Lista 9 - Implementação de Puzzle
Inteligência Artificial

Caroline Freitas Alvernaz

[1]Instituto de Ciências Exatas e Informática – PUC Minas

1. Proposta

Desenvolvimento de um quebra-cabeça de 8 peças dispostas em um tabuleiro 3x3 em que deve-se movimentar uma peça por vez a fim de reorganizar uma imagem. Para isso, devem ser aplicados três algoritmos de busca para que o puzzle realize a solução automática e apresente a quantidade de movimentos usados por cada método e o tempo necessário para executar cada um.

2. Métodos utilizados

Para este trabalho, foram escolhidos os algoritmos Busca Gulosa, Busca Uniforme e A*.

2.1. Busca Gulosa

A busca gulosa, ou best-first greedy search, expande os nós de menor custo, avaliando-os de acordo com a função heurística apenas e, portanto, não garante a solução ótima e não é completa, porém é um algoritmo rápido se a heurística for boa. A heurística utilizada para a construção do puzzle foi a de Manhattan, ou seja, calcula a diferença entre a posição (considerando o tabuleiro como uma matriz 3x3) correta da peça e a posição original, garantindo sempre uma heurística admissível, pois o algoritmo de Manhattan dá a melhor solução independente de como as peças do tabuleiro estão dispostas, portanto a heurística será sempre menor ou igual ao custo real, ou seja, não superestima o custo. A função do algoritmo é descrita por: **f(n) = h(n)**, sendo n o número de vértices e h(n) a heurística, neste caso de Manhattan, atribuída a cada nó.

2.2. Busca Uniforme

A busca uniforme utiliza-se do algoritmo de Dijkstra para encontrar a solução: utiliza uma fila de prioridade que organiza os custos acumulados do caminho desde o início do percurso até o vértice alcançado. Assim, analisa-se apenas os pesos das arestas g(n), tendo sua função definida por **f(n) = g(n)**. Assim, diferente da busca gulosa, é garantida a busca ótima e completa, pois todos os custos reais são analisados e, por isso, em contrapartida, se torna um algoritmo mais lento que a busca gulosa.

2.3. Busca A*

Por fim, o algoritmo A* utiliza-se das duas métricas: custo real e estimativa de custo e, portanto, sua função é descrita por **f(n) = h(n) + g(n)**. O A* também garante a solução ótima e a busca completa, quando a heurística é admissível. Para este algoritmo, também foi utilizada a distância de Manhattan para cálculo da heurística.

3. Puzzle

O puzzle foi desenvolvido em linguagem HTML e possui como interface inicial a figura abaixo. Link para acesso ao código: Puzzle-GitHub.

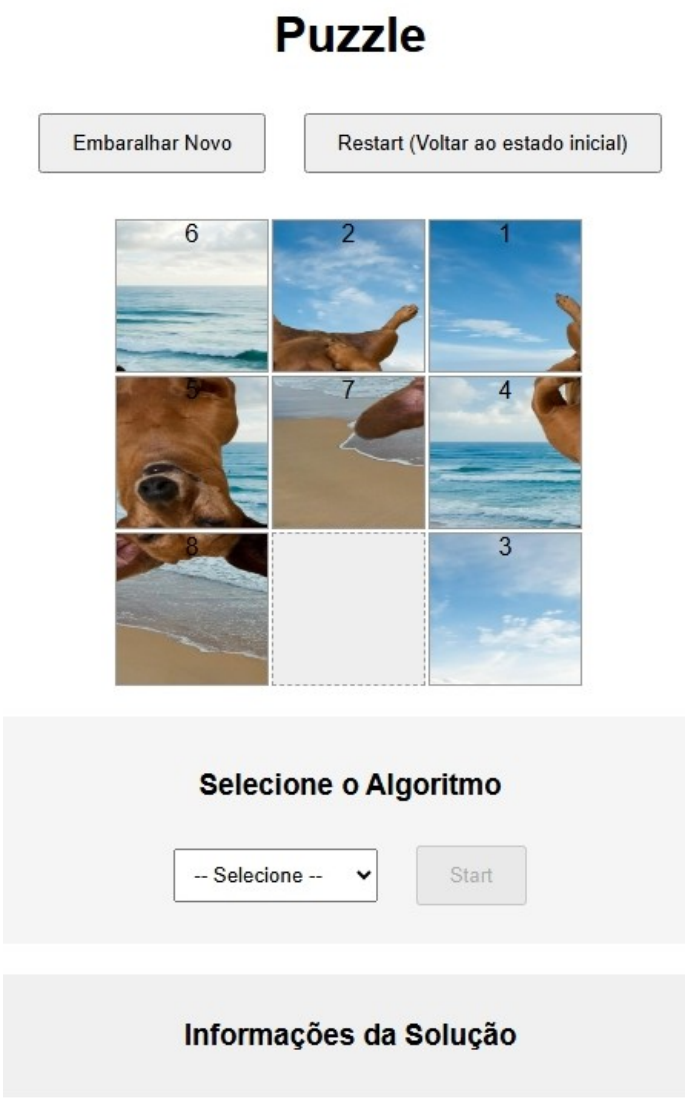

Figura 1. Interface inicial do puzzle.

Para iniciar o puzzle, deve-se selecionar o algoritmo a ser utilizado para chegar na solução ou embaralhar o puzzle e, após, fazer a seleção do algoritmo e depois clicar em ”Start”. Após, o puzzle começa a realizar sua resolução de acordo com o algoritmo escolhido e abaixo aparecem quantos movimentos foram necessários e o tempo gasto de processamento, conforme mostra a imagem abaixo.

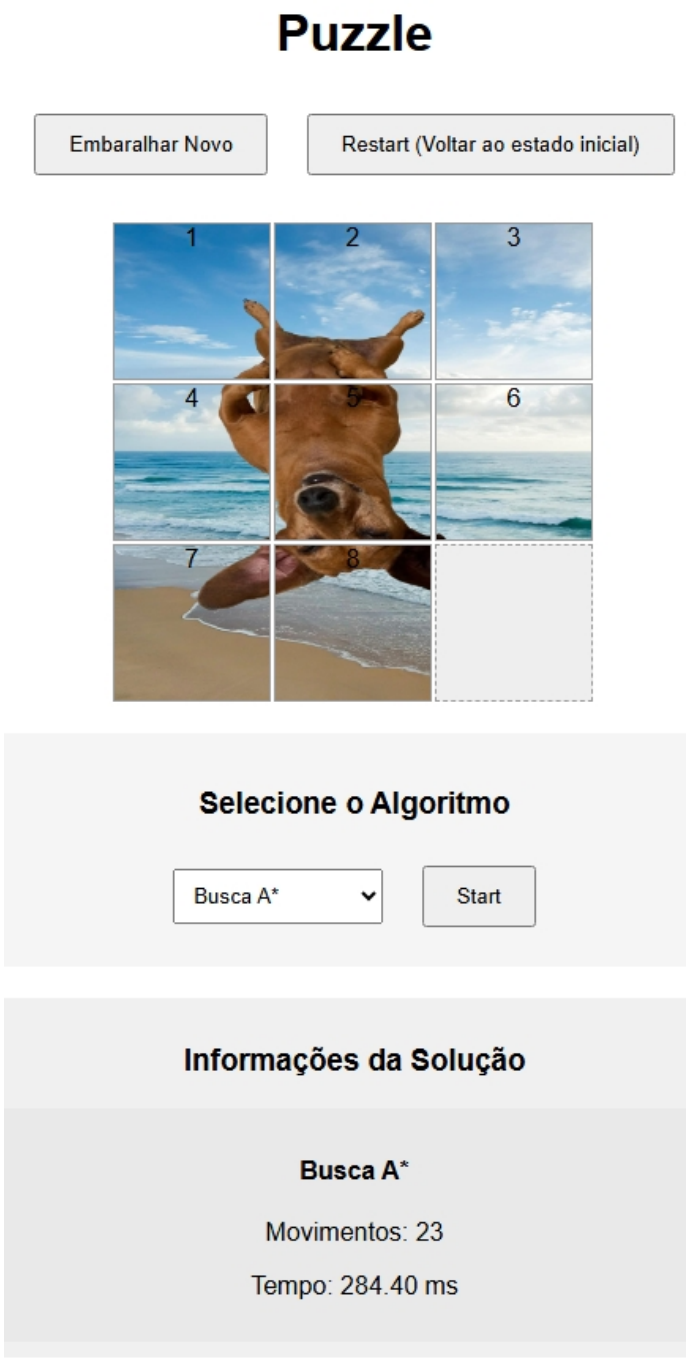

Figura 2. Interface com o puzzle resolvido via Busca A*.

Ainda, para que se possa ser feita a comparação entre os métodos, deve-se clicar no botão ”Restart (Voltar ao estado inicial)”para que o puzzle volte a ficar embaralhado como antes de iniciar a organização com o algoritmo escolhido. Após, é possível escolher um novo método a ser aplicado com o puzzle embaralhado da mesma forma de antes.

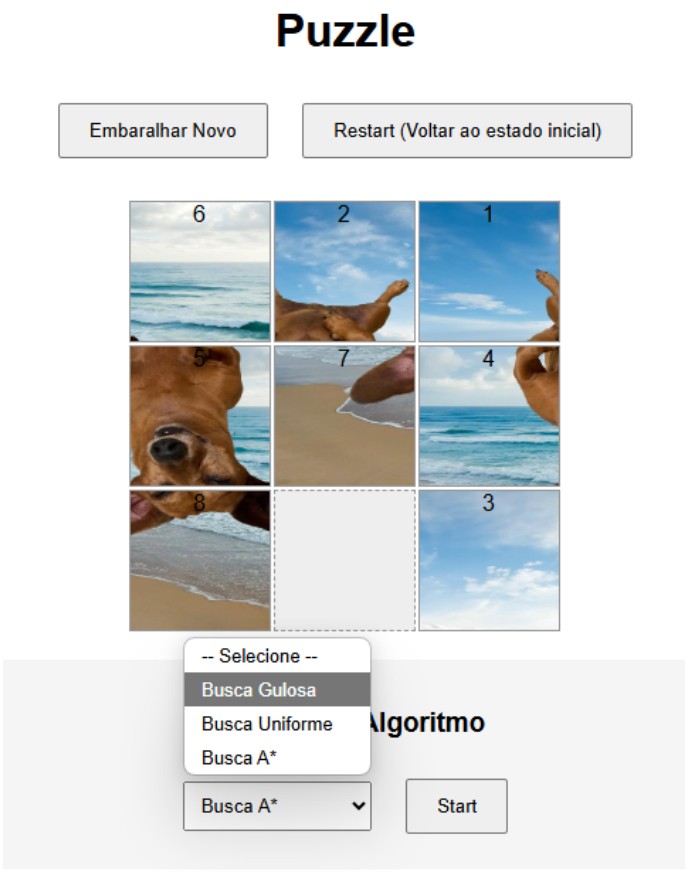

Figura 3. Interface para escolher a próxima busca a ser feita.

4. Resultados

Após a utilização do puzzle e realização de alguns testes, foi possível notar que o algoritmo de busca gulosa possui uma alta eficiência, uma vez que é, geralmente, o mais rápido entre os três analisados, porém tem uma baixa eficácia, já que não garante solução ótima nem busca completa. Já o algoritmo de busca uniforme tem a menor eficiência dos três, mas garante a solução ótima e busca completa. Por fim, o algoritmo A* tem uma média eficiência, contudo uma alta eficácia, pois, sendo a heurística admissível, garante solução ótima e busca completa.

Foi realizado um teste com o estado inicial do puzzle demonstrado pela figura abaixo.

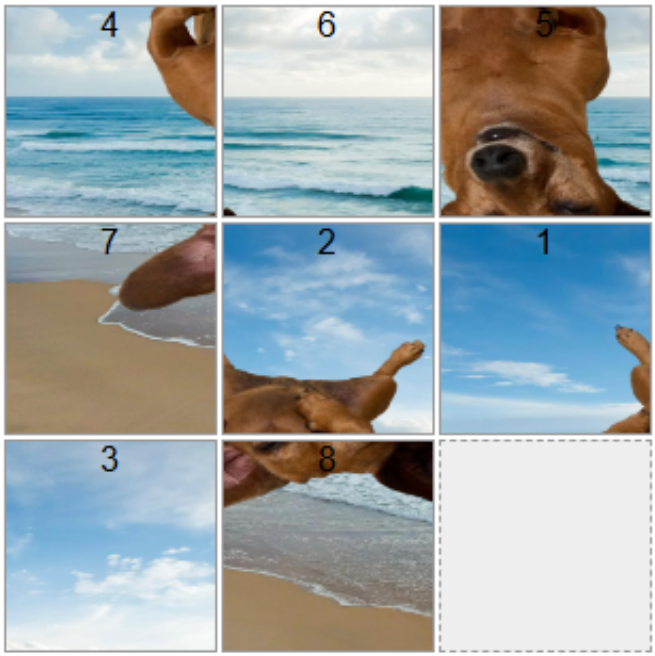

Figura 4. Estado inicial do puzzle.

E, a seguir, estão descritos os resultados.

Busca Gulosa

Tempo médio gasto para encontrar a solução: 2,8 ms.

Número de movimentos para chegar à solução: 46.

Busca uniforme

Tempo médio gasto para encontrar a solução: 15.883,3 ms.

Número de movimentos para chegar à solução: 22.

Busca A*

Tempo médio gasto para encontrar a solução: 41,2 ms.

Número de movimentos para chegar à solução: 22.

5. Conclusão

O algoritmo A*, para os testes realizados e para o exemplo apresentado, se mostrou com o melhor custo-benefício, uma vez que ele fornece a solução ótima e a busca completa em um tempo razoável que, por mais que seja mais demorado que a busca gulosa, não apresenta um tempo médio de execução tão longo quanto a busca uniforme que também apresenta a solução ótima, ou seja, a mesa solução do A*. Por fim, é importante salientar que os tempos médios de execução dependem do estado inicial do puzzle, podendo os algoritmos apresentar tempos maiores ou menores que os apresentados e ter seus tempos mais aproximados, ou mais afastados uns dos outros, porém a solução ótima, dada uma heurística admissível, será a mesma.