FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

FEUP

iRouting: Utilização de técnicas inteligentes de otimização em sistemas de navegação GPS

Diogo José dos Santos Rocha

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Rosaldo José Fernandes Rossetti (Doutor)

Co-Orientador: António Fernando Vasconcelos Cunha Castro Coelho (Doutor)

Julho de 2012

iRouting: Utilização de técnicas inteligentes de otimização em sistemas de navegação GPS

Diogo José dos Santos Rocha

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Doutor Eugénio da Costa Oliveira

Arguente: Doutor Fernando Augusto Cruz e Silva Mouta

Vogal: Doutor Rosaldo José Fernandes Rossetti

Julho de 2012

Resumo

Este documento visa dar uma abordagem geral sobre o problema inicial e todos os detalhes que levaram à sua solução no âmbito da dissertação "*iRouting*: utilização de técnicas inteligentes de otimização em sistemas de navegação GPS", inserida no Mestrado Integrado em Engenharia Informática e Computação na Faculdade de Engenharia da Universidade do Porto.

Este projeto surgiu essencialmente devido às imposições cada vez maiores por parte dos utilizadores de sistemas de navegação GPS e da necessidade de se adaptar o mais possível o funcionamente desses sistemas a cada utilizador. Desta forma, o uso recorrente que um utilizador faz destes sistemas apresenta-se como um ponto importante a influenciar a escolha futura de rotas, as quais deverão então considerar o histórico de utilização do respetivo utilizador.

Isto levou a que se pretendesse desenvolver uma nova abordagem para algoritmos de *routing* em ambientes dinâmicos mas considerando múltiplos critérios no processo de escolha do caminho mais curto, sem nunca deixar de considerar a complexidade implícita tanto a nível espacial como temporal. Estes múltiplos critérios a usar são efetivamente provenientes das características que fazem parte de um perfil de utilizador.

Para atingir os objetivos deste projeto, inicialmente teve de ser desenvolvido um ambiente de testes no qual foi integrado um algoritmo de *routing* tradicional(*Dijkstra*). Numa segunda fase passou-se ao desenvolvimento de novas abordagens as quais fizeram uso essencialmente de algoritmos de *routing* e heurísticas já existentes, como ponto de partida. Numa fase final a comparação das novas abordagens com os algoritmos já existentes foi um passo fundamental com vista à pretendida melhoria na complexidade dos algoritmos.

Este foi um projeto desenvolvido num ambiente controlado com uma forte componente exploratória e realizado e implementado como uma prova de conceito, pelo que como ponto de partida importante, não só para este como para o desenvolvimento de qualquer projeto inovador, foi feita uma investigação e análise do estado da arte essencialmente ao nível do *routing* dinâmico e dos perfis de utilizador.

Abstract

This document aims to show a general approach about the initial problem and all details about it solution on the dissertation "iRouting: use of intelligent optimization techniques for GPS navigation systems", inserted in the Master in Computer Engineering in the Faculty of Engineering of the University of Porto.

This project arose essentially due to increasing requirements on the part of users of GPS navigation systems and the need to adapt the operation of these systems to each user as much as possible. Thus, the recurrent use of these systems by the users presents itself as an important point to influence the future choice of routes, which should consider the historical using of the respective user.

To achieve the goals of this project was initially developed a test environment in which was integrated a well known routing algorithm(Dijkstra). On the second phase was developed new approaches which essencially use routing algorithms and existing heuristics, as a starting point. On the final phase the comparison between the new approaches and the existing algorithms was a fundamental step towards the desired improvement in the complexity of algorithms.

This project was developed with a strong exploratory component and was carried out and implemented as a proof of concept on a controlled environment. So, as an important starting point for developing any innovative project, was made a research and analysis of the state of the art in the scope of this project, mainly relatively to dynamic routing and user profiles.

Agradecimentos

Antes de mais gostaria de agradecer a todos os que fizeram com que fosse possível chegar ao momento de "enfrentar" esta dissertação.

Começo por aqueles que mesmo tendo ficado mais longe, depois da entrada na Faculdade, nunca deixaram de estar presentes. Por isso, aos poucos mas bons que se mantiveram desde os bons momentos da Secundária de Paredes, resta-me agradecer pelos convívios que continuamos e continuaremos certamente a ter daqui em diante.

Aos que já vinham desde a infância, outrora *GR*, agradeço todos os votos deixados e que tenham todo o sucesso que me desejam, porque sei que o conseguirão.

Mais diretamente relacionado com esta dissertação, quero começar por agradecer ao José Luís Pereira pelo apoio que me prestou com o uso de algum software do LIACC da FEUP. Além disso, o meu muito obrigado ao Prof. Rosaldo Rossetti e ao Prof. António Coelho por todo o apoio ao longo destes 6 meses, e mesmo durante o semestre anterior. Foram duas pessoas sempre disponíveis para ajudar, para orientar da melhor forma possível o desenvolvimento da dissertação, para prestar auxílio nos momentos menos bons que aconteceram e para compreender e estar a par de qualquer situação inesperada que tenha ocorrido.

Por outro lado, permitir estar no ensino superior, acarretar com todas as despesas que isso envolve, estar sempre disponível para a qualquer hora do dia ou da noite auxiliar no que for preciso, permitir novas experiências como a possibilidade de ter ido de ERASMUS, aguentar os tempos sem ir a casa porque os trabalhos têm de ser entregues a tempo e horas, os exames ultrapassados e as queimas vividas, ter de "aturar" os momentos de mau humor que as dificuldades da tese provocam e tudo o mais, não está ao alcance de qualquer um. Por isso, a toda a minha família, em especial aos meus pais e à minha irmã só espero que o meu futuro consiga compensar tudo o que fizeram e passaram por mim.

Por fim resta-me agradecer aos que percorrerem estes 5 anos comigo, vivendo cada momento como o último da nossa vida académica e que estiveram sempre presentes não só nos bons momentos mas principalmente nos maus, porque foram esses que deram mais vontade e prazer de superar com todos vós. Por isso, a todos os que fizeram parte da minha vida académica deixo um obrigado por terem lá estado, em especial aos *Caveiras07*.

Aos que talvez merecessem um obrigado especial, e que sabem quem são, agradecer-vos-ei todos os momentos que iremos passar daqui para a frente e *forever V*.

A todos, o meu MUITO OBRIGADO!!!

Diogo Rocha

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

AIE	Agente Infra-estrutura
ATIS	Advanced Traveller Information Systems
AVE	Agente Veículo Exploratório
AVI	Agente Veículo Informativo
BD	Base de dados
CLARA	CLustering Large Applications
CLARAN	CLustering Large Applications based on Randomized Search
DBSCAN	Density-based spatial clustering of applications with noise
DENCLUE	Density-based clustering
GIS	Geographic Information Systems
GPS	Global Positioning System
ITS	Intelligent Tranportation Systems
LIACC	Laboratório de Inteligência Artificial e Ciência de Computadores
LPA*	Lifelong Planning A*
MAS	Multiagent Systems
MCDM	Multiple Criteria Decision Making
NWAUF	Normalized Weighted Additive Unity Function
OSM	Open Street Map
OSPF	Open Shortest Path First
SGBD	Sistema de Gestão de Bases de dados
SP	Shortest-Path
STING	Statistical Information GRid-based method
SUMO	Simulation of Urban Mobility

Capítulo 1

Introdução

Com este relatório pretende-se documentar detalhadamente todas as fases de investigação, desenvolvimento e análise da tese de mestrado "*iRouting*: utilização de técnicas inteligentes de otimização em sistemas de navegação GPS".

Esta dissertação baseou-se no desenvolvimento de novos algoritmos de *routing*, ou melhoramento de alguns já existentes, de modo a ser possível usar múltiplos critérios no processo de escolha de rotas. As novas abordagens deveriam portanto considerar a existência de perfis de utilizador, cujas características seriam usadas como critérios, de modo a permitirem a sugestão de rotas diferentes e personalizadas.

Algo também expectável foi a obtenção de uma complexidade aceitável, tanto temporal como espacial, pelo que a comparação dos vários algoritmos implementados foi algo que assumiu bastante importância para a satisfação dos resultados obtidos. À parte disto, tornou-se ainda importante a concordância com a empresa NDrive que orientou para esta problemática e que sugeriu alguns aspetos que foram tidos em consideração na tomada de algumas decisões.

Neste capítulo é então abordado o âmbito em que este projeto se insere considerando-se a motivação que levou ao surgimento do mesmo, uma definição mais detalhada do real problema detetado e a especificação dos objetivos finais pretendidos. Além disto, define-se a estrutura global do relatório.

1.1 Motivação inicial

A Navegação GPS representa um elemento central e imprescindível para os sistemas avançados de informação ao viajante, referidos na literatura como *Advanced Traveller Information Systems* (ATISs), tanto que já vem sendo utilizada e desenvolvida há bastante tempo. Apesar disso, pouco avanço se tem verificado tanto ao nível dos algoritmos de escolha de rotas como da sua documentação na literatura, mesmo sendo este um dos pontos mais importantes deste tipo de sistemas de informação.

Foi exatamente da análise desta situação que surgiu grande parte da motivação para esta dissertação, visto que a satisfação dos utilizadores é um ponto chave nesta questão. Assim, as exigências destes aumentam de dia para dia pois o surgimento de um novo avanço tecnológico cria a expectativa de que mais outro se aproxima.

Desta forma, o desenvolvimento de novos algoritmos de *routing*, sendo estes personalizados, leva não só a uma melhoria na qualidade da informação prestada, porque são satisfeitas as necessidades e/ou preferências do utilizador, como obviamente se eleva o nível de satisfação de quem faz uso destes sistemas de informação. Para isso, torna-se também necessário atingir-se algoritmos eficientes de modo a que as melhorias conseguidas o sejam de forma consistente, garantindo assim o bom desempenho do processo de escolha de rotas.

1.2 Definição do Problema

Se se pretender indicar um ponto como principal elemento a inovar nesta dissertação, esse ponto é claramente o processo de escolha de rotas em sistemas de navegação GPS. Isto porque esta escolha ainda é feita usando algoritmos tradicionais, como é o caso do *Dijkstra* e do A*.

Efetivamente, estes são dois dos algoritmos mais conhecidos e usados em problemas de caminho mais curto (*Shortest Path* - SP) e, apesar de não serem tão eficientes em ambientes dinâmicos, como é o caso do tráfego em tempo real, como o conseguem ser em redes estáticas, continuam a ser suficientes para responder aos problemas como aqueles com que os sistemas de navegação GPS se deparam.

Isto acontece porque os sistemas de navegação GPS continuam ainda hoje a simplificar o problema de escolha de rotas reduzindo-o a encontrar o "caminho mais curto" entre uma determinada posição inicial e uma posição final.

No entanto a expressão "caminho mais curto" pode ter várias interpretações e o que acontece atualmente é que, mesmo usando melhoramentos dos dois algoritmos referidos anteriormente, as funções de custo continuam a considerar essencialmente tempo e/ou distância como medidas de desempenho.

No entanto, com o surgimento dos Sistemas Inteligentes de Transportes, ou *Intelligent Transportation Systems* (ITSs), e do conceito de "Transportes do futuro" tem-se questionado a hipótese de considerar novas medidas de desempenho a serem usadas como múltiplos critérios nos algoritmos de escolha de rotas visto que de acordo com os hábitos ou preferências dos utilizadores, o que para um é o SP para chegar a um determinado destino, para outro pode não o ser, mesmo que as posições de origem e destino sejam as mesmas.

Deve-se isto ao facto de que dois utilizadores que pretendam fazer uma viagem com os mesmos pontos de início e fim, apesar de supostamente a rota a sugerir ser a mesma, na realidade isso pode não acontecer e os caminhos sugeridos serem até bastante diferentes. Isto pode acontecer se se pensar que um utilizador pode na maioria das suas viagens escolher a viagem mais rápida e com postos de combustível como ponto de interesse, por exemplo, enquanto outro preferir rotas em que

terá de fazer o menor número de manobras possível. Neste caso, a necessidade de ir de encontro aos hábitos de utilização de cada um levaria às possíveis diferenças nas rotas.

No entanto esta inclusão de novas medidas de desempenho nos algoritmos de *routing*, como são tradicionalmente abordados, leva a que a sua complexidade tanto temporal como espacial tenda a agravar-se consideravelmente, podendo mesmo tornar-se num problema NP-completo.

É precisamente neste ponto que esta dissertação se insere. Pretende-se assim desenvolver técnicas inteligentes que permitam obter algoritmos de *routing* eficientes, a serem aplicados em ambientes dinâmicos, mas que considerem as características dos perfis de utilizador como múltiplos critérios a incorporar.

1.3 Objetivos

Tendo em conta o problema já definido, esta dissertação tem como objetivo principal a investigação, desenvolvimento e análise de algoritmos de escolha de rotas, a serem aplicados em sistemas de navegação GPS. Além disso é também pretendido o uso de múltiplos critérios a influenciar o processo de *routing*.

Assim sendo, mais detalhadamente pode-se dizer que os objetivos passam por:

1. Investigar heurísticas e técnicas inteligentes de otimização, com aplicação em algoritmos de melhor caminho;
2. Definir e criar perfis de utilizadores padrão para serem integrados nas novas abordagens;
3. Adaptar os algoritmos de *routing* e as funções de custo para passarem a considerar perfis de utilizador como nova métrica de avaliação;
4. Análise dos algoritmos desenvolvidos em grafos de diferentes complexidades;
5. Comparação dos novos algoritmos com as abordagens tradicionais.

1.4 Estrutura da Dissertação

Esta dissertação, além deste capítulo inicial de contextualização do trabalho a desenvolver, contém ainda cinco outros capítulos.

Assim, no Capítulo 2 pretende-se fazer uma revisão do estado da arte, no âmbito do projeto a desenvolver, sendo apresentadas as abordagens mais usadas atualmente no contexto em que esta dissertação se insere.

No Capítulo 3 pretende-se essencialmente modelar o problema em todas as suas componentes e assim conseguir apresentar uma metodologia a seguir para abordar e solucionar cada um dos pontos que serão necessários ultrapassar para se atingirem os objetivos finais.

Seguidamente, no Capítulo 4, serão documentadas todas as etapas que acabaram por acontecer de modo a seguir a metodologia definida. Desta forma é dada uma visão pormenorizada de tudo o

que foi desenvolvido e implementado no âmbito desta tese e que será alvo de análise no Capítulo 5. Este capítulo conta com os principais testes, aos quais o projeto foi submetido, bem como uma análise das novas abordagens tanto ao nível das soluções obtidas como da eficiência de cada uma delas.

Por último serão expostas, no Capítulo 6, as conclusões finais que se podem tirar desta dissertação, não só relativamente à satisfação dos objetivos propostos mas também tendo em conta o trabalho que poderá ser desenvolvido futuramente no seguimento desta dissertação.

Capítulo 2

Revisão Bibliográfica

Este capítulo visa apresentar um revisão geral do que vem sendo desenvolvido no âmbito do *routing* dinâmico em sistemas de navegação GPS. Desta forma primeiro serão abordados alguns dos algoritmos mais usados, atualmente, bem como alguns dos aspetos mais promissores nesta área, como é o caso da Meta-Heurística baseada no comportamento das colónias de formigas ou do pré-processamento com base em hierarquias.

Posteriormente é dada também bastante relevância ao perfis de utilizador, visto ser uma aspeto preponderante no âmbito desta dissertação, para que seja possível o uso não só de múltiplos critérios em problemas de SP mas também de técnicas inteligentes de otimização no processo de escolha de rotas. Esta parte englobará ainda uma profunda análise sobre medidas de proximidade e algoritmos de *clustering*, tanto para classificar como para agrupar perfis.

2.1 Escolha de rotas

O problema de escolha de rotas entre dois pontos tem sido amplamente investigado nas mais variadas áreas, como é o caso da Inteligência Artificial ou da Investigação Operacional, por exemplo. No entanto, e apesar de já haverem algumas abordagens, uma questão que continua a ser desafiadora é desenvolver algoritmos eficientes e menos complexos computacionalmente para problemas de SP, a serem aplicados em redes dinâmicas [HWZ07]. Redes dinâmicas não são mais do que grafos em que as suas entidades(vértices, arestas, pesos) podem variar com o decorrer do tempo [NL08].

Dois dos algoritmos mais conhecidos para escolha do SP são efetivamente o algoritmo de *Dijkstra* [Dij59] e o algoritmo A* [HNB68]. Apesar de as abordagens existentes que usam estes dois algoritmos terem sido desenvolvidas para redes estáticas e não serem tão eficientes quando aplicadas diretamente em ambientes dinâmicos [HWZ07], esses algoritmos serão analisados até para que seja possível fazer comparações e tirar conclusões.

Adicionalmente, mas não menos importante, acontece que os algoritmos de *routing* tradicionais consideram apenas um critério como métrica de escolha de rotas [MGT06], pelo que esta situação é também alvo de atenção na Secção 2.2.

2.1.1 Algoritmo de *Dijkstra*

O algoritmo de *Dijkstra* resolve problemas de SP, com um único ponto de partida, em grafos estáticos dirigidos, com pesos não-negativos, em tempo polinomial [NL08]. Para apresentar o algoritmo que agora tem o seu nome, E. W. Dijkstra considerou dois problemas [Dij59]:

1. Construir uma árvore de comprimento total mínimo entre os n nós de um grafo em que existe um e um só caminho entre dois nós;

2. Encontrar o caminho de comprimento total mínimo em dois nós dados s e t.

Antes de se perceber os passos do algoritmo é importante ter a noção de que cada vértice vai ter associada uma distância à origem, que inicialmente será infinita, e um nó pai, à partida nulo. Todos os nós do grafo estarão divididos em dois conjuntos: nós marcados mas ainda não analisados (denotado conjunto *S'*) e nós analisados mas permanentemente marcados (denotado conjunto *S*).

Considerando então que para dois quaisquer nós u e v, $g(u)$ representa a distância de u à origem e $c(u,v)$ representa o peso associado à ligação entre u e v, começa-se por alterar a distância do nó origem para 0 e passá-lo do conjunto *S'*, que contém todos os nós no início, para o conjunto S. Em seguida consideram-se as ligações, do nó s que foi transferido para *S*, para todos os outros possíveis nós. Se $g(n) > g(s) + c(s,n)$, para qualquer nó n atingível por s então $g(n)$ passa a ter o valor de $g(s) + c(s,n)$ e s é agora pai de n.

Um aspeto importante no seguimento deste algoritmo é a escolha do próximo nó a explorar. O que acontece é que os nós de *S'* são geridos usando uma *priority queue* e ordenados consoante a sua distância ao nó inicial. Desta forma, e devido à atualização das distâncias para os sucessores do último nó que foi adicionado a *S*, o nó que está no topo da fila será o próximo a ser analisado porque é o que apresenta menor distância à origem.

Quando o nó destino é atingido ou quando a fila de prioridade fica vazia, o algoritmo termina.

Ao longo deste processo vai sendo construída a árvore referida anteriormente com os nós que vão sendo transferidos para *S*, e respetivas ligações [HWZ07] [NL08] [Dij59].

Uma variação apresentada ao algoritmo de *Dijkstra* passa por uma pesquisa bi-direcional em que, em vez de se construir apenas um árvore de SP tendo o nó s como raiz, também se constrói a mesma árvore mas considerando como raiz o nó de destino t no grafo inverso. De salientar que para qualquer ligação (u,v) que exista neste grafo reverso, existia a mesma ligação no sentido contrário no grafo original [NL08].

A vantagem desta abordagem é que assim que um nó v seja explorado e passe para o conjunto *S*, em ambas as pesquisas, garante-se que a concatenação dos menores caminhos $s \to v$ e $v \to t$, encontrados nas respetivas abordagens, é um SP $s \to t$.

A grande desvantagem deste algoritmo é que não usa heurísticas e portanto expande igualmente em todas as direções da rede pelo que vai acabar por explorar um vasta área, e que por vezes não é necessária, antes de atingir o nó final. O mesmo já não acontece se se utilizar a abordagem bi-direcional visto que o espaço de pesquisa irá ser reduzido, até um fator de 2 (ver Figura 2.1 e Figura 2.2). Apesar disto, é garantido que se atinge sempre o melhor caminho possível [HWZ07].

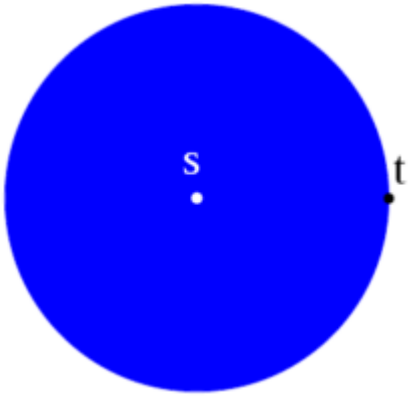


Figura 2.1: Espaço de pesquisa do algoritmo de *Dijkstra* no caso Uni-direcional [NL08].

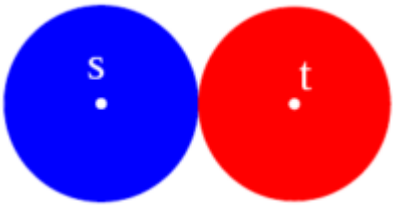


Figura 2.2: Espaço de pesquisa do algoritmo de *Dijkstra* no caso Bi-direcional [NL08].

Além desta duas abordagens enunciadas, este algoritmo também já foi aplicado em redes dinâmicas tanto para encontrar caminhos mais curtos de todos os nós para um só e vice-versa em [Cha98]. No entanto, foi de notar que a abordagem bi-direcional não pode ser aplicada visto que o tempo de partida no nó de destino é desconhecido [NL08].

2.1.2 Algoritmo A*

O algoritmo A* tem como base o algoritmo de *Dijkstra* anteriormente enunciado. No entanto passa a considerar uma heurística com o objetivo de acelerar o processo de pesquisa do melhor caminho entre o nó de origem e o nó de destino.

O processo usado para calcular o melhor caminho passa por, tal como no algoritmo de *Dijkstra*, dividir os nós em dois conjuntos: nós já analisados e determinados e os nós ainda por analisar. No entanto, no processo de análise de cada nó, em vez de se considerar apenas a distância à origem $g(n)$, para o nó n, é também considerada uma estimativa da distância/proximidade deste até ao

nó objetivo, denotada por $h(n)$. Como o caminho ainda não está completo, este valor não está determinado e portanto tem de ser "adivinhado"[HWZ07].

Desta forma, o próximo nó a ser analisado vai ser não aquele com menor valor de $g(u)$ mas o que tem uma soma de $g(u)+h(u)$ menor [ZON]. Ao usar-se uma heurística, em que se estima um valor de $h(n)$ que não é superior à real distância entre o mesmo e o nó de destino, pode-se garantir que [NL08]:

- O A* encontra sempre os caminhos mais curtos;
- Nunca explora mais nós do que os que são explorados no algoritmo de *Dijkstra*;
- Se $h(n)$ for realmente uma boa aproximação abaixo do valor real, o algoritmo A* orienta a pesquisa em direção ao nó objetivo t em vez de expandir em todas as direções como acontecia no caso do *Dijkstra* (ver Figura 2.3).

Figura 2.3: Espaço de pesquisa do algoritmo A* [NL08].

Como já referenciado [NL08], este algoritmo foi aplicado pela primeira vez num cenário dependente do tempo [CL02], com a propriedade FIFO, sendo que uma versão muito mais eficiente foi também apresentada usando pontos de referência [DW07]. Estes pontos de referência não são mais do que uma técnica de pré-processamento que é baseada na desigualdade triangular.

As duas propriedades referidas anteriormente podem ser enunciadas como o seguinte [ZON]:

- **FIFO** - Para todos os nós n e $t_1 \leq t_2$, $\quad t_1 + h(n,t_1) \leq t_2 + h(n,t_2)$;
- **Desigualdade Triangular** - Para todas as arestas $e = (u,v)$ e tempo t, $h(u,t) \leq c_e(t) + h(v,t+c_e(t))$, em que $c_e(t)$ representa o custo de usar a aresta e no tempo t e $h(u,t)$ define a estimativa da distância do nó u ao nó objetivo, no tempo t.

Apesar da redução do espaço de pesquisa, o algoritmo A* continua ainda a explorar nós que não vão levar ao caminho ótimo pelo que torna-se importante analisar o algoritmo LPA* (*Lifelong Planning A**) que foi proposto exatamente com o objetivo de acelerar o processo de pesquisa do A* [HWZ07].

2.1.3 Algoritmo LPA*

Este algoritmo, que surgiu como um incremento ao A*, baseia-se no reuso de informação de pesquisas anteriores para encontrar o SP. O LPA* encontra repetidamente SPs para dois pontos dados, um de origem e outro de destino num dado grafo. Pode-se então dizer que este algoritmo

resolve problemas de planeamento de caminhos em grafos finitos conhecidos em que o custo das arestas aumenta ou diminui com o decorrer do tempo [KLF04].

O LPA* tem então duas pesquisas: a primeira é igual à que é usada no A* mas as seguintes pesquisas são muito mais rápidas porque são reusadas as partes da árvore de pesquisa anterior que são idênticas na nova árvore de pesquisa. Isto pode reduzir drasticamente o tempo de pesquisa, por exemplo, em situações em que há apenas ligeiras alterações no grafo ou quando as alterações surgem já muito perto do nó objetivo e assim é possível reutilizar uma grande parte da árvore de pesquisa anterior [HWZ07].

Antes de se passar diretamente à explicação dos princípios enunciados por este algoritmo, convém definir três conjuntos de nós:

- Conjunto finito que contêm todos os nós do grafo, denotado como S;
- Conjunto de todos os nós sucessores de $n \in S$, denotado como $succ(n) \subseteq S$;
- Conjunto de todos os nós antecessores de $n \in S$, denotado como $pred(n) \subseteq S$.

Considere-se ainda o custo de se mover de um nó n para um nó $n' \in succ(n)$ dado por $0 < c(n,n') \leq \infty$, a distância de um nó n ao nó de origem s dada por $g(n)$ e a heurística $h(n)$, apresentada no algoritmo A*, que representa uma previsão da distância do nó n ao nó objetivo t.

A distância à origem tem de satisfazer as seguintes condições [KLF04]:

- $g(n) = 0$, se $n = s$, em que s é o nó de partida;
- $g(n) = min_{n' \in pred(n)}(g(n') + c(n',n))$, nos restantes casos.

Enquanto que, para que os nós sejam consistentes, é também preciso garantir que [HWZ07]:

- $h(n) = 0$, se $n = t$, em que t é o nó de destino;
- $h(n) \leq c(n,n') + h(n')$ para todos os nós $n \in S$ e $n' \in succ(n)$ com $n \neq t$.

Além destes dois valores, $g(n)$ e $h(n)$, é também mantida uma terceira estimativa, denominada *rhs*, que consiste na grande novidade deste algoritmo. Assim, para cada nó $n \in S$, $rhs(n)$ representa uma nova estimativa da distância de n à origem, baseada nos valores de g, mas que está um passo à frente pelo que é uma estimativa melhor informada do que g. Esta estimativa satisfaz a seguinte relação:

- $rhs(n) = 0$, quando n é o nó de origem;
- $rhs(n) = min_{n' \in pred(n)}(g(n') + c(n',n))$, nos outros casos.

O valor de *rhs* vai então ser importante na escolha do próximo nó a ser retirado da fila de prioridade, mantida pelo LPA*, para analisar. Esta fila de prioridade mantém todos os nós localmente inconsistentes que são aqueles cujo valor de g é diferente do de *rhs*. Este conceito, apesar de

importante, não leva à necessidade de se fazer com que todos os nós sejam consistentes no LPA* porque, em vez de isso, é usada a heurística h para convergir a pesquisa e atualiza-se só os nós envolvidos no SP [KLF04].

Posto isto, o próximo nó da fila de prioridade a ser analisado será sempre aquele que tiver menor $k(n)$, como é denominado na literatura, e que corresponde a um vetor com duas componentes($k_1(n)$ e $k_2(n)$) em que:

- $k_1(n) = min(g(n), rhs(n)) + h(n)$;
- $k_2(n) = min(g(n), rhs(n))$.

Se se tiver em conta que no algoritmo A* g(n) e rhs(n) correpondem ao mesmo valor, pode-se concluir que o A* e o LPA* têm comportamentos semelhantes visto que $k_1(n)$ e $k_2(n)$ correspondem a f(n) = g(n) + h(n) e a g(n), respetivamente.

No entanto, e apesar de ser bastante mais eficiente do que o A* devido ao reuso de informação previamente calculada, este algoritmo acaba por não conseguir dar resposta a casos em que o nó de origem é alterado ao longo do tempo. Isto acontece porque as distâncias à origem são então alteradas e deixam de ser válidas para o nó atual pelo que é necessário fazer uma nova análise a partir do zero, perdendo-se assim a grande vantagem do LPA* de fazer uma pesquisa incremental. De modo a colmatar esta falha foi proposto um melhoramento ao LPA* [HWZ07].

2.1.4 Melhoramento do LPA*

O principal objetivo deste melhoramento passa por encontrar uma alternativa em que se consiga manter o valor da distância de cada nó de um grafo ao nó de origem para que seja possível usá-lo em pesquisas sub-sequentes, mesmo que o nó inicial seja alterado, de modo a conseguir-se tirar partido do aspeto mais importante do algoritmo LPA*.

Tendo isto em conta, e considerando que um objeto em movimento ao recalcular sucessivamente a sua rota para atingir um determinado objetivo vai estar consecutivamente a alterar a sua "posição inicial", o único ponto que irá permanecer sempre o mesmo é o nó de destino.

Foi precisamente esta a observação que levou a que em [HWZ07] fosse proposto uma modificação ao algoritmo LPA* que passa por inicialmente considerar que se um utilizador pretende encontrar o SP entre dois pontos s e t, com $s, t \in S$, é possível trocar a direção da pesquisa passando t a ser o nó de origem e s o de destino.

Depois desta alteração, o nó origem (originalmente o nó objetivo), nunca mais sofre alterações pelo que a distância de cada nó $n \in S$ ao agora nó de origem do SP a calcular vai permanecer inalterável e assim pode ser usada em pesquisas sub-sequentes. No entanto, é preciso ter em atenção que, para cada nó, o valor da heurística *h(n)* deve ser modificada e ajustada à inversão efetuada e o valor de cada aresta deve também ser alterado, no caso de estarmos a falar de grafos dirigidos.

Não diretamente relacionado mas, conjuntamente com esta melhoria, foram também propostas duas abordagens de modo a ser possível diminuir ainda mais o espaço de pesquisa do SP entre

um dado ponto inicial e um dado ponto final. Isto foi feito prioritariamente para otimizar o processo de escolha do próximo nó a analisar para se diminuir o tempo despendido a analisar nós desnecessários.

De entre estas duas técnicas uma delas baseia-se essencialmente no seguinte conceito: o caminho em linha reta entre dois pontos u e v é sempre o caminho mais curto, se esse caminho existir. No entanto raramente este caminho existe na realidade e portanto o objetivo passa a ser encontrar o ponto mais próximo desta linha reta virtual, como sendo o próximo ponto a analisar. Desta forma, esta nova heurística é definida como o seguinte [HWZ07]:

Considere-se duas distâncias, h_1, que representa a distância em linha reta entra um determinado nó e o nó objetivo muitas vezes usada no A*, e $h_2 = d(n-L)$, em que L é a linha virtual que liga o nó n ao nó objetivo e a função d representa a distância Euclidiana entre o referido nó e essa linha. Estas duas medidas são então utilizadas para ordenar os nós a examinar, na fila de prioridade. Num primeiro nível os nós são priorizados pela soma de $g(n)$ com $h_1(n)$, tal como no A*, e posteriormente os nós priorizados com valores semelhantes no primeiro nível serão ordenados com base na distância h_2.

A outra técnica, anteriormente referida, passa por logo à partida limitar o espaço de pesquisa usando o Retângulo Limitado Mínimo (ou *Minimum Bounded Rectangle*) como é referido na literatura. Assim é delimitada uma área, envolvendo os nós de origem e destino, com base no pressuposto de que em redes de tráfego típicas cada aresta está apenas ligada aos seus nós vizinhos e o tempo de viagem numa ligação está relacionado com o seu tamanho. Assume-se portanto que os nós fora desta área têm baixa probabilidade de fazerem parte do SP [FSR06]. Foi também enunciado o uso de uma elipse como possível forma geométrica a usar para definir esta área [HWZ07], o que não vai ser aqui algo de profunda análise visto ser algo que poderá variar de acordo com a especificidade de cada problema.

2.1.5 Meta-Heurística das Colónias de Formigas

Um princípio que tem vindo a assumir grande importância na otimização de algoritmos de *routing* é o simples comportamento das formigas na cooperação para realizar certas tarefas, como a procura de comida, e o sucesso que conseguem atingir.

Basicamente, e tendo como base o exemplo de procura de comida, o que acontece é que as formigas começam por vaguear procurando atingir o objetivo de encontrar comida. Quando conseguem encontrar comida, elas retornam enquanto vão deixando um rasto de feromonas. Posteriormente outras formigas vão também vaguear e possivelmente encontrar e seguir o referido rasto reforçando-o. Este rasto com o passar do tempo vai ser preponderante para que as formigas escolham o melhor caminho para encontrar comida devido à característica de evaporação das feromonas, ou seja, se não houver novas formigas a passar pelo mesmo sítio e a reforçar o rasto, este vai acabar por evaporar e deixar de influenciar as seguintes formigas na escolha de um caminho, que possivelmente as levará até um local com comida.

Assim sendo, em caminhos mais curtos o rasto vai sendo reforçado mais vezes do que num caminho mais longo, porque vão haver mais formigas a percorrer este caminho por intervalo de

tempo. Isto vai levar a que ao fim de um certo tempo todas as formigas estejam a percorrer o caminho ótimo para encontrar comida, como é ilustrado na Figura 2.4

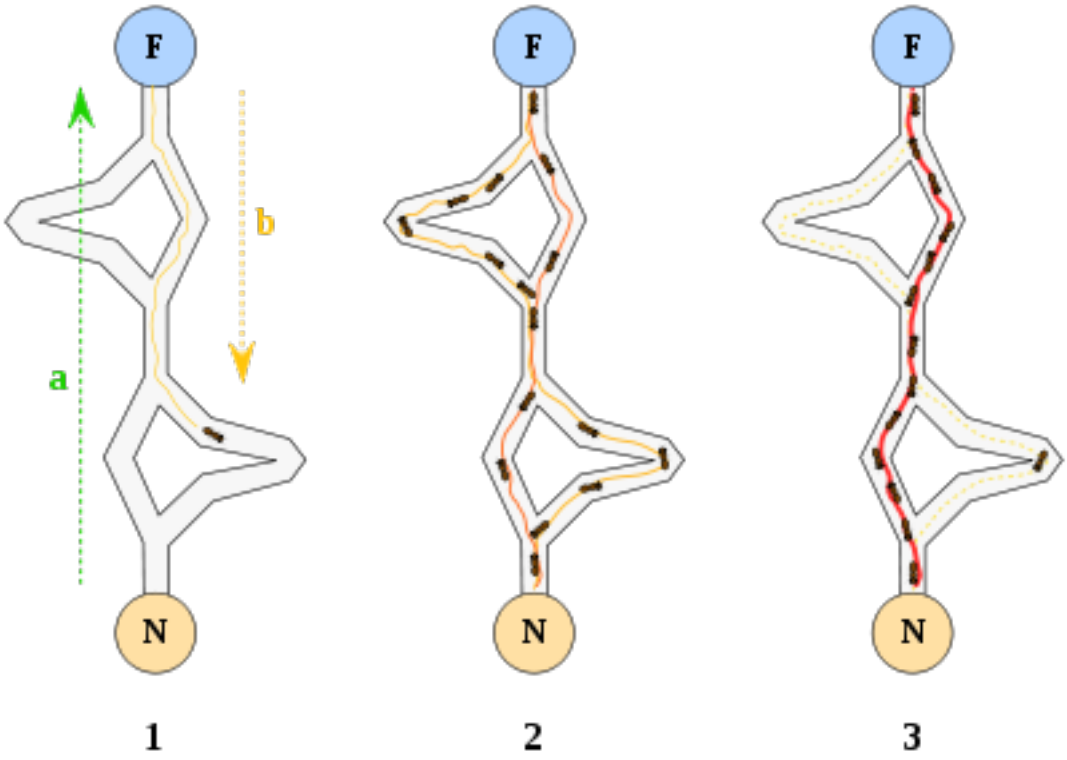


Figura 2.4: Comportamento das formigas na procura de comida [Ram11].

Nessa mesma imagem pode-se observar que as formigas seguem no sentido *a* com o objetivo de encontrar comida e quando o conseguem retornam por um caminho no sentido *b* deixando então o dito rasto de feromonas. Na situação 1 é possível identificar a primeira formiga a efetuar esse processo aleatoriamente sendo que com o passar do tempo, como é possível ver na situação 2, várias formigas vão seguindo o reforço deixado por outras e os caminhos com menor distância à comida vão ter uma maior concentração de feromonas. Isto leva então a que na situação 3 praticamente todas as formigas ja usem o caminho ótimo para chegar do seu habitáculo *N* ao depósito de comida *F*.

Como já verificado, uma importante característica deste tipo de comportamento coletivo é que a informação global está disponível no local [WHH07]. Tendo isto em conta e considerando que o rasto de feromonas indica a direção em que é necessário mover-se para encontrar comida num local remoto, podemos chegar a dois princípios de engenharia:

- Colocar informação relevante como sinais no ambiente para fornecer localmente informações sobre propriedades globais;

- Limitar a validade desta informação, como acontece com a evaporação das feromonas, e atualizá-la enquanto permanecer válida para habilitar o sistema a lidar com mudanças e distúrbios.

Foi precisamente com base nestes princípios que foram apresentadas abordagens de sistemas de multiagentes representativos, denominados na literatura como *delegate multiagent systems(MASs)*. Um MAS delegado consiste num número de agentes autónomos que estão situados ou embutidos no ambiente e que podem localmente observar e atuar no ambiente pelo que a coordenação do comportamento global resulta da interação entre os diferentes agentes. Sistemas de *routing* de veículos antecipatório usando estes MAS foram delineados pela primeira vez em 2007 [WHH07] e mais recentemente tornados mais robustos em 2011 [CHW11].

Assim, consideremos três tipos de agentes:

- Agentes veículo - representam um veículo que pretende escolher uma rota a seguir e tem duas grandes responsabilidades bem distintas: por um lado explorar e procurar possíveis rotas para o seu destino e desse conjunto selecionar uma que pretende seguir; por outro, deve informar os outros agentes da sua intenção para que estes possam incorporar esta previsão de ocupação na sua própria exploração;

- Agentes infraestrutura - representam e gerem os principais elementos das estradas e as informações neles contidas evaporam com o tempo a menos que sejam atualizadas pelos agentes veículo;

- Ambiente virtual - representa a rede física de trânsito mapeada numa representação num grafo.

Em [CHW11] são ainda propostos dois agentes que irão simular o comportamento das formigas, anteriormente descrito, de modo a não ser necessário haver diretamente comunicação entre os agentes veículo e infraestrutura. Assim teremos:

- Formigas de exploração - têm como objetivo explorar possíveis rotas entre a posição atual e o destino obtendo informação dos agentes infraestrutura e reportando-a para o agente veículo que as enviou;

- Formigas de intenção - são enviadas quando o agente veículo escolhe uma rota, de modo a informar os agentes infraestrutura da sua intenção. Isto permite que estes possam ter informação das previsões de tráfego para fornecerem a futuras formigas exploratórias.

Desta forma, o condutor poderá a qualquer momento alterar a sua rota de modo a satisfazer melhor os seus objetivos, sendo para isso importante não esquecer o princípio da evaporação falado anteriormente para que a intenção anterior deixe de ser considerada ao fim de um determinado intervalo de tempo.

Esta é de facto uma abordagem com enormes vantagens, desde logo pelo facto de ser uma abordagem descentralizada, evitando assim o aglomerar de informação num centro de controlo e

também porque são feitas previsões de trânsito em vez de se utilizar informações em tempo real, o que permite antecipar o *routing* de um veículo. Outro ponto importante a referir é que a abordagem mais recentemente proposta considera apenas o tempo como métrica na escolha da intenção, pelo que poderia ser um ponto de melhoria passar a considerar-se múltiplos critérios [CHW11].

2.1.6 Routing Hierárquico

Algo também bastante importante e ainda não abordado é o pré-processamento. Esta é de facto uma técnica bastante interessante que permite acelerar a cálculo de rotas e se torna bastante vantajosa, essencialmente, em grafos de grande dimensão.

É precisamente aqui que se insere o *routing* hierárquico, como abordado em [Sch08], nomeadamente através de três algoritmos: *Highway hierarchies*, *Highway-node routing* e *Transit-node routing*. O primeiro destes algoritmos consiste basicamente em classificar as estradas, por importância, de acordo com a respetiva hierarquia que está inerente à rede de estradas. Desta forma pretende-se ignorar tanto mais estradas de menor importância quanto maior for a distância aos nós de origem e destino, considerando uma pesquisa bidirecional.

Relativamente ao algoritmo de *Highway-node routing*, apresenta especial vantagem na capacidade que tem para reagir a situações inesperadas, como engarrafamentos. Isto deve-se principalmente a duas considerações:

- O peso associado a uma qualquer aresta pode mudar a qualquer instante;
- O procedimento que constrói uma rede de estradas baseia-se apenas em pesquisas locais.

Tendo isto em conta, a hierarquia de estradas pode ser eficientemente atualizada, quando ocorre uma alteração numa aresta, visto que só é necessário repetir a pesquisa na zona afetada.

O algoritmo *Transit-node routing* baseia-se essencialmente nos seguintes passos :

- Encontrar, para o grafo específico, o conjunto de nós de transição.
- Calcular a distância entre todos esses nós e manter uma tabela com essas distâncias.
- Dividir o percurso pretendido em três fases: ir da origem a um nó de transição, navegar entre dois nós de transição e ir desde o segundo nó de transição até ao destino.

Tendo isto em consideração, a grande vantagem deste algoritmo passa pela segunda fase e pelo pré-processamento das ligações entre os nós de transição. Estes não são mais do que nós que, considerando uma hierarquia de estradas, nos leva ao nível mais elevado.

Este algoritmo comporta-se tanto melhor quanto maior for a distância entre a origem e o destino do percurso pois pode-se escolher uma distância computada à priori muito maior. Desta forma é sempre preciso ter em atenção se se justifica este algoritmo visto que temos de reservar bastante memória para guardar os cálculos iniciais, o que pode até não ser vantajoso. Outro aspeto também a ter em consideração é que o número de nós de transição não é assim tão elevado quanto

isso e podem não existir nós com estas características no caminho mais curto, entre a origem e o destino, o que também acaba por inviabilizar esta abordagem.

Por forma a resolver estas situações pode considerar-se um filtro de localidade, a aplicar aos nós de origem e destino, e manter apenas as ligações entre os nós considerados por este filtro, chamados de nós de acesso, o que reduz o consumo de memória e facilita a implementação deste algoritmo. O filtro de localidade referido é dependente do problema em questão e visa definir uma área envolvente ao nó onde é aplicado e que se considera ser suficiente para atingir o nível mais elevado da hierarquia de estradas.

Posto isto, e como foi percetível, todas estas abordagens se baseiam na hierarquia de estradas que para ser construída envolve dois passos:

1. Redução de arestas;

2. Redução de nós.

O primeiro passo consiste em, considerando um raio de vizinhança pré estabelecido, definir para cada nó o seu bairro, que não é mais do que o conjunto de nós onde se consegue chegar sem ultrapassar a distância máxima referida. Tendo os bairros formados, tem-se que definir os pontos que permitem ligar um bairro aos restantes e definir as melhores ligações entre os mesmos. Desta forma passa-se já a ter num nível hierárquico superior relativamente ao grafo de origem, visto que já se considera bairros como sendo um único nó e as várias ligações que existiam entre os mesmos foram reduzidas a uma só (a melhor entre elas). Na Figura 2.5 estão representadas as vizinhanças dos nós s e t e a aresta que liga um bairro ao outro.

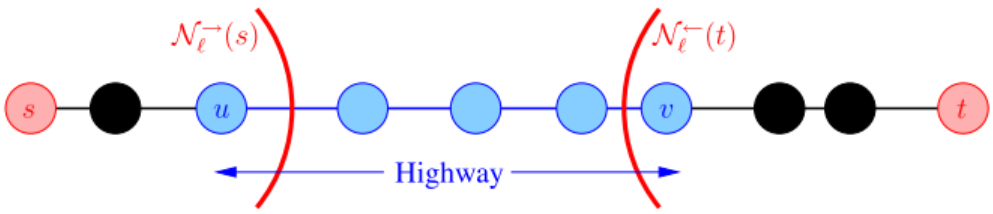


Figura 2.5: Ligação entre dois bairros representados pela vizinhança de s e t [Sch08].

Tendo este novo nível então definido, passa-se para o segundo passo de modo a definir-se aquilo que é chamado na literatura de *core*. Isto consegue-se através da redução de nós, referidos na literatura como *bypassed nodes*, que consiste em eliminar os nós de baixo grau (nós em que o número de arestas incidentes é baixo) e construir arestas de atalho como é ilustrado na Figura 2.6.

Estes dois passos podem ser repetidos iterativamente, aplicando-se sempre a redução de arestas ao *core* do nível anterior atingindo-se assim uma hierarquia com tantos níveis quantos os desejados.

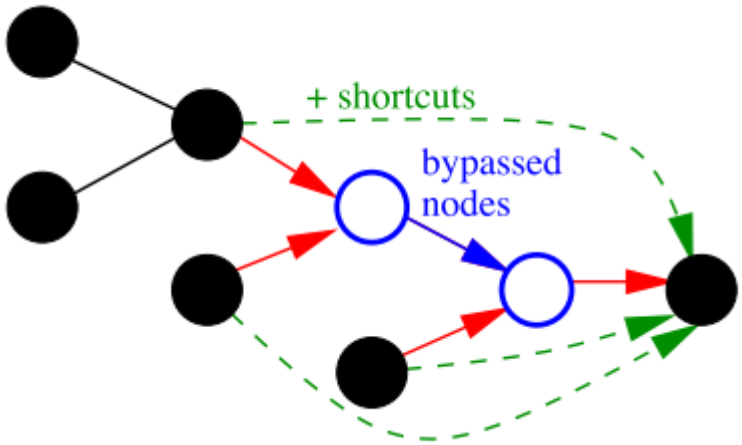


Figura 2.6: *Core* de uma rede de estradas através da redução de nós [Sch08].

2.2 Perfis de Utilizador

Esta secção aborda temas extremamente importantes no desenvolvimento deste projeto de dissertação. Os perfis de utilizador assumem-se, então, como um ponto crítico deste trabalho, visto que consistem num grupo de características que representam os aspetos mais importantes que permitem escolher uma rota em detrimento de outra. Desta forma são também "integrados"nos algoritmos de *routing* dinâmico de modo a passar-se a fazer pesquisas tendo em consideração múltiplos critérios, e não apenas o tempo ou distância como acontece habitualmente.

Tendo em conta a possibilidade de existência de uma grande diversidade de perfis de utilizador, torna-se bastante importante conseguir agrupá-los por semelhanças que existam entre as características dos mesmos. Isto assume bastante importância essencialmente para utilizadores recentes que tenham poucos registos de utilização de um sistema de navegação GPS. Desta forma, mesmo que a estatística da sua utilização do sistema não seja deveras relevante ou conclusiva, este utilizador ao apresentar semelhanças, ainda que poucas, com alguns dos grupos já existentes pode assim ver serem-lhe recomendadas rotas possivelmente do seu interesse. Além disso, permite também sugerir rotas usuais de um utilizador a qualquer outro que esteja no mesmo grupo.

O processo de formação de grupos, contendo elementos com um certo grau de proximidade entre si, assenta essencialmente nos seguintes passos [JMF99]:

1. Representação dos padrões (opcionalmente incluindo seleção/extração de características);
2. Definição da medida de proximidade dos padrões apropriada ao domínio dos dados;
3. Agrupamento;
4. Abstração de dados (se necessário);
5. Avaliação dos dados de saída (se necessário).

Assim sendo, torna-se necessário analisar essencialmente os três primeiros passos representados na Figura 2.7, o que irá acontecerá nas secções 2.2.1, 2.2.2 e 2.2.3, respetivamente.

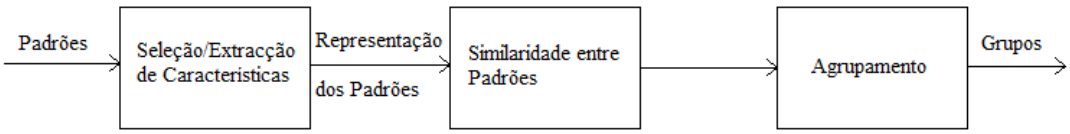


Figura 2.7: Primeiras três etapas do agrupamento de dados.

2.2.1 Criação de perfis

A criação de perfis de utilizador, ou de quaisquer outros conjuntos de dados, para serem futuramente analisados e agrupados, é um passo bastante importante. Isto essencialmente porque a seleção do conjunto de características a fazer parte de um perfil e a maneira como são representadas pode levar tanto a que o processo de *clustering* seja muito eficiente como bastante complexo. Este processo assenta basicamente na formação automática de grupos de dados, perfis de utilizador neste caso, consoante o grau de semelhança que apresentam.

Os conjuntos de dados deste tipo, também denominados de *clusters*, são descritos pelas características que o constituem e são usualmente representados como um vetor de características. O tipo de dados que estas características representam são fundamentais para determinar os correspondentes mecanismos de medidas de similaridade ou dissimilaridade, a serem analisadas posteriormente, pelo que estas técnicas são dependentes do problema em questão. Estes dados são essencialmente de dois tipos: quantitativos (contínuos, discretos ou intervalos) e qualitativos (nominais ou ordinais).

Assim, usualmente representa-se o conjunto de dados a analisar por uma matriz de padrões multidimensional n X m em que n representa o número de perfis a considerar e m o número de características que fazem parte de um perfil [JMF99].

Um problema que se pode colocar face a isto é o facto de uma característica aparecer num perfil e não noutro. Para contornar isto, pode-se proceder a uma ou mais transformações a todas as características, processo denominado na literatura como extração de características, de forma a normalizá-las e a obter um conjunto de dados apropriado a ser usado nas fases seguintes do processo de *clustering*. No entanto, estas transformações podem ser um pouco arriscadas devido à possibilidade de perda de informação, pelo que se deverá ser muito rigoroso e ter em atenção se o problema em questão e as características associadas apresentam condições para serem normalizadas[JMF99]. Contudo, e como forma de contornar este risco pode-se optar por considerar uma métrica de medida de proximidade entre perfis que permita comparar vetores com diferente número de elementos, se aplicável ao problema em questão.

2.2.1.1 Múltiplos Critérios

Algoritmos de *routing* tradicionais consideram apenas um critério como métrica, como por exemplo o tempo. O que acontece é que os critérios a usar em escolhas de rotas geralmente entram em conflito e competem uns com os outros pelo que, na realidade, uma rota que otimiza um objetivo não vai otimizar os outros.

Como apresentado por Malakooti et al. [MGT06], várias tentativas tem sido feitas, ao longo do tempo, com o objetivo de tentar resolver problemas de *routing* com múltiplos critérios para vários tipos de redes. Climaco et al. [CCP03] implementou uma abordagem de *routing* bi-critério para redes multimédia, com o objetivo de minimizar o consumo de recursos e o custo do impacto negativo em fluxos de tráfego na rede. Roy et al. [RBD02] utilizou uma abordagem de *routing* multiobjetivos para *wireless multicasting* em tempo real. Yuan [Yua03] incorporou uma técnica de otimização bi-critério para *routing* OSPF(*Open Shortest Path First*), pretendendo minimizar o congestionamento da rede e o impacto de falhas de ligações. No entanto, nenhuma destas técnicas resolveu problemas de *routing* com múltiplos critérios, considerando objetivos simultâneos, como um método específico de Tomada de Decisão com Múltiplos Critérios(*Multiple Criteria Decision Making* - MCDM), denominado *Normalized Weighted Additive Utility Function* (NWAUF) [MGT06].

Tendo então em conta os métodos MCDM, o processo de tomada de decisão envolve escolher uma de entre várias alternativas, o que engloba os seguintes 5 passos:

1. Identificar e avaliar os valores dos critérios;
2. Identificar o conjunto de alternativas;
3. Identificar alternativas eficientes;
4. Escolher um método para ordenar as alternativas;
5. Aplicar o método anterior e escolher a melhor alternativa.

Posto isto, o método NWAUF é baseado na normalização de valores de critérios e na utilização de pesos de importância, os quais variam entre 0 e 1 com uma soma cumulativa de um.

Considere-se um conjunto discreto de alternativas a_j, com j=1,2, ..., n, e a seguinte função de utilidade aditiva ponderada:

$$U(a_j) = \sum_{i=1}^{k} w_i f_{ij} \tag{2.1}$$

Onde, para a alternativa a_j, $w_1, w_2, ..., w_k$ representam a importância dos pesos para cada objetivo, sendo todos eles números positivos e a sua soma obrigatoriamente um número constante, geralmente 1 (por exemplo, $w_1, w_2, ..., w_k = 1$).

Usando a equação (2.1) é então possível determinar a melhor alternativa. Contudo é mais fácil avaliar a importância dos pesos w_i quando todos os valores dos critérios estão normalizados para

uma mesma escala e assim os pesos passam a corresponder exatamente à importância de cada objetivo. Esta abordagem requer alguns passos adicionais para normalizar os objetivos.

Assim, a NWAUF é dada por:

$$U(a_j) = w_1 f_1' + w_2 f_2' + ... + w_k f_k' \tag{2.2}$$

Onde $f_1', f_2', ..., f_k'$ são os valores normalizados de $f_1, f_2, ..., f_k$. Com isto, o valor da função U que no caso normal (2.1) podia ter qualquer valor, passa agora a estar também ele normalizado.

2.2.2 Métricas de Proximidade

A relação entre diferentes padrões, e consequentemente entre diferentes grupos de padrões, é geralmente dada por funções que medem a similaridade ou dissimilaridade entre os mesmos.

Uma função de distância, para um conjunto de dados X, é definida para satisfazer as seguintes condições [XW05]:

1. Simetria: $D(x_i, x_j) = D(x_j, x_i)$;
2. Positividade: $0 \leq D(x_i, x_j) \leq 1$, para todo o x_i, x_j;
3. Desigualdade triangular: $D(x_i, x_j) \leq D(x_i, s_k) + D(x_k, s_j)$, para todo o x_i, x_j, x_k;
4. Reflexividade: $D(x_i, x_j) = 0 \iff x_i = x_j$.

Assim como acontece com uma função de similaridade em relação ao seguinte:

1. Simetria: $S(x_i, x_j) = S(x_j, x_i)$;
2. Positividade: $0 \leq S(x_i, x_j) \leq 1$, para todo o x_i, x_j;
3. $S(x_i, x_j) S(x_j, x_k) \leq [S(x_i, x_j) + S(x_j, x_k)] S(x_i, x_k)$, para todo o x_i, x_j, x_k;
4. $S(x_i, x_j) = 1 \iff x_i = x_j$.

Torna-se então importante analisar as métricas mais usadas, quer para medir a distância quer para medir a proximidade entre dois conjuntos de dados. Deve-se isto não só a que a escolha desta função deve ser cuidadosa, visto depender do tipo de dados a analisar e do algoritmo de *clustering* a usar, mas também porque pode produzir resultados que não sejam os mais indicados para o problema em questão.

Assim sendo, a **correlação de Pearson** é uma importante medida que é muito usada, por exemplo, em algoritmos de recomendação de filtragem colaborativa, como se pode observar na abordagem proposta em [Gon10], e que é dada por $D_{ij} = (1 - r_{ij})/2$, onde $r_{ij} = \frac{\sum_{l=1}^{d}(x_{il} - \bar{x_i})(x_{jl} - \bar{x_j})}{\sqrt{\sum_{l=1}^{d}(x_{il} - \bar{x_i}) \sum_{l=1}^{d}(x_{jl} - \bar{x_j})}}$. O que esta relação faz é medir o grau de correlação linear entre as variáveis de dois objetos, grau esse que pode variar entre -1 e 1, valores estes que representam uma correlação linear negativa perfeita e uma correlação linear positiva perfeita, respetivamente. Quando não existe qualquer correlação entre os objetos ou variáveis a analisar o valor obtido é 0.

Por outro lado temos a **distância Euclidiana** que é uma das métricas mais usadas e que não é mais do que um caso especial da **distância de Minkowski**, dada por $D_{ij} = (\sum_{l=1}^{d} |x_{il} - x_{jl}|^p)^{1/p} = \|x_i - xj\|_p$, em que $p = 2$ e d representa um número de características que representam um qualquer perfil, no caso em questão. Esta técnica funciona muito bem para casos em que os dados são contínuos, em que se tem características compactas, devido à tendência de, nestas métricas, características com um domínio mais alargado se sobreporem às outras. Um solução encontrada para este problema passa pela normalização dos dados de modo a serem representados todos dentro de uma mesma gama de valores. Outra especificação da distância de Minkowski é a **distância de Mahalanobis**, em que se usa $p = 1$ na fórmula original.

Comparando as tabelas 2.1 e 2.2 podemos então observar que este método da distância Euclidiana pode trazer resultados enganadores. Assim, observa-se que em ambos os exemplos a proximidade resultante é igual ($\sqrt{2}$) apesar de que no primeiro caso temos dois perfis sem qualquer semelhança enquanto que no segundo os perfis são quase iguais. Isto acontece porque na distância Euclidiana considera-se a ausência de atributos tão importante como a sua presença e em problemas de grande dimensão a presença de um atributo é geralmente muito mais importante do que a ausência de outro.

Perfil	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
P1	1	0	0	0	0	0	0	0	0	0
P2	0	0	0	0	0	0	0	0	0	1

Tabela 2.1: Par de perfis com dimensão 10 e contendo características binárias - Exemplo 1.

Perfil	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
P3	1	1	1	1	1	1	1	1	1	0
P4	0	1	1	1	1	1	1	1	1	1

Tabela 2.2: Par de perfis com dimensão 10 e contendo características binárias - Exemplo 2.

Para contornar este problema algumas medidas foram propostas, como é essencialmente o caso da **similaridade dos cossenos** e do **coeficiente de Jaccard** [ESK03].

A **similaridade dos cossenos** é então uma métrica geralmente usada, por exemplo, em *clustering* de documentos visto ser uma técnica independente do tamanho do vetor que representa o objeto de dados a analisar. Esta similaridade é dada por $S_{ij} = \cos\alpha = \frac{x_i x_j}{\|x_i\|\|x_j\|}$. Assim vetores que apontem na mesma direção vão ter ângulo, entre eles, cujo cosseno é 1 o que representa total similaridade enquanto que caso o valor seja 0 isso significa que os vetores são perpendiculares e portanto não apresentam qualquer tipo de semelhança.

Quanto ao **coeficiente de Jaccard**, este baseia-se na divisão entre a interseção e a reunião de dois vetores, $J(x_i, x_j) = \frac{|x_i \cap x_j|}{|x_i \cup x_j|}$. No entanto apresenta uma restrição que passa por só poder ser aplicado a problemas com atributos puramente binários sendo necessário para isso normalizar as variáveis, o que pode obviamente ser feito mas é preciso ter atenção devido à perda de informação. Existe também uma extensão desta métrica, o **coeficiente de Tanimoto**, cujo conceito é o

mesmo que o de Jaccard mas permite trabalhar com vetores cujos atributos não são exclusivamente binários. Assim, a similaridade entre dois vetores, dada por este coeficiente, é representada por $T(x_i, x_j) = \frac{x_i x_j}{d_{x_i}^2 + d_{x_j}^2 - x_i x_j}$, onde d_x representa o número de elementos que fazem parte do vetor x.

Visto existirem uma grande quantidade de métricas para calcular a distância ou similaridade entre dois pontos, apenas as medidas acima referidas foram alvo de uma análise mais profunda porque são as mais referenciadas a nível da literatura na área em questão. No entanto outras métricas também são abordadas em alguns artigos como a **correlação de Spearman** [Gon10], **distância de Manhattan** [Ber02] ou a **distância de Hamming** [Ber02].

2.2.3 Algoritmos de Agrupamento

Por forma a agrupar os perfis a serem criados, é necessário, além das métricas para calcular a proximidade entre perfis, ter em conta algoritmos de *clustering* com o objetivo de se formar grupos que exibam duas principais propriedades: baixa similaridade com outros grupos e alta similaridade entre os elementos que contêm [EAS11].

Desta forma, existe uma grande variedade de formas de classificar os algoritmos sendo que a classificação a ter em conta se baseia na apresentada por Elavarasi et al. [EAS11] e portanto teremos os algoritmos divididos nas seguintes categorias:

1. Algoritmos hierárquicos;
2. Algoritmos divisivos;
3. Algoritmos espectrais;
4. Algoritmos baseados na Rede;
5. Algoritmos baseados na Densidade.

Antes de se passar diretamente à análise destas categorias convém referir um aspeto importante que está relacionado com o tipo de *clustering* a efetuar ser *hard* ou *fuzzy*, como é apresentado nos artigos da área. Isto influenciará obviamente o agrupamento de dados visto que um agrupamento *hard* obriga a que um objeto esteja apenas num grupo enquanto que num agrupamento *fuzzy* existe a possibilidade de um objeto ser incluído em mais do que um grupo. No entanto um agrupamento *fuzzy* pode ser convertido em *hard* visto que é atribuído um grau de pertença aos vários grupos, para cada objeto de entrada, e se se deixar o objeto apenas no grupo em que tem maior grau de pertença passa-se então a ter um agrupamento *hard* [JMF99].

2.2.3.1 Algoritmos hierárquicos

O princípio base deste tipo de algoritmos de *clustering* é agrupar objetos de dados de modo a formar uma estrutura em árvore. Há então duas formas opostas de o fazer. Por um lado define-se que cada objeto é um grupo e em cada iteração vão se juntando grupos com base no critério de semelhança usado. A outra abordagem passa por considerar que todos os objetos estão num mesmo

grupo sendo depois divididos também com base no critério escolhido. A estas duas abordagens dá-se o nome de algoritmos aglomerativos e algoritmos divisivos, respetivamente.

Para um grupo com X elementos, algoritmos hierárquicos divisivos têm sempre $2^{X-1}-1$ possibilidades de divisão de um grupo em dois subconjuntos, pelo que esta não é uma abordagem muito usada em termos práticos [XW05].

Os critérios essencialmente usados por algoritmos hierárquicos para calcular a similaridade entre um novo grupo, entretanto criado, e os restantes são de três tipos: *single-linkage*, *average-linkage* e *complete-linkage*, como são denominadas na literatura [Ber02]. Posto isto, na *single-linkage* considera-se a distância entre dois grupos como sendo a menor distância entre dois quaisquer membros, um de cada grupo, considerando-se então que dois grupos estão tão distantes como a menor distância possível entre eles. Relativamente ao critério *complete-linkage* é exatamente o oposto do anterior pelo que a distância entre dois grupos é dada pela maior distância possível entre um qualquer membro de um grupo e outro qualquer membro de outro grupo. Por fim, o critério *average-linkage* assume-se como um ponto intermédio entre os dois anteriores, daí que a distância entre dois grupos seja então a média das distância entre dois quaisquer membros dos grupos em questão.

Estes algoritmo apresentam então algumas vantagens interessantes, que passam pela [Ber02]:

- Flexibilidade incorporada em relação ao nível de granularidade;
- Facilidade de manipulação de qualquer métrica de semelhança ou distância;
- Consequentemente, aplicabilidade a todo o tipo de características.

Enquanto que apresentam as seguintes desvantagens:

- Indefinição quanto ao critério de paragem (frequentemente o número de grupos requerido);
- O facto de a maioria destes algoritmos não revisitar grupos intermédios já contruídos com o objetivo de os melhorar.

Usualmente, estes algortitmos apresentam uma complexidade tanto temporal como espacial de $O(N^2)$ [XW05], pelo que para um número elevado de dados a considerar começa a ser desvantajoso.

2.2.3.2 Algoritmos divisivos

Os algoritmos divisivos essencialmente dividem os objetos de dados a considerar em K partições. Estas são construídas com base num objetivo e representam os grupos. Para se implementar corretamente um algoritmo deste tipo, deve ter-se sempre em consideração dois aspetos:

- Cada grupo deve conter pelo menos um elemento;
- Cada elemento deve pertencer a exatamente um grupo.

Um dos algoritmos mais conhecidos e alvo de inúmeras investigações e combinações com outras técnicas é o *k-means*. Este algoritmo consegue resolver problemas conhecidos de *clustering* e assenta nos seguintes passos:

1. Colocar *K* pontos no espaço representado pelos objetos. Estes pontos, que são referidos como centróides, irão representar o centro de cada grupo que irá conter elementos;
2. Atribuir cada objeto ao grupo que tem o centróide mais próximo;
3. Recalcular as posições dos centróides depois de todos os objetos já estarem alocados;
4. Repetir os passos 2 e 3 até que a posição dos centróides já não se altere.

Analisando os passos anteriores, um aspeto bastante importante deste algoritmo, e que pode ser visto como uma desvantagem, é que os *K* grupos a existirem precisam de ser definidos logo à partida. Isto é efetivamente um aspeto que pode condicionar todo o restante processo de *clustering* pelo que tem de ser definido com bastante cuidado e tendo em consideração o problema em questão.

Além disso, a colocação inicial destes pontos também é preponderante para o desenrolar do algoritmo visto que maus resultados podem ser obtidos devido à sobreposição de pontos de dados, quando um ponto está próximo do centro de outro grupo. Desta forma, a colocação inicial dos centróides deve ser o mais afastada possível.

Um aspeto que algoritmos deste tipo têm sempre em conta é ainda a minimização de uma função objetivo que não é mais do que uma métrica de proximidade para calcular a diferença entre um ponto e o respetivo centóide, a qual se pretende então minimizar o mais possível.

Apesar de ser garantido que o algoritmo *k-means* vai sempre terminar, não é possível ter a certeza de que se vai sempre obter a solução ótima. No entanto é um algoritmo simples e que pode ser conjugado com outras técnicas e adequado ao problema em questão [EAS11] com o uso de algoritmos genéticos, arrefecimento simulado, algoritmos evolucionários ou a otimização das colónias de formigas.

Algumas das derivações do *k-means* mais conhecidas são o *k-medoids*, o *CLARA* (*Clustering Large Applications*) e o *CLARANS* (*Clustering Large Applications based on Randomized Search*) [XW05]. Estes algoritmos foram desenvolvidos essencialmente com o objetivo de reduzir os pontos desvantajosos já referidos, como sendo a escolha inicial do número de grupos, a disposição inicial desses grupos, a capacidade de lidar com ruído, a função de minimização de proximidade, entre outros.

2.2.3.3 Algoritmos espectrais

Algoritmos espectrais não são mais do que um conjunto de técnicas que se baseiam numa matriz de similaridade. Esta é uma matriz *N* X *N*, em que *N* representa o número de objetos a agrupar, em que cada elemento representa a semelhança entre dois objetos medida por uma métrica predefinida.

Assim os grupos vão ser formados pela divisão dos dados, usando essa matriz, e seguindo essencialmente três etapas [EAS11] [VM03]:

1. Pré-processamento: construção da matriz de similaridade;
2. Mapeamento espectral: construção de vetores de eigen para a matriz de similaridade;
3. Pós-processamento: agrupamento dos pontos de dados.

As principais vantagens do agrupamento espectral prendem-se com:

- Não são assumidas grandes definições acerca dos grupos;
- Simples de implementar;
- Objetivo não considera ótimos locais;
- Estatisticamente consistente;
- Funciona mais rápido.

Alguns exemplos de algoritmos deste tipo são o algoritmo *SM*, o algoritmo *KVV* e o algoritmo *NJW* [SMA08].

Por outro lado, o principal ponto fraco apresentado por esta abordagem é ter alta complexidade computacional, pelo que para grandes conjuntos de dados apresenta uma complexidade de $O(N^3)$.

2.2.3.4 Algoritmos baseados na Rede

O mecanismo que está inerente a estes algoritmos consiste em dividir o espaço dos objetos num número finito de células para formar uma estrutura em rede sobre a qual vão sendo executadas as operações de *clustering*. As principais características destas abordagens, que também podem ser vistas como vantagens, são:

- Não calcular distâncias;
- Agrupamento é realizado em pontos de dados resumidos;
- As formas são limitadas à união das células;
- A complexidade do algoritmo é geralmente $O(c)$, em que c é o número de células povoadas, pelo que não depende do número de objetos a agrupar;
- Tempo de processamento baixo.

Um dos exemplos mais conhecidos deste tipo de algoritmos é o STING (*Statistical Information Grid-based method*) [HK06], apesar de serem reportados na literatura outros exemplos como o *WaveCluster* [SCZ98].

2.2.3.5 Algoritmos baseados na Densidade

Algoritmos baseados na densidade fazem com que os grupos de dados continuem a crescer enquanto a densidade na vizinhança não exceda determinado limiar. Uma das grandes vantagens destes algoritmos passa pela capacidade em eliminar o ruído no conjunto de dados. Alguns dos exemplos mais conhecidos são o *DBSCAN* e o *DENCLUE* [HK06] sendo que estas abordagens apresentam então as seguintes características:

- Manipulação de grupos de dados de forma arbitrária;
- Capacidade de manipular o ruído;
- Necessidade de apenas uma análise ao conjunto de dados de entrada;
- Necessidade de parâmetros de densidade para ser inicializado.

A complexidade apresentada também é um aspeto bastante positivo destes algoritmos visto que, por exemplo, tanto o *DBSCAN* como o *DENCLUE* apresentam uma complexidade temporal de $O(Nlog(N))$, em que N representa o número de objetos a considerar [XW05].

2.3 Sumário

Relativamente a este projeto existiam alguns pontos fulcrais alvo de desenvolvimento e implementação como o uso de algoritmos de *routing* para ambientes dinâmicos, o uso de perfis de utilizador, com base nas estatísticas de utilização dos mesmos, conjugado com técnicas inteligentes de criação, comparação e agrupamento dos mesmos e a inclusão de múltiplos critérios nos referidos algoritmos.

Tendo isto em conta, começou-se por investigar os algoritmos de escolhas de rotas aplicados atualmente em problemas de SP. Relativamente aos tradicionais algoritmos de *Dijkstra* e A* foi notada a sua grande aplicação em alguns dos problemas mais conhecidas de SP apesar da desvantagem que apresentam essencialmente ao nível da sua complexidade quando implementados em ambientes dinâmicos. Quanto ao LPA*, outro dos algoritmos também abordados, apresenta-se como uma melhoria bastante interessante ao A*, que por si só já é uma melhoria do algoritmo de *Dijkstra* com a inclusão do conceito de heurística, mas peca essencialmente pela sua grande dependência ao ponto de origem, o que leva desde logo a ser descartado para ambientes dinâmicos, e também pelo facto de que o melhoramento proposto, apesar de colmatar esta desvantagem, pode não ser possível aplicar em qualquer tipo de grafos.

Assim sendo, o uso de sistemas multiagente representativos assume-se cada vez mais como uma das abordagens mais promissoras nesta área essencialmente quando conjugados com uma meta-heurística baseada no comportamento das colónias de formigas na procura de comida e também no fenómeno de evaporação já abordado. No entanto, um desvantagem da abordagem deste tipo mais recentemente proposta foi não considerar ainda múltiplos critérios a usar no processo de escolha do SP.

Considerando esta fraqueza, e tendo a noção de inovação e de contribuição científica sempre presente, uma perspetiva desta dissertação poderia passar pela inclusão do conceito de "perfil de utilizador" integrado com os agentes. Assim sendo, e considerando a existência dos 3 tipos de agentes já abordados (agente veículo exploratório (AVE), agente veículo informativo (AVI) e agente infraestrutura (AIE)), o objetivo passaria por ter um agente veículo que contenha um perfil baseado no histórico de utilizações que faz do sistema e das rotas que geralmente usa. Posto isto, um AVE ao analisar as várias possibilidades de rotas considera não só a perspetiva de ocupação das vias por parte de outros veículos, e do tempo que irá despender em cada ponto, mas também dos perfis que entretanto os AIE vão adquirindo.

Esta construção do perfil do AIE seria feita com base na informação que os AVI passam quando um condutor decidir a rota pela qual pretende seguir. Assim, o AVI informa os AIE não só da intenção de ocupação daquela rota por parte do veículo que representa mas também do perfil associado ao mesmo. Isto permitia que o AIE fosse "autocriando"o seu perfil com base no "tipo"de veículos/utilizadores que por lá passavam, considerando cada novo perfil como uma nova utilização que ele próprio fez do sistema e usando então modelos estatísticos para conjugar o seu perfil atual com o novo recebido.

A par da abordagem anterior, surge ainda a implementação de um algoritmo baseado em hierarquias, que se apresentou também como algo capaz de reduzir em grande escala a complexidade apresentada por um qualquer algoritmo a aplicar após um fase de pré-processamento.

Desta forma surge a possibilidade de ter uma computação do grafo a considerar, em determinada situação, numa fase anterior à aplicação do algoritmo tanto ao nível dos nós como das arestas ou até de ambos. Qualquer destas abordagens terá sempre grande contributo científico ainda mais se tivermos em consideração a sua conjugação com os perfis de utilizador.

Considerando estas perspetivas enunciadas e a importância que os perfis de utilizador assumem nesta dissertação como outro critério a influenciar a seleção do SP, foi feita uma análise essencialmente das métricas de proximidade e dos algoritmos de *clustering* mais usados, e consecutivamente com melhores desempenhos, em problemas deste género.

Dessa análise podem retirar-se algumas conclusões interessantes, desde logo que é difícil e arriscado definir com grande certeza qual a melhor métrica de proximidade, visto esta depender bastante das características que definem um perfil. No entanto, a similaridade dos cossenos e o coeficiente de Tanimoto são duas métricas bastante interessantes, de acordo com o problema em questão, visto a primeira poder ser usada independentemente do tamanho do vetor de características, o que permite ter um maior número de critérios a caracterizar um perfil, e a segunda ser uma extensão ao coeficiente de Jaccard, o qual já revela resultados bastante satisfatórios, mas com aplicação em variáveis não exclusivamente binárias o que é preponderante no âmbito desta dissertação.

Ainda relativamente aos algoritmos a usar para agrupar perfis, aquele que foi alvo de maior análise foi o *k-means*. Isto porque este é efetivamente um dos algoritmos mais conhecidos e usados em problemas típicos de *clustering* e apesar de implicar que se defina à partida os K grupos a existirem, o que pode também não ser uma abordagem vantajosa, há sempre a possibilidade de o

combinar com outras técnicas inteligentes. Assim seria possível colmatar a desvantagem referida e adaptá-lo ao problema em questão, como já foi feito e referido anteriormente com algoritmos genéticos ou a otimização das colónias de formigas. Outros algoritmos que se concluiu serem bastante interessantes foram os algoritmos hierárquicos que permitem ter vários grupos a influenciar a escolha do SP, uns com mais influência que outros. Além disso, um agrupamento *fuzzy* é também interessante visto que o facto de ter um perfil inserido em vários grupos é uma abordagem que pode também ter um grau de pertença associado e assim influenciar de várias maneiras a escolha do caminho a seguir.

Posto isto, uma grande contribuição deste projeto é a integração de múltiplos critérios no processo de escolha de rotas de forma a personalizar esta escolha e a adequá-la aos vários tipos de utilizadores considerados. Isto foi conseguido quer através da aplicação de melhorias a algoritmos já existentes quer pelo desenvolvimento de novas abordagens através do uso de heurísticas e outras técnicas inteligentes de otimização. Outro aspeto com grande contributo é a comparação entre as várias abordagens, a qual certamente dá uma ideia mais concreta de quais os pontos fortes e aqueles em que se pode melhorar cada abordagem e também as situações a que mais se adequa cada uma delas.

Capítulo 3

Modelação do Problema

Neste capítulo vai-se então abordar mais detalhadamente o problema para o qual se pretende encontrar uma solução. Desta forma será definida cada parte que constitui o problema em si de modo a poder-se delinear todas as etapas pelas quais se tem de passar, e todos os problemas a ultrapassar, para concluir a dissertação com sucesso e correspondendo aos objetivos propostos.

Com isto pretende-se também clarificar a interligação entre as diferentes fases do projeto e a contribuição que uma vai ter para com as outras.

Assim, e apesar de o problema já apresentado estar essencialmente relacionado com algoritmos de *routing* e perfis de utilizador, muito mais para além disso tem de se ter em consideração quando se fala de um problema deste tipo. Desta forma é preciso dar também grande atenção ao tratamento dos mapas a considerar e à construção de um ambiente de testes.

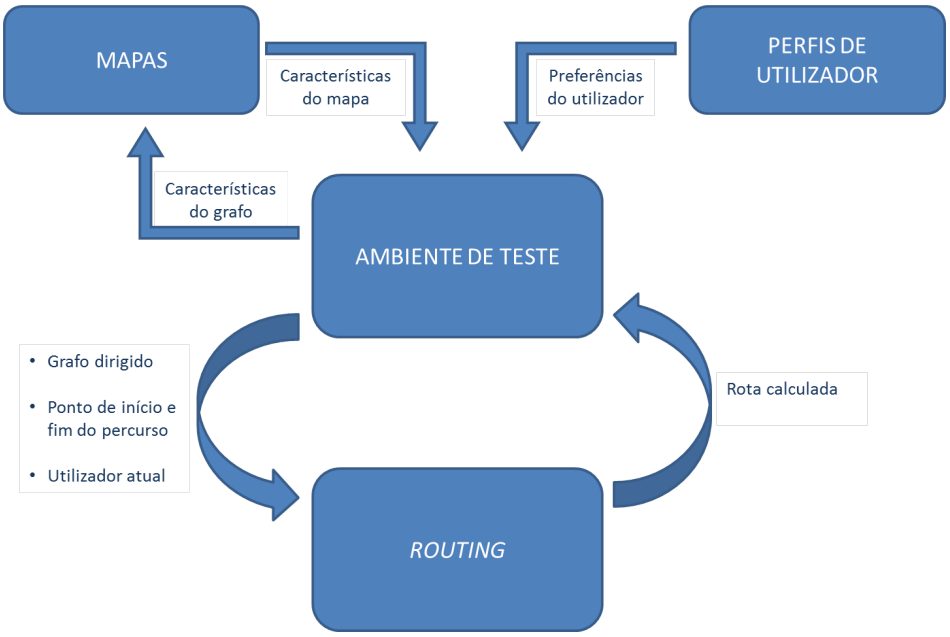


Figura 3.1: Modelação do problema nas suas partes constituintes.

É precisamente isto que é necessário considerar e que está representado na Figura 3.1. Nessa figura são claramente percetíveis quatro módulos: mapas, perfis de utilizador, ambiente de testes e *routing*. No entanto, os dois primeiros acabam por ser integrados e funcionarem como sub-módulos do ambiente de testes. Além disso, é possível através de cada seta verificar a interligação que existe entre cada parte e o fluxo de informação que acontece entre cada módulo.

Tudo isto apresentado nesta figura é detalhado nas secções seguintes, de forma a levar a uma melhor compreensão de cada sub-problema em que o problema já definido foi dividido e da forma como foram abordados.

3.1 Desenvolvimento de um modelo de dados para mapas

Uma parte importante do projeto a desenvolver passa pelos mapas que irão ser usados de forma a ser possível testar as novas abordagens .

Como todo e qualquer mapa, aqueles que serão usados devem conter informações geográficas de pontos, representando interseções, e das ligações entre os mesmos. Além desta informação normal, é necessário, para o problema em questão, que seja possível consultar no mapa todo um conjunto de características mais alargado. Isto deve-se essencialmente ao facto de se pretender um *routing* multicritério e personalizado, o que fará com que todas as características que podem fazer parte de um perfil de utilizador tenham de ser consideradas no mapa.

Assim sendo, quer estando associado às interseções ou então às estradas que as ligam, tem de ser sempre possível num processo de pesquisa optar por uma alternativa, em detrimento de outra(s), o que só é conseguido caso as características do perfil do utilizador atual estejam contidas no mapa e na zona em análise. Caso isto não aconteça passa-se a ter um *routing* que deixa de ser multicritério pois mesmo que sejam consideradas as características dos perfis estas não vão influenciar em nada por não estarem definidas nos mapas.

Tendo isto em conta, é necessário desenvolver um modelo de dados de modo a manter todas as informações dos mapas a utilizar no ambiente de testes. Como se está a falar de um problema que lida com redes de estradas reais e portanto com localizações geográficas, a base de dados a desenvolver também tem de ser georreferenciada.

Este modelo de dados tem de ser minucioso para que não ocorram falhas tanto a importar mapas de uma base de dados para um ambiente de testes como no processo inverso que terá de ser possível de modo a conseguir-se testar os novos algoritmos desenvolvidos e a ter um ambiente flexível e que permita adequar os mapas às necessidades momentâneas.

3.2 Definição de perfis de utilizador

Os perfis de um utilizador são uma das partes mais importantes deste projeto de dissertação. Isto acontece porque são eles os principais responsáveis pelas alterações, pelas melhorias, pelas otimizações aos algoritmos já existentes ou até pela implementação de novas abordagens.

Isto acontece essencialmente pelos seguintes motivos, que se pretendem que sejam preponderantes na inovação associada a esta tese:

1. *Routing* multi-critério;
2. Escolha de rotas personalizadas.

Relativamente ao ponto 1, o perfil de utilizador é fundamental visto que os múltiplos critérios a considerar são exatamente todas as características que um utilizador pode preferir.

Desta forma, diferentes perfis têm de fazer com que determinadas características das vias sejam favorecidas relativamente a outras. É inclusive possível ter um perfil que permita escolher uma rota através de um algoritmo puramente tradicional, bastando para isso que não haja preferência daquele tipo de utilizador por nenhuma das características possíveis.

No que concerne ao aspeto número 2, é facilmente percetível a importância que um perfil terá na personalização de uma rota visto que possibilitará que sejam sugeridas rotas diferentes a utilizadores com diferentes perfis, ainda que os pontos de partida e chegada sejam os mesmos.

Desta forma, para definir um perfil é necessário considerar as seguintes etapas:

1. Definição das características a considerar;
2. Identificação das preferências para cada característica;
3. Normalização de todas as preferências.

Desta forma, esta é uma fase bastante importante no desenvolvimento desta dissertação essencialmente no que diz respeito à definição das características que permitem classificar um perfil e que, como já referido, podem por conseguinte influenciar a eficiência das abordagens de escolha de caminho mais curto.

3.3 Construção de um ambiente de testes

Para que seja possível ter perceção e avaliar as abordagens desenvolvidas é necessário ter um ambiente de testes. Com isto pretende-se simular a escolha de rotas por parte de um utilizador de um sistema de navegação GPS.

Desta forma, o pretendido é que o utilizador esteja perante um ambiente em que possa:

1. Ter uma visualização que lhe permite perceber a zona do mapa em que se encontra;
2. Navegar no mapa e posicionar-se facilmente na zona pretendida;
3. Escolher os pontos de origem e destino do seu percurso;
4. Calcular a melhor rota que se lhe adequa;
5. Visualizar a rota que deve seguir.

Para que isto seja possível pretende-se a utilização e adequação de um emulador de escolha de rotas ao problema em questão.

Desta forma, uma das preocupações passa por interligar o emulador com a base de dados que serve de armazenamento de mapas. Na pior das hipóteses isto pode levar ao desenvolvimento por completo de um módulo de comunicação entre o emulador e uma base de dados ou então, no caso de esta já existir, adequar apenas o modelo de dados, desenvolvido para os mapas, para que a replicação destes seja feita na íntegra e sem qualquer tipo de ruído.

O outro aspeto a ter conta está relacionado com a consideração de perfis de utilizador. Neste caso torna-se necessário desenvolver e adicionar ao emulador as funcionalidades necessárias para permitir a criação de perfis e a associação destes a utilizadores para poderem ser usados nas novas abordagens desenvolvidas. Esta é previsivelmente a fase mais custosa do desenvolvimento do ambiente de teste, denominado de *Test-Bed*, no caso de ser possível a utilização do emulador.

Caso não venha a ser possível o desenvolvimento do *Test-Bed* com base num emulador *open-source*, a alternativa passará por criar um emulador próprio assente sobre um editor de redes.

Como é expectável, neste caso a criação de uma rede de estradas, através da importação de mapas modelados numa base de dados, e a navegação na mesma apresenta-se como sendo algo que não levará a uma implementação tão rebuscada como no caso anterior visto que previsivelmente o editor já se encontra habilitado a importar mapas de várias fontes.

No entanto, todo o processo de escolha de rotas, e que diz respeito aos pontos 3, 4, e 5 na enumeração anterior, teria de ser implementado visto raramente ser algo contabilizado. Assim sendo, passa a ser necessário readaptar a interface com o utilizador de modo a permitir que este escolha os pontos de origem e fim que pretende para o seu percurso.

Além disto, a necessidade de integrar uma camada de processamento de grafos e escolha de melhor caminho é algo bastante claro porque, ao contrário do caso anterior, isso ainda não existe e é necessário para futuramente adicionar novas abordagens e servir inclusive de termo de comparação. Ainda referente a este aspeto é de extrema importância que a rota calculada seja apresentada ao utilizador de forma a que este tenha um perfeita perceção da mesma e lhe permita identificar claramente por onde deve seguir para chegar ao seu destino.

Algo que neste caso também acontece, à semelhança do que foi apresentado caso a escolha incida sobre um emulador, diz respeito à implementação dos perfis de utilizador.

Tendo em consideração o que foi referido anteriormente pode dizer-se que o ambiente de teste a desenvolver deve ser a integração de vários módulos como é apresentado na Figura 3.2.

Na figura referida anteriormente pode então observar-se que o **módulo Visual** apresenta uma relação bidirecional com a base de dados visto ser pretendida não só a importação como a exportação de mapas. Além disso, outra comunicação deste género verifica-se com o **módulo de *Routing***, visto que quando é solicitada pelo utilizador a escolha de uma rota, o primeiro módulo deve informar o segundo da topologia do grafo que lhe está inerente e dos pontos extremos do caminho a obter. A relação é bidirecional visto que, após o cálculo do caminho mais curto, a informação gerada deve ser retornada para ser apresentada na interface com o utilizador. Este último módulo

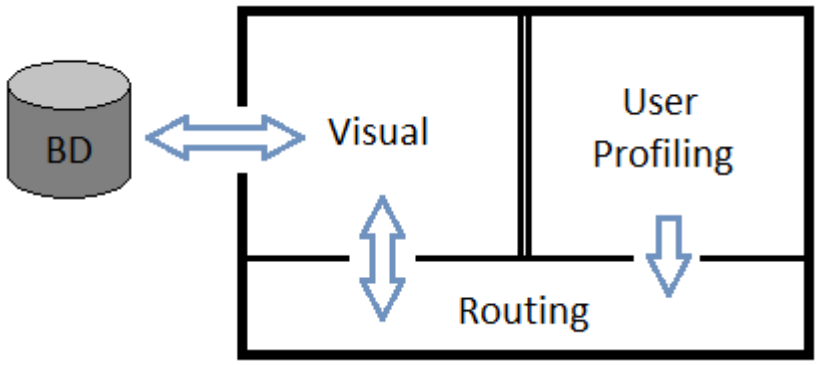


Figura 3.2: Ambiente de testes a desenvolver.

tem também naturalmente uma ligação com o **módulo de *User Profiling*** de modo a considerar o perfil do utilizador atual no processo de escolha de rotas.

3.4 Desenvolvimento e implementação de novas abordagens de *routing*

Apesar de todas as outras etapas já enunciadas e detalhadas, e que são fundamentais para o desenvolvimento do projeto, a fase de delineamento e implementação de novas abordagens de *routing* é efetivamente o ponto central desta dissertação.

Com isto pretende-se desenvolver o maior número de abordagens diferenciadas possível com o principal objetivo de ter uma vasto leque de soluções para comparar com o algoritmo base, implementado no ambiente de testes. Além disto, a comparação entre as várias implementações é também um aspeto a ter em conta relativamente à eficiência dos algoritmos desenvolvidos.

Para que se consigam então obter algoritmos eficientes é necessário ter em consideração os seguintes parâmetros:

- Número de nós analisados;
- Número de arestas analisadas;
- Espaço em memória ocupado durante o processamento;
- Tempo de processamento;
- Distância total do percurso;
- Tempo total do percurso.

Assim, pretende-se retirar conclusões relativamente aos aspetos mencionados acima e com isso determinar até que ponto uma abordagem se torna compensatória, ou não, relativamente às demais. No entanto, é importante descriminar até que ponto uma abordagem pode ser valorizada ou não relativamente a outra. Assim, dos pontos anteriores os dois últimos são os que apresentam

menor importância visto que com um algoritmo tradicional de escolha de caminho mais curto, como o caso do *Dijkstra* ou A*, consegue-se obter os melhores valores possíveis para essas duas características. Quanto ao restantes, um algoritmo será tanto melhor quanto mais aspetos conseguir otimizar.

Além disto, é necessário ponderar as conclusões retiradas da eficiência com a satisfação das necessidades do utilizador. Isto leva a que, como os algoritmos utilizam o perfil do utilizador para gerar rotas o mais personalizadas possível, não se pretenda apenas desenvolver estratégias que reduzam muito o tempo de processamento se, em termos da rota gerada, continuar a ser algo genérico ou que não se adequa ao utilizador em questão.

Isto leva a que sejam consideradas sempre estas duas vertentes, como ilustrado na Figura 3.3.

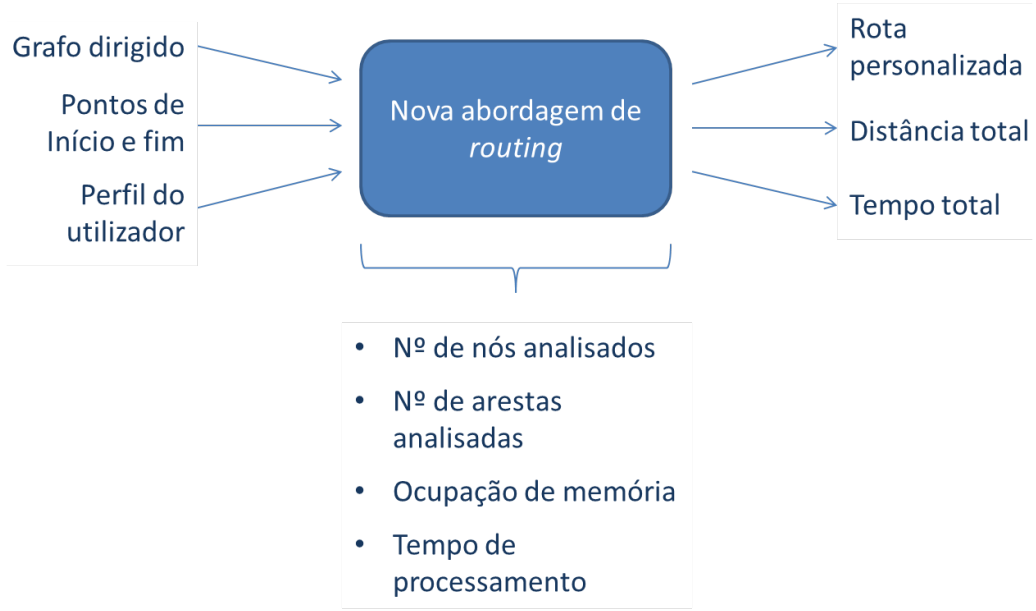


Figura 3.3: Dados de entrada e saída e otimizações pretendidas com cada abordagem de *routing* desenvolvida.

Na imagem anterior pode então ver-se três secções bem distintas: dados de entrada, dados de saída e objetivos em processamento.

Relativamente aos dados de entrada, ilustrados do lado esquerdo, há que realçar que, para além do grafo que representa o mapa e dos pontos de início e fim do percurso pretendido, se pretende usar o perfil de utilizador nas abordagens a desenvolver. Este *input* é extremamente necessário por forma a resolver o problema do *routing* multicritério, *routing* este que deverá levar também a uma rota personalizada.

Como é possível observar no lado direito da imagem, além da obtenção de rotas que se pretendem ser o mais adequadas possível ao utilizador atual, por forma a corresponder às necessidades e exigências cada vez mais crescentes dos utilizadores de sistemas inteligentes de transportes, é também apresentada a distância e o tempo total do percurso sugerido pela respetiva abordagem. Isto acontece porque é efetivamente uma das formas de se conseguir analisar e comparar os algoritmos.

Apesar disto, os aspetos que se deve ter em consideração para se avaliar a eficiência do algoritmo são ilustrados na zona central. De referir que todos eles se referem a questões durante o processamento da melhor rota a seguir.

Durante a fase de processamento, e por forma a dar resposta aos objetivos que se pretendem atingir, em termos de eficiência, surge outro problema aliado aos algoritmos. Este problema prende-se com as possibilidades que existem de melhorar algoritmos já existentes, e fortemente utilizados, e a adequação de cada uma dessas soluções ao algoritmo base em questão ou à especificidade do problema. Assim é preciso fazer esta análise de modo a solucionar qual a técnica que melhor se adequa e melhor reage com cada algoritmo podendo falar-se de otimizações de funções de custo, de implementação de novas heurísticas ou até do desenvolvimento de novas abordagens por completo de maneira a conseguir-se fazer diferentes técnicas convergirem para um objetivo comum.

No entanto, apesar de uma fase de processamento extremamente otimizada permitir obter soluções eficientes, pode ser necessário recorrer a outras técnicas que estarão indiretamente relacionadas. Isto diz respeito essencialmente a considerar uma fase de pré-processamento em que técnicas inteligentes seriam usadas por forma a melhorar a fase de processamento, mesmo que usando um algoritmo tradicional.

O pré-processamento de um mapa é algo que pode ser bastante importante visto que muitas das vezes os mapas estão habilitados para serem usados noutros ambientes diferentes e podem conter informações desnecessárias. Por estas e outras razões, tanto filtrar como fazer uma análise prévia do mapa pode solucionar o problema de otimizar o processo de escolha de rotas e conseguir-se assim que esse cálculo seja muito mais rápido e menos custoso.

3.5 Conclusões

A modelação do problema em questão, apresentada na Figura 3.1 e detalhada neste capítulo, permitiu tirar algumas conclusões interessantes relativamente a cada um dos pontos aos quais se pretende dar uma resposta com esta dissertação, às fases que terão de ser ultrapassadas e respetivas tarefas que terão de ser desenvolvidas e à forma como abordar o problema através da metodologia a seguir.

Posto isto, esta dissertação pretende então tirar conclusões e dar respostas relativamente aos seguintes problemas:

- Utilização de algoritmos de *routing* multicritério em sistemas de navegação GPS;
- Eficiência dos algoritmos ao nível da sua complexidade espacial e temporal;
- Personalização das rotas de modo a ir de encontro às necessidades dos utilizadores.

Para dar resposta a estes problemas torna-se necessário o desenvolvimento de quatro fases principais que estão relacionadas com:

- Mapas;
- Perfis de utilizador;
- Ambiente de testes;
- Algoritmos de *routing*.

Estas fases não são todas diretamente necessárias para dar resposta aos problemas em questão. No entanto, tornam-se fundamentais para que seja desenvolvido algo consistente e que permita simular uma situação o mais real possível.

Além disto, foi importante separar em módulos tudo o que faz parte deste projeto e definir mais claramente em que é que cada fase contribui para as restantes e por conseguinte para a prossecução dos objetivos desta dissertação.

Desta forma, as duas fases principais deste projeto serão as que estão relacionadas com os perfis de utilizador e os algoritmos de *routing*.

No entanto, para que a escolha das rotas seja feita como pretendido é necessário que se tenha acesso ao grafo dirigido que representa o mapa onde se pretende navegar, ao ponto de início e de fim do percurso pretendido e às preferências do utilizador atual. Estas preferências não são mais do que o perfil que irá ser considerado aquando do processamento do mapa.

Estes dados de entrada, que são necessários nesta fase, são provenientes da fase de desenvolvimento do ambiente de testes. Isto acontece porque o ambiente de testes pode ser definido como sendo a integração das implementações do módulo relativo aos mapas e do módulo que concerne aos perfis de utilizador. Assim, para se criar o *test-bed* é preciso obter informações sobre as características dos mapas, e sobre as preferências do utilizador atual, que posteriormente são disponibilizadas através de uma interface mais apelativa e que permite ao utilizador escolher quais os pontos de partida e destino do percurso que pretende fazer.

Com a modelação dos dados dos mapas definida, há um fluxo de informação no sentido inverso ao definido anteriormente permitindo que as características dos grafos criados no ambiente de testes sejam armazenadas na base de dados para futura utilização.

Além disto, e como é possível observar na Figura 3.1, a rota calculada será apresentada em termos visuais no ambiente de testes após ser calculada na fase relativa ao *routing*.

Tendo isto em conta, pode-se concluir que esta metodologia leva à resolução do problema definido sendo que a fase de definição e implementação dos algoritmos de *routing* só acontecerá após estar criado o ambiente de testes, com a integração da interface visual com os mapas e os perfis de utilizador, além da implementação de um algoritmo tradicional com aplicação em problemas de caminho mais curto.

Assim sendo, o início da implementação do projeto passa pelo desenvolvimento do modelo de dados e pela criação dos perfis de utilizador. Posteriormente passa-se para a integração disto com um emulador de *routing* ou um editor de redes de modo a criar o *test-bed* e na fase final são então implementadas todas as abordagens que serão alvo de análise e avaliação, no sentido de solucionar o problema detetado.

Capítulo 4

Implementação

No presente capítulo são abordados todos os processos do decorrer do desenvolvimento do projeto inerente a esta dissertação.

Assim, tendo em conta a metodologia e as perspetivas de solução já abordadas no Capítulo 3, pretende-se dar uma visão mais detalhada das decisões tomadas e do que foi realmente implementado, alterado ou descartado relativamente ao que já havia sido representado.

Além deste aspeto são abordadas as tecnologias usadas para cada módulo do projeto desenvolvido, sempre que isto se justificar.

Desta forma, irá abordar-se inicialmente o desenvolvimento do ambiente de testes, denominado de *Test-Bed*, e todos os seus constituintes e posteriormente as abordagens de *routing* efetivamente implementadas.

4.1 Test-Bed

O ambiente de testes desenvolvido pode ser definido como uma integração de três partes. A primeira delas prende-se com os mapas a serem usados por forma a processar-se as rotas pretendidas e a testar as abordagens desenvolvidas. A segunda passou pela definição e implementação das funcionalidades relativas aos perfis de utilizador e a última pela utilização de um editor de redes para integrar o que havia sido desenvolvido nas duas fases anteriores. Nesta fase foi ainda adicionada uma camada de cálculo de rotas e implementada uma abordagem tradicional (algoritmo de *Dijkstra*) por forma a servir de termo de comparação com as restantes abordagens desenvolvidas no âmbito desta dissertação e que estão definidas na Secção 4.2.

4.1.1 Modelo de dados

O modelo de dados desenvolvido teve por base aquilo que já era utilizado pela Tele Atlas [Atl05] e está representado na Figura 4.1.

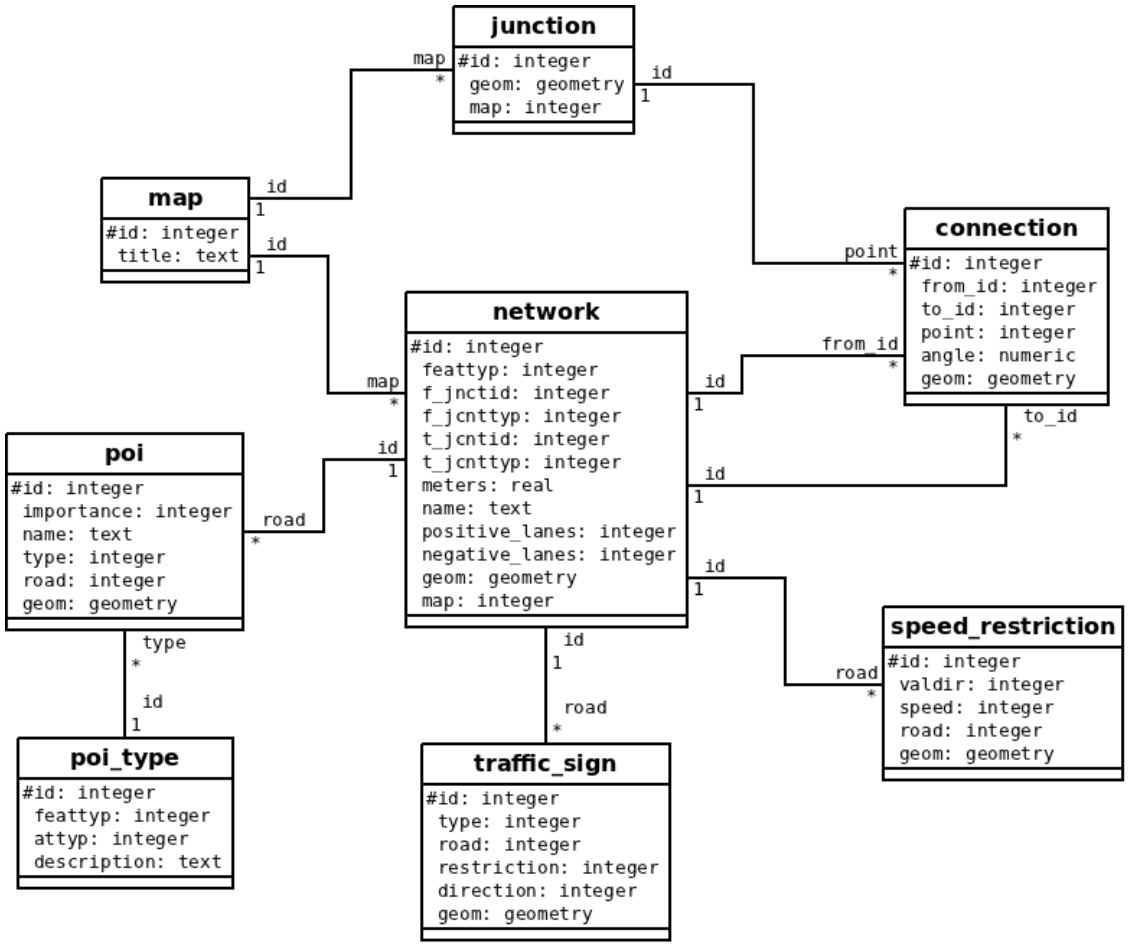


Figura 4.1: Modelo de dados.

De entre o modelo apresentado, as tabelas ***junction*** e ***network*** são aquelas que se pode dizer serem as que se apresentam como fundamentais em qualquer modelo de dados relativo a mapas. Isto porque a primeira refere-se aos pontos assinalados no mapa, que no problema em questão vão ser os elos de ligação entre as mais diversas estradas, enquanto que a segunda representa segmentos que ligam dois pontos, que vai servir para representar as estradas no mapa.

A tabela ***map*** é aquela que de entre as restantes está mais fora do âmbito do projeto e que foi considerada por se tornar fundamental para a importação e exportação de mapas através do editor de redes.

Desta forma, todas as restantes tabelas foram definidas como necessárias e fundamentais para ser possível manter e usar informações dos mapas que serão úteis e fulcrais para serem usadas nos algoritmos desenvolvidos.

Assim temos as seguintes tabelas:

- ***poi***: representa um ponto de interesse, e toda a informação que lhe está associada, e que tem uma chave estrangeira que permite identificar, através da tabela network, qual o segmento no qual está colocado. Cada ponto de interesse estará ainda associado a um tipo, de um conjunto de possibilidades mantidas na tabela ***poi_type***. Esta tabela vai ser importante

para permitir adequar as rotas as utilizadores, essencialmente quando estes apresentarem características de turista.

- ***speed_restriction***: por forma a ser possível considerar o tempo como medida de custo nas funções associadas ao algoritmos de *routing*, esta tabela foi preponderante para permitir associar a cada segmento o(s) limite(s) de velocidade respetivo(s).

- ***connection***: tabela preponderante para se ter uma rede de estradas viável e na qual seja possível construir-se rotas pois define todas as ligações que existem num mapa. Essas ligação acontecem sempre num ponto definido na tabela junction e permitem conectar dois segmentos (tabela network) que incidem sobre esse ponto. Além deste aspeto, é mantida a informação relativa ao ângulo correspondente a cada conexão que será um aspeto importante no momento de escolha de rotas e de privilegiar as características de um utilizador ecológico.

- ***traffic_sign***: apesar de esta tabela permitir manter informação sobre uma diversidade de sinais de trânsito, o objetivo com a criação da mesma passou por saber a localização apenas de sinais luminosos que são um fator que no momento de definir um percurso podem ser determinantes para utilizadores que pretendam escolher a rota mais rápida. Além disto, cada sinal estará sempre associado a um segmento no qual vai acabar, indiretamente, por também delimitar a velocidade permitida.

Com isto, no momento de proceder ao processamento dos algoritmos de *routing* é possível obter informações dos limites de velocidade e das localizações tanto de sinais luminosos como de pontos de interesse, no que diz respeito às estradas. Ao nível das junções, é possível saber as estradas que se podem seguir, tendo em conta aquela que foi usada para chegar ao ponto em questão, e também o grau de viragem correspondente a cada manobra.

Como é possível observar na Figura 4.1, praticamente todas as tabelas apresentam o campo *geom* que é um atributo espacial.

O modelo de dados ilustrado na figura anterior foi delineado e posteriormente criada uma base de dados objeto-relacional usando *PostgreSQL* com *PostGIS*. A escolha foi esta devido à necessidade de ter uma base de dados georreferenciada o que foi possível com o uso do *PostGIS* que é uma extensão espacial ao SGBD *PostgreSQL* e que permite a este suportar objetos geográficos e portanto ser usado como uma base de dados espacial para sistemas de informação geográfica (*GIS*), como é referido em [Fre09].

Por forma a facilitar o desenvolvimento dos aspetos mencionados foi usado o *pgAdmin III* que é uma ferramenta de administração para *PostgreSQL*. Foi também usada, embora em menor escala, a framework *uDig* para permitir a edição e visualização de mapas armazenados em *shapefiles* e servir de elo de ligação entre estes e a base dados.

4.1.2 Perfis de Utilizador

Um dos aspetos mais importantes desta dissertação passa pela personalização das rotas. Como já referido isto consegue-se tendo em conta as preferências que cada utilizador tem quando usa um sistema de navegação GPS.

Desta forma foi definido que cada utilizador teria associado um perfil que iria conter determinadas características às quais corresponderiam as respetivas preferências por parte do utilizador em questão.

Assim, um perfil é representado como um vetor de m posições, em que m representa o número de características possíveis de estarem contidas num perfil. Cada uma dessas posições contém um valor, compreendido entre 0 e 1, que representa a preferência dada pelo utilizador, a que corresponde aquele perfil, àquele critério, como se pode ver na Figura 4.2.

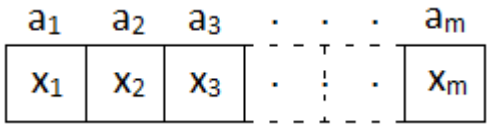


Figura 4.2: Perfil de utilizador genérico.

Nesta figura pode-se observar que cada posição corresponde ao atributo i, em que $1 \leq i \leq m$, e a preferência do utilizador atual apresenta um valor x_i.

Caso a preferência do utilizar seja extrema relativamente a uma característica e esta seja de privilegiar relativamente às restantes o valor de x será 1. Caso seja um atributo que não revela qualquer importância, x assumirá o valor 0. Todos os outros valores intermédios podem ser atribuídos além de que podem naturalmente existir características que tenham a mesma preferência por parte de um mesmo utilizador.

Foram então definidos desta forma os perfis dos utilizadores a considerar no âmbito do problema apresentado como propulsor desta tese. Assim, e tendo em conta a auscultação da NDrive que apresentou e orientou para esta problemática, foram definidos que os tipos de utilizadores mais interessantes a usar neste projeto e aos quais a personalização de rotas pode ser melhor aplicável e mais útil seriam condutores com interesse turístico e/ou ecológico.

Tendo isto em consideração, e também a grande quantidade de dados, que não se possui, e que seria necessária para que o histórico de utilizadores fosse algo conclusivo, foi decidido que seriam criados perfis estáticos que permitissem tirar conclusões e que englobassem características dentro das que correspondiam ao tipo de condutores sugerido.

Assim sendo, cada perfil de utilizador é formado por um vetor de quatro posições que dizem respeito a:

1. Declive;
2. Manobras;
3. Turismo Cultural;

4. Turismo Ocasional.

De entre estas características, as duas primeiras são as que enquadram mais no aspeto ecológico enquanto as seguintes, como o próprio nome indica, referem-se mais ao foro turístico.

Declive

Característica relacionada com a inclinação das estradas e a existência de oscilações nas mesmas.

Desta forma, um utilizador terá um preferência mais próxima de 1 quanto maior for o seu interesse em seguir estradas planas e pouco acentuadas.

Manobras

Este critério diz respeito ao grau das curvas que se terão de efetuar em cada ponto de junção de estradas.

Desta forma, um utilizador terá um preferência mais próxima de 1 quanto maior for o seu interesse em seguir estradas que o irão levar a curvas suaves na ligação com as seguintes.

Uma curva é considerada tanto mais suave quanto menor for o grau que lhe corresponde. Assim sendo, considerou-se a existências de três tipos de curvas:

- Suave - grau de curvatura inferior a 45^{o};
- Normal - grau de curvatura entre 45^{o} e 90^{o};
- Acentuada - grau de curvatura superior a 90^{o}.

Turismo Cultural

Um utilizador que considerar este aspeto está efetivamente a dar importância à existência de pontos de interesse do tipo *cultural*, e à quantidade em que ocorrem na rota que pretende seguir.

Pontos de interesse deste tipo são aqueles que têm uma categoria entre as seguintes:

- *Lodging* (Ex: Hotel, hostel, pensão etc.);
- *Airline Access* (Ex: Aeroporto, zona de aviação, etc.);
- *Railway Station* (Ex: Estação de comboio, metro, autocarro, etc.);
- *Petrol Station* (Postos de abastecimento de veículos com/sem GPL);
- *Ferry Terminal* (Terminais para travessias de barco ou comboio);
- *Parking Garage* (Parques de estacionamento para automóveis pagáveis ou não);
- *Worship* (Ex: Igreja, Sé, etc.);
- *Monument* (Ex: Museu, Ponte, Paço, Estádio, Palácio, Castelo, etc.).

Desta forma, quanto mais interessado estiver o utilizador em passar pelo maior número possível de pontos das categorias já enunciadas mais próxima a sua preferência vai estar do valor 1.

Turismo Ocasional

Esta característica pode ser descrita exatamente como a anterior com a alteração das categorias dos pontos de interesse preferidos.

Assim, além das categorias comuns, que são:

- *Lodging*;
- *Airline Access*;
- *Railway Station*;
- *Petrol Station*;
- *Ferry Terminal*;
- *Parking Garage*.

são considerados ainda como pontos de interesse ocasional aqueles que pertencerem às seguintes categorias:

- *Public Services* (Ex: Correios, Banco, Multibanco, etc.);
- *Health Care* (Ex: Farmácias, Postos de primeiros socorros, etc.);
- *Eating&Drinking* (Ex: Café, restaurante, bar, etc.);
- *Leisure* (Ex: Teatro, cinema, centro comercial, etc.).

Considerando então as características definidas anteriormente e o tipo de condutores que se apresentam como principais destinatários, foi determinado que seria necessário os seguintes perfis:

- **ECO** - perfil puramente ecológico que permite considerar apenas os critérios **Declive** e **Manobras**;
- **ECO_TUR** - perfil que considera ambas as vertentes com os critérios ecológicos a prevalecerem sobre os turísticos;
- **ECO&TUR** - perfil intermédio em que são considerados todos os critérios com igual preponderância;
- **TUR_ECO** - perfil que considera ambas as vertentes com os critérios turísticos a prevalecerem sobre os ecológicos;

- **TUR_CUL** - perfil puramente turístico com incidência em pontos de interesse culturais e portanto priveligiando o critério **Turismo Cultural**;
- **TUR_OCA** - perfil puramente turístico com incidência em pontos de interesse ocasionais e portanto priveligiando o critério **Turismo Ocasional**.

Com os perfis descritos anteriormente, e que podem ser observados na tabela 4.1, consegue-se então considerar os extremos, o ponto intermédio e os pontos equidistantes entre estes que são os pontos fundamentais da gama de valores possíveis.

		Atributos			
		Declive	Manobras	Turismo Cultural	Turismo Ocasional
Perfis	ECO	1	1	0	0
	ECO_TUR	0.75	0.75	0.25	0.25
	ECO&TUR	0.5	0.5	0.5	0.5
	TUR_ECO	0.25	0.25	0.75	0.75
	TUR_CUL	0	0	1	0
	TUR_OCA	0	0	0	1

Tabela 4.1: Matriz dos perfis definidos para testes

4.1.3 NetEditor: editor de redes

Por forma a construir um ambiente de testes, com todas as suas partes constituintes, foi usado como base o *NetEditor*. Este é um editor de redes que foi desenvolvido pelo Laboratório de Inteligência Artificial e Ciência de Computadores (LIACC) da FEUP e que teve como intuito permitir a modelação de redes de tráfego urbanas e a sua exportação para diferentes simuladores, como é o caso do SUMO (*Simulation of Urban Mobility*).

Este editor de redes foi desenvolvido usando a linguagem de programação *C++* e a *framework Qt* pelo que tudo o que foi desenvolvido e integrado no mesmo, desde *plugins* e *widgets* até à implementação de algoritmos, seguiu as tecnologias que já vinham sendo utilizadas.

Do *NetEditor* foi essencialmente usada a componente de modelação e a interface com o utilizador visto que a simulação não se enquadrava dentro dos objetivos desta tese.

4.1.3.1 Plugins para a base de dados

Por forma a adaptar o *NetEditor* à problemática desta tese foi necessário inicialmente criar dois plugins: *db_import* e *db_export*.

Ambos os plugins foram desenvolvidos para permitir a comunicação entre o editor e a base de dados criada. Assim, o *db_import* permite ao utilizador escolher o mapa que pretende importar da BD, a partir do nome com que foi gravado, e consoante isso importar as seguintes informações correspondentes:

- nós (*junction*);
- arestas (*network*);
- limite máximo de velocidade em cada segmento (*speed_restriction*);
- manobras possíveis, entre arestas, em cada nó (*connection*).

Adicionalmente são importados os pontos de interesse (*poi*), que foram criados como dados sintéticos, e alocados de forma relativamente aleatória aos segmentos existentes. Esta alocação é feita desta forma de modo a conseguir-se gerar mapas de teste com diferentes características em situações diferentes, apesar de o mapa base ser o mesmo.

De referir que os sinais luminosos (tabela *traffic_sign*) foram descartados visto que apenas terão influência subjetiva na velocidade das vias e esta pode ser manipulado através da interface do *NetEditor*.

Utilizando as funcionalidades disponibilizadas pelo *NetEditor*, e algumas entretanto adicionadas, é possível também editar as características do mapa. De entre as alterações possíveis destacam-se as seguintes:

- Adicionar/remover nós;
- Adicionar/remover arestas;
- Alterar o calibre das vias;
- Alterar o limite máximo de velocidade entre 50km/h e 120km/h;
- Adicionar/remover manobras;
- Editar a posição dos nós (e consecutivamente das arestas incidentes no mesmo).

Relativamente ao plugin *db_export*, funciona exatamente como o *db_import* mas no sentido inverso. Assim permite ao utilizador guardar um mapa entretanto criado através da modelação possível com o editor de redes, escolhendo o título do mesmo.

4.1.3.2 Adaptações para o *routing*

Tendo em consideração a necessidade de permitir ao utilizador escolher as rotas que pretende efetuar e de lhe fornecer a visualização do percurso a seguir, foi necessário alterar e adicionar algumas funcionalidades.

Pontos de início e fim

Inicialmente todo os pontos que apenas tivessem segmentos que incidiam sobre o mesmo eram considerados pontos de chegada enquanto que aqueles de onde apenas partiam arestas definiam-se como pontos de partida.

Para ser possível a escolha de rotas, a definição de pontos de partida e chegada foi alterada para passar a ser feita manualmente. Assim, o utilizador tem a possibilidade de ele próprio escolher quais os pontos de início e fim que pretende para a sua rota sendo que apenas pode existir uma partida e uma chegada.

Seleção de perfis

Tendo em conta que os perfis foram definidos estaticamente, cada utilizador não tem um perfil associado mas sim um conjunto de perfis pelos quais pode optar de modo a "influenciar"a escolha da rota para as características que lhe interessam.

Posto isto, foi adicionado um novo *widget* de forma a ser possível escolher um dos perfis já definidos na Secção 4.1.2, o qual também contém as várias possibilidades de *routing* implementadas e que serão analisadas na Secção 4.2. Na Figura 4.3 pode então ser vista a interface desse *widget* para com o utilizador.

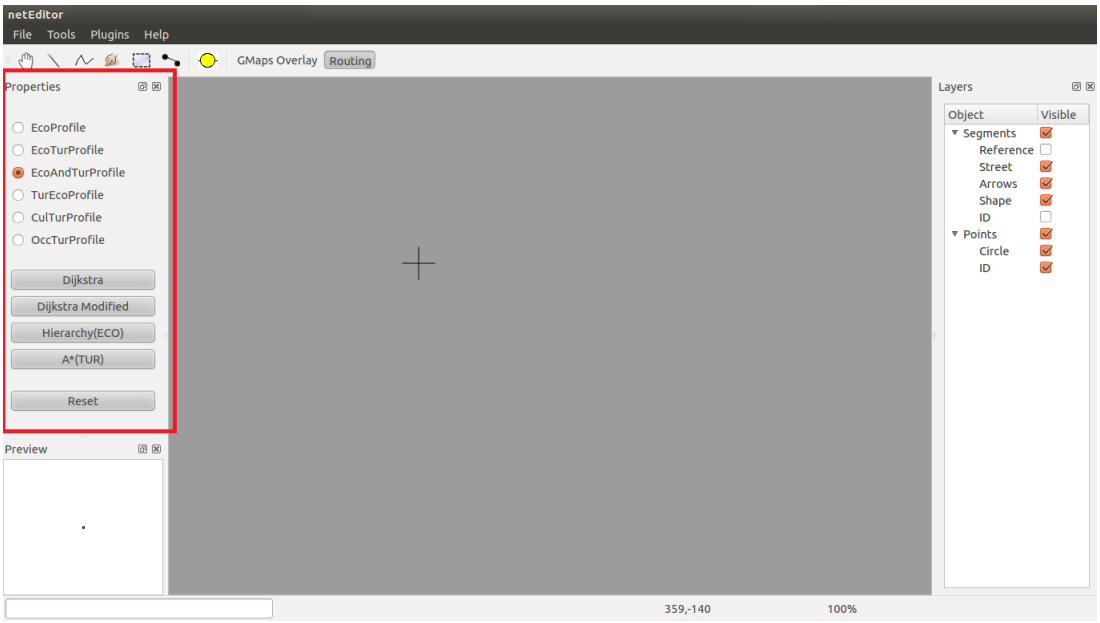


Figura 4.3: Editor de redes com o *widget* contendo os perfis de utilizador e as possibilidade de *routing*.

Algoritmos - *Dijkstra* como base de comparação

Como já visto na Figura 4.3, a interface para possibilitar fazer a escolha de rotas usando diferentes abordagens foi também desenvolvida e integrada.

No entanto, dos algoritmos apresentados apenas o *Dijkstra* foi implementado nesta fase visto ser aquele que foi escolhido para servir como base de comparação relativamente às restantes abordagens implementadas e que serão detalhadas na Secção 4.2.

Nesta implementação do algoritmo de *Dijkstra*, a função de custo considera não só a distância como também o tempo.

Considerando então que é dado um grafo $G(N,S)$, onde este algoritmo é aplicado, em que N representa o conjunto de todos os nós e S o conjunto de todas as arestas, pretende-se que em cada passo a função de custo seja minimizada como é mostrado na equação (4.1). Esta função $f(n,s)$ é resultado portanto da soma de duas parcelas, $g(n,s)$ e $t(s)$, em que a primeira representa a distância do nó n à origem e a segunda devolve o tempo que se demora a percorrer a aresta s. Assim, a distância à origem é dada pela soma da distância acumulada até chegar ao ponto atual ($accDist(n)$) com o comprimentos do segmento s ($d(s)$), como é mostrado na equação (4.2).

Relativamente ao tempo, como é exibido na equação (4.3), calcula-se pela divisão entre a distância do segmento s e a velocidade máxima permitida para circular nesse segmento ($v(s)$). Este cálculo é multiplicado por 60 para se obter o resultado em minutos sendo que a velocidade máxima permitida só pode oscilar entre 50km/h e 120km/h.

$$Min \quad f(n,s) = g(n,s) + t(s) \quad , \quad n \in N \quad e \quad s \in S \tag{4.1}$$

$$g(n,s) = accDist(n) + d(s) \tag{4.2}$$

$$t(s) = \frac{d(s)}{v(s)} * 60 \quad , \quad 50km/h < v(s) < 120km/h \tag{4.3}$$

4.2 Abordagens desenvolvidas

Uma vez concluído o ambiente de testes, entrou-se na fase relacionada com a algoritmia, a qual poderia ter sido encarada de duas formas:

- **Uma abordagem** - desenvolver só um algoritmo e apenas incidir sobre a otimização do mesmo;
- **Várias abordagens** - desenvolver diferentes possibilidades de resolver o mesmo problema focando os esforços na diversidade de otimizações.

De entre essas possibilidades optou-se pela segunda visto que, apesar de nem todas as abordagens comportarem a globalidade dos aspetos referidos na apresentação do problema, foi possível tirar conclusões mais efetivas relativamente ao comportamento de cada uma e comparar os pontos em que cada uma se comporta melhor ou pior relativamente às restantes.

Assim sendo, foram desenvolvidas três abordagens diferentes para a escolha de rotas, em que aplicaram diferentes técnicas de otimização tendo como base o algoritmo de *Dijkstra*, a hierarquia de grafos e o algoritmo A*, como é apresentado nas secções seguintes.

4.2.1 Algoritmo de *Dijkstra* modificado

A primeira abordagem que foi desenvolvida resultou da modificação do algoritmo de *Dijkstra*, já previamente implementado, de forma a alterar a sua função de custo.

O objetivo desta alteração passou principalmente pela inclusão das preferências de um utilizador, através do seu perfil, no cálculo da melhor rota a seguir. Com isto passou-se ainda a ter um algoritmo completamente direcionado para os múltiplos critérios, que era um dos objetivos principais, além da completa adequação ao utilizador através da personalização da rota a sugerir, correspondendo esta às características do perfil utilizado.

Desta forma, a função de custo passou a considerar os seguintes critérios:

- Distância;
- Tempo;
- Declive;
- Manobras;
- Turismo Cultural;
- Turismo Ocasional.

Devido a isto, os dados de entrada necessários para proceder à execução do algoritmo desenvolvido passaram a ser:

- $G(N,S)$ - grafo constituído pelo conjunto de nós N e pelo conjunto de arestas S;
- *start* - nó de início do percurso pretendido, em que $start \in N$;
- *final* - nó de fim do percurso pretendido, em que $final \in N$;
- p - vetor de 4 posições que representa o perfil do utilizador atual.

Tendo isto em consideração, e por termo de comparação com o que já foi descrito para o algoritmo de *Dijkstra* tradicional em 4.1.3.2, passamos a ter uma função de custo a considerar dois termos: o custo tradicional(*custoTrad*) e o custo do perfil(*custoProf*). Desta forma, o objetivo deste algoritmo passou a ser, em cada iteração, o seguinte:

$$Min \quad f(n,s,p) = f_1 * custoProf(n,s,p) + f_2 * custoTrad(n,s) \quad , \quad n \in N \quad e \quad s \in S \qquad (4.4)$$

Na equação (4.4) há três aspetos importantes a ter em consideração:

1. Dados de entrada: n representa o nó atual em processamento, s o possível segmento a seguir e p o perfil do utilizador, em que para cada posição i, com $0 \leq i < 4$, a preferência x terá sempre de ser um valor no intervalo [0;1], não necessariamente inteiro.

2. Fatores: cada parcela da soma que resulta em f é multiplicada por um fator parametrizável (f_1 e f_2). Estes dois fatores são usados com o intuito de ser possível privilegiar uma parcela relativamente à outra e têm de respeitar sempre a condição $f_1 + f_2 = 1$.

3. Funções de custo: o valor de *custTrad* refere-se ao custo de seguir o segmento em questão tendo em conta o tempo e a distância enquanto que a função *custProf* retorna um valor de acordo com as características do perfil do utilizador, como será detalhado de seguida.

custoTrad

Esta parcela entra nos cálculos de f como sendo a função de custo definida no algoritmo tradicional e que considerava apenas o tempo e a distância como critérios.

Desta forma, o valor retornado pela equação (4.1) (que se assume como sendo f_T para não causar confusões de leitura) seria a representação do *custoTrad*. Isto apenas não acontece porque devido à gama de valores bastante diferente que era obtida entre o *custoTrad* e o *custoProf* foi necessário normalizar estes custos.

Assim definiu-se que ambos iriam assumir um valor entre 0 e 1, inclusivé. Posto isto, e considerando uma regra de três simples, obtém-se o valor de *custoProf* através da equação (4.5).

$$custoTrad(n,s) = f_T(n,s)/maxCustoTrad \tag{4.5}$$

A incógnita *maxCustoTrad* corresponde ao valor máximo previsível para a rota total a efetuar. Este é também um valor parametrizável e que foi definido como sendo k vezes a distância Euclidiana entre os pontos *start* e *final*.

O k é também um valor ajustável pelo que o valor de *custoTrad* poderá então oscilar.

custoProf

Esta função foi aquela que assumiu maior importância tendo em conta que usou os dados do perfil do utilizador e portanto definiu em que medida um determinado segmento "interessa" ao utilizador atual ou quando é que uma estrada deve ser descartada em detrimento de outra que se define ter maior proximidade com o perfil em análise.

Assim sendo, e tendo em conta que os perfis já definidos na secção 4.1.2 têm 4 critérios de classificação, o valor do *custoProf* é dado por:

$$custoProf(n,s,p) = \sum_{i=0}^{m} x_i * c_i \quad , \quad m = 3 \tag{4.6}$$

Nesta equação, o valor de m é sempre igual a uma unidade inferior ao número de características a considerar de modo a percorrer-se na íntegra os dois vetores com as características do perfil do utilizador e da zona do mapa a considerar.

Desta forma, o valor x_i corresponde à preferência do utilizador relativamente ao atributo i enquanto que o valor c_i diz respeito à ocorrência desse critério no mapa.

Tendo em consideração que c_i não é calculado de igual forma para todas as características torna-se necessário analisar caso a caso. Assim sendo, o valor de *i* irá fazer corresponder a um dos seguintes critérios que será calculado da seguinte forma:

- $Declive = \begin{cases} 1 & \text{se } |grau| < 5\% \\ 0 & \text{se } 5 \leq |grau| < 10\% \\ -1 & \text{outros casos} \end{cases}$
- $Manobras = \begin{cases} 1 & \text{se } ang < 45^{\text{o}} \\ 0 & \text{se } 45^{\text{o}} \leq ang < 90^{\text{o}} \\ -1 & \text{outros casos} \end{cases}$
- $TurismoCultural = \begin{cases} 0 & \text{se aresta não possui pontos de interesse culturais} \\ 1+\ln(N_c) & \text{caso contrário} \end{cases}$
- $TurismoOcasional = \begin{cases} 0 & \text{se aresta não possui pontos de interesse ocasionais} \\ 1+\ln(N_o) & \text{caso contrário} \end{cases}$

Relativamente às equações anteriores é importante referir que o *grau* se refere à percentagem de inclinação da estrada representada pelo segmento em processamento enquanto que a variável *ang* diz respeito ao ângulo de curvatura correspondente à manobra que o condutor terá de efetuar para seguir o segmento referido. No entanto, devido a insuficiência de dados que permitiriam considerar o *Declive*no mapa, esta característica é considerada sempre como sendo 0.

Em relação às equações das características turísticas, as variáveis N_c e N_o assumem um valor igual ao número de pontos de interesse da categoria cultural e ocasional, respetivamente, que o segmento em processamento contiver.

Ainda relativamente a isto, a escolha da função logarítmica *ln()* deveu-se à necessidade de se privilegiar a existência de vários pontos da mesma categoria num mesmo segmento, tendo sempre em consideração que esse valor deixa de ser tão preponderante a partir de um número não muito elevado.

Desta forma, o que acontece é que para um utilizador que apresenta um perfil com interesse turístico-cultural, o algoritmo privilegia a escolha de segmentos com 2 pontos dessa categoria em vez de segmentos com apenas 1. O mesmo acontecerá com 3 em relação a 2 mas a partir de um certo valor deixa de fazer sentido continuar a diferenciar e é exatamente isso que se pretende e se pode ver na Figura 4.4

Finalizada esta análise importa referir que, como já tinha acontecido com a parcela *custoTrad*, também este valor foi normalizado para uma gama entre 0 e 1 através da divisão do valor resultante de *custoProf* pelo máximo valor possível que aquele utilizador poderia obter caso todas as características do mapa correspondessem ao seu perfil.

Desta forma, o máximo valor possível é calculado usando a equação (4.6) mas em que os valores correspondentes ao grafo assumem os máximos possíveis, ou seja, 1 para o declive e para as manobras e $1+\ln(nPois)$ para o turismo cultural e ocasional. De referir que o argumento

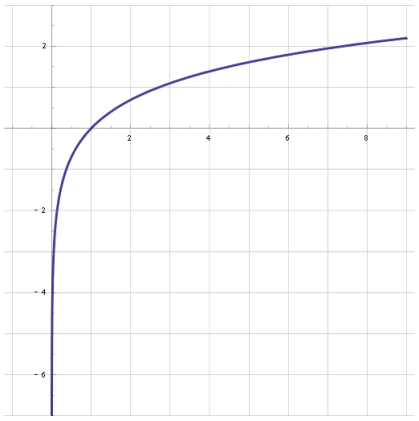


Figura 4.4: Representação gráfica da função $ln()$.

da função logarítmica é *nPois* que corresponde ao número máximo de pontos de interesse por segmento que é um valor limitado visto que estes são alocados aleatoriamente aos segmentos e poderiam haver segmentos com um número absurdo de pontos de interesse.

Com esta nova abordagem para o algoritmo de *Dijkstra*, conseguiu-se então passar a ter as rotas a serem escolhidas com base em múltiplos critérios, 6 no caso, e também a serem sugeridas de forma pessoal e personalizada, como era objetivo, e de acordo com as preferências de cada um.

4.2.2 *Routing* hierárquico ao nível dos nós

Com um objetivo, ou uma forma de enquadrar o problema, um pouco diferente da abordagem anterior foi implementado um algoritmo tendo como ponto de partida o conceito de hierarquia.

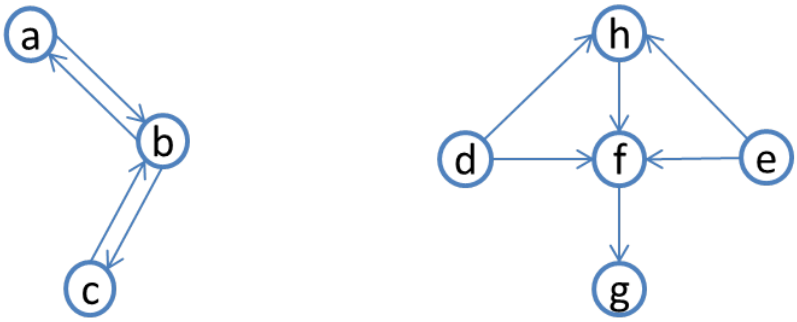
Esta abordagem desenvolvida teve um foco especial não no processamento do grafo para a escolha da rota final mas sim num pré processamento do mesmo.

Tendo isto em conta, e sempre os perfis de utilizador como ponto a considerar, as técnicas inteligentes usadas neste algoritmo foram direcionadas para o *routing* ecológico e desta forma tiveram especial incidência nos nós do grafo e nas manobras do condutor ao longo da sua rota.

Assim sendo, foram usadas essencialmente duas técnicas de pré processamento: uma para ignorar os nós que não são necessários na fase de processamento e outra para criar uma hierarquia com os nós do grafo.

Relativamente aos nós a ignorar, o objetivo passou por remover aqueles nós cujo processamento nunca alteraria o normal fluxo do algoritmo, tipicamente nós com um baixo grau de entrada e de saída. De referir que o grau de entrada de um nó é igual ao número de arestas incidentes no mesmo enquanto que o grau de saída corresponde ao número de arestas que partem do referido nó.

Posto isto, considerou-se dois tipos de nós: aqueles em que apenas existe uma conexão entre dois segmentos e os que têm um grau de saída igual a 1, independentemente do grau de entrada. Estes dois tipos de nós estão ilustrados na Figura 4.5.



(a) Nó com ligação apenas entre 2 segmentos. (b) Nó com grau de saida = 1.

Figura 4.5: Nós a serem ignorados em pré processamento.

Nessa figura estão então representados os nós b e f que devem ser removidos. Esta técnica passa essencialmente pela criação de arestas fictícias. Assim, e no caso (a), seriam criadas duas ligações artificiais entre os nós a e c, uma em cada sentido, e seriam removidas as restantes ligações e o nó b. Estas ligações criadas passam a ter as características relativas aos pontos de interesse, aos limites de velocidade e ao comprimento de acordo com as que foram removidas. Assim sendo:

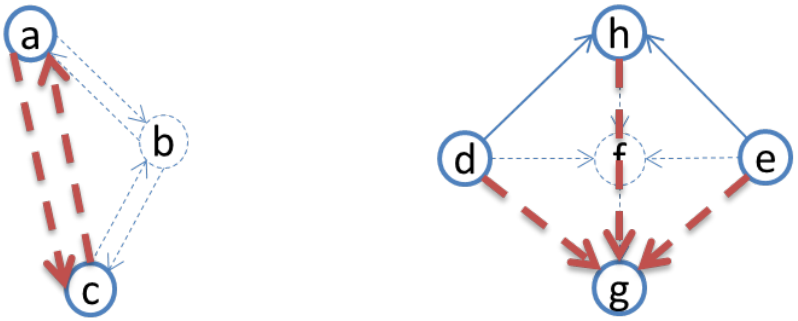
- Pontos de interesse: conjunto de todos os pontos de interesse dos segmentos removidos que está a substituir;
- Velocidade: limite máximo passa a ser igual ao mínimo dos limites dos segmentos removidos que está a substituir;
- Comprimento: tipicamente seria a soma do comprimento dos segmentos removidos que está a substituir mas passa a ser substituído pela equação (4.7), onde s1 e s2 se referem aos segmentos removidos. Nessa mesma equação a função *d()* corresponde ao comprimento do segmento e o fator l serve para "beneficiar"ou "prejudicar"a soma dos comprimentos pelas condições apresentadas em (4.8).

$$Comprimento = (d(s_1) + d(s_2)) * l \tag{4.7}$$

$$l = \begin{cases} 1.5 & \text{se nó removido era de nível 0} \\ 1 & \text{se nó removido era de nível 1} \\ 0.5 & \text{se nó removido era de nível 2} \end{cases} \tag{4.8}$$

Estas alterações anteriormente descritas estão ilustradas na figura 4.6.

Na equação (4.8) foram descritas algumas condições que envolviam o nível dos nós. Isto foi algo ainda não abordado anteriormente mas que está já interligado com a próxima técnica de pré processamento que passa pela criação de hierarquias no grafo original.



(a) Nó *b* ignorado. (b) Nó *f* ignorado.

Figura 4.6: Duas situações diferentes em que nós são ignorados.

Assim sendo, um nó pode ter um de três níveis possíveis (0, 1 ou 2) que são definidos de acordo com as ligações que existem nele mesmo. Como já havia sido delineado anteriormente, as conexões foram divididas em três grupos, consoante o grau que lhe é inerente, sendo eles:

- Suave - grau de curvatura inferior a 45º;
- Normal - grau de curvatura entre 45º e 90º;
- Acentuada - grau de curvatura superior a 90º.

Tendo isto em consideração, a cada nó vai ser atribuído um nível de acordo com a percentagem de ligações que tem de cada um dos grupos acima descritos, como é descrito de seguida.

- Nível 0 - maior percentagem de nós do grupo "Acentuada";
- Nível 1 - maior percentagem de nós do grupo "Normal";
- Nível 2 - maior percentagem de nós do grupo "Suave".

Assumindo isto, para um dado grafo G, são efetuadas duas iterações: na primeira é construído um novo grafo G' a partir do grafo G mas ao qual foram removidos os nós de grau 0 e na segunda iteração atingisse o grafo G", que corresponde ao nível mais elevado definido, obtido através da remoção dos nós de nível 1 do grafo G'.

Esta eliminação sucessiva de nós de modo a atingir níveis superiores pode levar, em certas situações, a não se obter um grafo conexo o que acaba por não ser problemático tendo em conta o algoritmo seguido para o processamento, como se verá mais à frente.

Na Figura 4.7 pode então ver-se um grafo base criado aleatoriamente. Cada nó contém no seu interior um número indicativo do nível respetivo que foi utilizado para criar a hierarquia representada na Figura 4.8.

Com estas duas técnicas de pré processamento desenvolvidas conseguiu-se desde logo reduzir o número de nós e de segmentos do grafo original além de que se criou uma hierarquia de estradas

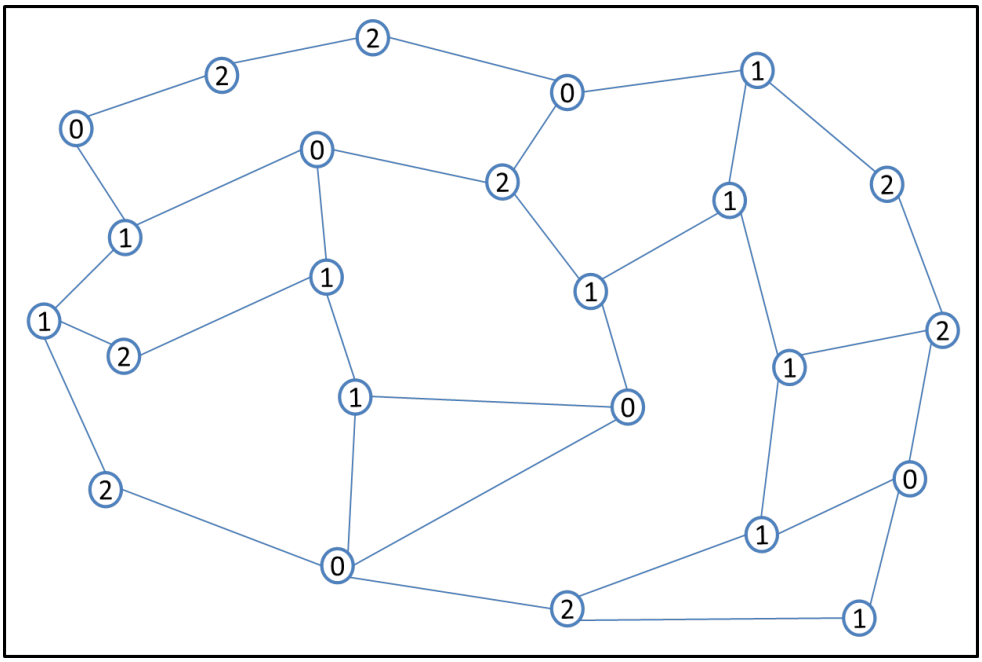
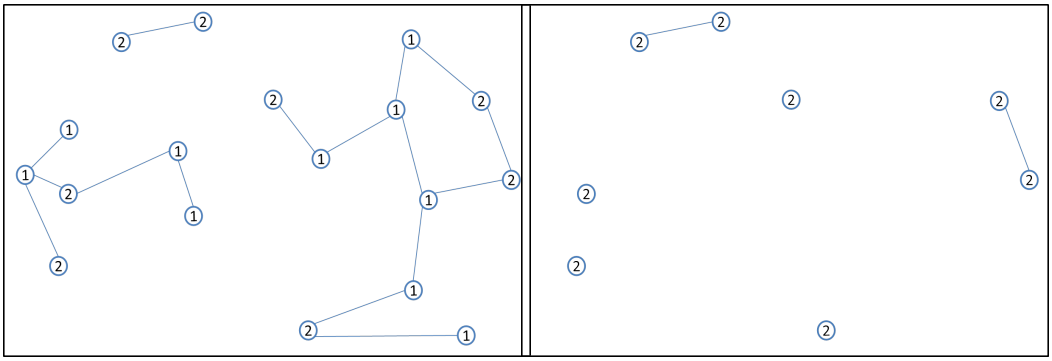


Figura 4.7: Grafo base a hierarquizar



(a) Nível 1. (b) Nível 2.

Figura 4.8: Representação dos 2 níveis de um grafo de acordo com a hierarquia dos seus nós constituintes.

tendo em consideração os níveis de cada nó, e por conseguinte os ângulos das conexões existentes nos mesmos, que é um dos fatores fundamentais para o *routing* ecológico. Desta forma, pode dizer-se que quantos mais nós se atingir do nível mais elevado mais ecológico vai ser o percurso.

Esta seleção dos nós a seguir é efetuada já numa fase diferente e com base no algoritmo de *Dijkstra*.

Assim, e ao contrário do que aconteceu na abordagem apresentada na Secção 4.2.1, a função de custo apresentada inicialmente através da equação 4.1, e que considerava os critérios de distância e de tempo, foi mantida incidindo as otimizações na definição de sucessor de um nó, de modo a ir de encontro ao que já havia sido feito em pré processamento.

Desta forma, um sucessor de um nó não é todo e qualquer outro nó atingível diretamente mas apenas aquele que tiver o nível hierárquico mais elevado possível. Desta forma a escolha dos sucessores passa por três etapas:

1. Selecionar todos os nós atingíveis diretamente;
2. Calcular o nível mais alto possível entre os nós selecionados;
3. Extrair apenas os nós que pertençam a esse nível.

Com isto consegue-se reduzir o número de nós sucessores restringindo a pesquisa àqueles que localmente estão num nível mais elevado e portanto vão levar a ligações entre segmentos com ângulos de curvatura mais suaves, indo assim ao encontro das características de um perfil ecológico. Inerente à diminuição do número de nós sucessores está o número de arestas analisadas em cada iteração que diminui também consideravelmente.

Todo este fluxo de processos está representado através de um fluxograma na Figura 4.9. Nesse fluxograma, apesar de a fase de processamento estar ligada com a anterior, esta ligação apenas indica que o pré processamento do grafo vai funcionar como dado de entrada no processo de escolha da rota a sugerir ao utilizador.

4.2.3 Nova heurística para o algoritmo A*

A última abordagem desenvolvida foi aquela que deixou completamente de parte o algoritmo de *Dijkstra* e surgiu como uma nova heurística aplicada ao algoritmo A*.

Este algoritmo, um dos mais conhecidos e usados em problemas deste tipo, usa uma função de custo que contém duas parcelas: a primeira baseada na distância do nó atual à origem (correspondente à função de custo do algoritmo de *Dijkstra*) e a segunda que se refere à estimativa da distância/proximidade ao destino como medida para acelerar o processo de cálculo da rota.

Foi precisamente nesta segunda parcela, que é a heurística usada no algoritmo tradicional, que se enquadrou esta nova abordagem.

Assim sendo, o objetivo da nova heurística desenvolvida passava por estimar não só a distância ao destino mas também os pontos de interesse pelos quais seria possível passar até o atingir. Com este objetivo conseguir-se-ia então orientar a pesquisa para utilizadores com perfil turístico, quer seja cultural que seja ocasional, ao contrário da abordagem descrita em 2.1.6.

Esta heurística é representada portanto por uma função que engloba a multiplicação de dois fatores, como é apresentado na equação 4.9.

$$Min \quad h(n) = goal(n) * p(n) \quad , \quad n \in N \tag{4.9}$$

A função *goal(n)* é responsável por devolver a distância do nó *n* ao nó objetivo *t*. Este valor é então dado pela distância Euclidiana entre os dois como é mostrado pela equação 4.10.

$$euc(n,t) = \sqrt{|t.x() - n.x()|^2 + |t.y() - n.y()|^2} \tag{4.10}$$

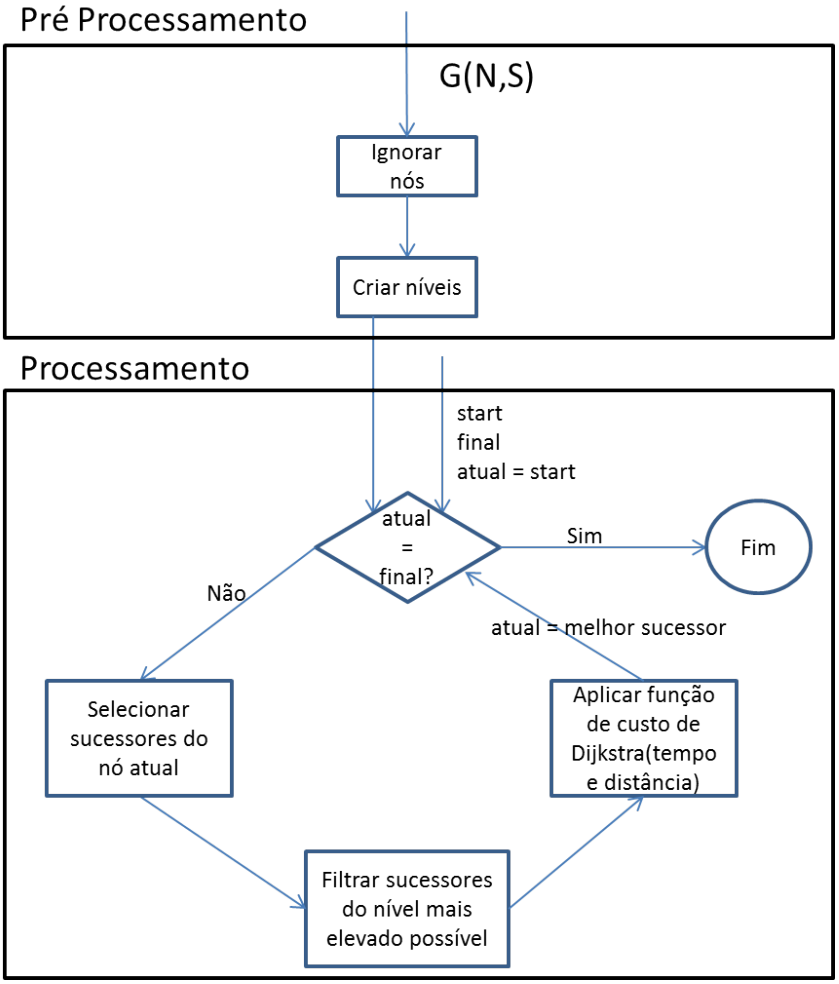


Figura 4.9: Fluxograma das fases de pré processamento e processamento relativo ao *routing* hierárquico.

Quanto à função *p(n)*, esta está relacionada efetivamente com o número de pontos de interesse atingíveis. No entanto, como o objetivo passa por minimizar a heurística, a abordagem desenvolvida passou por fazer uma estimativa inicial do número de pontos de interesse pelos quais o condutor passaria, até atingir o nó objetivo, e na análise de cada nó verificar a estimativa para este e calcular diferença de estimativas de modo a minimizar a "perda" de pontos de interesse por onde passar.

Posto isto, e analisando passo a passo, o valor de *p(n)* é calculado da seguinte forma:

1. Calcular a previsão do número de pontos de interesse atingíveis a partir do nó de início;
2. Definir esse valor como o número máximo alcançável;
3. Em cada iteração, e para cada nó:
    - Calcular a previsão do número de pontos de interesse atingíveis a partir deste nó;

- Calcular a percentagem de pontos de interesse "perdidos" ou "ganhos" através da expressão $(maxPois - nPois)/maxPois$, em que *maxPois* representa o número máximo atingível no início e *nPois* o número atingível no momento.
- Somar 1 ao valor anterior, que está compreendido no intervalo [0;1], de modo a ser possível influenciar o valor da distância prevista ao destino de forma positiva, negativa ou até neutra.

4. Retornar a diferença calculada.

Este valor retornado contribuirá tanto mais para a escolha do respetivo nó como sucessor quanto menor for o seu valor. Assim sendo, nos casos extremos em que $nPois = maxPois$ ou $nPois = 0$, o valor desta função será 1 ou 2, respetivamente. Posto isto, quanto menor for o número de *nPois* relativamente a *maxPois* mais inflacionada vais ser a distância ao destino e portanto menos "possibilidades" terão esses segmentos de serem seguidos. De referir que o valor estimado para um nó pode ser maior do que o que foi definido inicialmente como o máximo e isso leva ao retorno de um valor menor do que 1, ou até mesmo negativo, o que faz com a heurística tenha uma valor ainda menor e assim sejam ainda mais privilegiados estes nós.

Ainda relativamente a esta abordagem torna-se necessário e fundamental descrever detalhadamente a forma como é calculada a previsão dos pontos de interesse por onde se poderá passar a partir de um determinado nó.

Assim, o conceito fundamental deste cálculo passa por delimitar uma área englobando o nó atual e o nó de destino. De acordo com o que está documentado na literatura [LLHH05], uma elipse apresenta-se como a forma geométrica mais simples, além do círculo, que se pode implementar para problemas de distância, e já foi sugerida como forma de reduzir o espaço de pesquisa para se obter o caminho ótimo entre dois pontos [HWZ07].

Com base nesta considerações, a elipse foi efetivamente a forma utilizada também nesta heurística para determinar o número de pontos de interesse atingíveis, assumindo-se que o melhor caminho irá estar dentro da mesma e portanto também os segmentos por onde se irá passar, e que contêm pontos de interesse.

Um problema que pode comportar o uso da elipse prende-se com as características da mesma, nomeadamente com a forma como a construir, ou seja, o tamanho dos seus eixos e os focos a usar.

Tendo em consideração a elipse ilustrada na Figura 4.10 e a noção de que uma elipse é o lugar geométrico dos pontos cuja soma das distâncias aos focos (*F1* e *F2*, neste caso) é sempre constante e igual a *2a* (comprimento do eixo maior), estes foram efetivamente os valores preponderantes para a definição da mesma. Assim sendo, foi decidido o seguinte:

- F1: posição do nó a ser processado no momento;
- F2: posição do nó final do percurso pretendido;
- a = $euc(F1, F2) + 2 * bound$, em que *bound* é um valor parametrizável e corresponde à distância do foco até ao vértice mais próximo. Atualmente este valor corresponde a 5% da

distância entre focos de modo a permitir relacionar este valor com a distância do percurso pretendido sem dar no entanto uma margem muito grande para andar na direção contrária ao destino ou até ultrapassar este e ter de voltas para trás no seu percurso.

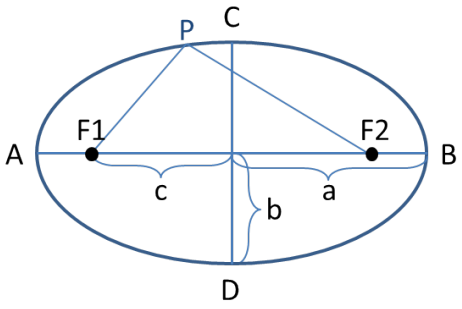


Figura 4.10: Elipse.

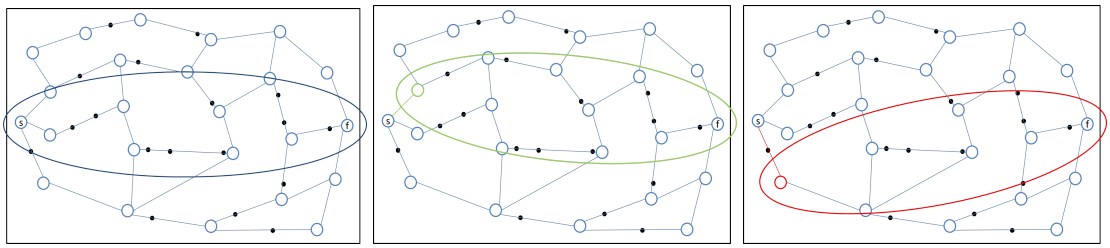
Com isto podem ser testados diferentes valores para o tamanho do eixo maior da elipse e analisar assim as implicâncias que a alteração deste valor efetivamente tem.

Considerando então a equação que está inerente à definição de elipse já descrita, todo e qualquer ponto de interesse para estar na zona delimitada pela mesma terá de respeitar a seguinte condição:

$$euc(p1,F1) + euc(p1,F2) \leq 2a \tag{4.11}$$

Desta forma, o número de pontos de interesse atingíveis corresponderá ao número dos que respeitarem a condição anterior adicionado à quantidade contida no segmento utilizado para chegar ao nó atual.

Assim, na Figura 4.11 pode ver-se no caso (a) que a elipse definida englobava 10 pontos de interesse (pontos a negro) entre o ponto de inicio (s) e o ponto de fim (f). Na mesma figura, os casos (b) e (c) mostram uma suposta análise a dois sucessores do nó s em que vemos que para o sucessor a verde havia uma previsão de 11 pontos de interesse enquanto que para o vermelho esse valor era apenas de 9.



(a) (b) (c)

Figura 4.11: Pontos de interesse englobados pelos limites de uma elipse.

Assim sendo, para esta situação o valor da função $p(n)$ seria:

- Verde: $1 + \frac{10-11}{10} = 0.9$
- Vermelho: $1 + \frac{10-9}{10} = 1.1$

Com isto, o valor da heurística passaria a aumentar numa percentagem de 10% num dos casos e diminuir, na mesma proporção, no outro pelo que, se se analisasse apenas a questão dos pontos de interesse, o sucessor verde seria preferido relativamente ao vermelho.

Neste exemplo, bem como no que foi referido anteriormente, os pontos de interesse foram tratados como sendo todos iguais. No entanto um dos dados de entrada necessários para esta abordagem é o perfil do utilizador de modo a que o cálculo seja descriminado relativamente a encontrar pontos de interesse culturais ou ocasionais.

Com esta abordagem foram então cumpridos os objetivos de implementar uma nova heurística em que fossem também considerados os pontos de interesse, e por conseguinte os perfis turísticos.

Capítulo 5

Análise de resultados

Neste capítulo pretende-se essencialmente demonstrar o funcionamento das abordagens implementadas e definidas no capítulo anterior.

Posto isto, são definidos alguns casos de teste que permitem posteriormente tirar conclusões efetivas sobre o cumprimento dos objetivos propostos.

Tendo em consideração que o projeto de dissertação passou pelo desenvolvimento de várias abordagens para a escolha de rotas, com a aplicação de técnicas inteligentes para se obter algoritmos eficientes e que gerassem resultados personalizados, não se pretende gerar muitos casos de teste mas sujeitar as diferentes abordagens aos mesmos testes.

Assim sendo, pretende-se testar os vários algoritmos desenvolvidos, e considerando os diferentes perfis existentes, tanto numa rede gerada de forma sintética e de dimensão menor como numa rede real e de dimensão já considerável.

5.1 Rotas geradas

Por forma a testar todos os aspetos desenvolvidos foram usadas essencialmente três redes. Destas, duas eram artificiais e uma real.

As redes artificiais foram criadas usando as funcionalidades de modelação de redes do *NetEditor* e posteriormente gravadas na base de dados com recurso ao *plugin* de exportação também criado no âmbito desta tese. À semelhança destas duas redes, que se apresentam de seguida, muitas outras foram criadas com o objetivo de servir como base de testes. No entanto, tendo em consideração que era necessário que as redes fossem bastante intuitivas e que permitissem através apenas da visualização avaliar as diferenças que eram obtidas com cada abordagem, estas foram efetivamente as que se revelaram como os melhores exemplos de teste.

Quanto à rede real foi importada do *Open Street Maps* (OSM) e refere-se a uma parte do mapa da cidade do Porto.

Todas as redes contam com um conjunto de pontos de interesse, que são mantidos na base de dados, e que são alocados a segmentos que a constituem.

Na representação das redes no editor foram usadas algumas cores para identificar pontos específicos. Assim sendo:

- Ponto de início da rota - verde;
- Ponto de fim da rota - vermelho;
- Ponto de interesse turístico-cultural - azul claro;
- Ponto de interesse turístico-ocasional - azul;
- Ponto de interesse de ambas as categorias - magenta.

Estas três redes estão representadas nas figuras 5.1, 5.2 e 5.3.

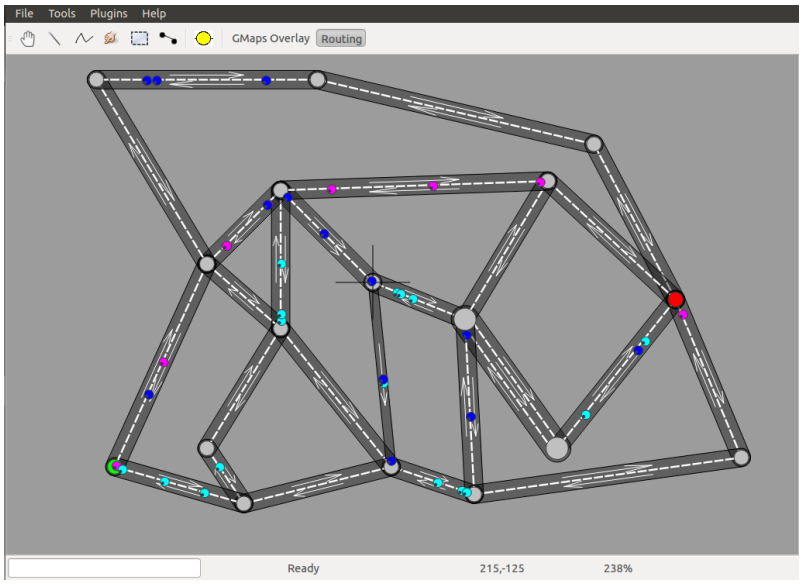


Figura 5.1: Rede artificial 1.

Relativamente à rede artificial 1, para o mesmo percurso foram testadas as situações respeitantes às abordagens desenvolvidas. Assim, usaram-se não só os diferentes algoritmos como também os diferentes perfis, o que perfez um total de oito testes.

Desses testes obtiveram-se quatro sugestões de percursos diferentes. Como é possível observar nas figuras 5.4, 5.5 e 5.6, as diferenças ocorreram essencialmente quando foram usados perfis de utilizador com uma vertente turística mais acentuada, ou mesmo exclusiva, e também ao usar a abordagem hierárquica. Estas diferenças são consideradas relativamente à imagem (a), da Figura 5.4, que representa o percurso base obtido usando o algoritmo de *Dijkstra*. Posto isto, e relativamente às abordagens com um objetivo principalmente ecológico, é possível observar que não devolveram resultados diferentes do resultado base.

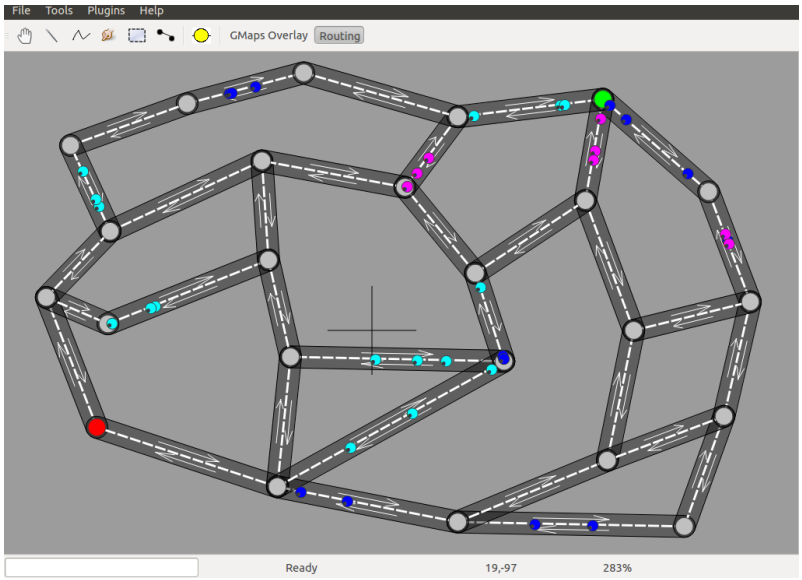


Figura 5.2: Rede artificial 2.

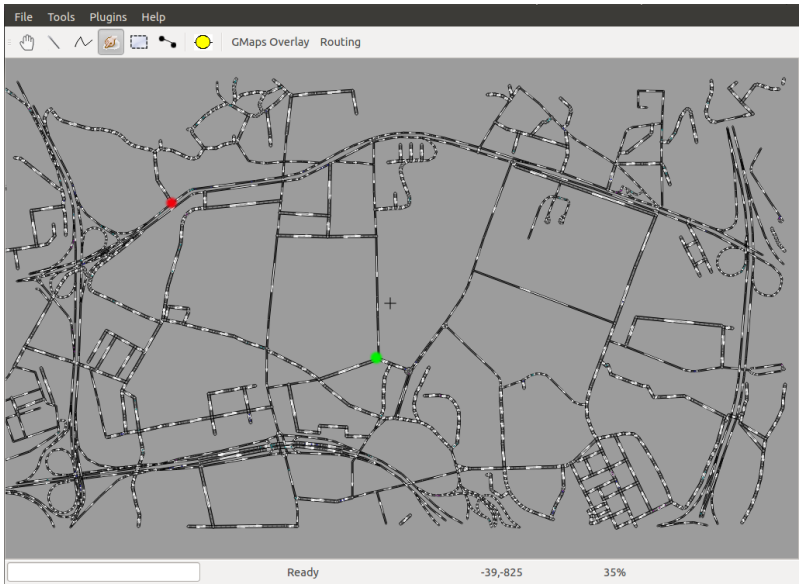


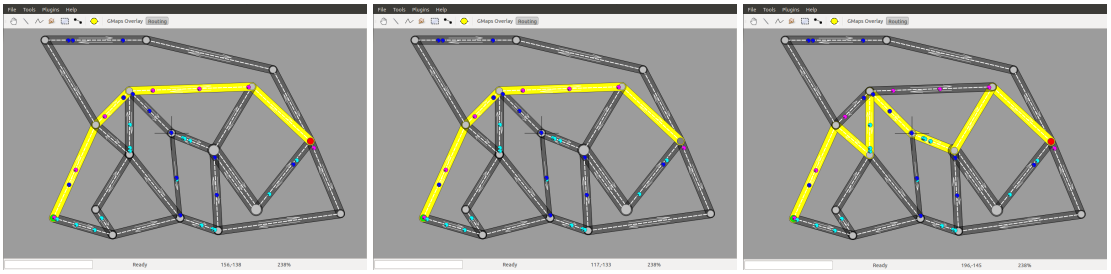
Figura 5.3: Rede real - zona específica da cidade do Porto.

Assim, existe uma zona central no mapa que contém um número mais elevado de pontos de interesse pelo que, apesar de levar a um aumento da distância total do percurso, acabou por ser uma zona preferencial nas abordagens relativas a utilizadores turísticos.

Quanto à abordagem hierárquica, em termos de rota acabou por não ser tão vantajosa. Como é possível observar na respetiva imagem, no terceiro ponto do percurso, um ponto de nível mais elevado, acabou por ser feita uma manobra muito acentuada. Isto aconteceu porque este nó tinha uma grande percentagem de possibilidade de corresponder a uma manobra suave mas o melhor nó

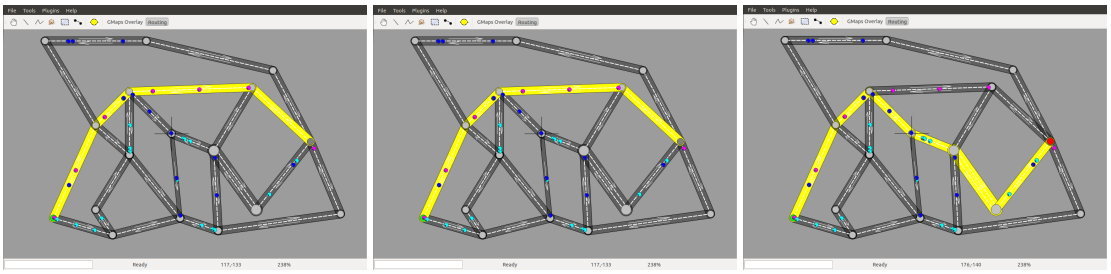
sucessor levou a que assim não fosse.

Relativamente a utilizadores com perfil acentuadamente ecológico, as diferenças não se fizeram notar uma vez que a rota de base já não implicava um conjunto de manobras muito acentuadas. Assim, e devido a que tanto a distância como os perfis tinham a mesma importância, não se verificaram nenhumas diferenças. Esta mesma importância foi definida pelo valor 0.5 que foi atribuído a cada variável parametrizável, f_1 e f_2, na equação 4.4,



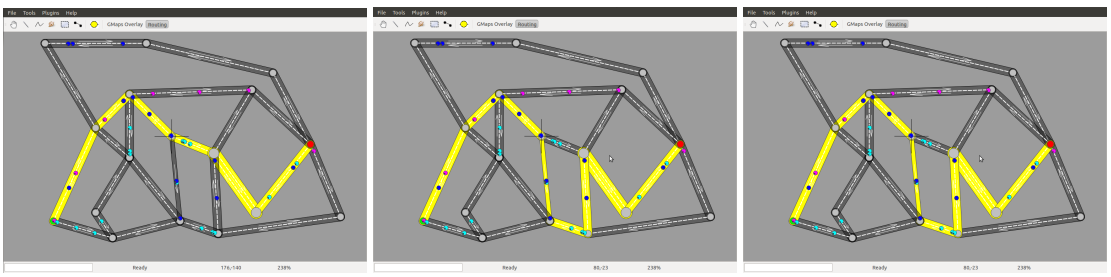
(a) Algoritmo de *Dijkstra* tradicional. (b) Algoritmo A* com nova heurística. (c) Pré-processamento hierárquico.

Figura 5.4: Rotas obtidas com 3 abordagens díspares sobre a rede artificial 1.



(a) Perfil ECO. (b) Perfil ECO_TUR. (c) Perfil ECO&TUR.

Figura 5.5: Rotas obtidas com 3 perfis diferentes com maior componente ecológica sobre a rede artificial 1.



(a) Perfil TUR_ECO. (b) Perfil TUR_CUL. (c) Perfil TUR_OCA.

Figura 5.6: Rotas obtidas com 3 perfis diferentes com maior componente turística sobre a rede artificial 1.

Ao analisar-se os mesmos testes efetuados anteriormente, mas agora sobre a rede artificial 2, é possível tirar algumas conclusões diferentes, comparativamente com o caso anterior, mas não contrárias.

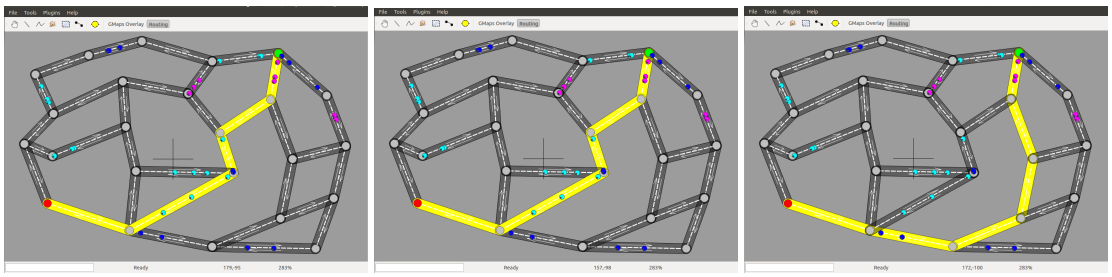
Em relação ao conjunto de imagens 5.7 aconteceu novamente o mesmo que no caso anterior, ou seja, a nova heurística implementada para o algoritmo A* retornou um caminho igual ao que já havia sido calculado através do algoritmo de *Dijkstra*, ao contrário do que aconteceu com a abordagem hierárquica.

Nesta abordagem acaba por não haver também grandes diferenças mas as poucas que há são bastante significativas. Isto pode afirmar-se porque é fácil observar que, a partir desta abordagem, foram excluídos do percurso final dois nós que envolviam curvas mais acentuadas do que nas soluções anteriores.

Relativamente aos testes envolvendo mais diretamente os perfis de utilizador, é possível verificar que foram escolhidas rotas completamente diferentes das anteriores. Assim, em 6 testes obtiveram-se 3 rotas diferenciadas o que é considerável tendo-se em conta a personalização das rotas pretendidas.

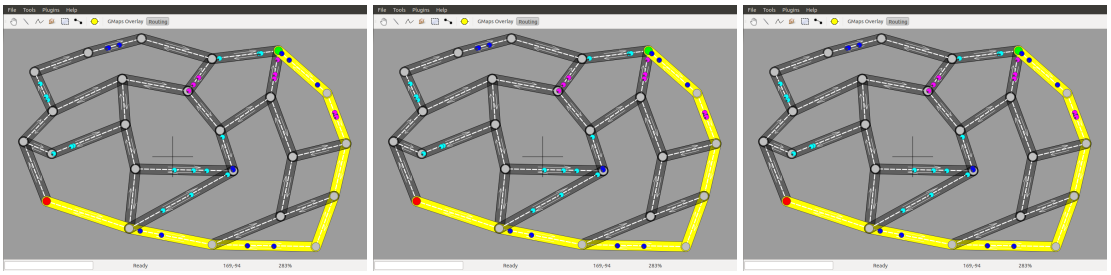
Ainda relativamente aos perfis, convém salientar que aos perfis envolvendo componente ecológica foi sugerida sempre a mesma rota ao contrário do que aconteceu com os turísticos. Isto aconteceu porque, como é possível ver nas imagens (b) e (c) da Figura 5.9, as rotas turísticas acabaram por envolver várias manobras agressivas pelo que, mesmo considerando em igual percentagem as componentes turística e ecológica, nunca os pontos de interesse se conseguiram sobrepor à distância da rota e às manobras referidas.

Comparando apenas as duas rotas turísticas, as diferenças começam logo nos segmentos iniciais em que, para o turista cultural (pontos de interesse a azul claro) foi logo escolhido um primeiro segmento que acabou por influenciar a restante rota. Além disso, é possível observar que na zona intermediária do mapa foi efetuado outro desvio, embora que curto, para "visitar"outros pontos de interesse da mesma categoria.



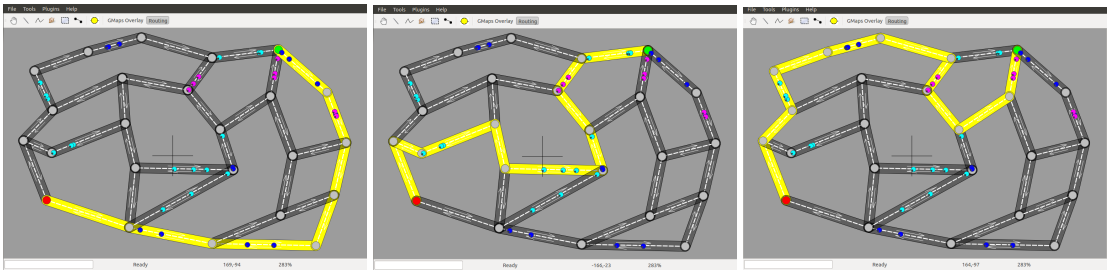
(a) Algoritmo de *Dijkstra* tradicional. (b) Algoritmo A* com nova heurística. (c) Pré-processamento hierárquico.

Figura 5.7: Rotas obtidas com 3 abordagens díspares sobre a rede artificial 2.



(a) Perfil ECO. (b) Perfil ECO_TUR. (c) Perfil ECO&TUR.

Figura 5.8: Rotas obtidas com 3 perfis diferentes com maior componente ecológica sobre a rede artificial 2.

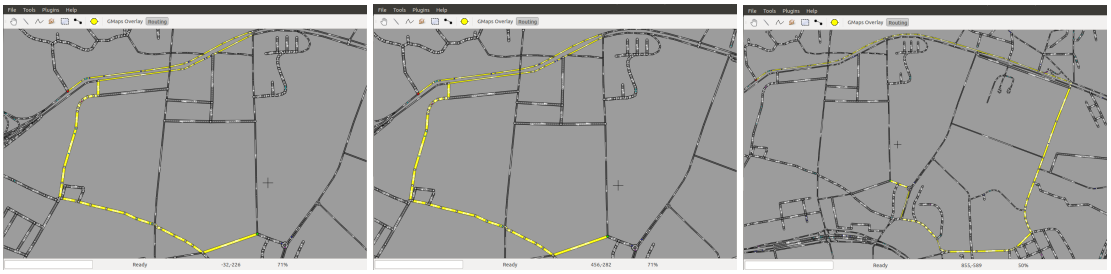


(a) Perfil TUR_ECO. (b) Perfil TUR_CUL. (c) Perfil TUR_OCA.

Figura 5.9: Rotas obtidas com 3 perfis diferentes com maior componente turística sobre a rede artificial 2.

No que diz respeito ao último destes 3 mapas, aqui as diferentes abordagens foram submetidas a um grafo de dimensões bem superiores. Apesar disso, as conclusões acabaram por ser bastante semelhantes às referidas em relação à rede artificial 2.

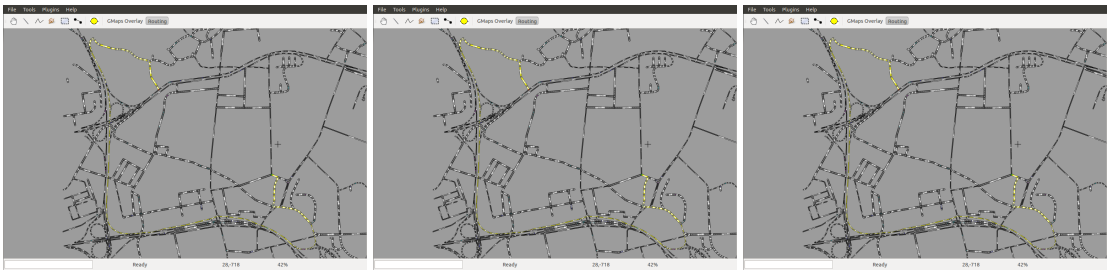
A principal diferença neste mapa deveu-se a que o elevado número de estradas não permitisse ter um número de pontos de interesse tão significativos numa qualquer zona e que levasse a ter rotas mais personalizadas nesses casos.



(a) Algoritmo de *Dijkstra* tradicional. (b) Algoritmo A* com nova heurística. (c) Pré-processamento hierárquico.

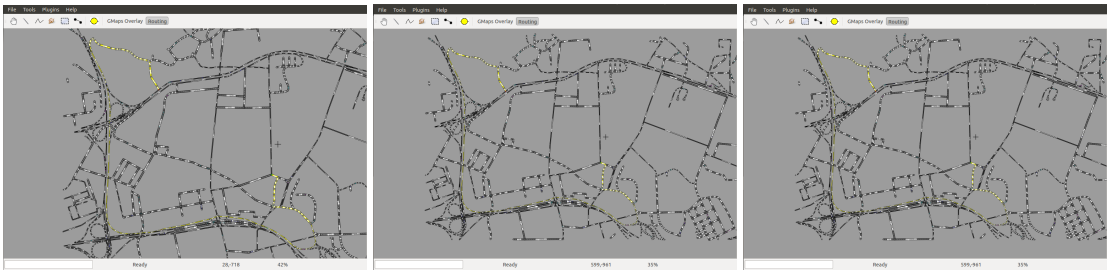
Figura 5.10: Rotas obtidas com 3 abordagens díspares sobre a rede real.

Outros testes mais específicos também foram desenvolvidos, dos quais se destacam dois.



(a) Perfil ECO. (b) Perfil ECO_TUR. (c) Perfil ECO&TUR.

Figura 5.11: Rotas obtidas com 3 perfis diferentes com maior componente ecológica sobre a rede real.



(a) Perfil TUR_ECO. (b) Perfil TUR_CUL. (c) Perfil TUR_OCA.

Figura 5.12: Rotas obtidas com 3 perfis diferentes com maior componente turística sobre a rede real.

O primeiro deles refere-se ao processo de ignorar nós, já detalhado na Secção 4.2.2. Assim, no exemplo da Figura 5.13, a rota que está marcada foi selecionada apenas com a análise de 2 nós (inicial e final) e de um segmento (segmento artificial criado). Isto é bastante vantajoso em situações mais complexas até porque neste caso o tempo de processamento foi de apenas 1 milissegundo.

O último teste relevante prende-se com a análise comparativa entre o algoritmo de *Dijkstra* e duas abordagens puramente ecológicas: usando pré processamento de hierarquia e através da modificação da função de custo do algoritmo de *Dijkstra*.

Assim, o que se verifica é que as duas abordagens ecológicas seguem rotas semelhantes mas bem diferentes da terceira. Isto deve-se às duas curvas bem acentuadas que fazem parte da rota sugerida através do algoritmo de *Dijkstra* tradicional. No entanto, como é possível verificar na Figura 5.14, os dois nós que estão marcados a preto irão ter duas ligações no caso do pré processamento (situação (c)). Isso leva a que o que foi criado artificialmente, e que corresponde a essa parte do percurso na situação (b), seja preterido por ter um comprimento maior. O segmento menor não tinha sido seguido no caso (b) porque o nó a negro superior tem uma curva bastante acentuada e que não vai tanto de encontro ao perfil ecológico como o nó que foi escolhido como sucessor do nó negro que está na zona inferior.

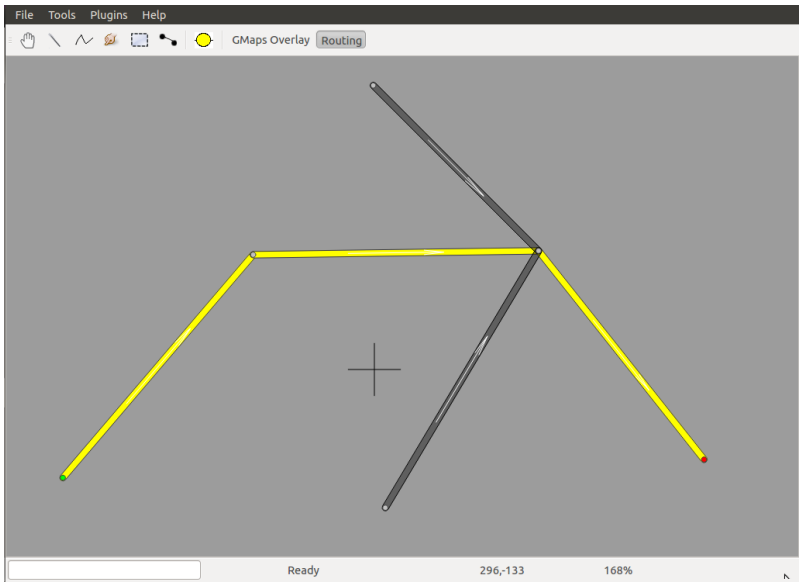
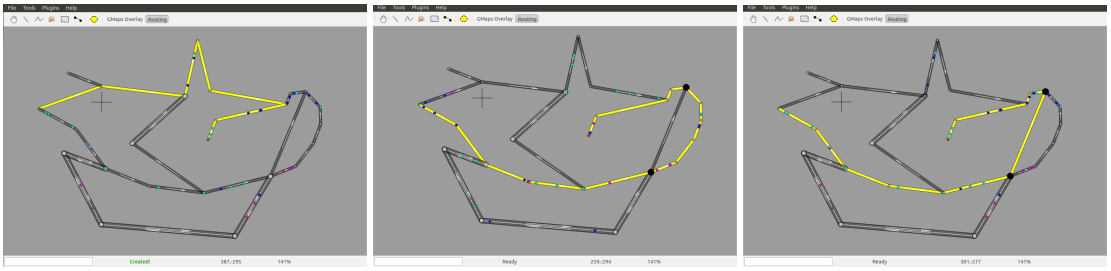


Figura 5.13: Rota calculada após se ignorar nós em pré processamento.



(a) Algoritmo de *Dijkstra* tradicional.

(b) Algoritmo de *Dijkstra* considerando perfil ECO.

(c) Pré-processamento hierárquico.

Figura 5.14: Comparações de duas abordagens puramente ecológicas

5.2 Eficiência das novas abordagens

Um aspeto bastante importante, além da rota que efetivamente é sugerida e que foi analisada anteriormente, está relacionado com a eficiência de cada abordagem.

Como forma de medir esta componente foram considerados os seguintes parâmetros:

- Número de nós analisados
- Número de arestas analisadas
- Distância do percurso
- Tempo previsto do percurso
- Tempo de processamento

Relativamente aos testes efetuados sobre as redes 5.1, 5.2 e 5.3 obtiveram-se os valores apresentados nas tabelas 5.1, 5.2 e 5.3, respetivamente, sendo que estas redes englobavam:

- Rede artificial 1
  - Número de nós = 17
  - Número de arestas = 25
  - Número de pontos de interesse = 35
- Rede artificial 2
  - Número de nós = 25
  - Número de arestas = 33
  - Número de pontos de interesse = 36
- Rede real
  - Número de nós = 1894
  - Número de arestas = 1394
  - Número de pontos de interesse = 180

		Parâmetros				
		Nós	Arestas	Distância(m)	Tempo(min)	Processamento(msec)
Abordagens	Dijkstra	16	42	418.01	0.294	9
	ECO	8	22	418.01	0.294	11
	ECO_TUR	8	22	418.01	0.294	11
	ECO&TUR	13	36	491.062	0.344	17
	TUR_ECO	13	36	491.062	0.344	10
	TUR_CUL	13	36	491.062	0.344	10
	TUR_OCA	13	36	680.275	0.434	16
	Hierarquia	11	26	556.533	0.376	2
	A*	10	28	418.01	0.294	10

Tabela 5.1: Valores dos parâmetros de eficiência na rede artificial 1

Relativamente aos parâmetros apresentados nas tabelas referidas anteriormente, nem todos têm a mesma preponderância. Assim, considerando uma escala de 1 a 5 em que 1 significa pouco importante e 5 muito importante, foram atribuídos os seguintes valores:

- Número de nós: 4
- Número de arestas: 4
- Distância do percurso: 2

		Parâmetros				
		Nós	Arestas	Distância(m)	Tempo(min)	Processamento(msec)
Abordagens	Dijkstra	24	64	352.709	0.269	5
	ECO	15	38	496.674	0.596	15
	ECO_TUR	14	35	496.674	0.596	8
	ECO&TUR	19	50	496.674	0.596	17
	TUR_ECO	19	50	496.674	0.596	16
	TUR_CUL	23	61	513.47	0.353	13
	TUR_OCA	24	64	528.972	0.332	17
	Hierarquia	10	23	413.283	0.402	2
	A*	10	27	352.709	0.269	4

Tabela 5.2: Valores dos parâmetros de eficiência na rede artificial 2

		Parâmetros				
		Nós	Arestas	Distância(m)	Tempo(min)	Processamento(msec)
Abordagens	Dijkstra	820	1331	2820.09	1.410	518
	ECO	646	1115	4087.15	2.044	1330
	ECO_TUR	755	1275	4110.67	2.055	1260
	ECO&TUR	755	1275	4110.67	2.055	1147
	TUR_ECO	755	1275	4110.67	2.055	1239
	TUR_CUL	646	1115	4087.15	2.044	1048
	TUR_OCA	647	1117	4087.15	2.044	1235
	Hierarquia	203	453	9665.39	4.833	199
	A*	704	1164	2820.09	1.410	1167

Tabela 5.3: Valores dos parâmetros de eficiência na rede real da zona do Porto

- Tempo estimado do percurso: 2
- Tempo de processamento: 5

Posto isto, a eficiência de cada abordagem mede-se segundo a equação 5.1

$$Min \quad E = \sum_{i=1}^{4} fe_i * p_i \tag{5.1}$$

Nessa equação, fe_i representa o fator de preponderância e p_i o valor do respetivo parâmetro no teste em questão. Importa também referir a razão de se pretender minimizar essa função. Efetivamente, o que acontece é que todos os parâmetros apresentam valores mais eficientes quanto menor este for. Portanto, no resultado final, as abordagens mais eficientes serão aquelas que apresentarem um valor menor de E.

Os cálculos da eficiência para cada um dos 3 testes encontram-se ilustrados nos gráficos 5.15, 5.16 e 5.17.

Da análise dos gráficos anteriores é possível tecer algumas considerações relevantes, essencialmente se se tiver como comparação a eficiência apresentada pelo algoritmo de *Dijkstra*.

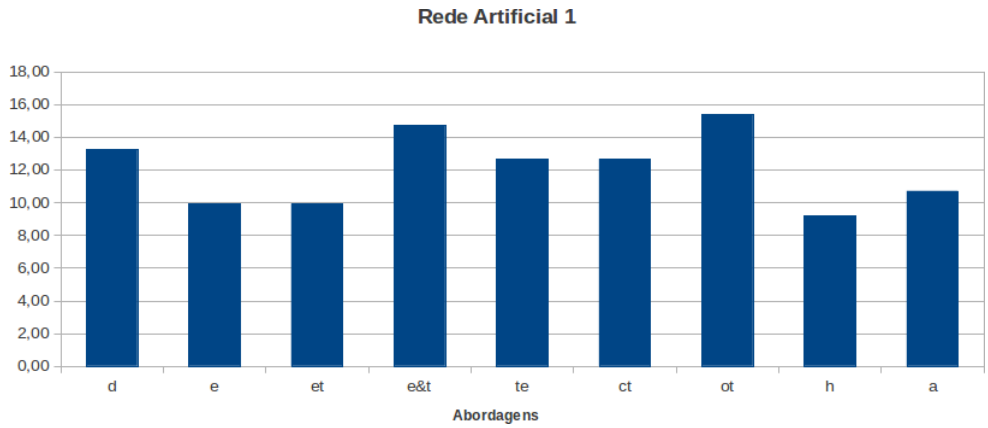


Figura 5.15: Eficiência das várias abordagens nos testes a que foi submetida a rede artificial 1.

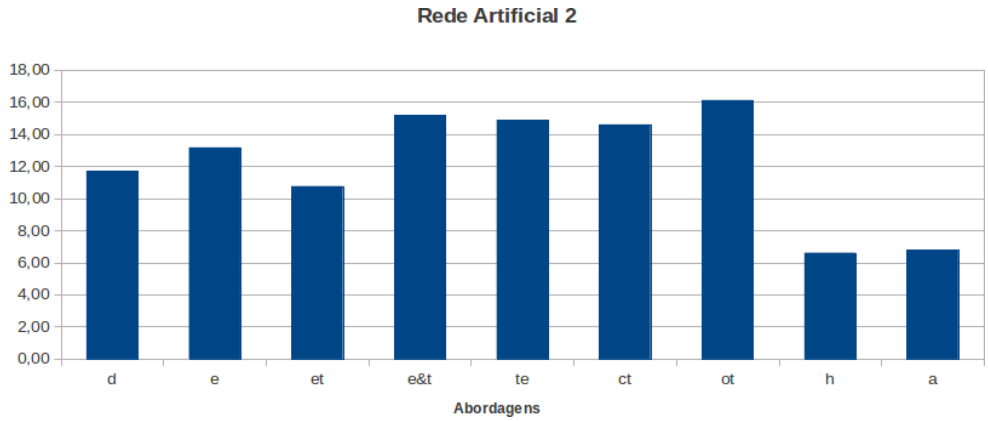


Figura 5.16: Eficiência das várias abordagens nos testes a que foi submetida a rede artificial 2.

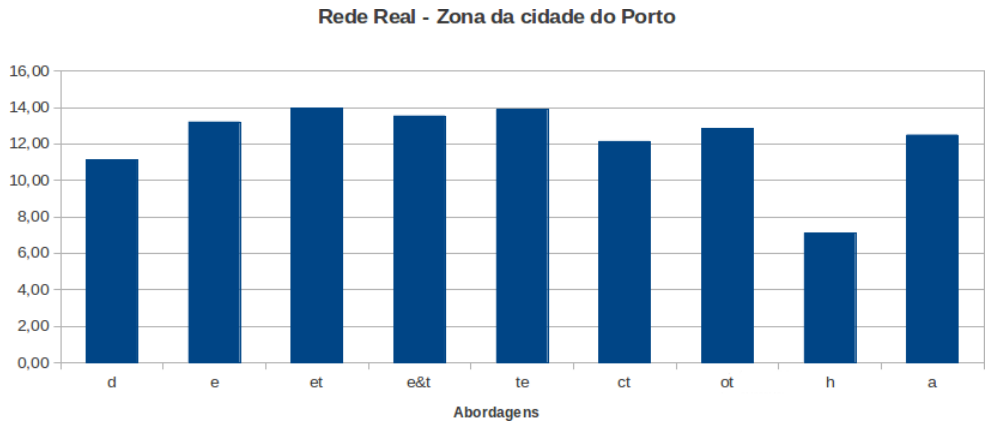


Figura 5.17: Eficiência das várias abordagens nos testes a que foi submetida a rede real de parte da cidade do Porto.

Assim sendo, e relativamente às duas redes artificiais, é percetível que os algoritmos que consideram perfis ecológicos tiveram um desempenho muito melhor no primeiro caso do que

no segundo. Isso deveu-se essencialmente a que na rede artificial 1 a solução apresentada pelo algoritmo de *Dijkstra* já era a mais ecológica pelo que a consideração da distância, embora que com um fator menor, levou a uma diminuição do desempenho por parte destes.

Quanto às abordagens relativas a perfis turísticos apresentam valores semelhantes aos obtidos pelo algoritmo de *Dijkstra*, no primeiro caso, e relativamente piores no segundo. Se se for mais ao detalhe pode verificar-se que, em ambas as situações, estes apresentam um tempo de processamento superior o que acaba por prejudicar a sua eficiência.

Quanto à abordagem baseada em hierarquias e à nova heurística do A*, estas apresentam valores consideravelmente melhores do que todas as restantes abordagens, principalmente devido ao baixo tempo de processamento e ao reduzido número de arestas e nós analisados. Relativamente a este aspeto é importante referir que os valores em termos de consumo de memória (nós e arestas) são totalmente satisfatórios visto que se conseguem valores bastante inferiores ao caso base em 15 dos 16 testes efetuados.

Assim, pode concluir-se que mesmo aquelas abordagens que apresentam um tempo de processamento pior conseguem um consumo de memória bastante eficiente e de acordo com o esperado.

Por outro lado, e analisando agora os dados obtidos para a rede de maior dimensão, é possível verificar que o desempenho das abordagens desenvolvidas foi pior, excetuando no caso da abordagem hierárquica.

Esta abordagem apresentou valores ainda mais satisfatórios do que nos testes anteriores tanto em relação ao tempo de processamento com ao consumo de memória, apesar de sugerir rotas mais longas.

Outra conclusão importante prende-se com o facto de que as restantes abordagens apresentam valores bastante semelhantes na globalidade dos parâmetros. Isto pode indicar que em todas elas a componente da distância assume valores ainda demasiada importância o que quanto maior for a dimensão do grafo mas prejudicial vai ser, tendo em conta a personalização de rotas.

Desta forma, uma alteração dos valores parametrizáveis de forma a dar mais preponderância aos perfis apresenta-se como uma solução viável a este aspeto.

5.3 Sumário

Atendendo aos resultados apresentados anteriormente, e à constatação dos mesmos, naturalmente se chega à conclusão que nem todas as abordagens desenvolvidas se tornam tão propícias a serem aplicadas em todo o tipo de grafos.

Assim, pode-se considerar que as diferenças ao nível das redes existentes se prendem com a sua dimensão, em termos do número de nós e de arestas que as constituem, e com a sua topologia, essencialmente em relação ao grau dos nós, aos ângulos apresentados entre as várias arestas incidentes em cada nó e ao número de pontos de interesse existentes.

Desta forma, relativamente à dimensão, a abordagem baseada no pré-processamento hierárquico é a que apresenta melhores resultados com o aumento do número de nós e arestas. Verificou-se então que nas redes artificiais, de menor dimensão, os valores apresentados para a eficiência,

embora melhores, não demonstravam uma diferença tão clara como aconteceu no caso da rede real usada para os testes apresentados anteriormente.

Isto leva a concluir que este é efetivamente o algoritmo que se aplica melhor a redes de maior dimensão, o que não implica que não tenha um bom desempenho nos grafos com menor número de nós e arestas. O que acontece neste caso é que o comportamento das outras abordagens não é tão pior como o verificado na rede real.

No que diz respeito à topologia do grafo, o abordagem hierárquica é mais vantajosa nas redes em que o pré-processamento possa reduzir a dimensão do grafo e calcular à priori o maior número de troços possível. Assim sendo, as redes em que isto acontece são aquelas com uma maior percentagem de nós com baixo grau de saída. Esta abordagem é também propícia para percursos que envolvam longas distâncias porque geralmente não implicam um número tão elevado de manobras como andar no centro de uma cidade, e assim a hierarquia pré-computada é aproveitado o mais possível.

Quanto aos restantes algoritmos, as abordagens que surgiram da modificação do algoritmo de *Dijkstra* aplicam-se essencialmente a redes com um número considerável de pontos de interesse, ou a percursos em zonas turísticas, visto que foram as que apresentam uma maior capacidade de sugestão de rotas mais personalizadas e adequadas ao perfil do utilizador. Além deste tipo de grafos, funcionam também bastante bem em redes com um variado número de opções semelhantes para chegar ao destino visto a adequação satisfatória que se consegue relativamente aos perfis que consideram com preponderante as manobras e os declives das estradas.

Relativamente à nova heurística apresentada para o algoritmo A* acabou por não ser tão eficiente como a abordagem hierárquica, em redes de grandes dimensões, nem gerar rotas tão personalizadas como o algoritmo de *Dijkstra* modificado, em grafos com topologia propícia. Ainda assim, e visto que apresentou sempre valores interessantes e satisfatórios relativamente ao número de nós e arestas analisados, pode ser interessante de aplicar em redes intermédias em que nenhuma das outras abordagens se revele como irremediavelmente melhor.

Capítulo 6

Conclusões

O presente capítulo contém essencialmente um resumo de tudo o que já foi apresentado e detalhado anteriormente. Assim é possível identificar os pontos essenciais inerentes ao desenvolvimento deste projeto de dissertação.

Isto possibilita também uma clarificação em relação ao estado de concretização e satisfação dos objetivos desta tese e também da consideração de algum trabalho futuro, tanto por não ter sido concluído como por se proporcionar como complemento ao que já foi desenvolvido.

Assim sendo, o problema que desencadeou esta tese e que dizia respeito a considerar múltiplos critérios em algoritmos de escolha de rotas, para sistemas de navegação GPS, foi abordado através da implementação de diferentes abordagens com a pretensão de fazer uma comparação que permitisse obter conclusões à cerca das mais valias de cada uma relativamente às restantes.

Estas abordagens desenvolvidas tiveram como base de comparação o algoritmo de Dijkstra. Além disso, este também foi usado nas novas implementações de formas diferentes. Por um lado foi o algoritmo usado para processar o grafo após uma fase de pré processamento e por outro foi para este algoritmo que foi desenvolvida uma nova função de custo que considerando um número aceitável de critérios.

Posto isto, o pré-processamento fez-se essencialmente ignorando e hierarquizando nós enquanto que a nova função de custo passou a considerar as características de perfis de utilizador que foram criados estaticamente.

Além desta abordagem, uma nova heurística, com especial foco para os pontos de interesse, foi também implementada e integrada com o algoritmo A*.

6.1 Satisfação dos Objetivos

Tendo em consideração os desenvolvimentos conseguidos com esta tese, foram feitas comparações entre as várias abordagens implementadas porque era um dos objetivos e porque possibilitou tirar conclusões relativamente à satisfação dos demais.

Com base nessa análise e nas soluções geradas pode dizer-se que foi conseguida a integração de vários critérios numa mesma função de custo como forma de permitir a escolha de rotas mais específicas e personalizadas.

No entanto, nem todas as abordagens desenvolvidas se revelaram tão eficientes em grafos de grande como de pequena dimensão. Além disso, o desempenho foi variado tendo em consideração as diferentes métricas definidas.

Desta forma, pode considerar-se que os objetivos foram cumpridos com satisfação apesar de que nem todas as abordagens conseguiram dar resposta a todos os objetivos.

Assim, a função de custo para o Dijkstra funcionou extremamente bem para uma pesquisa multicritério e para a sugestão de rotas personalizadas apesar de nem sempre apresentar um bom desempenho.

Por outro lado, a nova heurística implementada para o algoritmo A* (Secção 4.2.3) foi vantajosa essencialmente ao nível da resolução de problemas de memória visto que em todas as situações apresentava sempre valores claramente satisfatórios em termos da quantidade de nós e arestas analisadas apesar de não apresentar rotas muito diferenciadas relativamente ao algoritmo de *Dijkstra*.

Relativamente à escolha de rotas com base na hierarquização é possível afirmar que foi a abordagem que se tornou mais eficiente, independentemente da dimensão do grafo, revelando sempre valores ótimos e com considerável diferença para os restantes.

Com isto, conclui-se que todos os objetivos foram cumpridos com sucesso tendo esta dissertação dado um importante contributo para a área da Inteligência Artificial nomeadamente em relação à otimização de algoritmos com aplicação em problemas de caminho mais curto e análise de grafos, com foco para os sistemas de navegação GPS e orientada a condutores de veículos particulares.

6.2 Trabalho Futuro

Com base no trabalho já desenvolvido até ao momento, existem alguns aspetos que poderiam ser alvo de complemento num futuro mais imediato do que outros.

Desta forma, e para ser desenvolvido num futuro próximo, apresenta-se essencialmente a definição de perfis dinamicamente através do histórico de utilização. Assim poderiam ser analisadas as respostas de cada abordagem desenvolvida com o decorrer do tempo e as alterações que vão surgindo com o aglomerar de utilizações do sistema. Além de isso poderia haver uma "correlação" entre utilizadores através de técnicas de *clustering* e da sugestão de rotas. Por outro lado, e visto que houve um número ainda considerável de fatores parametrizáveis e cujas alterações tinham repercussões importantes nas rotas geradas, uma análise minuciosa de cada uma dessas variáveis poderia ser interessante de modo a conseguir-se otimizar ao máximo cada algoritmo desenvolvido.

A médio prazo seria interessante aplicar as abordagens desenvolvidas a outro tipos de utilizadores finais, que não condutores de carro particular. Seria então interessante passar a considerar

transportes públicos, não só por terem restrições bem mais específicas mas porque servem um quantidade e diversidade de pessoas bastante grande.

Como trabalho futuro a considerar mais a longo prazo, seria também importante dar especial atenção ao desenvolvimento de um interface mais interativa e adequada à personalização de rotas conseguida com estas abordagens. Isto poderia passar por permitir ao utilizador tomar opções em cada momento, como escolher que uma rota seja mais adequada ao seu perfil ou que noutra este aspeto tenha um peso menor, ou que permitisse ter um acompanhamento das rotas anteriormente efetuada e ter possibilidade de definir as que teriam maior ou menor interesse para considerações futuras.

Referências

[Atl05] Tele Atlas MultiNet^{TM} Shapefile 4.3.1 Format Specifications. (May), 2005.

[Ber02] Pavel Berkhin. Survey of Clustering Data Mining Techniques. *Techniques*, 10(c):1–56, 2002.

[CCP03] JCN Climaco, JMF Craveirinha e Marta M. B. Pascoal. A bicriterion approach for routing problems in multimedia networks. *Networks*, 41(4):206–220, 2003.

[Cha98] I. Chabini. Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Record: Journal of the Transportation Research Board*, 1645:170–175, 1998.

[CHW11] Rutger Claes, Tom Holvoet e Danny Weyns. A Decentralized Approach for Anticipatory Vehicle Routing Using Delegate Multiagent Systems. *IEEE Transactions on Intelligent Transportation Systems*, 12(2):364–373, June 2011.

[CL02] I. Chabini e Shan Lan. Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks. *IEEE Transactions on Intelligent Transportation Systems*, 3(1):60–74, March 2002.

[Dij59] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, December 1959.

[DW07] Daniel Delling e Dorothea Wagner. Landmark-based routing in dynamic graphs. *Experimental Algorithms*, 2:52–65, 2007.

[EAS11] S.A. Elavarasi, J. Akilandeswari e B. Sathiyabhama. A SURVEY ON PARTITION CLUSTERING ALGORITHMS. *learning*, 1(1):1–14, 2011.

[ESK03] Levent Ertöz, Michael Steinbach e Vipin Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *SIAM international conference on data mining*, volume 47, 2003.

[Fre09] Tiago Freitas. *Geospatial Data Processing for GPS Navigation Systems*. PhD thesis, FEUP: Faculdade de Engenharia da Universidade do Porto, 2009.

[FSR06] L Fu, D Sun e L Rilett. Heuristic shortest path algorithms for transportation applications: State of the art. *Computers & Operations Research*, 33(11):3324–3343, November 2006.

[Gon10] Songjie Gong. A Collaborative Filtering Recommendation Algorithm Based on User Clustering and Item Clustering. *Journal of Software*, 5(7):745–752, 2010.

[HK06] J Han e M Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2006.

[HNB68] PE Hart, NJ Nilsson e Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions of Systems Science and Cybernetics*, ssc-4:100–107, 1968.

[HWZ07] B. Huang, Q. Wu e F. B. Zhan. A shortest path algorithm with novel heuristics for dynamic transportation networks. *International Journal of Geographical Information Science*, 21(6):625–644, July 2007.

[JMF99] AK Jain, MN Murty e PJ Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3), 1999.

[KLF04] S Koenig, Maxim Likhacev e David Furcy. Lifelong Planning A*. *Artificial Intelligence*, 155(1-2):93–146, May 2004.

[LLHH05] Hongga Li, Hua Lu, Bo Huang e Zhiyong Huang. Two ellipse-based pruning methods for group nearest neighbor queries. *Proceedings of the 13th annual ACM*, 2005.

[MGT06] Behnam Malakooti, Sanjaya Gajurel e Siva Tanguturi. MCLSR: Multiple Criteria Link State Routing. *vorlon.case.edu*, 2006, 2006.

[NL08] Giacomo Nannicini e Leo Liberti. Shortest paths on dynamic graphs. *International Transactions in Operational Research*, 15(5):551–563, 2008.

[Ram11] Jorge Alpedrinha Ramos. *Solving the Extended Vehicle Scheduling Problem using MetaHeuristics*. PhD thesis, 2011.

[RBD02] Abhishek Roy, Nilanjan Banerjee e Sajal K Das. An Efficient Multi-Objective QoS Routing Algorithm for Real-Time Wireless Multicasting. *IEEE Semiannual Vehicular Technology Conference*, 3:1160–1164, 2002.

[Sch08] Dominik Schultes. Route planning in road networks. 2008.

[SCZ98] Gholamhosein Sheikholeslami, S. Chatterjee e A. Zhang. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In *Proceedings of the International Conference on Very Large Data Bases*, pages 428–439. INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS, 1998.

[SMA08] Jorge M Santos, Joaquim Marques de Sa e Luis a Alexandre. LEGClust- a clustering algorithm based on layered entropic subgraphs. *IEEE transactions on pattern analysis and machine intelligence*, 30(1):62–75, January 2008.

[VM03] D Verma e M Meila. A comparison of spectral clustering algorithms. Technical report, University of Washington, 2003.

[WHH07] Danny Weyns, Tom Holvoet e Alexander Helleboogh. Anticipatory vehicle routing using delegate multi-agent systems. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 87–93. IEEE, 2007.

[XW05] Rui Xu e Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.

[Yua03] D. Yuanroshi. A Bi-Criteria Optimization Approach for Robust OSPF Routing. *IEEE Workshop on IP*, 2003.

[ZON] Liang Zhao, Tatsuya Ohshima e Hiroshi Nagamochi. A * Algorithm for the time-dependent shortest path. *Science*.