Relatório - 8 - Puzzle

- Ivan Vilaça de assis - ivanassis07@gmail.com - 2021421931

- **Estruturas e modelagem do problema:**

Na modelagem deste problema, foi criada uma *class Node* que representa um estado do tabuleiro em um dado momento (*board)*. Os elementos desta classe possuem atributos para facilitar a reconstrução do caminho (*parent, children*) e a aplicação dos algoritmos (*cost, heuristics, depth).*

Na representação do board, optou-se por utilizar um *array* de 1 dimensão e efetuar as possíveis ações, como a expansão dos nós, com pequenos ajustes para ter o comportamento como o esperado na matriz do tabuleiro.

Dentro da classe, definimos as possíveis ações que são os movimento do bloco em branco no tabuleiro o qual pode ocorrer para cima, baixo esquerda e direita. Todas essas foram agrupadas em uma única função que é responsável por expandir um nó.

Nosso tabuleiro pode se iniciar em qualquer estado, mas a cada mudança dele utilizamos o teste objetivo presente na classe (*goalTest)* para checar se já estamos na configuração desejada.

Em relação aos algoritmos de busca, optamos por separá-los em arquivos diferentes. Nos que algoritmos que utilizam uma fila de prioridade (UCS, A*, Greedy), foi optado por usar uma fila normal e ordená-la de acordo com o custo requisitado por cada um destes algoritmos.

As heurísticas adotadas também ficaram separadas e, especificamente, na distância Manhattan foi implementada uma função auxiliar para pegar os índices do bloco na matriz do tabuleiro a partir do array de uma dimensão.

- Discussão sobre os algoritmos:

O grupo dos algoritmos de busca sem informação (BFS, IDS, UCS) consistem no que usam só dados de um estado que já estavam presentes na formulação inicial do problema.

Nesse grupo podemos destacar que o BFS e o UCS possuem complexidade temporal e espacial exponenciais. O BFS é um algoritmo completo, sempre termina, e ótimo, sempre encontra o melhor caminho, quando o custo é crescente com a profundidade. O UCS também é completo e ótimo.

O IDS é um algoritmo que junta os aspectos positivos do BFS, sendo completo e ótimo, com o aspecto positivo do DFS que é a complexidade espacial linear. A base dele é fazer várias buscas de profundidade aumentando o limite a busca a cada iteração até encontrar o objetivo.

O segundo grupo, o da busca com informação, usa não apenas a informação do modelo presente no problema, mas também uma heurística que consiste em uma estimativa de custo para se atingir o objetivo a partir de um estado.

O algoritmo Greedy se baseia apenas na heurística para escolher um novo estado, mas ele não é muito bom porque não é completo e nem ótimo, além de possuir uma complexidade de tempo e espaço exponenciais.

Já o algoritmo A*, considera o custo real dado no problema mais a estimativa advinda da heurística. Ele é melhor que o Greedy, pois caso a heurística seja admissível, a solução que ele fornecerá sempre será ótima e ele é completo. Sendo que admissível indica uma heurística a qual é sempre menor ou igual ao custo real.

- **Heurísticas escolhidas:**

Número de peças fora do lugar:

Nessa abordagem, pegamos um estado e somamos a quantidade de "azulejos" que estão fora do seu local em relação ao objetivo final.

Esta heurística é admissível porque todo bloco deslocado em relação ao seu objetivo terá que mover pelo menos uma casa para chegar em sua posição final. Então ela é um limite inferior em relação ao total de movimentos que serão necessários para resolver o problema.

Distância de Manhattan:

Nessa heurística, identificamos todos os "azulejos" que estão fora da sua posição final e somamos a distância de Manhattan destes blocos em suas posições atuais em relação às suas posições finais. Esta métrica utiliza a soma das distâncias horizontais e verticais de um bloco em relação a sua posição final.

Essa heurística é admissível porque é sempre menor que o custo real de movimentos para se chegar ao estado final, visto que cada peça só pode movimentar uma casa na horizontal ou vertical.

- **Comparação entre os algoritmos:**

	Tempo (s) do algoritmo por problema				
	B	I	U	A (misp.Tiles)	G (Manhattan)
Problema 3	0,0006	0,0033	0,0003	0,0004	0,0016
Problema 6	0,0035	0,0116	0,0014	0,0008	0,121
Problema 9	0,0199	0,1115	0,0102	0,0038	0,037
Problema 12	0,0831	0,2929	0,9189	0,0077	1,1544
Problema 15	0,4813	21,3621	28,2707	0,0562	34,3953
Problema 18	1,6196	4,2717	523,0844	0,5404	411,8774

	Quantidade de nós expandidos por cada algoritmo				
	B	I	U	A (misp.Tiles)	G (Manhattan)
Problema 3	6	33	16	3	16
Problema 6	42	198	111	6	111
Problema 9	226	607	642	41	642
Problema 12	1018	2748	2802	76	2802
Problema 15	5554	209167	14652	588	14652
Problema 18	18888	36539	49572	2058	49572

É possível notar como o A* se sobressai em relação aos demais com uma vantagem significativa de tempo, o que evidencia como a escolha de uma boa heurística pode promover o aumento de desempenho ao diminuir o espaço de busca significativamente como mostrado na segunda tabela.

Ao comparar o A* com o Greedy, também reparamos em como o uso guloso da heurística isoladamente não é uma boa decisão, tanto em relação ao tempo gasto quanto ao espaço de busca explorado.