

SICO7A

SISTEMAS INTELIGENTES 1

Aula 03 E - Buscas Heurísticas

Prof. Rafael G. **Mantovani**

Roteiro



- 1** Introdução
- 2** Subida de Encosta
- 3** Busca pela Melhor Escolha
- 4** Exercícios
- 5** Referências

Roteiro

- 1 Introdução**
- 2 Subida de Encosta**
- 3 Busca pela Melhor Escolha**
- 4 Exercícios**
- 5 Referências**

Introdução



Roteiro



- 1 Introdução
- 2 Subida de Encosta
- 3 Busca pela Melhor Escolha
- 4 Exercícios
- 5 Referências

Subida de Encosta

- ***Hill Climbing***

- Maneira mais simples de se implementar a busca heurística
- Analogia: alpinista cego sempre segue o caminho mais íngreme até não poder avançar

Subida de Encosta

□ *Hill Climbing*

- Maneira mais simples de se implementar a busca heurística
- Analogia: alpinista cego sempre segue o caminho mais íngreme até não poder avançar

- * Expande o estado atual e avalia seus filhos
- * o **melhor** filho é selecionado e expandido
- * nenhum dos outros filhos ou genitores é considerado

Subida de Encosta



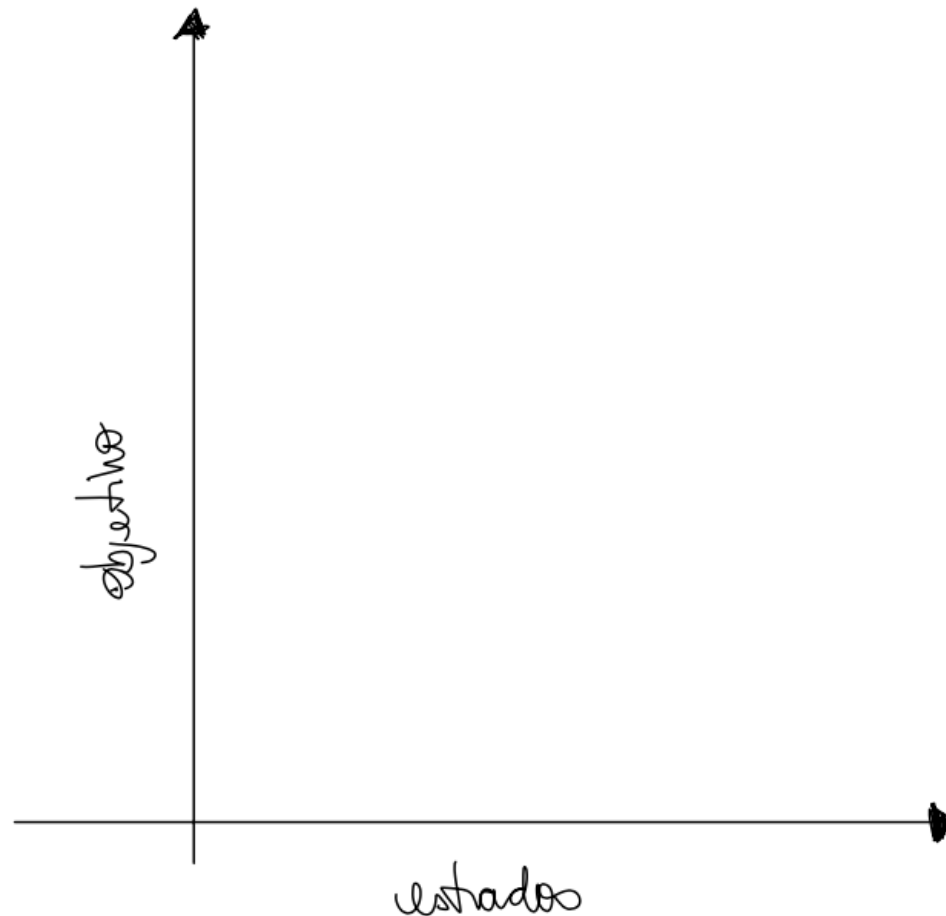
PROBLEMAS

Subida de Encosta

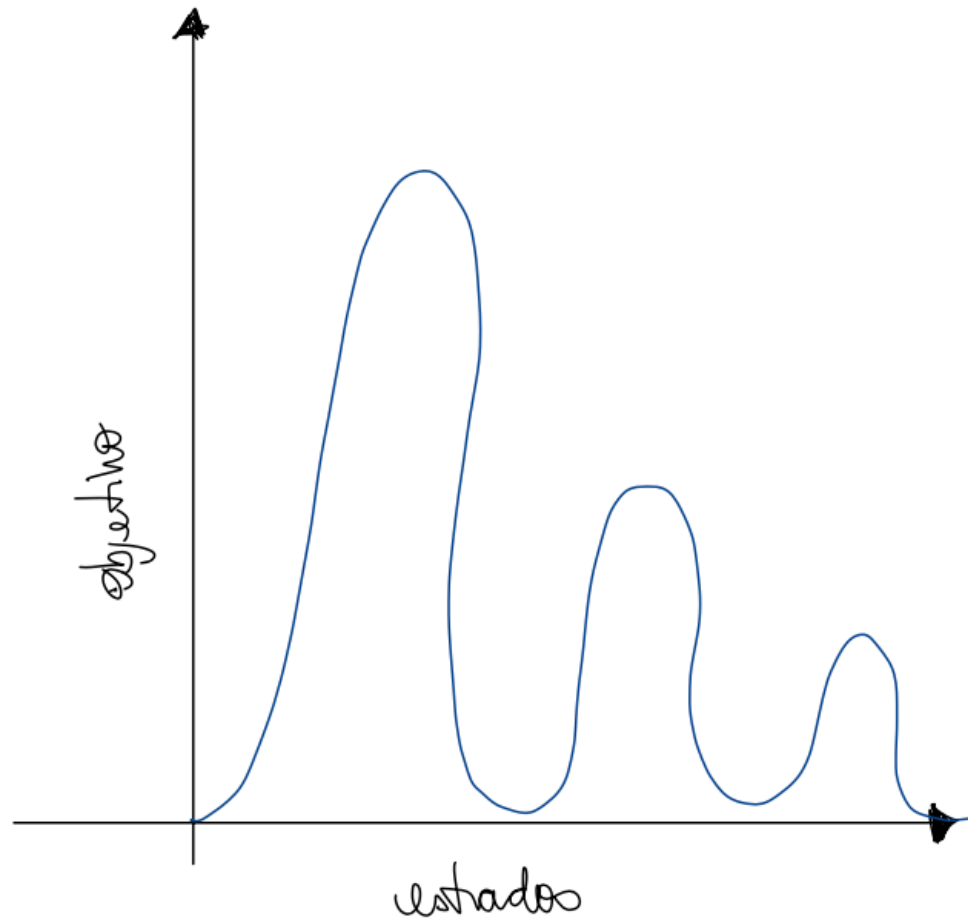
PROBLEMAS

- Algoritmo não regista o histórico do processo de subida
- Um problema é sua tendência de ficar preso no máximos locais

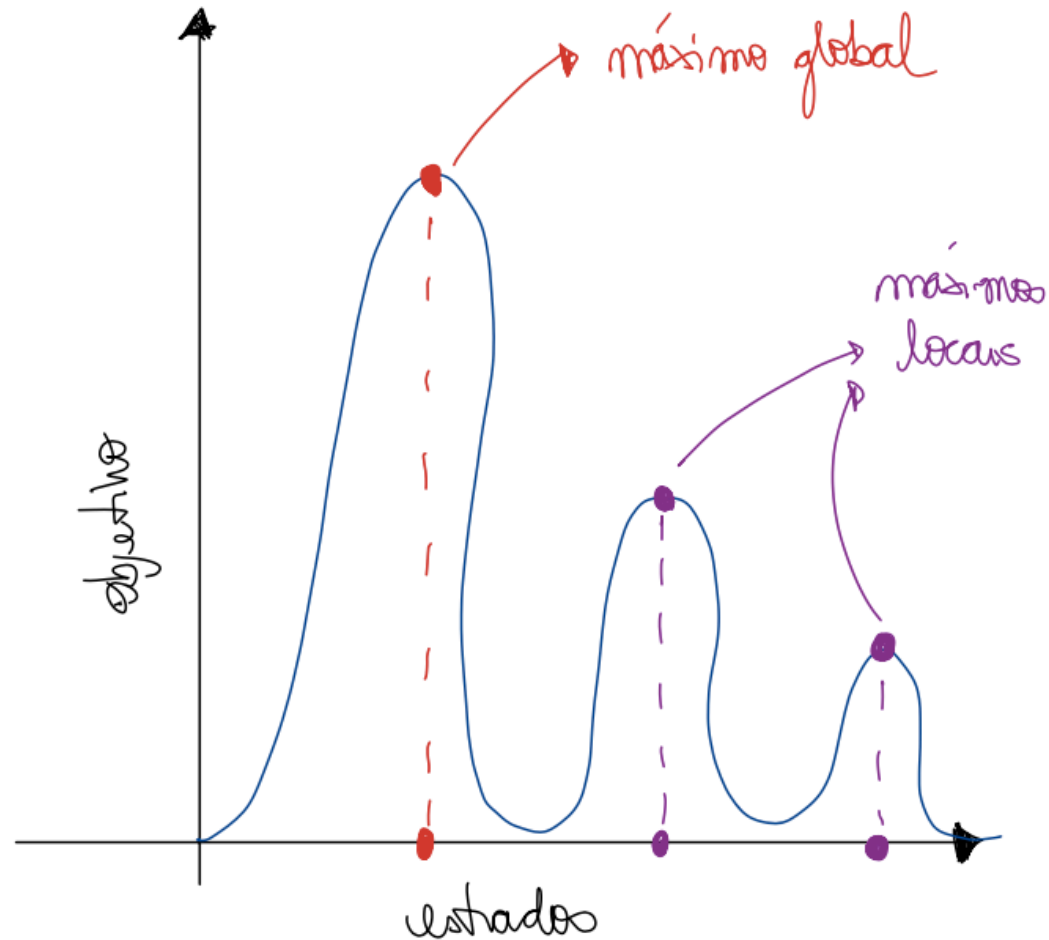
Subida de Encosta



Subida de Encosta



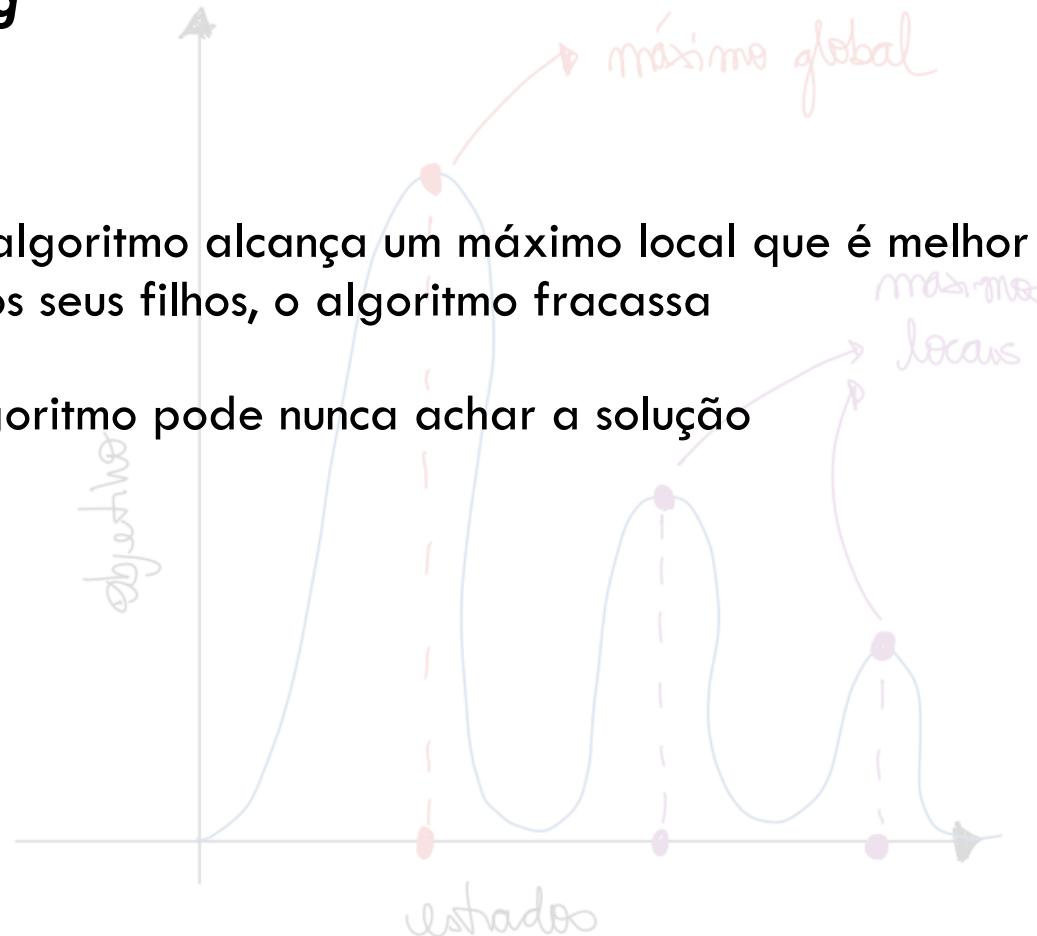
Subida de Encosta



Subida de Encosta

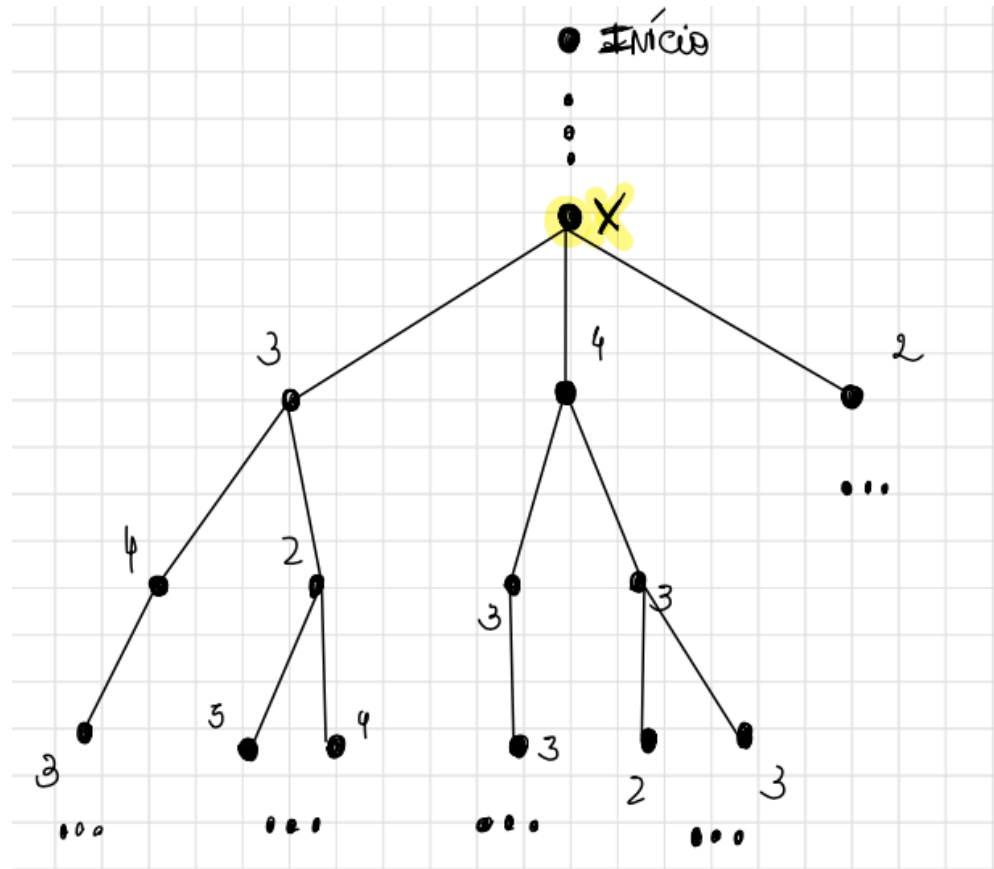
□ Hill Climbing

- Se o algoritmo alcança um máximo local que é melhor que qualquer um dos seus filhos, o algoritmo fracassa
- O algoritmo pode nunca achar a solução



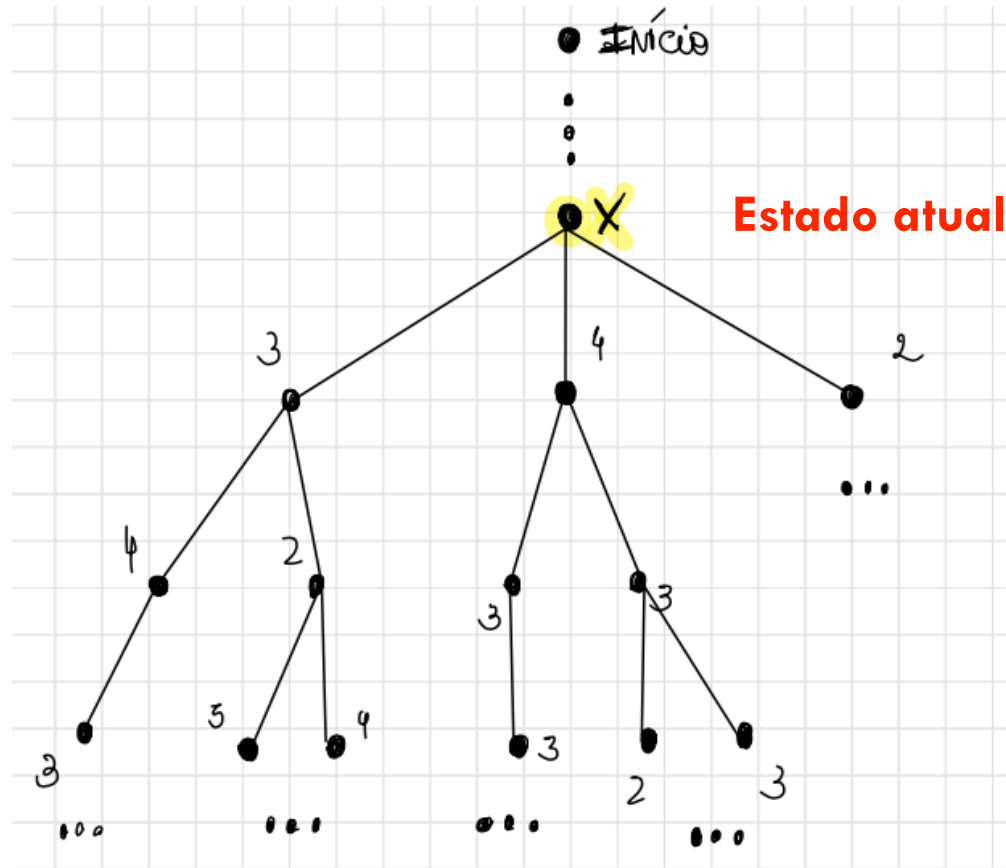
Subida de Encosta

Fig 1: Problema do máximo local para Hill Climbing com antecipação de 3 níveis



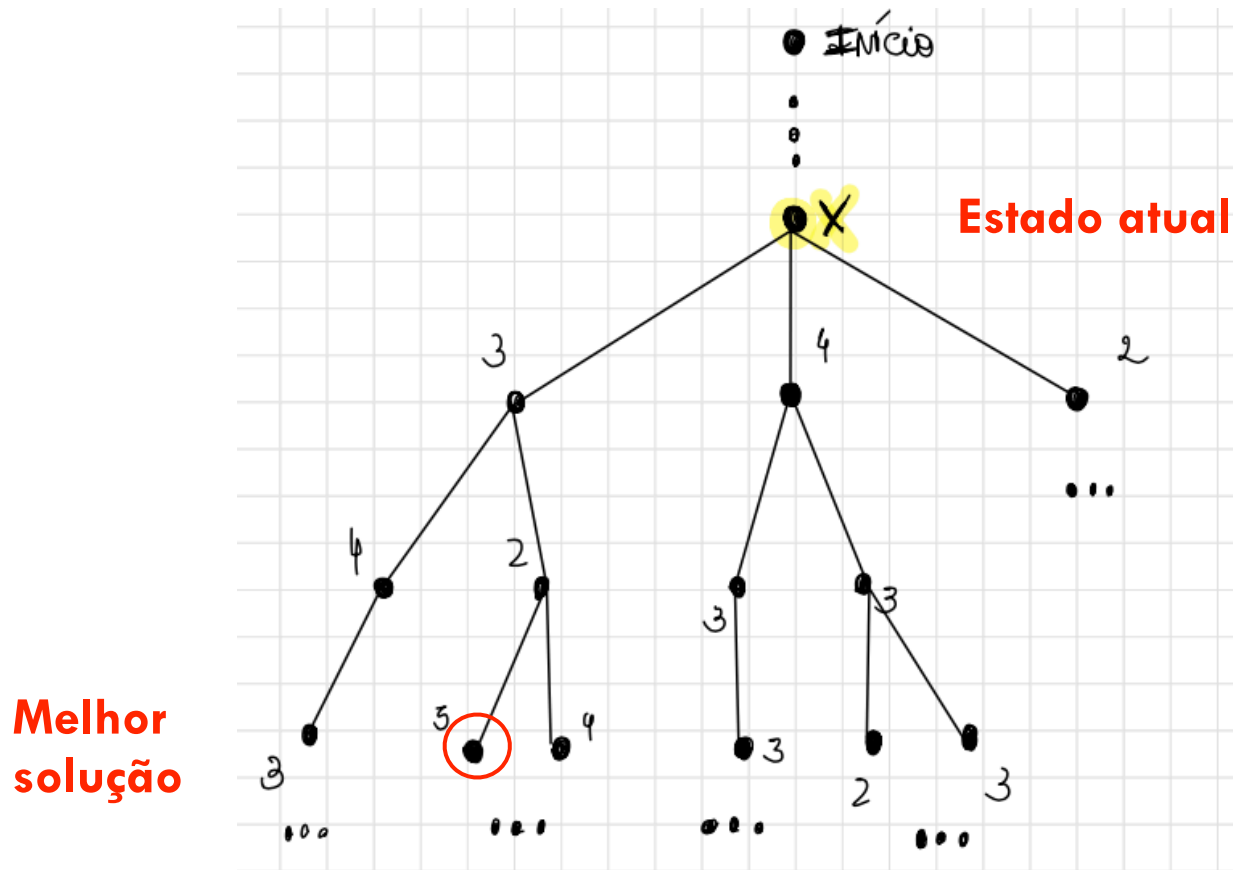
Subida de Encosta

Fig 1: Problema do máximo local para Hill Climbing com antecipação de 3 níveis



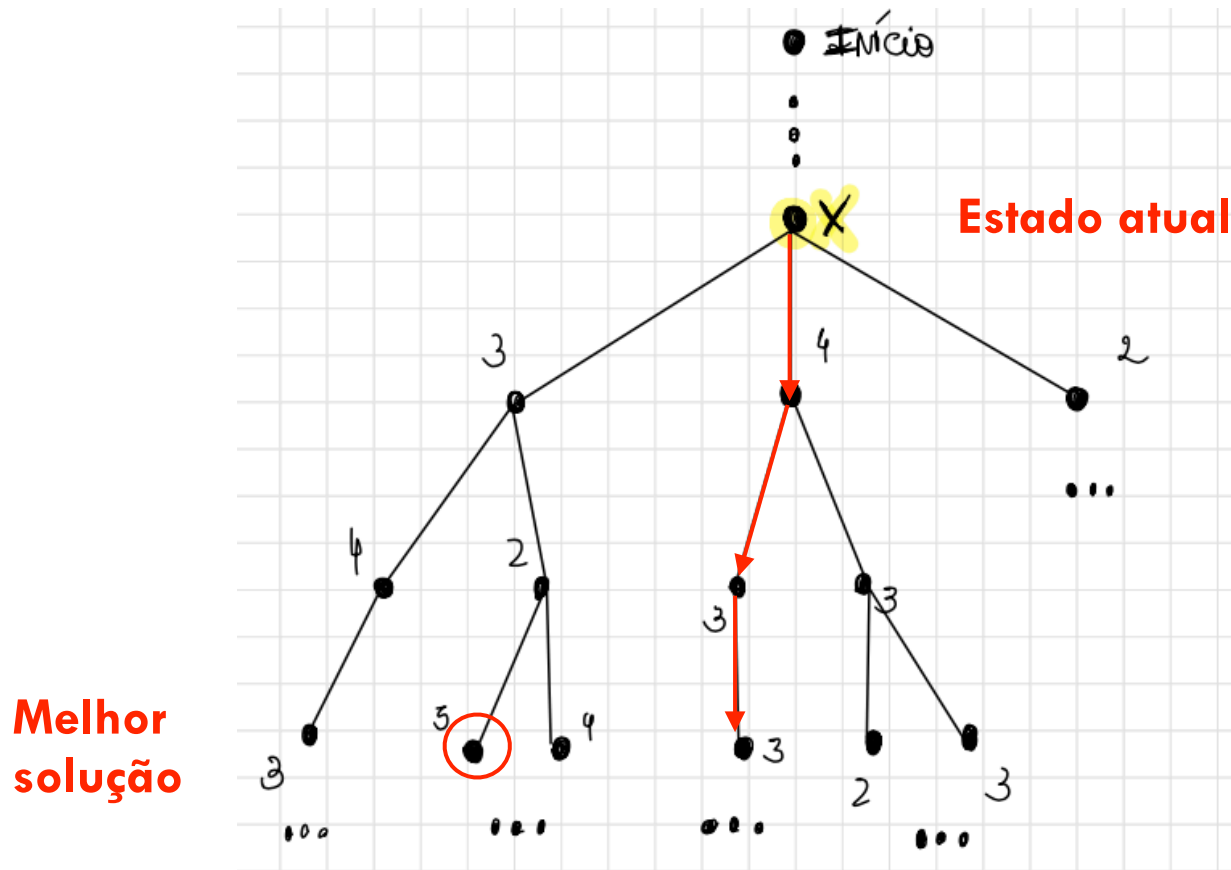
Subida de Encosta

Fig 1: Problema do máximo local para Hill Climbing com antecipação de 3 níveis



Subida de Encosta

Fig 1: Problema do máximo local para Hill Climbing com antecipação de 3 níveis



Subida de Encosta



□ ***Solução?***

Pseudocódigo: Subida de Encosta

Hill_Climbing (Inicial):

Pseudocódigo: Subida de Encosta

Hill_Climbing (Inicial):

```
1.  X = [Inicial]                                // Inicialização
2.  Enquanto é possível gerar filhos de X faça:
3.      Se X for um objetivo, então retorne o Caminho de Inicial até X
4.      Senão
5.          Gere filhos de X
6.          Avalie os filhos de X usando uma função heurística H
7.          X = melhor filho de acordo com H      // proximo estado
8.      fim se-senão
9.  fim enquanto
10. retorna FALHA
11. fim algoritmo
12.
```

Roteiro



- 1 Introdução
- 2 Subida de Encosta
- 3 Busca pela Melhor Escolha
- 4 Exercícios
- 5 Referências

Busca pela Melhor Escolha



Busca pela Melhor Escolha

- **Backtracking**
- **Busca em profundidade**
- **Busca em amplitude**
- **Subida de encosta**



Podem funcionar eficientemente, de acordo com a função de heurística usada

Busca pela Melhor Escolha

- Backtracking
 - Busca em profundidade
 - Busca em amplitude
 - Subida de encosta
- } Podem funcionar eficientemente, de acordo com a função de heurística usada

☹... Requer algoritmo mais flexível e eficiente

* Busca pela melhor escolha (**Fila de Prioridade**)

Busca pela Melhor Escolha

Sugestão de implementação: Usar duas listas de estados para registrar o progresso através do espaço de estado

■ **ABERTOS**

■ **FECHADOS**

Busca pela Melhor Escolha

Sugestão de implementação: Usar duas listas de estados para registrar o progresso através do espaço de estado

■ **ABERTOS:** registra a fronteira atual da busca

■ **FECHADOS:** registra os estados já visitados

Busca pela Melhor Escolha

Sugestão de implementação: Usar duas listas de estados para registrar o progresso através do espaço de estado

- **ABERTOS:** registra a fronteira atual da busca
- **FECHADOS:** registra os estados já visitados
- **HEURÍSTICA:** avalia os estados de acordo com uma estimativa de proximidade com um objetivo. Os estados são mantidos **ordenados** na lista de **ABERTOS**.

Busca pela Melhor Escolha

Sugestão de implementação: Usar duas listas de estados para registrar o progresso através do espaço de estado

■ **ABERTOS:** registra a fronteira atual da busca

■ **FECHADOS:** registra os estados já visitados

■ **HEURÍSTICA:** avalia os estados de acordo com uma estimativa de proximidade com um objetivo. Os estados são mantidos **ordenados** na lista de **ABERTOS**.

* Cada iteração considera o estado “mais promissor” da lista de ABERTOS

Pseudocódigo: Busca pela melhor escolha

Busca_Melhor_Escolha (Inicial):

```
1.  ABERTOS = [Inicial]                                // Inicialização
2.  FECHADOS = [ ]
3.  Enquanto ABERTOS != [ ] faça:
4.      Remova o estado mais à esquerda de ABERTOS, chame-o de X
5.      Se X for um objetivo, então retorne o Caminho de Inicial até X
6.      Senão
7.          Gere filhos de X
8.          Para cada filho de X faça:
9.              Caso:
10.                 → o filho não está em ABERTOS e não está FECHADOS:
11.                     atribua ao filho um valor heurístico
12.                     acrescente o filho a ABERTOS
```

Pseudocódigo: Busca pela melhor escolha

Busca_Melhor_Escolha (Inicial):

```
13.  | | | | | → o filho já está em ABERTOS:
14.  | | | | |   Se o filho foi alcançado por um caminho mais curto, então:
15.  | | | | |   | * de ao estado em ABERTOS o caminho mais curto
16.  | | | | | → o filho já está em FECHADOS:
17.  | | | | |   Se o filho foi alcançado por um caminho mais curto, então:
18.  | | | | |   | * retire o estado de FECHADOS
19.  | | | | |   | * adicione o filho em ABERTOS
20.  | | | | |
21.  | | | | |   fim caso
22.  | | | | |   fim para
23.  | | | | |   fim se-senão
```

Pseudocódigo: Busca pela melhor escolha

Busca_Melhor_Escolha (Inicial):

```
24.  | Coloque X em FECHADOS
25.  | Reordene os estados em ABERTOS de acordo com o método heurístico
    | (melhor mais à esquerda)                                // Fila de Prioridade
26.  | fim enquanto
27.  | retorna FALHA                                           // ABERTOS está vazio
28.  fim algoritmo
```

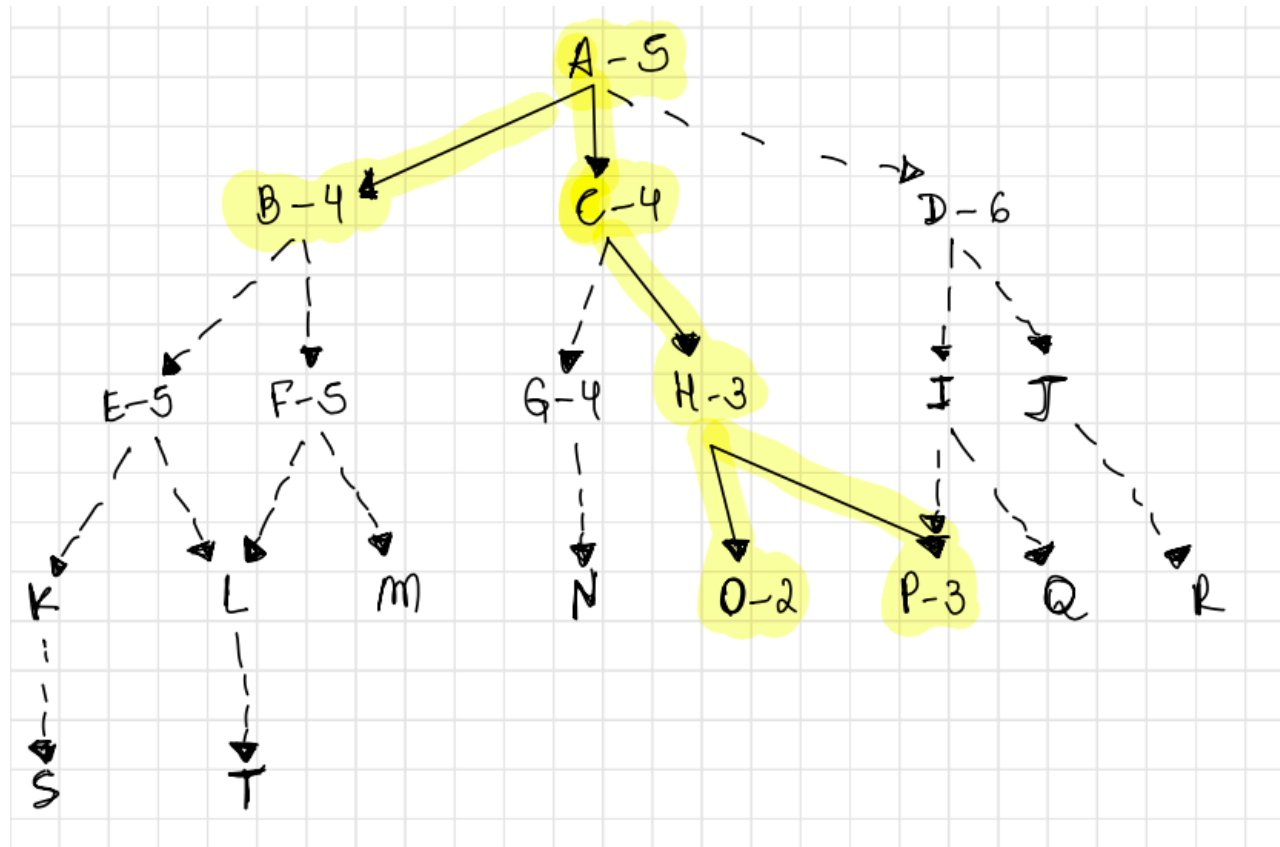
Busca pela Melhor Escolha



Exemplo:

Busca pela Melhor Escolha

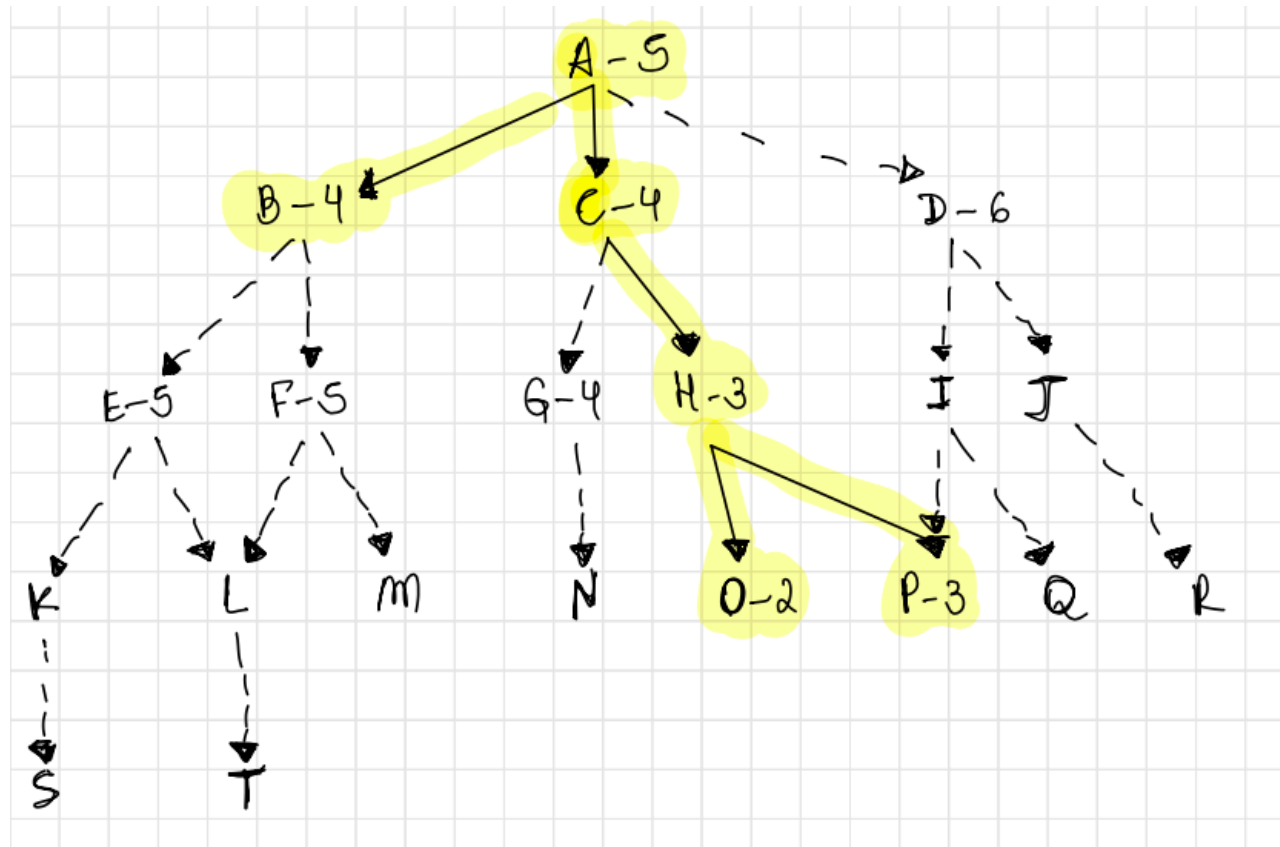
Exemplo:



Busca pela Melhor Escolha

Exemplo:

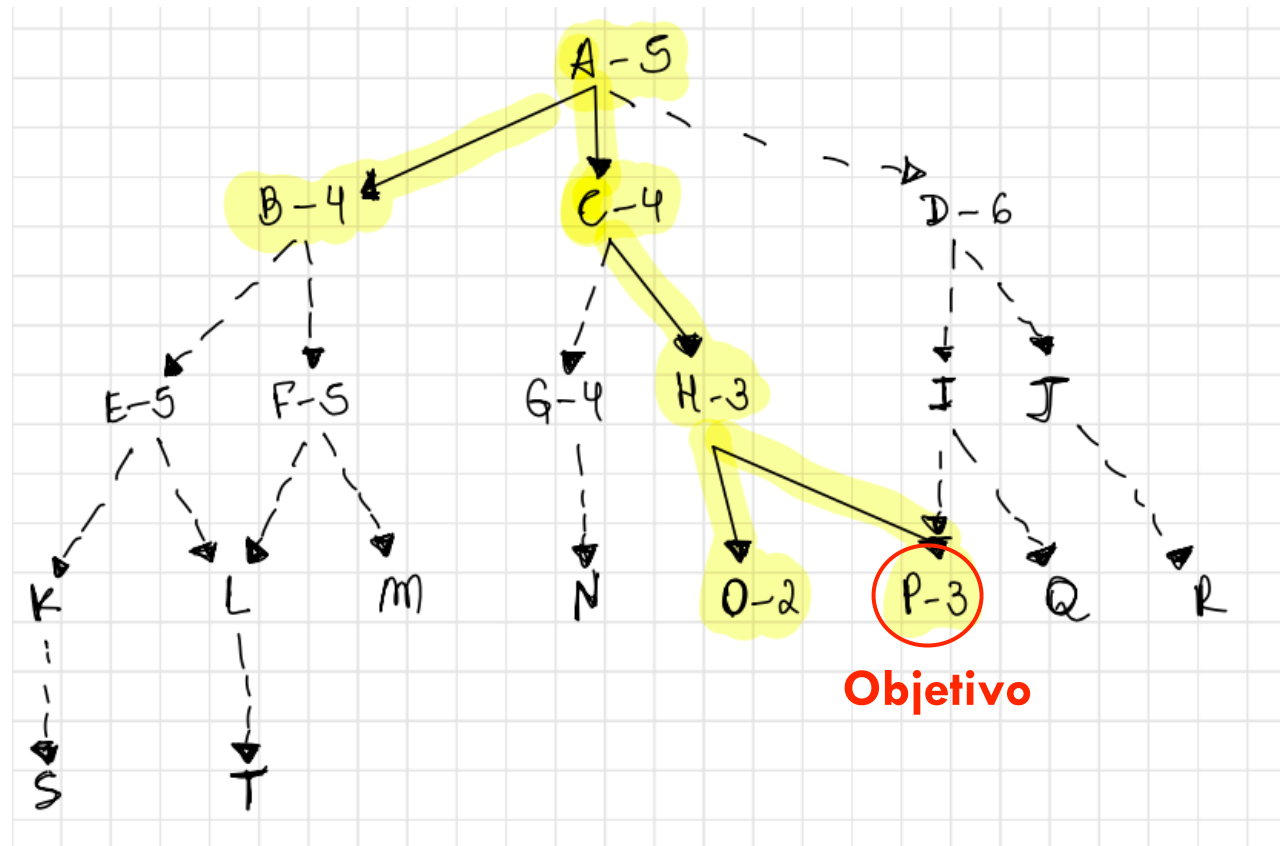
$H(x)$:
menores
valores
são
melhores



Busca pela Melhor Escolha

Exemplo:

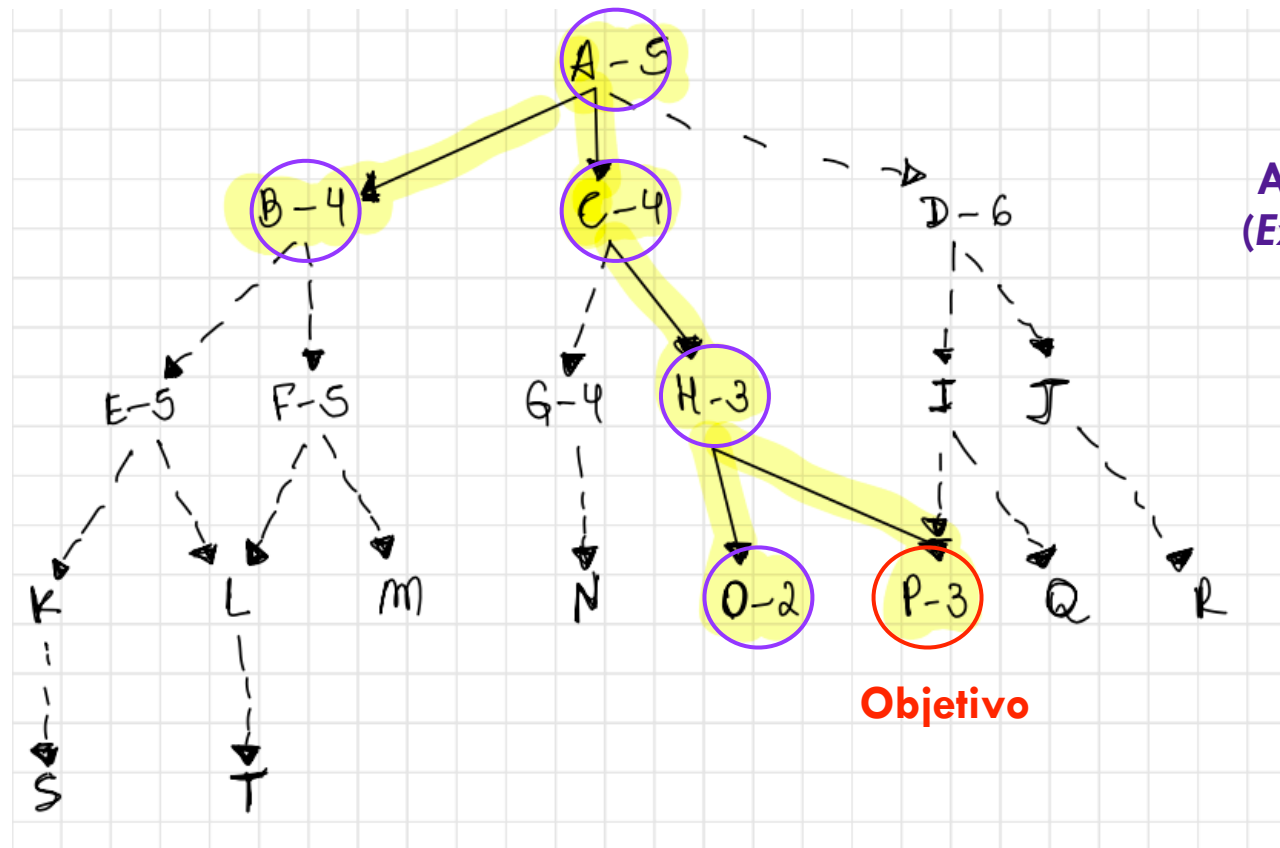
$H(x)$:
menores
valores
são
melhores



Busca pela Melhor Escolha

Exemplo:

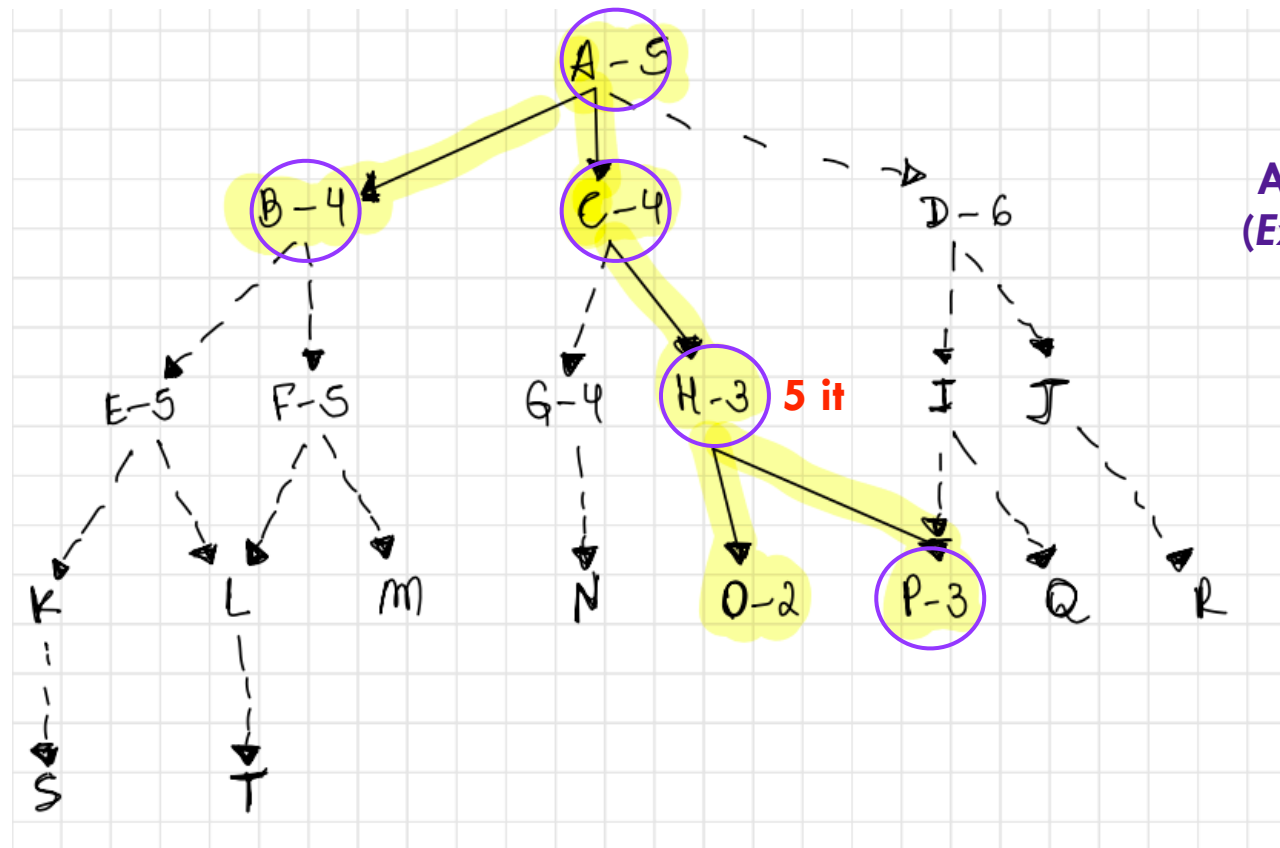
$H(x)$:
menores
valores
são
melhores



Busca pela Melhor Escolha

Exemplo:

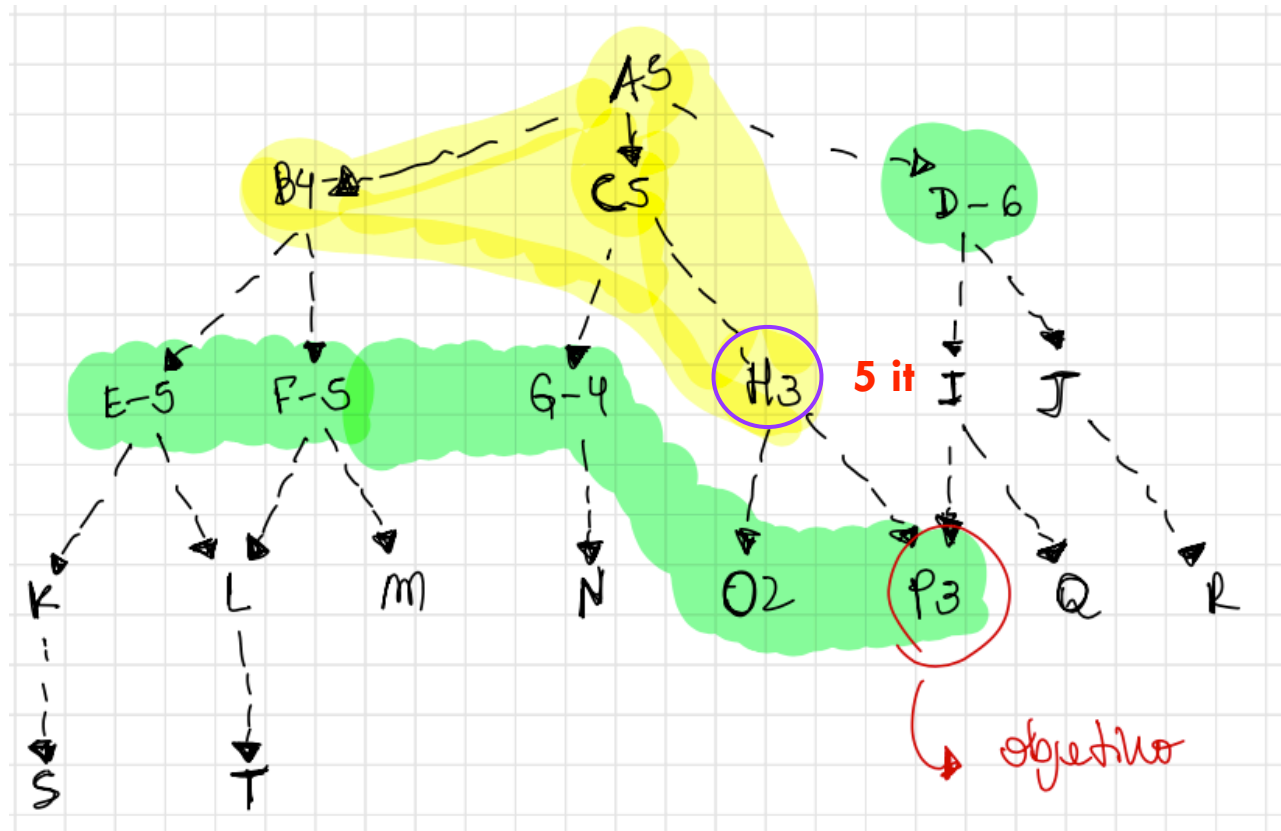
$H(x)$:
menores
valores
são
melhores



Estados
Avaliados
(Expandidos)

Busca pela Melhor Escolha

Exemplo:



● Estados ABERTOS

● Estados FECHADOS

Busca pela Melhor Escolha

Características:

- a cada iteração o algoritmo busca a melhor escolha, e retira da lista de **ABERTOS**
- estados duplicados não são retidos. Os caminhos podem ser atualizados, sempre com a melhor estimativa (heurística)
- **ABERTOS** funciona como uma **Fila de Prioridade**
- consegue se recuperar de possíveis erros que os outros algoritmos sofrem quando avaliam um máximo/mínimo local, retrocesso na lista **ABERTOS**

Roteiro



- 1 Introdução
- 2 Subida de Encosta
- 3 Busca pela Melhor Escolha
- 4 Exercícios
- 5 Referências

Exercícios

1) Implemente o algoritmo **Hill Climbing** para o quebra-cabeça de 8 peças.

2	8	3
1	6	4
7		5

estado
inicial

1	2	3
8		4
7	6	5

estado
objetivo

Exercícios

2) Implemente o algoritmos **Busca pela melhor escolha** para o quebra-cabeça de 8 peças.

2	8	3
1	6	4
7		5

estado
inicial

1	2	3
8		4
7	6	5

estado
objetivo


Exercícios



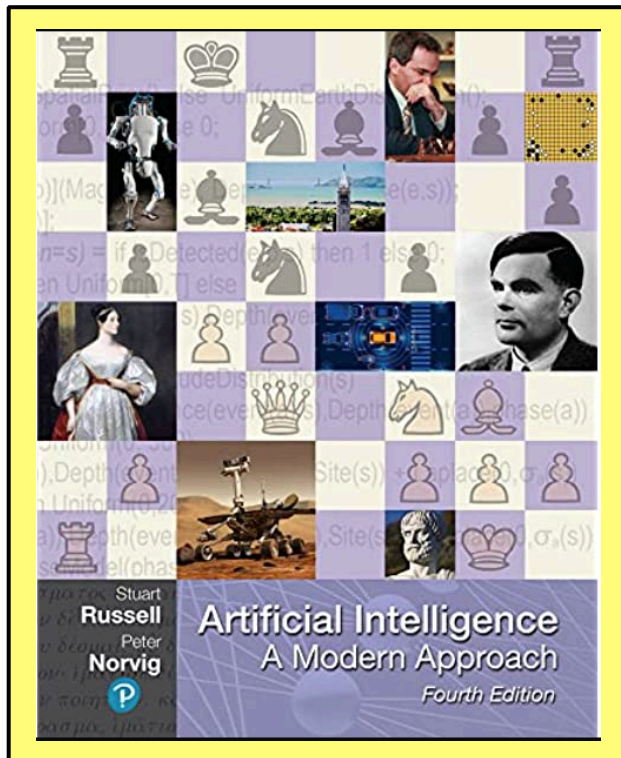
3) Realize testes nos seus algoritmos com diferentes estados iniciais e finais.

Roteiro

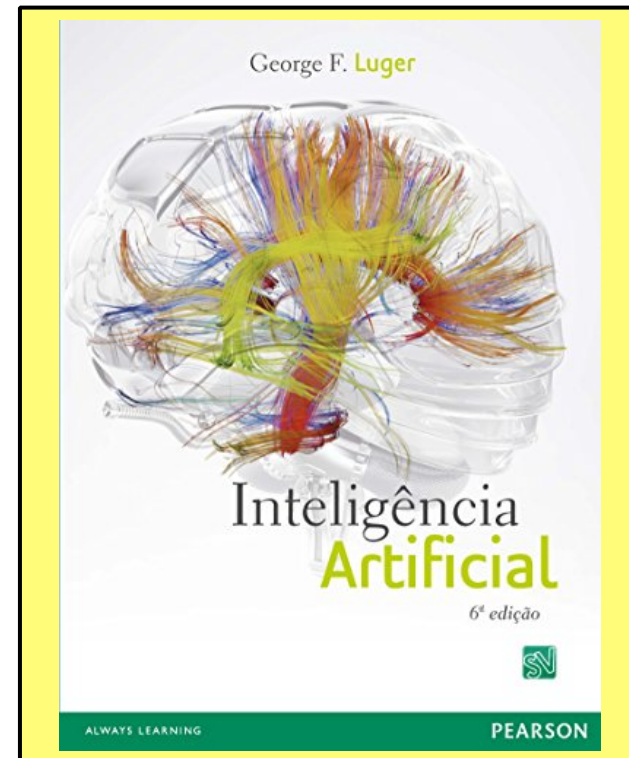


- 1 Introdução
 - 2 Subida de Encosta
 - 3 Busca pela Melhor Escolha
 - 4 Exercícios
 - 5 Referências
- 

Referências sugeridas



[Russel & Norvig, 2021]



[Luger, 2013]

Perguntas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br