Inteligência Artifical

812839 - Vinícius Miranda de Araújo

Lista de Exercícios 08

Questão 1

---

1. Algoritmo de Busca em Largura

- Ordem de Visitação = {A, B, C, D, E, F, G, H, I}
- Caminho Solução = {A, B, E, I}
- Não possui heurística

2. Algoritmo de Busca em Profundidade

- Ordem de Visitação = {A, B, D, E, H, I}
- Caminho Solução = {A, B, E, I}
- Não possui heurística

3. Custo Uniforme

- Ordem de Visitação = {A, C, B, E, F, G, D, K}
- Caminho Solução = {A, C, G, K}
- Não possui heurística

4. Algoritmo de Busca Gulosa

- Ordem de Visitação = {A, B, E, I}
- Caminho Solução = {A, B, E, I}
- A heurística é admissível

5. Algoritmo A*

- Ordem de Visitação = {A, B, E, C, G, K}
- Caminho Solução = {A, C, G, K}
- A heurística é admissível

Questão 2

---

1. Sim, a heurística de Manhattan é admissível. A distância de Manhattan calcula, para cada peça, o número de movimentos horizontais e verticais necessários para que ela alcance sua posição correta ignorando colisões com outras peças. Ela nunca superestima o custo real, porque o custo mínimo para mover uma peça é, no melhor caso, exatamente a quantidade de passos Manhattan. Portanto, como $h(n) \leq h^*(n)$ para todo n, a heurística é admissível.

2. Número de peças fora do lugar, semelhante a ideia de Distância Hamming. Nessa heurística seria contabilizada quantas peças estão fora de sua posição correta, sem considerar o espaço vazio.

> Exemplo: se 3 peças estão fora do lugar, $h(n) = 3$.

- Essa heurística seria admissível porque o número de peças fora do lugar representa o mínimo número de movimentos necessários para colocá-las no lugar, considerando apenas um movimento por peça.

Questão 3

Letra B

I. A primeira solução encontrada pela estratégia de busca em largura é a solução ótima.

> VERDADEIRO. Em problemas onde todas as ações têm o mesmo custo, a busca em largura (BFS) expande os nós por níveis (menor profundidade primeiro). Assim, a primeira solução encontrada será a que exige o menor número de ações, ou seja, uma solução ótima (de menor custo).

II. A primeira solução encontrada pela estratégia de busca em profundidade é a solução ótima.

> FALSO. A busca em profundidade (DFS) explora caminhos profundamente antes de explorar alternativas, podendo encontrar uma solução profunda e longe do ótimo. Ela não garante que a primeira solução encontrada será a melhor.

III. As estratégias de busca com informação usam funções heurísticas que, quando bem definidas, permitem melhorar a eficiência da busca.

> VERDADEIRO. Busca com informação (como A* e busca gulosa) utiliza funções heurísticas para orientar a busca em direção ao objetivo. Quando a heurística é bem escolhida (informativa e admissível), ela reduz o número de nós explorados, aumentando a eficiência da busca.

IV. A estratégia de busca gulosa é eficiente porque expande apenas os nós que estão no caminho da solução.

> FALSO. A busca gulosa prioriza nós com menor valor heurístico, mas não garante que esses nós estejam realmente no caminho da solução. Ela pode desviar e explorar caminhos errados se a heurística for mal definida. Portanto, não é verdade que ela expande apenas nós no caminho da solução.

Questão 4

Letra A

Utilizando o algoritmo de Busca em Largura e partindo do vértice A, a ordem de visitação é: {A, B, C, D, E, F}

Questão 5

Letra E

I. A estratégia de busca em largura encontra a solução ótima quando todos os operadores de mudança de estado têm o mesmo custo.

> VERDADEIRO. A busca em largura (BFS) expande os nós em ordem de profundidade. Quando todos os custos são iguais, o caminho com menos ações será o de menor custo, ou seja, a primeira solução encontrada será a ótima.

II. A estratégia de busca em profundidade sempre expande um menor número de nós que a estratégia de busca em largura, quando aplicadas ao mesmo problema.

> FALSO. A busca em profundidade pode até expandir menos nós em alguns casos, mas não é garantido que sempre o faça. Ela pode entrar em caminhos profundos desnecessariamente e explorar muitos nós inúteis antes de achar a solução, enquanto a BFS explora niveladamente.

III. A estratégia de busca heurística encontra sempre a solução de menor custo.

> FALSO. Depende da estratégia. Por exemplo, a busca gulosa não garante solução ótima. Apenas algoritmos como A* garantem solução ótima, desde que a heurística seja admissível e consistente.

IV. A estratégia de busca heurística expande um número de nós em geral menor que o algoritmo de busca em largura, mas não garante encontrar a solução ótima.

> VERDADEIRO. Isso descreve bem a busca gulosa: usa heurísticas para ser mais eficiente que a BFS, mas pode não encontrar a melhor solução, pois ignora o custo acumulado.

V. O algoritmo de busca heurística que utiliza uma função heurística admissível encontra a solução ótima.

> VERDADEIRO. Esse é o caso do A*: se a heurística é admissível (nunca superestima o custo real), então o A* encontra a solução ótima.

Questão 6

Letra A

a) a busca gulosa minimiza $h(n)$.

> VERDADEIRO. A busca gulosa escolhe o nó com menor h(n)

b) a busca A* minimiza $h(n)$.

> FALSO. A busca A* minimiza o custo total estimado f(n) = g(n) + h(n), não h(n).

c) a busca de custo uniforme minimiza $h(n)$.

> FALSO. A busca de custo uniforme não usa h(n), ela minimiza g(n).

d) a busca gulosa minimiza $h(n)$ somente se a heurística for admissível.

> FALSO. A busca gulosa sempre tenta minimizar h(n), independentemente de ser admissível. Mas não garante solução ótima.

d) a busca A* minimiza $h(n)$ somente se a heurística for admissível.

> FALSO. A busca A* minimiza o custo total estimado f(n) = g(n) + h(n), não h(n).

Questão 7

Letra B

a) $\exists n \rightarrow h(n) \leq h^r(n)$

> FALSO. Deve valer para todos os nós, não apenas um.

b) $\forall n \rightarrow h(n) \leq h^r(n)$

> VERDADEIRO. Definição correta de heurística admissível (nunca superestima o custo real).

c) $\exists n \rightarrow h(n) > h^r(n)$

> FALSO. Deve valer para todos os nós, não apenas um e contradiz a definição, não pode haver superestimação em nenhum caso.

d) $\exists n \rightarrow h(n) > h^r(n)$

> FALSO. Deve valer para todos os nós, não apenas um e contradiz a definição, não pode haver superestimação em nenhum caso.

e) $\exists n \rightarrow h(n) < h^r(n)$

> FALSO. Deve valer para todos os nós, não apenas um e contradiz a definição, não pode haver superestimação em nenhum caso.

Questão 8

Letra D

A ordem de visitação, utilizando o algoritmo de busca em largura, seria = {A, B, C, D, E, F}. Portanto, o caminho solução seria = {A, C, F}

Questão 9

$$f(n) = (2-w).g(n) + w.h(n)$$

Quando $w = 0$:

$$f(n) = (2-w).g(n) + w.h(n)$$
$$f(n) = (2-0).g(n) + 0.h(n)$$
$$f(n) = 2.g(n)$$

> Realiza a Busca Custo Uniforme.

Quando $w = 1$:

$$f(n) = (2-w).g(n) + w.h(n)$$
$$f(n) = (2-1).g(n) + 1.h(n)$$
$$f(n) = g(n) + h(n)$$

> Realiza a Busca A*.

Quando $w = 2$:

$$f(n) = (2-w).g(n) + w.h(n)$$

$$f(n) = (2-2).g(n) + 2.h(n)$$
$$f(n) = 2.h(n)$$

> Realiza a Busca Gulosa.

Questão 10

---

1. Em relação à busca A*:
    1. Ordem de Visitação:
        - h0 = {S, B, D, C, A, G}
        - h1 = {S, B, C, G}
        - h2 = {S, B, D, G}
    2. Caminho Solução:
        - h0 = {S, B, C, G}
        - h1 = {S, B, C, G}
        - h2 = {S, B, D, G}
    3. Todas as heurísticas são admissíveis.
2. Em relação à busca Gulosa:
    1. Ordem de Visitação:
        - h0 = {S, A, G}
        - h1 = {S, A, G}
        - h2 = {S, B, D, G}
    2. Caminho Solução:
        - h0 = {S, B, C, G}
        - h1 = {S, A, G}
        - h2 = {S, B, D, G}
3. Em relação à busca em Profundidade:
    1. Ordem de Visitação:
        - v = {S, A, G}
    2. Caminho Solução:
        - s = {S, A, G}
4. Em relação à busca em Largura:

1. Ordem de Visitação:
    - v = {S, A, B, G}
2. Caminho Solução:
    - s = {S, A, G}

Questão 11

Letra E

Fazendo a soma das peças que são diferentes entre todos:

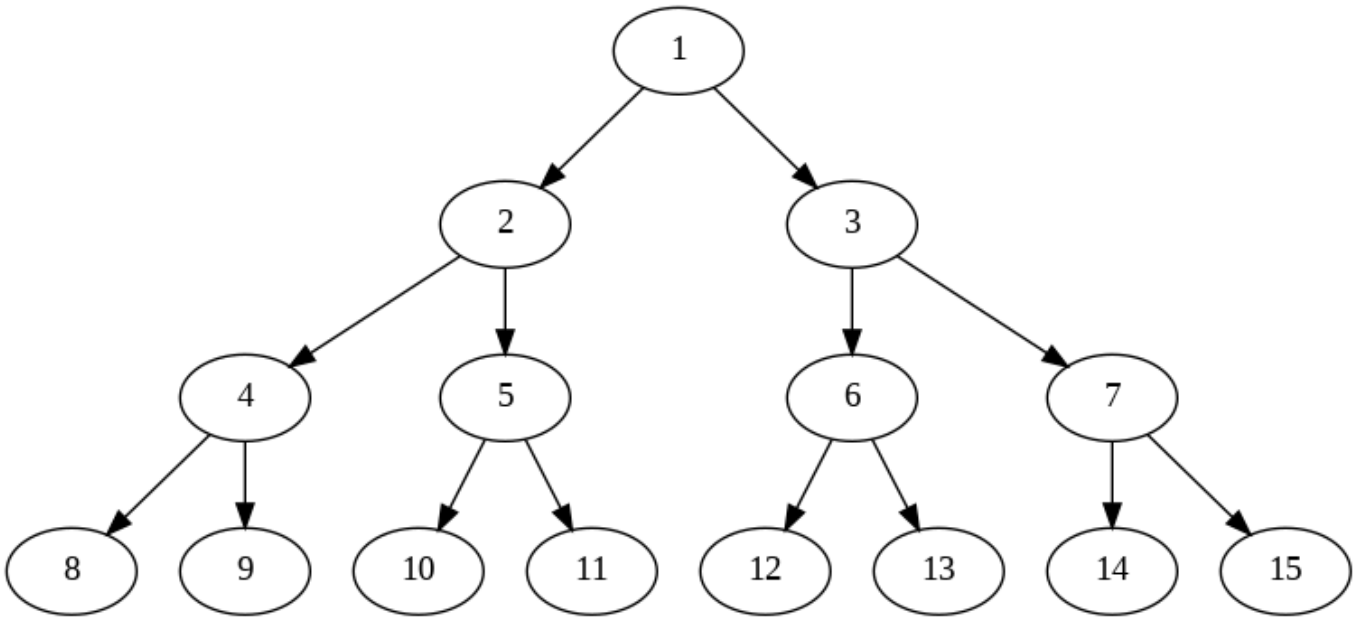
- E1:
    - d(1) = 3
    - d(7) = 2
    - d(8) = 2
    - soma = 7
- E2:
    - d(1) = 3
    - d(7) = 3
    - d(8) = 1
    - soma = 7
- E3:
    - d(1) = 3
    - d(6) = 1
    - d(7) = 2
    - d(8) = 1
    - soma = 7

> Portanto, ambas assertivas são falsas, dado que todos os 3 estados tem soma das distâncias iguais, assim é possível ir do E0 para qualquer E (E1, E2 ou E3).

Questão 12

---

a)



b) Odem de Visitação:

- Usando Busca por Extensão (Largura):

  v = {1, 2, 3, 4, 5, 6, 7, 8, 9 , 10, 11}

- Usando Busca em Profundidade Limitada (limite = 3):

  v = {1, 2, 4, 8, 9, 5, 10, 11}

- Usando Busca em Profundidade Iterativa:

  1: 1,

  2: 1, 2, 3,

  3: 1, 2, 4, 5, 3, 6, 7,

  4: 1, 2, 4, 8, 9, 5, 10, 11

  v = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11}

> Sobre Profundidade Iterativa e Limitada: https://slideplayer.com.br/slide/2751384/

Questão 13

---

1. Vantagens:
   - Ótimo (se a heurística for admissível).
   - Completo (se custo mínimo > 0).
   - Pode ser mais eficiente que BFS ou DFS.

- Com uma heurística bem desenhada, encontra soluções rápidas e boas.

2. Desvantagens:
    - Uso intensivo de memória: A* guarda todos os nós gerados.
    - Pode ser lento se a heurística for ruim (piora para BFS).
    - Difícil criar uma boa heurística em muitos problemas.

Questão 14

- IDA* (Iterative Deepening A*)
    - Combina A* com busca em profundidade iterativa.
    - Usa menos memória.
    - Ideal para grandes espaços de estados.
- RBFS (Recursive Best-First Search)
    - Usa recursão e backtracking, mantendo apenas os caminhos necessários na memória.
    - Menor uso de memória que A*.
- Memory-Bounded A* (como MA*, SMA*)
    - Limitam o uso de memória explicitamente.
    - Trocam nós menos promissores quando falta espaço.
- WA* (Weighted A*)
    - Usa $f(n) = g(n) + w * h(n)$, com $w > 1$.
    - Mais rápido, mas não garante solução ótima se $w \neq 1$.

FIM