Métodos de Busca

GUILHERME LIMA HERNANDEZ RINCÃO *

*Engenharia de Computação - Graduação
E-mail: ra169052@students.ic.unicamp.br

I. INTRODUÇÃO

Foram testados os métodos de busca sem informação Breadth First Search e Depth First Search, e o método de busca informada Best First Search para encontrar um caminho a ser percorrido por um robô entre uma posição inicial e uma posição final em um mapa com paredes. A linguagem utilizada foi Python, com os algoritmos de busca da biblioteca AIMA (https://github.com/aimacode/aima-python/blob/master/search.ipynb).

II. MODELAGEM DO PROBLEMA

A. Mapa

O mapa foi representado como um vetor bidimensional 60x60, no qual cada elemento (x, y) representa uma posição do mapa (nó) e seu valor indica se esta posição é livre ou uma parede. Para cada teste também são definidas coordenadas para o ponto de partida do robô (azul) e a posição final desejada (verde). O custo para transitar entre uma posição do mapa e suas adjacentes foi definido como 1.

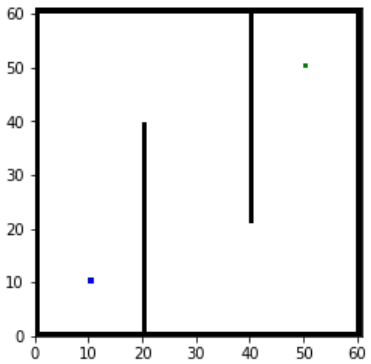

Figura 1. Visualização do mapa

B. Estado do robô

O estado do robô foi definido como sua posição (x, y) no labirinto.

C. Conjunto de ações

Quatro ações foram definidas.

- Mover para direita: incrementa x se a posição a direita (x + 1, y) é livre.
- Mover para esquerda: decrementa x se a posição a esquerda (x - 1, y) é livre.
- Mover para cima: incrementa y se a posição acima (x, y + 1) é livre.
- Mover para baixo: decrementa y se a posição abaixo (x, y - 1) é livre.

D. Teste do estado objetivo

O teste do estado objetivo checa se a posição sendo visitada tem suas coordenadas (x, y) iguais às coordenadas (x, y) do objetivo.

III. ESPECIFICAÇÕES DAS BUSCAS

Como o problema foi modelado em forma de grafo, existe possibilidade de laços ao procurar caminhos. Portanto, para evitá-los, todas as buscas utilizam de verificação de nós já visitados.

Para cada busca foram testados os seguintes conjuntos de posições iniciais e finais para o robô:

Tabela I
POSIÇÕES INICIAIS E FINAIS.

Número	Início	Fim
1	(32, 12)	(37, 17)
2	(10, 10)	(50, 50)
3	(59, 59)	(1, 1)
4	(19, 1)	(21, 21)

Nas próximas seções serão apresentadas as soluções para cada método onde as cores no mapa representam as seguintes propriedades dos nós:

Tabela II
POSIÇÕES INICIAIS E FINAIS.

Cor	Significado
Branco	Não visitado
Preto	Parede
Azul	Início
Verde	Fim
Laranja	Fronteira
Cinza	Explorado
Vermelho	Último explorado
Rosa	Solução

IV. BUSCAS SEM INFORMAÇÃO

A. Breadth First Search

O BFS é uma busca em extensão, completa pois o mapa é finito e ótima pelos custos dos caminhos serem iguais.

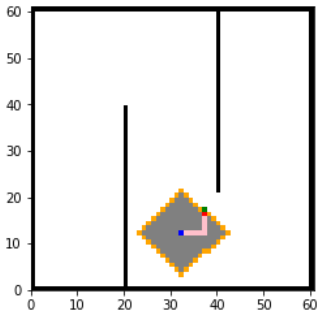

Figura 2. Teste 1 do método Breadth First Search.

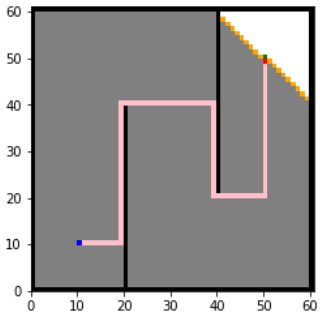

Figura 3. Teste 2 do método Breadth First Search.

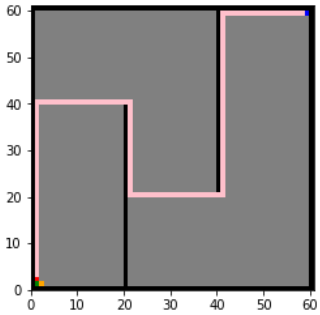

Figura 4. Teste 3 do método Breadth First Search.

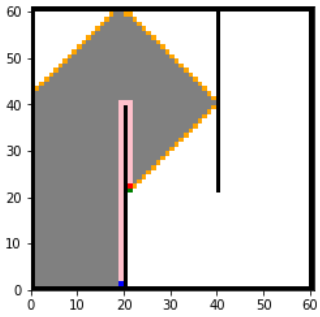

Figura 5. Teste 4 do método Breadth First Search.

B. Depth First Search

O DFS é uma busca em profundidade, completa pois o mapa é finito e não ótima já que a solução não será o melhor caminho possível.

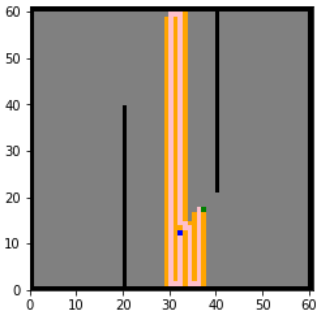

Figura 6. Teste 1 do método Depth First Search.

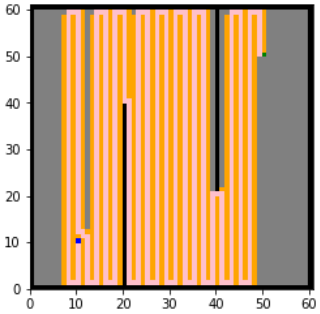

Figura 7. Teste 2 do método Depth First Search.

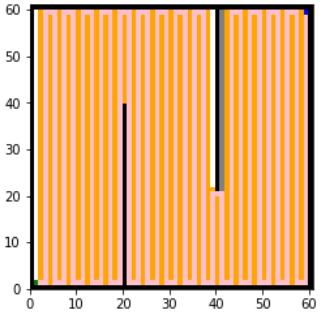

Figura 8. Teste 3 do método Depth First Search.

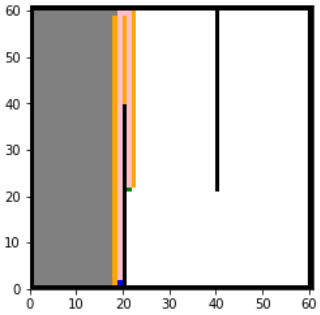

Figura 9. Teste 4 do método Depth First Search.

V. Busca informada - Best First Search

Best First Search é uma busca gulosa pela melhor escolha. Essa busca é completa. Os testes foram realizados com duas heurísticas:

- Distância euclidiana: distância de uma linha reta entre o nó e o objetivo.

- Distância Manhattan: distância horizontal somada com a distância vertical do nó até o objetivo.

A. *Distância euclidiana*

Nesse caso a solução é ótima pois a heurística é admissível.

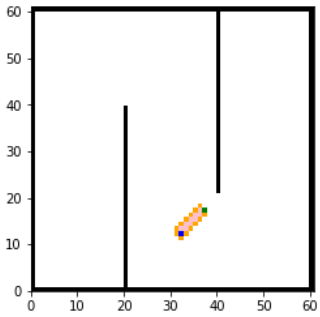

Figura 10. Teste 1 do método Best First Search com heurística de distância euclidiana.

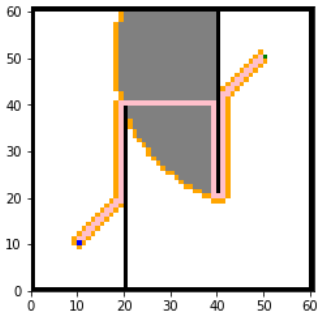

Figura 11. Teste 2 do método Best First Search com heurística de distância euclidiana.

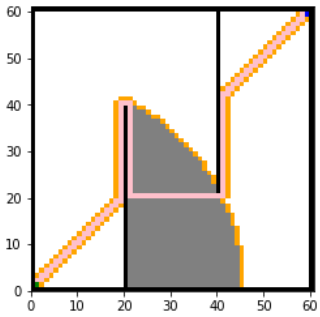

Figura 12. Teste 3 do método Best First Search com heurística de distância euclidiana.

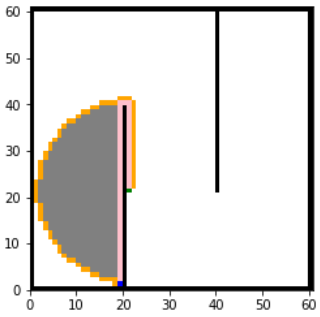

Figura 13. Teste 4 do método Best First Search com heurística de distância euclidiana.

B. *Distância manhattan*

Para essa heurística, a solução não será ótima pois ela superestima o custo para alcançar o objetivo.

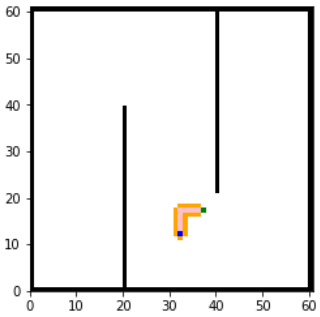

Figura 14. Teste 1 do método Best First Search com heurística de distância manhattan.

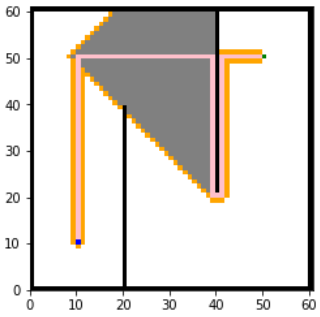

Figura 15. Teste 2 do método Best First Search com heurística de distância manhattan.

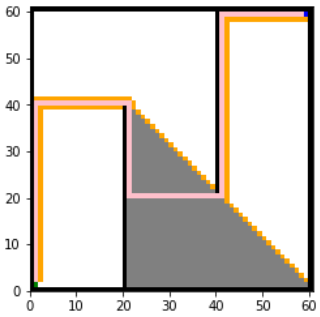

Figura 16. Teste 3 do método Best First Search com heurística de distância manhattan.

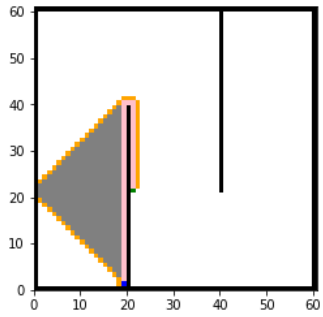

Figura 17. Teste 4 do método Best First Search com heurística de distância manhattan.

VI. Desempenho

Abaixo estão os tempos de execução para cada método e configuração de mapa.

Tabela III
DESEMPENHO DOS MÉTODOS DE BUSCA EM TEMPO DE EXECUÇÃO

Mapa I	BFS	DFS	BestFS-E	BestFS-M
1	4,49ms	1049,30ms	0,50ms	0,50ms
2	83,44ms	1434,55ms	36,46ms	54,91ms
3	81,91ms	1582,77ms	44,92ms	51,91ms
4	37,44ms	205,84ms	16,47ms	10,98ms

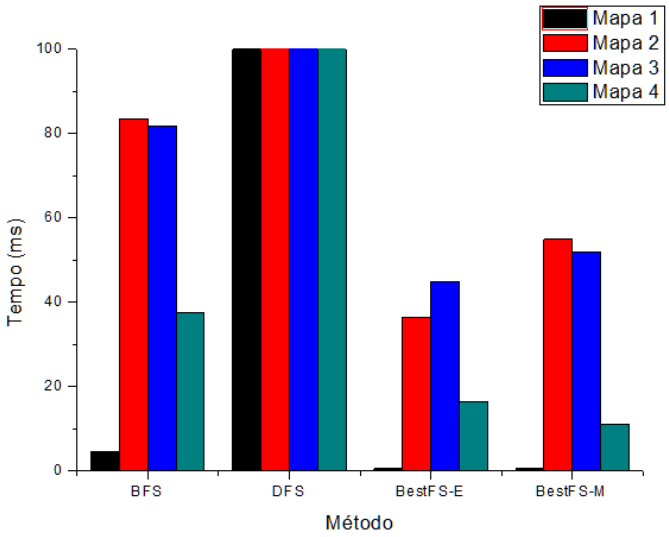

Figura 18. Tempos de execução.

VII. Análise

A. Breadth First Search

Na Figura 2 é evidente o funcionamento do BFS em grafos. Os nós são expandidos como um 'alagamento', portanto o desempenho é melhor para os mapas em que o objetivo está próximo ao início. Esse método chega a ser somente 30ms mais lento que as buscas informadas. Portanto para esse problema em que as restrições (paredes) guiam a expansão até o objetivo, o BFS é o metodo de busca não informada mais adequado.

B. Depth First Search

O caminho percorrido pelo DFS é totalmente dependente da ordem de empilhamento especificada para as ações. No caso apresentado nas figuras, o DFS primeiro executa as ações de andar para baixo e para esquerda. Assim o desempenho é melhor nos casos em que o objetivo está a esquerda do ponto inicial. Porém, mesmo em casos simples o desempenho do algoritmo é ordens de grandeza acima da busca BFS. Portanto, para esse problema em que a altura da árvore de ações é grande, a DFS não é eficaz.

C. Best First Search

Para as duas heurísticas, a busca informada é significantemente mais rápida que as buscas não informadas. Nas figuras vemos que nós desnecessários não são expandidos. Comparando as heurísticas de distância, é interessante notar que em certos casos onde há obstáculos, a Manhattan é mais rápida que a Euclidiana (observe Figura 13 e Figura 17) já que a área expandida no entorno do obstáculo é menor. Da Tabela III, a heurística euclidiana é mais rápida no mapa 2 que no mapa 3. Isso é invertido na heurística manhattan. Observando a Figura 15 vemos que a distância manhattan faz com que no início seja percorrido um caminho desnecessário acima da parede da esquerda, por isso o caminho feito para superar a parede da direita é maior. Já na Figura 16, a disposição do mapa faz com que o caminho percorrido para desviar das paredes seja quase ótimo pois não há distância vertical percorrida desnecessariamente.

VIII. Código

O código para esse projeto pode ser encontrado no GitHub em (https://github.com/guilhermelhr/search-algorithms-study).