Uma imagem com Gráficos, captura de ecrã, símbolo, design

Descrição gerada automaticamente

**Universidade do Minho**

Escola de Engenharia

Licenciatura em Engenharia Informática

**Unidade Curricular de Inteligência Artificial**

Ano letivo de 2023/2024

**Relatório do projeto**

**Implementação de Algoritmos de Procura para a Otimização Autónoma de Circuitos de Entrega Sustentáveis**

**Grupo 10**

Diogo Gonçalves - 101919

Diogo Neto - 98197

Guilherme Oliveira - 95021

Tiago Pereira - 96429

**1 – Introdução**

Atualmente considerado o pai da inteligência artificial, IA, Alan Turing desenvolveu, em 1958, o “Teste de Turing”, sob a premissa “Conseguem as máquinas pensar? No manual “*Artificial Intelligence: A Modern Approach*” de Stuart Russell e Peter Norvig, são definidas quatro potenciais definições que diferenciam um computador:

* **Human Approach:**
* Sistemas que pensam como humanos
* Sistemas que agem como humanos
* **Ideal Aproach:**
* Sistemas que pensam racionalmente
* Sistemas que agem racionalmente

O conceito definido por *Turing* estaria inserido na categoria “Sistemas que agem como humanos”, no entanto esta ainda é uma área com várias definições distintas, mas, na sua forma mais redutora, IA é o campo que combina ciências da computação e grandes coleções de dados de modo a criar uma boa solução para um problema. Uma das formas de armazenar esta informação é através de grafos.

Um grafo é um tipo de dados abstrato que pode ser usado para representar um conjunto complexo de ligações entre objetos. Os elementos que compõem um grafo e são utilizados para armazenar informação designam-se nodos, estes, por sua vez, possuem ligações entre si, as arestas. No contexto do nosso projeto, este tipo de dados serve para representar uma rede de estradas em que cada nodo é uma cidade e as respetivas arestas as diferentes ligações rodoviárias, contudo, qualquer que seja o uso e área usada, os grafos são estruturas que podem modular de uma forma simples um problema inicialmente complexo.

Para que um grafo seja usado eficazmente, torna-se necessário que as formas de fazer a travessia entre os seus nodos sejam também elas o mais eficaz possível, de modo que um problema de pesquisa num grafo seja facilmente solucionado.

De modo a solucionar um problema de pesquisa em grafos, é essencial encontrar a solução ótima de acordo com os critérios propostos.

Formalmente, um problema de pesquisa define-se através da especificação dos seus componentes:

* + - - Estado inicial em que se começa;
    - - Ações que podem ser tomadas num estado aleatório;
    - - Verificação do estado obtido e do estado pretendido;
    - - Custo numérico do caminho entre dois estados;

O objetivo de qualquer um destes algoritmos é encontrar a sequência de ações que corresponde ao caminho do menor custo entre o estado inicial e o final. Em IA, estes algoritmos de travessia são divididos em duas formas:

* + **Algoritmos de pesquisa não informada** - Também conhecida como pesquisa cega, opera sem informações adicionais sobre o estado objetivo além daquelas definidas no problema. Estes algoritmos, como a Pesquisa em Largura ou a Pesquisa em Profundidade, exploram o espaço de procura sem qualquer direção específica para alcançar a meta, o que, muitas vezes, leva a uma eficiência menor em termos de tempo e recursos computacionais.
  + **Algoritmos de pesquisa informada** - Esta pesquisa utiliza heurísticas para guiar a procura, proporcionando uma abordagem mais direcionada e eficiente. Estas heurísticas são estimativas que ajudam a avaliar o quão próximo um determinado estado está objetivo. Algoritmos como A\* (A-Estrela) e Pesquisa Gulosa são exemplos de pesquisa informada, onde a procura é otimizada com base em informações adicionais, tornando-a geralmente mais rápida e eficiente a encontrar a solução desejada em comparação com a pesquisa não informada.

**2 - Caso de Estudo**

**2.1 - Descrição do problema**

Em contexto académico, foi proposta a realização de um programa que aplique vários algoritmos a um conjunto de dados da empresa de distribuição *Health Planet* com o objetivo de realizar as entregas da forma mais ecológica possível

O presente caso de estudo centra-se na otimização das rotas de entrega com o intuito de maximizar a sustentabilidade. Os estafetas dispõem de diferentes meios de transporte, cada um com um consumo energético distinto, para realizar entregas em várias ruas dentro de uma freguesia. Após uma entrega, cada cliente avalia o seu estafeta num *ranking* de 0 a 5 estrelas, e, se a entrega não for feita no prazo selecionado pelo cliente, o estafeta irá sofrer uma penalização neste ranking.

Tendo em conta as regras de transporte de encomendas, presentes enunciado, segue abaixo uma tabela que transcreve as mesmas. A tabela contém campos como o tipo de veículo utilizado pelo estafeta, a velocidade média de cada veículo, o limite de peso máximo que cada veículo pode transportar e o impacto que cada Quilograma (Kg) tem na velocidade do veículo, representada em quilómetros por hora (Km/h).

|  |  |  |  |
| --- | --- | --- | --- |
| Veículo | Velocidade Média | Limite de carga | Decréscimo de velocidade por cada Kg |
| Bicicleta | 10 Km/h | 5 Kg | 0.6 Km/h |
| Mota | 35 Km/h | 20 Kg | 0.5 Km/h |
| Carro | 50 Km/h | 100 Kg | 0.1 Km/h |

Tabela 1: Regras de transporte de encomendas nos veículos

**2.2 - Representação do Percurso de entrega**

A representação do circuito de entregas será efetuada através de um grafo, onde os nodos representam os pontos de entrega e as arestas as rotas entre eles. Este modelo permitirá visualizar as múltiplas possibilidades de percursos e ajudará na identificação da rota ótima

Modelo disponível posteriormente na figura 3.

**2.3 Formulação do Problema como um Problema de Pesquisa**

O problema será formulado como um problema de procura, onde o estado inicial é a localização inicial do estafeta, e o estado objetivo é a conclusão das entregas dentro do prazo estipulado. Os algoritmos incluem a seleção do meio de transporte e a escolha da rota, com o custo associado a cada solução sendo calculado com base no tempo, consumo energético e penalizações por atraso.

**3- Descrição da Solução Proposta para o Caso de Estudo**

**3.1 Percursos Criados**

Os circuitos propostos serão criados com base na análise dos grafos, procurando efetuar a entrega todas as encomendas de uma forma que minimize o consumo energético. Serão consideradas as variáveis de peso das encomendas e as limitações específicas de cada meio de transporte.

Os algoritmos usados e decisões tomadas serão discutidos posteriormente.

**3.2 Construtores e Criação de Estruturas**

A construção das estruturas necessárias para a implementação dos algoritmos será realizada através de classes e funções específicas, assegurando a eficácia e eficiência do sistema de procura. Serão criados construtores para inicializar os estados, definir os operadores e calcular os custos.

Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente

Figura 1: Construtor da classe Node

Associada à classe Graph, a classe Node é a parte central de um grafo que vai representar pontos de entrega do percurso.

Este código foi baseado no que foi estudado nas aulas práticas da unidade curricular, a baixo segue um excerto incompleto do código da classe Graph.

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Figura 2: Construtor da classe Graph

3.3 **Algoritmos Implementados**

3.3.1 **Pesquisa Não Informada**

Serão implementados algoritmos de pesquisa não informada, como a Pesquisa em Largura e Pesquisa em Profundidade, que exploram o espaço de procura sem o uso de informações adicionais sobre a distância ao objetivo.

* **DFS (Depth-First Search) -** Pesquisa em Profundidade:

**Conceito**: O DFS é um algoritmo de pesquisa que começa num nodo inicial e explora tão profundamente quanto possível ao longo de cada ramo antes de retroceder.

**Funcionamento**:

Opera sob o princípio de “mergulhar” verticalmente num grafo, percorrendo um caminho até que não seja mais possível avançar. Em cada nodo, o algoritmo escolhe um caminho não visitado e segue por ele, descendo sequencialmente até atingir o nodo final. Este processo é intrinsecamente recursivo, pois cada nova profundidade alcançada requer que o algoritmo continue a avançar até que todos os caminhos possíveis tenham sido verificados ou até que a meta seja alcançada. Uma característica marcante deste método é a sua eficiência em situações onde a solução pode ser encontrada sem a necessidade de percorrer toda a estrutura.

Por outro lado, o DFS não garante que o primeiro caminho encontrado seja o mais curto, apenas assegura que um caminho será encontrado, caso exista. Isto significa que, enquanto o algoritmo pode ser extremamente rápido ao encontrar uma solução, especialmente em estruturas onde os nodos estão localizados no final do grafo, este não se preocupa com a otimização do caminho em termos de distância ou custo. Portanto, o DFS é ideal para situações onde é necessário verificar a existência de uma conexão ou caminho entre dois pontos, sem que haja a necessidade de que este seja o caminho ótimo.

Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente

Figura 3 – Algoritmo DFS

* **BFS (Breadth-First Search)** - Pesquisa em Largura:

**Conceito**: O BFS é um algoritmo que começa num nodo inicial e explora todos os vizinhos deste nodo, de seguida, para cada um desses vizinhos mais próximos, explora os seus vizinhos não visitados e assim sucessivamente, até que todos os nodos sejam visitados.

**Funcionamento**:

Este algoritmo adota uma abordagem meticulosa e sistemática, explorando um grafo ou árvore nível por nível, a partir do nodo raiz, permitindo-lhe examinar todos os vizinhos de um determinado nodo antes de se deslocar para os nodos do próximo nível do grafo.

Ao manter uma fila para controlar a ordem de exploração dos nodos, o BFS assegura que toda a estrutura de dados seja percorrida de maneira uniforme. Esta estratégia proporciona uma visão completa de todos os caminhos possíveis de menor profundidade antes de progredir para os mais profundos, tornando o BFS particularmente eficaz para localizar o caminho mais curto ou de menor custo entre dois pontos num grafo.

A capacidade do BFS para encontrar o caminho mais eficiente é uma vantagem significativa, no entanto, essa eficiência tem um custo: o BFS pode exigir mais tempo e recursos de memória, pois mantém um registo extenso dos nodos já visitados e dos que ainda necessitam de ser explorados. Esta característica torna-o menos ágil que o DFS em situações onde o espaço de memória é uma preocupação ou quando a estrutura de dados é extremamente vasta e apenas a existência de um caminho é requerida, sem a necessidade de que seja o mais curto. Apesar dessas considerações, o BFS continua a ser uma ferramenta poderosa e insubstituível para problemas que demandam uma solução otimizada.

Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente

Figura 4 – algoritmo BFS

É de considerar que o nosso algoritmo para além de encontrar o caminho até ao destino, faz o caminho de volta à empresa.

3.3.2 **Pesquisa Informada**

A pesquisa informada, também conhecida como pesquisa heurística, difere da pesquisa não informada por utilizar conhecimento adicional sobre o problema para encontrar soluções mais eficientemente, esta abordagem é exemplificada por algoritmos como o A\* (A-Estrela) e a Pesquisa Gulosa, que incorporam heurísticas - estimativas inteligentes que avaliam quão 'perto' um estado está do objetivo final.

* **A\* (A-Estrela):**

**Conceito**: O A\* é um algoritmo refinado que combina aspetos de pesquisa em largura com heurísticas para direcionar a sua busca, este não só leva em conta o custo já acumulado para alcançar um determinado nodo (como o BFS), mas também uma estimativa heurística do custo para atingir o objetivo a partir desse nodo.

**Funcionamento**: O algoritmo utiliza uma função de avaliação *f(n) = g(n) + h(n)*, onde *g(n)* é o custo do caminho do nodo inicial até o nodo *n*, e *h(n)* é a estimativa heurística do custo mais barato de *n* até o objetivo final. O A\* seleciona o caminho que minimiza esta função, balanceando assim a exploração de caminhos e a expansão de nodos próximos. Esta metodologia permite ao A\* ser extremamente eficaz em encontrar o caminho mais curto de forma otimizada, evitando a exploração desnecessária de caminhos menos promissores.

Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente

Figura 5 – algoritmo A\*

* **Pesquisa Gulosa:**

**Conceito**: A Pesquisa Gulosa, ou algoritmo greedy, é semelhante ao A\* na utilização de heurísticas, mas foca exclusivamente na parte h(n) da função de avaliação, ignorando o custo g(n) já percorrido. Este algoritmo sempre escolhe o caminho que parece mais próximo do objetivo, baseando-se apenas na estimativa heurística.

**Funcionamento**: Ao avaliar os sucessores de um nodo, a Pesquisa Gulosa seleciona o nodo que tem a menor estimativa heurística para o objetivo, independentemente do custo acumulado até aquele ponto. Esta abordagem pode levar a soluções mais rápidas em certos casos, mas não garante a solução ótima, pois pode ser facilmente levada por caminhos que parecem promissores inicialmente, mas que acabam por ser menos eficientes no longo prazo.

Ambos os algoritmos são mais adequados para problemas onde é possível estimar a distância até o objetivo, estes são amplamente usados em problemas de caminho mais curto, como em sistemas de navegação GPS, algo semelhante ao caso em estudo. A escolha entre A\* e Pesquisa Gulosa dependerá do contexto específico e da natureza do problema: enquanto o A\* é mais geral e potente, garantindo encontrar o caminho mais curto se a heurística for admissível, a Pesquisa Gulosa pode ser mais rápida em casos onde a primeira solução aceitável é suficiente e o custo dos caminhos não é uma preocupação primordial.

Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente

Figura 6 – Algoritmo Gulosa

* **DLS - Depth-Limited Search - Pesquisa em Profundidade Limitada:**

**Conceito**: O DLS é um algoritmo que impõe um limite de profundidade na exploração de um grafo, o que significa que a pesquisa vai apenas até um nível de nodo especificado, este limite é uma forma de prevenir o problema do DFS padrão que pode entrar em loops infinitos em grafos com ciclos ou continuar indefinidamente em árvores de profundidade infinita. Ao mesmo tempo, permite uma busca que possa ser controlada para não gastar recursos excessivos em áreas pouco promissoras do espaço de busca.

**Funcionamento**: Assim como no DFS, o DLS explora um caminho até o final antes de retroceder, mas com a restrição de que ele não vai além de uma profundidade predefinida. Se essa profundidade limite for atingida sem encontrar o objetivo, o algoritmo retrocede e continua a pesquisa em outras direções. O DLS é especialmente útil em situações onde há um bom entendimento de até que ponto do espaço de procura e pode ser valioso explorar, ajudando a economizar tempo e recursos.

A implementação do DLS exige uma função adicional que verifique constantemente o nível de profundidade atual em relação ao limite estabelecido e se a pesquisa alcançar esse limite sem sucesso, esta não percorrerá esse caminho e retornará à bifurcação anterior para explorar outras opções. Esta técnica é uma abordagem eficaz para controlar o custo da busca em termos de tempo e memória, sendo particularmente útil em problemas onde a profundidade pode crescer rapidamente, no entanto, a escolha do limite de profundidade é crítica: se for muito baixo, soluções possíveis podem não ser descobertas; se for muito alto, o algoritmo pode degenerar para um DFS padrão com suas desvantagens associadas.

**Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente**

Figura 7– Algoritmo Gulosa

**4 - Menu interativo**

De uma forma simplista, o menu interativo é a interface de utilizador (*UI*), que permite ao mesmo a escolha e visualização das diferentes funcionalidades implementadas.

Uma imagem com texto, software, número, Página web

Descrição gerada automaticamente

Figura 8: Menu de Utilizador

Como é possível observar na figura 1, o utilizador tem acesso a várias funcionalidades de consulta de informações dos diferentes elementos que compõem a estrutura de dados da *Health Planet,* assim como a possibilidade de gerar e representar circuitos/grafos.

* **Funcionalidade 13** – Gerar circuitos de entrega para um determinado território

Uma imagem com texto, captura de ecrã

Descrição gerada automaticamente

Figura 9: Circuito/Grafo representado em texto

Esta opção presente no nosso programa cria e mostra vários percursos possíveis para um destino indicado pelo utilizador, é de salientar que na figura acima não estão todas as possibilidades geradas pelo algoritmo.

* **Funcionalidade 14** – Representar pontos de entrega através de um grafo

Uma imagem com captura de ecrã, file, mapa, texto

Descrição gerada automaticamente

Figura 10: Grafo do circuito

Esta opção do programa permite ao utilizador visualizar as freguesias onde existem pontos de entrega, representadas por cada nodo do grafo, assim como a distância entre estes pontos, representada pelas arestas do grafo.

* **Funcionalidade 15** – Comparar circuitos de entrega

Uma imagem com texto, captura de ecrã, software, Página web

Descrição gerada automaticamente

Figura 11: Algoritmos aplicados a circuitos

Esta funcionalidade leva em conta uma encomenda específica e aplica diferentes algoritmos ao dataset gerando assim várias possibilidades e permite ao utilizador comparar a distância e tempo entre estes algoritmos.

Mostra a melhor possibilidade de cada um dos 5 algoritmos

* **Funcionalidade 16** - Mostrar circuito de entrega

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Figura 12: Nodos de um grafo e distância percorrida representados em texto

Na sequência da funcionalidade anteriormente descrita, esta permite ao utilizador a visualização e comparação dos percursos, em texto, assim como a distância total percorrida.

**5 – Discussão das decisões tomadas e comparação dos resultados obtidos**

De acordo com o enunciado, segue abaixo o código que traduz a tabela 1.

**Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente**

Figura 13: Decisões tomadas para o tipo de veiculo

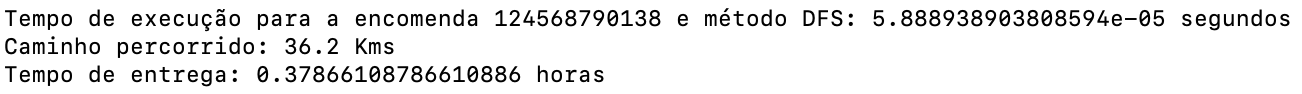
Para a tomada de decisão do custo por prazo, tomamos a liberdade de utilizar os seguintes valores:

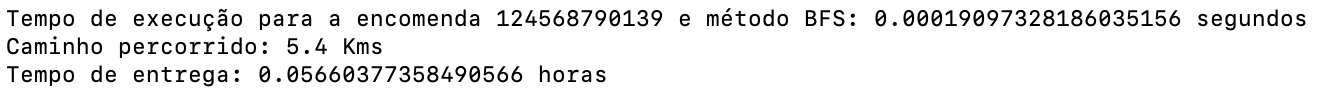
**Uma imagem com texto, captura de ecrã, Tipo de letra

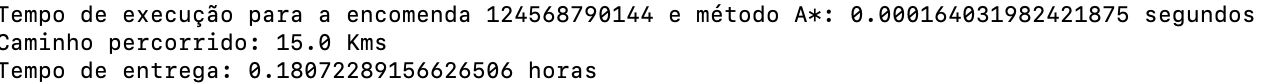
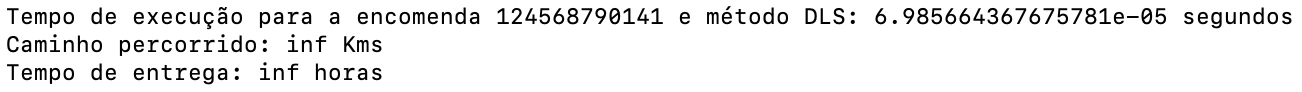
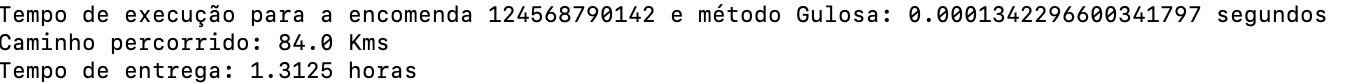
Descrição gerada automaticamente**

Figura 14: Decisões tomadas para o preço

Comparação de resultados

****

****

****

**6 – Conclusão**

Os algoritmos de pesquisa não informada e informada foram explorados e implementados, demonstrando as suas capacidades e limitações. Enquanto a pesquisa não informada ofereceu um ponto de partida sólido para a exploração do espaço de procura sem direcionamento heurístico, a pesquisa informada, com o uso de algoritmos como A\* e Pesquisa Gulosa, trouxe eficiência e otimização ao processo, proporcionando soluções mais próximas do ideal em termos de custo e tempo. Além disso, a inovação da pesquisa em profundidade limitada mostrou ser uma ferramenta valiosa para contornar as limitações de recursos em grafos extensos.

Este projeto não só confirmou a relevância da inteligência artificial e da pesquisa em grafos para o desenvolvimento de soluções de entrega sustentáveis mas também reforçou o potencial das abordagens heurísticas na otimização de problemas complexos. A capacidade de adaptar-se e escolher entre diferentes métodos de procura, levando em consideração as características específicas do problema, mostrou ser um fator determinante para o sucesso na busca de soluções otimizadas.

Finalmente, através da experimentação e análise dos resultados obtidos, concluímos que, apesar dos algoritmos de pesquisa não informada terem o seu valor, são os algoritmos de pesquisa informada, como o A\*, que se destacam ao fornecer soluções mais rápidas e eficazes, alinhadas com as demandas por sustentabilidade e eficiência. Este projeto é um testemunho da capacidade da IA em contribuir significativamente para os avanços na otimização de processos, sustentabilidade e inovação tecnológica.

**7 – Bibliografia**

Norvig, P., Russell, S., Pearson; Artificial Intelligence: A Modern Approach, 1995, 3rd edition (December 1, 2009)

Costa E., Simões A., (2008), Inteligência Artificial-Fundamentos e Aplicações

https://pt.wikipedia.org/wiki/Alan\_Turing