Trabalho Prático de Inteligência Artificial

Licenciatura em Engenharia Informática

Universidade do Minho
Inteligência Artificial
Grupo 19
2021/2022

Abstract

Este documento procura descrever as tarefas desenvolvidas no âmbito da 2ª Fase do trabalho prático da Unidade Curricular de Inteligência Artificial. O trabalho tem como tema "Métodos de Resolução de Problemas e de Procura". Na primeira fase, pretende-se que seja desenvolvido um sistema de representação de conhecimento e raciocínio com capacidade para caracterizar um universo de discurso na área da logística de distribuição de encomendas. Na fase atual (2ª), pretende-se que o sistema anterior seja atualizado de modo a incorporar um sistema de recomendação de circuitos de entrega de encomendas.

1. Objetivos

Este documento tem como propósito descrever detalhadamente as particularidades da nossa solução para o problema proposto. Como tal, iremos começar por descrever o processo de formulação de problemas utilizado, incluindo a representação do estado do sistema. De seguida apresentamos uma explicação detalhada das estratégias de procura utilizadas (informada e não informada) pelo sistema. Posteriormente, complementamos a nossa solução com uma análise comparativa entre as diferentes estratégias de procura implementadas e, finalmente, concluímos o relatório com alguns comentários acerca do trabalho.

2. Formulação do Problema

A formulação do problema como um problema de pesquisa parte da descrição dos seguintes componentes:

i. Representação do estado inicial (atual)
ii. Representação do estado objetivo
iii. Descrição dos operadores (nome, pré-condições, efeitos)
iv. Custo da solução;

Sendo este um problema de procura, o estafeta tem de executar uma série de ações que o levem do estado inicial ao estado objetivo (final).

Estado Inicial: Centro de distribuição *Green Distribution*, com encomendas para entregar.
Estado Objetivo: Centro de distribuição *Green Distribution*, sem encomendas para entregar.

Para que um estafeta transite entre estados deverá deslocar-se entre as cidades representadas na base de conhecimento.
Operador: *mapa*(local_A, local_B, distância).
Argumentos do Operador: dois locais, p.ex, Aveiro e Beja.
Pré-Condição: No caso particular da nossa implementação, um caminho é representado pelo predicado *mapa*(local_A, local_B, distância). O caminho do local_A para o local_B é válido se existir, na base de conhecimento, o respetivo predicado *mapa.* Da mesma forma, o predicado é válido se e só se o Estado Atual for "local_A" ou "local_B".
Custo da solução: O custo da solução é a soma das distâncias das arestas que constituem o caminho.

3. Estratégias de Procura Implementadas

Neste capítulo descrevemos as estratégias de procura, informada e não informada, utilizadas no trabalho prático.

3.1. Procura Não Informada

3.1.1. Depth-First Search

A procura em profundidade (DFS) segue uma estratégia que prioriza a expansão dos nós mais profundos. Começando pela raiz (da árvore de procura), um ramo é explorado na totalidade, antes de retroceder (*backtracking*).
É uma procura que necessita de pouca memória, sendo que é preferível quando um problema tem muitas soluções. No entanto, não pode ser utilizada em árvores de profundidade infinita, pois pode ficar presa e ramos sem solução.
O tempo de procura depende de $O(b^m)$, onde b é o fator de ramificação (número máximo de sucessores de um nó) e m é a profundidade máxima da árvore de procura. Caso a melhor solução esteja num nível inferior ao nível máximo de profundidade, esta estratégia não é ideal.
O espaço utilizado é linear e depende de $O(bm)$.
Este método de pesquisa não é completo, ou seja, não temos a garantia que irá encontrar a solução, se existir.

3.1.2. Breadth-First Search

A procura em largura (BFS) segue uma estratégia que prioriza a expansão dos nós de menor profundidade. Começando pela raiz (da árvore de procura), são explorados os vértices vizinhos. Posteriormente, para cada um desses vértices, exploramos os seus vizinhos inexplorados, e assim sucessivamente.
É uma pesquisa sistemática, que demora muito tempo e ocupa muito espaço. Dependendo de um fator de ramificação b (número máximo de sucessores de um nó), a pesquisa é completa, se e só se b for finito.
O tempo de procura e o espaço utilizado dependem de $O(b^{d+1})$, onde d representa a profundidade da melhor solução.
No geral, este método apenas deve ser utilizado para problemas pequenos, onde d é relativamente pequeno.
Caso a árvore de procura tenha um fator de ramificação b muito grande, este método necessita de uma quantidade proporcional de memória, dado que mantém todos os nodos, de um determinado nível, em memória.
Este método de pesquisa é completo, ou seja, se existir uma solução temos a garantia que irá ser encontrada.

3.1.3. Busca Iterativa limitada em profundidade

A procura iterativa limitada em profundidade utiliza a procura em profundidade (DFS) como uma sub-rotina:

- Verificar a raiz da árvore de procura.
- Fazer um DFS procurando um caminho de comprimento 1.
- Se não houver um caminho de comprimento 1, fazer um DFS procurando um caminho de comprimento 2.

- Se não houver um caminho de comprimento 2, fazer um DFS procurando um caminho de comprimento 3.
- ...

No geral, esta é a melhor estratégia (não-informada) para problemas com um grande espaço de pesquisa e em que a profundidade da solução não é conhecida.
O tempo de procura e o espaço utilizado dependem de $O(b^d)$, onde b é o fator de ramificação (número máximo de sucessores de um nó) e d representa a profundidade da melhor solução.
Este método de pesquisa é completo, ou seja, se existir uma solução temos a garantia que irá ser encontrada.

3.2. Procura Informada

3.2.1. Gulosa

A estratégia utilizada pelo algoritmo *Gulosa (Greedy)* passa por expandir sempre o nó que parece estar mais perto da solução. Para tomar esta decisão é utilizada uma heurística, que no caso particular deste problema é dada por

$$h(n) = \textit{estimativa da distância em linha reta até ao estado objetivo}$$

Este algoritmo de procura não é completo, ou seja, não temos a garantia que irá encontrar a solução, se existir. O algoritmo também não é ótimo, dado que não encontra sempre a solução ótima.
O tempo de procura, assim como a complexidade no espaço, são $O(b^m)$, onde b é o fator de ramificação (número máximo de sucessores de um nó) e m é a profundidade máxima da árvore de procura. Mais detalhes no capítulo de Resultados.

3.2.2. A estrela (A)*

A estratégia utilizada pelo algoritmo *A** combina uma pesquisa uniforme com uma gulosa, na medida em que procura minimizar a soma do caminho já efetuado com o mínimo previsto que falta até à solução. Esta estratégia evita expandir caminhos mais custosos e como tal, toma decisões baseadas na função:

$$f(n) = g(n) + h(n)$$

A função $g(n)$ representa o custo total, até ao momento e a função $h(n)$ diz respeito ao custo estimado para chegar ao objetivo (estimativa da distância em linha reta até ao estado objetivo).
A complexidade temporal da A* depende da heurística. No pior caso temos $O(b^d)$, onde b é o fator de ramificação (número máximo de sucessores de um nó) e d representa a profundidade da melhor solução.
Este algoritmo é completo e, se a heurística escolhida for admissível, é também ótimo.

4. Resultados

Neste capítulo apresentamos os resultados dos testes de performance em função da estratégia de procura utilizada. Em seguida comentamos e comparamos as diferentes estratégias apresentadas.
Os resultados de performance foram obtidos num sistema com as seguintes características.

Processador	AMD Ryzen 1400 3.2GHz
Memória RAM	16GB
Sistema Operativo	Pop!_OS 21.10 64-bit

O grafo utilizado para representar o espaço de procura é apresentado na figura abaixo.

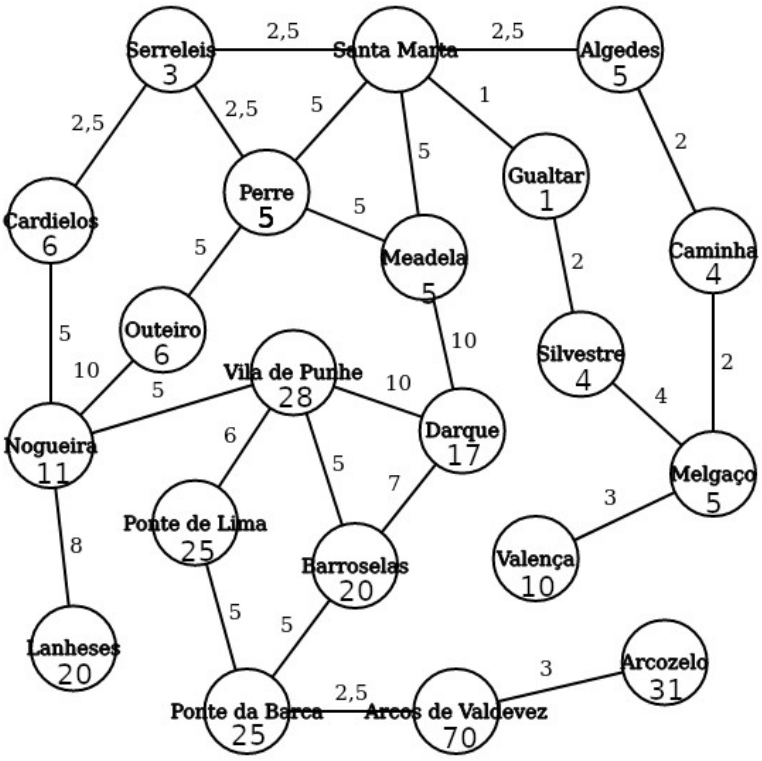

Figure 1: Grafo representativo do problema.

No grafo acima podemos ver que, as arestas representam a distância real entre dois locais enquanto que o valor que se encontra dentro das circunferências diz respeito à estimativa da distância em relação ao local objetivo, neste caso "Santa Marta".
Como os locais de entrega podem ser qualquer freguesia presente no grafo, a solução que encontramos para tirar proveito de algoritmos de procura informada foi, tornar "Santa Marta" o estado final necessário para calcular as estimativas. Deste modo o cálculo é feito após o estafeta se deslocar do estado inicial, para o local de entrega e por fim para o estado final, referido atrás.

A tabela abaixo expõe os resultados dos testes de performance realizados. Os valores de tempo/espaço utilizados são uma média aritmética de 4 iterações de procura. Para simplificar, o limite utilizado no algoritmo de Busca Iterativa limitada em profundidade é de 3. Os 4 percursos escolhidos para realizar este teste foram:

1. Santa Marta → Algedes → Santa Marta
2. Santa Marta → Arcozelo → Santa Marta
3. Santa Marta → Barroselas → Santa Marta
4. Santa Marta → Sarreleis → Santa Marta

Estratégia	Tempo (*ms*)	Encontrou a melhor solução?
DFS	1.28	0/4
BFS	0.29	2/4
BILP	0.2	2/4
Gulosa	0.16	2/4
A*	0.16	2/4

Table 1: Resultados dos testes de performance.

Com os resultados dos testes percebemos que, para esta formulação do problema, o algoritmo *Depth-First Search* (DFS) é o claro derrotado, chegando a ser quase 4 vezes mais lento do que o segundo algoritmo mais lento. Isto deve-se ao facto de, duas das 4 procuras realizadas em resultados muito próximos da raiz, o que faz com que o algoritmo entre em ramos, percorra esses ramos na totalidade e retorne sem resposta. Isto pode acontecer várias vezes, enquanto que num algoritmo de BFS as soluções seriam encontradas na primeira iteração.
Para resolver o *bottleneck* de performance deste algoritmo, temos a Busca Iterativa Limitada em Profundidade (BILP). Esta pode ser uma alternativa, no entanto é um algoritmo inseguro, sendo que, para um utilizador que não tem qualquer noção como o grafo é, torna-se difícil a escolha de um limite de profundidade. Para estes testes escolhemos o limite de 3 e o algoritmo não conseguiu encontrar caminho para a freguesia *Arcozelo*. Concluímos que este algoritmo não é uma opção viável.
O *Breadth-First Search* (BFS) consegue um melhor resultado tanto a nível de tempo de execução como também a encontrar melhor solução tornando esta a opção preferida a nível de algoritmos de pesquisa não informada.
Nos algoritmos de pesquisa informada, há pouca diferença de performance tanto a nível de tempo de execução como a encontrar o melhor caminho, isto deve-se certamente aos valores da heurística serem muito próximos das distancias reais.

5. Conclusão

5.1. Estratégias

Após analisar os algoritmos apresentados, concluímos que a melhor estratégia para a nossa formulação deste problema seria uma combinação do algoritmo de procura não informada BFS com o algoritmo de procura informada A*. Conseguimos juntar a performance dos dois, tirando vantagem da segurança que este último algoritmo traz perante o algoritmo *Gulosa*.
Apesar de, analisando os testes, estes dois algoritmos (A* e *Gulosa*) não serem muito diferentes (tendo o "gulosa" uma pequena vantagem), o A* permite um desenvolvimento do grafo com uma heurística não tão perto do custo real. Admitimos que esta conclusão poderá ser diferente se o grafo aumentar, dependendo se este aumenta em profundidade ou largura.

5.2. Comentário Final

Em tom de conclusão, gostaríamos de realçar a importância que tiveram as estratégias de formulação de problemas no nosso crescimento, não só dentro desta Unidade Curricular, mas também no espectro mais geral. Estamos no geral satisfeitos com a solução encontrada mas, mesmo assim, houveram coisas que não fizemos, ou fizemos menos bem, nomeadamente: não existe a representação de conhecimento imperfeito e a heurística utilizada é sub-ótima (não tem em conta o tempo de viagem ou outros fatores como trânsito, qualidade estradas, etc.).

Referências

Russell and Norvig (2009). Artificial Intelligence – A Modern Approach, 3rd edition.
Costa E., Simões A., (2008), Inteligência Artificial – Fundamentos e Aplicações, FCA.