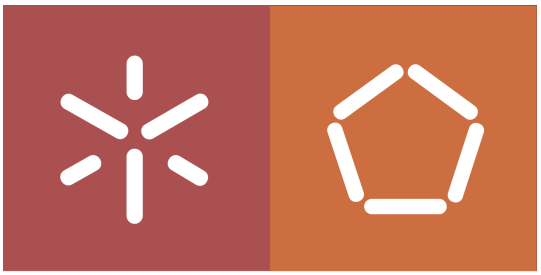

Universidade do Minho
Licenciatura em Engenharia Informática

Inteligência Artificial
Grupo 44

Gonçalo Daniel Machado Costa - **A100824**
José Eduardo Silva Monteiro Santos Oliveira - **A100547**
Marta Raquel da Silva Rodrigues - **A100743**
Pedro Afonso Moreira Lopes - **A100579**

Índice

Avaliação 2
Introdução 2
Descrição do Problema 2
Representação da cidade de Braga 3
Formulação do Problema 4
Funcionalidades Implementadas 5
Algoritmos Implementados 6
Pesquisa não informada 6
Procura em largura (BFS) 6
Procura em Profundidade (DFS) 7
Custo Uniforme 7
Pesquisa informada 7
Gulosa (Greedy Search) 8
A* (A estrela) 8
Resultados 8
Conclusão 9

Avaliação

Gonçalo Daniel Machado Costa - A100824 → **Delta** = 0
José Eduardo Silva Monteiro Santos Oliveira - A100547 → **Delta** = 0
Marta Raquel da Silva Rodrigues - A100743 → **Delta** = 0
Pedro Afonso Moreira Lopes - A100579 → **Delta** = 0

Introdução

Este projeto foi realizado no âmbito da unidade curricular de Inteligência Artificial, e teve como objetivo o desenvolvimento de problemas através da conceção e implementação de diversos algoritmos de procura, tendo em particular atenção à sustentabilidade. Deste modo, neste relatório, serão apresentadas e justificadas as diversas estratégias de procura e todo o restante trabalho, como a descrição e formulação do problema e as decisões tomadas.

Descrição do Problema

A empresa Health Planet tem como objetivo utilizar um meio de entrega de encomendas mais sustentável/ecológico, fazendo-se dispor de diversos meios de transporte, possuindo diferentes níveis de consumos de energia, tornando uns mais ecológicos que outros. Tendo em consideração contextos reais, existirão dois tipos de entregas, de uma encomenda e um serviço, quando um estafeta realiza várias entregas numa única. De qualquer modo, quando o estafeta realiza a encomenda, deve obter uma caminho para o seu destino no transporte adequado para diminuir a sua pegada ambiental, tentando sempre evitar atrasos, para poder cumprir com os prazos limite impostos pelo cliente. O problema também deverá ser capaz de fornecer todas as informações necessárias para uma entrega, como preço da entrega, valor que irá depender do transporte e da encomenda em si.
Para obtermos os percursos capazes de realizar as entregas de encomendas, procuramos aplicar diferentes algoritmos de pesquisa sobre grafos. De forma complementar, avaliamos a performance e eficiência de cada um deles.

A Health Planet faz entregas nas ruas de uma cidade, sendo assim, decidimos aplicar o problema a algo que nos é mais familiar, a cidade de Braga. Por uma questão de simplificação, decidimos reduzir o número de freguesias da cidade de Braga a ser consideradas para as localizações das ruas onde as encomendas poderão ser entregues. No mapa abaixo podemos ver as freguesias ou agrupamentos escolhidos:

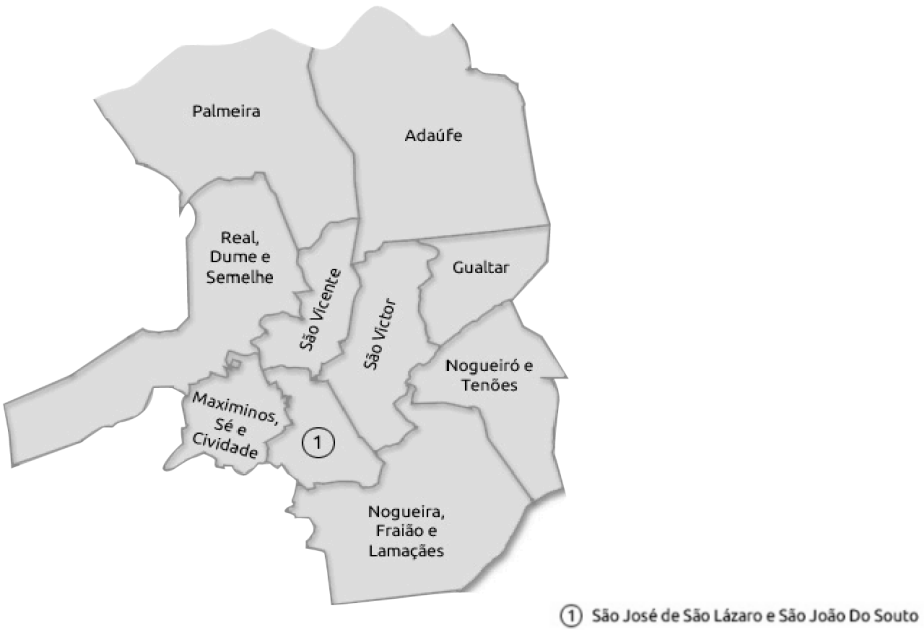

Figura 1. Alguns agrupamentos de freguesias da cidade de Braga

Ruas escolhidas dentro das Freguesias selecionadas:

1. Rua da Confeiteira, Palmeira
2. Rua de Redondo, Adaúfe
3. Rua de São Martinho, Dume
4. Rua 5 de Outubro, Real
5. Rua da Universidade, Gualtar
6. Rua Santa Margarida, São Vicente
7. Rua de São José, São Vitor
8. Rua do Raio, São Vitor
9. Avenida Dom João II, Nogueiro
10. Rua do Fujacal, São Lázaro
11. Rua Joãozinho Azeredo, Maximinos
12. Rua da Igreja, Nogueira
13. Rua da Senra, Lamaçães

Representação da cidade de Braga

De forma a desenvolver um caso prático e pôr à prova os algoritmos que implementamos, definimos o seguinte grafo com a informação das ruas e freguesias (nodos) e dos caminhos (arestas) que as unem:

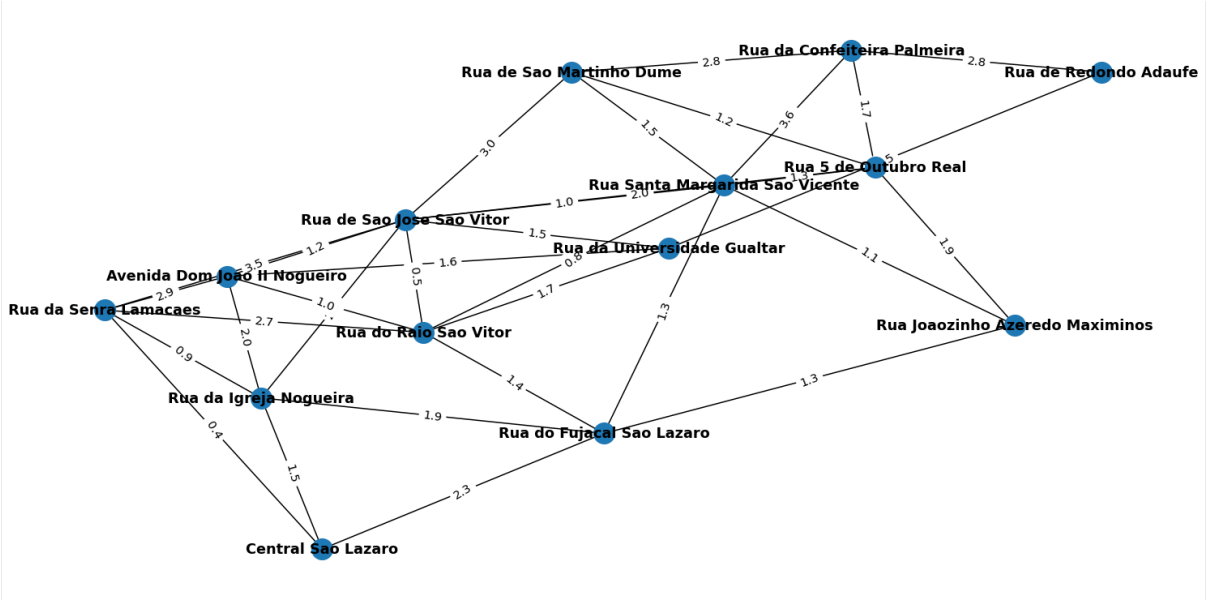

Figura 2. Grafo das ruas escolhidas

Obtivemos, então, um grafo não orientado e pesado, cujo peso de cada aresta corresponde às distâncias entre as ruas que essa liga. A central da Health Planet, encontra-se representada como “Central, São Lazaro” e é a partir dela que todos os percursos se inicializarão.

Formulação do Problema

Os algoritmos implementados permitiram a obtenção de rotas de entregas ótimos, dada uma lista de entregas de encomendas, a realizar pode ser representado por um problema de pesquisa em grafo, cujos nodos representam várias moradas para as entregas e a central da Health Planet. Visto que, se pretende minimizar a pegada ecológica, tentando respeitar ao máximo o prazo limite, podemos, assim, atingir esse objetivo e a otimização procurando minimizar o tempo total de viagem ao realizar o caminho mais curto, tendo em conta o mapa onde se encontra e as restrições de cada veículo.

Uma vez que o problema se encontra num ambiente determinístico e completamente observável, no qual o agente “sabe” o estado em que se encontrará e as soluções são sequências, podemos afirmar que este problema é de estado único. A formulação do problema pode ser completada com as seguintes informações:

Estado inicial: Central, em São Lázaro, que representa a sede da Health Planet.

Estado Objetivo: Qualquer rua onde se pretenda entregar encomendas.

Operadores: Deslocação entre cada rua, isto é, travessia entre quaisquer ruas que estejam ligadas por arestas do grafo.

Solução: A solução ideal é o caminho (sequência de freguesias) que tem um custo menor, ou seja, menor soma de distâncias para fazer a entrega, que comece na posição inicial e termine numa posição objetivo.

Custo da Solução: Soma das distâncias entre as ruas por onde se passa para fazer as entregas.

Funcionalidades Implementadas

Para realizar esta fase do trabalho prático, implementámos os algoritmos de procura solicitados no enunciado (DFS, BFS, Custo Uniforme, Gulosa e A*) que serão essenciais para obter um caminho entre um local e o destino e cuja formulação já foi mencionada.

Contudo, o pedido não era simplesmente obter um caminho entre uma freguesia e a sede, mas sim, obter um percurso completo para o estafeta, isto é, um ciclo cujo local de partida é a sede e que passa no ponto de entrega. Definimos uma destas funções para cada tipo de algoritmo. Assim, caso um serviço possuísse apenas uma única encomenda, usaria um método direto para esse cálculo, e se a encomenda indicada se encontrasse num serviço com múltiplas encomendas usaria outra forma que calcularia o percurso dando prioridade sempre a encomenda com menor prazo fornecido pelo cliente no momento do registo da encomenda. Sabendo que o custo de cada percurso obtido era apenas a distância total deste, definimos uma função que para a encomenda indicada, conhecendo a distância que essa encomenda terá de percorrer, determina que meio de transporte o estafeta deverá usar com base no tempo que irá demorar a percorrer esse percurso.

Inicialmente, definimos funções que permitem transformar documentos para obter instâncias de clientes, estafetas e encomendas, para facilitar a inicialização do programa. Como funcionalidade extra, permitimos que o utilizador, tanto para os clientes como para estafetas, registe novas entidades e visualize as listas registadas dos mesmos, indicando o nome e rua caso pretenda registar um cliente e um nome caso pretenda registar um estafeta. No momento em que as encomendas são “carregadas” estas serão agrupadas em serviços em função da sua data de entrega, verificando sempre se o seu peso passa do máximo possível a ser transportado. Caso isso aconteça, a nova encomenda que passa do limite é agrupada noutro serviço para essa mesma data.

Também é possível criar novas encomendas, caso assim pretenda, fornecendo informações sobre o id do cliente a quem ela está associada, o peso, o volume, o prazo limite e a data que pretende que esta seja entregue. Esta passa a estar registada num dicionário com todas as encomendas. Aí permitimos que se visualize um plano inicial da entrega para essa encomenda assumindo que é entregue no momento e não é colocada em nenhum conjunto de encomendas. Esta funcionalidade também permite neste instante escolher um estafeta que poderá fazer a encomenda, no entanto, não implica que no momento real da entrega desta encomenda, seja esse estafeta a entregá-la, nesses casos decidimos aleatorizar quem a entrega.

Como foi mencionado no enunciado, definimos ainda funções que permitem gerar um percurso com várias paragens, ou seja, dado uma lista de encomendas funções irão gerar um percurso que poderá ser seguido pelo estafeta e que irá passar por todos os locais de entrega necessários. Definimos uma função para cada algoritmo já mencionado. De forma a determinar qual o melhor circuito, definimos um predicado que para uma dada encomenda aplica todos os predicados definidos que permitem gerar um percurso completo e que compara os resultados obtidos para obter o melhor percurso a nível de distância. Quando se obtém um percurso, obtêm-se também o meio de transporte que deverá ser usado e o tempo que irá demorar a percorrer o circuito. Calcula-se ainda o preço da encomenda, em função do meio de transporte, do peso e do volume pelo cliente.

Finalmente, atribui uma classificação ao estafeta que realizou a entrega em função do tempo gasto para a entrega, ou seja, por quanto mais tempo passar da data limite estabelecida pelo cliente pior é a sua classificação.

Cada estafeta pode decidir o que valoriza mais, a sua avaliação ou a sua pegada ecológica. Caso a avaliação dele seja mais alta, ele está disposto a demorar mais tempo a realizar a entrega para garantir que a sua pegada ecológica é a mais baixa possível. Porém, se a avaliação dele for mais baixa ele pode considerar utilizar um transporte mais rápido para evitar atrasos e assim garantir uma melhor classificação.

Adicionalmente, permitimos que se visualizem as ruas (nodos) que a empresa cobre nas suas entregas, ver os caminhos possíveis entre essas ruas (arestas) e, também visualizar o grafo produzido num ambiente gráfico para uma melhor perceção do grafo produzido (como é possível ver na figura 2).

Implementamos também algo que permitisse guardar os novos dados registados, caso estes tenham acontecido. Decidimos fazer este método à parte por receio que o programa ficasse muito pesado sempre que houvesse um registo, permitindo assim fazê-lo a qualquer instante.

Algoritmos Implementados

Pesquisa não informada

Neste projeto, implementamos três algoritmos de pesquisa não informada:

- BFS
- DFS
- Custo Uniforme

Estes algoritmos encarregam-se de tentar encontrar o estado objetivo sem ter em consideração os custos das arestas quando escolhem o próximo nó a ser visitado.

Procura em largura (BFS)

Efetuar uma pesquisa em largura primeiro (BFS) consiste em começar por visitar um nodo do grafo, visitar todos os seus adjacentes, avançar para o próximo nível de profundidade (para um nodo adjacente ao anterior) e efetuar o mesmo processo em cada nodo dessa mesma profundidade até encontrarmos uma solução (nodo pretendido estar na lista de nodos visitados). Tornando-se um método de pesquisa muito sistemático, trazendo vantagens quando as arestas têm custo muito baixo. No entanto, trata-se de um algoritmo de pesquisa que, normalmente, demora muito tempo e que sobretudo ocupa muito espaço, uma vez que são necessárias estruturas mais pesadas para guardar todos os caminhos, como por exemplo através do uso sets, e para efetuar a pesquisa de filas (*queues*) para armazenar a ordem de pesquisa dos nodos. Sendo que, no nosso problema, o tamanho do grafo é muito elevado, este algoritmo é, evidentemente, pouco eficiente.

Procura em Profundidade (DFS)

Formalmente, um algoritmo de pesquisa em profundidade (DFS) realiza uma procura não informada que avança para através da expansão do primeiro nodo do grafo, pesquisando o nodo objetivo num nível de profundidade cada vez maior, até que o alvo da pesquisa seja encontrado ou até encontrar um nodo que não possui mais nodos adjacentes por visitar. Posto isto, retrocede e começa no próximo nodo. São usadas estruturas simples para guardar o caminho visitado, sendo por isso, necessária pouca memória. O seu uso é também vantajoso para problemas com múltiplas soluções, pois a probabilidade de estar a procurar um caminho possível aumenta. No entanto, não garante que é retornada a melhor solução e torna-se pouco eficiente em grafos com uma profundidade elevada e poucas soluções. Contudo, a nossa implementação garante que não existem ciclos no processo de pesquisa impedindo que nodos que já tenham sido visitados sejam outra vez explorados no processo da pesquisa.

Custo Uniforme

Um algoritmo de custo uniforme explora os percursos expandindo para nodos com o menor custo acumulado, para tal guarda o custo total do caminho do estado inicial até esse nó. Torna-se semelhante à procura em largura se todos os custos das arestas forem iguais. De igual modo, também mantém uma fila de prioridade (*queue*) organizada de acordo com o custo total necessário para alcançar os nodos a partir do inicial, o que faz com que seja um algoritmo que ocupa muito espaço, visto que são usadas estruturas pesadas para guardar os caminhos. Porém, o resultado deste algoritmo é sempre um caminho ótimo, desde que o grafo seja finito.

Pesquisa informada

As estratégias de procura informada implementadas neste projeto foram:

- Gulosa (greedy)
- A*

A técnica de pesquisa informada baseia-se no conhecimento específico do problema para dar informações extras para a solução do problema. A pesquisa informada pode ser vantajosa em termos de custo, onde a otimização é alcançada com custos mais baixos de pesquisa.

As informações extra são conhecidas como heurísticas e, para o nosso problema, foram utilizadas heurísticas baseadas na **distância em linha reta** entre dois nodos do grafo. Deste modo, os valores de cada heurística, tendo em base o local onde o percurso se inicia, já se encontram definidas desde o início visto que o mapa para os percursos será sempre estático. Esta heurística foi escolhida, visto que, todos os princípios a cumprir podem ser calculados a partir da distância, isto é, o custo de uma entrega é definida através da soma das distâncias entre as arestas atravessadas e, igualmente, a decisão feita para suportar a escolha de um meio de transporte é tomada tendo em consideração o tempo

tomado e, sendo que os meios de transporte tem uma velocidade fixa, torna bastante fácil o cálculo para o tempo necessário através da distância e da velocidade dos veículos.

Gulosa (Greedy Search)

Um algoritmo de pesquisa gulosa escolhe, em cada iteração, o nodo que parece estar a uma menor distância do objetivo final, utilizando para tal uma heurística. O tempo de procura torna-se bastante reduzido caso a função heurística for boa. O nodo escolhido passa a fazer parte da solução que o algoritmo constrói. No entanto, isto provoca uma maior suscetibilidade a falsos começos e portanto, não garante que encontra sempre a melhor solução. Todos os nós são mantidos em memória e é necessário detectar estados repetidos, o que acaba por ocupar muito espaço na memória.

A* (A estrela)

A pesquisa A* evita expandir caminhos que são dispendiosos, combinando para tal a pesquisa gulosa com a uniforme, minimizando a soma do caminho já efetuado com o mínimo previsto que falta até a solução. É, por isso, um algoritmo ótimo e completo, caso a sua heurística for admissível, no entanto tem uma complexidade no tempo exponencial e mantém todos os nodos em memória, o que acaba por ocupar muito espaço.

Resultados

Utilizando um percurso com uma paragem singular na Rua da Igreja em Nogueira, e partindo da Central da Health Planet em Lamaçães, estes foram os resultados conseguidos por cada algoritmo:

Estratégia	Tempo	Espaço	Custo	Solução Ótima?
DFS	O(b^m)	O(b*m)	19.4	Não
BFS	O(b^d)	O(b^d)	1.5	Não
Custo Uniforme	O(b^m)	O(b*m)	1.3	Sim
Greedy	Pior caso:O(b^m) Melhor caso:O(bm)	Pior caso:O(b^m) Melhor caso:O(bm)	1.5	Não
A*	Exponencial	Nr de Nodos	1.3	Sim

Conclusão

Com o desenvolvimento deste projeto, pudemos observar que, tal como lecionado nas aulas, o algoritmo de pesquisa A* é o mais eficaz entre todos os algoritmos implementados, não só por chegar sempre à solução ótima, mas por, na maior parte das vezes, ser o método com menores complexidades tanto a nível temporal como espacial. Por outro lado, por vezes conseguimos obter um percurso ótimo utilizando o algoritmo de pesquisa Gulosa. Contudo, dentro dos algoritmos de pesquisa não informados é improvável obter uma solução ótima através do BFS e DFS, dado que estes pretendem apenas obter um caminho para o objetivo, não tendo em atenção o custo das arestas, sendo, por isso, mais pertinentes para a procura de caminhos em grafos não pesados, o que acaba por não ser o caso do problema proposto. Ainda nos algoritmos de procura não informada, podemos também constatar que o algoritmo de custo uniforme é dos algoritmos de procura não informada que devolve melhores resultados, dado que utiliza os custos para ponderar o caminho a percorrer. Assim chegamos à conclusão que para uma pesquisa cujo objetivo é chegar à solução ótima devemos usar o algoritmo A* ou o Custo Uniforme, apesar de também ser bom o método da gulosa não é tão constante quanto à otimalidade da solução. Os restantes algoritmos podem obter boas soluções, mas é muito menos provável que tal aconteça.

Concluindo, este projeto permitiu-nos aprofundar e pôr em prática as técnicas de formulação de problemas e a sua posterior resolução, bem como a implementação de alguns algoritmos de procura lecionados nas aulas, aproximando-as de um contexto mais realista. Vale realçar que ao longo do desenvolvimento deste projeto enfrentamos diversas dificuldades, destacando, por exemplo, a dificuldade em atribuir as encomendas a estafetas específicos, para resolver tal problema aleatorizamos as atribuições das mesmas.

Apesar disso, acreditamos que, de um modo geral, conseguimos atingir todos os objetivos propostos pelo enunciado e estamos bastante satisfeitos com os resultados que atingimos.