Relatório Trabalho I

Álvaro Silva
João Paulo da Cunha Faria

Introdução

Este relatório apresenta a implementação e a análise comparativa de algoritmos de busca em espaço de estados, aplicados ao problema de resolução de um labirinto. O objetivo principal é consolidar os conceitos de buscas não informadas e informadas, avaliando seus respectivos desempenhos através de métricas quantitativas e qualitativas.

Para este estudo, foram implementados e analisados quatro algoritmos distintos, divididos em duas categorias:

- **Buscas Não Informadas:** Busca em Largura (BFS) e Busca de Custo Uniforme.
- **Buscas Informadas:** Busca Gulosa (Greedy Best-First Search) e A*.

A avaliação de desempenho foi conduzida com base em métricas de tempo de execução, uso de memória (definido como o número máximo de nós armazenados simultaneamente nas estruturas de dados), número de nós expandidos, completude e optimalidade. A análise final discute os *trade-offs* inerentes a cada estratégia e o papel fundamental da heurística na eficiência das buscas informadas.

Metodologia

Esta seção descreve a representação do problema, o formato dos dados de entrada, os algoritmos implementados em Python com suas bibliotecas nativas e as funções heurísticas utilizadas.

Representação do Problema

O problema, proposto na disciplina de Inteligência Artificial, consiste em implementar e comparar algoritmos de busca para encontrar um caminho (preferencialmente ótimo) do ponto inicial 'S' ao objetivo 'G' em diferentes labirintos.

O labirinto é lido a partir de um arquivo de texto ('.txt') onde a grade é representada por caracteres específicos: 'S' (início), 'G' (objetivo), '#' (parede) e '.' (caminho livre). A partir dessa representação, o problema foi modelado como uma busca em um espaço de estados:

- **Estado:** Uma tupla $(linha, coluna)$ representando a posição atual no grid.

- **Estado Inicial:** A posição correspondente ao caractere 'S'.
- **Teste de Objetivo:** Verifica se o estado atual corresponde à posição 'G'.
- **Ações:** Movimentos para Norte, Sul, Leste e Oeste.
- **Modelo de Transição:** Função que retorna o novo estado após uma ação, validando limites e obstáculos.
- **Custo do Caminho:** Custo unitário para cada passo.

Labirintos Utilizados

Foram utilizados três labirintos com diferentes topologias e níveis de complexidade para testar o desempenho dos algoritmos.

```
S....
.###.
..#..
.###.
....G
```

Listing 1: Labirinto 1 - Simples

```
.............
..........#..
..........#..
..........#..
#.......S.#..
G###########.
.............
.............
.............
.............
.............
.............
.............
.............
.............
```

Listing 2: Labirinto 2 - Caminho longo com armadilha heurística

```
...#.........
.#.#......#..
.#.#......#..
.#.#......#..
.#.#....S.#..
.#.#########.
.#...........
.############
.............
############.
...#...#...#.
G#...#...#...
```

Listing 3: Labirinto 3 - Complexo e sinuoso

Algoritmos Implementados

Busca em Largura (BFS - Breadth-First Search)

A Busca em Largura é um algoritmo de busca não informada que explora os nós vizinhos antes de se aprofundar. Para gerenciar a fronteira, utiliza uma estrutura de dados do tipo **Fila (Queue - FIFO)**. É completo e ótimo para problemas com custo de passo unitário.

Busca em Profundidade (DFS - Depth-First Search)

A Busca em Profundidade explora um ramo da árvore de busca o mais profundamente possível antes de retroceder (backtrack). Utiliza uma **Pilha (Stack - LIFO)** para gerenciar a fronteira. É completo em grafos finitos, mas não garante a optimalidade.

Busca Gulosa (Greedy Best-First Search)

A Busca Gulosa é um algoritmo informado que expande o nó que parece estar mais próximo do objetivo, guiando-se exclusivamente por uma função heurística, $h(n)$. Utiliza uma **Fila de Prioridade** ordenada por $h(n)$. Não garante a optimalidade.

Busca A* (A-Star)

O A* é um algoritmo informado que combina o custo do caminho percorrido ($g(n)$) com a estimativa heurística ($h(n)$). Ele expande o nó que minimiza a função $f(n) = g(n) + h(n)$, utilizando uma **Fila de Prioridade**. É completo e ótimo, desde que a heurística seja admissível.

Funções Heurísticas

Para os algoritmos de busca informada (Gulosa e A*), duas funções heurísticas foram implementadas e testadas: a **Distância de Manhattan** e a **Distância Euclidiana**. Ambas são admissíveis para este problema.

Dado um nó n na posição (x_1, y_1) e o objetivo G na posição (x_2, y_2):

- **Distância de Manhattan (Admissível para 4 direções):**

$$h(n) = |x_1 - x_2| + |y_1 - y_2|$$

- **Distância Euclidiana (Admissível):**

$$h(n) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Resultados e Análise

Os algoritmos foram executados nos três labirintos de teste. As métricas de desempenho coletadas — tempo de execução, pico de memória (número máximo de elementos na fronteira e na lista de explorados), nós expandidos e custo da solução encontrada — estão consolidadas nas Tabelas 1, 2 e 3.

Os resultados das buscas informadas (A* e Greedy BFS) apresentados nas tabelas referem-se à execução com a heurística de **Distância de Manhattan**, que se mostrou a mais adequada para um grid com 4 direções de movimento.

Tabela 1: Resultados Comparativos das Métricas de Busca (Labirinto 1)

Algoritmo	Tempo (s)	Memória	Nós Expandidos	Custo	Nós no Caminho
BFS	0.000078	21	18	8.0	9
DFS	0.000028	13	10	8.0	9
A*	0.000064	21	17	8.0	9
Greedy BFS	0.000031	14	9	8.0	9

Tabela 2: Resultados Comparativos das Métricas de Busca (Labirinto 2)

Algoritmo	Tempo (s)	Memória	Nós Expandidos	Custo	Nós no Caminho
BFS	0.000436	167	151	27.0	28
DFS	0.000710	243	141	47.0	48
A*	0.000507	102	75	27.0	28
Greedy BFS	0.000420	106	71	43.0	44

Tabela 3: Resultados Comparativos das Métricas de Busca (Labirinto 3)

Algoritmo	Tempo (s)	Memória	Nós Expandidos	Custo	Nós no Caminho
BFS	0.000271	103	102	73.0	74
DFS	0.000218	115	90	85.0	86
A*	0.000365	103	102	73.0	74
Greedy BFS	0.000272	107	98	81.0	82

Análise dos Resultados

A análise dos dados consolidados nas tabelas revela os *trade-offs* de cada algoritmo, que se tornam mais evidentes à medida que a complexidade do labirinto aumenta.

Optimalidade (Custo do Caminho)

A métrica de custo demonstra a diferença fundamental entre as garantias de cada algoritmo.

- **BFS e A*:** Em todos os três labirintos, o BFS e o A* encontraram caminhos de custo idêntico (8.0, 27.0 e 73.0), confirmando suas garantias teóricas de optimalidade.

- **DFS e Greedy BFS:** Ambos os algoritmos falharam em encontrar o caminho ótimo nos Labirintos 2 e 3. O DFS, por sua natureza de aprofundamento, encontrou caminhos significativamente mais longos (47.0 e 85.0). A Busca Gulosa, embora guiada por heurística, foi "enganada"pela topologia dos labirintos, encontrando soluções subótimas (43.0 e 81.0). Curiosamente, no Labirinto 1, todos os algoritmos encontraram a solução ótima, um acaso devido à simplicidade do problema.

Eficiência (Nós Expandidos e Memória)

A eficiência, medida pelo número de nós expandidos e pelo pico de memória, destaca o poder das heurísticas.

- **A* vs. BFS:** Comparando os dois algoritmos ótimos, o A* apresenta uma vantagem clara. No Labirinto 2, o A* expandiu apenas 75 nós, contra 151 do BFS, e utilizou significativamente menos memória (102 contra 167 elementos). Isso demonstra como a heurística $h(n)$ direciona a busca de forma eficaz, podando ramos desnecessários. No Labirinto 3, curiosamente, o desempenho de ambos foi idêntico, sugerindo um cenário onde a heurística não ofereceu vantagem, forçando o A* a se comportar de maneira similar ao BFS.

- **DFS:** O comportamento do DFS foi errático. Embora tenha sido o mais rápido e econômico no Labirinto 1, ele registrou o maior pico de memória no Labirinto 2 (243 elementos), pois sua pilha cresceu muito ao explorar um ramo profundo e complexo.

- **Greedy BFS:** A Busca Gulosa foi consistentemente a que expandiu o menor número de nós nos labirintos mais complexos (71 no Lab 2, 98 no Lab 3, embora o DFS tenha sido menor no Lab 3). Isso a torna muito rápida, mas, como visto, ao custo da optimalidade.

Conclusões Gerais da Análise

Os testes confirmam as propriedades teóricas dos algoritmos. O **BFS** é uma escolha segura para garantir o ótimo em problemas de custo unitário, mas sua exploração "cega"o torna ineficiente em espaços de estados grandes. O **DFS** e o **Greedy BFS** são rápidos, mas não confiáveis, pois não garantem a optimalidade, como visto nos Labirintos 2 e 3.

O **A*** se destaca como a solução mais robusta, combinando a garantia de optimalidade do BFS com a eficiência de uma busca guiada por heurística, resultando no melhor equilíbrio entre desempenho e correção.