Trabalho 1 - Algoritmos de Busca

Jeferson Aires de Sousa[1], **Antonio Deivid Santos Costa**[1]

[1]Universidade Federal do Ceará (UFC) – Campus Quixadá
Caixa Postal 63.900-000 – Quixadá – CE – Brazil

{jefersonoaires, antoniodeivid}@alu.ufc.br

1. Implementação

O projeto foi inteiramente implementado em Python no formato de Python puro, sem quaisquer frameworks de desenvolvimento. As bilbiotecas utilizadas estão todas listadas no arquivo requirements.txt, na raíz do projeto, de modo que as versões de cada uma também estão presentes, facilitando um possível ajuste. A forma de ajustar as versões das biliotecas, tal como a explicação de como rodar os algoritmos estão presentes no arquivo readme.txt, também presente na raíz do projeto.

2. Experimentação

Parte 0 - Experimentação Livre

Cenários de Teste: Foram realizadas 50 iterações para cada algoritmo, utilizando coordenadas de origem e destino geradas aleatoriamente. Funções de Custo: f1, f2, f3 e f4 foram aplicadas a todos os algoritmos. Heurísticas: h1 e h2 foram utilizadas para os algoritmos A* e Busca Gulosa. Resultados Gerados: Para cada execução, foram coletados os seguintes dados: caminho encontrado, custo total, nós gerados, nós visitados, função de custo e heurística utilizada (se aplicável).

Os resultados demonstraram que o algoritmo de Dijkstra se destaca pela sua consistência em encontrar caminhos de menor custo, enquanto o A*, ao combinar funções de custo e heurísticas, apresenta uma eficiência superior, explorando o grafo de forma mais direcionada e com menor custo computacional. A BFS, embora garanta a exploração completa do grafo, mostrou-se menos eficiente em termos de nós explorados. Em contrapartida, a DFS apresentou caminhos mais longos e custos maiores, com baixa confiabilidade na busca por caminhos ótimos. Por fim, a Busca Gulosa teve desempenho variável, dependendo da qualidade da heurística utilizada, sendo menos estável que os demais algoritmos. Em suma, Dijkstra e A* se consolidaram como as opções mais eficazes, com o A* oferecendo uma vantagem adicional em eficiência.

Parte 1 - Largura vs. Profundidade vs. Custo Uniforme

Esta etapa da pesquisa investigou o comportamento dos algoritmos de busca em Largura, Profundidade e Custo Uniforme, analisando a consistência dos resultados em diversas condições.

Foram geradas 50 instâncias aleatórias de problemas de busca, e cada algoritmo foi avaliado em cada problema, variando as funções de custo (f1, f2, f3, f4), resultando em 200 registros por algoritmo (600 no total).

A **Busca em Largura (BFS)** demonstrou um comportamento consistente, com custo médio de 66,25, gerando em média 150,60 nós e visitando 69,52, independentemente da função de custo. Isso indica uma exploração uniforme e eficiente.

A **Busca em Profundidade (DFS)** apresentou um desempenho inferior, com custo médio significativamente mais alto (1162,95) e exploração de um grande número de nós (1069,88 gerados e 932,18 visitados). Isso demonstra que o DFS não é adequado para encontrar soluções de menor custo, especialmente em cenários com alta profundidade de busca.

A **Busca de Custo Uniforme** apresentou um custo médio semelhante ao BFS (63,31), indicando a capacidade de encontrar caminhos ótimos. Contudo, gerou e visitou uma quantidade ligeiramente maior de nós (159 e 73, respectivamente) em comparação com o BFS.

Em comparação geral, o DFS se destacou negativamente pelo alto custo. BFS e Custo Uniforme alcançaram soluções similares e mais eficientes, com o BFS mostrando-se ligeiramente mais eficiente em termos de nós explorados.

Parte 2 - Custo Uniforme vs. A*

Em experimentos comparando Busca de Custo Uniforme e A*, a função de custo f1 do Custo Uniforme se mostrou a mais eficiente, com o menor custo médio (60), gerando 83,5 nós e visitando 70. Já a função f3 foi a menos eficiente, com custo médio de 90 e gerando/visitando mais nós, indicando maior complexidade.

O algoritmo A* superou o Custo Uniforme, especialmente com a combinação f4+h2, que apresentou o menor custo médio (88,17), menos nós gerados (134,22) e visitados (60,74), além de maior estabilidade. Por outro lado, f2+h1 foi a combinação menos eficiente do A*, com o maior custo médio (109,00) e maior número de nós explorados, mostrando que escolhas ruins de funções e heurísticas prejudicam o desempenho.

Em comparação, o Custo Uniforme gerou em média mais nós (156) do que o A* (110,5 com as melhores combinações). Isso destaca a eficiência do A* em usar heurísticas para direcionar a busca, resultando em menores custos computacionais e melhor identificação de caminhos curtos.

Em resumo, o A* é geralmente mais eficaz e eficiente que o Custo Uniforme, especialmente com configurações otimizadas. Os experimentos enfatizam a importância de avaliar cuidadosamente funções de custo e heurísticas, principalmente em problemas que demandam baixo custo computacional e alta precisão.

Parte 3 - Busca Gulosa vs. A*

A **Busca Gulosa** apresentou um custo médio de 103,405, com custos idênticos para as heurísticas h1 e h2, indicando maior influência da função de custo. Gerou uma média de 209,89 nós e visitou uma média de 98,87 nós, com consistência entre heurísticas.

A **Busca A*** apresentou custo médio de 123,6372, com melhores resultados para f4 e piores para f2. A heurística h2 reduziu custos em relação a h1. Gerou uma média de 174,14 nós e visitou uma média de 81,45 nós, com consistência entre heurísticas.

Em comparação geral, a Busca Gulosa demonstrou custos mais baixos que o A*, mas é menos precisa. A* equilibra custos e exploração, com combinações vantajosas. A Busca Gulosa gera significativamente mais nós que A*.

Parte 4: Largura vs. Profundidade com Randomização da Vizinhança

Para o experimento, foram gerados aleatoriamente 20 pares de pontos inicial e objetivo, e para cada par, tanto o BFS quanto o DFS foram executados 10 vezes, com a ordem de expansão dos nós vizinhos sendo embaralhada a cada execução. Os caminhos encontrados e seus custos, calculados por quatro diferentes funções, salvando os dados em uma planilha.

Ao analisar os dados observa-se que a Busca em Largura, de forma geral, apresentou um desempenho superior em comparação à Busca em Profundidade em todas as funções de custo. Em termos de custo médio dos caminhos encontrados, a Busca em Largura obteve valores significativamente menores do que a Busca em Profundidade. Observou-se que a função de custo $f4$ foi a mais eficiente, resultando nos menores custos médios para ambos os algoritmos. Por outro lado, a Busca em Profundidade apresentou os custos mais elevados, com destaque para as funções $f1$ e $f3$, que geraram valores consideravelmente altos.

De maneira geral, a Busca em Largura apresentou resultados consistentemente superiores em termos de custo total, número de nós gerados e número de nós visitados. Esses fatores indicam que a Busca em Largura não apenas encontra caminhos mais curtos (menor custo médio), como também o faz explorando menos estados, o que torna sua execução mais eficiente em termos computacionais.

Já a Busca em Profundidade mostrou ser menos eficiente, com custos significativamente mais altos e maior exploração do espaço de busca. Isso é particularmente evidente em funções como $f3$, nas quais a Busca em Profundidade gerou caminhos muito mais caros e percorreu uma quantidade maior de estados do que a Busca em Largura.

Assim, considerando o equilíbrio entre eficiência no custo do caminho encontrado e o uso de recursos computacionais (nós gerados e visitados), a Busca em Largura é a melhor escolha para os cenários testados.

Parte 5 - Caminho Mínimo Com Uma Parada A Mais

O estudo avaliou o desempenho do algoritmo A* na resolução do problema de caminho mínimo com passagem obrigatória por farmácias, combinando diferentes funções de custo (c1, c2, c3, c4) e heurísticas (H1 e H2) em 25 execuções por combinação. Os resultados indicam que o algoritmo encontrou caminhos válidos na maioria das execuções, com algumas falhas. A heurística H2 majoritamente superou H1, apresentando custos menores e menos nós visitados. A função de custo c4 demonstrou maior eficiência com custos e nós visitados inferiores em comparação com as demais, enquanto c2 apresentou os piores resultados. A combinação H2 e c4 destacou-se como a mais eficiente.