Instituto Tecnológico de Aeronáutica - ITA
Inteligência Artificial para Robótica Móvel - CT213
Aluno: Gustavo Ferraresi Gottschild (ELE-25)
Data: 01 de abril de 2025

Relatório do Laboratório 2 - Busca Informada

1 Breve Explicação em Alto Nível da Implementação

Os três algoritmos de busca abordados na presente atividade de laboratório são generalizações do algoritmo *BFS* ("*Breadth-First Search*").

No caso, a estratégia de todos os algoritmos de caminho mínimo é:

1. Dado um grafo com custos não-negativos em suas arestas, definir os vértices de início e de fim;
2. Atribuir ao vértice de início o custo nulo e aos demais vértices o custo infinito;
3. Colocar o vértice de início numa fila de prioridades;
4. De agora em diante, repetir o seguinte procedimento até que a fila de prioridades esteja vazia ou que o vértice de fim seja tirado da fila de prioridades:
   (a) Extrair um vértice da fila de prioridades (que será o vértice sendo processado);
   (b) Marcar o vértice extraído como "fechado";
   (c) Descobrir todos os sucessores do vértice sendo processado;
   (d) Colocar todos os sucessores na *priority queue*;
   (e) Executar o seguinte procedimento com todos os sucessores do vértice sendo processado:
      i. Para cada sucessor do vértice sendo processado, ver se seu custo atual é menor que a soma do custo do vértice em processamento com o da aresta que vai dele ao seu sucessor. Caso seja menor, atribuir o vértice em processamento como "pai"(isto é: antecessor ótimo) de seu sucessor e atualizar seu custo como o custo de seu pai mais o custo da aresta mencionada;

   Com o vértice-objetivo sendo retirado da fila, basta usar *backtracking* para reconstruir o caminho ótimo que vai do vértice de início ao vértice de fim.

O que os diferencia entre si é **o que será usado como chave de ordenação na fila de prioridades**: sendo $g(v_1)$, $h(v_1, v_{end})$ e $f(v_1, v_{end})$, respectivamente, o custo do melhor caminho do início até v_1 (que já é seu custo mínimo, pois v_1 foi retirado da fila de prioridades), a heurística calculada entre v_1 e v_{end} e $g(v_1) + h(v_1, v_{end})$. Então:

1.1 Algoritmo de Dijkstra

Utiliza $g(v_1)$ como chave.

1.2 Algoritmo *Greedy Search*

Utiliza $h(v_1, v_{end})$ como chave.

1.3 Algoritmo A*

Utiliza $f(v_1, v_{end})$ como chave.

2 Figuras Comprovando Funcionamento do Código

Abaixo, na Fig. 1, têm-se os resultados de 5 execuções de cada um dos algoritmos estudados: o algoritmo de Dijkstra (1ª coluna), de busca gulosa (2ª coluna) e A* (3ª coluna), bem como, sendo $i \in \{0, ..., 4\}$, a i-ésima execução de cada um dos algoritmos foi executada em exatamente o mesmo espaço de buscas, de forma a facilitar sua comparação.

Vale notar que, no título de cada uma das sub-figuras individuais, estão listados:

- **O nome do algoritmo de busca;**
- **Qual é a execução do algoritmo de busca;**
- **O custo ótimo encontrado pelo algoritmo de busca;**

3 Comparação entre os Algoritmos

Tabela 1 com a comparação do tempo computacional, **em milissegundos**, e do custo do caminho entre os algoritmos usando um Monte Carlo com 100 iterações.

Tabela 1: Tabela de comparação entre os algoritmos de planejamento de caminho.

Algoritmo	**Tempo computacional (ms)**		**Custo do caminho**	
	Média	**Desvio padrão**	**Média**	**Desvio padrão**
Dijkstra	99,859	3,900	80,161	38,670
Greedy Search	109,86	3,43	103,34	59,41
A*	154,13	4,75	80,351	38,693

Analisando-se os resultados:

- **Tempo de execução:** Ao contrário do esperado, o algoritmo de busca gulosa tem um tempo de execução um pouco maior que o do Dijkstra. Ademais, o algoritmo A* teve um tempo de execução notavelmente maior que o do algoritmo de Dijkstra e que o de busca *greedy*. Vale notar que tais anomalias ocorreram por motivo desconhecido (talvez algumas das implementações usadas no método `"a_star()"`tenham usado comandos, invocações ou estruturas de dados demasiadamente lentos, o que contribuiu consideravelmente para as discrepâncias observadas);

- **Custo ótimo:** como esperado, os algoritmos de Dijkstra e A* têm, em média, custos ótimos aproximadamente iguais, os quais são menores que o custo médio do algoritmo de busca *greedy*.

Na Fig. 2, têm-se as saídas da execução de cada um dos algoritmos, corroborando os valores encontrados.

Figura 1: Resultados de cinco execuções de cada um dos algoritmos de busca implementados.

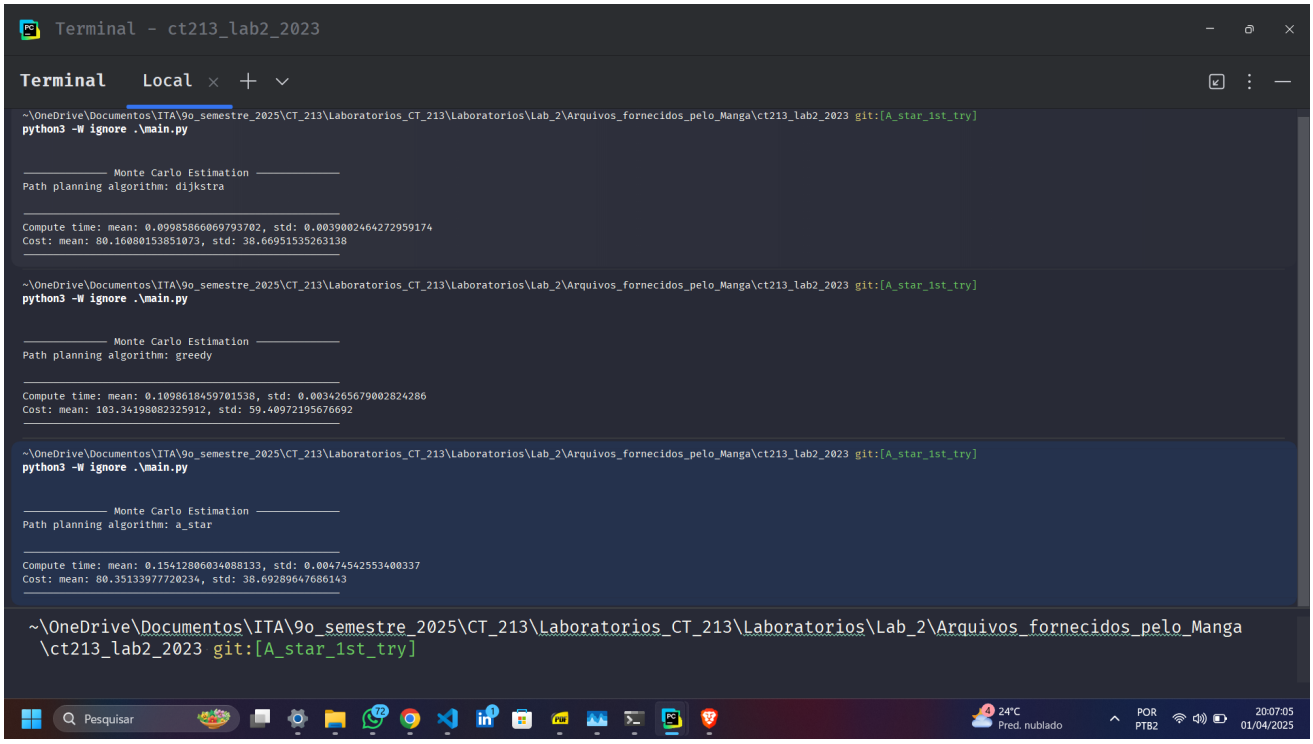

Figura 2: Resultados da estimação de Monte Carlo para os algoritmos analisados.