Inteligência Artificial

1ºTrabalho

Jogo dos 15



David Fernandes - up201605791

Sérgio Araújo – up201608365

Introdução

Um problema de busca/procura consiste nos sucessivos passos necessários para chegar de um ponto A(estado inicial) a um ponto B(estado final ou objetivo).

De modo a tornar essa tarefa o mais simples possível são utilizados algoritmos que reduzem o tempo das mesmas.

Estes algoritmos são bastante úteis para resolver problemas no mundo real e como no nosso quotidiano somos deparados com um número extenso de problemas vários algoritmos foram criados (e novos vão surgindo) como a ‘Busca em Largura’, ‘Busca em Profundidade’, ‘Pesquisa Binária’, ‘Algoritmo de *Dijkstra*’ e muitos mais.

Estratégias de Procura

* Procura não guiada (*blind* - “cega”)
* Busca em Profundidade (**DFS** – ***Depth-First Search***): Algoritmo de pesquisa utilizado em grafos caracterizado por começar a pesquisa na raiz do grafo partindo para o seu primeiro filho, neto, bisneto e assim sucessivamente até chegar ao limite do grafo. Se a solução não for encontrada o processo é repetido mas partindo do seu segundo filho. A complexidade temporal do DFS é O(V+E) sendo V o número de vértices do grafo e E o número de arestas.
* Busca em Largura (**BFS - Breadth-First Search**): Semelhante ao DFS porém são expandidos todos os nós de um nível D antes de serem expandidos os de nível D+1. Ou seja, expande todos os nós filhos e só depois os nós netos. A complexidade temporal do BFS é O(V+E) sendo V o número de vértices do grafo e E o número de arestas.
* Busca Iterativa Limitada em Profundidade: Uma espécie de fusão entre DFS e BFS que começa a procura de solução com um limite de profundidade 0 e vai iterando esse limite utilizando DFS em cada passo.
* Procura guiada (que usa alguma heurística para orientar a procura)

**Heurística** – Função utilizada em parceria com os algoritmos que auxilia nas escolha de certos caminhos.

**Busca Gulosa (greedy)** – Algoritmo baseado no DFS que faz uso de uma heurística para escolha de um caminho em lugar de outro.

**A\*** - Utilizado para descobrir o caminho mais curto entre dois nós num grafo.

É completo e ótimo.

O segredo para o sucesso deste algoritmo é que combina informação relativa à distancia dos nós ao nó inicial (como o algoritmo de Dijkstra) com uma heurística.

Descrição do Problema

Este primeiro trabalho envolveu o jogo dos 15 que diz respeito a uma matriz com números de 1 a 15 dispostos de forma aleatória e um espaço em branco utilizado para mover as peças uma a uma na vertical ou horizontal de modo a chegar a uma configuração final previamente planeada.

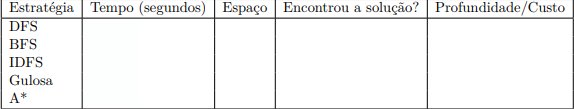
Descrição da Implementação

* A linguagem utilizada por o nosso grupo foi o Java pois é a linguagem com que estamos mais familiarizados.
* Criamos uma classe para o tabuleiro que integra um vetor para guardar os números, a posição do espaço em branco no tabuleiro, o movimento efetuado na peça, o tabuleiro pai, a distância de cada peça à sua posição final (*Manhattan Distance*) e a profundidade do tabuleiro na árvore de pesquisa.

Resultados

Configuração inicial: 6, 12, 0, 9, 14, 2, 5, 11, 7, 8, 4, 13, 3, 10, 1, 15

Configuração final: 14, 6, 12, 9, 7, 2, 5, 11, 8, 4, 13, 15, 3, 10, 1, 0



8

8

8

8

8

-

102 ms

142 ms

5 ms

48 ms

-

293

8

8

44

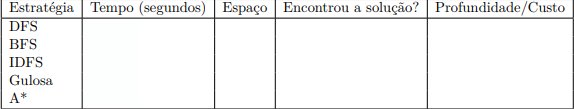
Não

Sim

Sim

Sim

Sim

Configuração inicial: 1, 2, 3, 4, 5, 0, 7, 8, 9, 6, 10, 12, 13, 14, 11, 15

1778 ms

1 ms

3 ms

4 ms

48 ms

40

31

4

4

13

Sim

Sim

Sim

Sim

Sim

4

4

4

4

4

Configuração final: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 0