

Resolução de Problemas por Busca

Busca Informada

Algoritmo de Busca Local

Inteligência Artificial – 2020/1

Algoritmos de Busca Local

- Os algoritmos estudados até agora são adequados para situações em que a solução é uma sequência de ações.
- Os **Algoritmos de Busca Local** não se enquadram nesse modelo clássico de busca pois avaliam e modificam apenas um ou mais estados atuais em vez de explorar sistematicamente os caminhos a partir de um estado inicial
- São mais adequados para problemas em que o **caminho para a solução não importa**.
- **A solução é o estado final**.
- Os algoritmos de busca local usam apenas um estado corrente e em geral se movem apenas para os estados vizinhos desse estado.

Algoritmos de Busca Local

- Exemplos de problemas para os quais a busca local é mais adequada:
 - Problema das oito rainhas
 - Projetos de circuitos integrados
 - Leiaute de instalações industriais
 - Escalonamento de jornadas de trabalho
 - Programação automática

Algoritmos de Busca Local

- Vantagens:
 - Usam pouca memória;
 - Geralmente encontram soluções **razoáveis** em grandes espaços de busca.
- Limitações:
 - Movimentos são **irrevogáveis** (nunca volta a um estado anterior para tentar caminhos alternativos);
 - Pode levar a uma **solução sub-ótima** ou não levar a uma solução.

Algoritmos de Busca Local

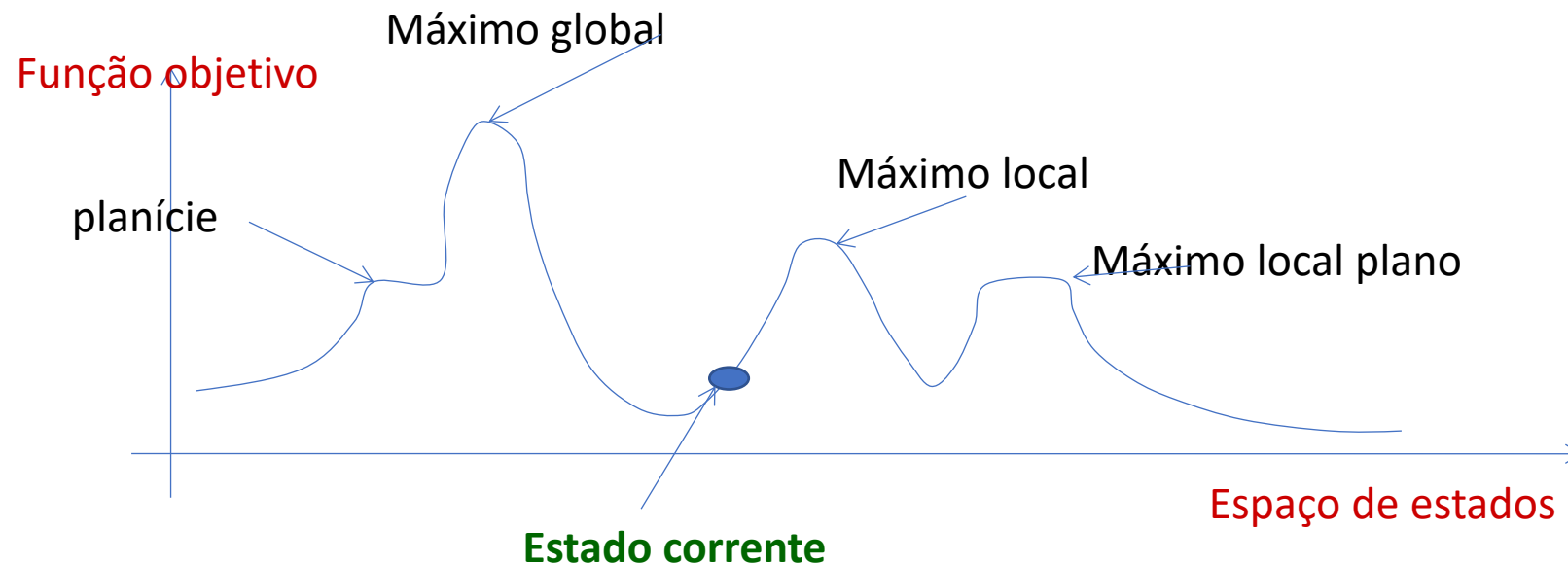
- Problemas de otimização:
 - Os algoritmos de busca local são adequados para tratar problemas de otimização:
 - O objetivo é encontrar o **melhor estado** de acordo com uma **função objetivo**

Topologia do espaço de estados

Tem uma posição (espaço de estados) e uma elevação (heurística)

Se a elevação representa custo, o objetivo será encontrar o Valor mais baixo – **mínimo global**

Se a elevação representa a função objetivo, o objetivo será Encontrar o valor mais alto – **máximo global**.



Algoritmo de Subida da colina (Hill-Climbing)

- Enquadra-se na categoria de algoritmos de busca local
- Consiste nos seguintes passos:
 - **Expande** um nó, avalia seus descendentes;
 - (Não armazena irmãos nem pais);
 - **Seleciona** o melhor entre os descendentes para continuar;
 - **Para** quando encontrar um nó melhor que todos os descendentes.

Algoritmo de Subida da colina (Hill-Climbing)

Variáveis locais:

Corrente (nó corrente)

Vizinho (próximo nó)

Procedure subida-da-colina

Corrente = estado inicial;

Repeat

Gere todos os filhos de Corrente e avalie;

Vizinho = sucessor de Corrente com valor mais alto;

If (valor de Vizinho \leq valor de Corrente)

 retorne Corrente;

Corrente = Vizinho;

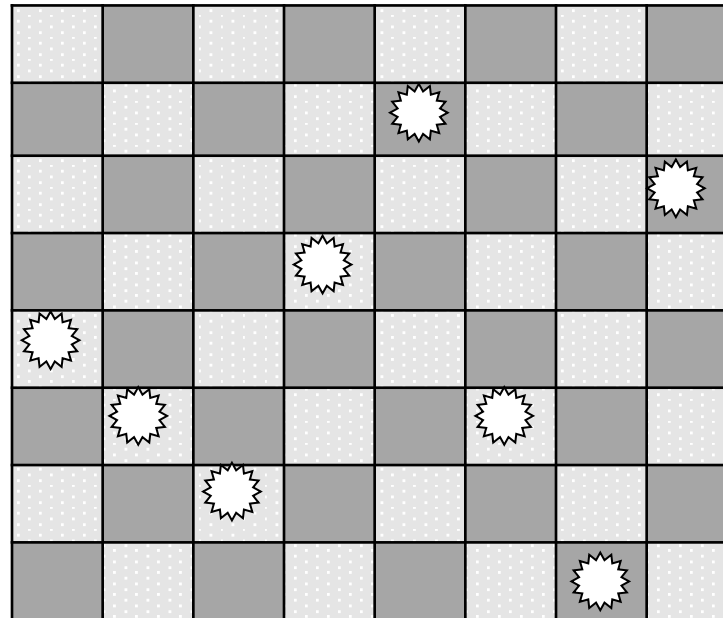
Se fosse usada uma estimativa de custo heurística h , encontraríamos o vizinho com o h mais baixo.

Exemplo: Problema das 8 rainhas

- Usa **formulação de estados completos**: cada estado tem 8 rainhas, uma em cada coluna.
- **Função sucessora**: retorna todos os estados possíveis gerados pela movimentação de uma única rainha para outro quadrado na mesma coluna
- Cada estado tem $8 \times 7 = 56$ sucessores
- **Função heurística**: número de pares de rainhas que estão atacando umas às outras, direta ou indiretamente.
- Caso exista mais de um melhor sucessor, a escolha é aleatória

Exemplo: Problema das 8 rainhas

- Estado inicial: 8 rainhas são colocadas aleatoriamente, uma em cada coluna











Movimentos: cada rainha pode mudar para uma das outras 7 linhas, na mesma coluna ($8 \times 7 = 56$ movimentos)

Exemplo: Problema das 8 rainhas

Heurística: número de pares de rainhas que estão atacando umas às outras, direta ou indiretamente.

Estado com
estimativa
heurística
 $h=17$

| | | | | | | | |
|---|--|--|---|---|---|--|---|
| 18 | 12 | 14 | 13 | 13 | 12 | 14 | 14 |
| 14 | 16 | 13 | 15 | 12 | 14 | 12 | 16 |
| 14 | 12 | 18 | 13 | 15 | 12 | 14 | 14 |
| 15 | 14 | 14 |  | 13 | 16 | 13 | 16 |
|  | 14 | 17 | 15 |  | 14 | 16 | 16 |
| 17 |  | 16 | 18 | 15 |  | 15 |  |
| 18 | 14 |  | 15 | 15 | 14 |  | 16 |
| 14 | 14 | 13 | 17 | 12 | 14 | 12 | 18 |









Os números em cada casa indicam o valor da função heurística caso a rainha da coluna correspondente mude para essa posição

Exemplo: Problema das 8 rainhas

Heurística: número de pares de rainhas que estão atacando umas às outras, direta ou indiretamente.

Os melhores movimentos tem $h=12$

Estado com
estimativa
heurística
 $h=17$

| | | | | | | | |
|---|---|---|---|---|--|---|--|
| 18 | 12 | 14 | 13 | 13 | 12 | 14 | 14 |
| 14 | 16 | 13 | 15 | 12 | 14 | 12 | 16 |
| 14 | 12 | 18 | 13 | 15 | 12 | 14 | 14 |
| 15 | 14 | 14 |  | 13 | 16 | 13 | 16 |
|  | 14 | 17 | 15 |  | 14 | 16 | 16 |
| 17 |  | 16 | 18 | 15 |  | 15 |  |
| 18 | 14 |  | 15 | 15 | 14 |  | 16 |
| 14 | 14 | 13 | 17 | 12 | 14 | 12 | 18 |

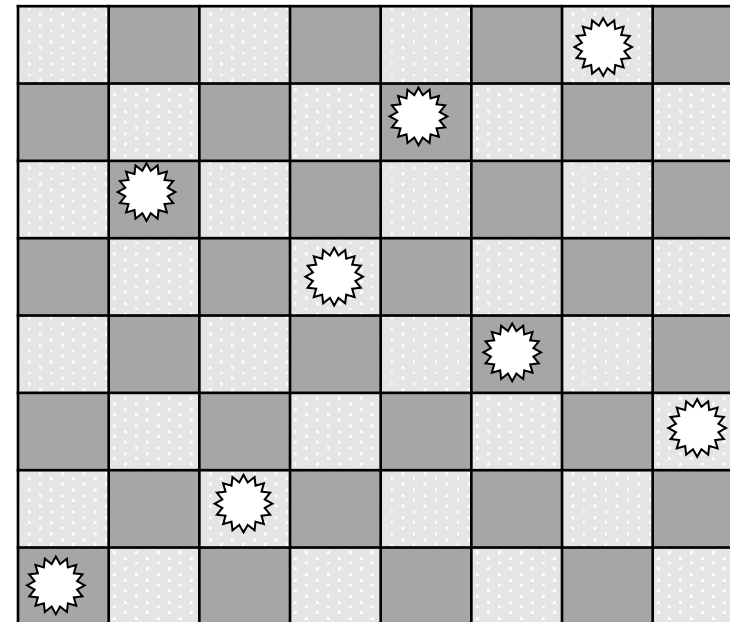
Exemplo: Problema das 8 rainhas

5 passos são necessários para sair da configuração da esquerda para a configuração da direita

Estado com heurística $h=17$

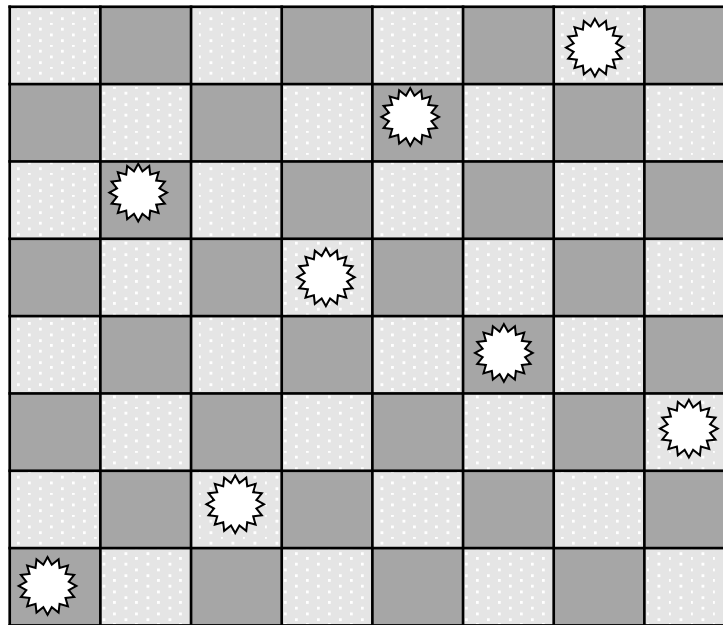
| | | | | | | | |
|----|----|----|----|----|----|----|----|
| 18 | 12 | 14 | 13 | 13 | 12 | 14 | 14 |
| 14 | 16 | 13 | 15 | 12 | 14 | 12 | 16 |
| 14 | 12 | 18 | 13 | 15 | 12 | 14 | 14 |
| 15 | 14 | 14 | | 13 | 16 | 13 | 16 |
| | 14 | 17 | 15 | | 14 | 16 | 16 |
| 17 | | 16 | 18 | 15 | | 15 | |
| 18 | 14 | | 15 | 15 | 14 | | 16 |
| 14 | 14 | 13 | 17 | 12 | 14 | 12 | 18 |

Estado com heurística $h=1$



Exemplo: Problema das 8 rainhas

Essa configuração é um **Mínimo local** no espaço de estados das 8-rainhas
Tem $h=1$ mas todo sucessor tem custo mais alto



A busca local termina aqui, sem encontrar o mínimo global

Fim do tópico Busca Informada