Trabalho Prático 1: Algoritmos de Busca no Pac-Man

Guilherme de Abreu Lima Buitrago Miranda[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)
31270-901 – Belo Horizonte – MG – Brazil

guilhermemiranda@dcc.ufmg.br - 2018054788

1. Introdução

O trabalho prático 1 da disciplina de Introdução à Inteligencia Artificial tem como objetivo a implementação de diferentes algoritmos de busca em espaço de estados e análise comparativa de algoritmos. Dessa forma, após o fornecimento do código fonte de um programa que emula o jogo Pac-Man, foi desenvolvido, utilizando a linguagem Python, diferentes soluções para abordar o problema.

Os algoritmos de busca em estado são bastante importantes na ciência da computação, sendo aplicáveis em diversos contextos, como em sistemas baseados em conhecimento, sequenciamento de produção e buscas na internet, por exemplo. Dessa forma, implementou-se: busca em produndidade (DFS), busca em largura (BFS), busca de custo uniforme (UCS), busca gulosa (best first search) e a busca A*. Por fim, também foi implementada uma nova heurística para o A*.

Nas seções a seguir, encontram-se detalhes relativos à implementação do trabalho, desafios enfrentados, conclusões e referências bibliográficas.

2. Metodologia e Análise Experimental

Após a finalização do passo 1 (familizariazção com o código fonte do Pac-Man), busquei, nos materiais vistos em aula, os pseudocódigos dos algoritmos necessários para implementação do trabalho. Dessa forma, o processo de "tradução" foi bastante trivial, principalmente levando em consideração a simplicidade da escrita na linguagem Python e o fato de que as estruturas de dados necessárias já estavam escritas no código-fonte.

Durante a implementação, pude perceber que, em alguns casos, a busca em profundidade não obtém o caminho mais curto até o objetivo, embora o custo da exploração de nós seja razoavelmente pequeno. Na busca em largura, o caminho mais curto é encontrado, embora seja bastante custoso (assim como a busca de custo uniforme, já que o custo de ir de uma coordenada para uma sucessora é unitário, não gerando, portanto, diferenças nesse caso). Já a busca gulosa apresenta uma boa relação "custo-benefício", mas a melhor solução pareceu ser a A*, observando o custo da exploração de nós e o caminho encontrado.

Dadas tais percepções, após a implementação, planejou-se comparar os algoritmos utilizando três tamanhos de layouts: tiny, medium e bigMaze, utilizando as métricas de custo, segundos gastos, nós expandidos e score, numa máquina i5-8250U (4 cores – 8 threads), 8GB RAM, SSD 120GB rodando Ubuntu 20.04 LTS. Os resultados encontrados foram os seguintes:

Tabela 1. Valores obtidos para o layout tinyMaze

TinyMaze				
Algoritmo	Total cost	Seconds	Nodes Expanded	Score
DFS (2)	10	0	15	500
BFS (3)	8	0	15	502
UCS (3)	8	0	15	502
best first search (5)	8	0	8	502
A* (6)	8	0	14	502
Heurística A* (7)	8	0	14	502

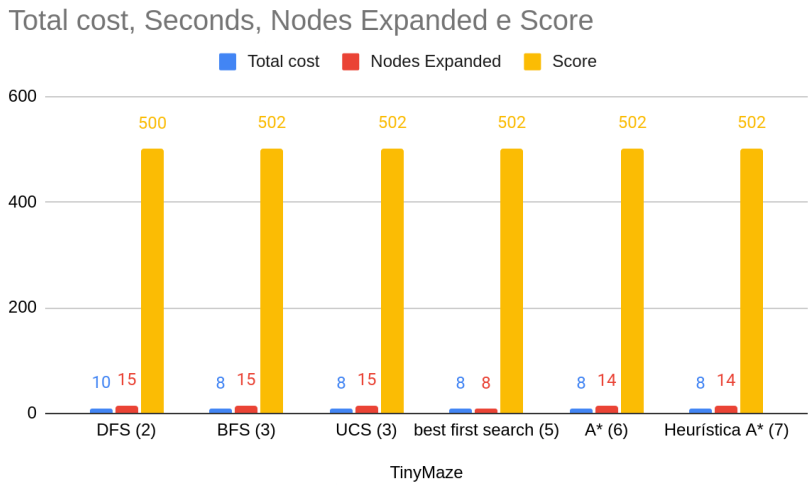

Figura 1. gráfico gerado para o layout tinyMaze

Tabela 2. Valores obtidos para o layout mediumMaze

MediumMaze				
Algoritmo	Total cost	Seconds	Nodes Expanded	Score
DFS (2)	130	0	146	380
BFS (3)	68	0	269	442
UCS (3)	68	0	269	442
best first search (5)	74	0	78	436
A* (6)	68	0	221	442
Heurística A* (7)	68	0,1	221	442

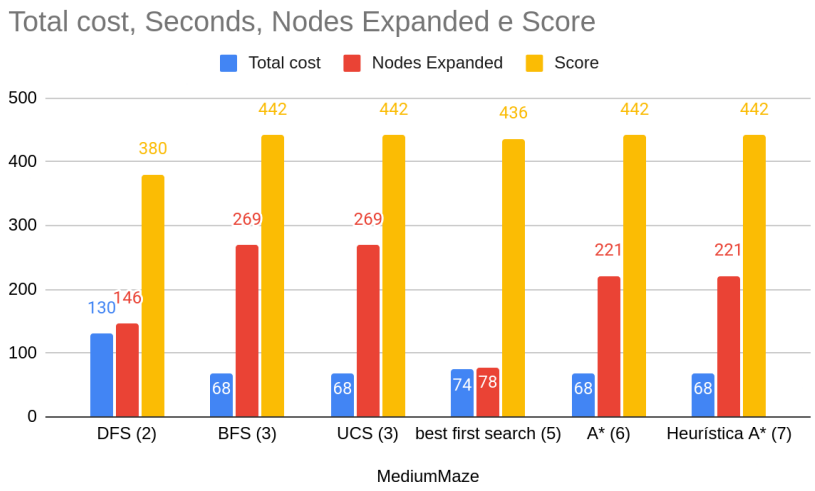

Figura 2. gráfico gerado para o layout mediumMaze

Tabela 3. Valores obtidos para o layout bigMaze

	BigMaze			
Algoritmo	Total cost	Seconds	Nodes Expanded	Score
DFS (2)	210	0	390	300
BFS (3)	210	0	620	300
UCS (3)	210	0,1	620	300
best first search (5)	210	0	466	300
A* (6)	210	0,1	549	300
Heurística A* (7)	210	0,6	549	300

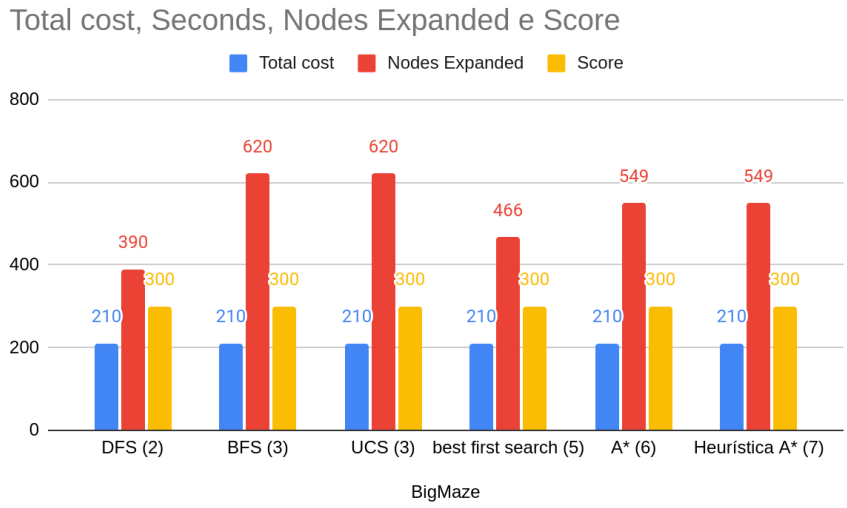

Figura 3. gráfico gerado para o layout bigMaze

Como pode ser observado, boa parte das características notadas na fase de planejamento e construção das soluções foram corroboradas pela análise experimental:

a busca em profundidade nem sempre obtém o melhor caminho possível (score), o número de nós expandidos da busca em largura e da busca de custo uniforme é o mais alto e a busca gulosa e o A* tentam buscar uma boa relação entre nós expandidos e o score final.

Se considerarmos o score mais importante que o número de nós expandidos, o A* é ainda melhor do que a abordagem gulosa, já que para o medium e para o bigMaze, os resultados de score foram melhores (embora, como pode-se imaginar, o número de nós expandidos também seja maior).

No passo 7 do enunciado do trabalho, é também requisitado a implementação de uma heurística para o A*. Para tal, lançou-se mão do método manhattanDistance, já implementado no arquivo utils. Levando em consideração os layouts acima observados, não há nenhuma alteração frente ao A* previamente implementado, mas, ao observar o layout trickySearch, melhorias são observadas: o custo de ambas continuam 60, mas o resultado é mais rapidamente calculado (2.8 segundos para o A* tradicional e 2.1 usando a nova heurística, usando a máquina supracitada), além de uma drástica redução no número de nós expandidos (16688 anteriormente contra 9551 agora), sendo que ambas atingem o mesmo resultado (570 pontos).

A heurística proposta também é admissível e consistente, já que é implementada usando a distância de manhattan e retorna sempre a maior distância encontrada, ou seja, ela nunca superestima o custo de alcançar o objetivo (é otimista) e obedece à lei da desigualdade do triângulo.

3. Conclusão

Com implementação de todos as algoritmos e da heurística propostos no enunciado do trabalho, análise experimental e escrita desta documentação, pode-se afirmar que este foi bastante proveitoso para efetivamente fixar os conceitos expostos até o presente momento na disciplina. Conforme visto, pode-se observar, na teoria e na prática, características marcantes dos algoritmos de busca como a busca em largura, em profundidade, A*, entre outros.

Fica evidente, também, que a escolha do melhor algoritmo depende de uma série de fatores, tais quais o objetivo da solução, quanto tempo pode ser empregado, a quantidade de memória que pode ser gasta, etc. Assim, não existe uma resposta simples e objetiva que responda a pergunta “qual o melhor algoritmo de busca em espaço de estados?”, mas sim diversas abordagens que podem ser melhores ou piores a depender do contexto em que serão utilizadas.

Referências

Slides da Disciplina