

Universidade do Minho

Mestrado em Inteligência Artificial

Relatório do Projeto
Computação Inspirada na Natureza
2025/2026

Grupo 1

Francisco Costa PG60387

João Mendes PG60388

Simão Silva PG60393

Vasco Macedo PG60394



Índice

Introdução.....	1
1. Dados e Pré-Processamento.....	2
1.1 Dados GTFS	2
1.2 Emissões de CO ₂	3
1.3 Travessias das Pontes	3
2. Construção do Grafo Multimodal	3
2.1 Tipos de nós.....	4
2.2 Tipos de arestas	4
3. Formulação do Problema	4
3.1 Funções Objetivo	4
3.1 Restrições	5
4. Algoritmo de Otimização	5
4.1 Representação da solução	5
4.2 População inicial	5
4.3 Algoritmo Evolutivo	6
4.4 Operadores genéticos	6
4.4.1 Crossover (Cruzamentos).....	6
4.4.2 Mutação	6
5. Arquitetura da Aplicação.....	7
5.1 Organização modular	7
5.1 Fluxo de Execução	7
5.3 Interface Web	8
6. Resultados Obtidos	9
7. Discussão.....	12
7.1 O que o sistema faz bem.....	12
7.2 Limitações	12
7.3 Impacto das escolhas algorítmicas	12
8. Conclusão	13
9. Referências	13



Introdução

No âmbito da cadeira de Computação Inspirada na Natureza, do 1º ano do Mestrado em Inteligência Artificial, foi-nos proposta a elaboração de um projeto com o propósito de ajudar utilizadores a escolher a melhor forma de se deslocarem no Grande Porto.

Atualmente, os cidadãos ponderam múltiplos fatores nas suas deslocações, como a pegada ecológica, o custo financeiro e o benefício físico da caminhada. Este cenário transforma o planeamento de rotas num problema de otimização multiobjectivo, onde frequentemente não existe uma solução única ideal, mas sim um conjunto de compromissos (*trade-offs*) entre objetivos conflituosos.

O presente projeto visa desenvolver um planeador de rotas multimodal que integra a rede do Metro do Porto, a rede de autocarros da STCP (Sociedade de Transportes Coletivos do Porto) e percursos pedonais. Ao contrário das abordagens clássicas de caminho mais curto, este sistema utiliza algoritmos de computação evolutiva para explorar o espaço de pesquisa e devolver um conjunto de soluções não dominadas (Fronteira de Pareto). O sistema permite ao utilizador personalizar as suas preferências — valorizando, por exemplo, a sustentabilidade ambiental (minimização de CO₂) ou a prática de exercício físico — e respeita restrições operacionais, como o limite máximo de transbordos e o tempo máximo de caminhada permitido.

Para concretizar este objetivo, foram utilizados dados reais no formato GTFS (*General Transit Feed Specification*), permitindo uma modelação fidedigna das redes de transporte, horários e frequências. A solução final inclui uma interface web interativa que facilita a visualização e comparação das diferentes rotas sugeridas pelo algoritmo.



1. Dados e Pré-Processamento

O desenvolvimento de um planejador multimodal que simule com precisão as redes de transporte público do Grande Porto exige a utilização de dados normalizados e detalhados. A base informacional deste projeto reside nos conjuntos de dados GTFS, fornecidos pelo Metro do Porto e pela STCP, complementados por informações específicas sobre emissões de dióxido de carbono e regras de caminhada. O pré-processamento é a etapa crucial que transforma esta coleção de ficheiros tabulares estáticos num modelo temporal e espacial dinâmico, apto para a construção do Grafo Multimodal e para a avaliação das funções objetivo.

1.1 Dados GTFS

Definição dos ficheiros utilizados, do tipo txt, provenientes da STCP e sua utilização no pré-processamento:

- *agency* – Agências de transporte público com serviços representados neste conjunto de dados.
- *calendar_dates* – Exceções para os serviços definidos no ficheiro *calendar.txt*.
- *calendar* – As datas de serviço são especificadas usando um cronograma semanal com datas de início e término. Essencial para garantir que o planeamento é feito para dias operacionais válidos.
- *routes* – Percursos de transporte público. Uma rota é um conjunto de viagens que são apresentadas aos passageiros como um único serviço. Usado para associar viagens aos seus modos de transporte (Metro ou STCP) e determinar o fator de emissão de CO₂.
- *shapes* – Regras para mapear as trajetórias dos veículos. Também chamadas de alinhamentos de percursos. Permite calcular a distância real percorrida, necessária para determinar as emissões CO₂ e o custo.
- *stop_times* – Horários a que um veículo chega e parte dos pontos de paragem em cada viagem. Fundamental para calcular o tempo de espera e o tempo de viagem.
- *stops* – Paragens onde os veículos embarcam ou desembarcam passageiros. Também define estações e entradas de estações. Define os nós (pontos de interesse) do grafo multimodal.
- *transfers* – Regras para efetuar ligações em pontos de transferência entre rotas. Define o custo temporal das arestas de transferência no grafo.
- *trips* – Deslocamentos para cada percurso. Um deslocamento é uma sequência de duas ou mais paragens que ocorrem durante um período de tempo específico. Liga as linhas (*routes.txt*) aos horários (*stop_times.txt*).

A lista dos ficheiros utilizados, provenientes do metro, são do mesmo tipo do que os provenientes da STCP, com a adição de mais dois:

- *fare_attributes* – Informação sobre tarifas para as linhas de transporte público.



- *fare_rules* – Regras para a aplicação de tarifas em itinerários.

1.2 Emissões de CO₂

O objetivo é minimizar as emissões, que dependem diretamente da distância percorrida pelo veículo e do fator de emissão associado a cada modo de transporte.

A caminhada é considerada neutra em carbono neste modelo, com uma emissão de 0 gramas por quilómetro. É essencial para ligar os nós de paragens a nós artificiais e garantir que os tempos de acesso ao transporte público são realistas. Os fatores de emissão utilizados são baseados em estudos de sustentabilidade do transporte e a velocidade pedonal é a velocidade média a que uma pessoa se desloca enquanto caminha. Estes valores estão definidos no ficheiro *src/constants.py*, o que permite uma gestão centralizada e fácil ajuste dos parâmetros de emissão para toda a aplicação.

- Metro: 40 gramas CO₂ / passageiro·km
- STCP (autocarro): 109.9 gramas CO₂ / passageiro·km
- Velocidade pedonal constante: 1.4 m/s ≈ 5 km/h

As emissões de CO₂ para um percurso completo são calculadas somando as emissões de cada segmento de transporte motorizado. A distância é estimada por *haversine* (fórmula usada para determinar a distância entre dois pontos que pertençam a uma esfera dadas as suas longitudes e latitudes) entre coordenadas. A forma de cálculo é dada por:

$$E_{total} = \sum_{\{i \in \text{segmentos}\}} D_i \times F_{emissão, modo(i)}$$

- E_{total} é a emissão total em gramas de CO₂.
- D_i é a distância do segmento i em quilómetros.
- $F_{emissão, modo(i)}$ é o fator de emissão do modo de transporte utilizado no segmento i (gCO₂/p.km).

1.3 Travessias das Pontes

A modelação precisa da componente pedonal entre o Porto e Vila Nova de Gaia. Isto exigiu uma atenção especial às regras de acessibilidade das pontes sobre o Rio Douro.

O sistema aplica uma política de restrição no módulo de construção do grafo (*graph_builder.py*) para as travessias entre as duas margens do rio. Estas regras estão centralizadas em ficheiros de configuração (*bridges_pedestrian_rules.txt* e *bridges_geometry.json*) e definem dois tipos de estruturas. As pontes em que é permitido andar a pé, sendo estas a ponte Luís I e a ponte Infante Dom Henrique, e as pontes onde é proibido transitar a pé, nomeadamente a ponte da Arrábida, do Freixo, de São João e a Maria Pia.

2. Construção do Grafo Multimodal

A Construção do Grafo Multimodal é a etapa central do projeto, onde os dados tabulares e temporais (GTFS) são transformados numa estrutura de grafo dinâmico. O



grafo, modelado através da biblioteca NetworkX, representa o espaço de pesquisa para o algoritmo evolutivo. O processo, gerido no módulo `graph_builder.py`, resulta num grafo $G = (V, E)$, onde V é o conjunto de nós (vértices) e E é o conjunto de arestas (ligações).

2.1 Tipos de nós

Os nós do grafo (V) não representam apenas as paragens físicas, mas também pontos estratégicos para permitir a interligação entre os diferentes modos de transporte:

- paragens STCP: nós associados às paragens de autocarro.
- estações do Metro: nós associados às estações de metro.
- nós artificiais para caminhada: pontos de interesse criados para ligar a origem e o destino do utilizador à rede principal, permitindo que a caminhada ocorra a partir de qualquer local.

2.2 Tipos de arestas

As arestas do grafo (E) definem as ligações e o custo (tempo, CO_2) associado à transição entre os nós:

- Arestas de transporte (Metro/STCP): ligam duas paragens consecutivas numa viagem específica, com base nos horários definidos em `stop_times.txt`. Estas arestas têm um custo temporal fixo e são a fonte primária das emissões de CO_2 .
- Arestas de caminhada: Ligam dois nós quaisquer que estejam dentro de um raio de caminhada aceitável. As *walk edges* entre paragens são criadas por proximidade (*walk_radius*) e *Wmas_s* é a restrição de tempo total a pé do percurso e também limita apenas a *walk edge* direta de origem para o destino se for adicionada. O custo é o tempo de caminhada, e as emissões são nulas. A sua criação deve respeitar as regras de travessia das pontes.
- Arestas de transferências: Ligam nós ou paragens adjacentes, representando a troca entre diferentes linhas ou modos de transporte. O custo é um pequeno tempo de espera calculado por *headway*.

3. Formulação do Problema

O problema do planeamento de rotas multimodal é formulado como um Problema de Otimização Multi-Objetivo (MOP). O objetivo não é encontrar uma única solução ótima global, mas sim identificar um conjunto de soluções de compromisso, conhecido como Fronteira de Pareto. O problema pode ser definido formalmente como a minimização (ou maximização) simultânea de um vetor de funções objetivo $F(P)$:

$$\min F(P) = [f_{\text{tempo}}(P), f_{\text{CO}_2}(P), f_{\text{caminhada}}(P)]$$

3.1 Funções Objetivo

O sistema considera três objetivos principais, que frequentemente entram em conflito entre si (a rota mais rápida pode não ser a mais ecológica):

1. Minimizar tempo total de viagem.



2. Minimizar emissões de CO₂ (dióxido de carbono).
3. Maximizar ou minimizar caminhada (exercício).

3.1 Restrições

Para garantir que as soluções geradas são viáveis e aceitáveis para o utilizador, o espaço de pesquisa é limitado por restrições rígidas ("hard constraints"). Se uma rota violar qualquer uma destas restrições, é aplicada uma penalização severa (PENALTY) à sua aptidão (*fitness*), excluindo-a efetivamente da seleção.

- Limite máximo de caminhada (w_{max_s}) – impede que o sistema sugira rotas com uma quantidade excessiva de caminhada, garantindo o conforto do utilizador.
- Máximo de transbordos (t_{max}) – limita a complexidade da viagem, restringindo o número de trocas de veículo.
- Proibição de atravessar certas pontes – garante que não são tidas em conta pontes onde não é possível certos meios de locomoção.

4. Algoritmo de Otimização

Para resolver o problema de otimização multi-objetivo formulado, foi implementado um Algoritmo Evolutivo (EA), especificamente o NSGA-II (*Non-dominated Sorting Genetic Algorithm II*), utilizando a biblioteca Python *deap*. Ao contrário dos métodos exatos, o NSGA-II trabalha com uma população de soluções, evoluindo-as ao longo de gerações para aproximar a Fronteira de Pareto ótima.

4.1 Representação da solução

Dado que o problema consiste em encontrar rotas num grafo direcionado $G = (V, E)$, adotou-se uma representação baseada em caminhos (*Path Representation*) com tamanho variável. Neste modelo, cada indivíduo (solução candidata) não é representado por um vetor de características fixas, mas sim por uma sequência ordenada de identificadores de nós que descreve o trajeto completo desde a origem até ao destino. Formalmente, um indivíduo (rota) P é definido como um vetor de nós:

$$P = [v_1, v_2, \dots, v_k]$$

- v_1 corresponde ao nó de origem;
- v_k corresponde ao nó de destino;
- k é o comprimento do caminho, número de nós, que é variável entre indivíduos, uma vez que diferentes rotas podem ter números distintos de paragens ou transferências;
- Cada v_i representa um ID único no grafo multimodal.

4.2 População inicial

A geração de rotas puramente aleatórias num grafo de grande dimensão resulta frequentemente em caminhos inválidos ou excessivamente longos. Para contornar o problema da convergência prematura e acelerar a descoberta de boas soluções, adotou-se uma estratégia de Inicialização Inteligente (*Seeding*), conforme sugerido no enunciado do projeto.



A população inicial não é totalmente aleatória, mas sim "semeada" com soluções heurísticas calculadas por algoritmos de caminho mais curto (Dijkstra ou A*) focados em objetivos individuais.

Inicialização por conjunto de caminhos extremos e intermédios via Dijkstra- λ (*weighted sum* normalizado de tempo e emissões), para cobrir melhor a frente de Pareto logo no arranque. O resto da população é completado com caminhos por *random walk* e/ou *shortest path* com ruído.

Esta abordagem garante que o algoritmo começa a evolução já com pontos extremos da Fronteira de Pareto conhecidos, focando o esforço computacional em encontrar os *trade-offs* intermédios.

4.3 Algoritmo Evolutivo

O ciclo evolutivo segue a estrutura padrão do NSGA-II, selecionando os sobreviventes com base em dois critérios hierárquicos:

1. Non-Dominated Sorting (Ordenação Não-Dominada): A população é dividida em "fronteiras" (F_1, F_2, \dots). A fronteira F_1 contém as soluções que não são dominadas por nenhuma outra. F_2 contém as soluções dominadas apenas por membros de F_1 , e assim sucessivamente. O objetivo é maximizar a pertença às primeiras fronteiras.
2. Crowding Distance (Distância de Aglomeração): Dentro da mesma fronteira, o algoritmo prefere soluções que estejam mais "isoladas" no espaço dos objetivos. Isto força o algoritmo a manter a diversidade, evitando que todas as soluções convirjam para um único ponto (ex: apenas soluções rápidas) e ignorando outras regiões do Pareto (ex: soluções ecológicas).

4.4 Operadores genéticos

A aplicação de operadores genéticos em caminhos de grafos exige cuidados especiais para manter a validade das rotas.

4.4.1 Crossover (Cruzamentos)

O operador de cruzamento combina informações de dois progenitores para criar descendentes. No contexto de rotas, utilizou-se uma abordagem baseada em interseções geográficas:

- Identifica-se um nó comum (interseção) que exista em ambos os caminhos dos progenitores.
- Trocam-se os segmentos de rota a partir desse ponto.
- Se não houver nós comuns (além da origem/destino), o cruzamento não é possível e os pais mantêm-se.

4.4.2 Mutação

A mutação introduz variabilidade e previne a estagnação em mínimos locais.

- Seleciona-se aleatoriamente um subsegmento da rota original (entre dois nós intermédios u e v).



- Este segmento é removido e substituído por um novo caminho entre u e v .
- Este novo subcaminho pode ser gerado aleatoriamente ou através de uma heurística local, permitindo explorar variantes de trajeto numa zona específica da cidade

5. Arquitetura da Aplicação

O sistema foi desenvolvido em Python (v3.10+), seguindo uma arquitetura modular que separa claramente a ingestão de dados, a lógica de otimização e a interface de apresentação. Esta estrutura facilita a manutenção, a testabilidade de componentes individuais e a futura expansão para novos modos de transporte.

5.1 Organização modular

- Importação GTFS (*loader.py*): este módulo carrega os ficheiros de texto GTFS (Metro e STCP) para estruturas de dados em memória (DataFrames do *pandas*). É aqui que são aplicados filtros de calendário para validar se os serviços operam no dia selecionado.
- Grafo Multimodal (*graph_builder.py*): converte os dados carregados num grafo direcionado utilizando a biblioteca NetworkX. Este módulo é responsável por criar os nós de transferência e aplicar as regras de restrição das pontes.
- Motor Evolutivo (*evolution.py*): implementa o ciclo de vida do algoritmo NSGA-II utilizando a biblioteca DEAP. Gere a população, executa o *loop* geracional e aplica os operadores genéticos.
- Avaliação (*fitness.py*): contém as funções de avaliação que calculam o custo de cada rota (Tempo, CO₂, Caminhada) e verificam as restrições rígidas (W_{\max} , T_{\max}).
- Heurísticas (*baselines.py*): fornece algoritmos de caminho mais curto (Dijkstra) para gerar a população inicial inteligente.
- Camada de Apresentação (*app/streamlit_app.py*): a interação com o utilizador é assegurada por uma aplicação web desenvolvida em Streamlit. esta *framework* permite a renderização rápida de componentes de dados e interatividade sem a necessidade de uma separação complexa entre *frontend* (JavaScript) e *backend*.

5.1 Fluxo de Execução

1. Input do Utilizador: Através da interface web, o utilizador define a origem, destino e preferências.
2. Validação e Geo-codificação: O sistema converte os nomes das paragens em nós do grafo válidos.
3. Inicialização: O grafo é carregado (ou recuperado da *cache*) e a população inicial é gerada com base em heurísticas.
4. Ciclo Evolutivo: O algoritmo NSGA-II executa durante N gerações, refinando as rotas para aproximar a Fronteira de Pareto.
5. Pós-Processamento: As soluções não-dominadas finais são decodificadas em instruções legíveis (segmentos de rota).
6. Visualização: Os resultados são apresentados ao utilizador sob a forma de tabela comparativa e descrição passo-a-passo.



5.3 Interface Web

A interface foi desenhada para ser intuitiva e permitir a exploração dos *trade-offs* multiobjetivo. As principais funcionalidades implementadas incluem:

- O utilizador pode ajustar em tempo real a importância relativa de cada objetivo (Tempo vs. CO₂ vs. Exercício) através de *sliders*, o que reordena imediatamente a apresentação das rotas sugeridas
- Apresentação clara dos KPIs de cada rota (emissões em gramas, distância a pé em metros), permitindo uma decisão informada.
- Uma descrição textual detalhada de cada segmento da viagem, incluindo tempos de espera e locais de transbordo.



Planeador multimodal CIN

Explora combinações tempo/CO₂/caminhada geradas pelo algoritmo multiobjetivo e recebe instruções passo a passo.

Preferência principal: ☒ Tempo ☐ CO₂ ☐ Exercício ☐ Equilíbrio

Peso tempo: 0.70 | Peso CO₂: 0.20 | Peso exercício: 0.10

Origem: Trindade | Destino: Gaia

Opções avançadas

- Tamanho da população: 60
- N.º de gerações: 35
- Limite total a pé (min): 30
- Transbordos máximos: 3
- Política de caminhada: minimize
- ☐ Incluir custo como objetivo extra

Gerar rotas

Figura 1 - Definição de rota e suas variáveis

Pareto: compara opções

Rota	Tempo total (min)	CO ₂ (g)	Caminhada (m)	Esperas (min)	Transbordos
5	6	15.5	70	830	0.6
6	7	19.2	53	1261	1.1
7	8	17.5	67	915	0.6
8	9	16.4	67	912	0.6
9	10	16.7	68	886	1.1
10	11	19.8	51	1317	1.1
11	12	20.2	50	1311	0.6
12	13	12.7	86	434	0.6
13	14	12.3	86	431	1.1
14	15	16.5	69	865	1.1

Seleciona rota

Rota #1 Rota #2 Rota #3 Rota #4
Rota #5 Rota #6 Rota #7 Rota #8
Rota #9 Rota #10 Rota #11 Rota #12
Rota #13 Rota #14 Rota #15

Rota selecionada #2

Tempo total	CO ₂	Caminhada	Transbordos
8.6 min	103 g	0.00 km	0

Passos da rota

- Esperar em Trindade (METRO) (0.6 min) pela próxima ligação
- metro - D: Trindade (METRO) → Aliados (METRO) (1.0 min)
- metro - D: Aliados (METRO) → São Bento (METRO) (2.0 min)
- metro - D: São Bento (METRO) → Jardim do Morro (METRO) (2.0 min)
- metro - D: Jardim do Morro (METRO) → General Torres (METRO) (2.0 min)
- metro - D: General Torres (METRO) → Câmara Gaia (METRO) (1.0 min)

Dados carregados de: outputs/pareto/pareto_solutions.json

Figura 2 - Resultados

6. Resultados Obtidos

A melhor maneira de averiguar o sucesso da solução criada é para o mesmo ponto de origem e de destino, verificar as rotas que são geradas consoante o peso do objetivo pretendido (tempo ou CO₂ ou exercício).

Ao manter inalterável as restantes opções avançadas, tais como, o tamanho da população (60), o número de gerações (35), o limite total a pé (30 min), o número de transbordos máximos (3) e a política de caminhada (maximize), correu-se o programa uma vez para cada objetivo pretendido, onde o valor do peso atribuído ao objetivo mais importante é de 0.90 e os restantes têm um valor de 0.20. A origem foi o Palácio de Cristal e o destino foi a Cordoaria.

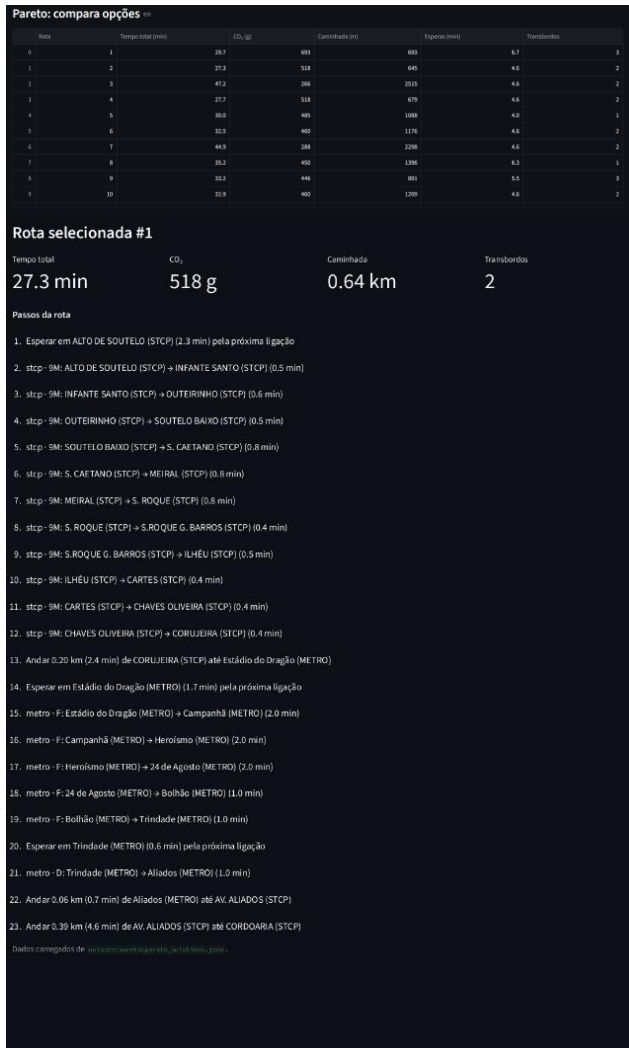


Figura 3 - Preferência no tempo total

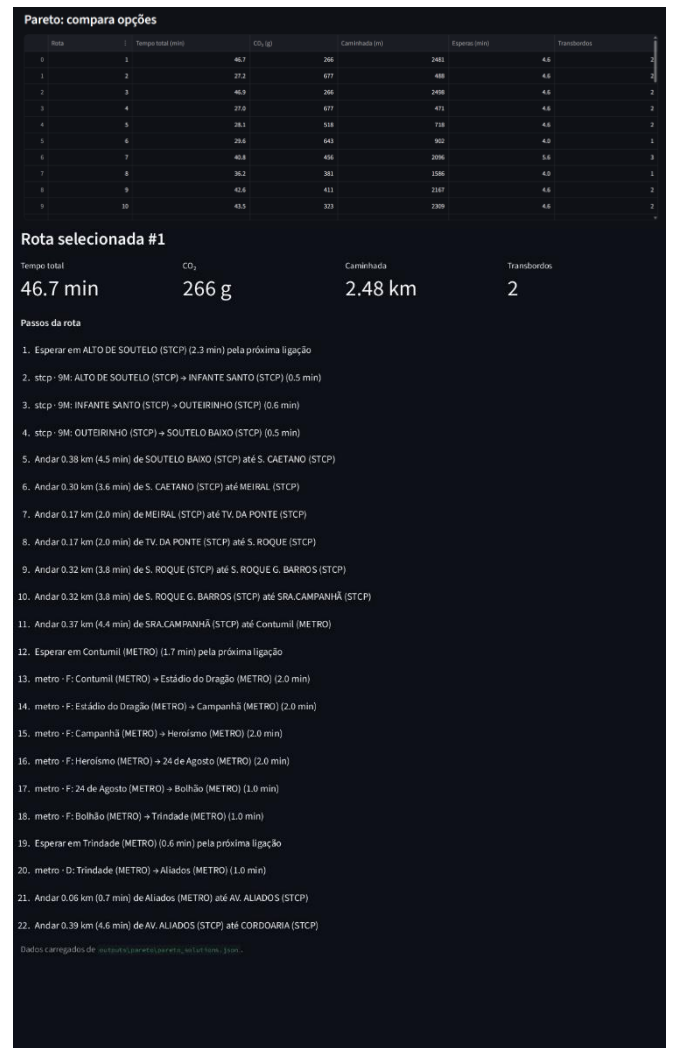


Figura 4 - Preferência nas emissões de CO₂

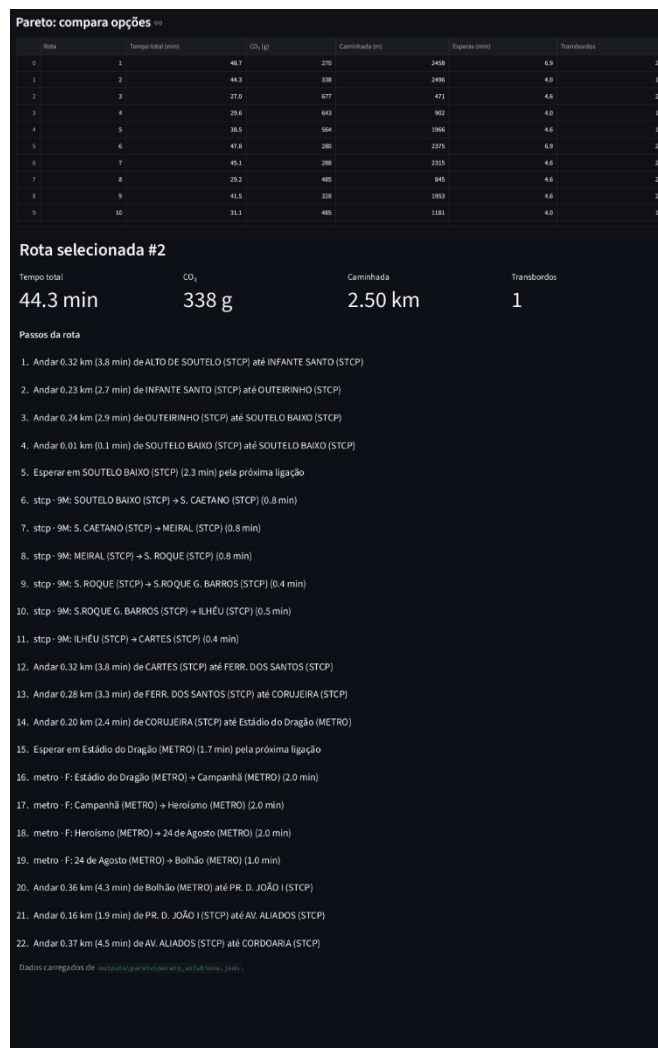


Figura 3 - Preferência no exercício

Como podemos constatar, as rotas geradas consoante a importância do fator definido estão de acordo com o objetivo pretendido. É de realçar que a margem entre os valores da distância da caminhada é pequena se compararmos os resultados entre os cenários onde o objetivo é o exercício e a menor emissão possível de dióxido de carbono. Isto acontece porque o meio de locomoção menos poluente é a caminhada e como resultado disso, quando se pretende minimizar as emissões de CO₂, a distância percorrida a pé terá um grande impacto na escolha da rota.

Para avaliar quantitativamente a qualidade das fronteiras Pareto obtidas, foi utilizada a métrica de hipervolume. O hipervolume mede simultaneamente a convergência e a diversidade das soluções não dominadas, calculando a área dominada pela fronteira em relação a um ponto de referência. Valores mais elevados de hipervolume indicam um melhor compromisso global entre os objetivos considerados. Esta métrica foi utilizada para comparar as soluções obtidas pelo NSGA-II com um baseline baseado em caminhos determinísticos. O hipervolume é calculado sobre a fronteira 2D (tempo total, emissões).



7. Discussão

A implementação de um planeador multimodal baseado em algoritmos evolutivos permitiu explorar soluções de mobilidade que escapam à lógica tradicional dos algoritmos de caminho mais curto uni objetivo. A análise dos resultados obtidos e do processo de desenvolvimento revela tanto as mais-valias da abordagem como as limitações inerentes à modelação.

7.1 O que o sistema faz bem

- diversidade de soluções: Uma das principais vantagens observadas no sistema é a capacidade de apresentar uma Fronteira de Pareto rica e diversificada. Ao contrário dos planeadores comerciais, que tendem a convergir apenas para a rota mais rápida, o nosso sistema identificou consistentemente alternativas onde um pequeno aumento no tempo de viagem resultava numa redução significativa das emissões de CO₂ (por exemplo, preferindo o Metro ao autocarro em trajetos longos) ou num aumento saudável da atividade física.
- Inicialização inteligente vs. Aleatória: A introdução de uma população inicial "semeada" com soluções heurísticas (Dijkstra) provou ser essencial. Nos testes preliminares com inicialização puramente aleatória, o algoritmo despendia as primeiras gerações a tentar encontrar caminhos válidos, resultando numa convergência lenta. A inicialização híbrida permitiu que o algoritmo evolutivo se focasse, desde o início, no refinamento e combinação de rotas já viáveis.

7.2 Limitações

- Natureza estática dos dados (GTFS): o modelo baseia-se em horários planeados e não em tempo real. Não considera atrasos operacionais, congestionamento de trânsito (crítico para a STCP) ou supressão de viagens.
- Regras pedonais simplificadas: Assumir uma velocidade de caminhada fixa de 1.4 m/s ignora a variabilidade humana (idade, condição física) e a topografia acidentada do Porto. O esforço de caminhar 500 metros numa subida íngreme é modelado de forma igual a 500 metros em plano, o que pode enviesar a atratividade das rotas a pé.
- Penalização de transferência: o custo de transferência foi modelado como um tempo fixo ou uma penalização no *fitness*. Embora funcional, este modelo não captura a complexidade real de mudar de cais numa estação grande como a Trindade versus uma paragem de autocarro simples.

7.3 Impacto das escolhas algorítmicas

O desempenho do planeador não depende apenas da formulação do problema, mas também da configuração intrínseca do algoritmo evolutivo.

A utilização de uma população inicial híbrida, contendo soluções extremas geradas por Dijkstra (menor tempo, menor CO₂), teve um impacto determinante na métrica de Hipervolume inicial.

Sem *seeding* o algoritmo inicia com um Hiper volume baixo e despende de várias gerações apenas para encontrar rotas topologicamente válidas, desperdiçando esforço computacional.



Com *seeding* o algoritmo inicia a evolução já na proximidade da Fronteira de Pareto real. Isto permitiu reduzir o número necessário de gerações, focando a pesquisa no preenchimento das lacunas (soluções de compromisso) entre os extremos pré-calculados.

Foram realizados testes de sensibilidade com diferentes tamanhos de população {20, 60, 100}. Observou-se que populações muito pequenas ($N=20$) tendem a perder diversidade genética, convergindo prematuramente para mínimos locais (e.g., focando-se apenas em rotas de Metro e ignorando combinações complexas com autocarro).

8. Conclusão

O presente projeto alcançou com sucesso o objetivo de desenvolver um planeador de rotas multimodal para a área do Grande Porto, aplicando técnicas de Computação Inspirada na Natureza para resolver um problema complexo de Otimização Multi-Objetivo.

Ao integrar dados reais GTFS do Metro do Porto e da STCP com um modelo de grafo pedonal detalhado, foi possível construir uma ferramenta que ultrapassa a lógica binária dos planeadores tradicionais. A utilização do algoritmo evolutivo NSGA-II, complementada por uma estratégia de inicialização inteligente baseada em heurísticas, provou ser uma abordagem eficaz para identificar a Fronteira de Pareto num espaço de pesquisa vasto.

Os resultados obtidos demonstram que não existe uma única "melhor rota" absoluta. O sistema revelou consistentemente soluções de compromisso valiosas, onde pequenos sacrifícios no tempo de viagem (e.g., +5 minutos) podem resultar em ganhos significativos na redução da pegada ecológica (e.g., -30% emissões CO_2) ou no aumento da atividade física. A interface desenvolvida permite ao utilizador final navegar nestes *trade-offs* de forma intuitiva, promovendo uma mobilidade mais consciente e sustentável.

9. Referências

- [1]
«Análise de Sustentabilidade». Acedido: 23 de dezembro de 2025. [Online]. Disponível em: <https://www.metrodoporto.pt/pages/358>
- [2]
«Conjunto de Dados - Portal de Dados». Acedido: 7 de dezembro de 2025. [Online]. Disponível em: https://opendata.porto.digital/dataset/?q=Infraestruturas+e+Mobilidade&res_format=GTFS
- [3]
«Non-Dominated Sorting Genetic Algorithm 2 (NSGA-II)», GeeksforGeeks. Acedido: 23 de dezembro de 2025. [Online]. Disponível em: <https://www.geeksforgeeks.org/deep-learning/non-dominated-sorting-genetic-algorithm-2-nsga-ii/>
- [4]
«Reference - General Transit Feed Specification». Acedido: 7 de dezembro de 2025. [Online]. Disponível em: <https://gtfs.org/documentation/schedule/reference/#agencytxt>



[5]

«STCP - Mobilidade Sustentável». Acedido: 23 de dezembro de 2025. [Online].

Disponível em: <https://stcp.pt/pt/sustentabilidade>