Relatório TP1 - Busca no espaço de estados

Mateus Pinto da Silva

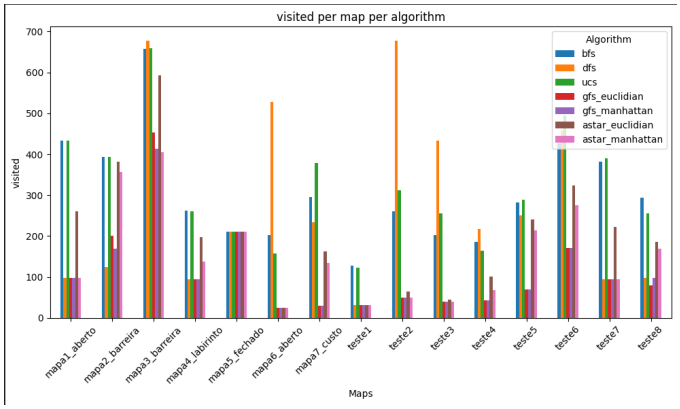
[1]Universidade Federal de Viçosa
mateus.p.silva@ufv.br

Introdução e desenvolvimento

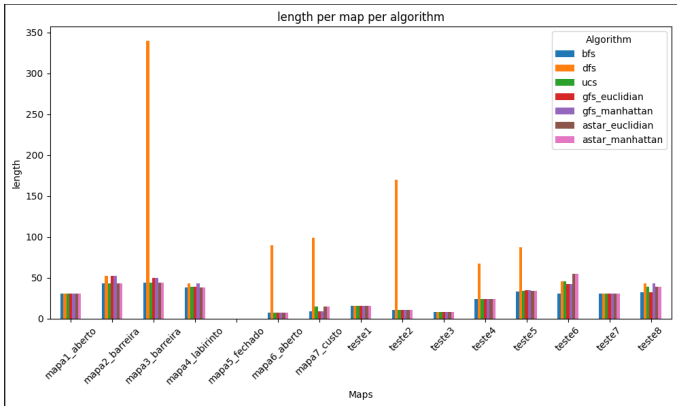
O presente relatório discute os resultados da implementação dos algoritmos de busca de estado: BFS, DFS, UCS, busca gulosa e A*. Nos dois últimos algoritmos, foram implementadas como funções heurísticas as distâncias euclidiana e de Manhattan. Foi tentado replicar ao máximo a implementação do professor, usando a mesma ordem de ramificação dos nós. Foram feitos 8 mapas, 7 cumprindo os desafios propostos e 1 mais aleatório.

Visão geral dos algoritmos

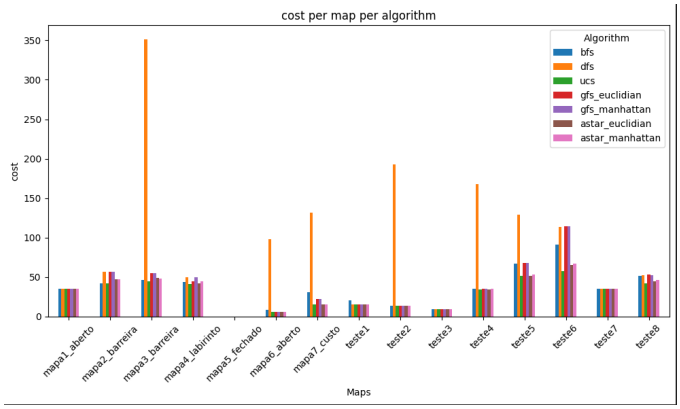
Em relação ao número de nós visitados, os algoritmos não informados apresentam os maiores valores, especialmente o DFS. Quanto aos informados, destaca-se o algoritmo guloso (GFS), visto que ele não se preocupa em achar um caminho com baixo custo.



Em relação ao tamanho do caminho, DFS novamente apresenta valores muito grandes, e todos os outros algoritmos apresentam valores muito próximos, o que é bastante esperado. O A* tem, em alguns casos, o tamanho do caminho maior, pois ele leva em consideração o custo de cada estado, preferindo caminhos maiores se eles tiverem o custo menor.

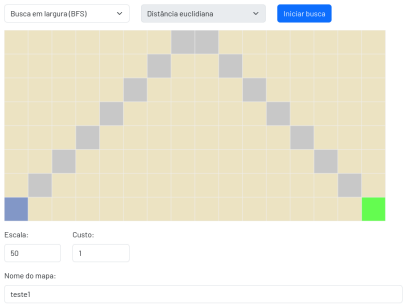


Quanto ao custo, DFS novamente com valores gigantes, e os algoritmos A* apresenta valores ótimos de custo, o que é o esperado.



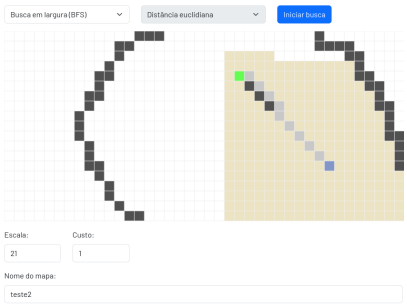
BFS não é ótimo

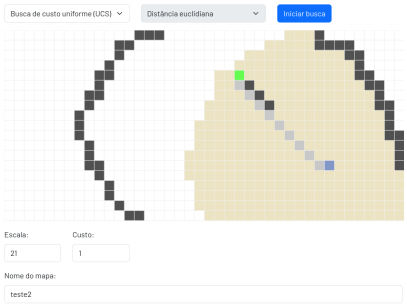
Casos assim são simples. Basta que o estado objetivo não esteja no sentido da primeira opção de movimento (aqui diagonal superior-direita).



BFS é equivalente à UCS

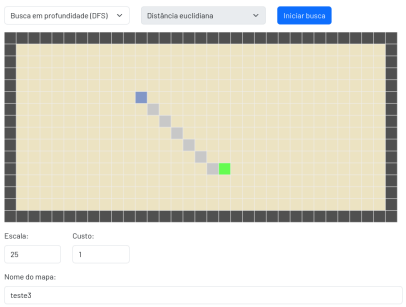
Simétrico ao exemplo anterior: é necessário que o estado objetivo esteja no sentido da primeira opção de movimento (aqui diagonal superior-direita). Caso haja alguns obstáculos, isso muda um pouco os caminhos mas não muda a solução significamente.





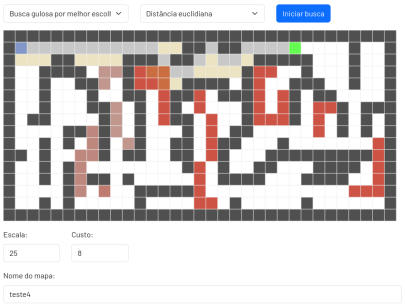
DFS retorna a solução ótima

Análogo ao anterior, porém note que essa solução visita todos os nós do mapa.



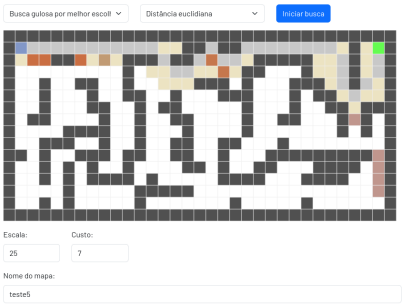
Greedy é ótimo

Greedy é ótimo, e isso independe um pouco da heurística, quando o caminho que se aproxima mais rapidamente do objetivo não tem bloqueios e/ou estados muito caros.



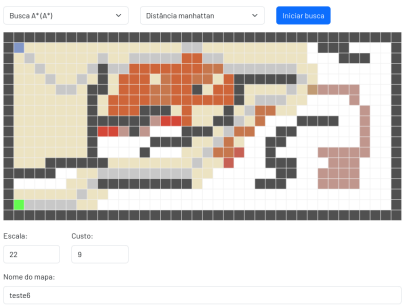
Greedy não é ótimo

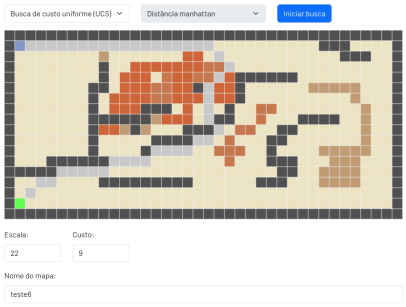
Simetricamente, Greedy não é ótimo quando o caminho que se aproxima mais rapidamente do objetivo tem bloqueios e/ou estados muito caros.



A* é melhor que UCS

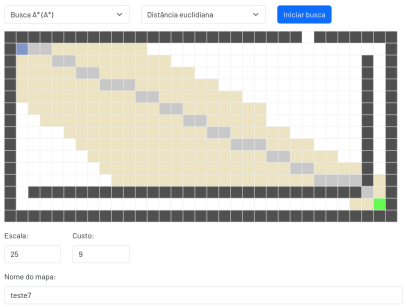
A* retorna um resultado melhor que UCS em custo quando o menor caminho tem um custo mais alto. Note que o UCS desconsidera os custos. Além disso, o UCS tende a expandir mais nós.

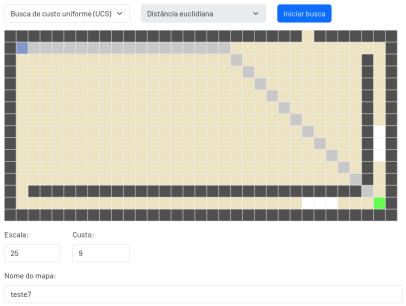




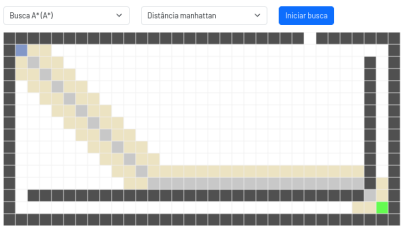
A* é equivalente à UCS

Simetricamente, A* equivale a UCS quando o menor caminho do estado inicial até o final é coincidentemente o caminho de menor custo. Note que eles podem apresentar caminhos diferentes, porém com tamanho do caminho e custo muito parecidos.





Utilizando distância de Manhattan, o número de nós visitado nesse caso é consideravelmente menor.



Aleatório

Criei também um caso aleatório de labirinto com pesos com finalidade de testar a diferença entre distância de Euclides e Manhattan, e também para ser um caso mais ”realista” para os gráficos da primeira seção.

Utiliznado Manhattan, nesse caso a diferença é bastante pequena.

Considerações finais

Os algoritmos de busca em estado são bastante diferentes entre si. Em geral, os algoritmos de busca informada se saem melhor para os mais diversos casos, exigindo menos visitas. Porém, é necessário que uma heurística admissível seja usada. Qual algoritmo específico escolher varia, é claro, caso-a-caso. Neste caso específico, para achar o caminho ótimo visitando o menor número de nós, seria preferível o A* com Manhattan.

Quanto as heurísticas, Manhattan aqui parece fazer mais sentido pois apenas movimentos em grid são possíveis, e a distância de Euclides diz sobre a distância em linha reta de dois pontos. Sendo assim, a distância de Manhattan é mais coerente com o problema em questão de encontrar um caminho de um ponto de Grid até outro assumindo movimentos nesse mesmo Grid. Claro que, para cada problema, é necessário e crítico escolher uma boa função heurística para algoritmos de busca de estado informadas.