

# Inteligência Artificial

BCC35G

Diego Bertolini

diegobertolini@utfpr.edu.br

<http://www.inf.ufpr.br/diegob/>

# Aula 004

- Aula Anterior:
  - Resolução de Problemas por Meio de Busca ;
- **Aula de Hoje:**
  - Busca com Informação ;

# Objetivo

**O que vocês devem saber ao final da aula:**

Discutiremos como as informações de estados podem evitar que os algoritmos se percam na escuridão .

# 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.

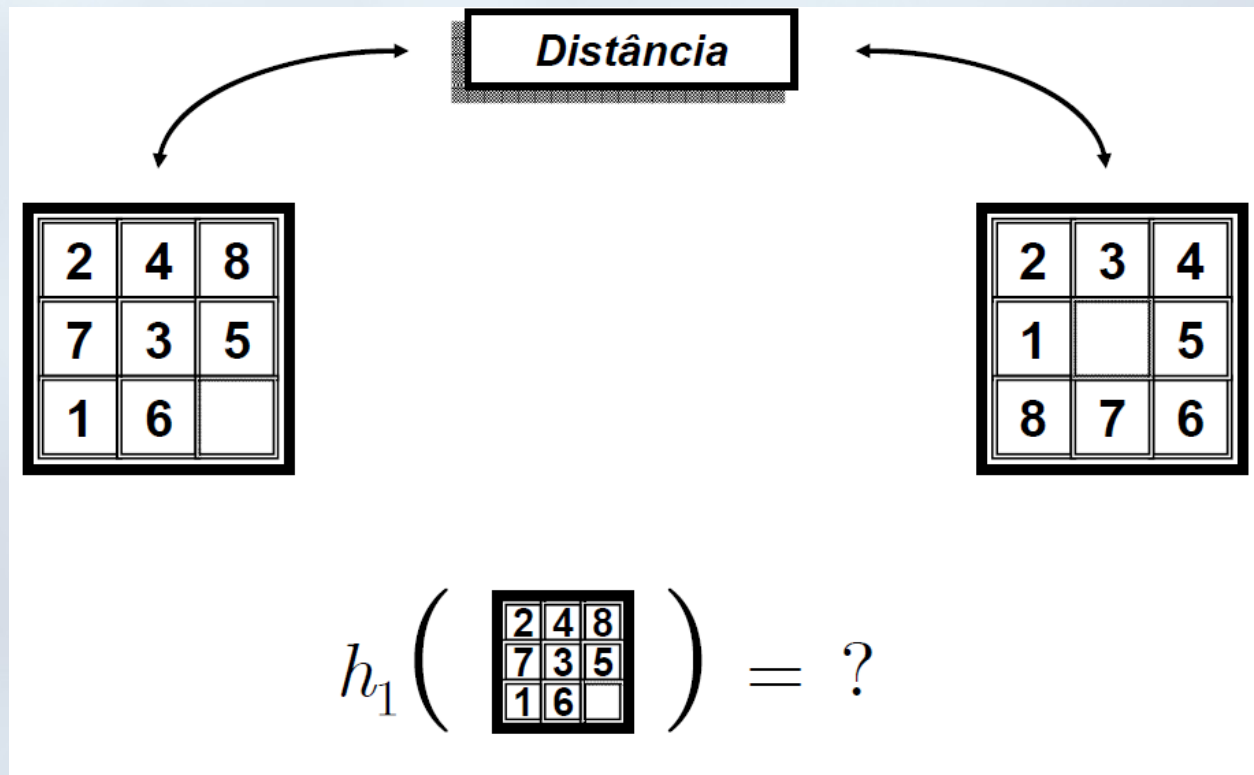
## **Exemplo:**

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



Estado Atual

Estado Objetivo



# Busca Heurística

Algoritmos de Busca Heurística:

Busca Gulosa

A\*

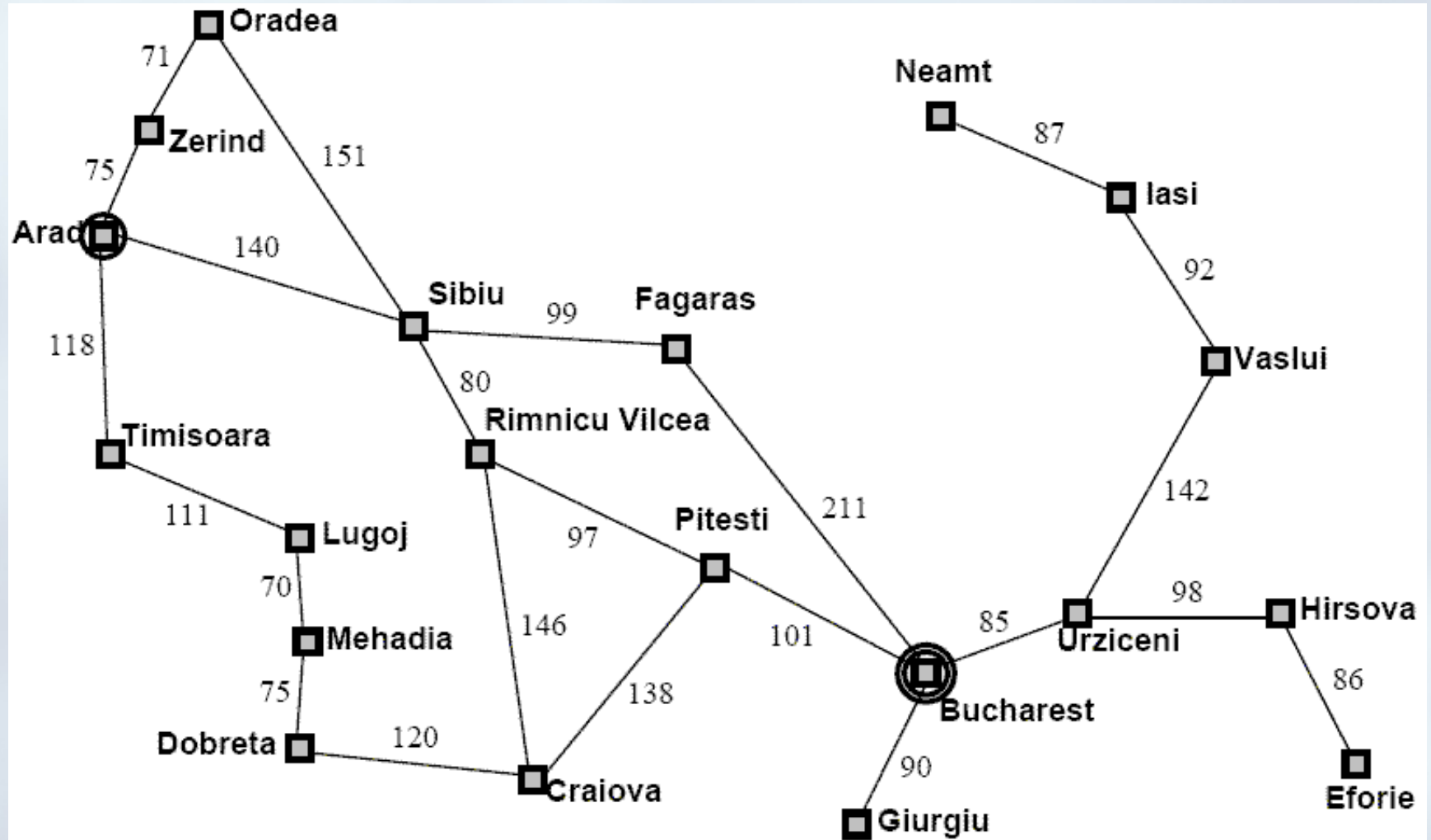
# Busca Gulosa

## Estratégia:

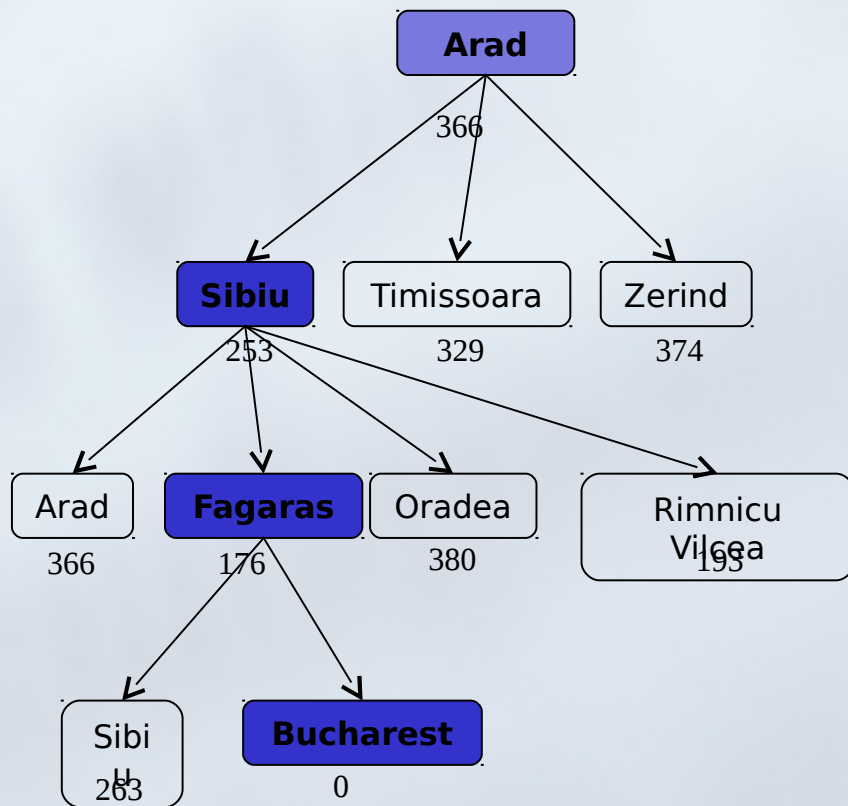
Expande os nós que se encontram mais próximos do objetivo (uma linha reta conectando os dois pontos no caso de distancias), desta maneira é provável que a busca encontre uma solução rapidamente.

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



# Busca Gulosa



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

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.

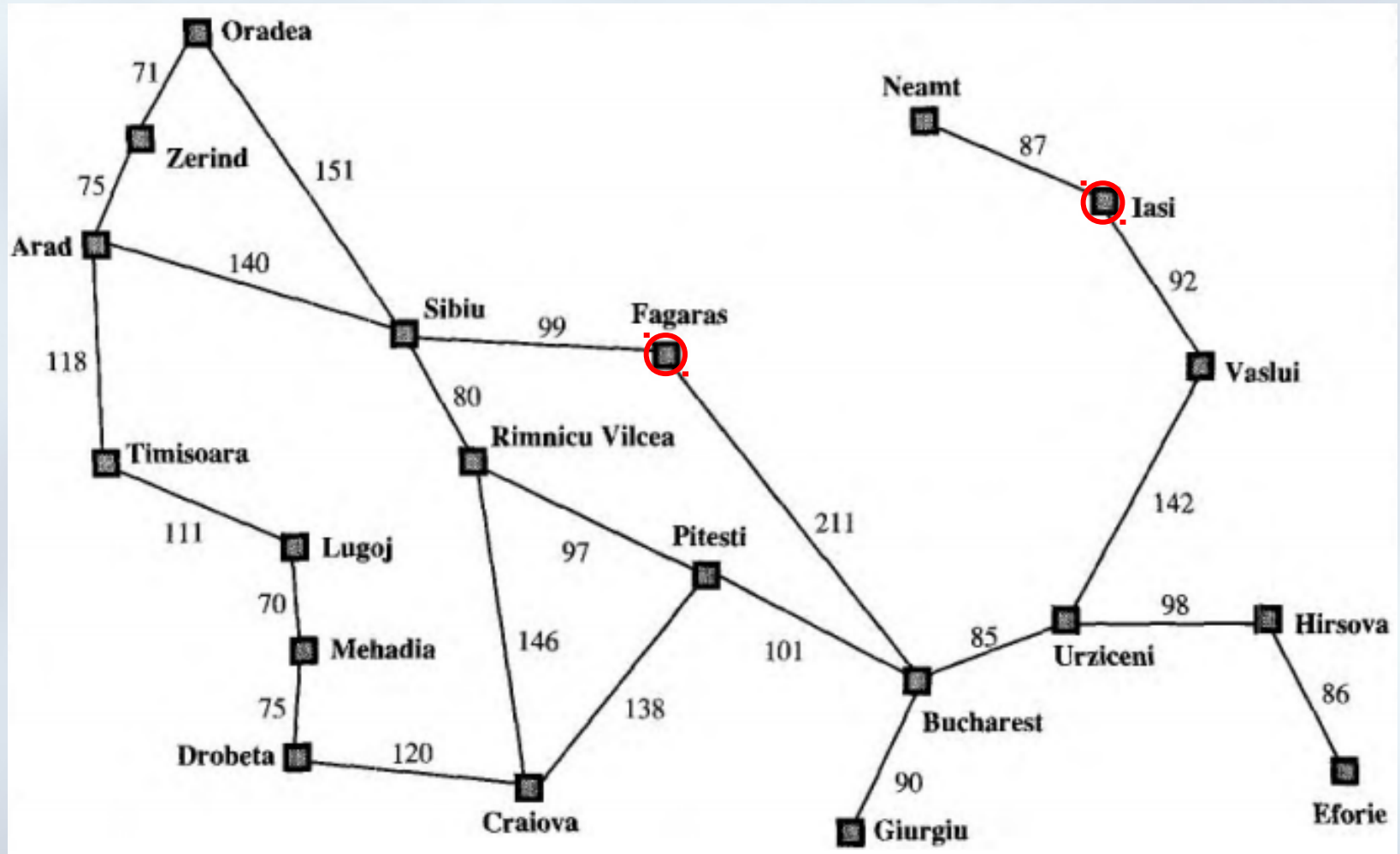
Não é completa:

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  $n$

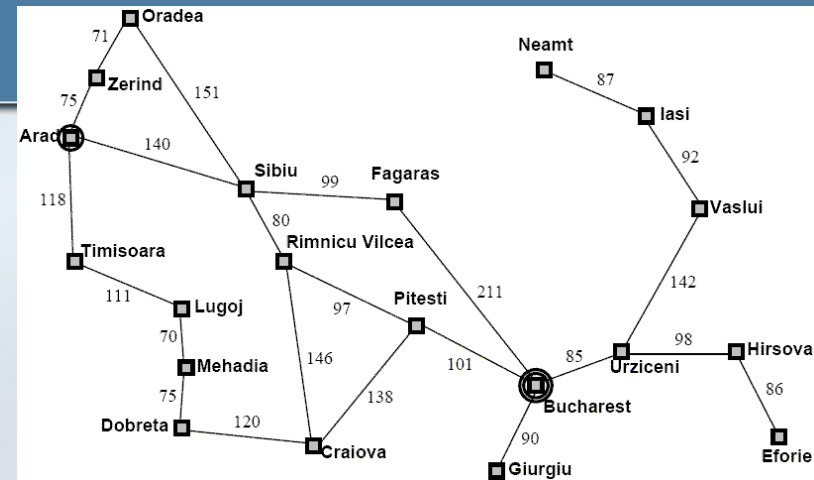
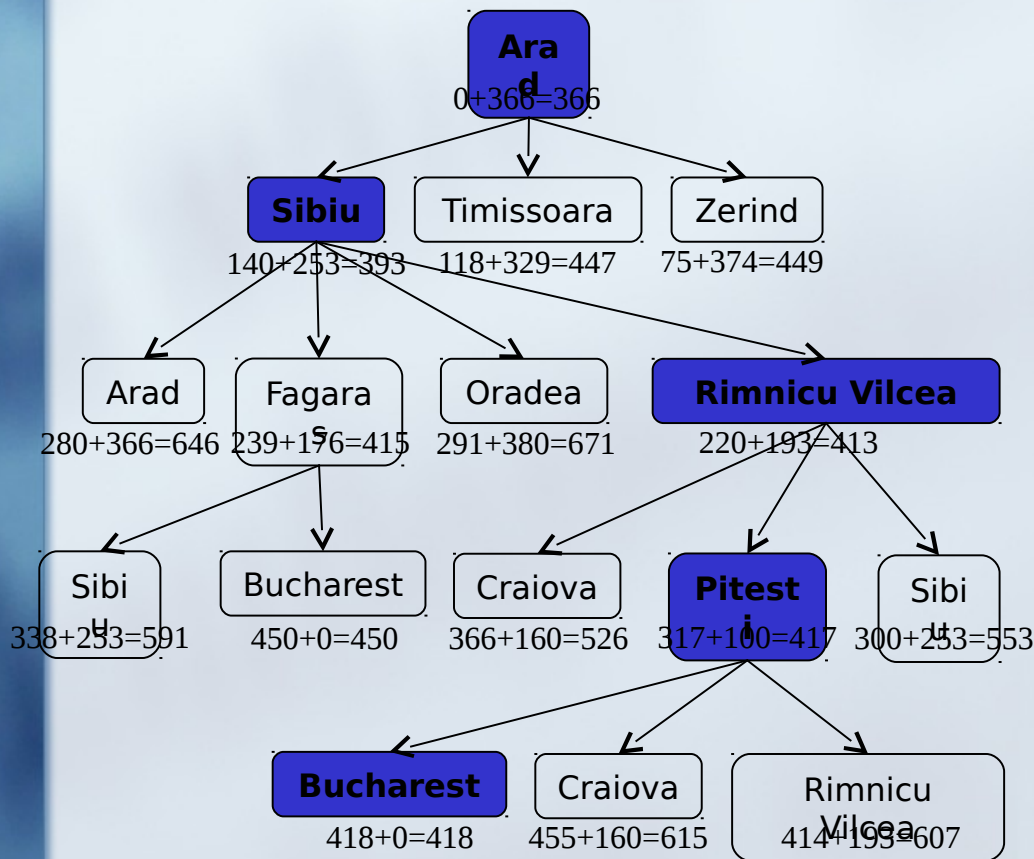
$h(n)$  = valor da heurística do nó  $n$  até um nó objetivo (distancia em linha reta no caso de distancias espaciais)

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

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?

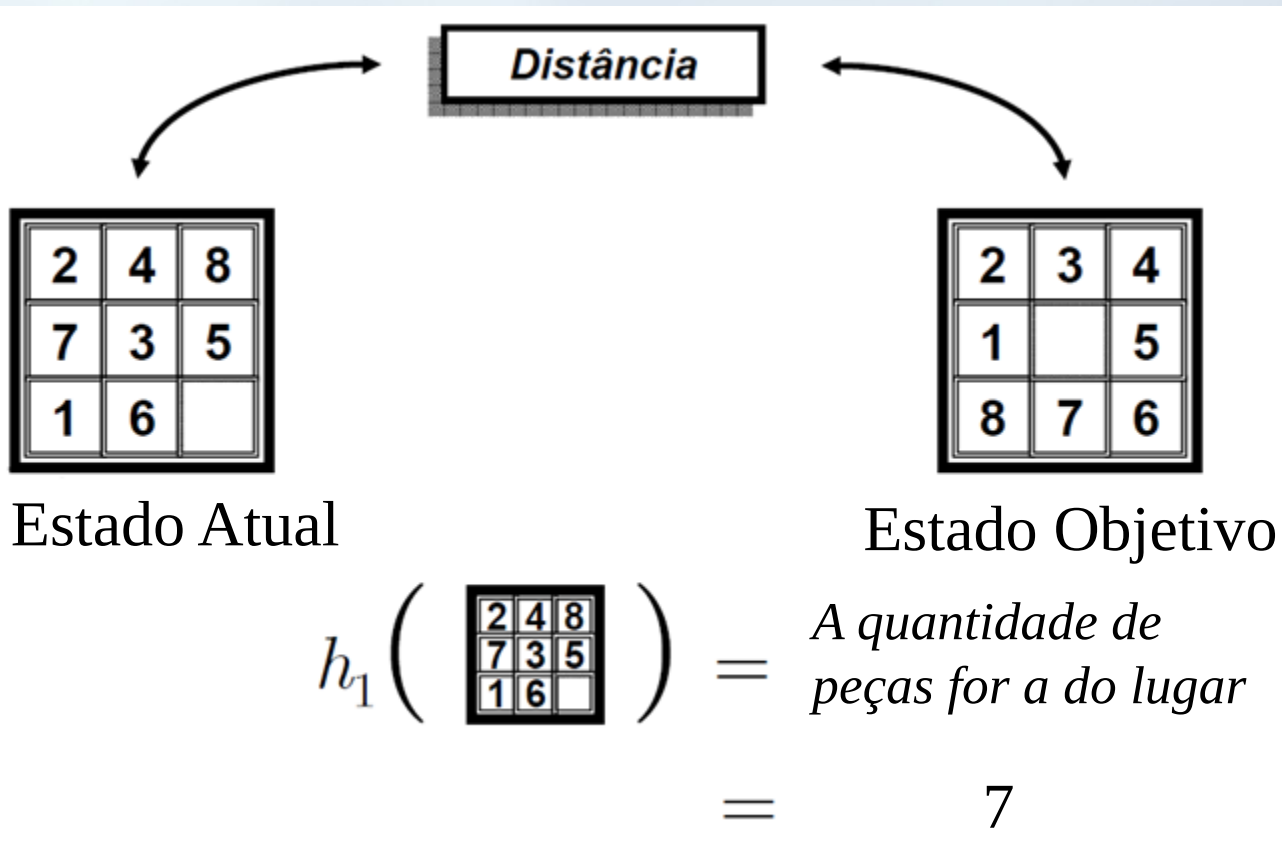
7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

# Definindo Heurísticas



# Definindo Heurísticas

2	4	8
7	3	5
1	6	

*Estado 1*

2	4	8
7	3	5
1		6

*Estado 2*

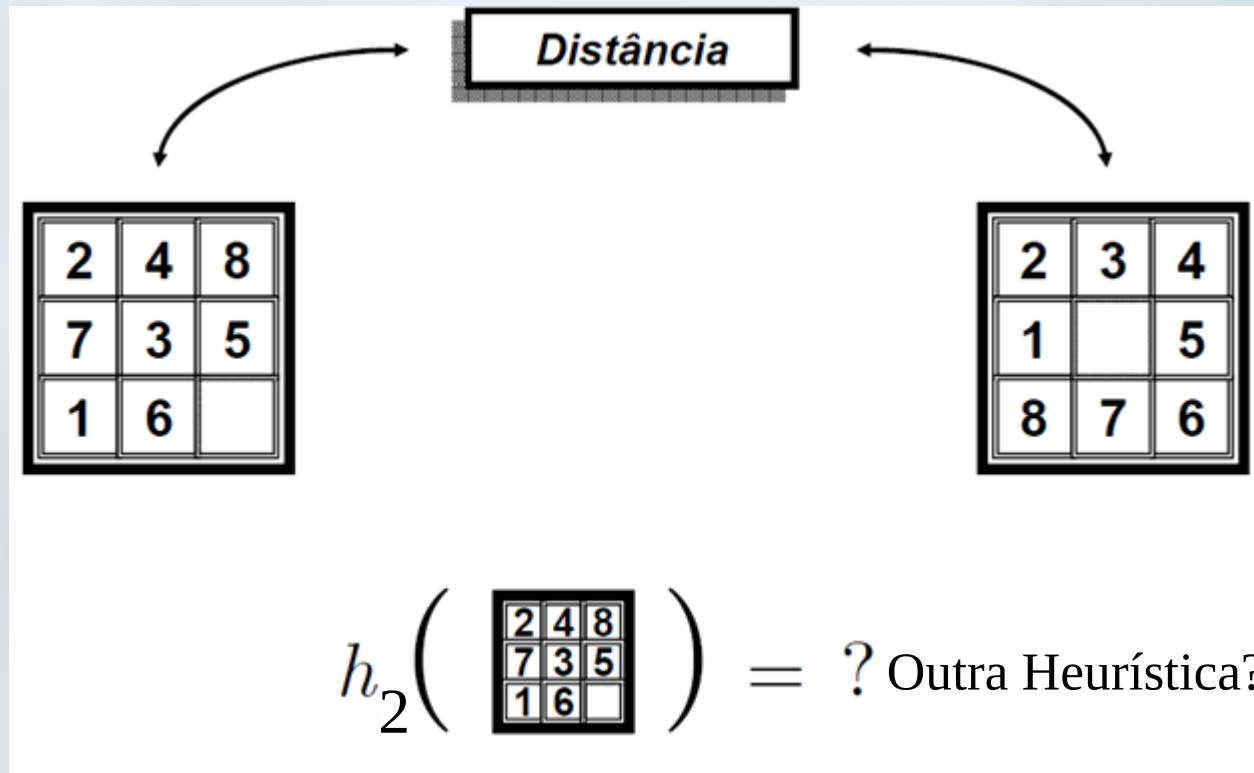
$$h_1 \left( \begin{array}{|c|c|c|} \hline 2 & 4 & 8 \\ \hline 7 & 3 & 5 \\ \hline 1 & 6 & \\ \hline \end{array} \right) = 7$$

2	3	4
1		5
8	7	6

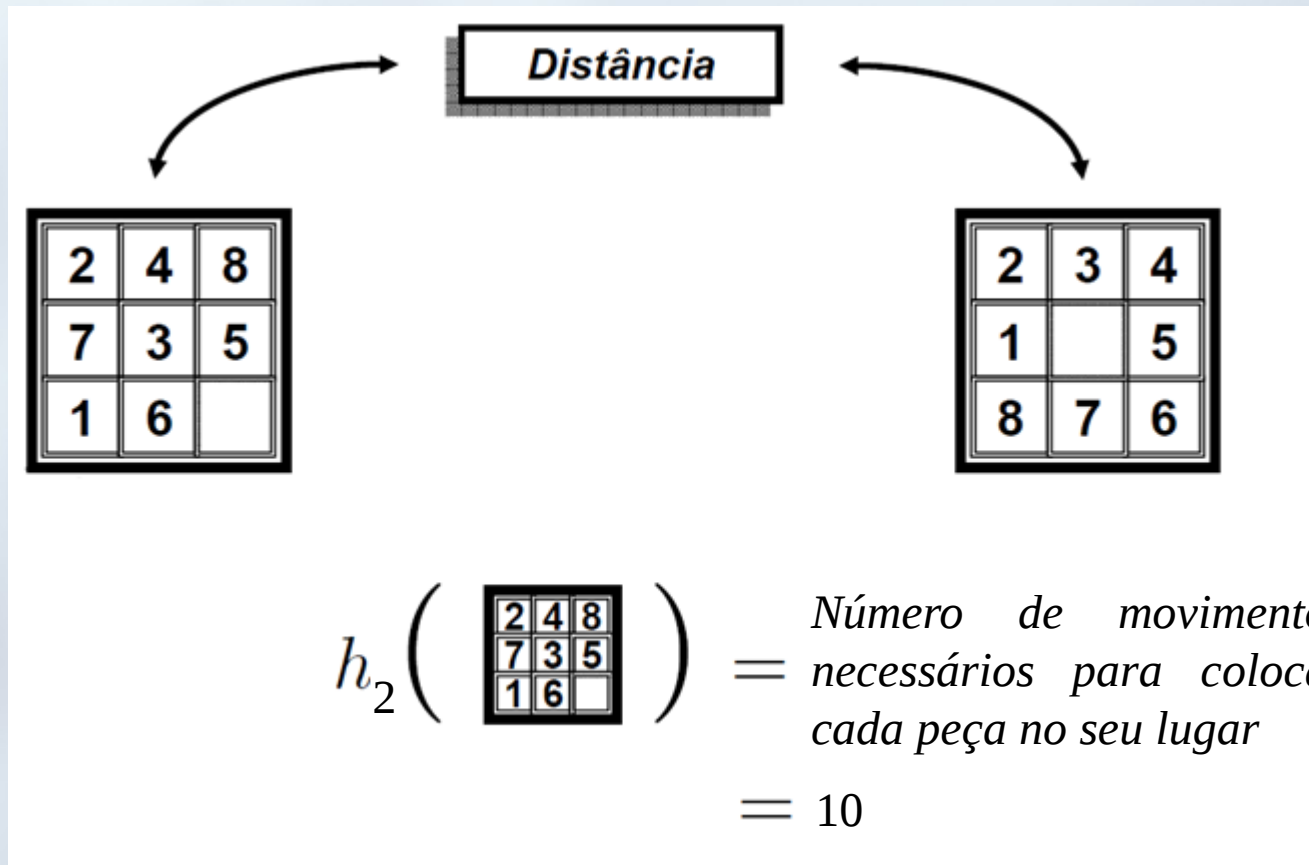
*Estado Final*

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

# Definindo Heurísticas



# Definindo Heurísticas





# Definindo Heurísticas

2	4	8
7	3	5
1	6	

*Estado 1*

2	4	8
7	3	5
1		6

*Estado 2*

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

2	3	4
1		5
8	7	6

*Estado Final*

$$h_2\left(\begin{array}{|c|c|c|}\hline 2 & 4 & 8 \\ \hline 7 & 3 & 5 \\ \hline 1 & & 6 \\ \hline\end{array}\right) = 9$$

# 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).

Qual das heurísticas é melhor?

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

# 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 distancia em linha reta é uma opção.

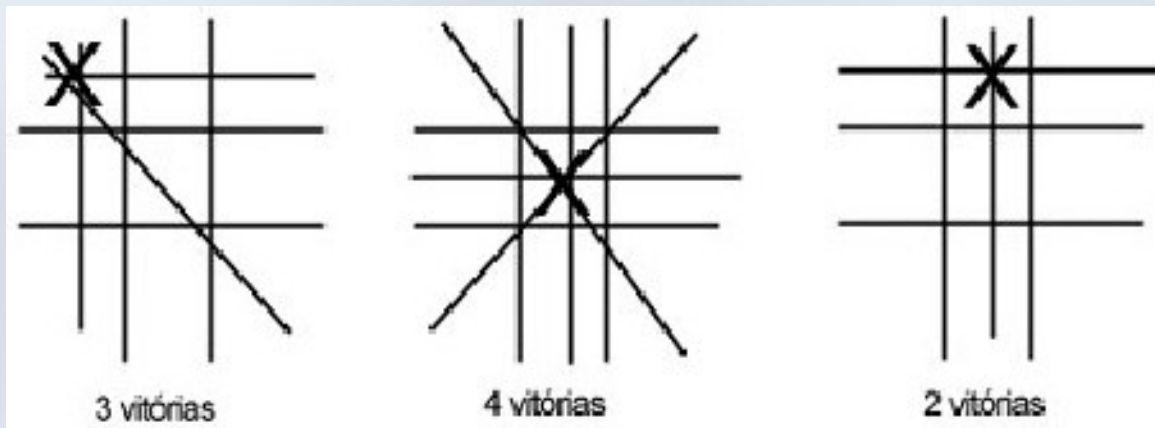
# Exemplo - A\*

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

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

# Exercícios

**(1) Qual seria uma boa heurística para o jogo da velha?**



# Bibliografia e Materiais.

Estes slides foram adaptados do Livro:

**Russell, S. and Norvig, P. Artificial Intelligence: a Modern Approach, 2nd Edition, Prentice-Hall, 2003. Capítulo 3: Informed Search and Exploration**

**Solving Problems by Searching;**

Adaptado das Aulas do Professor: Ederley – PU

