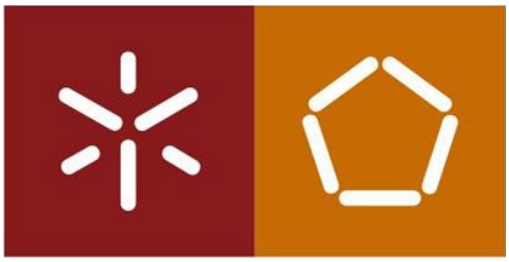


Universidade do Minho
Licenciatura em Engenharia Informática

Inteligência Artificial

Relatório

Trabalho Prático 2ª Fase

05/01/2022

André Gonçalves Pinto - a93173

Rui Pedro Alves - a93252

Índice

Índice de Imagens

Introdução

Este relatório foi realizado no âmbito do projeto da UC Inteligência Artificial, no qual nos foi pedido para implementar um sistema de distribuição de encomendas.

Nesta segunda fase de entrega, utilizando a linguagem de programação JAVA, e recorrendo a vários algoritmos de procura, implementamos um sistema de entrega de encomendas.

Nesta aplicação podemos decidir qual algoritmo queremos usar para realizar as entregas dos pedidos. O utilizador pode depois observar quais as trajetórias que cada algoritmo produz.

Esta aplicação é excelente para verificar as diferenças entre as pesquisas informadas e não informadas.

Base de Dados

A nossa implementação dispõem de vários componentes que formam o sistema, Pessoas, onde encontramos o **Cliente** (com rua associada) e o **Estafeta** (com o veículo associado), a classe **Veículo**s, com 3 subclasses, Bicicleta, Mota e Carro, a classe **Rua** que irá representar os nodos do grafo (mapa), temos ainda a classe **Produto** e o **Pedido** que irá reunir todas as outras classes como atributos.

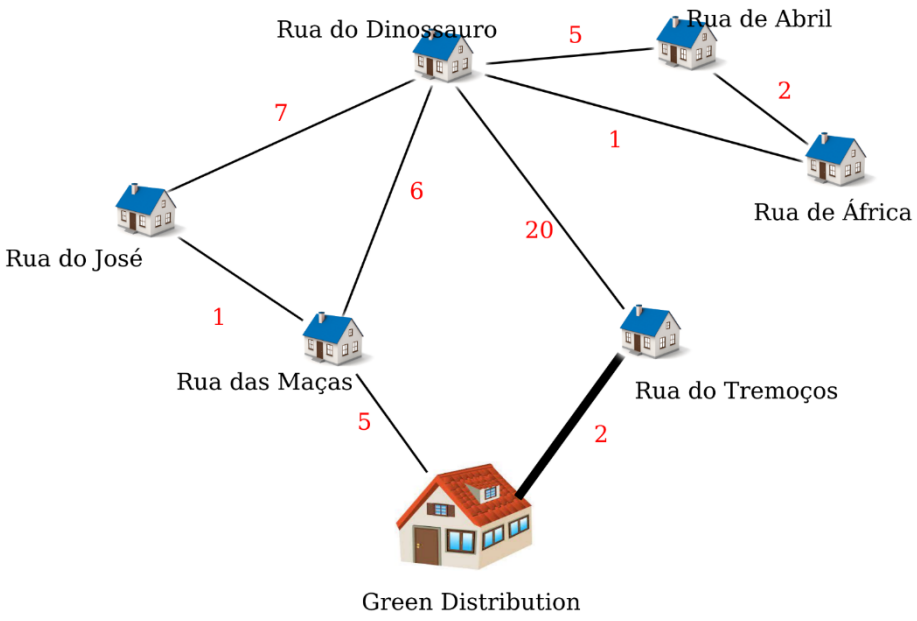


Figura 1 - Grafo

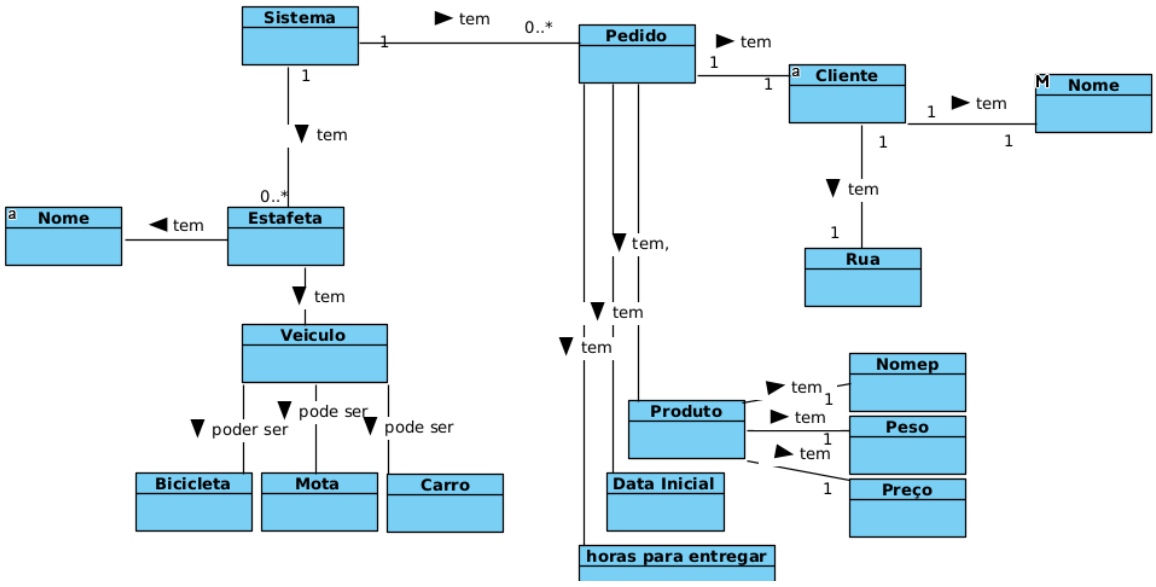


Figura 2 - Modelo de Domínio

Problema

O sistema tem como objetivo resolver Pedidos, efetuando as entregas dos mesmos. Um grafo irá representar todas as Ruas presentes no sistema (mapa) e, através de um Estafeta, onde o veículo será escolhido através de critérios de tempo e/ou eco, a entrega será realizada. A escolha do percurso será feita através do uso de algoritmos de procura.

Na escolha do melhor veículo, primeiramente verificamos, para cada veículo, se consegue transportar o respetivo peso da encomenda e se a velocidade a que este se desloca é suficiente para entregar a encomenda a horas. De seguida serão apresentados os melhores estafetas, em função do seu veículo, para a realização da entrega, o estafeta mais ecológico, priorizando sempre o veículo mais ecológico, e o estafeta mais rápido, priorizando o tempo de entrega sobre a ecologia do veículo.

Estado Inicial

Um estafeta começa a sua procura sempre a partir da rua Green Distribution, sendo que qualquer algoritmo suporta diferentes pontos de partida.

Estado Final

O destino final será a morada associada ao cliente que realizou o pedido. Tendo o caminho sido descoberto agora, o estafeta tem de regressar à rua Green Distribution. Como o caminho foi descoberto por um algoritmo é apenas necessário este ser revertido para o estafeta retornar à origem.

Algoritmos de Procura

Depth-First Search

O algoritmo de procura em profundidade caracteriza-se por começar num nodo raiz, neste caso (em grafo) um nodo dado como raiz, e explora tanto quanto possível cada um dos seus ramos, antes de retroceder (*backtracking*). Este algoritmo pode sofrer de se encontrar em um caminho infinito.

O algoritmo usa significamente menos memória que a pesquisa em largura pois este necessita memória proporcionalmente à profundidade do grafo.

Na nossa implementação usamos uma Stack<Rua> para guardar qual o caminho para chegar ao nosso nodo de destino. Sempre que o algoritmo faz backtracking é feito um pop da Stack, dessa forma temos o estado Inicial Final completado. O caminho percorrido é representado como uma Stack<Rua> em que o primeiro elemento corresponde ao nodo destino e o último ao nodo origem.

Breadth-First Search

O algoritmo de procura em largura começa pelo nodo raiz e explora todos os nodos adjacentes, para cada um desses nodos adjacentes, explora os seus adjacentes inexplorados e assim por diante, até que ele encontre o nodo final da procura.

Este algoritmo é completo, desde que haja memória suficiente e não se encontre na procura dentro de uma árvore infinita.

Para obter o caminho pretendido para o estafeta usamos uma estrutura **pathway** = Map<Rua,(Rua,Cost)> em que guardamos o nodo descendente e o nodo ascendente. Dessa forma quando obtemos a rua destino encontrada começamos a ascender na **pathway** até encontrarmos o nosso nodo origem.

Busca Iterativa Limitada em Largura

Este algoritmo tem o mesmo funcionamento do algoritmo de procura em largura, porém este está limitado a um certo nível, percorrendo o grafo apenas até esse nível.

Consideramos apenas os níveis, 0 sendo o nível do nodo de origem e o nível 1 com todas as ruas adjacentes ao nodo origem.

Gulosa

Um algoritmo guloso irá sempre escolher a opção (nodo) que parece mais promissora naquele mesmo instante.

Na nossa implementação o algoritmo guloso irá seguir sempre pelo nodo com menor distância ao nodo destino completamente ignorando a distância que se encontra do nodo inicial.

Este algoritmo pode não ser completo se ficar preso em caminho infinito.

Não é o mais eficaz pois na nossa implementação o algoritmo procura sempre a rua que está mais perto do nodo destino descartando a sua distância ao nodo inicial. Isto pode ter graves consequências pois uma rua que esteja perto do destino pode ter um custo de visita enorme.

A*(A Estrela)

O algoritmo AEstrela deriva do algoritmo Gulosa tendo uma diferença enorme que o leva a encontrar sempre o melhor caminho.

Neste algoritmo, no cálculo do índice de um nodo é também factorizada a distância a que se encontra do nodo inicial, assim como a distância que tem ao nodo destino.

A cada iteração do ciclo procuramos sempre o nodo do **OPENSET** com o menor índice, indicando que este é o que está mais perto da solução.

Tendo sido percorrido e por sua vez atualizado todos os nodos adjacentes no **OPENSET** este nodo é inserido no **CLOSESET.** Na próxima iteração é procurado o menor nodo do **OPENSET.**

Este algoritmo é completo e garante-nos sempre a escolha do melhor caminho possível.

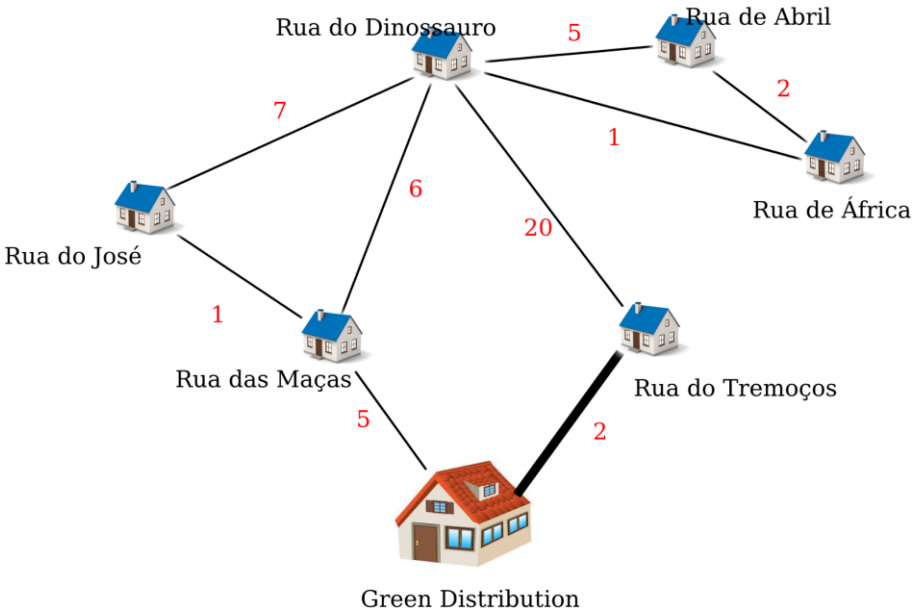
Resultados

Apresentamos em seguida uma análise comparativa de todos os algoritmos utilizados para um caso específico.

Em diferentes cenários e a cada nova execução dos algoritmos os dados obtidos irão ser distintos dos em baixo apresentados, com esta análise podemos observar os resultados vão depender de qual algoritmo foi utilizado.

Cenário de teste:

Ponto de partida: Green Distribution | **Objetivo:** Rua de África.



Algoritmo	Tempo de execução (ms)	Número de Nodos visitados	Custo (Kms)	Encontrou a melhor solução?
DFS	2	4	23	Não
BFS	2	7	23	Não
Busca Iterativa lvl 2	1	5	None	Não
Gulosa	1	4	23	Não
A*	1	6	12	Sim

User Interface

Compilação do programa -> executar o comando "make" no terminal

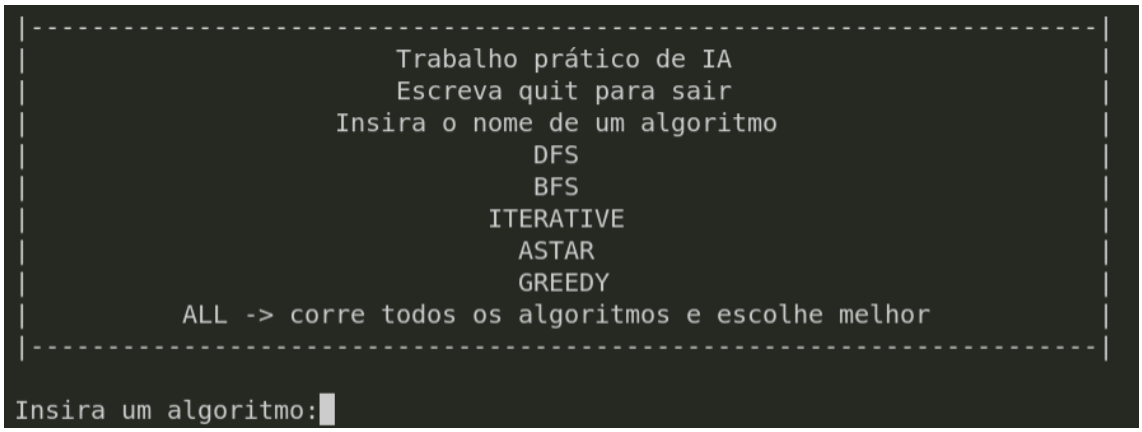


Figura 3 - Menu UI

Exemplo do pedido do Zé usando Breadth First

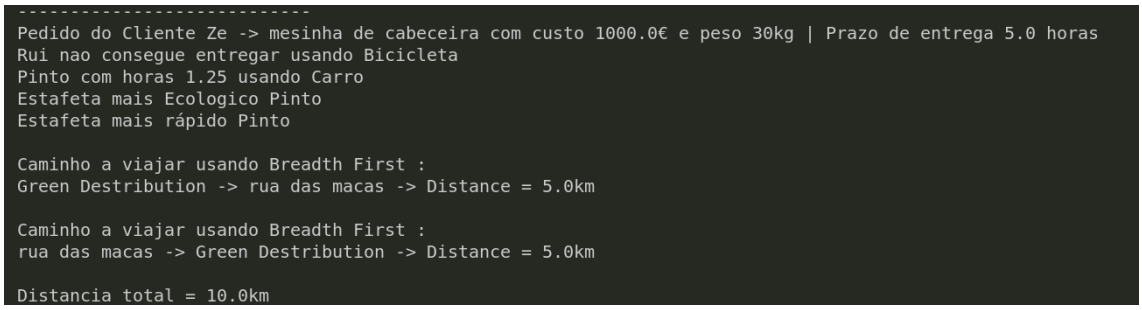


Figura 4 - BFS exemplo

OBS: Estafeta Rui não consegue entregar pois a bicicleta não suporta um peso de 30kg

Exemplo do pedido do Tomás usando Depth First

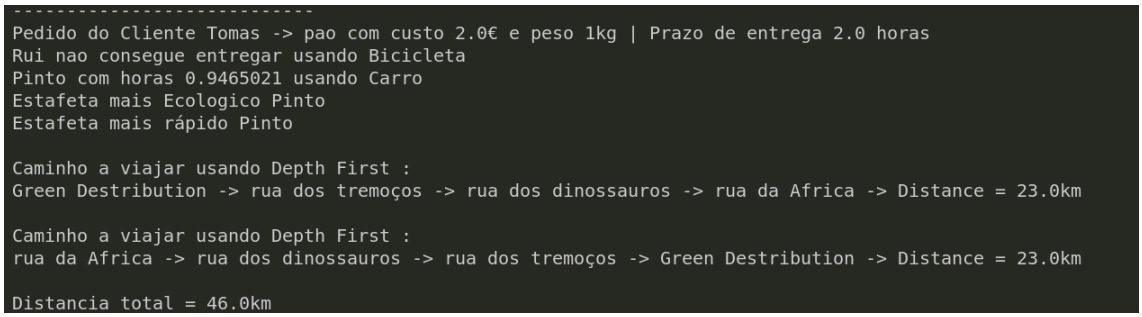


Figura 5 - DFS exemplo

OBS: Estafeta Rui não consegue entregar pois não entrega durante o prazo limite de 2 horas

Conclusão

Nesta segunda fase de entrega fizemos uma implementação de um sistema de entregas, foram criadas todas as classes necessárias ao sistema e implementados os vários algoritmos de procura exigidos, reparamos, comparando os vários resultados obtidos após a execução dos algoritmos, que o algoritmo A*, de procura informada, se realça sobre todos os outros, sendo este, na maioria dos casos, o algoritmo que encontra a melhor solução.

A pesquisa A* destaca-se sobre as outras devido a cada rua ter uma coordenada **x** e **y** associadas**,** caso não tivéssemos estes dados teríamos de optar por usar uma pesquisa não informada.

Cremos, que apesar de ter aumentado muito o tempo de desenvolvimento, a escolha de Java como linguagem de programação para esta fase foi correta. Ao escolher esta linguagem, todos os algoritmos tiveram de ser implementados de raiz, assim como a maneira de guardar toda a informação necessária, em contraste com a linguagem Prolog, onde, possivelmente, existiria uma vantagem na rapidez de implementação, mas não teríamos aprofundado tanto os nossos conhecimentos relativos a algoritmos de procura.

Em suma, pensamos ter atingido os objetivos propostos nesta fase, alargando o nosso conhecimento nesta área da inteligência artificial, compreendendo, assim, melhor estes variados algoritmos.