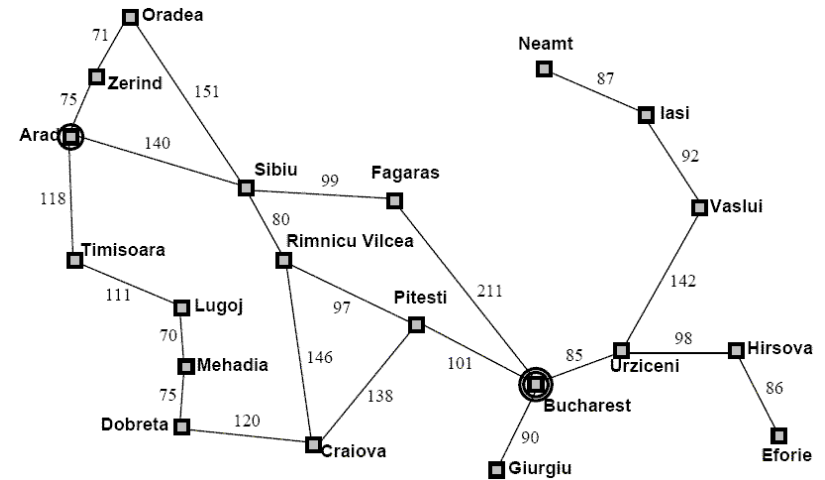
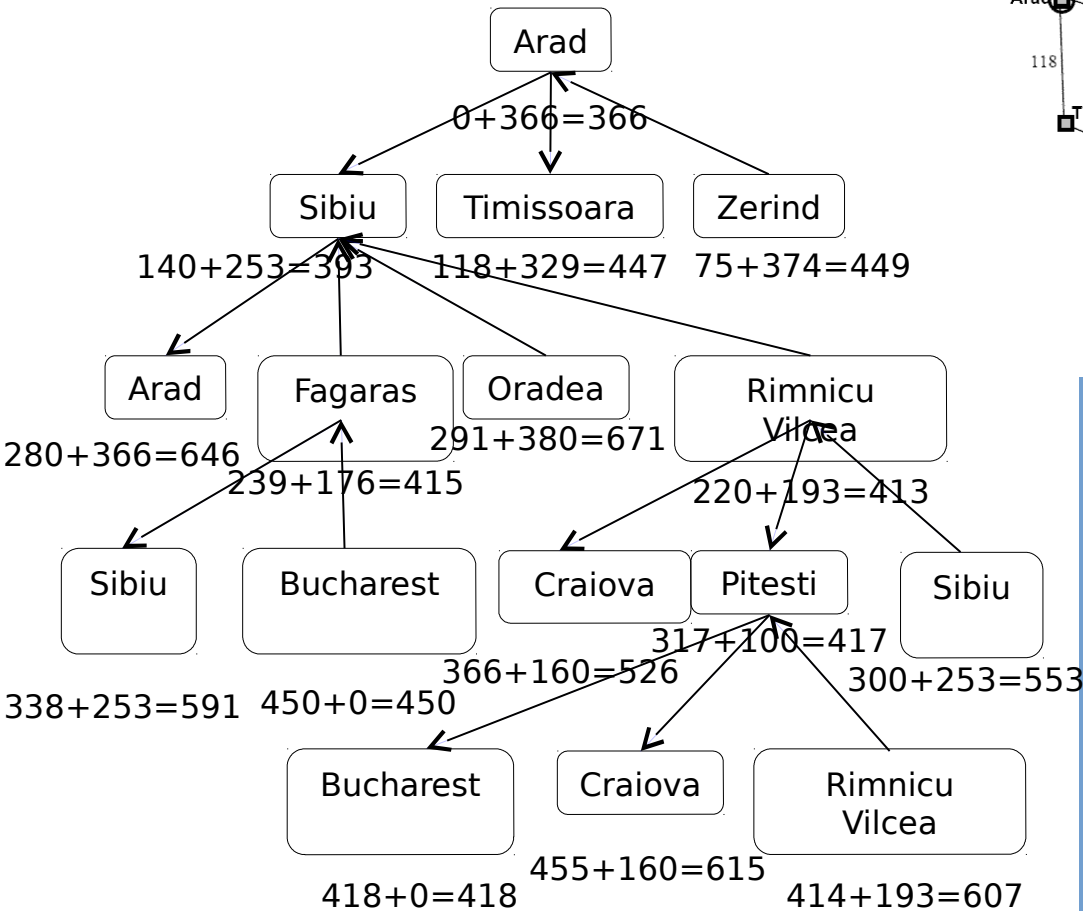
# Busca A\*

- **Estratégia:**

- Combina o custo do caminho  $g(n)$  com o valor da heurística  $h(n)$
- $g(n)$  = custo do caminho do nó inicial até o nó  $n$
- $h(n)$  = valor da heurística do nó  $n$  até um nó objetivo (distancia em linha reta no caso de distancias espaciais)
- $f(n) = g(n) + h(n)$

- **É a técnica de busca mais utilizada.**

# Busca A\*



Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80

# Busca A\*

- A estratégia é **completa** e **ótima**.
- **Custo de tempo:**
  - Exponencial com o comprimento da solução, porém boas funções heurísticas diminuem significativamente esse custo.
- **Custo memória:**
  - Guarda todos os nós expandidos na memória.
- Nenhum outro algoritmo ótimo garante expandir menos nós.

# Definindo Heurísticas

- Cada problema **exige** uma função heurística diferente.
- Não se deve superestimar o custo real da solução.
- Como escolher uma boa função heurística para o jogo 8-Puzzle?

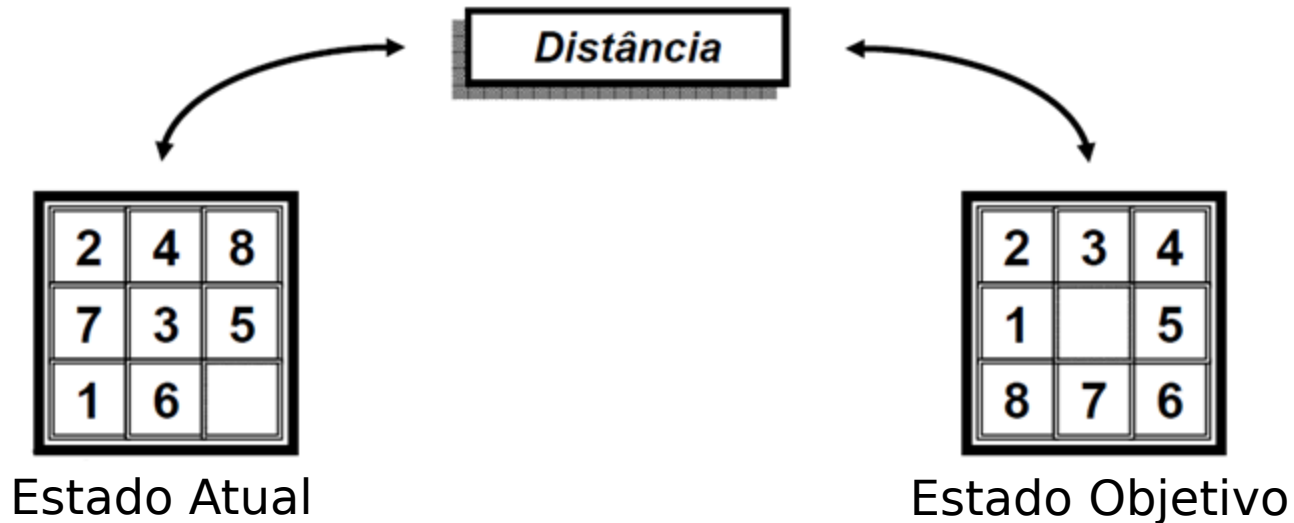
7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

# Definindo Heurísticas

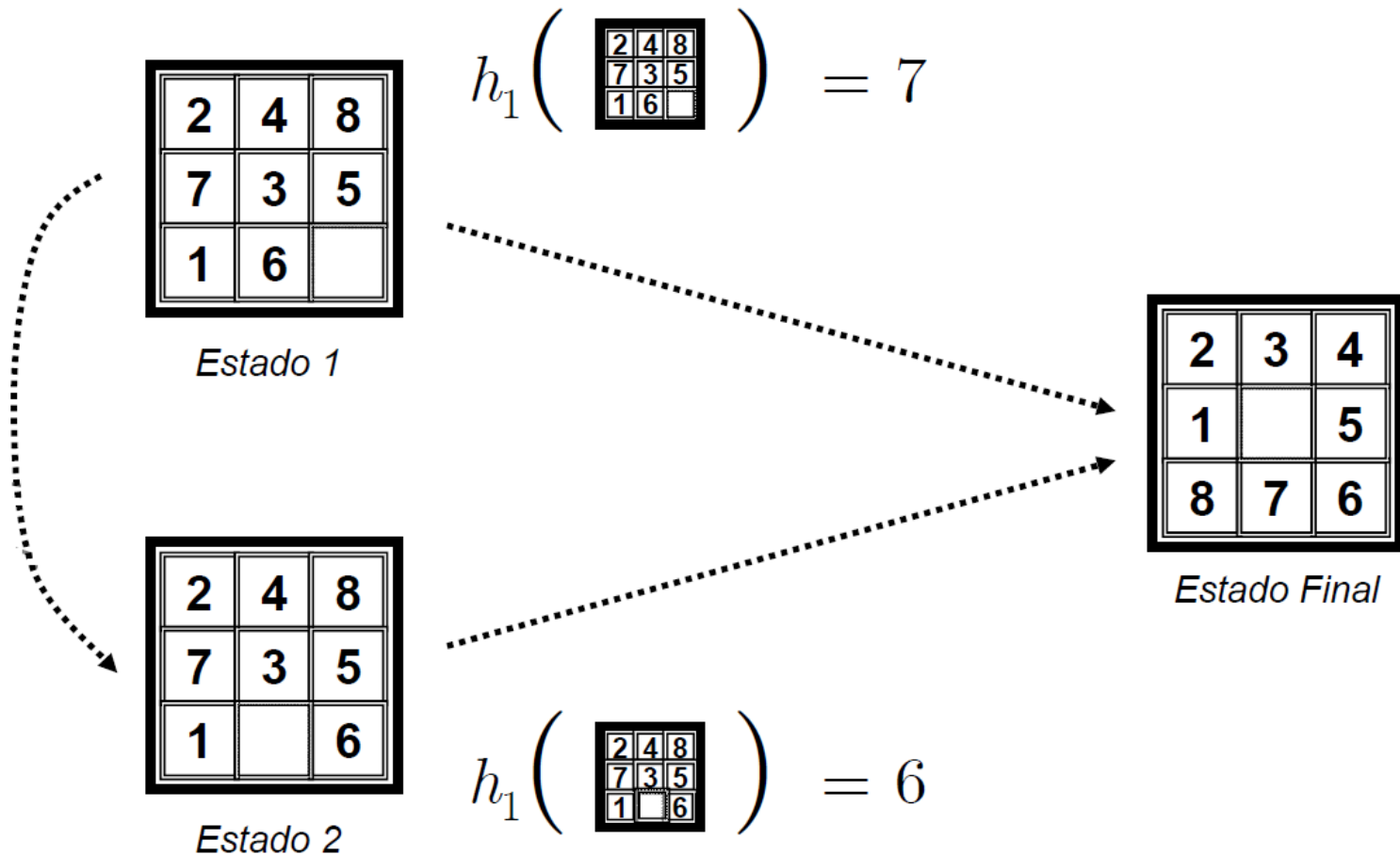


$$h_1 \left( \begin{array}{|c|c|c|} \hline 2 & 4 & 8 \\ \hline 7 & 3 & 5 \\ \hline 1 & 6 & \\ \hline \end{array} \right) = \begin{array}{l} \text{A quantidade de} \\ \text{peças fora do} \\ \text{lugar} \\ = 7 \end{array}$$

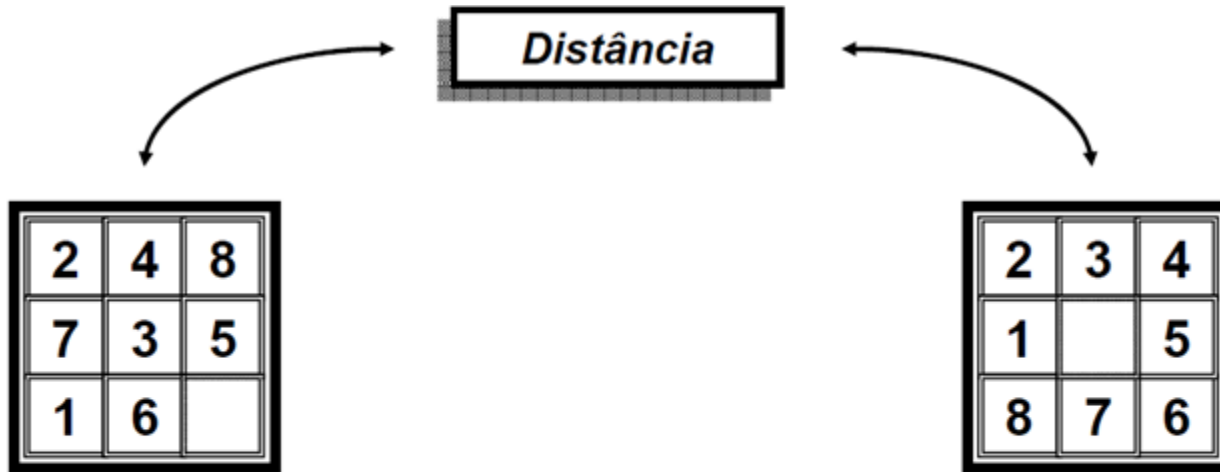
$h_1 = \text{n}^\circ \text{ de elementos em posições erradas}$



# Definindo Heurísticas



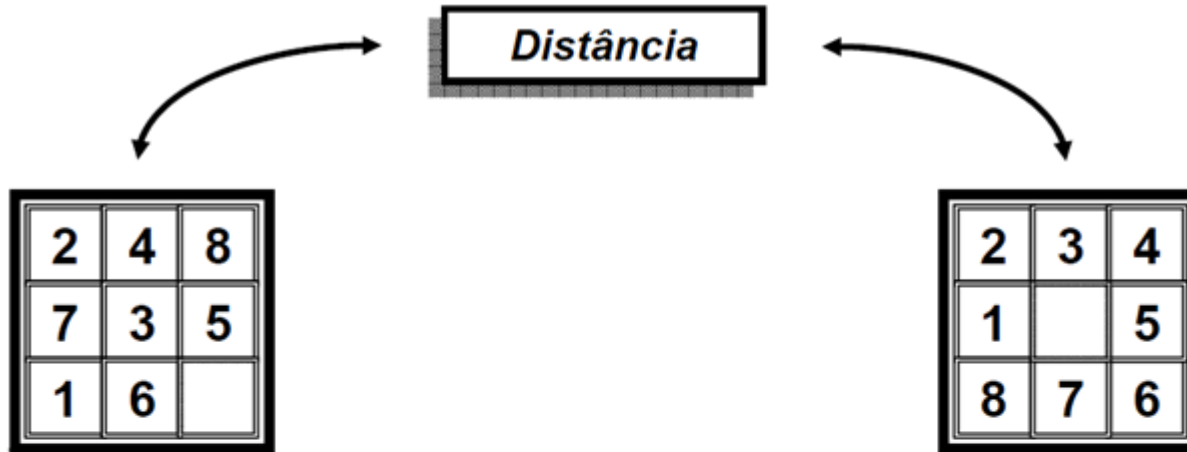
# Definindo Heurísticas



$$h_2\left(\begin{array}{|c|c|c|}\hline 2 & 4 & 8 \\ \hline 7 & 3 & 5 \\ \hline 1 & 6 & \\ \hline\end{array}\right) = ?$$

Outra Heurística?

# Definindo Heurísticas



## Distância de

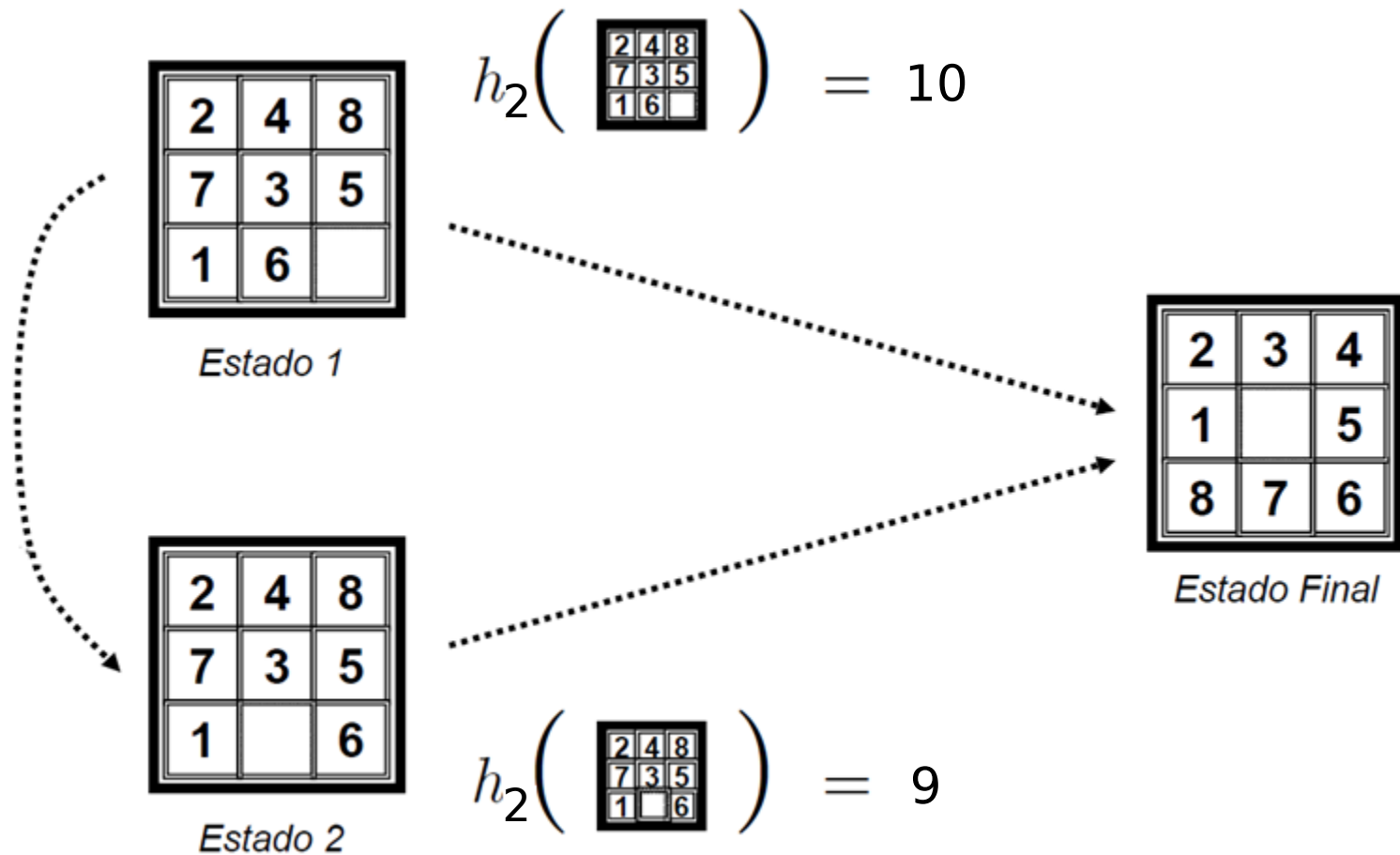
**Manhattan:** distância pombalina, distância de quarteirões ou distância de táxi. Recebeu o nome pois define a menor distância possível que um carro é capaz de percorrer numa malha urbana reticulada ortogonal, como se encontram em zonas como Manhattan ou a Baixa Pombalina.

$$h_2 \left( \begin{array}{|c|c|c|} \hline 2 & 4 & 8 \\ \hline 7 & 3 & 5 \\ \hline 1 & 6 & \\ \hline \end{array} \right) =$$
$$= 10$$

*Número de movimentos necessários para colocar cada peça no seu lugar*

$h_2$  = soma das distâncias de cada elemento à posição final - objetivo (*city block distance* - *Manhattan distance*)

# Definindo Heurísticas



# Definindo Heurísticas

- Como escolher uma boa função heurística para o jogo 8-Puzzle?

- $h_1$  = número de elementos fora do lugar.
- $h_2$  = soma das distâncias de cada número à sua posição final (movimentação horizontal e vertical).

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

- Qual das heurísticas é melhor?

# Definindo Heurísticas

- Função  $h_2(n)$ 
  - o espaço de estados gerado é menor  $\Rightarrow$
  - o algoritmo acha mais rapidamente a solução.

## □ Exemplo:

2	8	3
1	6	4
7		5

Início

$n$  movimentos  $\longrightarrow$

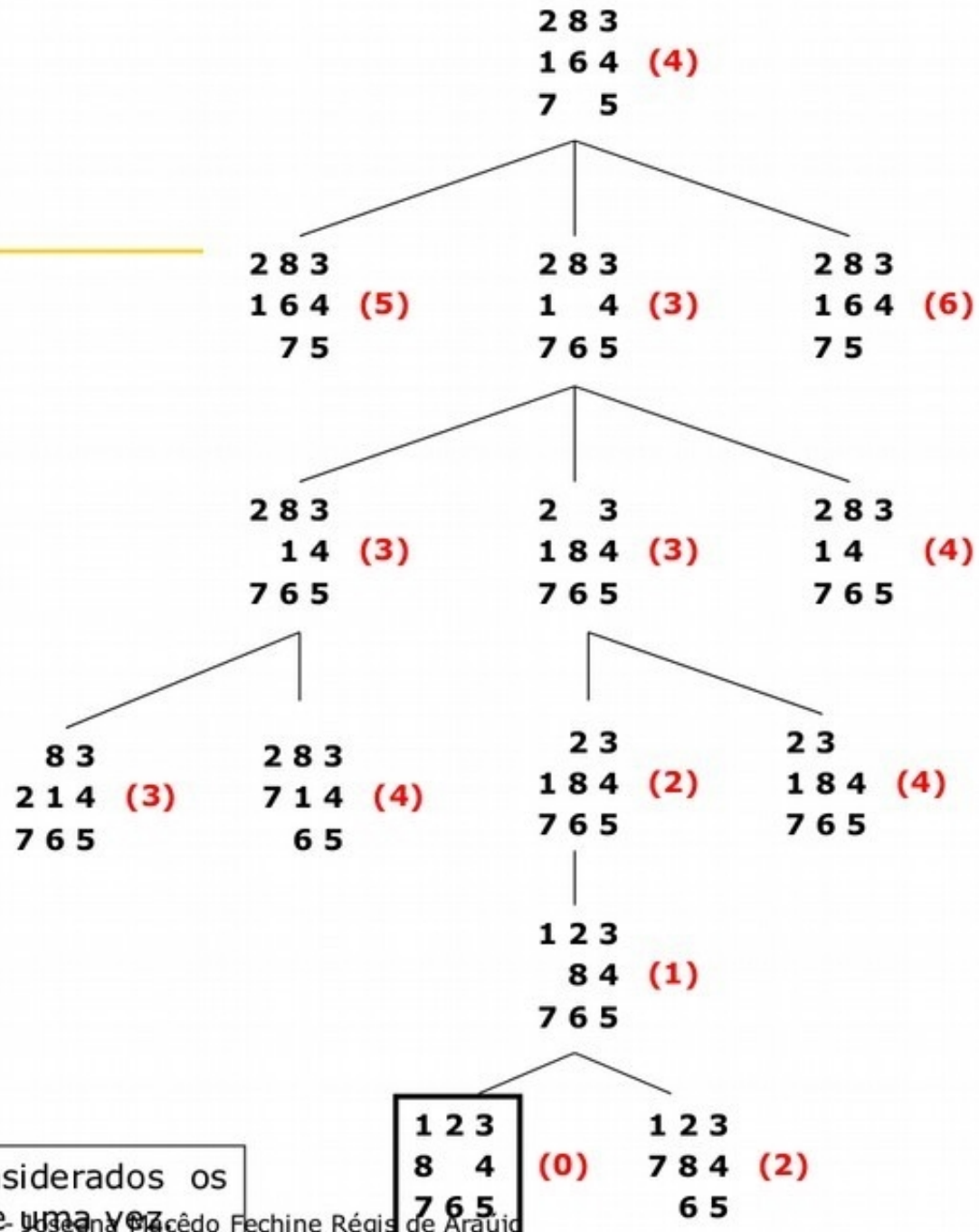
1	2	3
8		4
7	6	5

Objetivo

## Exemplo:

- Espaço de estados gerado com  $h_1(n)$  (para cada estado está indicado entre parênteses o valor da função heurística):

Estado Inicial	Estado Objetivo
2 8 3	1 2 3
1 6 4	8 4
7 5	7 6 5

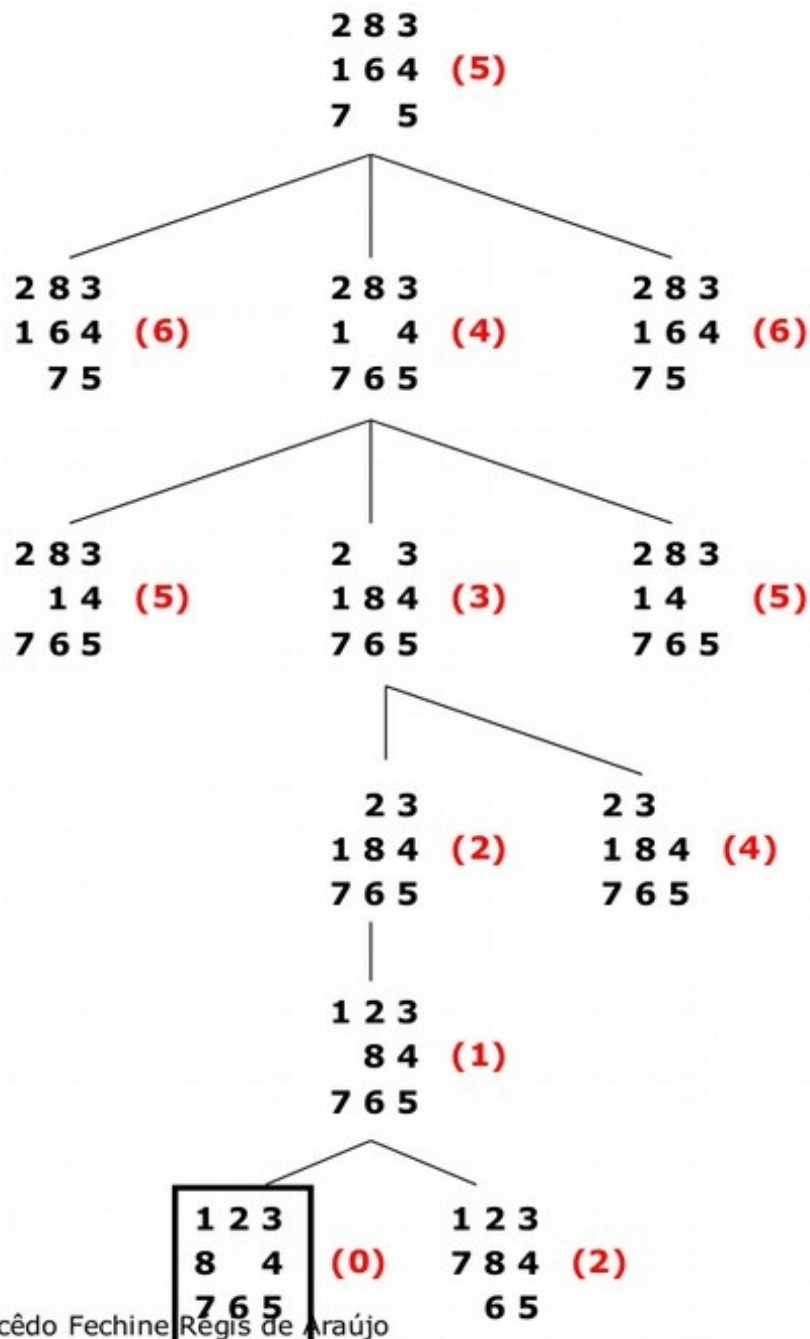


Neste exemplo não são considerados os nós que aparecem por mais de uma vez.

## Exemplo:

- Espaço de estados gerado com  $h_2(n)$

Estado Inicial	Estado Objetivo
2 8 3	1 2 3
1 6 4	8 4
7 5	7 6 5





# Busca Heurística

---

## Efeito da exatidão da heurística sobre o desempenho

- **Qualidade da função heurística:** medida a partir do fator de expansão efetivo ( $b^*$  ou fator de ramificação efetiva).
  - $b^*$  - fator de expansão de uma árvore uniforme com  $N+1$  nós e nível de profundidade  $d$

$$N+1 = 1 + b^* + (b^*)^2 + \dots + (b^*)^d ,$$

*$N$  = total de nós gerados pelo  $A^*$  para um problema;  
 $d$  = profundidade da solução.*

- **Mede-se empiricamente a qualidade de  $h$  a partir do conjunto de valores experimentais de  $N$  e  $d$ .**
  - uma boa função heurística terá o  $b^*$  muito próximo de 1.

# Definindo Heurísticas

▮ Uma boa função heurística  $\rightarrow$   $b^*$  muito próximo de 1.

■ Exemplo:  $h_2$  melhor que  $h_1$

□  $h_i$  domina  $h_k \Rightarrow h_i(n) \geq h_k(n), \forall n$  **no espaço de estados**

■ No exemplo anterior:  $h_2$  domina  $h_1$ ,

■ **Isso pode ser traduzido na forma:** A heurística 2 é melhor que a heurística 1, pois terá um menor fator de ramificação. Desde que  $h_1$  e  $h_2$  não superestimem o custo real.

# Definindo Heurísticas

- ▮ Um problema com a geração de novas funções heurísticas é que muitas vezes não se consegue obter uma única heurística “claramente melhor”.
- ▮ Podemos ter o melhor dos mundos através da definição:


$$h(n) = \max \{h^1(n), \dots, h^m(n)\}.$$

- Essa heurística composta utiliza qualquer função que seja mais precisa no nó em questão.

# Exemplo - A\*

	1	2	3	4	5
1	□		■		X
2			■		
3		■	■		
4					

# Exemplo - A\*

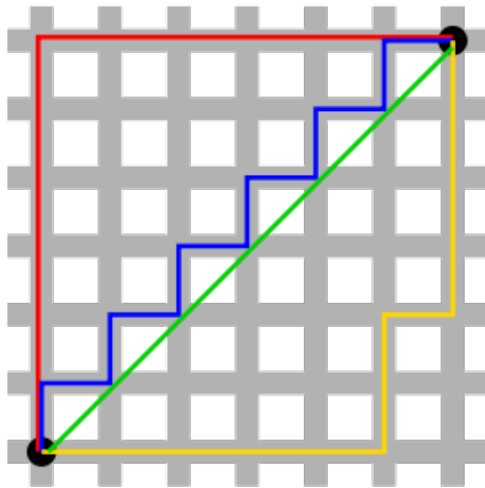
- Qual é o espaço de estados?
  - Quais são as ações possíveis?
  - Qual será o custo das ações?
- 

# Exemplo - A\*

- **Heurística do A\*:**  $f(n) = g(n) + h(n)$ 
  - $g(n)$  = custo do caminho
  - $h(n)$  = função heurística
- Qual seria a função heurística  $h(n)$  mais adequada para este problema?
  - A distância em linha reta é uma opção.

# Exemplo - A\*

- Como calcular a heurística  $h(n)$ ?
  - Distância de Manhattan



$$d_1(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|_1 = \sum_{i=1}^n |p_i - q_i|,$$

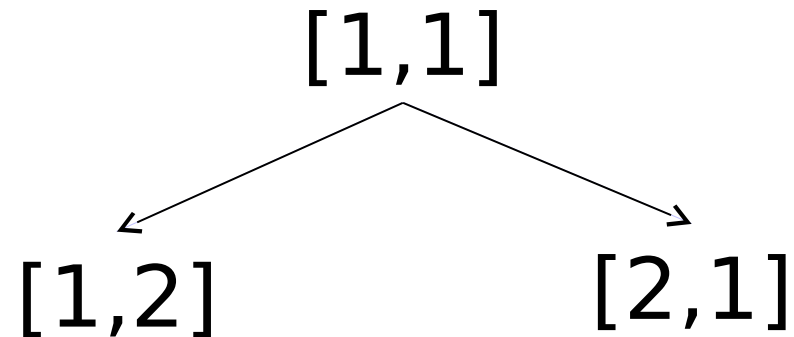
# Exemplo - A\*

- O próximo passo é gerar a árvore de busca e expandir os nós que tiverem o menor valor resultante da função heurística  $f(n)$ .

$$-f(n) = g(n) + h(n)$$



# Exemplo - A\*



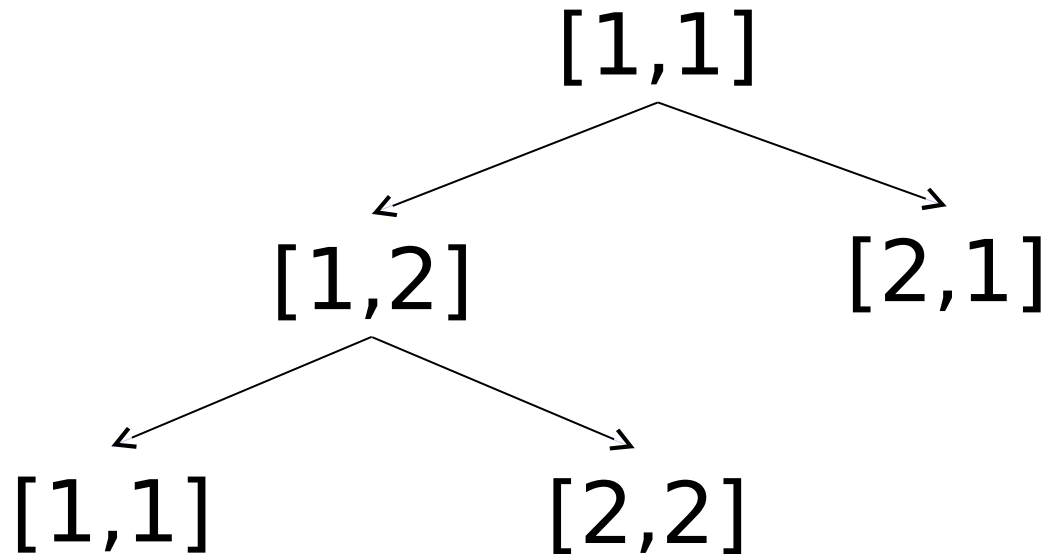
$$[1,2] = f(n) = ?? + ??$$

$$[2,1] = f(n) = ?? + ??$$

# Exemplo - A\*

	1	2	3	4	5
1	□				X
2					
3					
4					

# Exemplo - $A^*$



$$[1,1] = f(n) = ?? + ??$$

$$[2,2] = f(n) = ?? + ??$$

# Exemplo - A\*

	1	2	3	4	5
1		□	■		X
2			■		
3		■	■		
4					

# Leitura Complementar

- Russell, S. and Norvig, P. **Artificial Intelligence: a Modern Approach**, 3rd Edition, Prentice-Hall, 2009.
- **Capítulo 4: Informed Search and Exploration**

## Referência

Edirlei Soares de Lima - Notas de Aula  
Disponível em: <http://edirlei.3dgb.com.br/>

Notas de Aula - Inteligência Artificial - Joseana Macêdo  
Fechine Régis de Araújo  
Disponível em: <http://www.dsc.ufcg.edu.br/~joseana/>

