

# Inteligência Artificial

---

## Busca Heurística

Prof. Dr<sup>a</sup>. Andreza Sartori

[asartori@furb.br](mailto:asartori@furb.br)

# Documentos Consultados/Recomendados

- KLEIN, Dan; ABBEEL, Pieter. **Intro to AI**. UC Berkeley. Disponível em: <http://ai.berkeley.edu>.
- LIMA, Edirlei Soares. **Inteligência Artificial**. PUC-Rio, 2015.
- RUSSELL, Stuart J. (Stuart Jonathan); NORVIG, Peter. **Inteligência artificial**. Rio de Janeiro: Campus, 2013. 1021 p, il.
- VIERIU, Radu-Laurențiu. **Artificial Intelligence**. Università degli Studi di Trento, 2016.

# Conteúdo Programático

**Unidade 1:** Fundamentos de Inteligência Artificial

**Unidade 2:** Busca

**Unidade 3:** Sistemas Baseados em Conhecimento

**Unidade 4:** Aprendizado de Máquina e Redes Neurais

**Unidade 5:** Tópicos Especiais



# Conteúdo Programático

**Unidade 1:** Fundamentos de Inteligência Artificial

**Unidade 2:** Busca

**Unidade 3:** Sistemas Baseados em Conhecimento

**Unidade 4:** Aprendizado de Máquina e Redes Neurais

**Unidade 5:** Tópicos Especiais



# Conteúdo Programático

**Unidade 1:** Fundamentos de Inteligência Artificial

**Unidade 2:** Busca

2.1. Resolução de Problemas por meio de busca

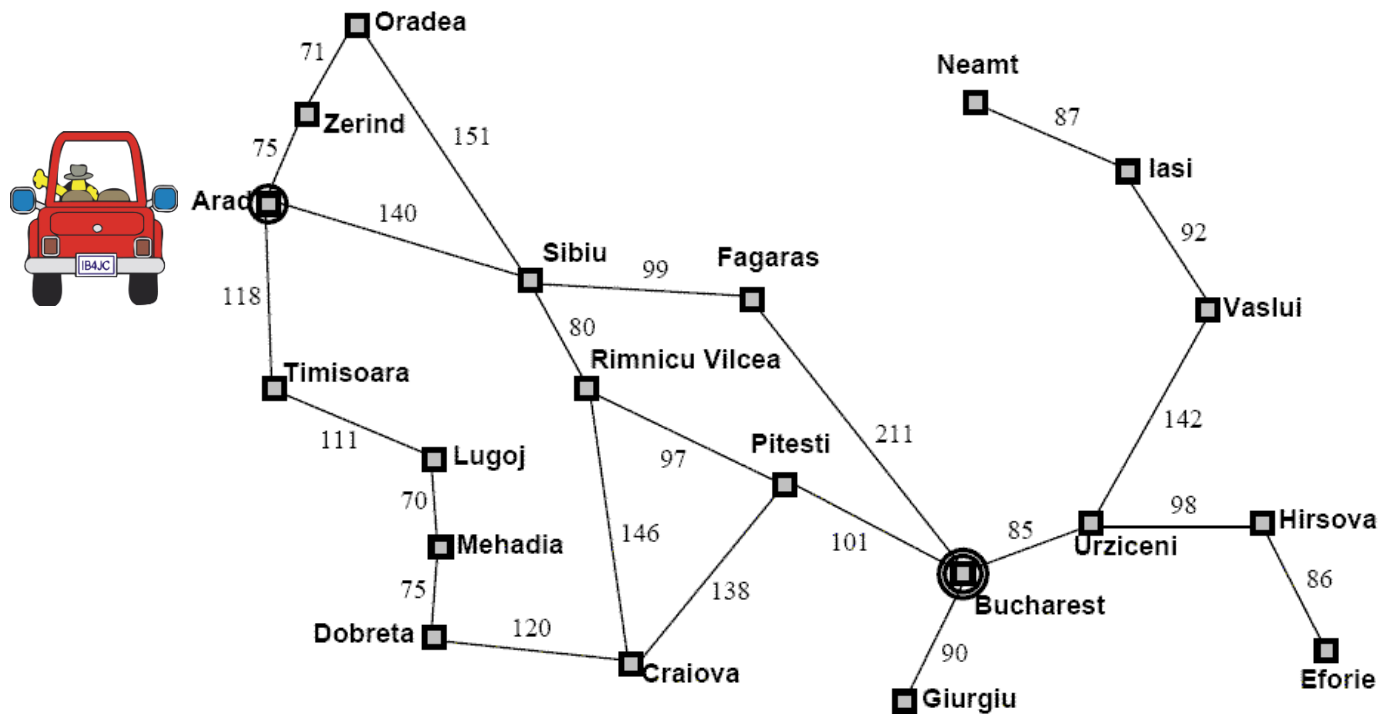
2.2. Busca Cega ou Exaustiva

**2.3. Busca Heurística**

2.4. Algoritmos Genéticos



# Relembrando: Problema de Busca



# Relembrando: Definição de um Problema

- **Estado Inicial:** Estado inicial do agente.
  - Ex: Em(Arad)
- **Estado Objetivo (Estado Final):** Estado buscado pelo agente.
  - Ex: Em(Bucharest)
- **Ações Possíveis (Função Sucessor):** Conjunto de ações que o agente pode executar.
  - Ex: Ir(Cidade, PróximaCidade)
- **Espaço de Estados:** Conjunto de estados que podem ser atingidos a partir do estado inicial.
  - Ex: Mapa da Romênia.
- **Custo de Caminho:** Custo numérico de cada caminho.
  - Ex: Distância em KM entre as cidades.

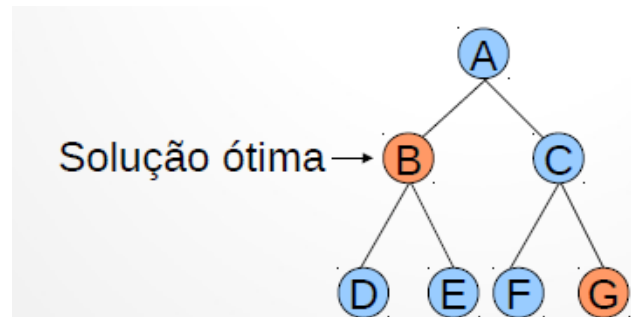
# Relembrando: Solução para um Problema

- A **solução** para um problema é um caminho desde o estado inicial até o estado objetivo (estado final).
- A qualidade da solução é medida pela função de custo de caminho, isto é, a **solução que tiver menor custo** de caminho entre todas as soluções.



# Medida de Desempenho do Algoritmo de Busca

- Uma estratégia de busca é definida pela escolha da **ordem da expansão de nós**
- Estratégias são avaliadas de acordo com os seguintes critérios:
  - **Completeza:** o algoritmo sempre encontra a solução se ela existe?
  - **Otimização (Custo de Caminho):** a estratégia encontra a solução ótima? - Qualidade da solução
    - Para passos com igual custo, é aquela em menor profundidade na árvore de busca



# Medida de Desempenho do Algoritmo de Busca

- Uma estratégia de busca é definida pela escolha da **ordem da expansão de nós**
- Estratégias são avaliadas de acordo com os seguintes critérios:
  - **Complexidade De Tempo (Custo de Busca):** quanto tempo ele leva para encontrar a solução? - Número de nós gerados
  - **Complexidade De Espaço (Custo de Busca):** quanta memória é necessária para executar a busca? - Número máximo de nós na memória.

## Custo Total

Custo do Caminho + Custo de Busca.

# Métodos de Busca

- **Busca Cega ou Exaustiva: (Sem informação)**

- Não tem nenhuma informação adicional sobre os estados, isto é, não sabe qual o melhor nó da fronteira a ser expandido. Apenas distingue o estado objetivo dos não objetivos.

- **Busca Heurística:**

- Ou busca com informação, estima qual o melhor nó da fronteira a ser expandido baseado em funções heurísticas.

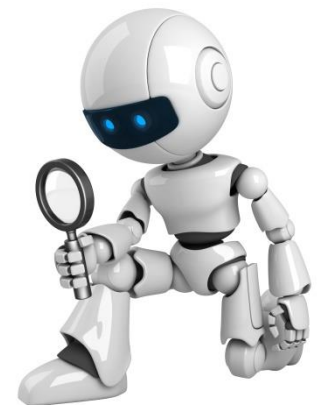
- **Algoritmos Genéticos:**

- Variante de Busca Local em que é mantida uma grande população de estados. Novos estados são gerados por mutação e por crossover, que combina pares de estados da população.

# Algoritmos de Busca Cega ou Exaustiva

As estratégias de busca sem informação se distinguem pela **ordem** em que os nós são expandidos.

1. Busca em extensão/largura;
2. Busca em profundidade;
3. Busca por aprofundamento iterativo;
4. Busca de custo uniforme.



# Métodos de Busca

- **Busca Cega ou Exaustiva: (Sem informação)**

- Não tem nenhuma informação adicional sobre os estados, isto é, não sabe qual o melhor nó da fronteira a ser expandido. Apenas distingue o estado objetivo dos não objetivos.

- **Busca Heurística:**

- Ou busca com informação, estima qual o melhor nó da fronteira a ser expandido baseado em funções heurísticas.

- **Algoritmos Genéticos:**

- Variante de Busca Local em que é mantida uma grande população de estados. Novos estados são gerados por mutação e por crossover, que combina pares de estados da população.

# Busca Heurística

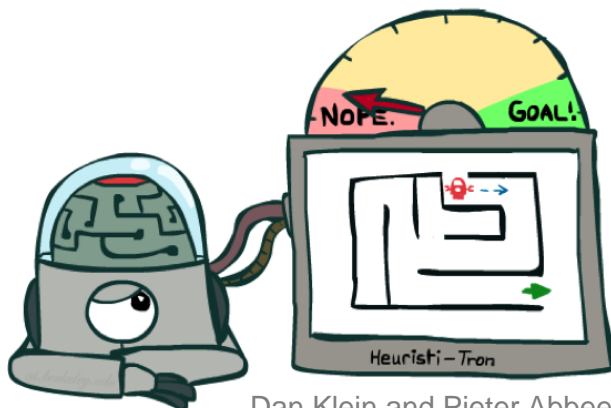
- Busca pela melhor escolha
  - Escolhe o nó que parece ser o melhor de acordo com a função de avaliação.
    - Utiliza uma função de avaliação para cada nó.
    - Expande o nó que tem a função de avaliação mais baixa.
    - Dependendo da função de avaliação, a estratégia de busca muda.
- Utiliza conhecimento específico sobre o problema para encontrar soluções.
  - Leva em conta o objetivo para decidir qual caminho escolher.

# Busca Cega X Busca Heurística

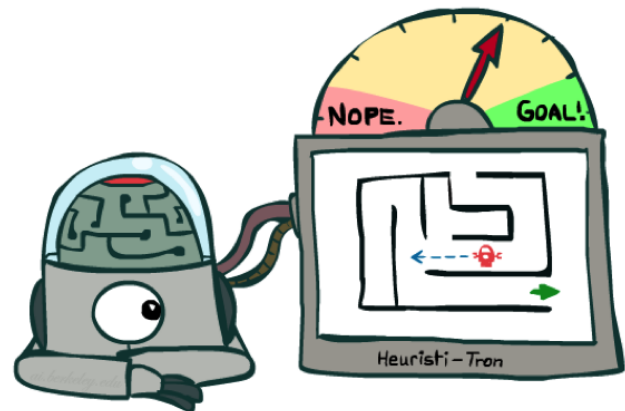
- Como encontrar um barco perdido?
  - **Busca Cega** -> Procura no oceano inteiro.
  - **Busca Heurística** -> Procura utilizando informações relativas ao problema.
    - Exemplo: correntes marítimas, vento, etc.

# Função Heurística - $h(n)$

- $h(n)$  = Custo estimado do caminho mais econômico do estado atual ( $n$ ) até o estado objetivo.
  - Retorna um número de um estado;
  - Mostra se está ou não próximo de um objetivo.
- Cada problema **exige** uma função heurística diferente.



Dan Klein and Pieter Abbeel  
ai.berkeley.edu



Dan Klein and Pieter Abbeel  
ai.berkeley.edu



# Busca Heurística

- Algoritmos de Busca Heurística

- Busca Gulosa

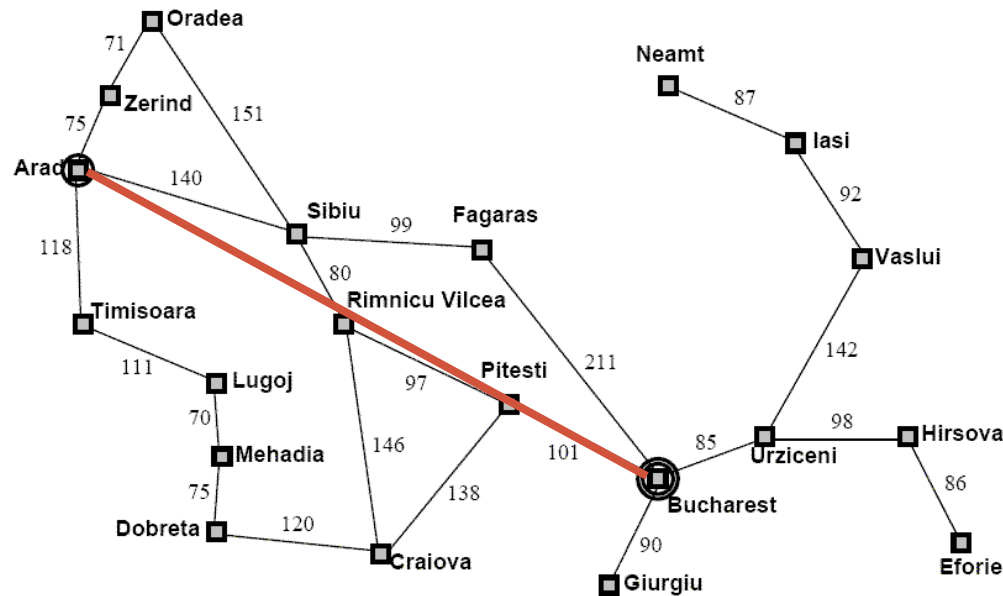


- $A^*$



# Busca Gulosa

- Expande os nós que estão mais próximos do objetivo, na suposição de que a busca provavelmente encontre uma solução rápida.
  - Localização de rotas da Romênia: utiliza-se a heurística de distância em linha reta conectando os dois pontos.



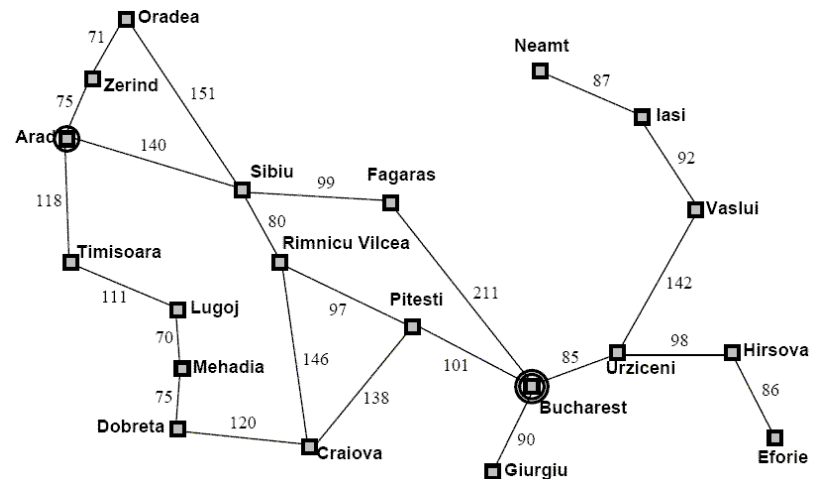
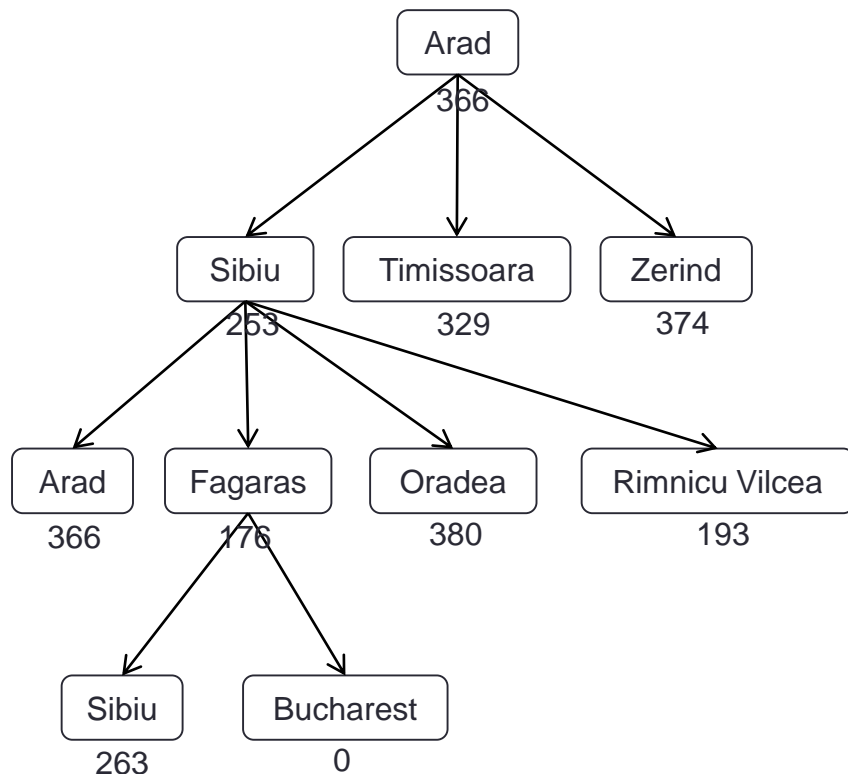
# Busca Gulosa

- Expande os nós que estão mais próximos do objetivo, na suposição de que a busca provavelmente encontre uma solução rápida.
  - Localização de rotas da Romênia: utiliza-se a heurística de distância em linha reta conectando os dois pontos.
- A implementação do algoritmo é semelhante a busca em profundidade, porém **utiliza uma função heurística** para decidir qual nó deve ser expandido.
- Avalia os nós usando apenas a função heurística:

$$f(n) = h(n)$$



# Busca Gulosa (Edirlei Soares de Lima - PUC)



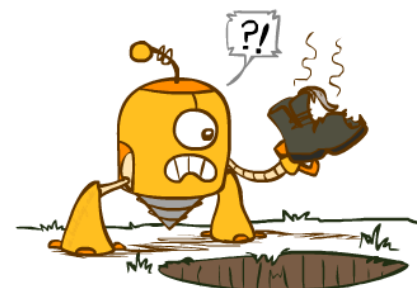
Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80

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

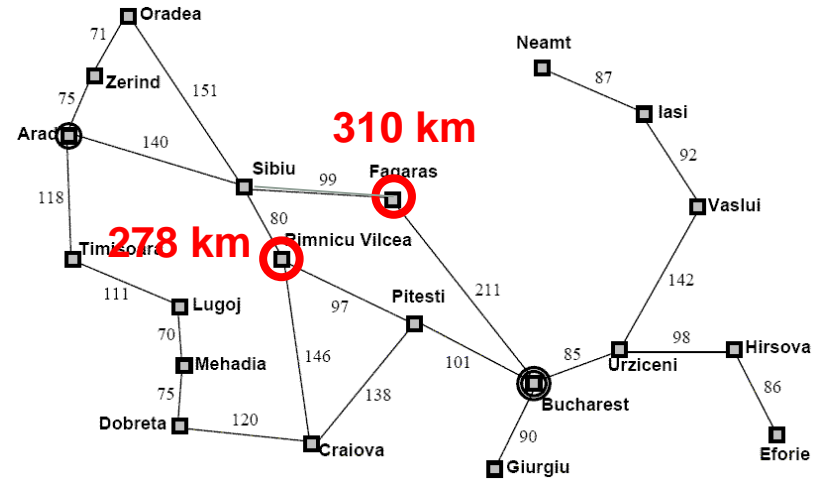
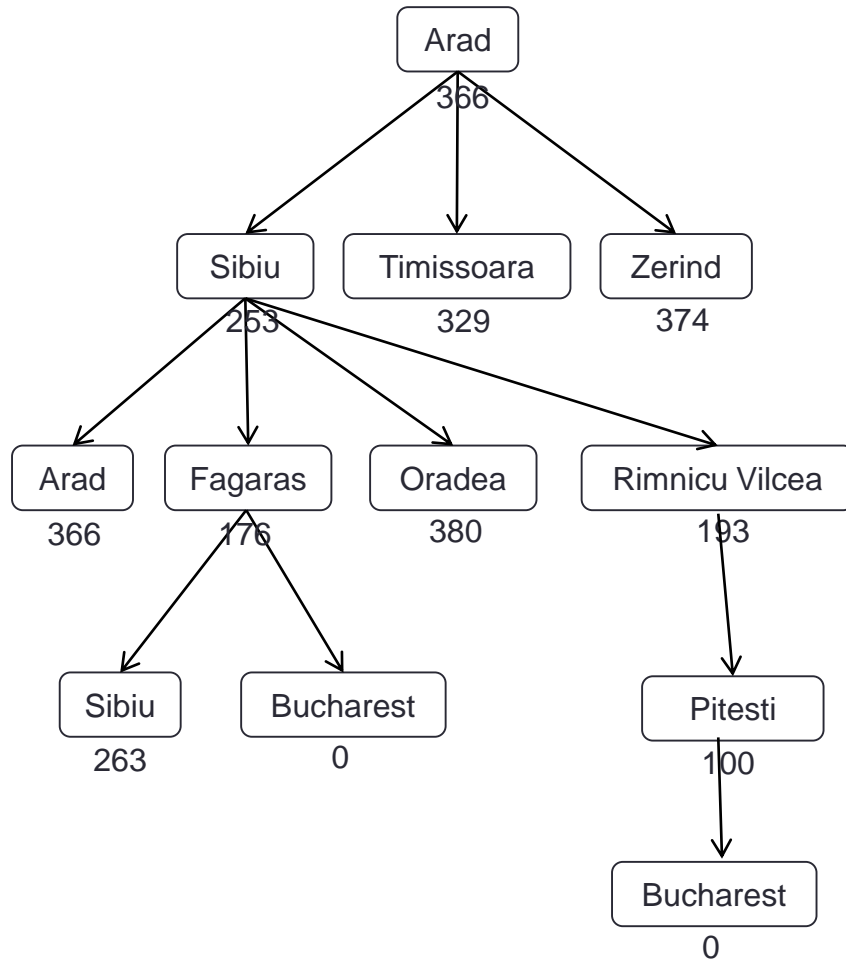
# Busca Gulosa

- **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.



# Busca Gulosa (Edirlei Soares de Lima - PUC)



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

- **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.

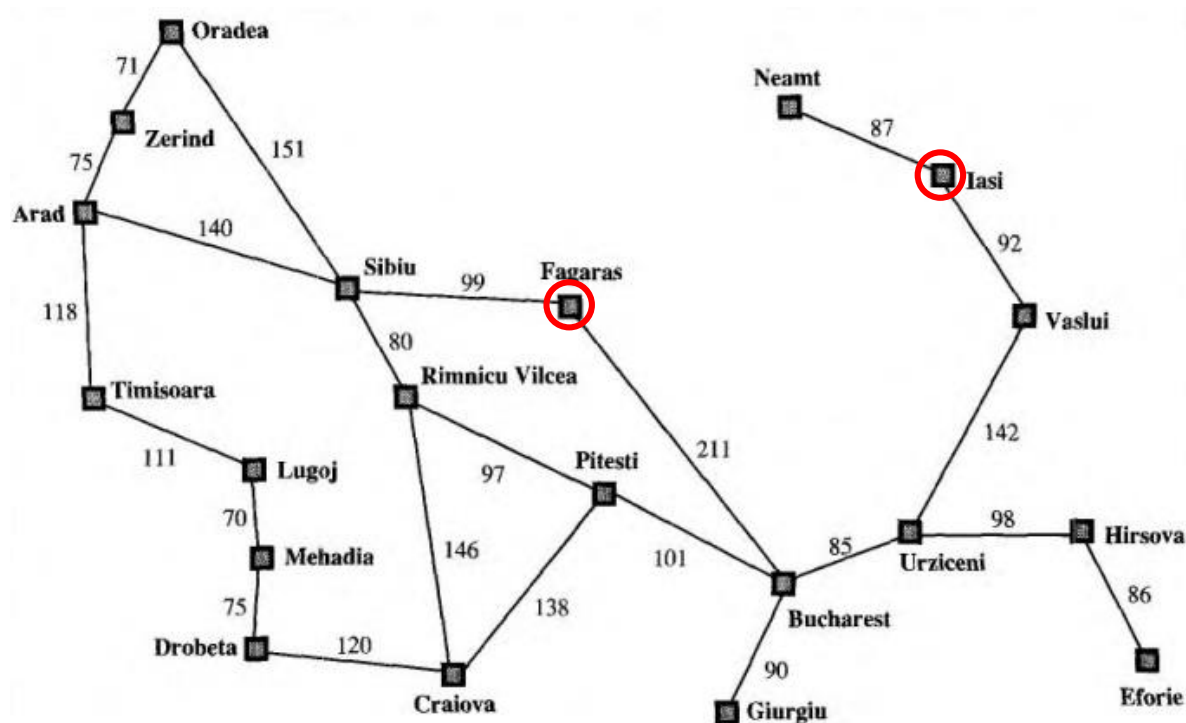
- **Não é completa:**

- Pode entrar em loop se não detectar a expansão de estados repetidos.
- Pode entrar em um caminho infinito e nunca retornar para experimentar outras possibilidades.



# Busca Gulosa

- Minimizar  $h(n)$  é uma ação suscetível a falsos inícios.
  - Ex. Ir de Iasi a Fagaras
    - Heurística sugerirá ir a Neamt, que é um beco sem saída.
    - Provoca a extensão de nós desnecessários.





# Busca Heurística

- Algoritmos de Busca Heurística

- Busca Gulosa



- $A^*$



# Busca Heurística



**Busca de Custo Uniforme**



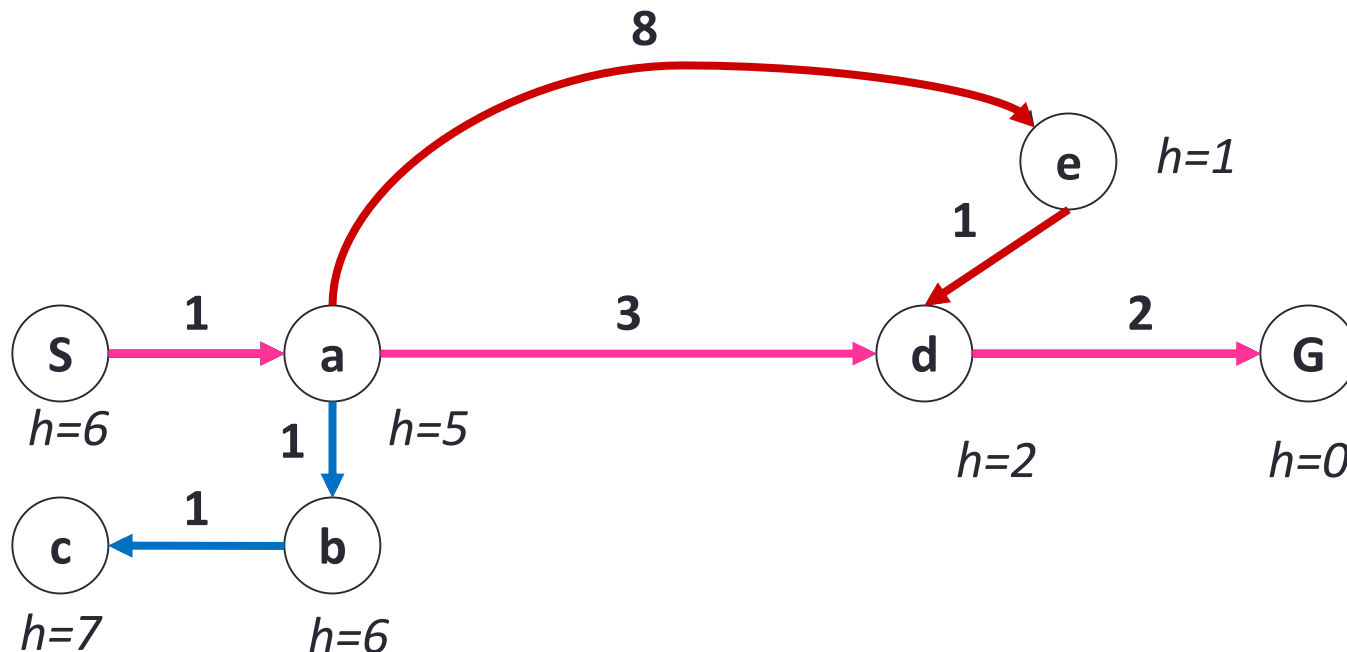
**Busca Gulosa**



**Busca A\***

# Busca A\*

- **Busca de custo uniforme:** expande por custo de caminho  $\rightarrow g(n)$
- **Busca Gulosa:** expande por proximidade do objetivo  $\rightarrow h(n)$



- **Busca A\*:** expande para o nó com menor valor de  $g(n) + h(n)$

# Busca A\*

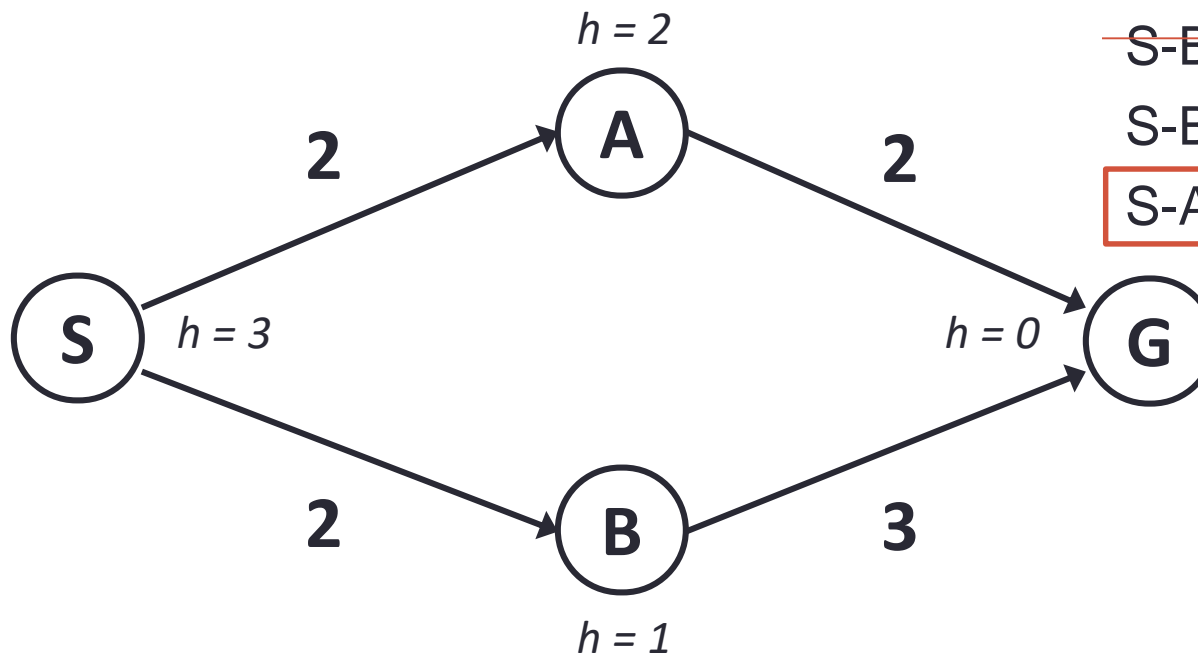
- Combina o **custo do caminho**  $g(n)$  com o **valor da heurística**  $h(n)$
- $g(n)$  = custo do caminho do nó inicial até o nó  $n$
- $h(n)$  = valor da função heurística do nó  $n$  até um nó objetivo

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

- Considera os nós que não foram expandidos ainda e expande o que tem o menor custo  **$f(n)$**

# Busca A\*

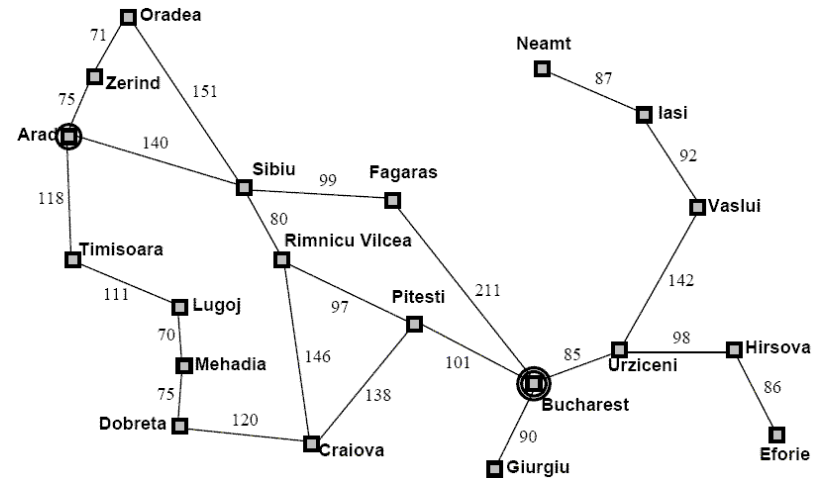
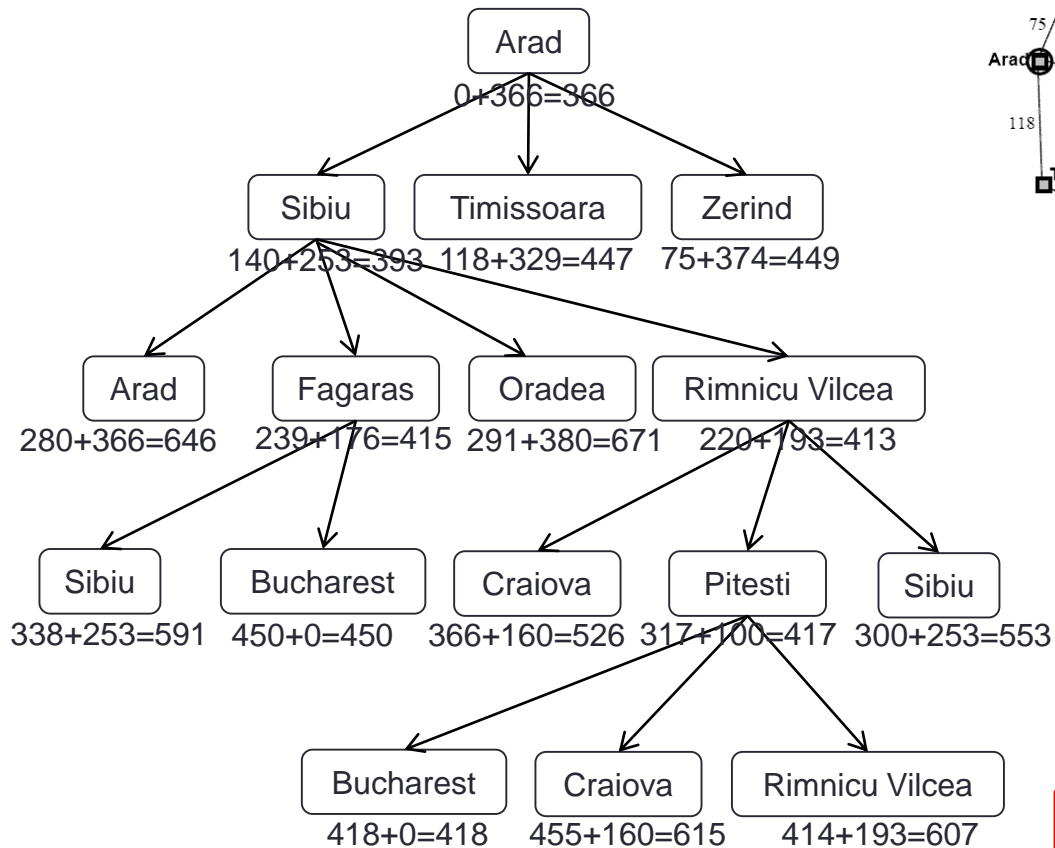
- Quando A\* deve parar?



	g	h	f
<del>S</del>	<del>0</del>	<del>3</del>	<del>3</del>
S-A	2	2	4
<del>S-B</del>	<del>2</del>	<del>1</del>	<del>3</del>
S-B-G	5	0	5
S-A-G	4	0	4

- O algoritmo expande todos os nós com  $f(n) \leq C^*$  (Custo da solução ótima).
  - Um algoritmo que não expande todos os nós com  $f(n) < C^*$  corre o risco de omitir a solução ótima.

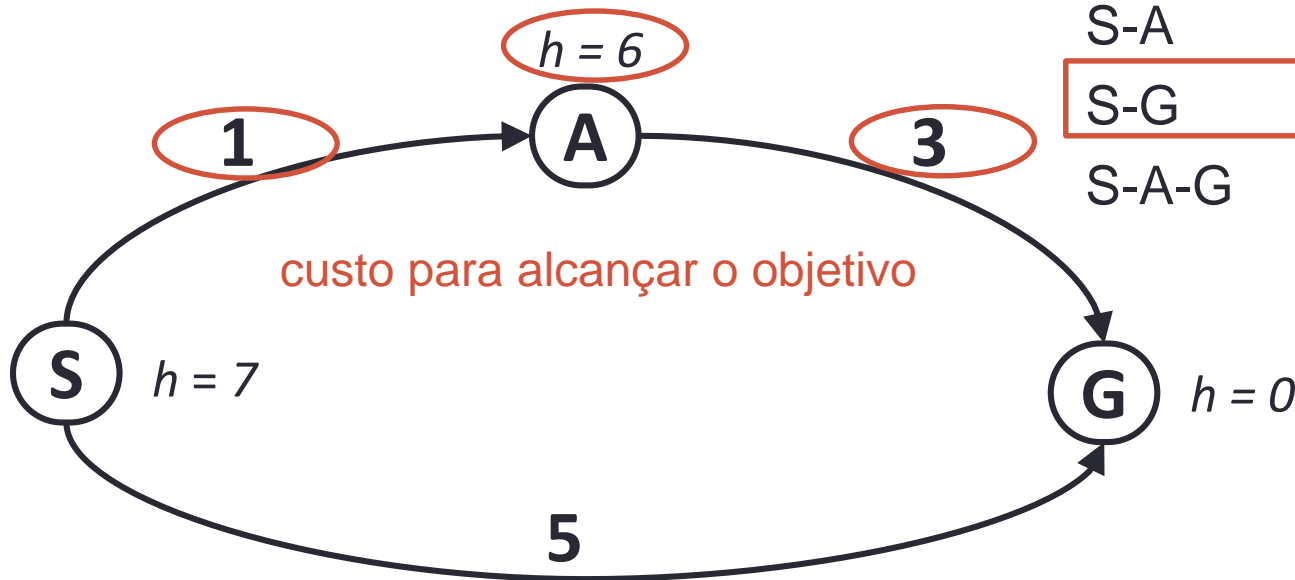
# Busca A\* (Edirlei Soares de Lima - PUC)



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**.
  - Desde que possamos garantir que  $h(n)$  é **admissível** ou **consistente**, onde o custo heurístico nunca superestime o custo para alcançar o objetivo.

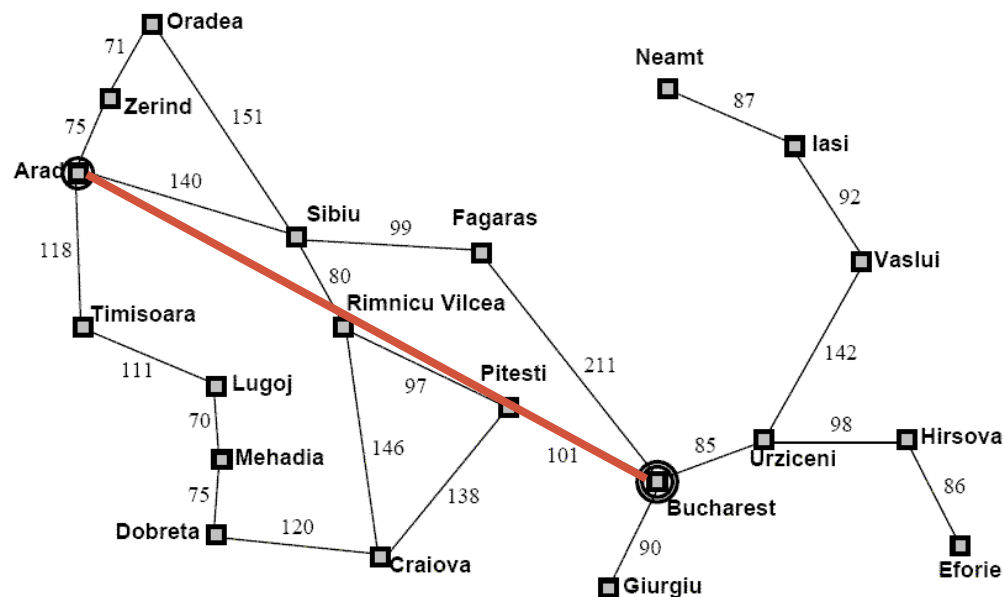


	g	h	f
S	0	7	7
S-A	1	6	7
S-G	5	0	5
S-A-G	4	0	4

# Responda Rápido!

No exemplo da localização de rotas da Romênia: utiliza-se a heurística de distância em linha reta, conectando os dois pontos. Este exemplo é admissível?

- Sim. Porque o caminho mais curto entre dois pontos quaisquer é uma linha reta, assim a linha reta não pode ser superestimativa.





# Busca A\*

- A estratégia é **completa e ótima**.
  - Desde que possamos garantir que  $h(n)$  é admissível ou consistente, onde o custo heurístico nunca superestime o custo para alcançar o objetivo.
- **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.

# Busca Heurística

<https://qiao.github.io/PathFinding.js/visual/>

The screenshot displays the PathFinding.js visualizer interface. The main area is a grid with a green square at (30, 60) representing the start, a grey vertical bar at (45, 60) representing an obstacle, and an orange square at (55, 60) representing the goal. The sidebar on the right contains the following elements:

- Select Algorithm**
  - › A\*
  - › IDA\*
  - ▼ **Breadth-First-Search**
    - Options
      - ☒ Allow Diagonal
      - ☐ Bi-directional
      - ☐ Don't Cross Corners
  - › Best-First-Search
  - › Dijkstra
  - › Jump Point Search
  - › Orthogonal Jump Point Search
  - › Trace
- Buttons**
  - Start Search
  - Pause Search
  - Clear Walls

The bottom status bar shows the following information:

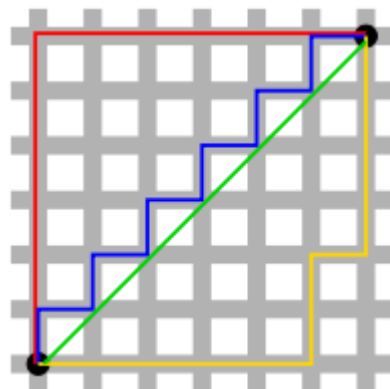
- length: 11.66
- time: 10.4500ms
- operations: 826
- Project Hosted on [Github](#)

# Distância Manhattan ou máxima, “city block”

Dado dois vetores  $X$  e  $Y$ , esta métrica é definida como o somatória dos módulos das diferenças:

$$d(X, Y) = \left( \sum_{i=1}^p |X_i - Y_i| \right)^1 =$$
$$|X_1 - Y_1| + |X_2 - Y_2| + \cdots + |X_p - Y_p|$$

$$d(x,y) = ( |linha(x) - linha(y)| + |coluna(x) - coluna(y)| )$$



Os caminhos vermelho, amarelo e azul têm o mesmo menor comprimento de caminho iguala a 12.

Na distância euclidiana, a linha verde tem comprimento aproximado de 8.49.

# Distância Manhattan ou máxima, “city block”

$$d(X, Y) = (\sum_{i=1}^p |X_i - Y_i|)^1 = |X_1 - Y_1| + |X_2 - Y_2| + \cdots + |X_p - Y_p|$$

$$d(x,y) = ( |linha(x) - linha(y)| + |coluna(x) - coluna(y)| )$$

Exemplo:

- $d(x,y) = |1-5| + |4-2|$
- $d(x,y) = 4 + 2$
- $d(x,y) = 6$

