UNIVERSIDADE ESTADUAL PAULISTA

TRABALHO DE INTELIGÊNCIA ARTIFICIAL

BUSCA CLÁSSICA E BUSCA HEURÍSTICA - RELATÓRIO

Autor: Gabriel Passos de Oliveira

Curso: Ciências da Computação

Professor responsável: Emilio Bergamin Júnior

1. Busca A* com a heurística hfora e com a heurística hM (Gabriel Passos de Oliveira)

- **Objetivo**

O código tem como objetivo resolver o quebra-cabeça de 8 peças, utilizando o algoritmo A* (A-star), uma técnica de busca informada que utiliza heurísticas para encontrar o caminho mais eficiente para alcançar o estado objetivo.

- **Estrutura de Dados**

Estado: Uma estrutura que representa o estado atual do quebra-cabeça. Possui um array de inteiros blocos que armazena as posições dos números no quebra-cabeça.

Nó (No): Uma estrutura que contém informações sobre um estado específico, incluindo o estado em si, o nó pai, o custo atual, as heurísticas locais e globais, e o custo total (f).

- **Funções Principais:**

1. criarEstado: Aloca memória para um novo estado e inicializa seus blocos com os valores fornecidos.
2. destruirEstado: Libera a memória alocada para um estado.
3. copiarEstado: Cria uma cópia de um estado existente.
4. findBlank: Encontra a posição do espaço em branco (representado por 0) no quebra-cabeça.
5. isObjetivo: Verifica se o estado atual é o estado objetivo.
6. distanciaManhattan: Calcula a distância de Manhattan entre a posição atual de um bloco e sua posição final.
7. heuristicaLocal: Calcula a heurística local (distância de Manhattan total).
8. heuristicaFora: Calcula a heurística fora-de-linha (número de blocos fora do lugar).
    a. swap: Troca os valores de duas variáveis inteiras.
9. gerarFilhos: Gera os possíveis estados filhos a partir de um estado pai, movendo o espaço em branco para cima, baixo, esquerda ou direita.
10. liberarNos: Libera a memória alocada para uma matriz de nós.
11. jaVisitado: Verifica se um estado já foi visitado anteriormente.
12. aEstrela: Implementa o algoritmo A* para resolver o quebra-cabeça, utilizando as funções auxiliares definidas anteriormente.

- **Funcionamento do Algoritmo:**

O algoritmo A* inicia com um estado inicial fornecido. Ele mantém uma lista aberta de nós a serem explorados, priorizando aqueles com menor custo total (f). O algoritmo então expande o nó atual gerando seus estados filhos, calculando seus custos e heurísticas, e os adicionando à lista aberta se eles ainda não foram visitados. Isso é repetido até que o estado objetivo seja alcançado ou não haja mais nós na lista aberta para explorar.

- **Resultados e Saída:**

O código imprime a solução encontrada, se existir, mostrando os passos necessários para chegar ao estado objetivo. Se o jogo não for solucionável, uma mensagem indicando isso é exibida.

- **Tratamento de Erros e Liberação de Memória:**

O código possui tratamento de erros para falhas na alocação de memória e libera adequadamente toda a memória alocada no final da execução.

- **Conclusão:**

O código implementa com sucesso o algoritmo A* para resolver o jogo do quebra-cabeça 8. Ele é capaz de encontrar uma solução para um estado inicial fornecido e exibir os passos necessários para alcançá-la. A implementação é eficiente e robusta, com tratamento adequado de erros e liberação de memória, garantindo uma execução segura e confiável.