

# Inteligência Artificial

## Métodos de Busca Local: Hill Climbing

João C. P. da Silva

Dept. Ciência da Computação - UFRJ

December 28, 2020

# Algoritmos de Busca Local e Problemas de Otimização

- Em muitos problemas de otimização, o caminho é irrelevante : o estado objetivo é a solução. Exemplo: problema das 8-rainhas

# Algoritmos de Busca Local e Problemas de Otimização

- Em muitos problemas de otimização, o caminho é irrelevante : o estado objetivo é a solução. Exemplo: problema das 8-rainhas
- **Objetivo:** Encontrar a configuração ótima ou encontrar a configuração que satisfaça as restrições do problema.

# Algoritmos de Busca Local e Problemas de Otimização

- Em muitos problemas de otimização, o caminho é irrelevante : o estado objetivo é a solução. Exemplo: problema das 8-rainhas
- **Objetivo:** Encontrar a configuração ótima ou encontrar a configuração que satisfaça as restrições do problema.
- **Algoritmos de busca local** que mantém um **único estado corrente** e tenta melhorá-lo, examinando somente seus vizinhos.

# Algoritmos de Busca Local e Problemas de Otimização

- Em muitos problemas de otimização, o caminho é irrelevante : o estado objetivo é a solução. Exemplo: problema das 8-rainhas
- **Objetivo:** Encontrar a configuração ótima ou encontrar a configuração que satisfaça as restrições do problema.
- **Algoritmos de busca local** que mantêm um **único estado corrente** e tenta melhorá-lo, examinando somente seus vizinhos.
- Úteis para resolver problemas de otimização cujo objetivo é encontrar o melhor estado de acordo com uma **função objetivo**.

# Hill-climbing

- **Busca local gulosa**

# Hill-climbing

- **Busca local gulosa**
- Utiliza apenas uma configuração do problema: **nó corrente**.

# Hill-climbing

- **Busca local gulosa**
- Utiliza apenas uma configuração do problema: **nó corrente**.
- Considera apenas os **nós vizinhos** ao nó corrente.



# Hill-climbing

- **Busca local gulosa**
- Utiliza apenas uma configuração do problema: **nó corrente**.
- Considera apenas os **nós vizinhos** ao nó corrente.
- Usando uma **função objetivo**, segue sempre em direção do nó cujo valor da função aumenta.

# Hill-climbing

- **Busca local gulosa**
- Utiliza apenas uma configuração do problema: **nó corrente**.
- Considera apenas os **nós vizinhos** ao nó corrente.
- Usando uma **função objetivo**, segue sempre em direção do nó cujo valor da função aumenta.
- Termina quando atinge o **máximo**: nenhum vizinho tem um valor da função objetivo maior que o do nó corrente.

# Hill-climbing

```
function HILL-CLIMBING(Estado_Inicial)  
  Corrente  $\leftarrow$  Estado_Inicial  
  while true do  
    Vizinho  $\leftarrow$  sucessor(Corrente) com maior valor de  $f(\text{sucessor}(\textit{Corrente}))$   
    if  $f(\textit{Vizinho}) \leq f(\textit{Corrente})$  then return Corrente  
    end if  
    Corrente  $\leftarrow$  Vizinho  
  end while  
end function
```

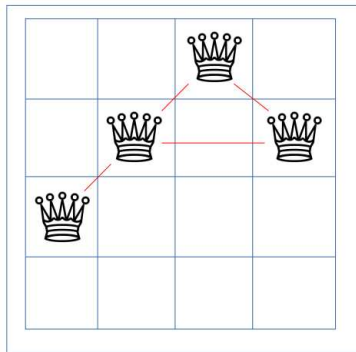
$f(\cdot)$  : função objetivo

# Hill-climbing - 4-Rainhas

- **Função heurística  $h(n)$ :** número de ataques no tabuleiro  $n$





# Hill-climbing - 4-Rainhas

- **Função heurística  $h(n)$ :** número de ataques no tabuleiro  $n$
- **Função objetivo  $f(n)$ :**  $-h(n)$



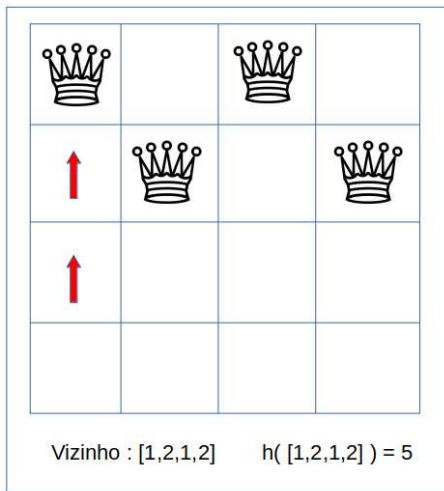
$$h(n) = 5 \rightarrow f(n) = -h(n) = -5$$

# Hill-climbing - 4-Rainhas

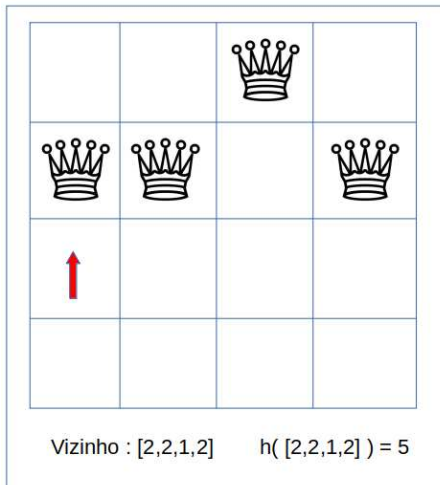
			
			
			

Nó corrente : [3,2,1,2]       $h([3,2,1,2]) = 5$

# Hill-climbing - 4-Rainhas







# Hill-climbing - 4-Rainhas



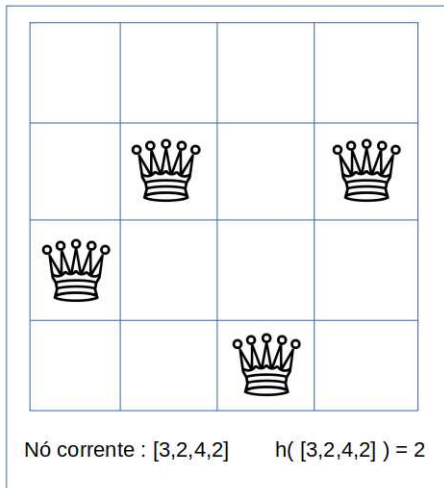


# Hill-climbing - 4-Rainhas





5	3		4
5		4	
	3	5	4
3	3	2	4

Nó corrente : [3,2,1,2]       $h([3,2,1,2]) = 5$

# Hill-climbing - 4-Rainhas







# Hill-climbing - 4-Rainhas

2	0	5	1
4		4	
	2	5	3
2	3		3

Nó corrente : [3,2,4,2]       $h([3,2,4,2]) = 2$





## Hill-climbing - 4-Rainhas

Nó corrente : [3,1,4,2]       $h([3,1,4,2]) = 0$   
**Solução !**






Progride rapidamente em direção a uma solução.

# Hill-climbing - 4-Rainhas







Nó corrente : [1,3,3,1]       $h([1,3,3,1]) = 4$

# Hill-climbing - 4-Rainhas





Vizinho: [2,3,3,1]       $h([2,3,3,1]) = 3$

# Hill-climbing - 4-Rainhas

Vizinho: [3,3,3,1]       $h([3,3,3,1]) = 4$





# Hill-climbing - 4-Rainhas

	4	4	
3	4	4	3
4			4
4	3	3	4

Nó corrente : [1,3,3,1]       $h([1,3,3,1]) = 4$







# Hill-climbing - 4-Rainhas





Nó corrente: [2,3,3,1]       $h([2,3,3,1]) = 3$

# Hill-climbing - 4-Rainhas

4	2	3	
	2	5	4
4			4
3	1	3	3





Nó corrente: [2,3,3,1]       $h([2,3,3,1]) = 3$

## Hill-climbing - 4-Rainhas

Nó corrente: [2,4,3,1]       $h([2,4,3,1]) = 1$

# Hill-climbing - 4-Rainhas

3	2	1	
	2	2	4
3	3		2
3		2	3

Nó corrente: [2,4,3,1]       $h([2,4,3,1]) = 1$

# Hill-climbing

- O algoritmo para quando o melhor nó sucessor e o nó corrente tem o mesmo valor.
- O que poderia ocorrer se ele continuasse?

# Hill-climbing

- O algoritmo para quando o melhor nó sucessor e o nó corrente tem o mesmo valor.
- O que poderia ocorrer se ele continuasse?  
Poderia entrar em loop se estivesse em um platô.
- Como resolver isso?

# Hill-climbing

- O algoritmo para quando o melhor nó sucessor e o nó corrente tem o mesmo valor.
- O que poderia ocorrer se ele continuasse?  
Poderia entrar em loop se estivesse em um platô.
- Como resolver isso?  
Colocar um limite no número de passos que ele pode andar 'de lado'

# Hill-climbing

- **Problemas**

- Máximo local
- Platôs (flat) e Ressaltos (shoulders)

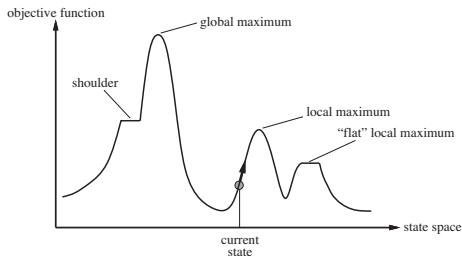


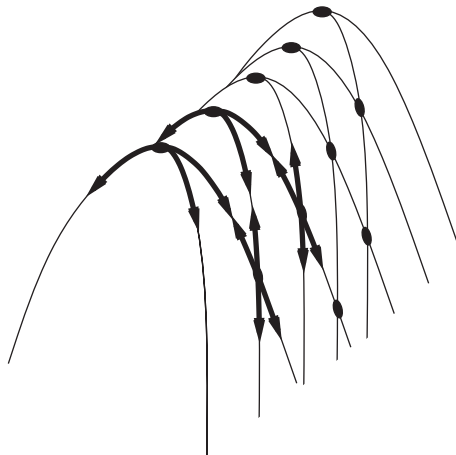
Figure: Artificial Intelligence: A Modern Approach - Russell and Norvig



# Hill-climbing

- **Problemas**

- **Cordilheiras (Ridges)**: sequência de máximos locais que dificultam a navegação do algoritmo



# Hill-climbing

## Variações

- **Hill-climbing Estocástico:** escolha aleatória entre os movimentos de subida. A probabilidade varia de acordo com a inclinação do movimento de subida. Converge mais lentamente, mas em certos casos encontra melhores soluções.

# Hill-climbing

## Variações

- **Hill-climbing Estocástico:** escolha aleatória entre os movimentos de subida. A probabilidade varia de acordo com a inclinação do movimento de subida. Converge mais lentamente, mas em certos casos encontra melhores soluções.
- **First Choice Hill-climbing:** implementa hill-climbing estocástico pela geração aleatória de sucessores até que um melhor que o nó corrente seja encontrado. Boa estratégia quando o nó corrente tem muitos sucessores.

# Hill-climbing

## Variações

- **Hill-climbing Estocástico:** escolha aleatória entre os movimentos de subida. A probabilidade varia de acordo com a inclinação do movimento de subida. Converge mais lentamente, mas em certos casos encontra melhores soluções.
- **First Choice Hill-climbing:** implementa hill-climbing estocástico pela geração aleatória de sucessores até que um melhor que o nó corrente seja encontrado. Boa estratégia quando o nó corrente tem muitos sucessores.
- **Random-restart hill-climbing:** faz uma série de buscas hill-climbing, gerando estados iniciais aleatórios, até encontrar uma solução.
  - Se cada busca hill-climbing tem probabilidade  $p$  de sucesso, então o número esperado de buscas é  $\frac{1}{p}$ .
  - **8-rainhas:**  $p \approx 0.14$ , são necessários 7 buscas para encontrar uma solução.

# Inteligência Artificial

## Métodos de Busca Local: Hill Climbing

João C. P. da Silva

Dept. Ciência da Computação - UFRJ

December 28, 2020