



Busca Informada

Inteligência Artificial

Pontifícia Universidade Católica de Campinas

Prof. Dr. Denis M. L. Martins

Objetivos de Aprendizado

- Entender os **fundamentos teóricos e práticos** das técnicas de busca informada.
- **Diferenciar** entre algoritmos de busca não informada e os de busca informada, reconhecendo suas garantias de completude e optimalidade.
- **Projetar e avaliar funções heurísticas** para problemas clássicos (labirinto, 8-puzzle, planejamento de rotas) e analisar seu impacto no desempenho dos algoritmos.
- Considerar **aplicações de busca informada a cenários do mundo real**, justificando escolhas heurísticas e discutindo limitações práticas (custo computacional, espaço, etc.).

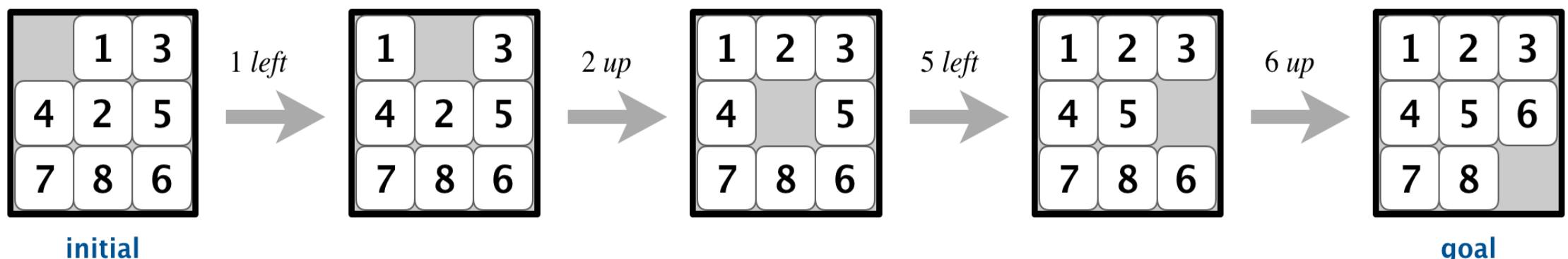


Panorama Geral

- **Busca não informada**: Explora o espaço de estados sem usar conhecimento adicional (ex.: BFS, DFS).
- **Busca informada (heurística)**
 - Usa conhecimento do problema para encontrar soluções de forma mais eficiente.
 - **Heurística**: função de avaliação que **estima o custo** de alcançar o objetivo.
 - Objetivo: **reduzir** o número de nós expandidos, mantendo ou aproximando-se da otimalidade.
- **Metáfora** – Navegar sem bússola vs. Navegar com bússola.
 - A heurística é a bússola que aponta na direção do objetivo.

Exemplo: Problema do 8-puzzle

- Quebra-cabeça de lógica composto por nove peças quadradas, numeradas de 1 a 8, dispostas em um tabuleiro (3×3). Uma das posições permanece vazia (na figura abaixo, \square cinza). O objetivo do jogo é reordenar as peças até que o tabuleiro esteja na configuração desejada (na figura, com as peças em ordem crescente).



Exemplo de jogadas no 8-Puzzle. Fonte da imagem: <https://8-puzzle.readthedocs.io/en/latest/>.

Como medir se estamos **chegando no objetivo**
no contexto do 8-puzzle?

Como medir se estamos **chegando no objetivo**
no contexto do 8-puzzle?

Distância de Manhattan

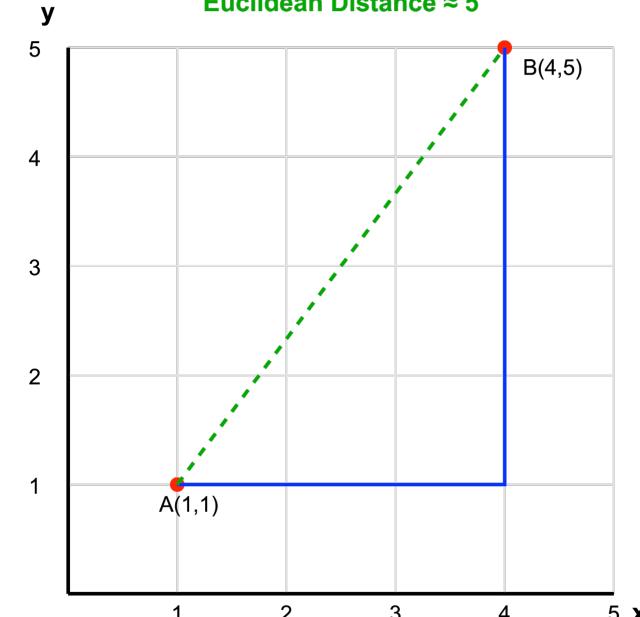
Número de peças fora do lugar

Heurística: Distância de Manhattan



Distância de Manhattan (em vermelho) e distância Euclideana (em azul).
Fonte: [Machine Learning with Swift](#).

$$\text{Manhattan Distance} = |1 - 4| + |1 - 5| = 3 + 4 = 7$$
$$\text{Euclidean Distance} \approx 5$$



Distância de Manhattan (em azul) e distância Euclideana (em verde).
Fonte: [DataCamp](#).

Heurística: Distância de Manhattan

A **distância de Manhattan** mede o número total de movimentos horizontais e verticais necessários para levar cada peça de sua posição atual até a posição correta.

$$h(n) = \sum_{i=1}^8 (|x_i - x_i^*| + |y_i - y_i^*|)$$

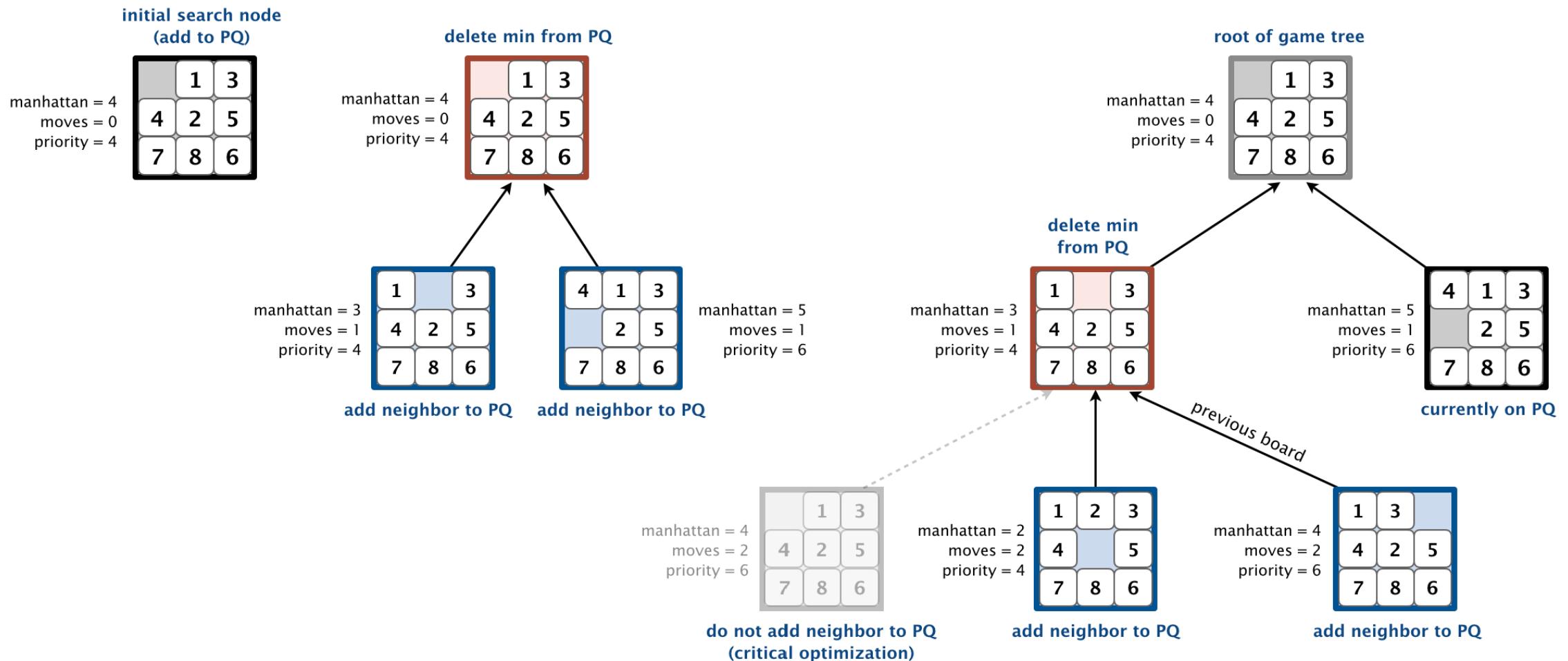
onde: (x_i, y_i) é a posição **atual** da peça i e (x_i^*, y_i^*) = posição **correta** da peça i .

Estado inicial:

2	8	3
1	6	4
7		5

Objetivo:

1	2	3
8		4
7	6	5



Fonte da imagem: <https://8-puzzle.readthedocs.io/en/latest/>

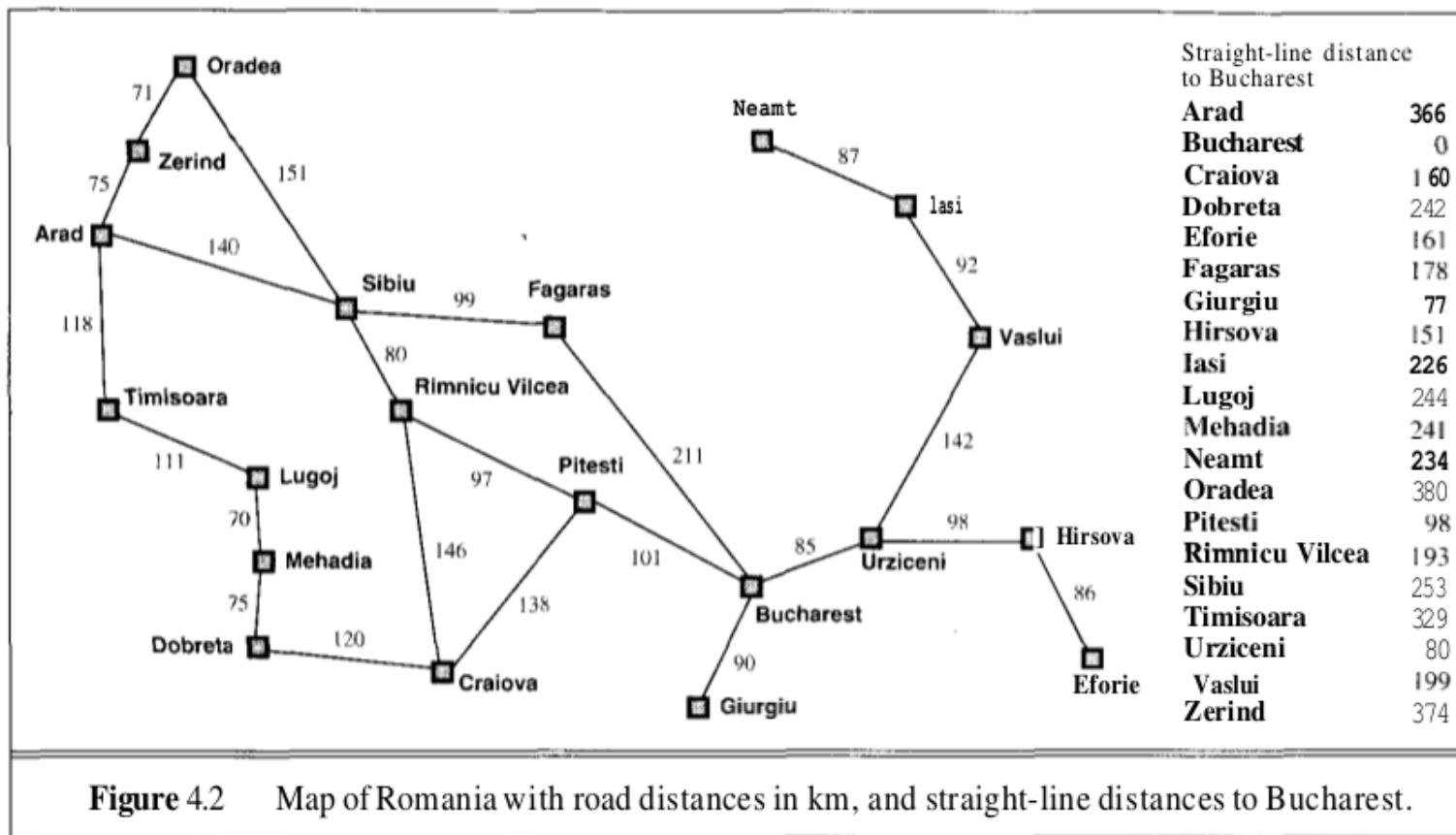
Heurística: Distância de Manhattan

- **Admissível?**
 - Sim: nunca superestima o custo real.
- **Fácil de implementar?**
 - Sim: simples e eficiente

A Distância de Manhattan é considerada uma
heurística **mais eficaz** que apenas contar
peças fora do lugar?

Exemplo: Distância entre Cidades da Romênia

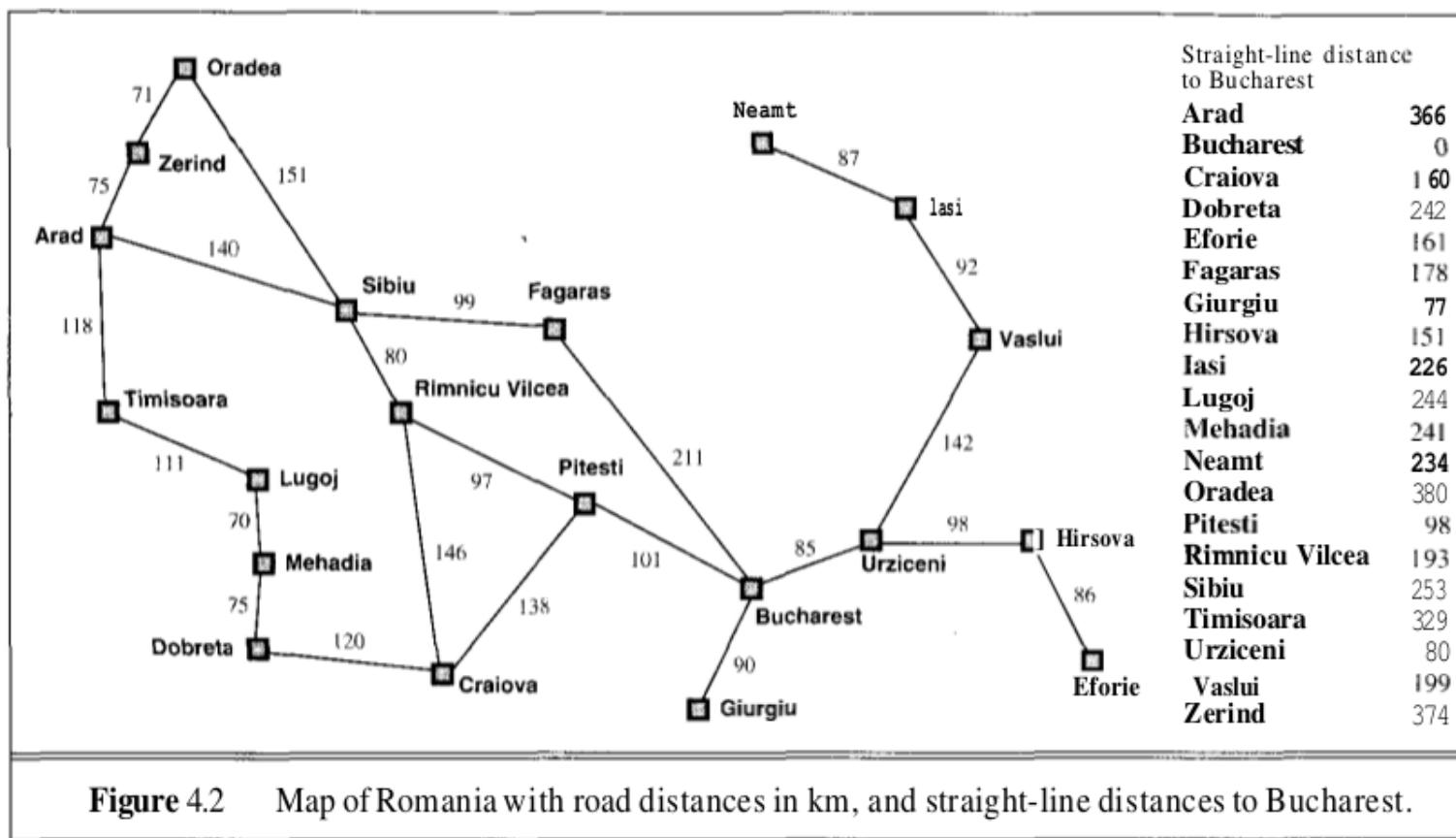
Considere encontrar o melhor caminho entre as cidades de Arad (estado inicial) e Bucharest (estado objetivo).



Qual **heurística** você usaria para encontrar
um bom (ou o melhor) caminho entre
Arad e Bucharest?

Exemplo: Distância entre Cidades da Romênia (cont.)

A tabela à direita da imagem mostra distância em linha reta estimada de cada cidade presente no mapa até à capital do país, Bucharest.



Algoritmos de Busca Informada

- Greedy Best-First Search
 - Usa apenas a heurística ($h(n)$)
 - Pode ser rápido, mas não garante ótima solução
- A* (A-estrela)
 - Combina:
 - custo acumulado ($g(n)$)
 - estimativa até o objetivo ($h(n)$)
 - Avalia: $f(n) = g(n) + h(n)$
 - Garante ótima solução se h for admissível.

Uma heurística é **admissível** se
nunca superestima o custo real até o objetivo.

Definição Formal de Heurística

Seja $G = (V, E)$ um grafo de estado.

Função heurística: $h : V \rightarrow \mathbb{R}_{\geq 0}$, onde $h(n)$ é a estimativa do custo mínimo restante para alcançar o objetivo a partir de n .

Admissibilidade: $\forall n, h(n) \leq h^*(n)$

- Não superestima o custo real h^* .

Consistência (ou Monotonicidade): $\forall (n, n'), h(n) \leq c(n, n') + h(n')$

Observação – A consistência implica admissibilidade.

Greedy-Best-First

1. Estrutura de dados: **Fila de prioridade** ordenada pelo valor $h(n)$.
2. Expande o nó com menor heurística.
3. **Completeness**: não garante solução se houver ciclos e heurística inválida.
4. **Optimality**: NÃO garantida (pode escolher caminhos sub-óptimos).

A* (A-estrela)

Função f(n): $f(n) = g(n) + h(n)$, onde $g(n)$ é o custo já percorrido até n .

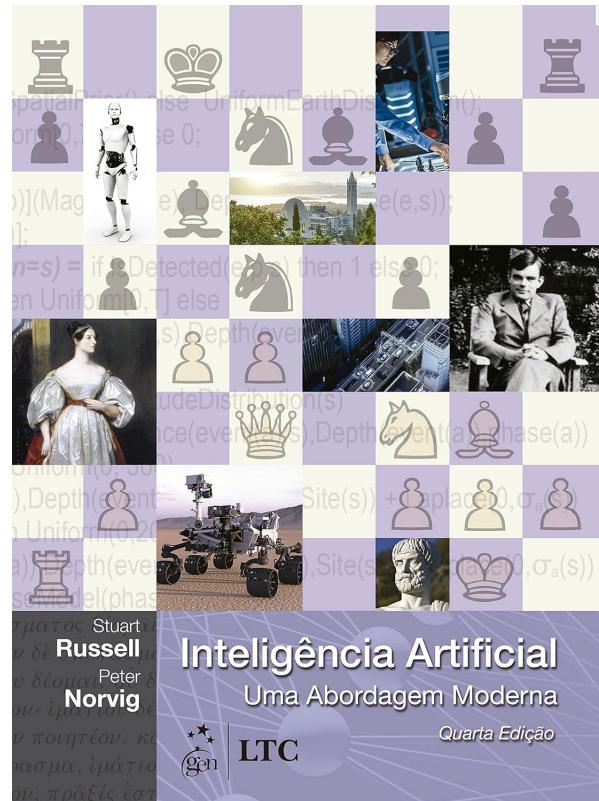
- Estrutura de dados: **Fila de prioridade** ordenada por $f(n)$.
- Se h for admissível e consistente, A* encontra um caminho ótimo.
 - Se custos forem positivos.
 - Se o fator de ramificação b é finito.
- **Complexidade:**
 - Tempo: $O(b^d)$ em pior caso, mas com heurística boa pode ser quase linear.
 - Espaço: $O(b^d)$ armazena em memória todos os nós gerados.

Avaliação de Heurísticas

1. **Precisão**: quão próxima está a estimativa ao custo real?
2. **Custo computacional**: heurística mais elaborada pode ser mais lenta.
3. **Generalização**: heurística construída para um domínio específico pode falhar em outro.

Resumo e Próximos Passos

- Busca informada reduz a complexidade exponencial através de heurísticas.
 - **Heurística**: estimativa do custo restante.
 - **Admissível**: nunca superestima.
 - **Consistente**: satisfaz desigualdade triangular; implica admissibilidade.
 - **Greedy-Best-First**: expande nó com menor $h(n)$.
 - **A***: usa $f(n) = g(n) + h(n)$.
 - O projeto da heurística é tão crucial quanto o algoritmo em si.



Atividade recomendada: Leitura do capítulo 3.

Perguntas e Discussão

- Qual é a diferença entre admissibilidade e consistência de uma heurística? Por que a consistência implica admissibilidade?
- Em quais cenários Greedy-Best-First pode superar A* em termos de tempo, apesar de não garantir otimalidade?
- Como o custo computacional de calcular uma heurística mais precisa afeta a eficiência geral do algoritmo?
- Como você justificaria a escolha de uma heurística em um problema real, considerando trade-offs entre precisão e velocidade?
- Em que situações a heurística pode levar o algoritmo a soluções sub-ótimas mesmo sendo admissível?