
**UNIVERSIDADE FEDERAL DO AMAZONAS**

**INSTITUTO DE COMPUTAÇÃO**

KEREN GUIMARÃES GÓES

LUCAS DE SOUZA BASTOS

# QUEBRA-CABEÇAS DE 8 PEÇAS (PUZZLE – 8)

MANAUS

2023

## DESCRIÇÃO DO PROBLEMA

O problema escolhido foi o quebra-cabeças de 8 peças, mais conhecido como 8-puzzle. Esse quebra-cabeças consiste em um jogo que embaralha números de 1 a 8 e os coloca em uma matriz 3x3 com uma posição vazia para realizar as movimentações. O objetivo do jogo é organizar a matriz de forma que os elementos fiquem dispostos em ordem crescente e a posição vazia seja o primeiro elemento da matriz

## PROCEDIMENTO DE BUSCA

O algoritmo utilizado para a busca foi o A*. O algoritmo A* é um algoritmo de busca informado que pode ser usado para resolver problemas de busca de caminho em um grafo ou rede. O objetivo do algoritmo é encontrar o caminho mais curto entre um estado inicial e um estado final, levando em consideração o custo do caminho já percorrido e uma estimativa do custo restante para alcançar o objetivo.

O algoritmo A* utiliza uma função heurística para estimar o custo restante para alcançar o objetivo a partir de qualquer estado do quebra-cabeças. No caso do 8-puzzle, uma heurística comumente utilizada é a distância de Manhattan (heurística utilizada no algoritmo desse trabalho), que mede a distância entre cada peça e sua posição final desejada, somando as distâncias horizontais e verticais. Quanto menor a distância de Manhattan, mais perto do objetivo o estado está.

O algoritmo começa com o estado inicial do puzzle e uma lista de nós abertos, que inclui o estado inicial. Para cada nó aberto, o algoritmo expande o nó para gerar seus nós filhos, que representam todos os estados possíveis que podem ser alcançados a partir desse estado. Em seguida, ele avalia cada filho, calculando seu custo total (custo atual mais estimativa de custo restante) e adicionando-o à lista de nós abertos.

O algoritmo então seleciona o nó aberto com o menor custo total e marca-o como visitado. Esse processo é repetido até que o nó final (ou objetivo) seja encontrado ou até que a lista de nós abertos esteja vazia, o que significa que não há mais estados a serem avaliados e que não há solução possível.

Se o nó final é encontrado, o algoritmo retorna o caminho percorrido para alcançar esse nó. Esse caminho pode ser construído retroativamente seguindo os nós pai de cada nó, começando pelo nó final e terminando no nó inicial.

Em resumo, o algoritmo A* é uma abordagem eficaz para encontrar soluções para o 8 puzzle, pois leva em consideração o custo total para alcançar o objetivo e usa uma heurística para estimar o custo restante. Isso permite que ele explore eficientemente o espaço de busca para encontrar a solução mais rápida possível.

## DISCUSSÃO SOBRE A ESCOLHA DA INTERFACE

Para a interface do jogo Puzzle tentamos implementar a interface tkinter, porém tivemos algumas dificuldades ao usá-la e como não encontramos soluções suficientes que sanassem nossas dúvidas trocamos para a interface pygame. O pygame é uma interface nativa Python

própria para jogos, como é bastante conhecida na comunidade Python, há bastante material sobre ela, além de ter uma documentação de fácil entendimento. Tivemos alguns problemas com ela, porém era quando íamos importar a interface no VisualStudioCode, que foi resolvido uns tempos depois. Mas no geral, auso dessa interface foi uma ótima escolha.

## INSTRUÇÕES PARA INSTALAÇÃO

Caso na máquina que será usado o programa ainda não tiver instalada a linguagem Python, é necessário realizar essa ação no site oficial Pyhton: https://www.python.org/downloads/. Ele pode ser usado em qualquer ambiente virtual que aceite linguagem Python com interface, indicamos o VisualStudioCode, que pode ser instalado no site: https://code.visualstudio.com/download. Também é preciso fazer a importação da interface pygame, através da instrução -pip install pygame. Por fim, para jogar o Puzzle, precisa fazer o download a pasta com os arquivos main.py, mod.py, sprites.py, PriorityQueue.py, PuzzleFinal.py, PuzzleState.py. Será necessário executar apenas o arquivo main.py. Após executar o arquivo, o jogador encontra a tela no estado final e tem a opção de embaralhar as “cartas”, resolver manualmente ou resolver automaticamente.