INSTITUTO FEDERAL DO ESPÍRITO SANTO
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA – PPCOMP

GUSTAVO AMORA BASÍLIO

ALGORITMOS DE BUSCA

Serra
2023

GUSTAVO AMORA BASÍLIO

ALGORITMOS DE BUSCA

Trabalho apresentado ao Programa de Pós-Graduação em Computação Aplicada – PPCOMP do Instituto Federal do Espírito Santo, como requisito para aprovação da Disciplina de Inteligência Artificial ministrada pelo Prof. Dr. Sérgio Nery.

Serra
2023

SUMÁRIO

1 FUNDAMENTAÇÃO TEÓRICA 2
1.1 BUSCA EM LARGURA (BFS) 2
1.2 BUSCA EM PROFUNDIDADE (DFS) 2
1.3 CUSTO UNIFORME (UCS) 2
1.4 A-ESTRELA (A*) 2
1.5 COMPARATIVO DAS COMPLEXIDADES 2
2 EXPERIMENTOS 4
2.1 ESPECIFICAÇÕES 4
2.2 Q1 4
2.3 Q2 5
3 RESULTADOS 6
3.1 Q1 6
3.2 Q2 7
REFERÊNCIAS 8

1 FUNDAMENTAÇÃO TEÓRICA

1.1 BUSCA EM LARGURA (BFS)

Segundo Russell e Norvig (2009) a Busca em Largura, do inglês *Breadth-first search* é um modelo de busca simples em grafos que inicia a expansão no nó raíz. A partir dele todos os seus sucessores são expandidos primeiramente e depois os sucessores destes nós, em uma estrutura de fila. Assim, o algoritmo somente avança para o próximo nível depois de todo o nível atual ser expandido, não levando em consideração o custo das arestas. Importante destacar que é um modelo de busca completo e por isso a sua complexidade é exponencial.

1.2 BUSCA EM PROFUNDIDADE (DFS)

Russell e Norvig (2009) definem a Busca em Profundidade (DFS - *Breadth-first search*) como um modelo de busca em grafos que inicia a expansão no nó raíz. A partir dele apenas um sucessor é escolhido para ser expandido inicialmente e continua a expansão de acordo com esta lógica, em uma estrutura de pilha. Assim, o modelo DFS segue até o nó no nível mais profundo até o ponto em que o último nó não tem mais vizinhos a serem expandidos. Caso encontre esse cenário, o algoritmo retorna aos nós anteriores até encontrar o primeiro que tenha vizinhos a serem explorador e retorna a lógica de profundidade.

1.3 CUSTO UNIFORME (UCS)

Ainda de acordo com Russell e Norvig (2009), a Busca de custo uniforme (UCS - *Uniform cost search*) a busca em largura tem uma lógica semelhante a do algoritmo BFS, contudo, em vez de seguir a expansão por nós mais rasos, o UCS expande o nó n com o custo do caminho g(n) mais baixo. Para isso, é utilizado uma estrutura de fila com prioridade.

1.4 A-ESTRELA (A*)

O algoritmo A-estrela (A* - *A-star*) é um algoritmo de busca informada. (RUSSELL; NORVIG, 2009). Sua implementação é baseada no algoritmo UCS porém além de expandir o nó apenas com menor custo do caminho g(n), a expansão leva em consideração uma heurística h(n). Assim, a fila de prioridade é definida pelo menor f(n) = g(n) + h(n).

1.5 COMPARATIVO DAS COMPLEXIDADES

Algorítimo	Complexidade Tempo	Complexidade Espaço
BFS	O(b^d)	O(b^d)
DFS	O(b^m)	O(bm)
UCS	O($b^{(}1 + [c/e]$)	O($b^{(}1 + [c/e]$)
A*	O((n + m) log n)	O(n)

(RUSSELL; NORVIG, 2009)

2 EXPERIMENTOS

2.1 ESPECIFICAÇÕES

Os códigos da Q1 e Q2 foram executados em uma máquina com as seguintes especificações:

Configuração
SO Windows 10 Pro 64bits
CPU AMD Ryzen 5 5600G, 3901 Mhz, 6 núcleos, 12 Processadores lógicos
GPU AMD Vega 7, 2GB
RAM 16 GB
ARMAZENAMENTO SSD 512 GB

2.2 Q1

Para a Q1 o código foi executado com a rede desligada e nenhum outro aplicativo local aberto além do editor (VSCode) e interpretador.

No problema proposto foi gerado um labirinto de 300 x 300 com metade de seus caminhos bloqueados aleatoriamente. A seed utiliza foi = 42. As métricas medidas foram:

1 .tempo de execução; 2. número de nós expandidos; 3. custo do caminho e 4. tamanho do caminho.

Para as métricas é esperado, de acordo com a teoria estudada e considerando que os custos dos caminhos são todos iguais a 1, que:

I) BFS: 1. tenha o maior tempo de execução; 2. expanda todos os nós possíveis; 3. tenha um menor custo do caminho pois todos tem o mesmo peso (se fossem diferentes os pesos essa não era a expectativa); 4. encontre o caminho ótimo.

II) DFS: 1. tenha um tempo de execução mais rápido que o BFS; 2. expanda menos nós que o BFS; 3. tenha um maior custo do caminho que o BFS; 4. não encontre o caminho ótimo.

III) UCS: como os pesos são iguais, se espera que se comporte igual ao BFS em todas as métricas.

IV) A*: 1. tenha o menor tempo de execução dentre todos os algoritmos; 2. expanda menos nós dentre todos os algoritmos; 3. tenha um custo do caminho igual ao BFS e UCS; 4. encontre o caminho ótimo.

2.3 Q2

Para a Q2, o script foi executado em ambiente virtual no google collab.

Para o problema proposto foram executados 3 algoritmos de busca: BFS, UCS e A*, cada um deles retornando o custo do caminho e o número de nós expandidos até o destino . A expectativa, de acordo com a teoria estudada é que:

I) BFS retorne o caminho mais raso porém não o mais barato. Todos os nós sendo expandidos.
II) UCS retorne o caminho mais barato e com menos nós expandidos que o BFS.
III) A* retorne o caminho mais barato e com menos nós expandidos que o BFS e o UCS.

3 RESULTADOS

3.1 Q1

Algorítimo	Tempo de execução (s)	Nós expandidos	Custo	Passos
BFS	241.958	54623	474	356
DFS	0.071	995	960	769
UCS	0.378	54623	476	356
A*	0.004	648	489	375

Sobre a consistência dos resultados com a expectativas, temos que:

I) De acordo com o esperado:

Esperado que os algoritmos BFS e UCS se comportassem de forma bem próxima. Era esperado que eles encontrassem o caminho ótimo, ou seja, em comparação aos demais fosse o menor valor. Também que encontrassem o menor custo, dado que os pesos entre caminhos são todos iguais a 1. Ainda, era esperado que eles retornassem o maior número de nós expandidos e os maiores tempos de execução, como de fato aconteceu.

Esperado que o Algoritmo DFS resultasse em menor tempo de execução e menor número de nós expandidos do que as mesmas métricas da dupla BFS - UCS. Ainda, era esperado que dentre todos os algorítimos, o DFS retornasse o maior custo e o maior caminho, como foi comprovado pelos resultados.

Era esperado que a busca A* retornasse o menor número de nós expandidos e o menor tempo de execução dentre todos os algorítimos, o que foi confirmado pelos resultados.

I) Não esperado:

Dado que h(n) executada no código é admissível para o problema, era esperado que o algoritmo A* encontrasse o caminho ótimo assim como o BFS e UCS. Porém, não foi esse o resultado. Apesar de próximo do ótimo, o valor não foi igual. Uma hipótese para isso ter acontecido é que a configuração do labirinto fez com que a heurística, em algum momento, levasse a expansão de um caminho um pouco diferente do ótimo. Isso não acontece com o BFS nem com o UCS pois ambos, no problema dado, expandiram os nós sistematicamente.

Outro ponto que não foi como o esperado foi o tempo de execução do UCS ser bem menor que o do BFS. Era esperado que ambos tivessem o comportamente bem similar ou igual nas métricas, o que de fato aconteceu exceto pelo tempo de execução. Essa diferença por ser por alguns motivos, como por exemplo: 1. utilização de fila de prioridade heapq no UCS e deque no BFS. A estrutura heapq pode ter melhor desempenho. 2. utilização da

função 'esta contido' no BFS executada em cada nó, enquanto no UCS não a utilizamos e sim uma comparação com a variável 'custo caminho'.

3.2 Q2

Algorítimo	Caminho	Custo	Nós expandidos
BFS	Arad, Sibiu, Fagaras, Bucharest	450	20
UCS	Arad, Sibiu, Rimnicu Vilcea, Pitesti, Bucharest	418	13
A*	Arad, Sibiu, Rimnicu Vilcea, Pitesti, Bucharest	418	9

Os resultados foram consistentes com a teoria. Como esperado, tanto os algorítimos UCS e A* encontraram o caminho ótimo considerando o menor custo. Também, os resultados mostram que em nenhum dos dois algorítimos a expansão dos nós foi sistemática, ou seja, não foi necessário expandir todos os nós do mundo do problema para encontrar a otimalidade. Ainda, o número de nós expandidos no UCS > A*. Esse resultado era esperado pois, ao considerar também o valor da heurística h(n) além do custo g(n)em uma fila de prioridades estamos dando mais informações para não expandir nós desnecessários.

REFERÊNCIAS

RUSSELL, Stuart J.; NORVIG, Peter. *Artificial Intelligence: a modern approach*. 3. ed. [S.l.]: Pearson, 2009.