Lista 4 IA

1-

Busca em Largura

1) A,B,C,D,E,F,G,H,I

2)Solução obtida: I

3)Não usa heurística

Busca em Profundidade

1)A,B,D,E,H,I

2) Solução obtida: I

3)Não usa heurística

Custo Uniforme

1)A,C,B,E,F,G,D,K

2) Solução obtida: K

3)Não usa heurística

Busca Gulosa

1)A,B,E,I

2) Solução obtida: I

3)Para a arvore a heurística é admissível, pois as estimativas fornecidas nunca são maiores que o custo real de cada caminho.

Algoritmo A*

1)A,B,E,I

2)Solução obtida: I

3) Para a arvore a heurística é admissível, pois as estimativas fornecidas nunca são maiores que o custo real de cada caminho.

2-

1. A heurística de Manhattan é admissível porque a soma das distâncias de todas as peças nunca será maior que o número mínimo de movimentos para resolver o puzzle.

2. A heurística de conflitos lineares é uma melhoria sobre a heurística de Manhattan. Ela leva em consideração não apenas a distância Manhattan, mas também situações onde duas peças estão na mesma linha ou coluna e se bloqueiam mutuamente, o que significa que elas não podem ser movidas diretamente para suas posições corretas sem que uma peça seja movida primeiro. Esses bloqueios exigem movimentos extras, que a heurística de Manhattan não captura.

3-

Resposta: a) I e III

II - Incorreto. A busca em profundidade pode encontrar uma solução, mas não garante que será a solução ótima, pois pode explorar um caminho mais profundo antes de explorar caminhos que levam a soluções melhores.

III – Incorreto, a busca gulosa use uma heurística para escolher o próximo nó a ser expandido, ela não garante que expanda apenas nós no caminho da solução e pode explorar caminhos subótimos.

4-

Resposta: a) A B C D E F

5-

Resposta: e) apenas as afirmativas I, IV e V são corretas.

II. Incorreto. A busca em profundidade pode expandir mais nós que a busca em largura, dependendo do problema, especialmente em espaços de estados muito grandes ou infinitos.

III. Incorreto. Apenas se a heurística for admissível e a busca for feita com o algoritmo A* a solução será a de menor custo.

6-

Resposta: a) a busca gulosa minimiza h(n).

A busca gulosa escolhe sempre o nó com o menor valor da função heurística, ignorando o custo acumulado.

7-

Resposta: b) $\forall n\ h(n) \leq h^r(n)$

Uma heurística h(x) é dita admissível quando nunca superestima o custo para atingir o objetivo a partir de um determinado nó x.

8-

Resposta: b) a b c d e f

9-

Quando w = 0

$f(n) = (2-0) \cdot g(n) + 0 \cdot h(n) = 2 \cdot g(n)$

Busca de custo uniforme, considera apenas o custo real g(n).

Quando w = 1

$f(n) = (2-1) \cdot g(n) + 1 \cdot h(n) = g(n) + h(n)$

Algoritmo A*, combina o custo real g(n) e a heurística h(n).

Quando w = 2

$f(n) = (2-2) \cdot g(n) + 2 \cdot h(n) = 2 \cdot h(n)$

Busca gulosa, considera apenas a heurística h(n).

10-

1.

h0

a) S, B, D, G

b) Caminho de custo 8

c) A heurística não é admissível pois não encontrou o melhor caminho

h1

a) S, B, C, G

b) Caminho de custo 6

c) A heurística é admissível pois foi encontrado o melhor caminho

h2

a) S, B, D, G

b) Caminho de custo 8

c) A heurística não é admissível pois não encontrou o melhor caminho

2.

h0

a) A heurística de todos os nós tem o mesmo valor, fazendo com que a seleção de nós para expandir seja aleatória.

b) Seguindo que caso empate o nó selecionado seja o da esquerda o caminho encontrado será:

S, A, G

h1

a) S, A, G

b) O caminho encontrado é S->A->G com custo total de 10

h2

a) S, B, D, G
b) O caminho encontrado foi S->B->D->G com custo total de 8

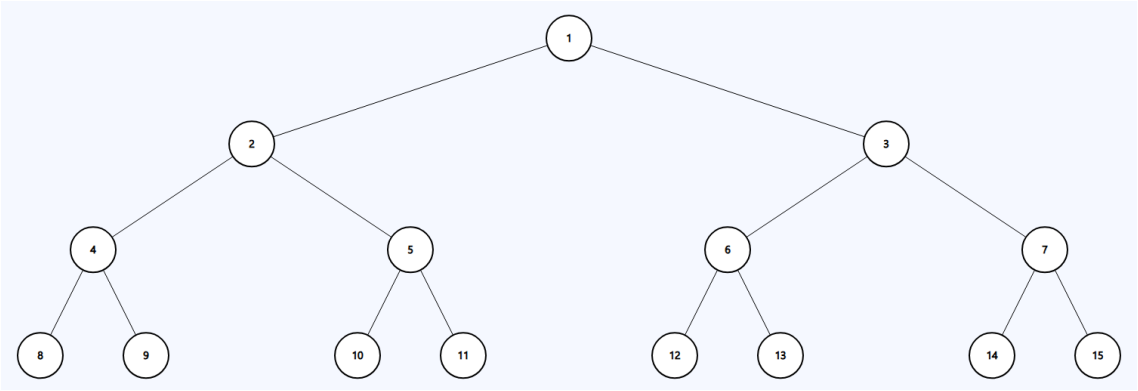
3.
a) Levando em consideração que o critério para escolher o nó é o mais à esquerda os nós expandidos são:
S, A, G
b) O caminho encontrado foi S->A->G com custo total de 10

4.
a) S, A, B, G
b) O caminho encontrado foi S->A->G com custo total de 10

11-
Resposta: c) A primeira asserção é uma proposição verdadeira, e a segunda é uma proposição falsa.

12-

a)



b)

ordens de visita:

Busca em Extensão: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

Busca em Profundidade Limitada (limite 3): 1, 2, 4, 8, 9, 5, 10, 11

Busca por Aprofundamento Iterativo: 1, 2, 4, 8, 9, 5, 10, 11

13-

Vantagens:

- Eficiente: O A* garante encontrar o caminho mais curto se a heurística utilizada for admissível.

-Heurística Personalizável: O desempenho pode ser ajustado por meio de diferentes funções heurísticas, permitindo uma adaptação ao problema específico.

- Busca Direcionada: Combina vantagens da busca em largura e profundidade, geralmente resultando em uma busca mais rápida.

Desvantagens:

- Alto Consumo de Memória: O A* pode exigir muita memória, principalmente em grafos grandes, o que pode ser um problema em ambientes com restrições.

- Desempenho Dependente da Heurística: A eficiência do A* varia significativamente com a escolha da heurística, e uma heurística inadequada pode resultar em tempos de execução muito mais longos.

14-

1. A com Heurísticas Adaptativas*
Utiliza uma heurística que se ajusta dinamicamente com base na experiência adquirida durante a execução do algoritmo. Isso pode ajudar a melhorar a eficiência em ambientes dinâmicos.
Melhoria: Permite que o algoritmo se adapte às características do espaço de busca, potencialmente reduzindo o tempo de execução.

2. A Bidirecional*
Realiza buscas simultâneas a partir do estado inicial e do estado objetivo, encontrando-se no meio.
Melhoria: Pode reduzir significativamente o tempo de execução e o espaço de memória necessário, especialmente em grandes espaços de busca, pois a área explorada é geralmente menor.

3. IDA (Iterative Deepening A)**
Combina a profundidade iterativa com a busca A*, limitando a profundidade de busca e aplicando a heurística em cada iteração.
Melhoria: Reduz o consumo de memória ao não armazenar todos os nós abertos, mantendo a optimalidade em relação ao A*.

15-

MAX pode ganhar se jogar de forma otimizada, a melhor jogada para MAX é retirar 2 palitos, assim restando 3 para MIN, que é uma posição que permite que MAX controle o jogo. Portanto, MAX deve sempre retirar um número de palitos que mantenha o total restante em um número ímpar (ou seja, 3, 1 ou 0). Isso impede que MIN ganhe.

16-

Resposta: a) 5.

A poda alfa-beta otimiza a busca na árvore minimax, ignorando nós que não afetam o resultado final. Ao avaliar a árvore fornecida, assim, 5 folhas não precisam ser visitadas para determinar a melhor jogada, otimizando o processo de busca.