Projeto 2: Laboratório de Busca Informada (UCS vs. Greedy vs. A*)

Autor: Marcos Antonio Teles de Castilhos

Disciplina: FGA0221 - Inteligência Artificial

Professor: Fabiano Araujo Soares, Dr.

1. Introdução

Este projeto serve como um laboratório visual para explorar e comparar algoritmos de busca que utilizam informações sobre o custo do caminho e/ou estimativas heurísticas para guiar a exploração do espaço de estados. O problema base continua sendo encontrar o caminho mais curto em um labirinto bidimensional, permitindo uma comparação direta com as estratégias de busca não informada do Projeto 1.

Os algoritmos implementados são:

- **Busca de Custo Uniforme (UCS - Uniform Cost Search):** Prioriza caminhos com menor custo acumulado (g(n)).
- **Busca Gananciosa (Greedy Best-First Search):** Prioriza caminhos que *parecem* mais próximos do objetivo, com base em uma heurística (h(n)).
- **A* (A-Star):** Combina o custo acumulado e a estimativa heurística (f(n) = g(n) + h(n)) para encontrar o caminho ótimo de forma eficiente.

Nota Importante: A Busca de Custo Uniforme (UCS) é tecnicamente classificada como uma busca **não informada**, pois não utiliza nenhuma heurística (informação sobre o objetivo) para guiar a busca. Ela apenas otimiza a expansão com base no custo já percorrido (g(n)). Incluí o UCS neste projeto como um ponto de comparação essencial para entender o papel isolado do custo (g(n)) antes de introduzir a heurística (h(n)) nas buscas informadas propriamente ditas (Greedy e A*).

2. Conceito de Busca Informada e Heurísticas

Diferente da busca cega, a **Busca Informada** utiliza conhecimento adicional sobre o problema para tomar decisões mais inteligentes sobre qual caminho explorar. Essa informação é encapsulada em uma **função heurística, h(n)**, que estima o custo para ir do nó atual n até o nó objetivo.

Uma boa heurística permite que o algoritmo explore significativamente menos nós do que uma busca cega, focando nos caminhos mais promissores. Para garantir que algoritmos como o A* encontrem a solução ótima (o caminho mais curto/barato), a heurística deve ser **admissível**, ou seja, ela nunca deve superestimar o custo real para alcançar o objetivo.

Neste projeto, utilizamos a Distância de Manhattan como função heurística:

h(n) = | linha_atual - linha_final | + | coluna_atual - coluna_final |

Esta heurística é admissível para um labirinto onde os movimentos são apenas horizontais e verticais, pois representa a distância mínima teórica sem considerar as paredes.

3. Algoritmos Implementados

Todos os algoritmos neste projeto utilizam uma **Fila de Prioridade** (implementada com **heapq**) para gerenciar a fronteira. A diferença fundamental entre eles reside em qual valor é usado para definir a prioridade de um caminho na fila.

3.1. Busca de Custo Uniforme (UCS)

- **Estratégia**: Sempre expande o nó na fronteira que tem o menor custo acumulado desde o início (g(n)).
- **Prioridade na Fila**: prioridade = g(n).

- **Propriedades**: É completo e ótimo em termos de custo total do caminho. Ignora completamente a heurística.
- **Visualização**: No nosso labirinto (custo 1 por passo), g(n) é a profundidade. A exploração se assemelha à do BFS, expandindo em ondas concêntricas, e o texto nas células mostra o valor g=....

3.2. Busca Gananciosa (Greedy Best-First Search)

- **Estratégia:** Sempre expande o nó na fronteira que *parece* estar mais próximo do objetivo, com base **apenas na heurística (h(n))**.
- **Prioridade na Fila:** prioridade = h(n).
- **Propriedades:** É rápida, mas **não é ótima** e não é completa (pode entrar em loops se não houver controle de visitados). Ela pode ser facilmente enganada por "becos sem saída" que pareciam promissores.
- **Visualização:** A exploração é muito direcionada ao 'E'. O texto nas células mostra o valor **h=...** (Distância de Manhattan até 'E'). A área explorada tende a ser menor, mas o caminho final pode ser mais longo.

3.3. A* (A-Star)

- **Estratégia:** Combina as informações do UCS e da Busca Gananciosa. Sempre expande o nó na fronteira com o menor valor de **f(n) = g(n) + h(n)**.
- **Prioridade na Fila:** prioridade = g(n) + h(n).
- **Propriedades:** É completo e **ótimo** (encontra o caminho de menor custo), desde que a heurística h(n) seja admissível. É geralmente muito mais eficiente que o UCS em termos de nós explorados.
- **Visualização:** A exploração é direcionada como na Busca Gananciosa, mas mais cautelosa. O texto nas células mostra os três valores: f (em destaque), **g** e **h**, permitindo ver como o algoritmo equilibra o custo real e a estimativa para tomar suas decisões.

4. Implementação e Visualização Gráfica

- **Função Principal: buscar_no_labirinto_informado** unifica a lógica dos três algoritmos, alterando apenas o cálculo da **prioridade** para o **heapq**.

- **Fila de Prioridade (heapq):** Essencial para esses algoritmos, garantindo que o nó com a menor prioridade (seja g(n), h(n) ou f(n)) seja sempre explorado primeiro.
- **Controle de Visitados:** O dicionário **visitados** agora armazena o *menor* g(n) encontrado para chegar a cada nó. Isso é crucial para a otimalidade do UCS e A*, pois permite que o algoritmo reabra um nó se encontrar um caminho mais barato até ele.
- **Visualização:** Semelhante ao Projeto 1, mas agora inclui a exibição dos valores **g, h** e **f** nas células, permitindo uma análise detalhada do "processo de pensamento" de cada algoritmo. A legenda das cores permanece a mesma.

5. Como Usar o Programa

1. **Pré-requisitos:** Python 3 e **matplotlib** instalados.
2. **Execução:**

   python busca-informada.py

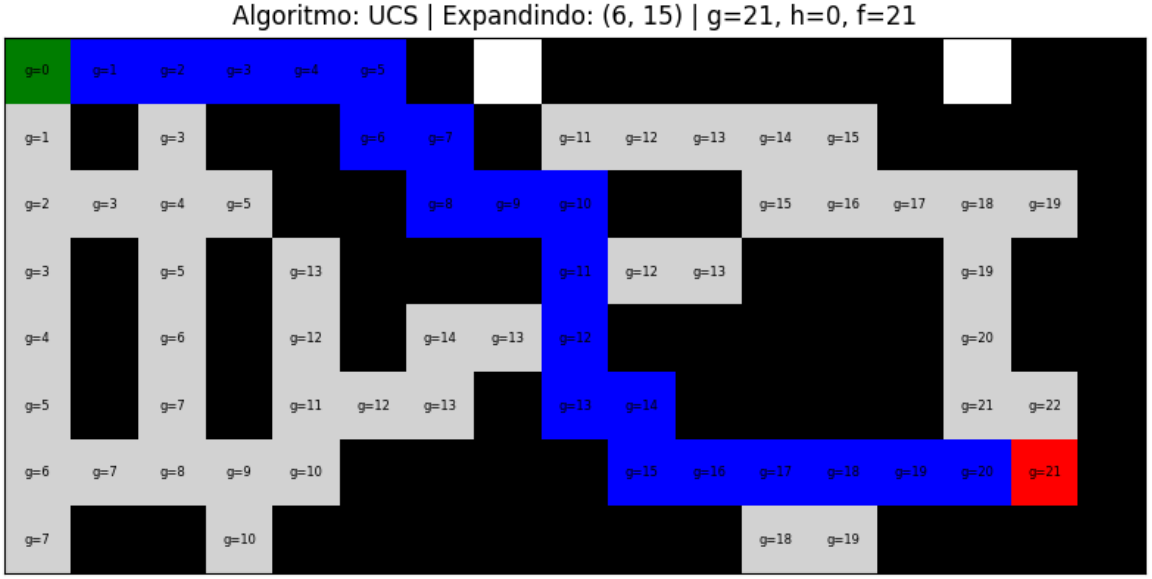
3. **Escolha do Algoritmo:** Digite **ucs, greedy** ou **a_star** e pressione Enter.
4. **Visualização:** Observe a animação na janela gráfica. Compare a área explorada, a "direção" da busca e os valores **g/h/f** mostrados para cada algoritmo.
5. **Resultado:** O caminho final será impresso no terminal. Compare o comprimento do caminho encontrado por cada algoritmo.

6. Imagens

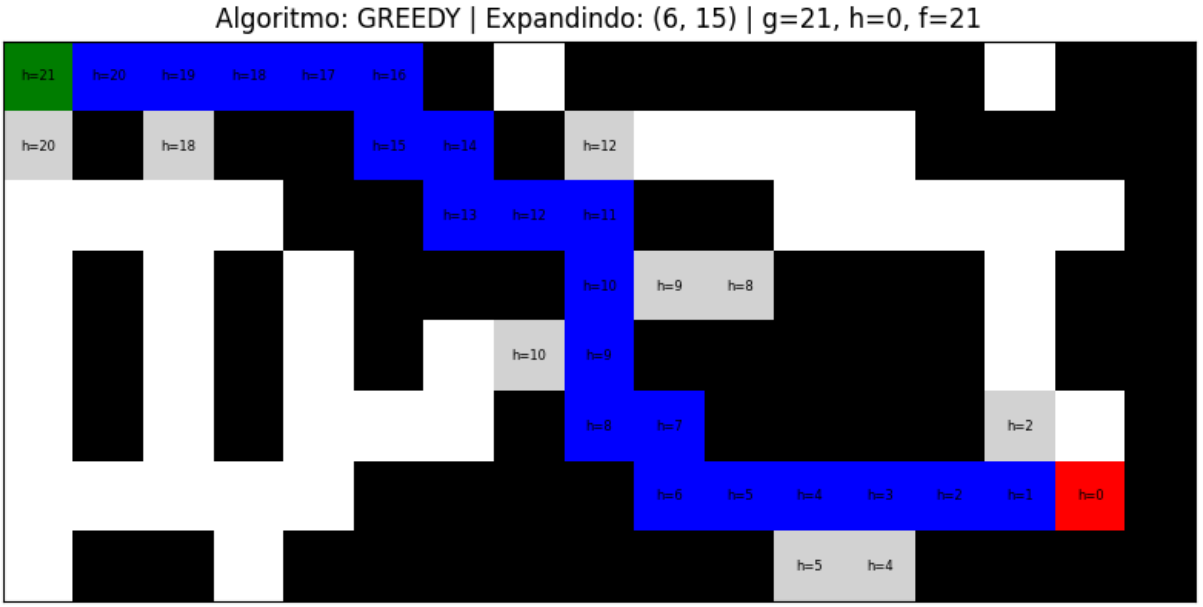
```
sh-5.2$ python3 busca-informada.py
--- Labirinto Original ---
S     # ###### ##
 # ##  #     ####
    ##   ##     #
 # # ###   ### ##
 # # #   ##### ##
 # #   #  ####  #
     ####      E#
 ## #######  ####

Escolha o algoritmo de busca (ucs, greedy, ou a_star):
```

```
Solução encontrada! Feche a janela gráfica para terminar.

--- Labirinto com Solução Final (Terminal) ---
S*****# ###### ##
 # ##**#     ####
    ######     #
 # # ###*  ### ##
 # # #  *##### ##
 # #   #**####  #
     ##########E#
 ## #######  ####

Caminho encontrado pelo A_STAR com 21 passos.
```

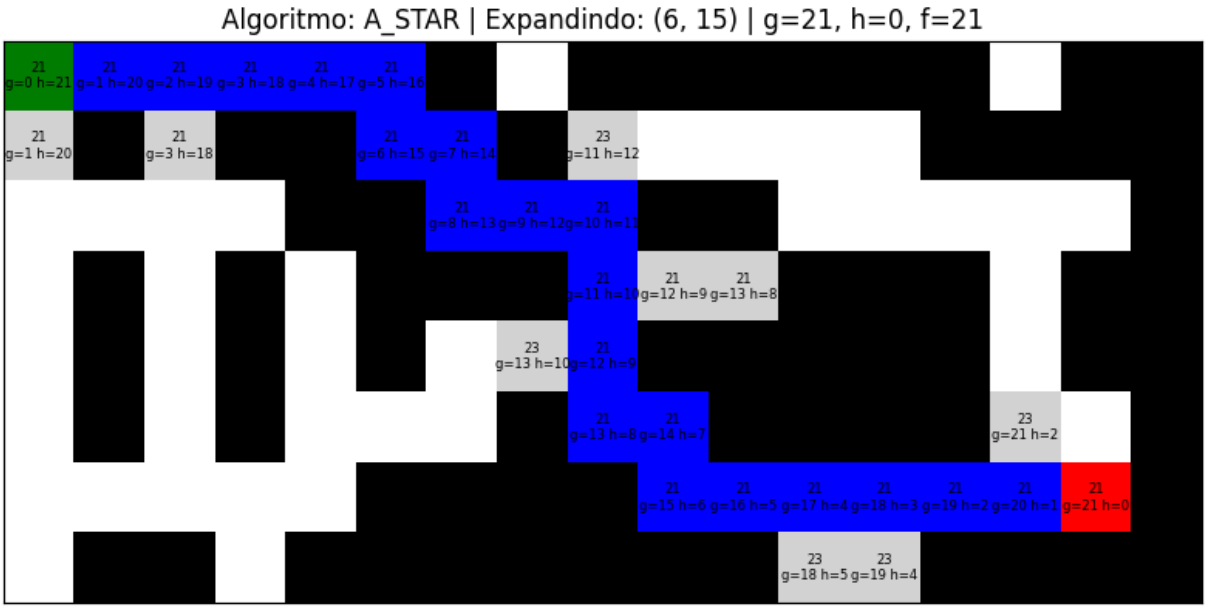
UCS



GREEDY



A*



7. Conclusão

Este projeto demonstra eficazmente os princípios da busca informada. A comparação visual entre UCS, Greedy e A* ilustra claramente:

- O UCS encontra o caminho ótimo, mas explora extensivamente.
- O Greedy é rápido e direcionado, mas pode falhar em encontrar o caminho ótimo.

- O A* combina o melhor dos dois mundos, usando a heurística para guiar a busca eficientemente, mas usando o custo real para garantir a otimalidade.

A implementação da heurística admissível (Distância de Manhattan) e o uso correto da fila de prioridade com as diferentes funções de avaliação (g(n), h(n), f(n)) são os pontos-chave deste projeto.