**Heurísticas em Algoritmos de Busca para o Quebra-Cabeça de 8 Peças:**

**Uma Avaliação Experimental**

Bruno Braga Guimarães Alves  
PUC Minas  
 Belo Horizonte

Victor Monteiro  
PUC Minas  
 Belo Horizonte

**1. Introdução**

Este trabalho explora heurísticas em algoritmos de busca para resolver o Quebra-Cabeça de 8 Peças. A implementação utiliza a classe Puzzle com heurísticas como Distância de Manhattan, Zero e Euclidiana, integradas ao algoritmo A\*. Além disso, estratégias de busca em profundidade (DFS) e busca em largura (BFS) são comparadas. O estudo visa avaliar o desempenho dessas abordagens por meio de métricas de espaço e tempo, fornecendo uma percepção sobre o impacto das heurísticas na eficácia dos algoritmos de busca.

**2. Algoritmos Escolhidos**

Na busca por soluções eficientes para o Quebra-Cabeça de 8 Peças, foram selecionados diferentes algoritmos, sendo o destaque principal o algoritmo A\* (A-star). Este algoritmo é uma escolha frequente devido à sua capacidade de combinar efetivamente a busca de custo uniforme com heurísticas informadas, proporcionando uma busca informada e eficiente no espaço de estados. A\* (A-star): O algoritmo A\* é uma técnica de busca que utiliza uma função de avaliação composta pela soma do custo do caminho percorrido até o estado atual (custo g) e uma heurística que estima o custo do caminho do estado atual ao estado objetivo (h). A função de avaliação é dada por f(n) = g(n) + h(n). Isso permite ao A\* priorizar estados que têm menor custo total esperado, tornando-o especialmente adequado para problemas de otimização.

**2.1.Heurísticas Aplicadas ao A\***

Distância de Manhattan: Esta heurística mede a distância entre as posições atual e objetivo de cada peça no Quebra-Cabeça, calculando a soma das distâncias horizontais e verticais. É uma heurística eficaz para avaliar a proximidade de um estado ao objetivo.

Heurística Euclidiana: A heurística Euclidiana calcula a distância euclidiana entre as posições atual e objetivo de cada peça. Essa abordagem leva em consideração tanto as distâncias horizontais quanto verticais, oferecendo uma perspectiva mais geométrica da proximidade ao estado objetivo.

Ao explorar o A\* e suas heurísticas, este estudo visa analisar como diferentes métodos de avaliação de estados impactam o desempenho do algoritmo na resolução do Quebra-Cabeça de 8 Peças. A avaliação experimental se concentrará em medir o espaço e o tempo necessários para encontrar soluções ótimas, proporcionando uma visão aprofundada do comportamento do A\* em diferentes cenários.

**3. Desenvolvimento dos Algoritmos**

Os algoritmos foram desenvolvidos utilizando a linguagem de programação Python, com destaque para a implementação do algoritmo A\* e suas heurísticas no contexto do Quebra-Cabeça de 8 Peças. Abaixo está um resumo do desenvolvimento:

Classe Puzzle: A classe Puzzle foi criada para representar o estado do Quebra-Cabeça, armazenando informações como o estado atual, o pai, a ação que levou a esse estado, o custo do caminho percorrido até o estado atual e a necessidade de uma heurística. Métodos como generate\_heuristic, goal\_test, find\_legal\_actions, generate\_child, e find\_solution foram implementados para realizar operações específicas, como a geração de heurísticas, verificação de estado objetivo, geração de filhos válidos e encontrar a solução.

Algoritmo A\* *(A-star*):\* A função *Astar\_search* foi desenvolvida para realizar a busca A\* com base em uma heurística específica (por exemplo, Distância de Manhattan ou Heurística Euclidiana). O algoritmo utiliza uma fila de prioridade (implementada como uma PriorityQueue) para explorar os estados de forma informada, priorizando estados com menor custo total esperado.

Outras Estratégias de Busca: Além do A\*, foram implementadas estratégias de busca em profundidade *(depth\_first\_search*), busca uniforme e busca em largura *(breadth\_first\_search)*. Cada uma dessas estratégias utiliza abordagens distintas para explorar o espaço de estados em busca da solução. Função Principal *(Main)*: Foi desenvolvida para executar o programa. Ela solicita ao usuário a entrada do estado inicial do Quebra-Cabeça, inicializa a classe Puzzle com esse estado e, em seguida, realiza a busca A\* com diferentes heurísticas (Manhattan e Euclidiana). As métricas de espaço e tempo são registradas e apresentadas ao usuário para análise comparativa.

**4. Testagem**

Dado a implementação das buscas já citadas, aqui vão os resultados das buscas para 10 estados iniciais do Quebra Cabeça de 8 Peças:

| 4 5 6  1 2 3  0 7 8 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 1754 | 1754 | 1754 | 9303 | 158539 |
| Tempo | 0.0327 | 0.0437 | 0.0219 | 0.1030 | 1.085 |

| 5 4 0  6 1 2  7 3 8 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 4587 | 4587 | 4587 | 75992 | 302449 |
| Tempo | 0.0945 | 0.1238 | 0.0625 | 0.6463 | 2.543 |

| 4 0 1  2 3 5  7 8 6 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 1501 | 1501 | 1501 | 65271 | 76390 |
| Tempo | 0.0527 | 0.0672 | 0.0345 | 0.7576 | 0.5856 |

| 8 7 6  0 2 5  1 4 3 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 1567 | 1567 | 1567 | 8409 | 154664 |
| Tempo | 0.0656 | 0.0692 | 0.0429 | 0.0781 | 28.340 |

| 0 8 4  5 7 1  2 3 6 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 1878 | 1878 | 1878 | 175574 | 191458 |
| Tempo | 0.0381 | 0.0464 | 0.0326 | 17.490 | 15.664 |

| 8 0 1  7 4 3  2 5 6 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 536 | 536 | 536 | 44406 | 33928 |
| Tempo | 0.0189 | 0.0214 | 0.0157 | 0.4005 | 0.6805 |

| 7 6 5  1 2 0  3 4 8 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 22424 | 22424 | 22424 | 27242 | 429011 |
| Tempo | 1.050 | 0.3736 | 0.9710 | 0.5045 | 3.557 |

| 4 5 1  2 3 8  7 6 0 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 887 | 887 | 887 | 1723 | 82160 |
| Tempo | 0.0182 | 0.0106 | 0.0227 | 0.0097 | 0.831 |

| 1 0 8  3 7 4  6 5 2 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 1439 | 1439 | 1439 | 182434 | 70345 |
| Tempo | 0.08597 | 0.03314 | 0.0191 | 1.244 | 0.6403 |

| 3 2 1  0 8 7  6 5 4 |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Manhattan | Euclidiana | Custo Uniforme | Profundidade | Largura |
| Num Jogadas | 1468 | 1468 | 1468 | 14219 | 76999 |
| Tempo | 0.0290 | 0.0356 | 0.0203 | 0.4742 | 0.7726 |

**5. Conclusão**

A implementação do algoritmo A\* com heurísticas como Distância de Manhattan e Euclidiana demonstrou um desempenho superior em comparação com outras estratégias, como busca em profundidade (DFS) e busca em largura (BFS).

Observamos que o algoritmo A\* se destaca devido à sua capacidade de combinar busca informada e eficiente no espaço de estados, priorizando estados com menor custo total esperado. As heurísticas utilizadas, desempenharam um papel fundamental na melhoria do desempenho do A\*, proporcionando uma avaliação mais precisa da proximidade ao estado objetivo.

Os resultados obtidos demonstram que, em termos de número de jogadas e tempo de execução, o algoritmo A\* com heurística de Distância de Manhattan tende a encontrar soluções ótimas de forma mais eficiente em comparação com a Heurística Euclidiana em alguns casos específicos. Contudo, a Heurística Euclidiana mostrou-se vantajosa em outros cenários, evidenciando a importância de selecionar a heurística mais adequada para um problema específico.

Além disso, as estratégias de busca em profundidade e busca em largura revelaram limitações significativas para resolver o Quebra-Cabeça de 8 Peças, resultando em um número substancialmente maior de jogadas e tempos de execução mais longos em comparação com o algoritmo A\*.

Portanto, com base nos resultados experimentais, é evidente que o algoritmo A\*, especialmente quando combinado com heurísticas eficazes, como a Distância de Manhattan, representa uma abordagem promissora e eficiente para resolver o Quebra-Cabeça de 8 Peças, proporcionando um equilíbrio entre otimização de recursos computacionais e busca por soluções ótimas.

**6. Código Desenvolvido**

[Codigo\_Desenvolvido](https://colab.research.google.com/drive/15uokPSNU2t7QsL_UWIhYwDLrlvsiBW7f?usp=sharing#scrollTo=pPwXAGCH4xRm)