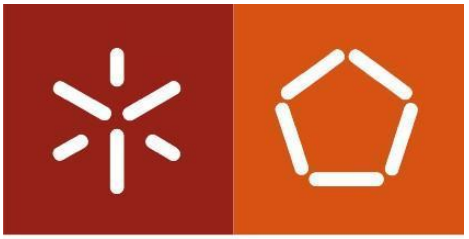

Universidade do Minho

Inteligência Artificial

Engenharia Informática

Universidade do Minho

Grupo 027

A100543, Rodrigo Viana Ramos Casal Novo

A100610, José Afonso Lopes Correia

A100702, Mariana Antunes Silva

A100747, João Paulo Campelo Gomes

2023/2024

Avaliação por pares:

A100543, Rodrigo DELTA = 0

A100610, José DELTA = 0

A100702, Mariana DELTA = 0

A100747, João DELTA = 0

Índice

1. Introdução

No âmbito da Unidade Curricular de Inteligência Artificial, foi-nos proposto o desenvolvimento de funcionalidades de resolução de problemas de pesquisa em grafos baseados no problema de entrega de encomendas da Health Planet.

A *Health Planet* é uma empresa de distribuição que tem como objetivo utilizar os meios de entrega mais sustentáveis e ecológicos para o nosso Planeta Terra. Neste sentido, existem ao dispor dos estafetas diferentes meios de transporte, com diferentes níveis de consumos de energia que podem utilizar.

Assim sendo, precisamos de fazer a formalização do problema procura, e subsequentemente a conceção e implementação dos algoritmos de procura em grafos para o mesmo. Estas funcionalidades serão posteriormente utilizadas para o cálculo de percursos de entregas de encomendas.

2. Descrição do problema

A *Health Planet* é uma empresa que procura diminuir a sua pegada ecológica, como tal acima de ganho monetário esta empresa de entregas procura minimizar a sua poluição sem nunca prejudicar o cliente, tentando por isso arranjar sempre o caminho mais curto entre as várias localidades e montar os percursos de forma a diminuir as emissões de carbono, tentando sempre que possível entregar as encomendas antes do prazo ou pelo menos minimizar o atraso.

3. Formulação do problema

Este problema pode ser representado num grafo. Sendo que este grafo vai representar a zona de entregas da *Health Planet*. Nós definimos este grafo sendo não direcionado, que cada nodo do grafo é uma rua, e as arestas tem informação da distância entre nodos e da condição das ligações (este parâmetro serve para simularmos situações do mundo real, tais como acidentes, trânsito ou chuva). No grafo os nodos são armazenados numa matriz em que cada posição representa as coordenadas de uma rua e as arestas são guardadas num dicionário.

Para abordar este problema, criamos um mapa para representar uma região, começamos por atribuir uma localização a cada posição de uma matriz quadrada com n posições, e de forma a dar algum realismo ao nosso mapa definimos as conexões entre as ruas da seguinte forma: selecionando um primeiro nodo verificamos quais as ligações a outros nodos que são possíveis perante alguns parâmetros (distância em linha reta, por exemplo), após essa verificação selecionamos aleatoriamente entre 1 a 3 dessas ligações, adicionando uma distância e uma condição, a essa conexão, por fim percorremos o grafo e adicionamos ligações se necessário para garantir que não há localidades isoladas.

Como não havia valores de poluição no enunciado, fizemos uma pesquisa no google para encontrarmos um relacionamento entre eles e com isso chegamos aos seguintes valores de poluição por km: bicicleta -> 0; mota ->96; carro->192. Cada um destes veículos possui capacidade, velocidade e penalização por peso igual àquela do enunciado.

3.1. Estado inicial e final

O estado inicial do problema consiste na representação da zona de entregas na forma de grafo, em saber os estafetas e cujos meios de transporte que estão disponíveis, e também quais os conjuntos de encomendas atribuídos a cada estafeta.

3.2. Estado Final

O objetivo do problema é encontrar os caminhos com menor custo na entrega de vários conjuntos de encomendas. Assim sendo, no estado objetivo teremos todas as encomendas entregues.

3.3. Operadores

Os operadores baseiam-se nos movimentos possíveis dos nossos atores, movimentos estes que podem alterar o estado do problema.

Operador “Mover estafeta”

Pré-condição: O estafeta encontra-se num dos nodos do grafo e tem uma rota associada.

Efeito: Conforme a escolha determinada pelo algoritmo, o estafeta desloca-se, ou não, para uma das ruas conectadas.

3.4. Custo da Solução

O custo da solução é a poluição total após a entrega de todas as encomendas.

4. Decisões tomadas

No nosso trabalho tivemos de tomar algumas decisões, nomeadamente:

- ➔ Escolha do veículo a ser utilizado para entrega: como o objetivo é tornar as entregas o mais ecológicas possível, não fazia sentido dar ao cliente a escolha de qual veículo utilizar, então decidimos utilizar um algoritmo descrito abaixo para decidir qual o melhor veículo para a entrega.
- ➔ O volume do veículo apenas é utilizado para determinar o preço do serviço.
- ➔ No nosso sistema não temos clientes, decidimos ter apenas encomendas que representam o cliente em si e as características da encomenda.
- ➔ Em vez de termos um conjunto de estafetas que são distribuídos pelos veículos, os veículos representam os estafetas.
- ➔ Para simular eventos decidimos ter um ficheiro (.txt) que simula acontecimentos desde a passagem do tempo, a alteração de condições das estradas, adicionar novos veículos e adição de novas encomendas.
- ➔ O preço do serviço é calculado com base em quatro fatores sendo este a prioridade (quanto maior a prioridade maior o valor a pagar), o volume que esta ocupa (quanto maior for mais tem de pagar), o somatório de pesos e o transporte(quanto maior for a percentagem do peso total reservada para a encomenda maior será o custo).
- ➔ A classificação do veículo é dada tendo em conta o tempo que demorou a ser entregue e o prazo de entrega, influenciando de forma negativa a avaliação se o tempo exceder o prazo de entrega.
- ➔ Decidimos guardar um grafo em um ficheiro (.txt), para podermos avaliar se a solução dada é a correta e para termos uma melhor demonstração.

5. Implementação de Tarefas

Descrição de todas as tarefas realizadas, bem como de todas as decisões tomadas pelo grupo de trabalho.

5.1. Algoritmos Implementados

5.1.1 Introdução

Neste trabalho achamos que não fazia grande sentido utilizarmos algoritmos não informados, uma vez que iremos percorrer e calcular rotas que passem pelos mesmos nodos e que como o nosso grafo assenta sobre localizações é possível estimar distâncias sem nunca termos lá ido.

Com isto em mente criamos uma matriz de 4 dimensões que contém informações referentes a fazer travessias entre os seus nodos, por exemplo matriz[0][0][10][12]. condição dá-nos a estimativa da condição entre a cidade localizada na posição (0,0) e a localizada na (10,12). Esta matriz vai sendo atualizada à medida que fazemos travessias entre os seus nodos de forma a aproximar o valor estimado do valor real a cada iteração. Para além da condição esperada esta matriz também tem informação referente à distância esperada e à fidelidade dos seus dados.

5.1.2. Algoritmos de procura informada

5.1.2.1 Clarke-Wright

Antes de podermos calcular o caminho mais curto entre dois nodos, precisamos de saber que caminho é que queremos calcular em primeiro lugar. Então utilizamos uma adaptação do algoritmo de Clarke-Wright para calcular as encomendas e a ordem destas que cada veículo tem de percorrer.

Para esse efeito começamos por calcular quais os pares de encomendas que mais distancia poupam, quando comparados a ir entregar uma e voltar para o armazém (esta informação é com base na matriz falada anteriormente), guardando esta informação num dicionário onde a chave é o id da encomenda pela qual começamos. Depois verificamos se há alguma encomenda que seja considerada urgente e se houver invés de priorizarmos o par que poupa mais começamos por essa encomenda. Após ter selecionado a encomenda inicial vamos buscar sempre a travessia que mais poupa evitando fazer travessias para encomendas que já entregamos. Durante este processo vamos sempre atualizando o peso da nossa

encomenda. Quando encontramos um nodo que não conseguimos chegar a tempo ou quando a capacidade do nosso veículo é excedida, exploramos a possibilidade de trocar para um veículo que o consiga fazer, e também exploramos a rota caso terminemos ali a rota desse veículo e enviarmos outro veículo para as restantes encomendas. Este processo é repetido até termos visitado todas as encomendas ou até ter ficado sem veículos. No final todas as rotas são comparadas e é selecionada a que deixa menos clientes em risco de haver atrasos e a que polui menos.

5.1.2.2 Selecionador de rotas

Apesar de a função descrita nos dar um conjunto de rotas próximo do ótimo, nem todas as rotas que ele nos dá valem a pena serem entregues já, por exemplo se uma das rotas entregar apenas um item que está muito longe do prazo de entrega podemos esperar que cheguem mais encomendas ou veículos que poluem menos antes de a enviar. Por isso, temos um algoritmo que atribui prioridades a rotas de acordo com a sua capacidade e tempo que pode esperar até colocar encomendas em risco, e depois marca as mais prioritárias para envio, removendo os veículos necessários da lista de veículos disponíveis.

5.1.2.3. Procura A*

Após termos a heurística definida e tendo os destinos definidos para uma rota, percorremos o grafo utilizando o algoritmo A-estrela.

Decidimos usar este algoritmo pois, sendo um algoritmo de pesquisa informada que tem em conta tanto o custo de deslocamento (no nosso caso, distância /condição da estrada) como a heurística(como explicado acima), permite-nos alcançar a solução ótima para o nosso problema.

Sendo um ambiente variável, a solução ótima poderá ser alterada durante o decorrer da entrega, como tal, sempre que a heurística de um nodo visitado é alterada, o programa reiniciará este algoritmo a fim de encontrar uma solução melhor.

5.1.2. Algoritmos de procura não informada

Numa fase posterior decidimos implementar tanto o algoritmos BFS como o algoritmo DFS, para confirmarmos as nossas expectativas de que estes algoritmos não chegaram à resposta ideal. Após diversos testes, chegamos à

conclusão pretendida. É importante referir que o algoritmo DFS atingiu valores extremamente elevados, não só por ser de pesquisa não informada, mas também porque, sendo o nosso grafo cíclico, este não retorna a nodos anteriores para avaliar um melhor cenário.

6. Conclusão

Apesar de o nosso algoritmo não nos garantir o valor ótimo, durantes os nossos testes ele sempre nos deu soluções perto dele, o que dado a escala do problema é ideal. Tendo tudo o referido acima em conta, achamos ter conseguido cumprir com os objetivos propostos. Após comparação e devido a estes não serem facilmente escaláveis, optamos por não usar algoritmos não informados neste projeto, decidimos então utilizar os algoritmos de procura informada, nomeadamente o algoritmo Clarke-Wright, o algoritmo Selecionador de rotas e o A*.