FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Data mining de dados geo-temporais para suporte à mobilidade

Tiago Silva

Mestrado Integrado em Engenharia Informática e Computação

Orientadora: Ana Aguiar (Prof. Doutora)

Co-orientadora: Eduarda Mendes Rodrigues (Prof. Doutora)

09 de Julho de 2012

Data mining de dados geo-temporais para suporte à mobilidade

Tiago Silva

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: João Mendes Moreira (Professor Auxiliar)

Arguente: Miguel Rocha (Professor Auxiliar)

Orientador: Ana Aguiar (Profesora Auxiliar Convidada)

09 de Julho de 2012

Resumo

Estima-se que o transporte pessoal seja das atividades que mais contribui com emissões de CO_2 para o ambiente. Segundo estudos já efetuados, uma grande parte das pessoas considera que o tempo para chegar ao destino é claramente o fator que mais pesa aquando da escolha do meio de transporte a utilizar.

É devido a isso, mas não só, que os transportes públicos raramente têm a recetividade ideal por parte da comunidade. A solução passa então pela partilha de veículo, mantendo assim todo o conforto do veículo próprio e não perdendo a capacidade de se deslocar em tempo útil.

O desenvolvimento de técnicas e algoritmos para suporte à mobilidade é um campo emergente mas está ainda a dar os primeiros passos. Por outro lado, os avanços proporcionados pela comunidade científica não têm ido no sentido do auxílio à partilha de veículo.

Nesta dissertação faz-se uso de *smartphones*, e da ubiquidade de que se fazem valer atualmente, para monitorizar as viagens dos utilizadores, recolhendo coordenadas geográficas através do sensor de GPS. Esta recolha desenrola-se segundo uma perspetiva de *participatory sensing* para uma recolha de dados maciça e que possibilite a extração de conhecimento útil a partir deles. Assim, são recolhidos dados de vários utilizadores que em conjunto servem para detetar possíveis sugestões de partilha de veículo por parte de 2 ou mais utilizadores.

Os dados recolhidos são um misto de dados geográficos e temporais, que originaram, inclusive, o título da dissertação: *Data mining de dados geo-temporais para suporte à mobilidade*, onde são aplicadas técnicas e algoritmos de *data mining* de forma a obter diferentes níveis de conhecimento até ser possível a procura de sugestões úteis e viáveis de partilha de veículo. Estas técnicas e algoritmos consistem não só na modificação e/ou junção de vários algoritmos já existentes na literatura mas também na criação de um novo algoritmo para descoberta de determinadas formas em conjuntos de dados espaciais em larga escala que sejam constituídos por dois tipos de dados diferentes.

Os resultados alcançados são bastante positivos sendo o sistema já capaz de fazer sugestões completamente automáticas na maior parte das situações em que é suposto fazê-lo. Estes resultados, aliados ao enorme potencial deste tipo de aplicações, auguram um futuro auspicioso nesta área.

A presente dissertação tenta tirar partido do notório fosso que existe atualmente em termos de aplicações (semi-)automáticas para *carpooling* visto não ser conhecido nenhum sistema, até ao momento, que seja capaz de fazer sugestões de partilha de veículo de forma automática.

Abstract

It is estimated that personal transportation is one of the activities which most contributes to the CO_2 emissions (carbon footprint). Based on past studies, a large number of people consider that the time spent to get to a place is clearly the most important factor when they have to choose a means of transportation. Due to that, and other factor, public transports are well received by the community.

A possible solution is ride sharing, which keeps all the comfort of the own vehicle and does not lose the capacity to move to anywhere in a good time.

The development of mobility support techniques is still an emergent field and it is still taking first steps. On the other hand, the advances provided by the scientific community have not been helpful in the case of ride sharing.

In this work smartphones are used, and the ubiquity which is characteristic to them, for monitoring the users'trips.

The monitoring is made through the GPS sensor which collects the geographic coordinates. This massive data collection is based on a participatory sensing perspective and the key idea is to enable an extraction of useful knowledge. In other words, data is collected from several users which in conjunction will be useful to detect shared routes by two or more users.

The data collected is a mix of geographic and temporal data, which have caused the title of the MSc thesis: *Data mining of geo-temporal data for mobility support*, where are applied algorithms and techniques of data mining to extract different levels of knowledge until be possible to seek useful and feasible suggestions of ride sharing. These algorithms and techniques are not only combinations and/or small modifications of algorithms which are already presented on the literature but also include the development of a new algorithm for discovering determined shapes in large sets of spatial data when the data set is composed by two different types of data.

The results have proven to be positive and the system is now able to make suggestions automatically in most situations where it is supposed to do so. These results, combined with the huge potential of this kind of applications, portend a successful future in this area.

This work tries to take advantage of the huge ditch in (semi-)automatic applications for carpooling as it is not known, until now, any system capable to make automatic suggestions of ride sharing.

Agradecimentos

Sem a colaboração de algumas pessoas não teria sido possível chegar a bom porto nesta dissertação e aproveito, por isso, para deixar aqui algumas palavras de apreço.

Em primeiro lugar, não podia deixar de expressar o meu mais sincero agradecimento às orientadoras, Prof. Doutora Ana Aguiar e Prof. Doutora Eduarda Mendes Rodrigues, pelo acompanhamento, pela disponibilidade, pelo empenho para que a dissertação fosse concluída com sucesso e pelos conhecimentos transmitidos ao longo de todo este tempo e que fizeram com que o resultado final fosse bastante mais rico.

Aproveito a oportunidade para agradecer também ao Vítor Ribeiro e ao João Rodrigues do Instituto de Telecomunicações do Porto pela disponibilidade para efetuarem modificações na aplicação de recolha de dados sempre que necessário.

Como não podia deixar de ser, quero agradecer a todos as pessoas que participaram na recolha de dados, e que assim tornaram possível o desenvolvimento desta dissertação, principalmente aos que fizeram parte da amostra que levou a cabo as primeiras recolhas de dados e que serviu também para melhorar a aplicação de recolha de dados.

Por fim, mas com a mesma importância que os anteriores, devo também a minha gratidão à minha namorada, Vanessa Correia, pela força que sempre me transmitiu e pelo seu incondicional apoio.

Tiago Silva

"O cientista descobre o que existe
enquanto o engenheiro cria o que nunca existiu"

Theodore von Kármán

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Abreviaturas e Símbolos

FAQ	*Frequently Asked Questions*
FEUP	Faculdade de Engenharia da Universidade do Porto
GPS	*Global Positioning System*
GIS	*Geographic Information System*
HCI	*Human-Computer Interaction*
KNN	*K-Nearest Neighbor*
MBR	*Minimum Bounding Rectangle*
MIT	*Massachusetts Institute of Technology*
SGBD	Sistema de Gestão de Base de Dados
SQL	*Structured Query Language*
UML	*Unified Modeling Language*
XML	*Extensible Markup Language*

Capítulo 1

Introdução

Em 2005, os cidadão Norte-americanos consumiram mil biliões de BTUs (British Thermal Units)[1] de energia, praticamente 6 vezes mais do que a média mundial por pessoa. Foram libertados cerca de 2.2 mil milhões de toneladas de dióxido de carbono, o maior causador das mudanças climatéricas no nosso planeta [FDK+09]. Em termos individuais, o transporte pessoal, em média, é o que mais contribui com emissões de CO_2 na América [WM08].

Estes são dados preocupantes e para reverter esta tendência são necessárias ações a vários níveis. A tecnologia já começou a assumir um papel significativo no suporte a comportamentos ambientais positivos e a área de suporte à mobilidade não é exceção. No entanto, existe ainda uma margem de progressão enorme e que deve ser explorada tão breve quanto possível.

1.1 Contexto e Enquadramento

A área *participatory sensing*, embora recente, começa a despoletar grande atratividade na comunidade científica. A existência de redes deste tipo permite que os utilizadores recolham, analisem e partilhem conhecimento local e é uma forma cada vez mais comum de ajudar no que diz respeito à mobilidade em áreas urbanas. Por seu turno, *collaborative sensing* é outro campo onde existe cada vez mais exploração e que possibilita também algumas abordagens bastante interessantes. *Collaborative sensing* está mais relacionado com a partilha de informação local entre dispositivos de forma a que ambos tenham acesso a um conjunto de informações que sozinhos não conseguiriam ter, por exemplo, um dispositivo que tem acesso à temperatura local partilhá-la com outro através de uma ligação por Bluetooth.

A ubiquidade dos dispositivos móveis, aliada a estes dois conceitos fundamentais, torna-os uma excelente plataforma para a conceção de soluções nesta área. Marc Weiser, considerado o pai da computação ubíqua, visualizou-a como um mundo onde a computação e a comunicação estariam convenientemente "à mão" e distribuídas por todo o nosso ambiente [Wei99]. Como o

[1] Unidade tradicional de energia equivalente a 1055 Joules

transporte/deslocação das pessoas é inerentemente uma atividade móvel, os dispositivos móveis são apropriados para recolher informação sobre essas atividades [FDK+09].

Hoje em dia é notória a necessidade de melhorar, de acrescentar valor, ao suporte à mobilidade. Com recurso ao uso da capacidade sensorial que os dispositivos móveis atuais apresentam, nomeadamente recolhendo dados contextuais como a posição geográfica do utilizador e analisando e explorando esses dados é possível a extração de conhecimento útil.

O âmbito desta dissertação insere-se essencialmente num contexto citadino onde uma grande parte das pessoas têm padrões de mobilidade bem definidos. De acordo com os resultados obtidos no inquérito levado a cabo (ver o Anexo D para mais detalhes sobre o inquérito) apenas 10% dos inquiridos afirmam não ter nenhuma viagem de rotina.

1.2 Motivação e Objetivos

A alternativa mais natural para as viagens que são feitas no carro particular é o uso de transportes públicos. Todavia os transportes públicos apresentam frequentemente uma enorme falta de atratividade. Não só devido ao menor conforto que apresentam mas também devido ao barulho típico de locais onde estão várias pessoas. As deslocações de e para os pontos de acesso a esses transportes constituem também um dos maiores inconvenientes ao seu uso, principalmente quando as condições climatéricas são extremas. Existe ainda a ineficiência que os transportes públicos frequentemente apresentam. Os tempos médios de viagem são tipicamente superiores aos de um veículo próprio e é também necessário contemplar os tempos de espera que com o veículo próprio não existem.

A juntar ao já enunciado, trabalhos anteriores já mostraram que fatores de motivação como a receção de *feedback* frequente e personalizado e o assumir de compromissos públicos têm um impacto positivo no comportamento ambiental das pessoas [ASVR05]. Numa revisão de mais de 20 estudos que exploraram os efeitos da receção de *feedback* nos padrões de consumo de energia em casa, concluiu-se que tipicamente a energia poupada está entre os 5% e os 12% [Fis08].

A partilha de veículo nas viagens de rotina surge assim como uma hipótese bastante aprazível. De acordo com o inquérito levado a cabo no âmbito desta dissertação (apresentado na Secção 6.3), 65% dos inquiridos estariam dispostos a participar num programa de recomendações automáticas de partilha de veículo se o sistema tivesse um mecanismo de avaliação dos utilizadores (de forma a garantir segurança a todos os utilizadores envolvidos). Para além disso, apenas 10% dos inquiridos consideram que não têm uma única viagem de rotina, o que faz com que a possibilidade de serem encontradas pelo menos duas viagens de rotina onde seja possível a partilha de veículo seja alta.

São fatores como os apresentados que constituem a principal motivação para o trabalho desta dissertação.

O objetivo principal desta dissertação consiste na investigação e implementação de técnicas de extração de conhecimento útil, a partir dos dados recolhidos pelos utilizadores através do sensor GPS dos seus *smartphones*, segundo uma perspetiva de *participatory sensing*, de forma a permitir

a determinação de sugestões de partilha de veículo úteis e viáveis. O objetivo principal acaba por ser a junção de vários objetivos mais pequenos, que de uma forma sumária se apresentam:

- Determinação dos locais de e para onde cada utilizador viaja frequentemente
- Determinação dos padrões de viagem de cada utilizador
- Determinação de padrões de viagem com rotas total ou parcialmente sobrepostas de forma a propor a sugestão de partilha de veículo nessas situações (6 tipos de sugestões detetáveis, detalhadas no Capítulo 4, Figuras 4.8 e 4.9)

Todos os objetivos aqui definidos deixam de fazer sentido se o sistema não os conseguir cumprir em tempo útil, isto é, não interessa apenas conseguir fazer as sugestões (por muito boas que sejam) se o sistema demorar demasiado tempo a determiná-las. Daqui resulta outro objetivo: escalabilidade. É expectável que um sistema como este possa chegar a ter um uso constante em termos de utilizadores na ordem dos largos milhares.

A sugestão automática de partilha de veículo é um ponto fulcral e que hoje em dia está ainda pouco explorado. É este o mote para a aplicação e desenvolvimento de técnicas sobre os dados geo-temporais recolhidos.

1.3 Descrição do Problema

A ideia explorada nesta dissertação consiste na determinação de sugestões de partilha de veículo com uma análise aprofundada dos dados recolhidos através dos *smartphones* de cada utilizador segundo uma perspetiva de *participatory sensing*.

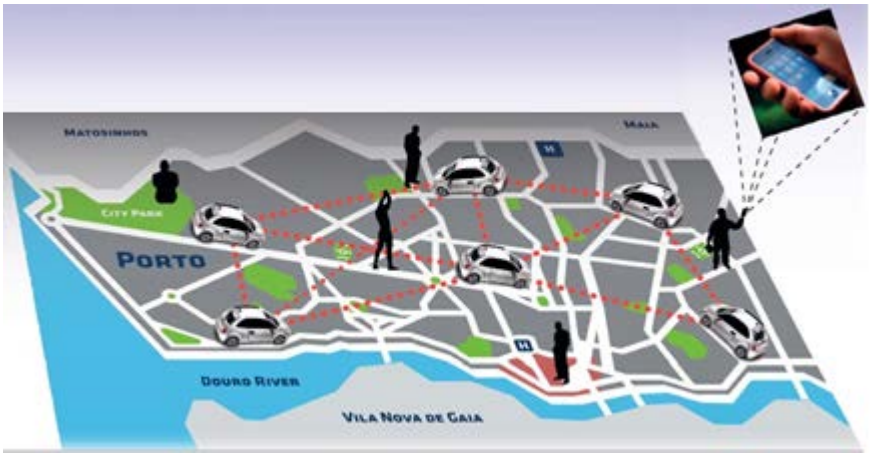


Figura 1.1: Processo de recolha de dados (adaptado de [RAV+11])

Na prática, cada utilizador recorre ao seu *smartphone* para recolher informações contextuais do ambiente onde está. Neste caso em concreto, a única atividade sensorial presente no *smartphone* é levada a cabo pelo sensor GPS que recolhe, a um ritmo predefinido, as coordenadas geográficas

por onde o utilizador viaja. Essas coordenadas geográficas são enviadas para o servidor, em tempo real ou não, e possibilitam a criação de um enorme conjunto de dados.

Sobre o conjunto de dados em bruto criado são aplicadas técnicas e algoritmos de extração de conhecimento de forma a criar várias camadas de conhecimento que possibilitem, no final, a identificação de sugestões para a partilha de veículo por parte de dois ou mais utilizadores.

1.4 Solução Apresentada e Contribuições Científicas

A solução apresentada é constituída por soluções para vários sub-problemas. Embora a solução e a sua conceção seja detalhada ao pormenor nas secções seguintes, interessa nesta altura definir alguns conceitos que são repetidamente mencionados nessas mesmas secções.

Inicialmente surgiu a necessidade da determinação dos pontos de estadia, ou *stay points*, de cada utilizador. Por pontos de estadia entende-se os locais onde cada utilizador em cada viagem permanece mais do que um certo período de tempo.

Foram feitos agrupamentos (*clusters*) dos pontos de estadia de cada utilizador para se perceber em que locais cada utilizador para/permaneçe frequentemente durante as suas viagens. A formação destes *clusters* está condicionada à existência de um número mínimo de pontos de estadia, número esse que está diretamente relacionado com a janela temporal em análise, ou seja, se se está a analisar os dados das últimas 8 semanas, o número mínimo é obrigatoriamente maior do que se se estiver a analisar os dados das últimas 4 semanas, por exemplo. A estes locais chamou-se locais significativos. Por outras palavras, um local significativo é um local onde o utilizador chega e/ou parte vezes suficientes, durante o período de tempo em análise, para se poder dizer que é possível que haja um padrão de viagem com partida e/ou chegada naquele local.

Já depois de se conhecerem os locais significativos de cada utilizador e sabendo, pela própria definição do termo no âmbito desta dissertação, que qualquer viagem frequente do utilizador tem como partida e chegada dois locais significativos do utilizador, são então procurados os padrões de viagem de cada utilizador. Na prática, são exploradas as viagens de cada utilizador que têm como partida e chegada dois dos locais significativos do próprio utilizador. Se o montante de viagens com essa origem e esse destino for superior a um *threshold* que é dependente do tempo a que os dados em análise respeitam, então a viagem é considerada um padrão de viagem, ou seja, uma viagem de rotina daquele utilizador.

Depois da junção destes vários níveis de conhecimento sobre os dados em bruto originalmente recolhidos pelos diversos utilizadores, passam a existir elementos suficientes para se poder proceder à procura de possíveis sugestões de partilha de veículo. Surge então a necessidade de comparar as rotas dos padrões de viagem propriamente ditas. O problema da comparação das rotas está na enorme quantidade de dados inerente ao próprio problema e que faz com que a análise "coordenada a coordenada" seja inviável do ponto de vista temporal. A solução apresentada seguiu então pelo caminho da compactação da informação, perdendo detalhe mas que permite a procura de soluções em tempo útil. Assim, foram determinados os padrões de viagem vizinhos, isto é, foram

determinados os pares de padrões de viagem, que por uma razão ou por outra, devem ver as suas rotas comparadas pois é possível que haja uma sugestão de partilha de veículo.

A solução apresentada foca-se na deteção de padrões de viagem diários e semanais, ou seja, são detetados padrões de viagem com viagens todos os dias ou todas as segundas-feiras, por exemplo. Padrões quinzenais ou mensais não são detetados. A ideia subjacente às sugestões de partilha de veículo passa pela partilha nas viagens de rotina, nomeadamente idas para o trabalho/faculdade/etc, ou seja, viagens que são feitas com uma certa regularidade e com "muita" frequência. É neste tipo de viagens que pode ser criado um maior impacto na sociedade pois são as viagens que praticamente toda a comunidade realiza.

Nesta dissertação a investigação teve um papel preponderante até porque, até à data, não é conhecido nenhum sistema de recomendação automática de partilha de veículo. Embora o uso de dados espaciais, mais concretamente dados geográficos, em conjunto com fatores temporais já tenham sido usados em algumas investigações, não é conhecida nenhuma que tenha como fim o suporte à mobilidade. Os algoritmos e técnicas aqui descritas, sempre com vista à determinação de sugestões úteis e racionais de partilha de veículo, representam um contributo para a comunidade científica na área de suporte à mobilidade.

Foi ainda desenhado e implementado um algoritmo, algoritmo ShapeDetector, que constitui a principal contribuição pessoal, que serve para a determinação de determinadas formas em conjuntos de dados enormes e que sejam constituídos por dois tipos de dados diferentes.

1.5 Estrutura da Dissertação

Na "Revisão Bibliográfica", Capítulo 2, é descrito o estado da arte e algumas das importantes contribuições existentes para a comunidade científica na área de dados espaciais (geográficos), *data mining*, *participatory* e *collaborative sensing* e suporte à mobilidade.

Nos capítulos "Determinação de Locais Significativos e de Padrões de Viagem" e "Determinação de Sugestões de Partilha de Veículo", Capítulos 3 e 4, são dadas a conhecer as técnicas e os algoritmos para determinar os padrões de viagem e para comparar as rotas dos mesmos de forma a descortinar possibilidades de sugestão de partilha de veículo. Nestas duas secções é dissecada a forma como foi idealizada a solução justificando as opções tomadas sempre que necessário.

Já no Capítulo 5, "Implementação", são detalhados os pormenores técnicos que fizeram com que a implementação da solução concebida nas duas secções anteriores fosse possível.

O capítulo "Avaliação dos Algoritmos", Capítulo 6, faz jus ao próprio nome e apresenta os resultados a que se chegou com a solução e a implementação descritas nas secções anteriores. Só com a avaliação dos resultados se percebe se o que foi idealizado, e posteriormente implementado, atinge aos objetivos estabelecidos ou não.

Por fim, no Capítulo 7, "Conclusão", é apresentado o sumário das conclusões retiradas ao longo da dissertação e são identificadas oportunidades para investigação futura.

Capítulo 2

Revisão Bibliográfica

Este capítulo é dedicado à revisão da literatura, ou seja, à análise crítica, meticulosa e ampla da literatura consultada no âmbito dos diferentes temas abordados na presente dissertação. O estado da arte é o ponto de partida para o desenvolvimento e aplicação de técnicas e algoritmos de *data mining* em dados geo-temporais com vista à recomendação de partilha de veículo e tem, por isso, um papel fundamental.

2.1 Captura de Localização

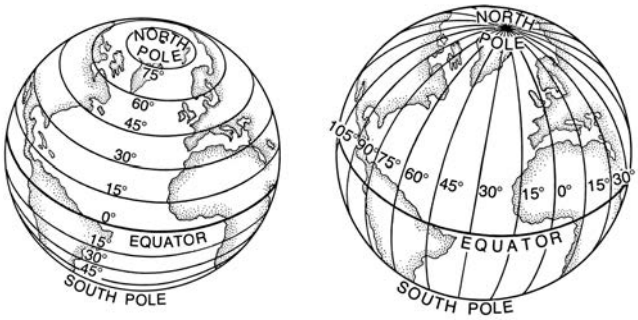
A captura da localização física de um objeto é feita com base na sua localização espacial que é, por sua vez, expressa pelo sistema de coordenadas geográficas. Através das coordenadas geográficas é possível expressar qualquer posição horizontal no planeta através de duas das três coordenadas existentes num sistema de coordenadas esférico, alinhadas com o eixo de rotação da Terra.

2.1.1 Sistema de Coordenadas Geográficas

O sistema de coordenadas geográficas faz uso das coordenadas latitude e longitude. Estas coordenadas podem ser expressas de várias formas, nomeadamente:

- Graus - Minutos - Segundos, onde cada grau é dividido em 60 minutos, que por sua vez se subdividem, cada um, em 60 segundos. Exemplo: 22º 54’ 21.64"S 47º 03’ 38.06"W
- Graus - Minutos decimais, onde cada grau é dividido em 60 minutos, que por sua vez são divididos decimalmente. Exemplo: 22º 54.361’ S 47º 3.634’ W
- Graus decimais, onde a latitude recebe a abreviatura *lat* e a longitude *long*; os valores positivos são para Norte (latitude) e Este (longitude) e os valores negativos são para Sul (latitude) e Oeste (longitude). Exemplo: lat -22.906014º lon -47.060571º

- Universal Transversa de Mercator, onde são usados três dados em vez de dois: o setor do globo terrestre, a distância relativa ao centro do meridiano e a distância ao Pólo Sul (para lugares no Hemisfério Sul) ou à Linha do Equador (para lugares no Hemisfério Norte). Exemplo: 23K 288651.66m E 7465404.76m S



(a) Latitude (b) Longitude

Figura 2.1: Paralelos e meridianos que representam as latitudes e longitudes, respetivamente, em graus (http://techpanacea.blogspot.com/ e http://pt.wikipedia.org/wiki/Longitude)

Nesta dissertação as coordenadas geográficas serão sempre expressas na forma de graus decimais na medida em que esta é a representação mais comum. Para além disso, essa é também a forma como as coordenadas recolhidas pela aplicação *MyDrivingDroid* são guardadas na base de dados. A aplicação MyDrivingDroid, descrita na Secção 2.5.1.1, é a aplicação que está a ser usada para a recolha de dados.

A latitude geográfica de um ponto na superfície da Terra, Figura 2.1a, equivale ao ângulo entre o plano equatorial e uma linha que passa por esse ponto e é normal à superfície de referência que aproxima a forma da Terra. A latitude mede-se para Norte e para Sul do Equador, entre -90º no Pólo Sul e +90º no Pólo Norte. A longitude, Figura 2.1b, descreve a localização de um lugar medido em graus, de 0º a -180º para Oeste ou a +180º para Este, a partir do Meridiano de Greenwich. Portanto, se se combinar estes dois ângulos, latitude e longitude, poderá ser indicada qualquer localização na superfície terrestre. Por exemplo, a cidade do Porto tem uma latitude de +41.15393º e uma longitude de -8.611565º. Significa isto que se se traçar um vetor desde o centro da Terra até um ponto a 41.15393º acima de Equador e 8.611565º a Oeste de Greenwich, irá passar pela cidade do Porto. As linhas traçadas de Oeste a Este têm valor constante de latitude e são chamadas de paralelos, enquanto os meridianos são as linhas que vão de Norte a Sul. Os paralelos e os meridianos ficam dispostos pela superfície do planeta Terra tal como se pode ver nas Figuras 2.1a e 2.1b, respetivamente.

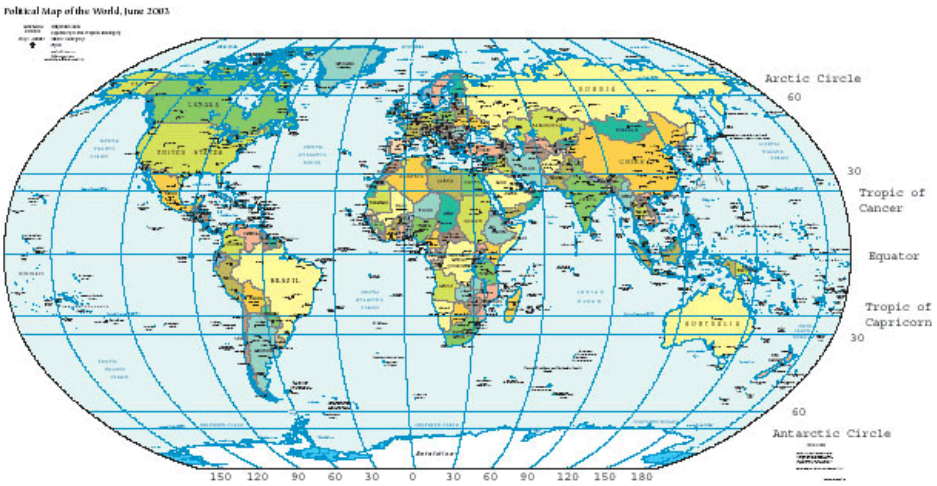


Figura 2.2: Mapa mundo (http://es.wikipedia.org/wiki/Latitud)

O Equador é o único paralelo tão largo quanto os meridianos. Estes círculos, o Equador e todos os meridianos cujo raio é o raio da Terra, são chamados de *grandes círculos*. É por isso que apenas ao longo da linha do Equador, a distância representada por um grau de longitude, se aproxima à representada por um grau de latitude. Acima e abaixo do Equador, os círculos que definem os paralelos de latitude vão ficando gradualmente mais pequenos, até se tornarem um único ponto nos Pólos Norte e Sul, onde os meridianos convergem (este fenómeno é visível na Figura 2.1). À medida que os meridianos convergem para os Pólos, a distância representada por um grau de longitude diminui até zero.

Como os paralelos e os meridianos não têm o mesmo tamanho, a distância representada por um grau de longitude é diferente da distância representada por um grau de latitude (exceto ao longo da linha do Equador que pertence aos *grandes círculos*). Com base na esferóide Clark 1866, a esferóide de referência para fazer o mapa do Norte da América, 1^o de longitude ao nível do Equador é igual a 111.321km, enquanto a 60^o de latitude são 55.802km. Devido a este facto, a distância entre pontos não pode ser medida com precisão utilizando unidades de medida angulares [JGP+03].

Casas decimais	Distância
0	111km
1	11.1km
2	1.11km
3	111m
4	11.1m
5	1.11m
6	111cm
7	11.1cm

Tabela 2.1: Precisão ao nível do Equador consoante os graus decimais das coordenadas

O número de casas decimais necessário para uma precisão ao nível do Equador está de acordo com a Tabela 2.1. Quanto mais perto dos Pólos, maior a precisão. Isto acontece com os graus de longitude porque os paralelos não são todos do mesmo tamanho. Os meridianos como têm todos a mesma dimensão, idêntica à da linha do Equador, a precisão dos graus de latitude está de acordo com a Tabela 2.1 para todos os pontos da superfície terrestre, assumindo a Terra como uma esfera.

2.1.1.1 Cálculo da Distância Entre Duas Coordenadas Geográficas

O facto da latitude e da longitude não terem um tamanho uniforme faz com que a distância entre quaisquer 2 pontos não possa ser medida com precisão recorrendo a unidades de medida angulares. Em [Sub75], Vincenty apresenta uma fórmula baseada no sistema de coordenadas geográficas definidas por uma esferóide. Até ao momento, esta é a fórmula que mais precisão garante, 0.5mm, para se calcular a distância entre 2 pontos à superfície terrestre. Contudo, o seu peso computacional é enorme e significativamente superior ao da fórmula de Haversine baseada na ortodromia, que assume a Terra como esférica. Esta fórmula que garante uma precisão de 0.3% é bastante menos complexa e computacionalmente menos dispendiosa[1].

Fórmula de Haversine A fórmula de Haversine é definida pela função

$$d = 2\arcsin\left(\sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) + \cos\phi_i \cos\phi_f \sin^2\left(\frac{\Delta\lambda}{2}\right)}\right) \tag{2.1}$$

onde ϕ_i, λ_i, ϕ_f e λ_f representam as coordenadas, latitude e longitude, dos pontos inicial e final, respetivamente, e $\Delta\phi$ e $\Delta\lambda$ representam as diferenças em latitude e longitude, respetivamente, entre os 2 pontos.

O valor calculado na Fórmula 2.1, d, representa a distância angular entre os 2 pontos na esfera. Portanto, a distância, em km, entre eles é dada por $r \times d$, onde r representa o raio da esfera em km.

2.1.2 Sistema de Posicionamento Global

Desde que o recetor se encontre no campo de receção de, pelo menos, quatro satélites GPS, é-lhe fornecido, pelo sistema de navegação por satélite, a sua posição, assim como informação horária a qualquer momento e em qualquer lugar. A obtenção da posição através do GPS é feita em 3 passos: (1) identificação dos satélites, (2) cálculo da distância e (3) cálculo da posição.

1. Identificação dos satélites. O Almanaque consiste num conjunto de dados que permitem ao recetor GPS saber onde cada satélite deveria estar a qualquer hora do dia. No caso do recetor não ter informação sobre a última posição nem sobre o Almanaque, tenta receber informação dos satélites mais próximos. A este processo dá-se o nome de arranque frio. No caso do dispositivo ter acesso à última localização e ao Almanaque, pode estimar a posição dos satélites e tentar aceder-lhes. Este processo é conhecido como arranque morno. Por fim, no caso do recetor guardar uma efeméride válida, ou seja, informações sobre a hora e data atuais e a situação de cada

[1] Fonte: http://www.movable-type.co.uk/scripts/gis-faq-5.1.html, acedido em: 08/01/2012

satélite, como, por exemplo, elementos de Kepler e parâmetros associados para compensar forças perturbadoras, calcula as posições dos satélites de uma forma muito mais rápida e precisa. Este processo, por sua vez, é conhecido como o arranque quente.

2. Cálculo da distância. Com as informações de navegação transmitidas é possível calcular as coordenadas dos satélites no espaço. O cálculo da distância é feito da seguinte forma: é medido o tempo que o sinal demora a chegar do satélite ao recetor e divide-se pela velocidade de propagação do sinal. O cálculo das distâncias tem de ser feito usando pelo menos quatro satélites, três para calcular as posições em três dimensões e o quarto para sincronizar o tempo entre os satélites e o recetor.

3. Cálculo da posição. As distâncias calculadas são na realidade pseudo distâncias na medida em que diferem das distâncias reais devido a um erro. Este erro deve-se à reflexão ionosférica que é responsável por abrandar os sinais vindos dos satélites e originar, com isso, distâncias falsas. Para compensar estes erros, os GPSs transmitem os coeficientes de uma fórmula de correção que o recetor aplica nas distâncias calculadas para conseguir resultados mais rigorosos. A seguir, o recetor calcula as coordenadas de cada satélite considerado durante o cálculo. Para cada satélite é extraída a efeméride da mensagem e calculada a sua posição, bem como tempo de transmissão. As coordenadas são dadas no sistema de coordenadas geográficas elipsoidal que utiliza o datum WGS84[2] [DG11].

Hoje em dia, os dispositivos móveis estão cada vez mais adaptados para receber dados capturados por GPS. A precisão dos dados recebidos tem vindo a evoluir, adaptando-se à realidade e contribuindo para a credibilidade desta informação [JM07]. A precisão varia de recetor para recetor e, consequentemente, de dispositivo móvel para dispositivo móvel. No entanto, esta variação não é muito significativa, isto é, consiste numa diferença de alguns metros.

2.1.2.1 Precisão da Posição

Inicialmente o sinal com melhor qualidade estava reservado para fins militares. O sinal disponível para uso civil era intencionalmente degradado. O GPS inclui uma característica (atualmente desativada) chamada *Selective Availability (SA)* que adiciona intencionalmente erros até 100 metros no sinal disponível. Isto foi criado para impedir o uso de guias de armas por parte do inimigo recorrendo a recetores de GPS. Contudo, o SA foi desativado a mando do presidente Bill Clinton a 1 de Maio de 2000. Desta forma a precisão dos GPSs civis passou de 100m para cerca de 20m.

Fatores como a posição dos satélites, o ruído no sinal de rádio, as condições atmosféricas e os objetos sólidos entre os satélites e o recetor afetam, logicamente, a precisão. O ruído pode criar um erro entre 1 a 10 metros, enquanto objetos como árvores, montanhas e grandes edifícios, podem induzir em erros até aos 30 metros.

A problemática da precisão do recetor GPS é bastante complexa mas geralmente consegue obter-se uma posição dentro de um raio de 15 metros da verdadeira posição[3].

[2] Datum refere-se ao modelo matemático teórico da representação da superfície da Terra ao nível do mar; o datum WGS84 é o utilizado pelo GPS

[3] Fonte: http://www.romdas.com/technical/gps/gps-acc.htm, acedido em: 08/01/2012

2.2 Dados Espaciais

Também conhecidos como dados geoespaciais ou informação geográfica, os dados espaciais são dados que se relacionam com objetos que ocupam o espaço. Podem ser descritos através de propriedades geométricas como, por exemplo, a localização e a área e/ou através de propriedades topológicas como, por exemplo, as adjacências.

Os dados espaciais são normalmente acedidos, manipulados e analisados através de um Sistema de Informação Geográfica que permite e facilita a análise, gestão e representação do espaço e dos fenómenos que nele ocorrem.

Devido às suas caraterísticas, os dados espaciais devem ser guardados em bases de dados propriamente preparadas para o efeito de forma a permitir um armazenamento e uma manipulação dos dados de forma eficiente.

2.2.1 Bases de Dados

Uma base de dados espacial é uma base de dados otimizada para armazenar e permitir *queries* aos dados que são relacionados com objetos no espaço, incluindo pontos, linhas e polígonos. Enquanto as bases de dados tradicionais suportam e respondem bem às necessidades de vários tipos de dados, é necessário adicionar funcionalidades por forma a processar dados espaciais[4].

Os dados espaciais requerem uma variedade de operações que não são, de todo, comuns nas bases de dados mais tradicionais. Algumas das *queries* suportadas pelo *Open Geospatial Consortium* (OGC)[5], em bases de dados espaciais, são:

- **Medidas espaciais** – distância entre pontos, áreas de polígonos, etc
- **Predicados espaciais** – permitem *queries* verdadeiro/falso como "existe algum restaurante num raio de 2km?"
- **Funções construtoras** – criação de novos campos (também conhecidos como "*features*" neste tipo de bases de dados) com *queries* SQL especificando apenas os vértices e construindo assim linhas (existe a possibilidade de construir polígonos "fechando" a linha)
- **Funções observadoras** – *queries* que retornam informação específica sobre uma "*feature*" como, por exemplo, a localização do centro de um círculo

Note-se, no entanto, que nem todas as bases de dados suportam todos estes tipos de *queries*.

2.2.2 Estruturas de Dados

Como descrito anteriormente, o uso de dados espaciais e, consequentemente, a forma como é necessário aceder-lhes diverge das abordagens tradicionais. O tipo de *queries* normalmente usadas sobre estes dados requerem estruturas de dados substancialmente mais eficientes.

[4]Fonte: http://en.wikipedia.org/wiki/Spatial_database, acedido em: 24/01/2012
[5]Organização voluntária internacional de padrões de consenso; http://www.opengeospatial.org/

Uma das caraterísticas das bases de dados é a existência de índices, ou seja, os dados são armazenados numa estrutura de dados que melhora a eficiência da procura dos dados. No entanto, os índices das bases de dados tradicionais não respondem de forma satisfatória às necessidades que os dados espaciais apresentam na medida em que são estruturas de dados baseadas no *matching* exato de valores, como *hash tables*, por exemplo, e não são úteis para pesquisas intervalares. Para além disso, ordenações uni-dimensionais de valores chave, tais como índices B-trees e ISAM, não funcionam porque o espaço de pesquisa neste caso é multi-dimensional [Gut84].

Algumas das estruturas usadas para indexar dados em bases de dados espaciais são: Quadtrees, Octrees, R-trees, R^+-trees, R^*-trees, kd-trees, entre outras.

2.2.3 R-trees

Tipicamente este é o método preferido para a indexação em bases de dados espaciais. De uma forma geral, consiste no agrupamento de objetos (formas, linhas e/ou pontos) usando uma fronteira mínima retangular (doravante denominada por MBR[6]).

A ideia inerente à construção da estrutura de dados é o agrupamento dos objetos que estão perto uns dos outros e representá-los com a sua MBR no nível imediatamente acima na árvore. Como todos os objetos estão dentro da MBR, qualquer *query* que não intersete aquele fronteira também não pode intersetar nenhum dos objetos que nela estão contidos.

As folhas da árvore numa R-tree contém registos da forma

$$(I, tuple-identifier)$$

onde $tuple-identifier$ refere-se a um tuplo na base de dados e I é um retângulo n-dimensional que é a fronteira do objeto espacial indexado.

Já os nós intermédios, contém registos da forma

$$(I, child-pointer)$$

onde $child-pointer$ é o endereço do seu descendente na árvore e I cobre todos os retângulos presentes nos descendentes do nó. Note-se que um nó intermédio pode ter entre m e M nós descendentes, sendo $m \leq \frac{M}{2}$.

Uma das dificuldades na elaboração destas estruturas de dados é a procura dos retângulos com a área mínima. Numa procura exaustiva existiriam 2^{M-1} possibilidades, onde M é o número máximo de registos num nó, e isso tornaria o algoritmo altamente ineficiente.

[6]Do inglês: *Minimum Bounding Rectangle*

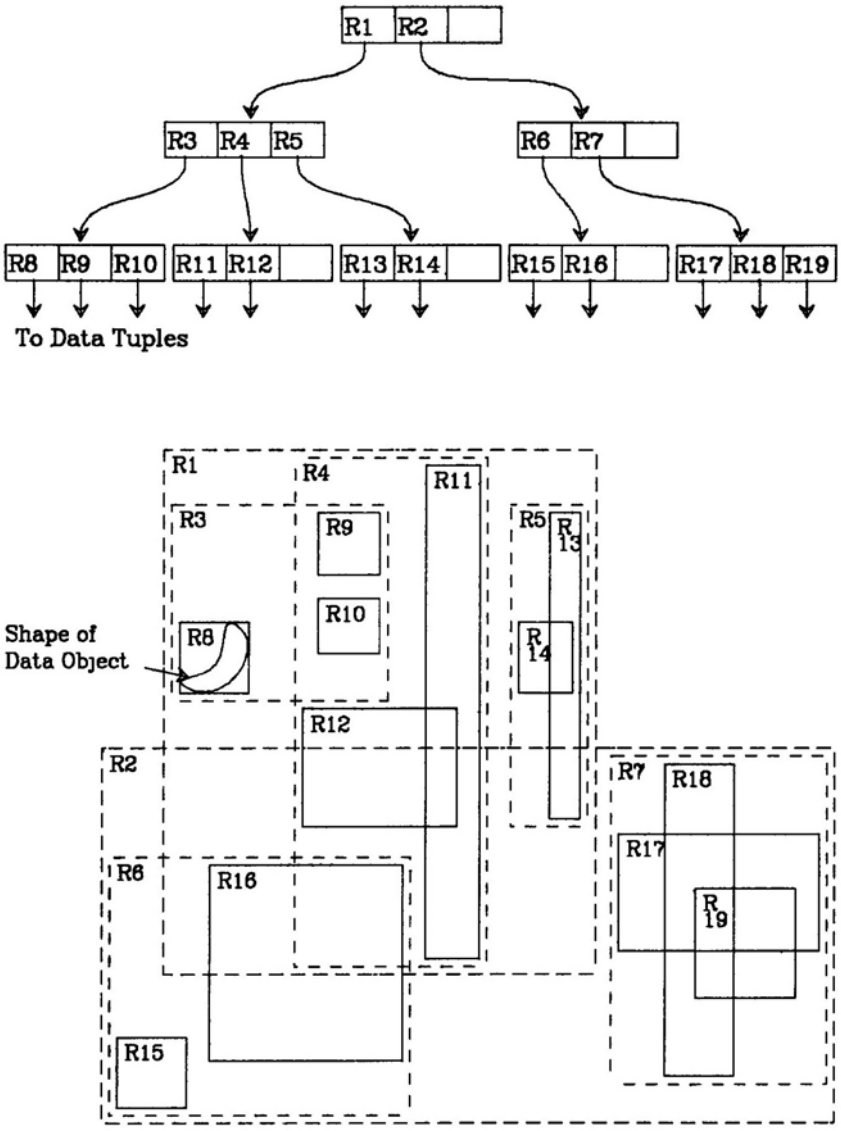


Figura 2.3: Exemplo de uma R-tree [Gut84]

Guttman, também em [Gut84], apresenta 2 algoritmos para este problema: (1) algoritmo de custo quadrático e (2) algoritmo de custo linear.

1. Algoritmo de custo quadrático. Este algoritmo coloca os 2 registos que gastariam maior área se ambos pertencessem ao mesmo grupo, em 2 grupos distintos. Os restantes registos são então atribuídos, um a um, aos grupos existentes. Em cada passo, é calculado para cada registo a área que seria necessário expandir em cada grupo no caso de ser aquele registo a ser inserido. O registo escolhido para ser inserido é o que tem a maior diferença de área expandida necessária entre os 2 grupos. Este é um algoritmo com custo quadrático em M e linear no número de dimensões dos registos [Gut84].

2. Algoritmo de custo linear. Este algoritmo funciona de forma similar ao algoritmo de custo quadrático mas difere dele na seleção dos 2 registos iniciais e na escolha do registo seguinte a ser

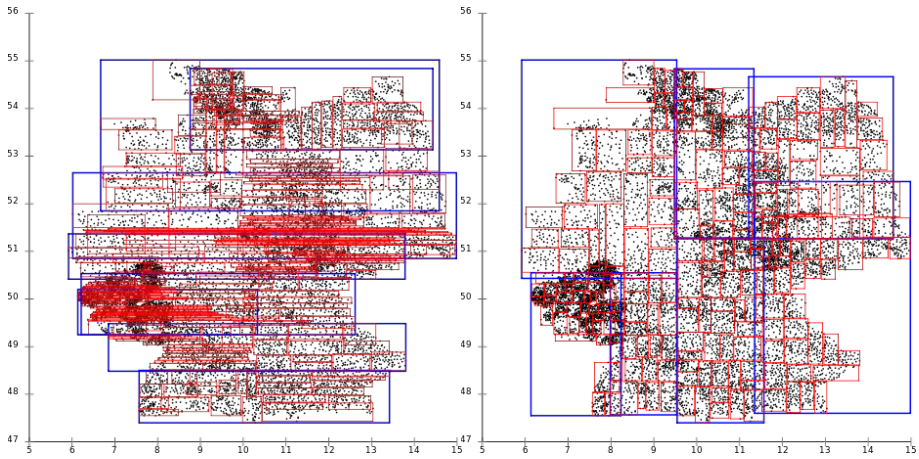
colocado nos grupos. Para a escolha dos 2 registos iniciais, são encontrados, em primeiro lugar, em cada dimensão, os registos cujo retângulo tem o maior lado no lado mais pequeno e o retângulo com menor lado no lado maior. Estes valores são posteriormente normalizados sendo divididos, ao longo da dimensão correspondente, pela largura de todo o conjunto de dados. É escolhido o par de registos com o maior valor normalizado ao longo de qualquer dimensão. A escolha do registo a ser inserido, em cada passo, nos grupos existentes é feita de forma aleatória. Este algoritmo é preferível ao anterior e apresenta um custo linear quer em M quer no número de dimensões dos registos.

O métodos de acesso aos dados espaciais é baseado na aproximação de um objeto espacial complexo pela fronteira mínima retangular com os lados do retângulo paralelos aos eixos do espaço de dados. Embora muita informação sobre o objeto seja perdida, o MBR preserva a informação mais importante sobre as propriedades geométricas do objeto espacial, nomeadamente a localização do objeto e a extensão do objeto em cada eixo [BSS+90].

R^*-trees Enquanto a estrutura de dados R-tree é baseada numa heurística de otimização da área do retângulo que engloba os objetos espaciais em cada nó da árvore, as árvores R^*-trees incorporam uma otimização combinada da área, margem e sobreposição dos retângulos.

A minimização da sobreposição dos retângulos é fundamental para a performance da estrutura de dados. No caso de haver muitas sobreposições de retângulos, ao ser feita uma *query* ou ao inserir um novo elemento, mais do que um ramo da árvore tem de ser explorado.

Em operações de eliminação e *query*, as R^*-trees funcionam da mesma que as R-trees. A diferença está nas operações de atualização e inserção onde é usado um algoritmo de separação de nós internos da árvore e os registos que estavam num nó que excedeu o limite máximo de elementos voltam a ser inseridos na árvore ao invés de ser calculada a melhor forma de os separar.



(a) R-tree com separação quadrática (b) R^*-tree

Figura 2.4: Resultado da aplicação de diferentes heurísticas (http://en.wikipedia.org/wiki/R*tree)

Esta ideia surgiu com base na observação de que a estrutura R-tree é muito suscetível à ordem a partir da qual os registos são inseridos. A re-inserção de registos permite-lhes encontrar um local na árvore mais apropriado do que o seu local original. Isto tem o efeito de produzir grupos de registos melhor agrupados nos nós e reduzir a sobreposição de retângulos. A re-inserção pode também ser vista como um método incremental de otimização da árvore que é despoletada quando um nó chega ao limite de registos que pode referenciar[7].

Na Figura 2.4 é visível o quão importante é o uso de uma boa heurística na construção da árvore. Na Figura 2.4a, onde foi aplicado o algoritmo de custo quadrático, descrito na Secção 2.2.3, existem muitas sobreposições de retângulos e muitos deles são transversais à imagem, têm uma largura grande e uma altura pequena. Essa caraterística acaba por ser um problema pois sempre que sejam pesquisados retângulos com altura maior que a deles (o que acontecerá muitas vezes), mais do que um retângulo terá de ser pesquisado, ou seja, mais do que um ramo da árvore terá de ser explorado. Já na Figura 2.4b, onde a árvore construída foi um R^*-tree, os retângulos têm muito pouca sobreposição e a estratégia de separação fez com que não existam "fatias". O resultado é bastante mais útil.

2.2.4 Discussão

As R^*-trees apresentam, efetivamente, um desempenho bastante satisfatório. Em comparação com a estrutura de dados com que mais se assemelha, as R-trees, apresenta uma heurística que produz retângulos muito mais apropriados para diversas aplicações e suporta de forma eficiente pontos e dados espaciais ao mesmo tempo. O método de re-inserção dos registos otimiza a árvore construída.

Contudo, as R^*-trees apresentam uma complexidade temporal superior à das R-tree principalmente devido às re-inserções. Mesmo assim, apresentam-se como uma alternativa altamente viável na medida que a diferença não é excessivamente grande e o custo de implementação é também apenas um pouco maior.

Para além destas estruturas de dados, existem bastantes mais mas foram descartadas logo à partida devido a algumas das caraterísticas que apresentam. Métodos de células, por exemplo, não são adequados para estruturas dinâmicas uma vez que as fronteiras das células têm de ser definidas à partida. As Quadtrees e as kd-trees, por seu turno, apresentam o problema de não terem em conta a paginação[8] da memória secundária. K-D-B-trees são úteis apenas para pontos e não para objetos. O uso de índices intervalares foi também sugerido em [BF79] mas o método não pode ser usado num espaço n-dimensional. "Corner stitching" é um exemplo de uma estrutura que funciona bem com objetos em espaços bi-dimensionais mas não é eficiente em pesquisas aleatórias em conjuntos de dados enormes.

[7] Fonte: http://en.wikipedia.org/wiki/R*tree, acedido em: 25/01/2012

[8] Importante esquema de gestão de memória através do qual o computador consegue armazenar e aceder a dados da memória secundária para serem usados na memória principal

2.3 Extração de Informação Relevante

Data mining consiste na extração automática de padrões a partir de dados guardados de forma eletrónica. William Frawly descreveu *data mining* como "*The nontrivial extraction of implicit, previously unknown, and potentially useful information from data [Wil]*".

Conhecimentos de domínios como estatística, aprendizagem automática, reconhecimento de padrões, inteligência artificial e mesmo visualização de dados são necessários para levar a cabo, com sucesso, a extração de informação relevante. Muito do trabalho a ser desenvolvido nesta dissertação consiste no desenvolvimento e aplicação de técnicas de aprendizagem automática para encontrar e expor padrões nos dados.

> "Um programa diz-se que aprende de uma experiência E com respeito a alguma classe de tarefas T e medição de desempenho P, se o seu desempenho nas tarefas T, medida por P, melhorar com a experiência E." [Mit97]

O processo de aprendizagem automática resulta no desenvolvimento de algoritmos, técnicas e mecanismos capazes de induzir conhecimento através de exemplos de dados.

> "The field of Machine Learning is concerned with the question of how to construct computer programs that automatically improve with experience." [Mit97]

Existem dois tipos principais de aprendizagem: aprendizagem supervisionada e não supervisionada. Na aprendizagem supervisionada as instâncias do conjunto de treino fazem parte do conjunto de dados de entrada e de saída. O conjunto de dados de entrada é constituído por vetores de valores (instâncias) e a saída é uma etiqueta que identifica a classe da própria instância. O objetivo é construir uma função que aceite instâncias válidas e consiga prever a sua classe. O algoritmo deve generalizar dos dados de treino para outras situações de forma razoável para conseguir um desempenho aceitável. Por seu lado, na aprendizagem não supervisionada as instâncias não possuem uma classe associada. É o algoritmo que tem de modelar o conjunto de dados de saída. Isto corresponde ao domínio das técnicas de *clustering* e aplicam-se quando as instâncias têm de ser divididas em grupos naturais.

Zhou *et al.*, em [ZFL+04], desenvolveram o algoritmo de *clustering*, baseado em densidade, algoritmo DJ-Cluster e, como forma de teste, com base no histórico espácio-temporal de cada utilizador descobriram o seu *gazetteer*[9] pessoal. Quando comparado com o conhecido algoritmo de *clustering* K-means, conseguiu superá-lo, de longe, nas 3 métricas de avaliação: *recall*, *precision* e *SurpriseFactor*, que significa, o número de locais descobertos que o próprio utilizador não tinha identificado à priori mas que, de facto, fazem sentido ser identificados.

Em [ZBST07], Zhou *et al.* novamente, para além de quererem descobrir os locais significativos de cada utilizador, passaram a querer identificá-los como *frequente e importante*, *frequente e não importante*, *não frequente e importante* ou *não frequente e não importante*. Para a descoberta dos locais compararam os algoritmos K-means e DJ-Cluster e para a classificação de cada um

[9] Dicionário geográfico; uma importante referência para obter informações sobre lugares e nomes de lugares

usaram técnicas de classificação, nomeadamente os classificadores KNN e C4.5. Foram recolhidos dados de 28 utilizadores num mês completo e calculados alguns atributos de cada local. Para a classificação dos locais, os atributos que originaram melhores resultados foram: (1) *Readings*, (2) *ReadingDays*, (3) *Visits* e (4) *VisitDays*.

1. ***Readings***. Tempo total passado no local.
2. ***ReadingDays***. Número de dias diferentes que serviram para calcular *Readings*.
3. ***Visits***. Número de visitas ao local.
4. ***VisitDays***. Número de dias diferentes que serviram para calcular *Visits*.

Já em [AS03], foi desenvolvido, recorrendo a modelos de Markov, um modelo preditivo dos locais a visitar de cada utilizador. Os modelos de Markov foram utilizados para representar as transições entre os locais detetados como importantes e os movimentos futuros eram previstos com base na transição com maior probabilidade a partir do local atual do próprio utilizador. Através do uso deste modelo tornaram-se visíveis sequências de locais que ocorrem frequentemente e revelaram-se também alguns padrões de movimento relevantes na vida dos utilizadores.

2.4 Identificação e Definição de Locais

Inicialmente há a necessidade de definir os locais habituais de partida e/ou chegada de cada utilizador de forma completamente automática. O problema da deteção destes locais, locais onde o utilizador passa uma parte substancial do seu tempo e viaja de e para lá frequentemente, é um problema onde "salta à vista" a aplicação de algoritmos de *clustering* baseados em densidade. Estes são os locais referenciados como locais significativos de um utilizador. Com a recolha de coordenadas geográficas através de um recetor GPS, ao estar muito tempo num determinado local, a densidade de pontos nesse local será maior que nos restantes. A aplicação deste tipo de algoritmos permite ainda que algumas viagens esporádicas sejam automaticamente descartadas como, por exemplo, a ida ao aeroporto uma vez por ano.

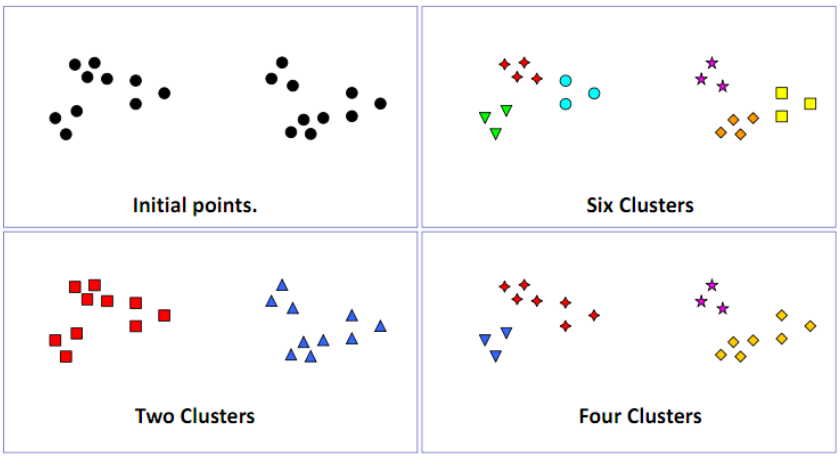


Figura 2.5: Formação de vários *clusters* com o mesmo conjunto de pontos originais (http://paginas.fe.up.pt/~ec/)

A ideia base de *clustering* consiste na divisão de instâncias em grupos naturais. *Clustering* é um processo de descoberta que agrupa conjuntos de objetos de dados, por forma a maximizar a semelhança entre os objetos dentro do mesmo grupo e a minimizar a semelhança entre objetos de grupos diferentes [KR90].

Contudo, em muitos casos a noção de conjunto não está bem definida. Na Figura 2.5 é visível este problema. Com o mesmo conjunto de pontos há várias formas viáveis de os agrupar. A definição de grupo depende da natureza dos dados e dos resultados pretendidos.

2.4.1 Deteção de Pontos de Estadia

Marcmasse e Schmandt, no sistema comMotion [MS00], basearam-se na perda de sinal do recetor GPS para identificar pontos de estadia. Sempre que existe uma interrupção de receção de sinal GPS inferior a um certo limite de distância, o utilizador é questionado se aquele local deve ou não ser considerado. Para além de não ser um sistema completamente autónomo, existem alguns casos, facilmente identificáveis, onde o sistema não responde da melhor forma às necessidades. Por exemplo, locais ao ar livre têm uma probabilidade bastante reduzida de serem considerados.

Em [KWSB04], foi desenvolvido um algoritmo de agrupamento baseado no tempo para extrair locais num campus. Sempre que há uma estadia, em termos de tempo, superior a um valor t e a distância para o local anterior é superior a um valor d, é considerado um novo ponto de estadia. Neste caso, para capturar a posição física no campus foi utilizado o endereço MAC dos pontos de acesso de uma rede Wi-fi.

Já mais tarde, com algumas semelhanças com o trabalho de Kang *et al.* em [KWSB04], surge [LZX+08].

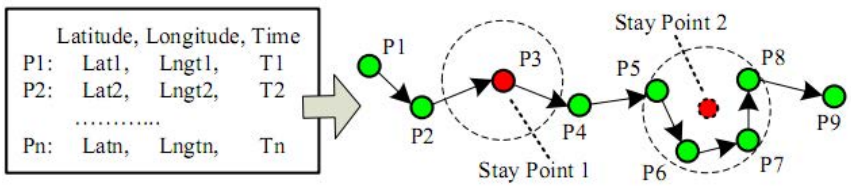


Figura 2.6: Exemplos de *stay points* [LZX+08]

Em [LZX+08], existem duas possibilidades para a deteção de *stay points*: **(*stay point* 1)** quando o utilizador permanece no mesmo local por um período de tempo superior a um certo limite e **(*stay point* 2)** quando o utilizador anda dentro de uma certa região, delimitada por um raio r, por um período de tempo superior a um certo limite. Neste caso, o *stay point* é o ponto com a média das coordenadas dos vários pontos da região.

No caso do *stay point 1*, são detetadas situações como, por exemplo, quando o utilizador entra num edifício e perde o sinal GPS. Já o *stay point 2* é capaz de detetar locais em espaços abertos e onde o utilizador anda "à volta" de uma certa região espacial.

Este algoritmo de deteção de *stay points* faz uso de 2 parâmetros: (1) um limite de distância, *distThreh*, que representa a distância máxima a que as coordenadas podem estar do ponto inicial da região para fazerem parte do *stay point* e (2) um limite de tempo, *timeThreh*, que representa o mínimo de tempo que o utilizador tem de estar na região para que o conjunto de pontos dentro dela sejam um *stay point*.

Algoritmo 1 Algoritmo StayPoint_Detection(*P, distThreh, timeThreh*) [LZX+08]

Input: A GPS log P, a distance threshold $distThreh$ and time span threshold $timeThreh$
Output: A set of stay points $SP = \{S\}$

```
1:  i = 0; pointNum = |P|; //the number of GPS points in a GPS log
2:  while i < pointNum do
3:      j = i + 1;
4:      while j < pointNum do
5:          dist = Distance(p_i, p_j); //calculate the distance between two points
6:          if dist > distThreh then
7:              ΔT = p_j.T − p_i.T; //calculate the time span between two points
8:              if ΔT > timeThreh then
9:                  S.coord = ComputeMeanCoord({p_k | i ≤ k ≤ j});
10:                 S.arvT = p_i.T; S.levT = p_j.T;
11:                 SP.insert(S);
12:             end if
13:             i = j; break;
14:         end if
15:         j = j + 1;
16:     end while
17: end while
18: return SP;
```

A complexidade temporal do Algoritmo 1 é *O(n)*, onde *n* representa o número de pontos GPS.

2.4.2 Determinação de Locais Significativos

Inicialmente, no sentido de determinar locais significativos através de pontos espaciais, foram levadas a cabo algumas experiências recorrendo a algoritmos de *clustering* por partição. Ashbrook e Starner, em [AS03], começaram por criar aquilo a que chamaram um *place* sempre que o recetor GPS perdia o sinal (experimentaram outro método mas com piores resultados) e após a determinação dos *places* do utilizador, com base nesses pontos, determinaram aquilo a que chamaram *locations*, do próprio utilizador. Isto é, recorrendo a uma versão do algoritmo K-means com umas pequenas alterações que corria sobre os *places* do utilizador, eram determinados os locais significativos do mesmo.

Contudo, não é previsível um grande futuro a este tipo de algoritmos de *clustering* em termos de aplicação nesta área. As suas limitações são demasiado problemáticas para a obtenção de soluções razoáveis. Em primeiro lugar estes são algoritmos que tipicamente necessitam do número de *clusters* a formar como *input* e este não é um valor, na maior parte dos casos, viável de calcular. Para além disso, o algoritmo K-means, em particular, não é um algoritmo determinístico. Outro inconveniente é o facto de neste tipo de algoritmos todos os pontos fazerem parte da solução final. Este é um enorme inconveniente pois ruído, *outliers* e viagens pouco frequentes para zonas diferentes das habituais fazem com que os centróides dos *clusters* sejam desviados da sua posição ideal.

O algoritmo CLARANS [NHS02], algoritmo de *clustering* por partição, que foi desenvolvido para ser aplicado em conjuntos de dados enormes, tem também o problema de não conseguir encontrar *clusters* não convexos, à semelhança dos restantes algoritmos de *clustering* por partição.

Em [ANN09] foi usada uma abordagem diferente na medida em que não recorrem diretamente a algoritmos de *clustering* para a determinação dos locais. Neste caso o problema é tratado num contexto diferente, onde o objetivo principal é extrair informação relevante como um passo em direção à melhoria da segurança no trabalho, mais concretamente em minas. Neste artigo é apresentado um algoritmo para "extrair locais significativos de um conjunto de pontos GPS [ANN09]" com uma abordagem que consiste na atribuição de uma pontuação, entre 0 e 1 onde 1 representa o ótimo e 0 o inverso, a cada posição GPS recolhida. Esta pontuação é calculada dependendo do objetivo, ou seja, dependendo do tipo de locais a extrair dos dados. No exemplo dado, para extrair as regiões de baixa velocidade de uma mina, foram usadas apenas as coordenadas geográficas recolhidas e a velocidade instantânea aquando da recolha de cada coordenada. Para cada veículo foram calculados 2 conjuntos ordenados $\{x_1,...,x_N\}$ e $\{\phi_1,...,\phi_N\}$. O primeiro conjunto contém posições de vetor $x_n \in \Re^2$ (as coordenadas GPS) e o segundo contém vetores de caraterísticas $\phi_n \in \Re^M$ (neste caso contém apenas a velocidade instantânea). A cada posição GPS foi atribuída uma pontuação e, posteriormente, apenas com as posições com pontuação máxima foram construídos grafos em que as arestas ligam os pontos consecutivos de cada segmento. As áreas ocupadas por cada grafo são, então, os locais extraídos como sendo os de baixa velocidade.

2.4.2.1 *Clustering*

Os algoritmos de *clustering* baseados em densidade são excelentes candidatos para determinar locais a partir de pontos visitados. Este tipo de algoritmos apresentam um enorme conjunto de potencialidades tais como determinar *clusters* de forma arbitrária, determinar um número variável de *clusters* e ignorar ruído e *outliers* nos dados.

Em [EKSX] é apresentado o algoritmo DBSCAN, que é um algoritmo de *clustering* baseado em densidade. Um dos requisitos que esteve na base do desenvolvimento deste algoritmo foi a exigência de ter uma boa eficiência ao trabalhar com bases de dados espaciais enormes, caso raro na maior parte dos algoritmos de *clustering* desenvolvidos até à data.

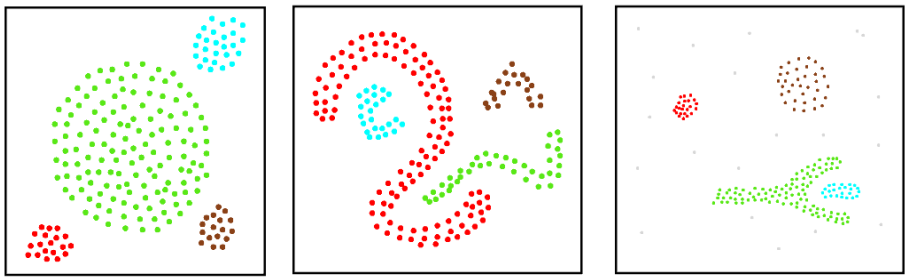


Figura 2.7: *Clusters* descobertos pelo algoritmo DBSCAN [EKSX]

Embora tenha havido uma tentativa de automatizar o processo de escolha dos parâmetros de entrada do algoritmo, na maior parte dos casos é necessário defini-los e, para isso, ter algum conhecimento do domínio do problema. Os 2 parâmetros de entrada são: (1) *Eps* e (2) *MinPts*.

1. *Eps*. Raio máximo da vizinhança (distância máxima a que um ponto pode estar para ser considerado vizinho).

2. *MinPts*. Número mínimo de pontos vizinhos.

Existem ainda 4 definições chave: (1) *core point*, (2) *directly density-reachable*, (3) *density-reachable* e (4) *density-connected*.

1. *Core point*. Ponto com pelo menos *MinPts* num raio *Eps*.

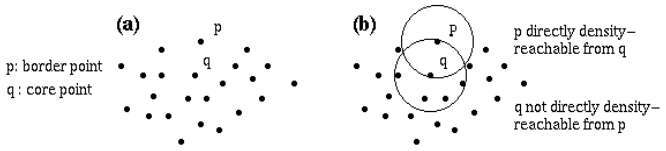


Figura 2.8: Noções de *core point* e de *border point* [EKSX]

2. *Directly density-reachable*. O ponto *p* é *directly density-reachable* pelo ponto *q* se o ponto *q* é um *core point* e o ponto *p* está a uma distância $\leq$ *Eps*

3. *Density-reachable*. O ponto *x* é *density-reachable* pelo ponto *y* se o ponto *y* chegar até ele navegando sempre por pontos *directly density-reachable*

4. *Density-connected*. Os pontos *x* e *y* são *density-connected* se existe pelo menos 1 ponto comum do qual ambos são *density-reachable*

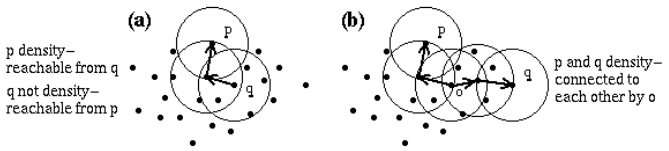


Figura 2.9: Noções de *density-reachable* e de *density-connected* [EKSX]

Posto isto, um *cluster* é definido como o conjunto máximo de pontos *density-connected*. São criados 2 *clusters* se os 2 pontos mais perto de cada um deles têm uma distância $\geq Eps$.

Algoritmo 2 Algoritmo DBSCAN($SetOfPoints, Eps, MinPts$) [EKSX]

```
1:  //SetOfPoints is UNCLASSIFIED
2:  ClusterId := nextId(NOISE);
3:  for i FROM 1 TO SetOfPoints.size do
4:     Point := SetOfPoints.get(i);
5:     if Point.ClId = UNCLASSIFIED then
6:        if ExpandCluster(SetOfPoints, Point, ClusterId, Eps, MinPts) then
7:           ClusterId := nextId(ClusterId);
8:        end if
9:     end if
10: end for
```

A variável *SetOfPoints* significa a base de dados toda ou o *cluster* descoberto na iteração anterior. A função mais importante é a função *ExpandCluster(SetOfPoints, Point, ClusterId, Eps, MinPts)* que retorna uma variável do tipo *boolean* e é apresentada na Função 3.

A complexidade temporal do Algoritmo 2 é *O(n * log n)*, onde *n* representa o número de pontos GPS.

Recentemente, em [ZFL+04], foi apresentado um algoritmo de *clustering* baseado em densidade, algoritmo DJ-Cluster, com algumas semelhanças em termos de implementação com o algoritmo DBSCAN. Todavia é mais simples tanto de implementar como de perceber a ideia subjacente. No entanto, as comparações de desempenho apresentadas em [ZFL+04] confrontam o algoritmo DJ-Cluster com o algoritmo K-means. Nessa comparação o algoritmo DJ-Cluster tem uma enorme vantagem, como seria de esperar. Seria bem mais interessante compará-lo com o algoritmo DBSCAN, por exemplo, na medida em que é um algoritmo do mesmo tipo e com muito melhores resultados em aplicações neste tipo de problemas.

Função 3 Função ExpandCluster($SetOfPoints, Point, ClId, Eps, MinPts$) : $Boolean$ [EKSX]

```
1:  seeds := SetOfPoints.regionQuery(Point, Eps);
2:  //no core point
3:  if seeds.size < MinPts then
4:     SetOfPoint.changeClId(Point, NOISE);
5:     return False;
6:  else
7:     //all points in seeds are density-reachable from Point
8:     SetOfPoints.changeClIds(seeds, ClId);
9:     seeds.delete(Point);
10:    while seeds <> Empty do
11:       currentP := seeds.first();
12:       result := SetOfPoints.regionQuery(currentP, Eps);
13:       if result.size ≥ MinPts then
14:          for i FROM 1 TO results.size do
15:             resultP := result.get(i);
16:             if resultP.ClId IN {UNCLASSIFIED, NOISE} then
17:                if resultP.ClId = UNCLASSIFIED then
18:                   seeds.append(resultP);
19:                end if
20:                SetOfPoints.changeClId(resultP, ClId);
21:             end if
22:          end for
23:       end if
24:       seeds.delete(currentP);
25:    end while
26:    return True;
27: end if
```

Em [XEKS], os autores do algoritmo DBSCAN, apresentam um novo algoritmo, o DBCLASD. Este é um algoritmo baseado na assunção de que os pontos dentro de um *cluster* estão uniformemente distribuídos. Apesar de nem sempre ser verdade, em muitos casos isso acontece.

Partindo desse pressuposto, se a distribuição de probabilidade da distância ao vizinho mais próximo dentro um *cluster* for descoberta, pode ser usada para decidir quando um ponto vizinho deve ser aceite como membro de um *cluster* ou não [XEKS].

É então analisada a distribuição esperada da distância dos pontos ao seu vizinho mais próximo. Consideremos um espaço de dados R, com volume $Vol(R)$, e um conjunto de pontos N tal que a probabilidade de um dos N pontos "cair" num sub-espaço S de R, com volume $Vol(S)$, é $\frac{Vol(S)}{Vol(R)}$, ou seja, os pontos "caem" de forma completamente independente no sub-espaço S.

A probabilidade da distância D de um ponto qualquer q ao seu vizinho mais próximo, no

espaço R, é $> x$ se não existe nenhum ponto dentro da hyper-esfera de raio x à volta do ponto q. Portanto,

$$P(D > x) = \left(1 - \frac{Vol(SP(q,x))}{Vol(R)}\right)^N \tag{2.2}$$

em que $SP(q,x)$ significa a hyper-esfera de raio x, à volta do ponto q.

A partir de 2.2 retira-se que

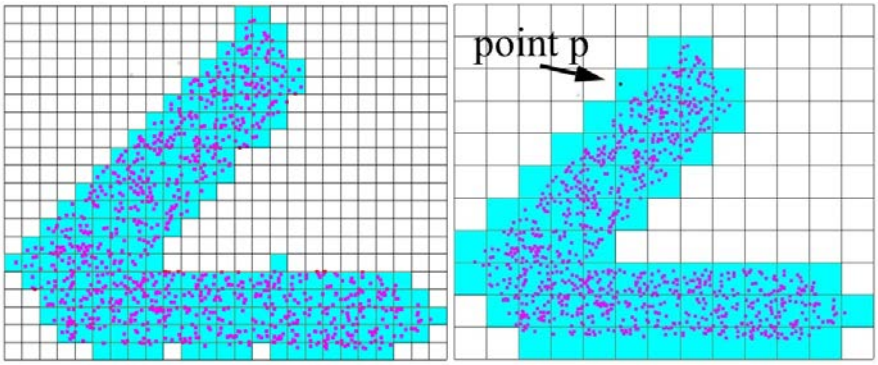
$$P(D \leq x) = 1 - \left(1 - \frac{Vol(SP(q,x))}{Vol(R)}\right)^N \tag{2.3}$$

A 2 dimensões, por exemplo, temos então que

$$F(x) = 1 - \left(1 - \frac{\pi x^2}{Vol(R)}\right)^N \tag{2.4}$$

Contudo, o $Vol(R)$ de um conjunto de pontos que pode ter uma forma arbitrária não é trivial de calcular.

Na realidade, a área, num espaço bi-dimensional, de um conjunto de pontos nem existe. A forma adotada consiste no cálculo de uma aproximação aos pontos que seja tão similar quanto possível à forma do *cluster* e que seja conectada. A área efetivamente calculada é a da aproximação.



(a) Aproximação ao conjunto de pontos (b) Aproximação com um novo ponto p

Figura 2.10: Evolução da grelha de aproximação a um conjunto de pontos [XEKS]

Para determinar o polígono aproximado aos pontos foi usada uma abordagem baseada numa grelha, como ilustrado na Figura 2.10. O comprimento da largura de cada célula da grelha é igual à maior distância ao vizinho mais próximo dos elementos do *cluster*.

O algoritmos DBCLASD é incremental, ou seja, a atribuição de um ponto a um *cluster* é baseada apenas nos pontos processados até ao momento. O funcionamento geral do algoritmo é a atribuição dos pontos vizinhos do *cluster* inicial a esse *cluster* desde que o conjunto de distâncias ao vizinho mais próximo do *cluster* resultante se mantenha dentro do valor da distribuição de distância esperada. Sempre que um ponto que já pertence a um *cluster* é considerado candidato

para outro, ou seja, está a uma distância relativamente pequena de pelo menos um dos pontos desse outro *cluster*, é tentada a junção dos dois *clusters*. Assim é evitado que exista uma separação de *clusters* errada, que poderia acontecer devido à ordem pela qual os pontos foram avaliados.

Este algoritmo apresenta, para além de todas as vantagens do DBSCAN, a não necessidade de parâmetros de entrada. Contudo, apresenta também uma menor escalabilidade. Em termos de eficiência este algoritmo supera claramente o algoritmo CLARANS (um algoritmo de *clustering* por partição desenvolvido para aplicação em bases de dados enormes) mas fica atrás do algoritmo DBSCAN. Até conjuntos de dados de 100 mil pontos a diferença não é muito significativa mas a partir daí existe realmente um fosso cada vez maior.

2.4.3 Discussão

O algoritmo para a deteção de pontos de estadia, apresentado na Secção 2.4.1, tem alguns problemas, nomeadamente o não considerar a repetição de locais e, também devido a isso, dar tanta importância a um ponto de estadia que só ocorre uma vez por ano como a outro que está num local onde é frequente serem criados pontos de estadia. A razão para estes problemas está relacionada com o facto de cada vez que é descoberto um local, o algoritmo assumir que é um novo local e não ter como o comparar com os já conhecidos.

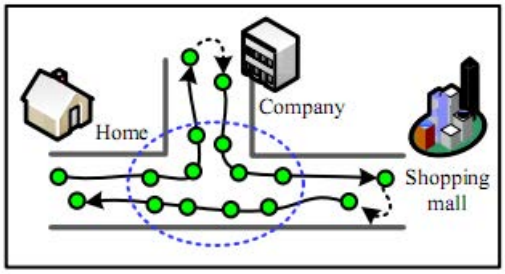


Figura 2.11: Deteção de locais significativos com algoritmos de *clustering* baseados em densidade [LZX+08]

Por outro lado, os algoritmos de *clustering* baseados em densidade, por si só, também não resolvem o problema da descoberta dos locais significativos. Conceptualmente, a ideia da aplicação destes algoritmos neste tipo de problemas é bastante boa. Contudo, é normal que o recetor GPS não receba sinal dentro de um edifício e, portanto, a densidade de pontos nesses locais (casa, local de trabalho, ginásio, etc) será igual, ou até menor, que no caminho para lá, por exemplo.

2.5 Suporte à Mobilidade

Participatory sensing é um campo relativamente novo relacionado com redes de sensores. Uma rede *participatory sensing* é uma rede na qual um conjunto de utilizadores colaboram com o objetivo de recolher dados, através de sensores, do seu ambiente [WARS09]. Em [BEH+06] é dado como exemplo, entre outros, a saúde pública. A ideia é a população voluntariamente

fornecer os seus sintomas médicos através de uma aplicação, ligada nos seus dispositivos móveis, e que pode fazer com que as autoridades rapidamente identifiquem e combatam problemas de saúde específicos no ambiente em geral.

Já quando se fala de *collaborative sensing*, fala-se em aparelhos com diferentes capacidades em termos de sensoriamento e que colaboram entre si de forma a fornecer uns aos outros dados de contexto. Esses dados podem ter enorme relevância na medida em que cada um dos aparelhos sozinho não os podia obter. Imaginemos o caso, ilustrado na Figura 2.12, em que a Alice tem sensor de GPS e bluetooth e o Bob tem apenas bluetooth. Neste caso, e se for possível uma comunicação por bluetooth, podem colaborar de forma a que o Bob fique a saber as suas coordenadas geográficas com uma margem de erro $\leq$ ao alcance do seu bluetooth.

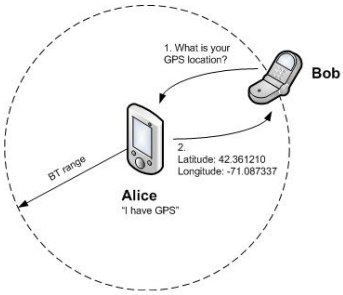


Figura 2.12: Exemplo de *collaborative sensing* [WARS09]

Se o Bob tivesse uma ligação à internet, e a Alice não, ele também podia fornecer-lhe informações como a temperatura atual, por exemplo.

Ambos os campos, *participatory* e *collaborative sensing*, podem ter um papel fundamental no que ao suporte à mobilidade diz respeito. Para além disso, a ubiquidade dos *smartphones* possibilita-lhes, atualmente, uma nova dimensão.

2.5.1 Projeto MISC

O projeto MISC - Massive Information Scavenging with Intelligent Transportation Systems[10], no âmbito do qual esta dissertação se insere, objetiva criar arquiteturas eficientes e seguras para recolhas maciças de dados urbanos e ser capaz de os guardar, processar e divulgar.

No projeto recorre-se a uma rede *participatory sensing* onde os dados recolhidos podem ser extremamente ricos e variados. A recolha é feita através de carros privados, táxis, autocarros e/ou camiões e armazenados remotamente. Destes dados podem ser extraídas informações como condições das rotas, características do tráfico ou até parâmetros ambientais para uso no âmbito de suporte à mobilidade.

Os principais objetivos do projeto assentam em 4 pilares: (1) objetivos científicos, onde é procurado, entre outras coisas, criar modelos adequados para redes de fluxos de informação em redes de transportes e ambientes urbanos e fornecer diretrizes sólidas para *designs* e arquiteturas

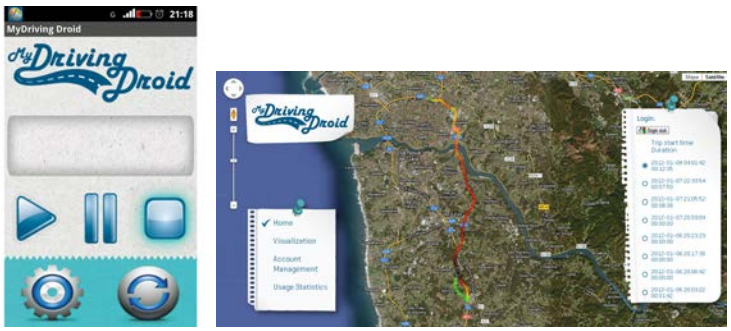
[10] http://www.it.up.pt/misc/

dependentes do sistema, (2) objetivos técnicos, que passam pelo desenvolvimento de protocolos de rede seguros que aproveitam informação geográfica para levar a informação realmente útil para o local certo, (3) objetivos sócio-económicos, como aumentar a segurança e a eficiência dos transportes públicos e redes de estradas recorrendo a tecnologias de recolha maciça de dados e fazendo a transferência destas tecnologias para empresas, e, por fim, (4) objetivos educacionais, que passam pela atração de alunos para a investigação, usar as ferramentas de recolha de dados desenvolvidas no próprio laboratório de investigação e adotar algumas das boas práticas do MIT.

2.5.1.1 Aplicação MyDrivingDroid

A aplicação MyDrivingDroid foi desenvolvida no âmbito do projeto MISC e funciona em *smartphones* com sistema operativo Android (versão 2.1 ou superior). Esta é a aplicação usada para a recolha de dados para esta dissertação. Os dados recolhidos a cada segundo através do sensor GPS são armazenados numa base de dados relacional MySQL.

Sempre que o utilizador desejar, e tiver uma ligação à internet estável, pode enviar os dados recolhidos anteriormente para um servidor remoto. Os dados são nessa altura armazenados na base de dados. Um dos principais inconvenientes das bases de dados relacionais foi internamente resolvido por uma arquitetura *backend*, separando o servidor principal de escrita na base de dados dos servidores secundários de leitura, prevenindo operações complexas de bloquear e deixar lento o servidor principal [RAV+11].



(a) Aplicação MyDrivingDroid (b) Exemplo de viagem no *site* do MyDrivingDroid

Figura 2.13: Aplicação e *site* do MyDrivingDroid

Como é possível ver na Figura 2.13b, depois de haver viagens guardadas no servidor remoto é possível vê-las no *site* do MyDrivingDroid[11].

2.5.2 Discussão

A ubiquidade dos dispositivos móveis e a cada vez maior capacidade de recolherem dados fiáveis do ambiente em que estão inseridos fazem com que a comunidade científica tenha enveredado pelo seu uso para de forma participativa e/ou colaborativa recolherem e/ou partilharem dados do

[11] http://misc.fe.up.pt/mydrivingdroid/

próprio ambiente e, posteriormente, explorarem-nos de forma a que seja extraído valor útil para a comunidade. O projeto MISC, apresentado na Secção 2.5.1, é um exemplo disso - orientado para o suporte à mobilidade - mas existem outros projetos e noutras áreas.

O projeto UrbanSensing[12], por exemplo, permite utilizar o dispositivo móvel para explorar e partilhar o impacto que estamos a provocar no ambiente e vice-versa. Através da análise dos caminhos seguidos pelo utilizador (é feita uma recolha de dados do sensor de GPS), condições climatéricas e estimativas dos padrões de tráfego e por aplicação de modelos científicos são produzidas várias estimativas.

Por seu lado, no projeto MetroSense[13], que se baseia na ideia de que a recolha de dados através de sensores está a ir numa direção centrada nas pessoas, o objetivo passa pelo desenvolvimento de novas aplicações e novos paradigmas para dispositivos móveis permitindo uma rede móvel de sensores global capaz de recolha de dados através de sensores à escala social. Este projeto é constituído por vários projetos e um deles, o CenceMe[14], está já disponível na AppStore para iPhone. A aplicação com o nome do projeto, CenceMe, é um sistema pessoal que recorre ao uso de sensores que permite que membros de redes sociais partilhem os seus dados recolhidos com os seus colegas. Os dados capturados e partilhados são o estado do utilizador em termos de atividade (sentado, a correr, etc), a disposição (feliz, triste, etc), os hábitos (no ginásio, no trabalho, etc) e o redor do utilizador (barulho, calor, claridade, etc).

Já no projeto WikiCity[15], a ser desenvolvido no MIT, está a ser criada uma plataforma para armazenamento e troca de dados. Estes dados estão acessíveis através de dispositivos móveis e interfaces web. O objetivo do projeto é saber até que ponto é possível ter uma cidade que se comporte como um sistema de controlo em tempo real. Este é também um exemplo onde através de *participatory sensing*, e dada a possibilidade de troca de informação, *collaborative sensing*, é esperado que venha a ser criado algum tipo de suporte à mobilidade na cidade em estudo.

2.6 Sistemas para *Carpooling*

Em relação a aplicações para *carpooling*, atualmente não é do nosso conhecimento nenhum sistema automático capaz de sugerir partilha de rotas. Embora haja uma cada vez maior sensibilização das pessoas para os problemas ambientais, pode dizer-se que não tem havido uma resposta direta, prática e concisa por parte da tecnologia.

No entanto são conhecidas algumas plataformas para partilha de veículo tanto a nível nacional como internacional. Conceptualmente são todas muito semelhantes. Em todas elas é suposto o utilizador colocar a informação das suas viagens frequentes, local de partida, local de chegada, dias e horas, indicar se é condutor, passageiro ou se lhe é indiferente e, em algumas delas, no

[12] http://urban.cens.ucla.edu/
[13] http://metrosense.cs.dartmouth.edu/
[14] http://metrosense.cs.dartmouth.edu/projects.html
[15] http://senseable.mit.edu/wikicity/

caso de ser condutor, os lugares disponíveis e a quantia cobrada por cada lugar no veículo. Um exemplo, a nível internacional, é o zimride[16]. Em Portugal o sistemas mais usado é o galpshare[17].

Wash, Hemphill e Resnick, em [WHR], fazem referência ao projeto RideNow. Este é um projeto com vista à ajuda de pessoas dentro de um grupo ou organização a coordenar a partilha de veículo. Foi criado um serviço, também ele chamado RideNow[18], que recorre a uma abordagem um pouco diferente da esperada. Ao invés de tentar servir a população de uma região inteira, o serviço pretende coordenar a partilha de veículo entre grupos ou pequenas organizações. O sistema desenvolvido permite que os utilizadores interajam com ele através de email ou da web e permite que as convenções surjam por elas mesmas na medida em que não força a que haja uma estrutura através do preenchimento de formulários com datas, locais, etc.

Já em [DDK12] é apresentado um sistema, sistema i-CAP, com vista ao aumento da eficiência dos transportes, recorrendo também ao conceito de *carpooling*. O sistema é constituído por conjuntos de condutores e passageiros que estão associados a um conjunto de parâmetros (através dos quais eles próprios se classificam numa escala de 1 a 10). Esses parâmetros estão divididos em 3 áreas: (1) parâmetros da viagem propriamente dita, mais concretamente o local, data e hora de partida e o local de chagada, (2) parâmetros do perfil pessoal, tais como a sua habilidade para conduzir, se fuma, o seu estilo de condução, etc e (3) parâmetros de serviço, como: pontualidade, velocidade, segurança, entre outros. Os passageiros atribuem também uma preferência a cada parâmetro, ou seja, um grau de importância. O *match* entre o passageiro e um condutor é feito através da exploração de conceitos de redes Bayesianas, mais concretamente do modelo Naïve.

O objetivo é selecionar o condutor mais apropriado entre todos os disponíveis. Para isso, a abordagem apresentada é constituída por 2 fases: a fase de descoberta robusta (*robust discovery phase*) e a fase de tomada de decisão (*decision-making phase*). A fase de descoberta robusta tem como objetivo identificar os valores mais prováveis dos parâmetros. Por outras palavras, primeiro são reunidos valores de referência dos parâmetros (através de avaliações) e depois é calculada a probabilidade de cada um desses valores de referência ser o valor do parâmetro (quanto mais próximo do valor do parâmetro estiver, maior probabilidade há). Posteriormente, na fase de tomada de decisão o sistema favorece a seleção de condutores que têm uma probabilidade alta de alcançar os valores dos parâmetros mais apropriados. Para isso é criada uma função objetivo (Função 2.5) para cada condutor.

$$OF_i = \sum_j max(Pr[V_j = rv_{ij}^k | D = i]) \times w_j \tag{2.5}$$

Onde i representa o condutor, j representa um parâmetro, w_j representa o peso atribuído pelo passageiro ao parâmetro j e $Pr[V_j = rv_{ij}^k | D = i]$ significa a probabilidade condicional do parâmetro j alcançar o k-ésimo valor de referência dado que está a ser considerado o condutor i.

[16] http://public.zimride.com/
[17] http://www.galpshare.pt/
[18] http://ridenow.org/

Assim, o condutor escolhido é o que obtiver maior valor no cálculo da função objetivo. No entanto, a seleção só é considerada válida se, e só se, o valor médio dos parâmetros mais prováveis do condutor for > 6.

Discussão O projeto RideNow tem um problema que é identificável logo à cabeça. Este sistema não é, de todo, escalável. Funcionará apenas para pequenos grupos de pessoas e continua a ser completamente manual. Acaba por ser mais do que já existe.

Por outro lado, o sistema i-CAP dá bastante importância a parâmetros como o estilo de condução, a pontualidade, a velocidade, a segurança, etc (que são naturalmente importantes!) mas descura quase na totalidade a rota seguida pelo utilizador. Aliás apenas podem ser feitas sugestões a utilizadores que iniciem e terminem a viagem nos mesmos locais na medida em que não existe qualquer informação sobre a rota seguida por eles (apenas é avaliado de 1 a 10 o uso de estradas primárias; nem quais as estradas usadas são identificadas). Este problema limita muito o âmbito do sistema pois a sua aplicação está circunscrita a um número limitado de situações.

É notória a necessidade de um sistema que permita que o utilizador de forma (quase) imediata verifique as suas possibilidades de partilha de veículo. Esta necessidade está relacionada com a crescente sensibilização das pessoas para a poluição do meio-ambiente e com a necessidade de redução de custos devido à atual crise económico-financeira que assola toda a Europa e não só.

Existe uma oportunidade clara nesta área e o avanço da tecnologia e o aparecimento de vários sensores nos *smartphones* impulsionam, sem dúvida, a exploração deste tipo de sistemas. Campos como *participatory* e *collaborate sensing* tomam, neste caso em concreto, um papel ativo e indispensável.

2.7 Síntese

Neste capítulo foram apresentadas e analisadas as ideias de fundo no que respeita ao âmbito desta dissertação. Foi ainda feito o enquadramento da própria dissertação num projeto mais amplo, o projeto MISC[19].

Os dados geográficos, embora espaciais, têm algumas particularidades que interessa que estejam presentes quando é feita a análise de técnicas e algoritmos de extração de conhecimento neste tipo de dados. Por exemplo, a sua representação e o cálculo da distância entre duas coordenadas geográficas, devido a estas não se encontrarem sobre um plano, são caraterísticas relevantes a ter em conta aquando de uma análise crítica e fundamentada.

A aplicação de técnicas de *data mining* a dados espaciais, normalmente designado por *data mining* espacial, está ainda a dar os primeiros passos. Contudo, o estudo de dados espaciais, incluindo os dados geográficos, como o seu armazenamento é uma área claramente estudada e aprofundada e para a qual existem atualmente imensas soluções e bastante satisfatórias. A exploração deste tipo de dados, ao ponto de extrair informação com valor, é o próximo passo. De forma a possibilitar a exploração adequada dos dados nesta dissertação foram abordadas as caraterísticas

[19] http://www.it.up.pt/misc/

das bases de dados e as estruturas de dados mais adequadas para o tratamento de dados espaciais, mais concretamente os dados geográficos.

Dessa análise resultou a conclusão de que o uso da estrutura de dados R^*-tree afigura-se como uma ótima solução. Conclui-se também que os SGBD atuais já disponibilizam índices do tipo espacial que fazem uso de algumas das estruturas de dados analisadas, nomeadamente R-trees, R^*-tree, B-trees, etc. A vantagem de usar a implementação dos SGBD prende-se com a garantia de escalabilidade e com a reduzida probabilidade de existirem erros na própria implementação.

Estas estruturas de dados permitem um acesso aos dados bastante eficaz em termos temporais. Essa característica possibilita as abordagens estudadas para a determinação de pontos de estadia e para a determinação de locais significativos. A própria aplicação de algoritmos de *clustering* baseados em densidade, como os apresentados, fica facilitada na medida em que é habitual ter de se verificar e aceder aos pontos que se encontram na vizinhança de outros.

A rápida evolução dos *smartphones*, e a melhoria e aumento de qualidade dos sensores que os acompanham, fez com que a recolha de dados do ambiente em que o portador do dispositivo móvel se encontra começasse a fazer parte do nosso quotidiano. Os conceitos de *participatory* e *collaborative sensing* emergiram rapidamente e fazem com que o conhecimento do ambiente envolvente seja ainda mais rico devido à partilha de informação entre dispositivos móveis. Estes fatores aliados ao facto dos sistemas atuais de *carpooling* serem ainda bastante arcaicos impulsionam a apresentação de soluções nesta área.

Capítulo 3

Determinação de Locais Significativos e de Padrões de Viagem

Apenas tendo como ponto de partida informação sobre os locais significativos de cada utilizador e quais os seus padrões de viagem poder-se-ia procurar utilizadores com padrões de viagem semelhantes ou até mesmo iguais. Nesse sentido, é apresentada a solução idealizada para a determinação de locais significativos e de padrões de viagem neste capítulo.

Naturalmente, uma viagem tem sempre um local de início e um de fim. À partida, esses dois locais seriam, respetivamente, o local onde foi iniciada e o local onde foi terminada a recolha de dados. Contudo, pelo meio podem existir paragens suficientemente demoradas que façam com que exista mais do que uma viagem desde que foi iniciada a recolha de dados até ao momento em que foi terminada. Por exemplo, se o utilizador iniciou a recolha de dados quando saiu de casa, foi ao ginásio, onde permaneceu 2 horas, e depois dirigiu-se para a faculdade, é natural que se considerem duas viagens, nomeadamente de casa para o ginásio e do ginásio para a faculdade.

Se admitirmos que um utilizador tem um padrão de viagem, como por exemplo ir todas as segundas-feiras de casa para a faculdade por volta das 10h, então é normal que o mesmo utilizador tenha pelo menos o local onde é a sua casa e o local onde é a faculdade como seus locais significativos. Ou seja, os padrões de viagem de um utilizador são sempre entre locais significativos do mesmo. Esta assunção é plausível devido à definição assumida no âmbito desta dissertação para a expressão "locais significativos". Um local significativo é um local onde o utilizador chega e/ou parte um número de vezes que possibilite dizer que é possível que o utilizador tenha um padrão de viagem com partida e/ou chegada naquele local (ver Capítulo 1 para mais detalhes).

Os dados são recolhidos através do recetor GPS de um *smartphone* e consiste nas coordenadas geográficas dos locais por onde o utilizador viaja (consultar a Secção 6.1 para mais detalhes).

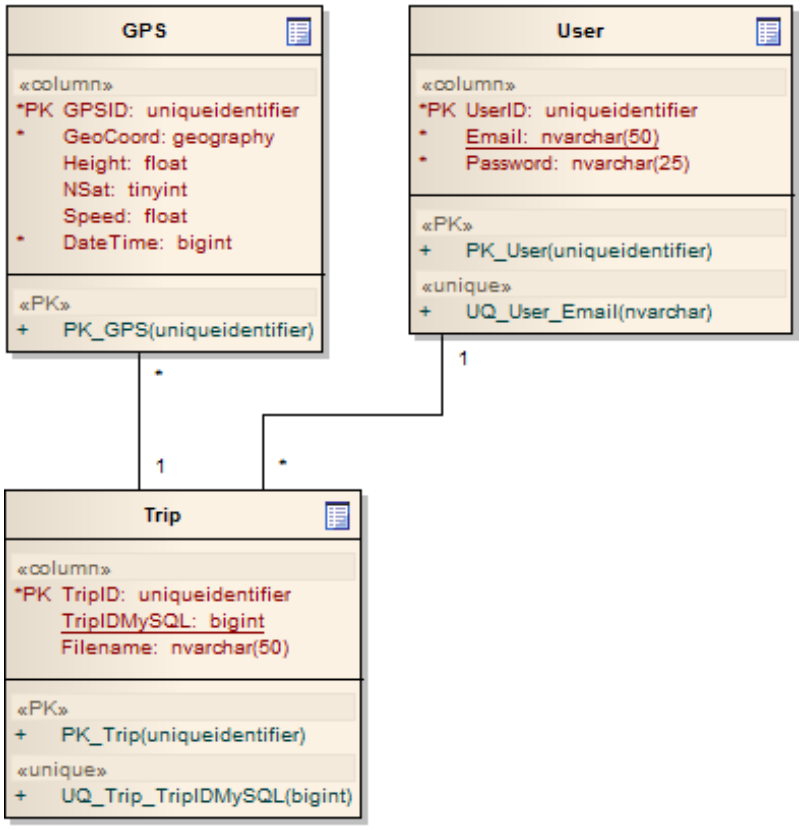


Figura 3.1: Esquema da base de dados (dados em bruto)

Em suma, pode dizer-se que partindo de um conjunto de dados em bruto, Figura 3.1, foram construídos vários níveis de conhecimento com base nos dados recolhidos originalmente e/ou nos dados calculados em etapas anteriores.

3.1 Determinação de Pontos de Estadia

Com vista à deteção dos locais significativos de cada utilizador, são inicialmente detetados os pontos de estadia do mesmo. Sabendo que um local significativo não é mais do que um local onde o utilizador chegou e/ou partiu vezes suficientes, durante o período de tempo em análise, para se poder dizer que é possível que haja um padrão de viagem com partida e/ou chegada lá, urge a necessidade de detetar os locais de partida e chegada de cada viagem de cada utilizador.

Algumas abordagens, descritas na literatura recente e discutidas na Secção 2.4.3, baseiam-se apenas em algoritmos de *clustering* baseados em densidade para a descoberta dos locais significativos dos utilizadores. A motivação para a deteção dos pontos de estadia, como pré-processamento, surge dos problemas enunciados aquando da discussão dessa abordagem, nomeadamente do facto de em locais fechados ser rara a deteção de sinal por parte do sensor GPS. Logo, sempre que o utilizador está num local onde passa muito tempo mas esse local é, por exemplo, dentro de um edifício, esse local não seria detetado na medida em que a densidade de pontos seria igual ou até

menor do que a densidade de pontos no caminho para lá. A deteção de pontos de estadia tem a vantagem, para além de resolver o problema, de não trazer complexidade significativa na medida em que os pontos de estadia podem ser determinados em paralelo com a recolha de dados sem grande esforço em termos computacionais. Nesse sentido, é necessário para além de se guardar a coordenada geográfica a um ritmo predefinido, calcular, de cada vez que é recolhida uma nova coordenada geográfica, a distância à primeira coordenada da série e se for superior a $DistThreh$ calcular o tempo que passou entre as 2 coordenadas. Um novo ponto de estadia é criado se esse intervalo de tempo é superior a $TimeThreh$, de acordo com o Algoritmo 1. Visto que as coordenadas geográficas já tinham de ser recolhidas e armazenadas, a complexidade inserida está apenas nos cálculos da distância e do tempo entre pares de coordenadas. É uma complexidade extremamente reduzida e que permite guardar de forma bastante compacta a sequência de locais visitados pelo utilizador.

Um ponto de estadia é detetado sempre que o utilizador para andar mais de $DistThreh$ demorou mais de $TimeThreh$. Sempre que numa viagem são detetados 2 pontos de estadia consecutivos, considera-se que o utilizador efetuou uma viagem de um ponto para o outro.

De forma a contornar as lacunas encontradas no algoritmo de deteção de pontos de estadia, apresentadas na Secção 2.4.3, e a adaptá-lo ao âmbito desta dissertação foram feitas as seguintes alterações à versão original (versão descrita na Secção 2.4.1):

1. Ponto de estadia no início e no final da recolha de dados. Passam a ser considerados pontos de estadia na primeira e na última coordenadas geográficas recolhidas. Esta decisão baseia-se na ideia de que, tendencialmente, o utilizador ligará e desligará a recolha de dados quando inicia ou termina uma viagem. Pode ligar o recetor GPS de manhã e desligá-lo à noite mas tendencialmente essa ação será feita quando parte ou chega a um local e não a meio do caminho. No entanto, se for feito a meio do caminho não será muito problemático na medida em que muito provavelmente o utilizador não ligará/desligará a recolha de dados sempre no mesmo local. Assim, o que acontecerá é que ficarão alguns pontos de estadia "espalhados" e no processamento seguinte dos dados serão ignorados pois não terão densidade suficiente para serem considerados.

2. Problema quando a recolha de dados é terminada num local onde deve ser detetado um ponto de estadia. A versão original do algoritmo não deteta pontos de estadia no caso do utilizador estar muito tempo num local e depois desligar a recolha de dados (antes de sair desse local). Isto acontece porque o algoritmo deteta um ponto de estadia se para andar mais de $DistThreh$ o utilizador demorou mais de $TimeThreh$. Se o utilizador desligar a recolha de dados mesmo depois de ter estado muito tempo no mesmo local, embora o $TimeThreh$ seja ultrapassado, o ponto de estadia só seria criado se o utilizador recolhesse uma coordenada geográfica com uma distância maior do que $DistThreh$ em relação à primeira coordenada naquele local. Se o utilizador desligar a recolha de dados, o ponto de estadia simplesmente não é detetado. A alteração descrita no ponto anterior serve também como solução para este problema.

3. Tratamento dos dados em caso de passagem em túneis. Nos casos em que o utilizador passava num túnel grande (fazendo uma viagem de metro subterrâneo, por exemplo) era detetado um ponto de estadia na última coordenada geográfica recolhida antes do utilizador perder o sinal

GPS. Isto acontecia porque a coordenada seguinte era, por exemplo, a 2km de distância (superior a $DistThreh$) e a diferença temporal entre as duas coordenadas era superior a $TimeThreh$. Neste caso, o utilizador para se deslocar 2km tinha demorado mais de $TimeThreh$, logo era criado um ponto de estadia. Para evitar este problema é feita a verificação deste caso particular. São ignorados os casos em que entre duas coordenadas geográficas recolhidas (ordenadas de forma crescente por tempo) há uma distância superior a 2km e um intervalo de tempo superior a 2min. Nestes casos, como houve uma perda de sinal GPS, nada se pode concluir e, portanto, não é extraído nenhum tipo de informação. Embora a recolha de dados armazene as coordenadas geográficas a cada segundo, os dados fornecidos aos algoritmos têm uma granularidade de 20 segundos (devido à redução da complexidade temporal dos algoritmos). Para se deslocar 2km em 20seg necessita-se de uma velocidade média de 360km/h $\left(\frac{3600seg \times 2km}{20seg}\right)$. Daí resulta o valor de 2km pois 360km/h é uma velocidade que muito dificilmente será atingida nas viagens de rotina dos utilizadores. Muito provavelmente sempre que exista uma coordenada com uma distância superior a 2km em relação à anterior é porque houve perda de sinal GPS.

Algoritmo 4 Algoritmo MyStayPoint_Detection(*P, distThreh, timeThreh*)

1: $i = 0$; $pointNum = |P|$; //the number of GPS points in a GPS log
2: **while** $i < pointNum$ **do**
3: $j = i + 1$;
4: **while** $j < pointNum$ **do**
5: //se entre 2 coords foram percorridos > 2km (ou seja, perdeu o sinal GPS)
6: **if** $Distance(p_{j-1}, p_j) > 2km$ **then**
7: $i = j$; *break*; //ignora os dados (não é possível tirar conclusões)
8: **end if**
9: $dist = Distance(p_i, p_j)$; //calculate the distance between two points
10: **if** $dist > distThreh$ **then**
11: $\Delta T = p_j.T - p_i.T$; //calculate the time span between two points
12: **if** $\Delta T > timeThreh$ **then**
13: $S.coord = ComputeMeanCoord(\{p_k \mid i \leq k \leq j\})$;
14: $S.arvT = p_i.T$; $S.levT = p_j.T$;
15: $SP.insert(S)$;
16: **end if**
17: $i = j$; *break*;
18: **end if**
19: $j = j + 1$;
20: **end while**
21: **end while**
22: //se não foi criado um ponto de estadia no local onde foi iniciada a viagem
23: **if** $|SP| == 0 \;||\; Distance(SP[0], p_0) > distThreh$ **then**
24: //guarda um novo ponto de estadia (onde foi iniciada a viagem)
25: $S.coord = p_0$;
26: $S.arvT = NULL$; $S.levT = p_0.T$;
27: $SP.insert(S)$;
28: **end if**
29: //se não foi criado um ponto de estadia no local onde foi terminada a viagem
30: **if** $Distance(SP[|SP| - 1], p_{|p|-1}) > distThreh$ **then**
31: //guarda um novo ponto de estadia (onde foi terminada a viagem)
32: $S.coord = p_{|p|-1}$;
33: $S.arvT = p_{|p|-1}.T$; $S.levT = NULL$;
34: $SP.insert(S)$;
35: **end if**
36: *return SP*;

O Algoritmo 4 representa o algoritmo de deteção de pontos de estadia implementado. Os três parâmetros do algoritmo são os mesmos que os do algoritmo original. O parâmetro *P* consiste nas

coordenadas geográficas recolhidas e de onde serão extraídos os pontos de estadia e os parâmetros *distThreh* e *timeThreh* são o limite de distância e de tempo respetivamente. O *distThreh* representa a distância máxima a que as coordenadas podem estar do ponto inicial da região para fazerem parte do ponto de estadia e o *timeThreh* representa o mínimo de tempo que o utilizador tem de estar na região para que o conjunto de pontos dentro dela contribuam para um ponto de estadia.

O algoritmo tem um funcionamento muito semelhante ao original (apresentado na Secção 2.4.1, Algoritmo 1) mas foram-lhe adicionadas as funcionalidades descritas anteriormente de forma a abarcar as melhorias enunciadas. Entre a linha 5 e a linha 8 é tratado o caso dos túneis grandes (em que o utilizador fica sem sinal GPS durante um "longo" período de tempo). Já entre as linhas 22 e 35 é garantido que se porventura não tiver sido detetado um ponto de estadia nos locais de início e fim da recolha de dados, passa a haver lá um.

(a) Pontos de estadia (b) Foco no local com a circunferência vermelha

Figura 3.2: Exemplo de pontos de estadia de um utilizador

Na Figura 3.2a são visíveis vários pontos de estadia ao longo da cidade. Fazendo *zoom* junto de cada um deles percebe-se que alguns estão isolados e outros não. Na Figura 3.2b foi feito *zoom* no local circundado pela circunferência vermelha na Figura 3.2a e é percetível que é costume o utilizador partir e/ou chegar àquele local. Na realidade aquele é o local de trabalho do utilizador que originou os pontos de estadia da imagem. São visíveis muitos pontos de estadia que estão isolados e estão espalhados pela cidade, muitos deles existem porque o utilizador deslocou-se àquele local para fazer uma tarefa não rotineira. Outra razão possível é ter iniciado ou terminado a recolha de dados a meio de uma viagem.

O objetivo da deteção dos pontos de estadia consiste numa forma bastante *leve* de representação dos locais de e para onde o utilizador viajou em cada viagem. Para além dessa representação compacta, permite ainda a deteção de pontos mesmo onde o sinal GPS não é recebido (escritórios, salas de aula, ginásio, casa, etc).

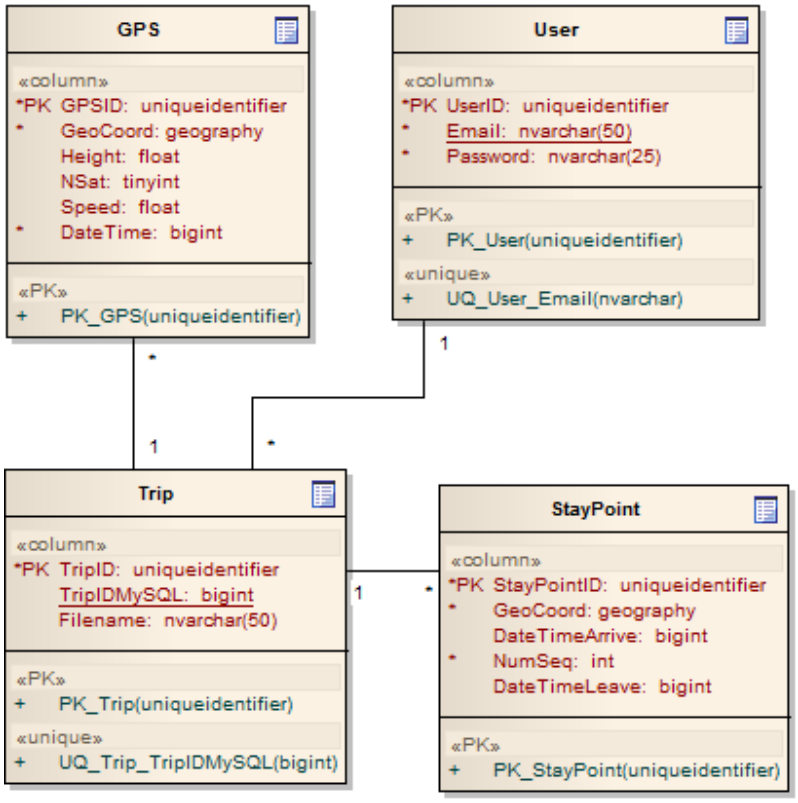


Figura 3.3: Esquema da base de dados (novo nível de conhecimento: pontos de estadia)

Existe então, a partir deste momento, um primeiro nível de conhecimento que é adicionado ao conjunto de dados em bruto. A Figura 3.3 representa a adição da tabela StayPoint.

3.2 Determinação de Locais Significativos

Um ponto de estadia, por si só, não permite determinar que aquele local é um local significativo do utilizador. Contudo, é natural que nos locais significativos de um utilizador exista uma densidade significativa de pontos de estadia. Na verdade, desde que um determinado local possua pelo menos tantos pontos de estadia quanto o número mínimo de viagens necessárias para ser considerado um padrão de viagem, já é considerado um local significativo. Por outras palavras, se está a ser feita uma análise das últimas 8 semanas e se se considera que para haver um padrão de viagem tem de haver pelo menos 5 viagens no mesmo dia da semana, mais ou menos à mesma hora e com partida e chegada iguais, e se há um local com 5 pontos de estadia de chegada, ou mais, então é um local significativo. Ou seja, é um local para onde é possível que haja pelo menos um padrão de viagem.

A deteção de locais significativos é feita com recurso ao algoritmo DBSCAN, descrito na Secção 2.4.2.1. O algoritmo encarrega-se de determinar os locais onde existe uma densidade de pontos de estadia superior a um certo limite. Neste caso, o limite é definido pelo número de viagens usado para se considerar que existe um padrão de viagem.

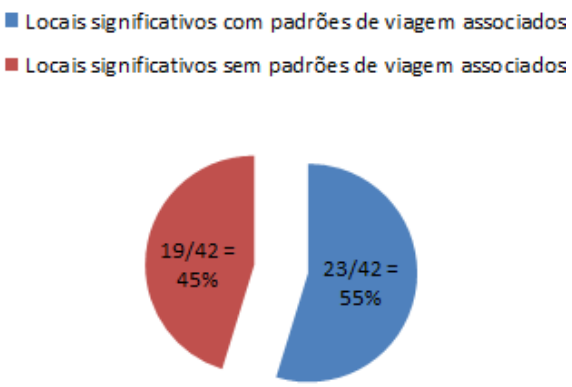


Figura 3.4: Distribuição de locais significativos com e sem padrões de viagem associados

Esta abordagem pode gerar locais significativos que, posteriormente, não terão nenhum padrão de viagem associado. No entanto, não descarta nenhum local de e/ou para onde possa haver padrões de viagem. Teoricamente corre-se o risco de gerar demasiados locais significativos e isso poderia ter influência negativa no desempenho em termos temporais. Contudo, essa acaba por ser uma falsa questão. Vejamos, o algoritmo DBSCAN tem uma complexidade temporal *O(n * log n)*, onde *n* representa o número de pontos que dão origem aos *clusters* (ver Secção 2.4.2.1). Neste caso em concreto, o *n* representa o número de pontos de estadia das viagens de um utilizador no intervalo de tempo em análise. Independentemente do número de *clusters* criados, ou seja, do número de locais significativos determinados para cada utilizador, a complexidade temporal é a mesma na medida em que cada ponto de estadia tem de ser "visitado" uma, e uma só, vez. Posto isto, a geração de locais significativos que posteriormente não tenham nenhum padrão de viagem associado não causa complexidade adicional.

Nos dados recolhidos por cerca de 25 utilizadores entre 27 de Fevereiro de 2012 e 15 de Abril de 2012 (detalhados na Secção 6.1) foram detetados 42 locais significativos. Desses 42, 23 tinham padrões de viagem associados, ou seja, cerca de 55% (Figura 3.4).

O facto de calcular os pontos de estadia a priori tem, para além das vantagens enunciadas na secção anterior, a vantagem de reduzir extremamente a complexidade temporal aquando da aplicação do algoritmo DBSCAN. O algoritmo, por si só, já tem uma complexidade temporal significativa e no caso de ser aplicado sobre as coordenadas geográficas recolhidas ao invés dos pontos de estadia extraídos dessas coordenadas seria incomportável. A título de exemplo, entre 06 de Fevereiro de 2012 e 01 de Abril de 2012 foram recolhidas 1.018.184 coordenadas geográficas e detetados 1.948 pontos de estadia, ou seja, menos de 1% de pontos de estadia face ao número de coordenadas geográficas recolhidas. Por outras palavras, reduziu-se o *n* da complexidade temporal do algoritmo DBSCAN para cerca de 0,19% $\left(\frac{1.948}{1.018.184} = 0,001913\right)$.

Figura 3.5: Locais significativos do utilizador

Na Figura 3.5 estão os locais significativos que foram originados pelos pontos de estadia da Figura 3.2a. Os locais significativos determinados são, neste caso em concreto, a casa e o local de trabalho do utilizador. Houve inúmeros pontos de estadia que foram descartados por não haver a densidade mínima requerida no local físico onde estavam. Os pontos de estadia que ficaram atribuídos a um *cluster* originaram uma única coordenada geográfica (o centróide dos pontos de estadia desse *cluster*) que identifica o local físico do local significativo.

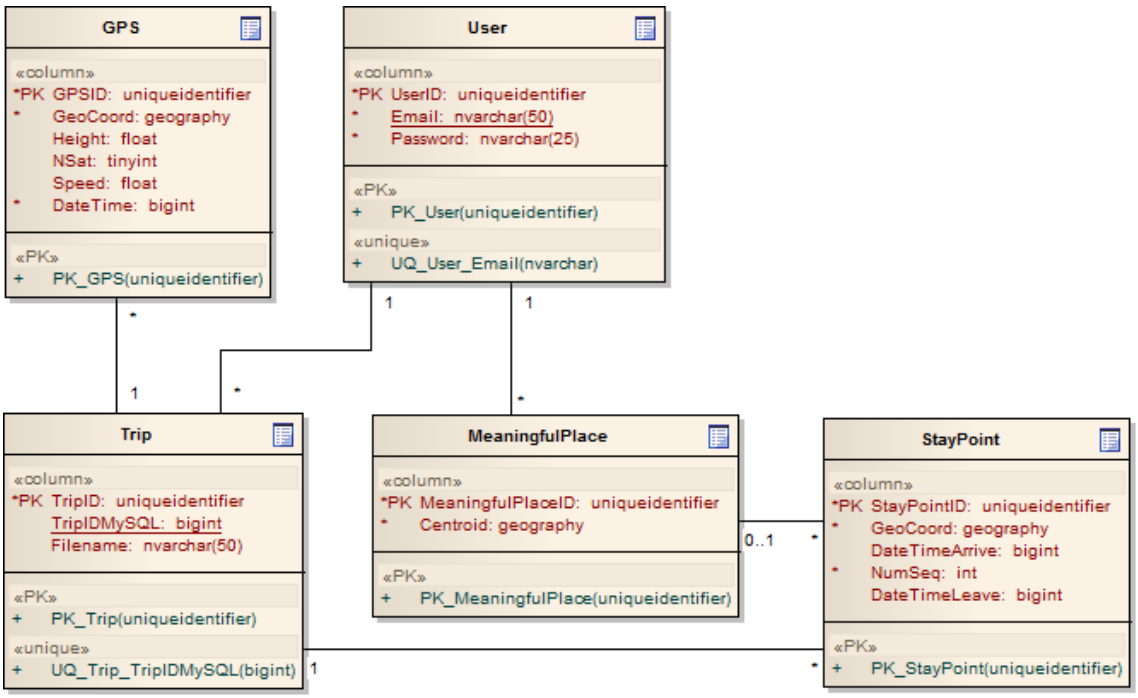


Figura 3.6: Esquema da base de dados (novo nível de conhecimento: locais significativos)

Para facilitar a tarefa seguinte, que consiste na determinação dos padrões de viagem, para além de serem guardados os locais significativos são também guardados os pontos de estadia que

lhes deram origem. No passo seguinte, o objetivo passa por descobrir as sequências de locais significativos que se repetem mais que *threshold* vezes. Isso é determinado de uma forma relativamente simples com recurso a um agrupamento por local significativo de partida e de chegada, nomeadamente recorrendo à cláusula *GROUP BY* em SQL (ver Secção 3.3).

É assim extraído mais um nível de conhecimento dos dados e são armazenados os locais significativos de cada utilizador. A Figura 3.6 dá conta disso e apresenta já toda a informação necessária para determinar os padrões de viagem dos utilizadores.

3.3 Determinação de Padrões de Viagem

É a partir dos locais significativos e das rotas do utilizador que se extraem os respetivos padrões de viagem. Esta extração ficou mais simples a partir do momento em que cada local significativo passou a "saber" os pontos de estadia que lhe deram origem e vice-versa.

Desta forma, com uma ordenação dos pontos de estadia de cada viagem, de forma crescente pela hora a que se realizou, é possível extrair a sequência de locais significativos na respetiva viagem. Um *output* possível é $LocalSignificativo1 \rightarrow LocalSignificativo2 \rightarrow NULL \rightarrow LocalSignificativo1$. Com base no *output* apresentado, conclui-se que o utilizador saiu de um local significativo para outro, permaneceu lá tempo suficiente para ser criado um ponto de estadia, depois foi a outro local onde não é frequente ir (local descrito como *NULL*), permaneceu lá tempo suficiente para ser criado um novo ponto de estadia e, por fim, voltou ao local significativo de onde tinha saído ao início.

Depois de ter todas as viagens entre locais significativos de um utilizador, elas são agrupadas de acordo com os locais de partida e chegada e o dia da semana (inicialmente é ignorada a hora a que se realizaram as viagens). Posteriormente, são descartados todos os pares de locais significativos que não respeitem o número de viagens necessárias para se considerar um padrão de viagem. Ou seja, se só há duas viagens do local significativo A para o B (e o mínimo requerido é 5) pode descartar-se essa sequência de locais significativos.

No passo seguinte é dada atenção à hora a que foram realizadas as viagens. Um par de locais significativos pode ter um número de viagens superior ao mínimo requerido mas se as viagens estiverem espalhadas pelas 24 horas do dia pode não ser possível dizer que é normal que o utilizador viaje a uma determinada hora entre aqueles dois locais significativos (é apenas possível dizer que é normal o utilizador viajar entre aqueles dois locais significativos mas não existe um padrão em termos da hora a que acontecem essas viagens). Assim, as viagens são agrupadas segundo a hora a que se realizaram. O agrupamento é feito com base num parâmetro (configurável) que define o intervalo máximo de tempo entre cada viagem do mesmo grupo. Ou seja, se se definir que o intervalo máximo é 120 minutos, dentro de um determinado grupo de viagens não há nenhuma viagem que diste de qualquer outra em mais de 120 minutos. Todos os grupos com um número de viagens que não respeite o número mínimo requerido para se considerar um padrão de viagem são também descartados. Destes grupos resultam os padrões de viagem de cada utilizador. Cada padrão de viagem é caracterizado por dois locais significativos, os locais de partida e chegada, a

hora média de partida, a hora média de chegada, o desvio padrão da hora de partida e o desvio padrão da hora de chegada.

Um par de locais significativos, no mesmo dia da semana, pode, naturalmente, ter mais do que um grupo. Isso significa que o utilizador costuma viajar entre aqueles dois locais significativos mais que uma vez naquele dia da semana. Contudo, isso não será detetado nos casos em que o utilizador faça as duas viagens com uma diferença temporal inferior ao parâmetro que define o intervalo máximo de tempo entre cada viagem do mesmo grupo. Por outras palavras, no caso hipotético do parâmetro que define o intervalo máximo ser 90 minutos e o utilizador sair de casa para o trabalho por volta das 9h, voltar por volta das 13h para almoçar e fazer o trajeto novamente por volta das 14h para ir trabalhar, não haverá problema pois entre as 9h e as 14h há mais de 90 minutos. Serão criados 2 grupos (um com média de partida $\cong$ 9h e outro com média de partida $\cong$ 14h). Mas no caso em que o utilizador faça o trajeto por volta das 9h e, por alguma razão, volte a fazê-lo por volta das 10h, então as viagens pertencerão todas ao mesmo grupo (com média de partida $\cong$ 9h30m).

Figura 3.7: Padrões de viagem do utilizador

Com os dados calculados para cada grupo é possível descrever as distribuições de partida e chegada. Um utilizador tem, tipicamente, uma hora a que costuma sair de casa para ir trabalhar. Mais uns minutos menos uns minutos, há uma hora a que costuma sair de casa. A probabilidade de sair mais cedo ou mais tarde é tanto menor quanto maior for a diferença para a hora a que costuma sair, ou seja, a média da hora de saída. Nesta dissertação é feita a assunção de que as horas de partida e chegada das viagens de rotina de cada utilizador são descritas pela distribuição normal (ver Anexo A.1 para mais detalhes).

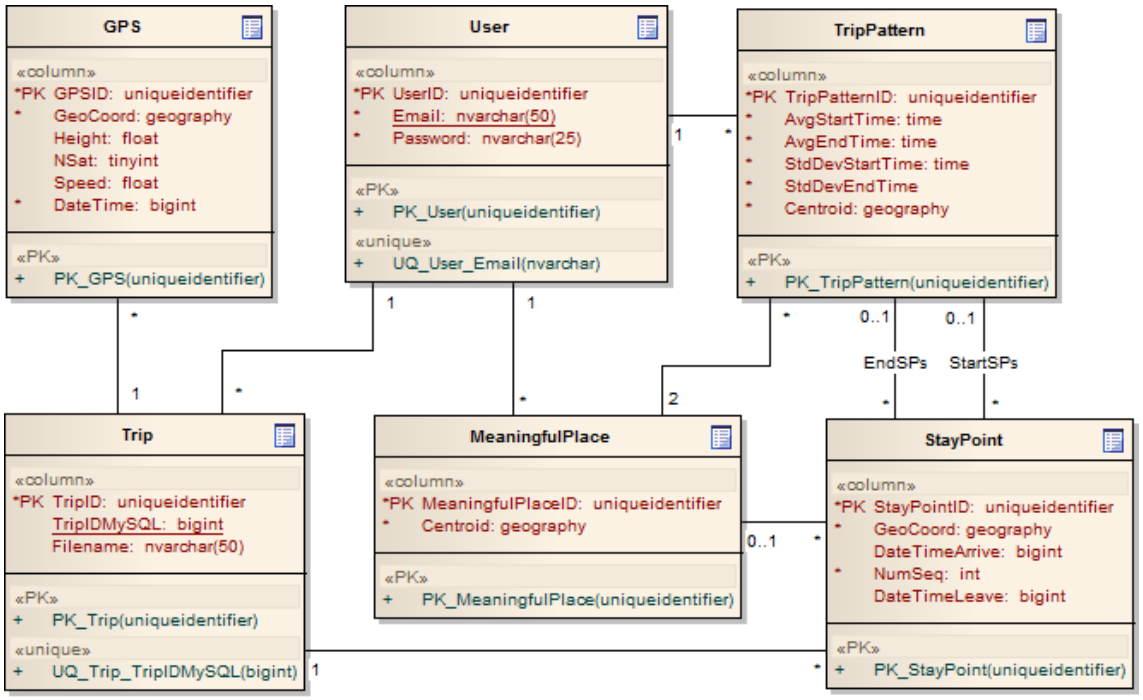


Figura 3.8: Esquema da base de dados (novo nível de conhecimento: padrões de viagem)

Um novo nível de conhecimento, os padrões de viagem de cada utilizador, são extraídos e armazenados (Figura 3.8). É sobre este nível de conhecimento que a procura de sugestões de partilha de veículo trabalha. As sugestões serão feitas única e exclusivamente entre padrões de viagem de utilizadores e a ideia consiste na exploração da rota desses padrões de viagem. Enquanto nesta primeira fase foi necessário descobrir os locais e o fluxo de movimento entre esses locais (deixando de parte a rota propriamente dita seguida nesse fluxo), na próxima fase são estudadas as rotas no sentido de serem encontradas partilhas dessas mesmas rotas.

Os centróides dos padrões de viagem são a coordenada geográfica média dos locais de partida e chegada do padrão de viagem e dos centróides dos segmentos das viagens (ver Secção 3.3) que fazem parte do padrão de viagem.

Cada padrão de viagem é caraterizado para além dos dias da semana em que ocorre e das caraterísticas necessárias para descrever a distribuição normal da hora de partida e chegada, por um centróide. Este centróide serve para, aquando da procura de sugestões de partilha de veículo, evitar que sejam comparados padrões de viagem que distam, por exemplo, 100km um do outro. Este primeiro filtro é calculado com base nos centróides de cada padrão de viagem.

Segmentação das Viagens O principal propósito da segmentação das viagens não é, nem de perto nem de longe, o cálculo do centróide dos padrões de viagem. O seu principal foco é ajudar a perceber quais as rotas que são parcialmente partilhadas. Contudo, foi aproveitado para ajudar no cálculo do centróide de cada padrão de viagem.

A segmentação das viagens permite apresentar a rota seguida pelo utilizador de uma forma muito mais compacta (perdendo detalhe). Cada viagem é dividida em vários segmentos. Cada segmento é constituído pelos seus *DateTimes* de início e de fim e pelo centróide das coordenadas geográficas que deram origem àquele segmento. Desta forma, se se olhar para a sequência dos centróides dos segmentos que constituem uma viagem, é possível analisar a rota (com um detalhe limitado) seguida pelo utilizador.

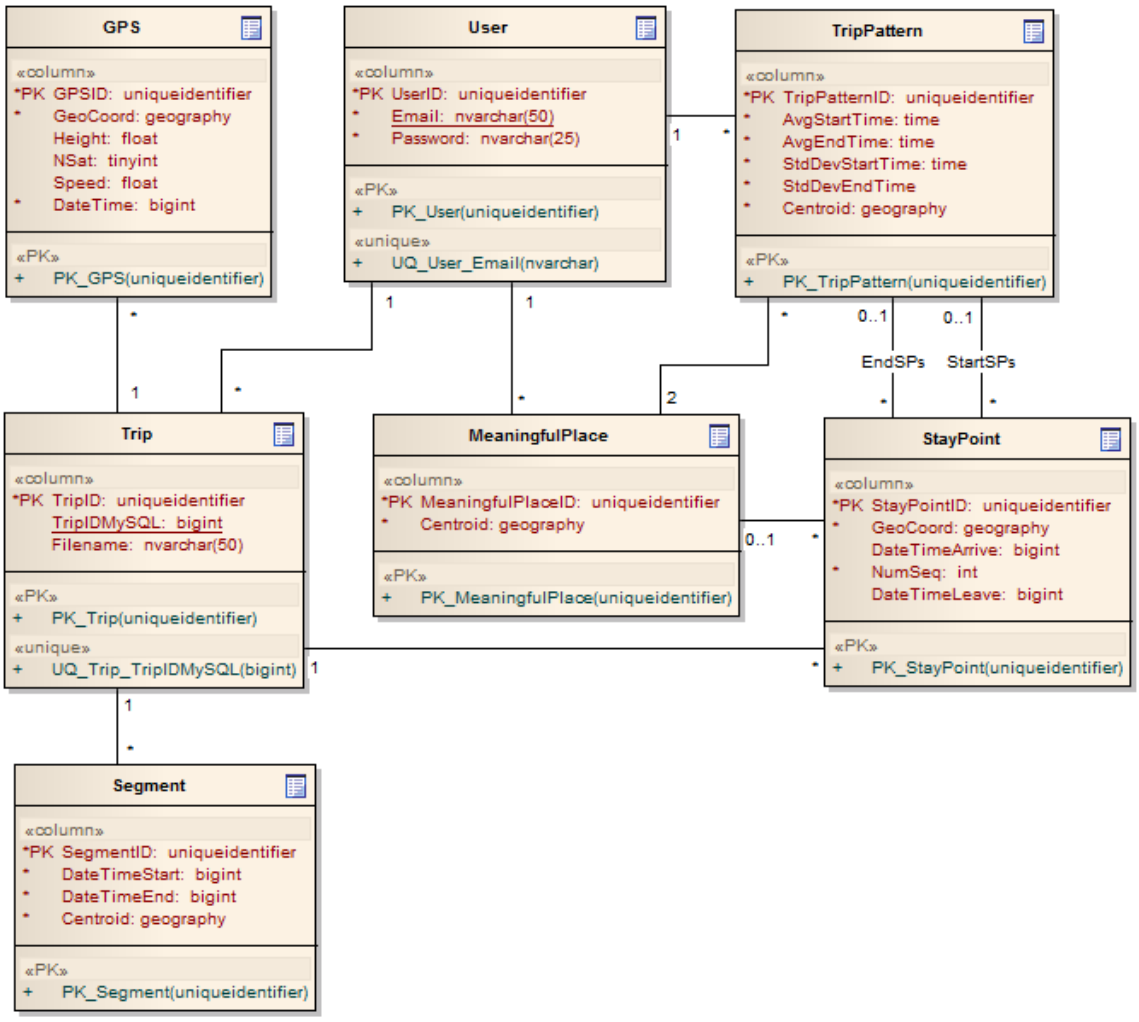


Figura 3.9: Esquema da base de dados (novo nível de conhecimento: segmentos das viagens)

Nos casos em que ambos os utilizadores iniciam e terminam as respetivas viagens em locais diferentes mas partilham parcialmente a rota (Figura 3.10), a segmentação das viagens toma um papel importantíssimo.

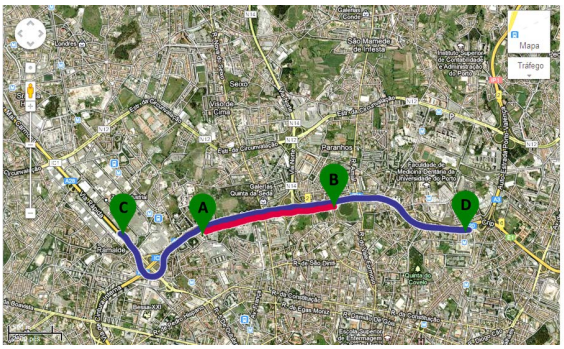


Figura 3.10: Rotas partilhadas parcialmente e com locais de início e fim de viagem distintos

É pela proximidade de alguns segmentos das rotas que é determinado se essas mesmas rotas são partilhadas pelos dois utilizadores ou não. Esta foi a principal motivação para a segmentação das viagens.

Para o cálculo dos segmentos é usado um parâmetro (configurável) que define a distância máxima para a criação de um segmento. No entanto, esta distância não é a distância percorrida pelo utilizador mas sim a distância ao primeiro ponto do segmento. Ou seja, se o parâmetro estiver definido para criar segmentos a cada 2km, e se o utilizador se deslocar inicialmente 1km em linha reta e a partir daí se deslocar segundo uma circunferência, haverá apenas um segmento pois o utilizador está sempre a 1km de distância do ponto inicial do segmento (onde iniciou a viagem). O cálculo é feito desta forma na medida em que a ideia subjacente ao cálculo dos segmentos é a extração da rota seguida pelo utilizador. Assim, se o utilizador se tiver deslocado 20km mas sempre no mesmo local (segundo uma circunferência com raio de 1km) não tem interesse ter uma sequência de segmentos demasiado perto uns dos outros.

À semelhança do cálculo dos pontos de estadia, esta abordagem tem a vantagem de não ser muito custosa na medida em que estes segmentos podem ser calculados em tempo real aquando da recolha das coordenadas geográficas. As coordenadas já tinham de ser armazenadas e, portanto, o custo acrescido está relacionado com o cálculo da distância de cada coordenada geográfica recolhida à primeira coordenada do segmento atual. Quando essa distância ultrapassa o *threshold* definido é calculada a média das coordenadas geográficas que pertencem àquele segmento. A complexidade temporal deste algoritmo é, portanto, linear, $O(n)$.

3.4 Sumário

Na fase da determinação de locais significativos, juntaram-se dois algoritmos: pontos de estadia e locais significativos. Inicialmente, como pré-processamento, são extraídos os pontos de estadia das viagens. Logo aí são descartados muitos dados que não têm interesse (pelo menos nesta fase), nomeadamente as coordenadas geográficas recolhidas quando o utilizador está a deslocar-se entre os locais. Ao conjunto de dados resultante é então aplicado o algoritmo de *clustering* baseado em densidade, DBSCAN, que agrupará os pontos de estadia que pertencem ao mesmo

local físico e descartará os locais que são visitados com uma frequência insuficiente para poder haver um padrão de viagem de e/ou para lá. O facto do algoritmo de *clustering* ter como conjunto de dados apenas os pontos de estadia faz com que, em termos computacionais, seja bastante mais eficiente do que se tivesse de trabalhar com todas as coordenadas geográficas recolhidas pelo utilizador.

A descoberta dos padrões de viagem, por sua vez, baseiam-se no agrupamento das viagens entre os pares de locais significativos encontrados nas diversas viagens de cada utilizador. Este agrupamento tem em conta o dia da semana em que ocorrem as viagens e se esses pares (origem $\rightarrow$ destino) se repetirem com uma frequência superior a um *threshold* (configurável) no mesmo dia da semana são agrupadas as viagens consoante a hora a que se realizaram. O agrupamento pela hora a que cada viagem se realizou pode originar mais do que um *cluster* em cada dia da semana para cada par de locais significativos. Os *clusters* que tenham um número de viagens que não obedeça ao *threshold* são eliminados e para os restantes são calculados os parâmetros que caracterizam uma distribuição normal para as horas de partida e chegada das viagens, nomeadamente a média e o desvio padrão.

De forma a compactar a informação sobre a rota seguida em cada viagem, as viagens são segmentadas e é calculado o centróide de cada um desses segmentos. A sequência desses centróides, ordenando os segmentos de forma crescente pelo sua data, transmitem uma ideia, sem muito detalhe, da rota seguida pelo utilizador. O principal objetivo deste cálculo é o auxílio à determinação das rotas que são parcialmente partilhadas, ou seja, que têm centróides de alguns segmentos das suas viagens relativamente perto uns dos outros.

Capítulo 4

Determinação de Sugestões de Partilha de Veículo

Depois de conhecer os padrões de viagem de cada utilizador surge a necessidade de perceber a rota seguida em cada um deles de forma a ser possível a comparação das mesmas e a sugestão de partilha de veículo sempre que possível. Neste capítulo é então detalhada a solução idealizada para a determinação de sugestões de partilha de veículo.

A dificuldade em estudar as rotas prende-se com a enorme quantidade de dados com que se tem de trabalhar. Um padrão de viagem que tenha 45 minutos e com um ritmo de recolha de dados de 1 segundo resulta em 2.700 coordenadas geográficas. A comparação é feita com outro padrão de viagem e se esse também tiver 2.700 coordenadas geográficas resultam 5.400. É incomportável, em termos computacionais, comparar coordenada a coordenada cada padrão de viagem de utilizadores diferentes. Deste entrave, surgiu novamente a necessidade de compactar informação. As formas de representação mais compactas encontradas ao longo do tempo ficam naturalmente aquém do detalhe dos dados em bruto. Todavia permitem uma análise viável em termos temporais e espaciais.

Existem dois tipos de sugestões: sugestões com pelo menos um local significativo partilhado por ambos os utilizadores (caso em que ambos os utilizadores partem do mesmo local, por exemplo) e sugestões em que ambos os utilizadores partem e chegam a locais diferentes mas têm uma parte significativa da rota partilhada.

4.1 Agrupamento de Padrões de Viagem

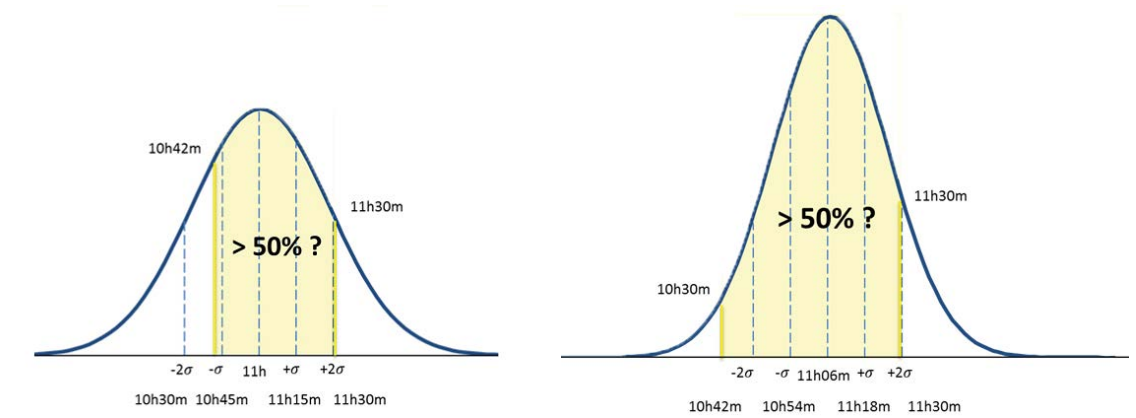
Se um utilizador tem padrões de viagem entre os mesmos locais significativos todos os dias da semana por volta da mesma hora, a rota seguida é muito provavelmente a mesma. É natural que o utilizador quando vai de casa para o trabalho para além de sair praticamente sempre mais ou

menos à mesma hora, siga a mesma rota. Cada padrão de viagem é então representado por uma única rota.

Se a rota é a mesma, faz sentido agrupar esses padrões de viagem de forma a que não existam comparações desnecessárias. Se um padrão de viagem partilha rota com outro, logicamente partilhará com todos os restantes que sigam a mesma rota.

Assim, é verificado para cada utilizador se tem padrões de viagem com origem e destino iguais e com partida e chegada a horas semelhantes. Em caso afirmativo, para validar se a rota é a mesma, é verificado se os centróides dos dois padrões de viagem estão a uma distância inferior a um *threshold* configurável. Se assim for, considera-se que as rotas são iguais e os padrões de viagem são agrupados.

Falta, no entanto, definir o que se entende por um padrão de viagem ter horas de partida e chegada semelhantes às de outro padrão de viagem. No âmbito desta dissertação, diz-se que um padrão de viagem tem uma hora de partida semelhante à de outro se tem uma probabilidade de partida maior ou igual a 50% no intervalo de tempo correspondente à probabilidade de 95% de partida do outro padrão de viagem, ou seja, de -2σ a $+2\sigma$ (ver Anexo A.1 para mais detalhes). Por outras palavras, se há um padrão com hora média de partida às 11h e com desvio padrão de 15 minutos e outro padrão com hora média de partida às 11h06m e com desvio padrão de 12 minutos, considera-se que a hora de partida dos dois padrões é semelhante se a probabilidade de partida no primeiro padrão entre as 10h42m e as 11h30m for maior ou igual a 50% e se acontecer o mesmo entre as 10h30m e as 11h30m no segundo padrão (como é visível na Figura 4.1).



(a) Padrão de viagem com $\mu = 11h$ e $\sigma = 15min$

(b) Padrão de viagem com $\mu = 11h06m$ e $\sigma = 12min$

Figura 4.1: Exemplo de determinação se 2 padrões de viagem são considerados com horas semelhantes

Se esta regra for verdadeira para ambos os padrões de viagem (considerando a distribuição normal das horas de partida e chegada), considera-se que tanto a hora de partida como a hora de chegada dos dois padrões são semelhantes.

Este cálculo é feito recorrendo às funções densidade de probabilidade de cada acontecimento

(ver Anexo A.1 para mais informações). De forma a calcular a probabilidade, ou seja, a área formada pela função densidade de probabilidade num determinado intervalo foi usado um método numérico (ver Anexo A.2 para mais informações), mais concretamente, o método de Simpson (ver Anexo A.2.2 para mais informações). O método de Simpson foi o escolhido para o cálculo da probabilidade na medida em que a relação entre a complexidade espacial e temporal e o erro cometido é aceitável no âmbito desta dissertação. Um dos defeitos da regra dos trapézios (descrita com detalhe no Anexo A.2.1) é o de cometer um erro sistemático em intervalos em que a segunda derivada da integranda mantém sinal constante. A regra de Simpson, de forma a evitar este problema, em vez de substituir a curva pelas cordas definidas por cada par de pontos consecutivos, substitui-a pelas parábolas definidas por cada trio de pontos consecutivos (ver Anexo A.2 para mais detalhes).

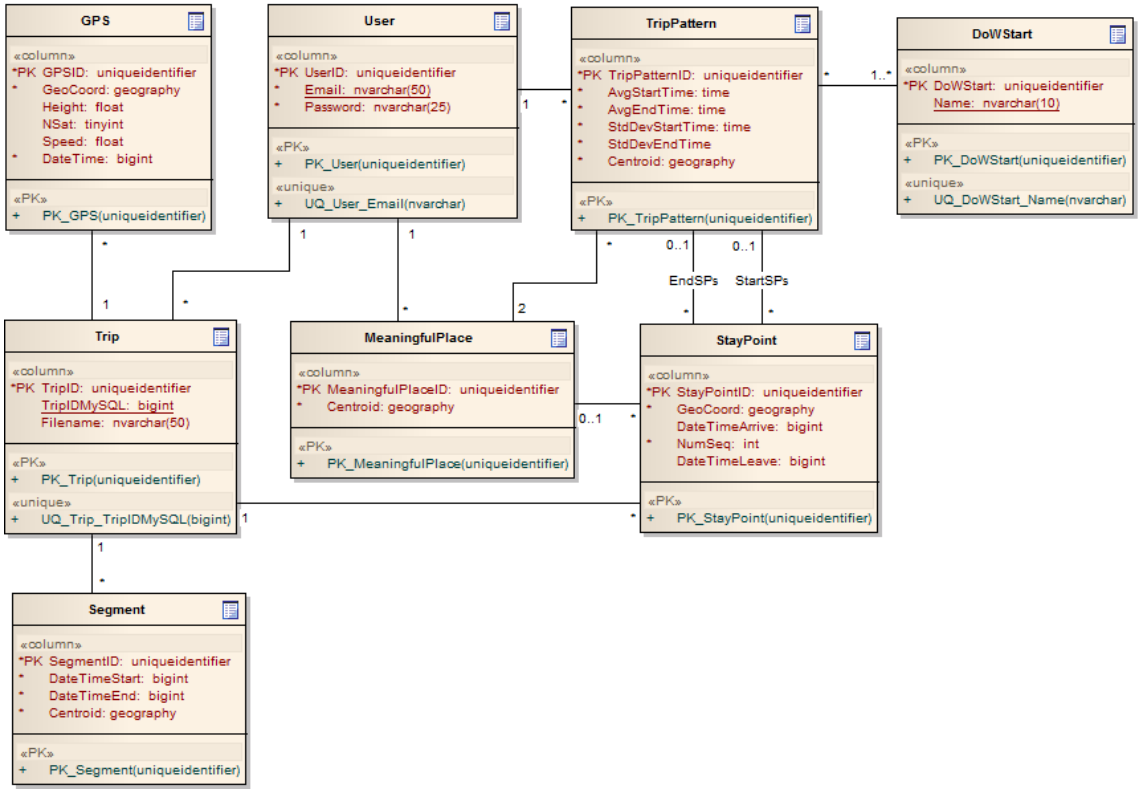


Figura 4.2: Esquema da base de dados (associação dos padrões de viagem a dia(s) da semana)

Os padrões de viagem passaram a estar associados a um ou mais dias da semana e, portanto, a estrutura da base de dados relacional (Figura 4.2) evoluiu nesse sentido.

4.2 Determinação de Padrões de Viagem Vizinhos

A comparação entre todos os pares de padrões de viagem é claramente inviável do ponto de vista temporal. Embora o centróide de cada padrão de viagem sirva para despistar os padrões

de viagem claramente distantes, essa informação não é suficiente quando é necessário aumentar o foco da análise. É possível distinguir padrões de viagem no Porto de padrões de viagem em Aveiro por exemplo, mas se se olhar somente para "dentro" do Porto ou de Aveiro, existem ainda demasiados padrões de viagem para ser comparados entre si. E, pior do que isso, muitos deles não têm nada em comum, o que faria com que a sua comparação constituísse um desperdício de recursos.

De forma a ultrapassar este obstáculo, são calculados para cada padrão de viagem os seus vizinhos. Existem dois tipos de vizinhos:

- **Vizinhos do tipo 1:** padrões que têm início e/ou fim relativamente perto um do outro
- **Vizinhos do tipo 2:** padrões que embora não tenham início e fim perto, têm partes da rota relativamente perto uma da outra

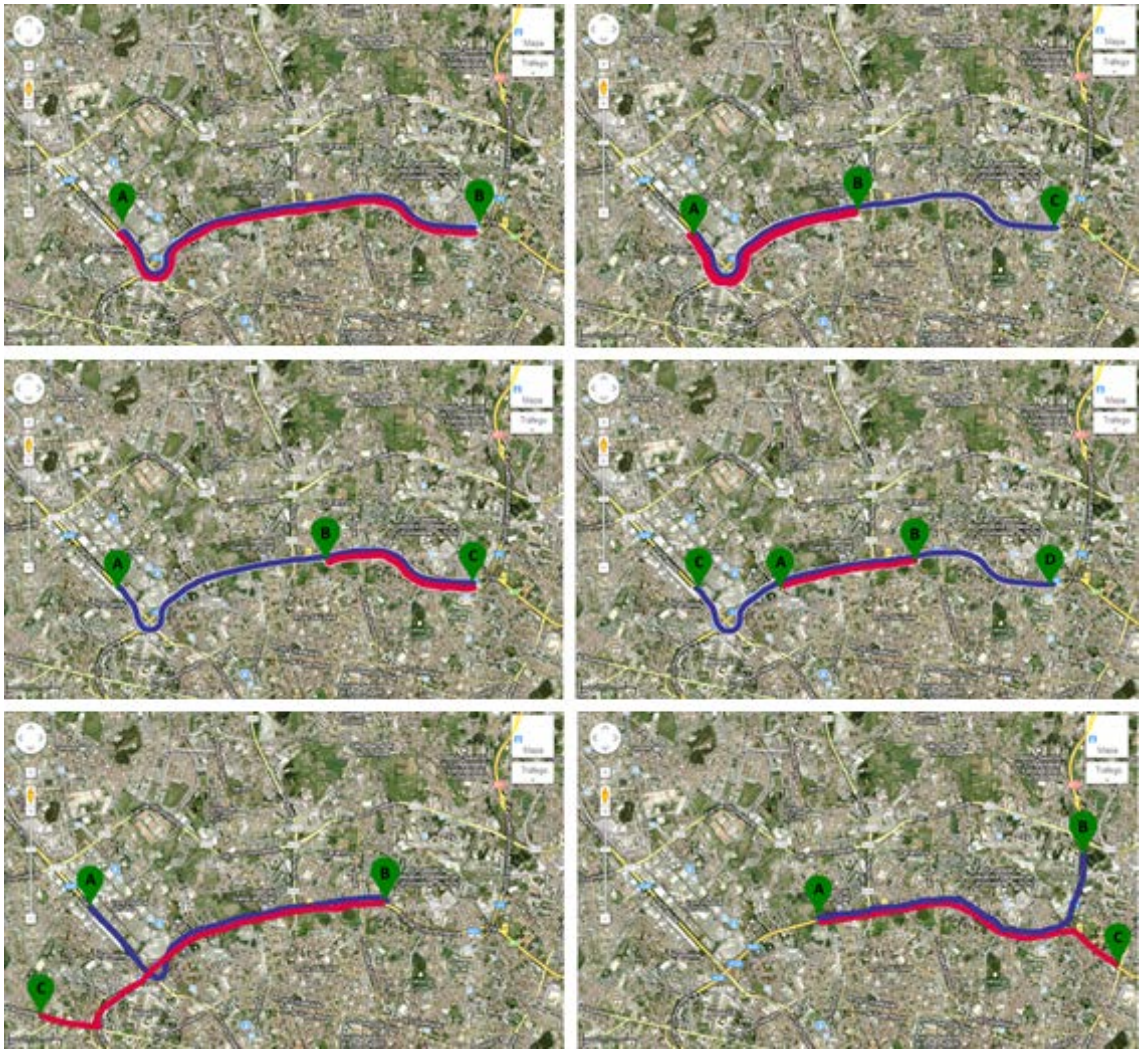


Figura 4.3: Situações em que os padrões de viagem são considerados vizinhos

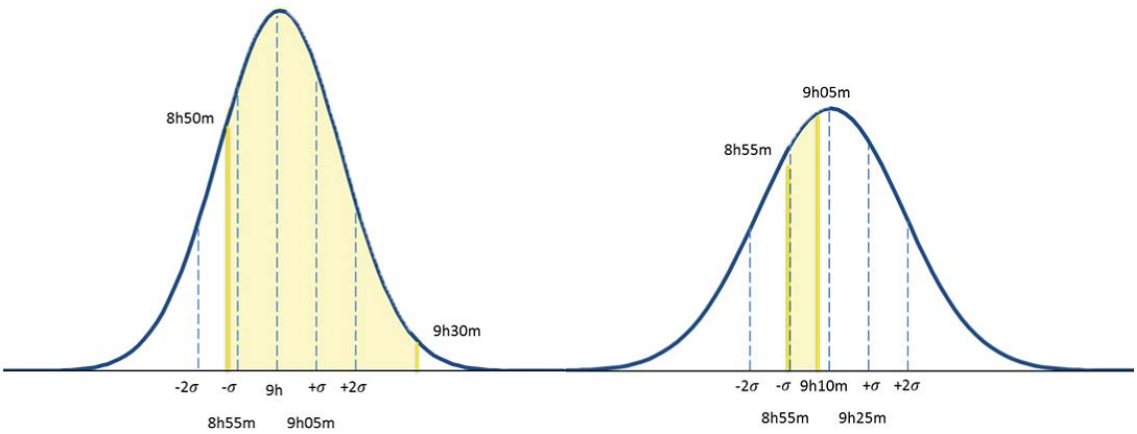
O facto de dois padrões de viagem serem vizinhos significa que as suas rotas serão comparadas para descobrir se faz sentido a recomendação de partilha de veículo entre os dois utilizadores ou não.

Vizinhos do tipo 1. Dois padrões de viagem são vizinhos do tipo 1 se são de utilizadores diferentes, se há pelo menos 1 dia da semana em que acontecem que é comum a ambos e se os locais significativos de partida e/ou de chegada dos 2 padrões de viagem estão a uma distância $\leq threshold$ (configurável). Para além destes requisitos é ainda necessário que as horas de partida e/ou chegada (dependendo do(s) local(is) significativo(s) que é(são) partilhado(s)) dos padrões de viagem sejam semelhantes.

Mais uma vez surge a necessidade de definir o que significa os padrões de viagem serem a horas semelhantes. Este conceito remete para a definição apresentada na Secção 4.1. No entanto, neste caso, o utilizador tem um papel mais ativo, isto é, o utilizador tem a possibilidade de configurar um parâmetro que indica a sua flexibilidade no que à hora de realizar a viagem diz respeito.

Por exemplo, se o utilizador indica que a sua flexibilidade é de 15 minutos, isso significa que ele está disponível para realizar a sua viagem entre -15 minutos e +15 minutos do que a hora habitual, ou seja, a hora média. Assim, para a determinação de padrões de viagem vizinhos, a hora de realização de um padrão de viagem é semelhante à de outro se dentro do intervalo de tempo definido pelo próprio utilizador a probabilidade do outro se realizar for $\geq 50\%$.

Daqui advém que a vizinhança entre padrões de viagem não é mútua. Ou seja, um padrão de viagem pode ser vizinho de outro mas esse outro pode não o ter como vizinho. Na Figura 4.4 pode observar-se um caso em que o padrão de viagem da Figura 4.4a não tem como vizinho o padrão de viagem da Figura 4.4b. No entanto, o padrão de viagem da Figura 4.4b tem o da Figura 4.4a como seu vizinho.



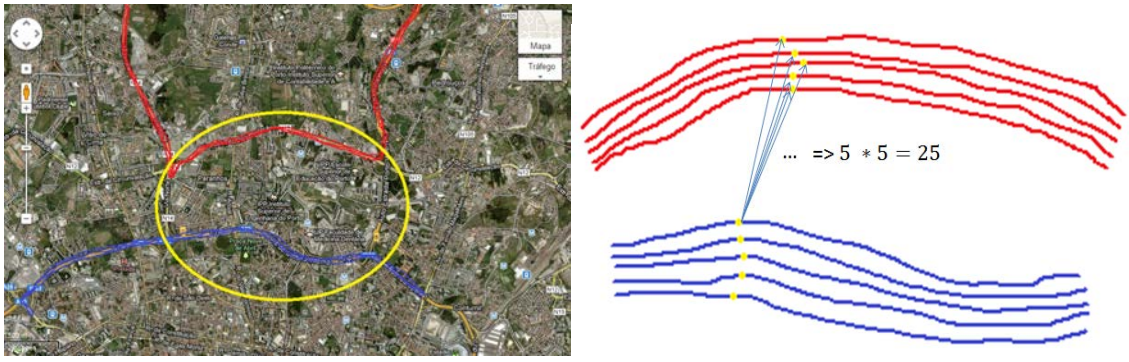
(a) Utilizador com $\mu = 9h$, $\sigma = 5min$ e $flexibilidade = 5min$ (b) Utilizador com $\mu = 9h10m$, $\sigma = 15min$ e $flexibilidade = 20min$

Figura 4.4: Exemplo de padrão de viagem vizinho de outro que não o tem como vizinho

Isto é natural na medida em que o padrão de viagem da Figura 4.4a, devido à sua curta flexibilidade, não aceita partilhar veículo com o utilizador do outro padrão de viagem sob as condições habituais do mesmo. No entanto, o padrão de viagem da Figura 4.4b indica que o utilizador está disposto a viajar a "qualquer" hora (flexibilidade de 20 minutos) e, portanto, à partida, está disposto a viajar sob as condições do padrão de viagem da Figura 4.4a.

Vizinhos do tipo 2. Para além dos padrões de viagem vizinhos do tipo 1, existem os padrões de viagem que partilham parte da rota mas que têm origens e destinos distantes e que são também vizinhos (vizinhos do tipo 2). Embora sejam chamados de "tipo 2", têm a mesma importância que os outros.

Então, dois padrões de viagem são vizinhos de tipo 2 se, à semelhança dos vizinhos de tipo 1, são de utilizadores diferentes e se há pelo menos um dia da semana em que acontecem que é comum a ambos. Para além disso, é necessário que, e esta é a principal funcionalidade que advém da segmentação das viagens (descrita na Secção 3.3), haja pelo menos tantos pares de segmentos das viagens daqueles padrões de viagem com centróides perto uns dos outros quanto a multiplicação do número de viagens que constituem os dois padrões de viagem. Por outras palavras, imagine-se que ambos os padrões de viagem são constituídos por 5 viagens. Para que se possa dizer que os utilizadores partilham parcialmente a rota tem de haver um número de pares de segmentos em que esses 2 segmentos estão perto um do outro e são de utilizadores diferentes igual a pelo menos $5 \times 5 = 25$ pois se as rotas estão perto em apenas 1 segmento de viagem há os tais 5×5 segmentos perto uns dos outros (há 1 segmento de cada viagem de um dos padrões de viagem que tem o seu centróide perto do centróide de um segmento de cada viagem do outro padrão de viagem).



(a) Rotas dos padrões de viagem

(b) *Zoom* no local circundado pela oval amarela

Figura 4.5: Exemplo com 2 padrões de viagem com 5 viagens cada: número de pares de segmentos com centróide perto $= 5 \times 5$

Se esse valor for ultrapassado e se a hora de passagem nos pares de segmentos com centróides próximos for semelhante então os padrões de viagem são considerados vizinhos do tipo 2 (a noção da hora ser semelhante remete para a explicação dada no parágrafo anterior).

Logicamente que se as rotas estão perto um da outra em mais do que 1 dos segmentos das viagens, o número de pares de segmentos próximos sobe abruptamente. Contudo, a partir do momento em que há pelo menos tantos pares de segmentos com centróides perto quanto a multiplicação do número de viagens dos 2 padrões de viagem, pode concluir-se que há pelo menos uma parte da rota (um segmento das viagens que originaram o padrão de viagem) em que elas se "tocam".

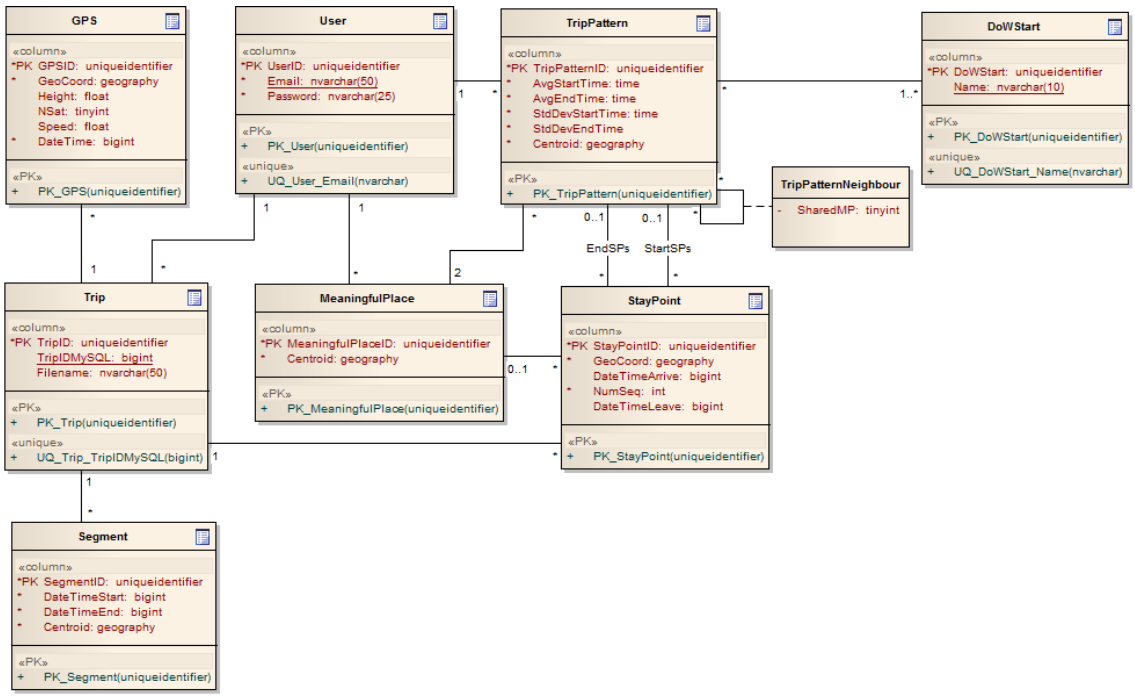


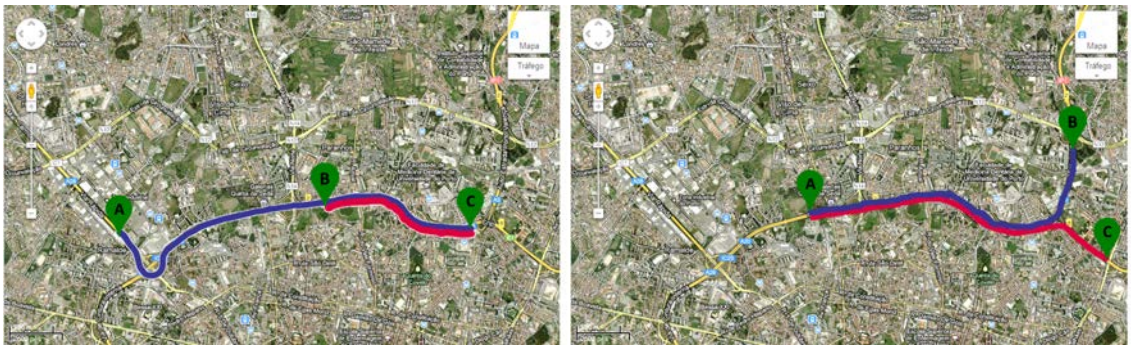
Figura 4.6: Esquema da base de dados (novo nível de conhecimento: padrões de viagem vizinhos)

De acordo com o enunciado anteriormente foi extraído mais um nível de conhecimento que permite acelerar o processo de comparação de rotas com vista à descoberta de sugestões de partilha de veículo. Esse novo nível de conhecimento é apresentado na Figura 4.6 e consiste na tabela com o nome *TripPatternNeighbour*.

4.3 Determinação de Sugestões de Partilha de Veículo

Depois de todos os passos descritos até ao momento, resta a tão aguardada determinação de sugestões de partilha de veículo.

De uma forma *high-level* pode dizer-se que há duas grandes formas que as rotas de padrões de viagem vizinhos podem tomar e que é necessário descortinar. Uma forma é quando uma das rotas é toda ela partilhada e a outra é quando existe uma bifurcação das rotas.



(a) Forma 1 (b) Forma 2

Figura 4.7: Exemplo das duas principais formas que as rotas de padrões de viagem vizinhos podem tomar

Todas as outras formas são extensões das apresentadas na Figura 4.7.

A solução idealizada para a forma presente na Figura 4.7a consiste na determinação se há coordenadas geográficas de um padrão de viagem perto de ambos os locais significativos do outro padrão de viagem ou não (Secção 4.3.1). Por outras palavras, sabendo que ambos os padrões de viagem terminam no mesmo local e que há um padrão de viagem que também tem coordenadas geográficas junto ao local significativo de origem do outro padrão de viagem então pode concluir-se que o primeiro padrão de viagem, na sua rota, também passa no local de início das viagens do segundo padrão de viagem.

Já a forma da Figura 4.7b mereceu o desenho e implementação de um novo algoritmo, algoritmo ShapeDetector, de forma a detetar formas em enormes quantidades de dados e em que o conjunto de dados é constituído por 2 tipos diferentes (Secção 4.3.2). Neste caso em concreto, a forma a detetar é a bifurcação das duas rotas.

4.3.1 Determinação da Proximidade da Rota aos Locais Significativos do Outro Padrão de Viagem

Com base nas informações contidas na base de dados é possível para cada padrão de viagem saber as viagens que o constituem e, por sua vez, é possível saber para cada viagem quais as suas coordenadas geográficas.

Tendo, portanto, as coordenadas geográficas de cada padrão de viagem e usando uma estrutura de dados adequada (ver detalhes na Secção 2.2), nomeadamente uma estrutura de dados espacial, é possível verificar para cada coordenada geográfica se é "num" dos locais significativos do outro padrão de viagem ou não, ou seja, se está a uma distância de um dos locais significativos do outro padrão de viagem que seja considerada pelo próprio utilizador como perto (recorrendo a um *threshold* configurável).

De forma a diminuir a complexidade temporal desta análise a granularidade dos dados usada é menor que a da recolha de dados, passando de uma coordenada geográfica a cada segundo

(granularidade usada na recolha de dados) para a existência de 20 segundos, no mínimo, entre cada coordenada geográfica.

Sabendo que há coordenadas geográficas de um dos padrões de viagem que se aproximam dos dois locais significativos do outro padrão de viagem pode concluir-se que a rota seguida naquele padrão de viagem sobrepõem-se, na totalidade, à do outro. Desta conclusão, sai uma sugestão para os dois utilizadores dos dois padrões de viagem no sentido de partilharem veículo aquando da realização das viagens daqueles padrões de viagem.

Algoritmo 5 Algoritmo DetermineProximityToMPs($TPIDUser1, TPIDUser2, Radius$)

```
1: //coordenadas geográficas do padrão de viagem do utilizador 1
2: GeoCoords := GetGeoCoordsFromDB(TPIDUser1);
3: //locais significativos do padrão de viagem do utilizador 2
4: MPs := GetMeaningfulPlacesFromDB(TPIDUser2);
5: //para cada local significativo identifica as coordenadas que estão "à sua volta" (dentro de um
   determinado raio)
6: GeoCoordsNearMPStart := GetCoordsFromDB(GeoCoords, MPs[Start], Radius);
7: GeoCoordsNearMPEnd := GetCoordsFromDB(GeoCoords, MPs[End], Radius);
8: if GeoCoordsNearMPStart! = NULL && GeoCoordsNearMPEnd! = NULL then
9:     RETURN TRUE;
10: else
11:     RETURN FALSE;
12: end if
```

Os parâmetros do Algoritmo 5 são os IDs dos padrões de viagem dos utilizadores (padrões de viagem vizinhos) e o raio (variável *Radius*) que consiste num valor (configurável) que indica a disponibilidade do utilizador para se desviar da sua rota habitual. Por outras palavras, se o utilizador está disponível para se deslocar 1km da sua rota (para permitir a partilha de veículo) e há um outro utilizador que inicia a sua viagem num local a 750 metros do da rota seguida habitualmente pelo utilizador então considera-se que o local de início de viagem do outro utilizador é num dos locais por onde a rota passa (pois 750m < 1km).

Naturalmente o algoritmo não foi implementado seguindo de forma linear o pseudocódigo apresentado no Algoritmo 5. Na verdade, a implementação consiste apenas numa *query* à base de dados. No entanto, a própria *query* acaba por ser constituída pelos passos enunciados no pseudo-código apresentado.

A complexidade temporal deste algoritmo é $O(2\ log\ n)$ pois a função $GetCoordsFromDB()$ consiste numa *query* a uma R^*-tree que tem complexidade temporal média $O(log\ n)$.

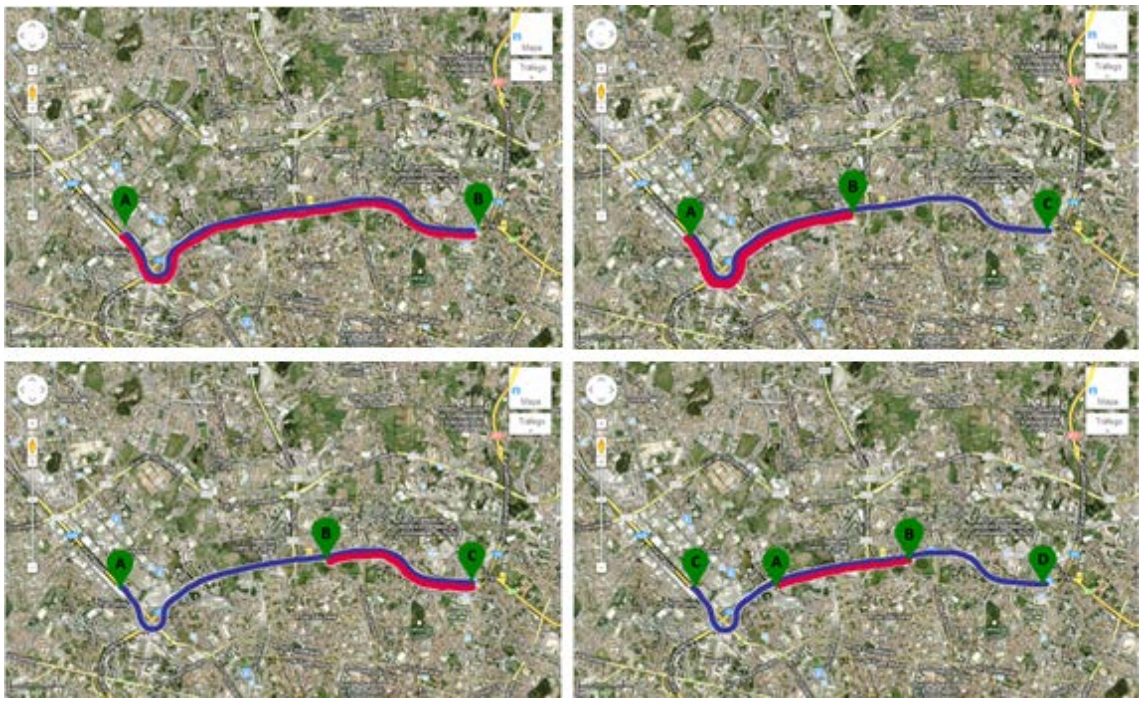


Figura 4.8: Situações detetadas com a determinação da proximidade da rota aos locais significativos do outro padrão de viagem

Todas as situações em que um dos padrões de viagem é todo ele partilhado com outro (Figura 4.8) são detetadas com a solução apresentada.

4.3.2 Algoritmo ShapeDetector

Nos casos em que há uma parte da rota que é partilhada e outra não, ou seja, quando há uma bifurcação das rotas, a solução anterior não resolve o problema.

Figura 4.9: Situações detetadas com o algoritmo ShapeDetector

As situações ilustradas na Figura 4.9 serviram de mote para o desenho e implementação do algoritmo ShapeDetector.

Como só há uma parte da rota que é partilhada, o problema está em encontrar o local a partir do qual isso acontece e/ou deixa de acontecer. Com vista a esse fim, foi desenhado um algoritmo que se baseia, em parte, na ideia dos algoritmos de *clustering* baseados em densidade e, em parte, no funcionamento de um neurónio de uma rede neuronal artificial.

Este é um algoritmo para ser aplicado a dados espaciais. O seu intuito é a deteção de determinadas formas nos dados e foi desenvolvido com base no pressuposto de que o conjunto de dados sobre o qual vai atuar é constituído por dois tipos de dados.

A parte dos algoritmos de *clustering* baseados em densidade está relacionada com a pesquisa que é feita nos dados, isto é, existe a definição de *core point* (embora com uma diferença) e de vizinhança à semelhança do algoritmo DBSCAN, apresentado na Secção 2.4.2.1. A diferença na definição de *core point* está na substituição da aplicação do parâmetro *MinPts*, que obriga a que haja pelo menos *MinPts* na vizinhança de um ponto para este ser considerado *core point*, pela obrigação de que na vizinhança de um ponto, para ser considerado *core point* tenha de existir uma determinada distribuição espacial de pontos. Por exemplo, tem de existir pelo menos 30% (valor configurável) de pontos de cada tipo. A parte dos neurónios das redes neuronais artificiais está relacionada com o passo seguinte do algoritmo. No caso do ponto ser um *core point*, a distância média entre os pontos de cada tipo é calculada e aplicada a uma função de ativação. Esta função de ativação serve, à semelhança dos neurónios das redes neuronais artificiais, para limitar a amplitude do valor calculado.

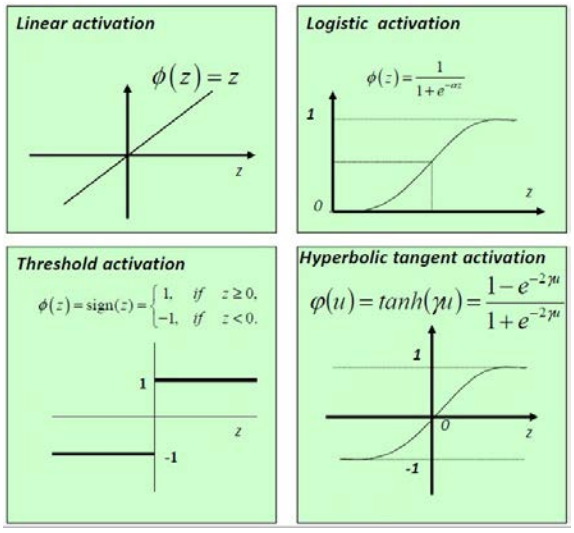


Figura 4.10: Exemplos de funções de ativação (http://paginas.fe.up.pt/~ec/)

A função de ativação a usar é dependente da forma que se quer encontrar no conjunto de dados. Para o caso concreto em que o algoritmo é aplicado, na deteção dos locais onde as rotas de dois utilizadores se juntam ou afastam, é usada uma distribuição espacial que exige que haja pelo menos 25% de ambos os tipos de dados num raio $\leq Eps$ (valores configuráveis) e como função de ativação é a apresentada na Figura 4.11.

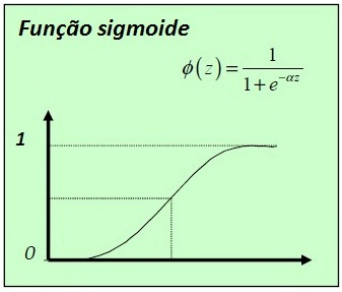


Figura 4.11: Função de ativação usada no algoritmo ShapeDetector

O parâmetro da função de ativação, representado pela letra α na Figura 4.11, influencia o declive da curva. Quanto maior o valor de α, menos acentuado é o declive da curva, ou seja, a função toma valores perto de 1 com valores no eixo do x maiores. O valor escolhido para o parâmetro α depende do valor do parâmetro Eps do algoritmo ShapeDetector. Assim, a função de ativação toma valores perto de 1 (valor máximo) apenas perto da distância correspondente ao valor do parâmetro Eps (máxima distância média possível entre 1 ponto e os seus pontos vizinhos do outro tipo). Por exemplo, quando o valor do parâmetro Eps é 600m, o valor do parâmetro α é -0,02 pois assim quando $x = 540m$ o valor da função de ativação é $\simeq 0,99$. No entanto, quando o valor do parâmetro Eps é 1000m, não faz sentido que desde os 540m até aos 1000m a função de ativação varie apenas 0,01 e então o valor do parâmetro α é 0,0125.

A função de ativação escolhida privilegia as zonas em que existe "grande" separação dos pontos dos dois utilizadores pois quanto maior for a distância média entre os pontos de cada utilizador maior será o valor retornado pela função de ativação. Contudo, é necessário que eles estejam relativamente perto uns dos outros, ou seja, que estejam a uma distância $\leq Eps$, que é um dos parâmetros do algoritmo.

No entanto, o algoritmo pode ser usado para detetar outras formas. Por exemplo, se quisermos detetar *clusters* onde exista uma determinada distribuição de dados de cada utilizador, pode recorrer-se à função de ativação da Figura 4.12. Neste caso a função de ativação valoriza a proximidade dos dados, ou seja, quanto menor a distância entre os pares dos diferentes tipos de dados, maior o valor retornado pela função de ativação.

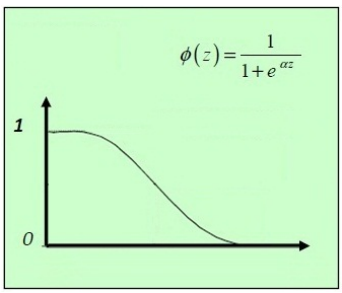


Figura 4.12: Possível função de ativação para determinar *clusters* com o algoritmo ShapeDetector

Após a limitação da amplitude do valor da distância entre os pontos de cada tipo para valores entre 0 e 1, são excluídos os pontos com um valor inferior a um certo *threshold* (configurável). Por outras palavras, com este *threshold* define-se o quão afastados os pontos têm de estar (sabendo que nunca estão mais afastados do que Eps metros) para se considerar que há uma bifurcação.

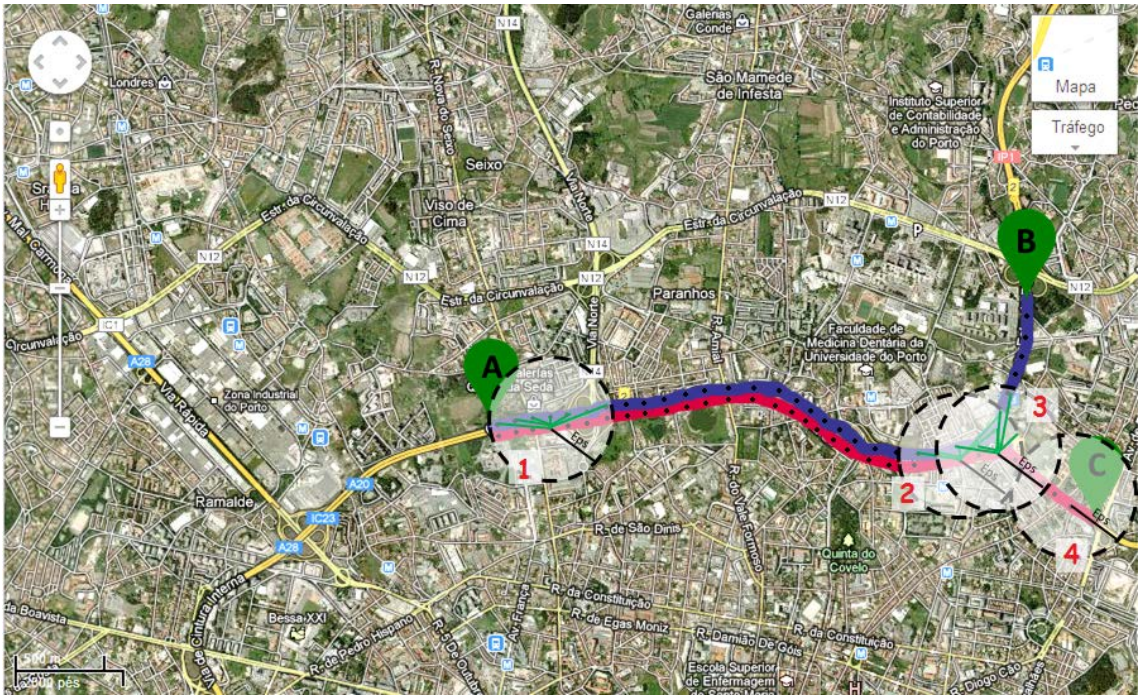


Figura 4.13: Funcionamento do algoritmo ShapeDetector

Na Figura 4.13, que simula o funcionamento do algoritmo ShapeDetector, a situação identificada com o número 1 teria um valor tão perto de 0 que seria certamente inferior ao *threshold* que indica o quão afastados devem estar os pontos para se considerar que estão numa bifurcação. As situações 2 e 3 seriam certamente "escolhidas" como tendo os pontos suficientemente afastados e, por fim, a situação 4 seria automaticamente ignorada por os pontos em análise não respeitarem a distribuição espacial (na vizinhança do ponto em análise apenas existem pontos de um tipo, os pontos vermelhos).

Tendencialmente os pontos com um valor superior ao *threshold* estão perto uns dos outros (como os pontos das situações 2 e 3 da Figura 4.13). Contudo, pode haver mais do que um ponto de bifurcação nas rotas e, naturalmente, haverá pontos que estarão distantes de outros na medida em que são de pontos de bifurcação diferentes. De forma a contemplar esses casos, os pontos são agrupados no final com o objetivo de determinar quantos pontos de bifurcação existem e quais os pontos com que foram detetados (que permitem também identificar o(s) local(is) onde existe(m) a(s) bifurcação(ões)). O agrupamento é feito de uma forma muito simples: todos os pontos com distância $\leq 2 \times Eps$ pertencem ao mesmo grupo. O valor $2 \times Eps$ surge na medida em que essa é a distância máxima a que dois pontos podem estar tendo sido originados pelo mesmo local de bifurcação.

Algorithm 6 ShapeDetector($SetOfPoints, Eps, SpatialDistribution, ActivationFunction$)

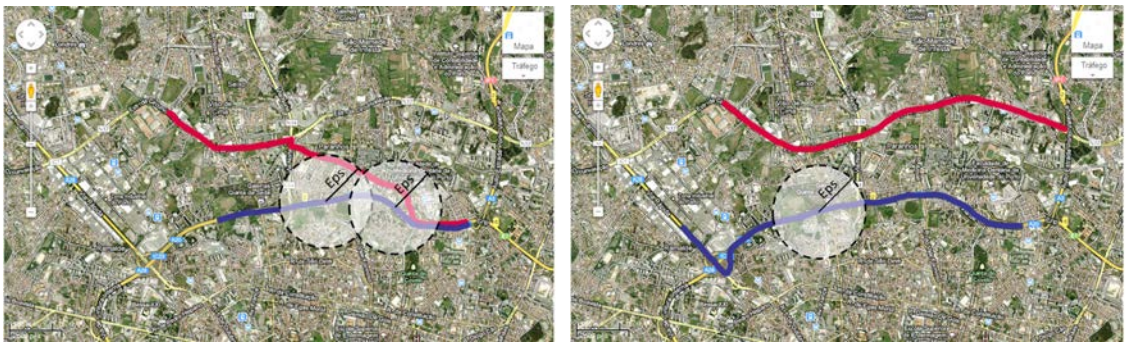
```
1:  //points of the user with least quantity of points
2:  PointsToAnalyse := Points.getPointsMinUser();
3:  for i FROM 1 TO PointsToAnalyse.size do
4:      Point := PointsToAnalyse[i];
5:      //if Point not explored
6:      if Point = UNEXPLORED then
7:          //points directly-density reachable from Point
8:          Seeds := Points.regionQuery(Point, Eps, SpatialDistribution);
9:          //if Point is core point
10:         if seeds ≠ NULL then
11:             //calc average dist and apply it to ActivFunc
12:             value := ComputePoint(Point, Seeds, ActivationFunction);
13:             if value ≥ T then
14:                 Point := value;
15:             else
16:                 Point := 0;
17:             end if
18:         end if
19:         Point := EXPLORED;
20:     end if
21: end for
```

A variável *Points* representa o conjunto de pontos dos 2 utilizadores. O parâmetro Eps é o mesmo do algoritmo DBSCAN, apresentado na Secção 2.4.2.1.

O algoritmo ShapeDetector faz uso de uma função *regionQuery*() que retorna, à semelhança da função com o mesmo nome no algoritmo DBSCAN, os pontos que estão na vizinhança de *Point*, ou seja, a uma distância menor ou igual a Eps. Esta ação pode ser suportada de forma eficiente por métodos de acesso espaciais como R^*-trees, como apresentado na Secção 2.2. No pior caso, a altura de uma R^*-tree é $O(log\ n)$ para uma base de dados de n pontos. Neste caso, o n é o número de pontos contidos na variável *Points*. Como a vizinhança de um ponto *Point* será mais pequena que todo o conjunto de dados existente, a complexidade temporal média de uma *query* a uma região é $O(log\ n)$. Como é suficiente avaliar apenas os pontos de um dos utilizadores, o que tiver menos pontos para analisar, ou seja, o que tiver menos coordenadas geográficas recolhidas nas viagens que originaram o padrão de viagem em análise, será pesquisada a *regionQuery*() apenas dos pontos contidos em *PointsToAnalyse* e, portanto, a complexidade temporal do algoritmo é $O(n' * log\ n)$, em que n' significa o número de pontos do utilizador com menor pontos, ou seja, *PointsToAnalyse.size*. Assim, $n' \leq \frac{n}{2}$.

O algoritmo ShapeDetector tem uma escalabilidade aceitável principalmente quando comparado com os algoritmo com aplicação em enormes quantidades de dados. O grande inconveniente,

um pouco à semelhança dos restantes algoritmos, são os parâmetros que fazem com que tenha de existir um conhecimento dos dados a priori para se poder escolher valores interessantes.



(a) Forma 1 (b) Forma 2

Figura 4.14: Exemplos de formas possíveis nos dados

A escolha dos parâmetros tem influência nos locais onde são detetadas as formas desejadas, como é possível ver na Figura 4.14. Imaginando que a distribuição espacial exigida é que haja pelo menos 45% de pontos de cada tipo, na Figura 4.14a apenas no círculo da direita seria detetado o encontro das duas rotas pois no primeiro círculo a quantidade de pontos vermelhos não respeita certamente o valor mínimo de 45%. Já na Figura 4.14b, nunca é detetado o encontro das rotas na medida em que, de acordo com o parâmetro Eps, as rotas não estão suficientemente perto uma da outra.

4.4 Sugestões de Partilha Não Suportadas

Embora a maior parte das sugestões possíveis sejam detetadas com as soluções apresentadas, existem ainda 3 situações (que são extensões das anteriores) que não o são.

(a) Situação 1 (b) Situação 2 (c) Situação 3

Figura 4.15: Situações não detetáveis com a solução atual

Na verdade, a situação ilustrada na Figura 4.15a é detetada com a solução implementada. No entanto, o que acontece é que no caso dos utilizadores seguirem em sentido oposto a sugestão também é feita. Ou seja, como não há nenhum local significativo partilhado pelas duas rotas, com a informação armazenada na base de dados, não é possível saber a direção que está a ser seguida

por ambos os utilizadores. É detetado que a rota é partilhada e no caso da hora da realização das viagens ser semelhante é feita a sugestão.

A solução para o problema fica acessível a partir do momento em que cada segmento de viagem passe a ter associada a orientação que o utilizador está a seguir naquele momento (Norte, Oeste, etc). Assim, só utilizadores com segmentos de viagens com centróides próximos e com a mesma orientação passariam a ser considerados vizinhos.

Já as situações das Figuras 4.15b e 4.15c acabam por ser a junção das duas soluções propostas (Secções 4.3.1 e 4.3.2). No entanto, estas sugestões acabaram por não ser implementadas, como já era previsto e devidamente enunciado aquando da "Preparação da Dissertação", devido a falta de tempo.

4.5 Sumário

De forma a tornar a procura de sugestões de partilha de veículo viável do ponto de vista temporal começou-se pela determinação de padrões de viagem vizinhos. Este conceito consiste, basicamente, na procura de pares de padrões de viagem que tenham parte (ou total) da rota partilhada pois apenas nesses casos é possível a sugestão de partilha de veículo.

Tendo então por base os padrões de viagem vizinhos, seguiu-se a determinação de sugestões de partilha de veículo propriamente dita. Foram identificadas as duas principais formas que as rotas de padrões de viagem vizinhos podem tomar (Figura 4.7) e elaboradas soluções para ambos os casos.

No caso em que uma das rotas é totalmente sobreposta pela outra, a solução passou pelo estudo de cada uma das rotas para se perceber se alguma delas se aproximava de ambos os locais significativos da outra. Se assim for, conclui-se que essa rota tem início antes ou no mesmo local que a outra e termina depois ou no mesmo local. Daqui resulta uma sugestão de partilha de veículo.

Já no caso em que existe uma bifurcação nas rotas, a solução passou pelo desenho e implementação de um novo algoritmo, o algoritmo ShapeDetector. Este algoritmo é baseado na ideia dos algoritmos de *clustering* baseados em densidade e no funcionamento de um neurónio de uma rede neuronal artificial. A combinação destas duas ideias resultou num algoritmo que tem como intuito a deteção de formas em enormes conjuntos de dados espaciais e parte do pressuposto de que o conjunto de dados sobre o qual vai atuar é constituído por dois tipos de dados distintos.

O funcionamento do algoritmo consiste na procura de *core points* à semelhança do algoritmo DBSCAN. No entanto, à definição já conhecida de *core point* foi adicionada uma restrição que está relacionada com a obrigação de haver uma determinada distribuição espacial dos dados. Para todos os *core points* é calculada a distância média aos pontos do outro tipo, neste caso, aos pontos do outro utilizador. Depois de calculado esse valor, ele é aplicado a uma função de ativação. Esta função de ativação serve, exatamente como nos neurónios das redes neuronais artificiais, para limitar a amplitude do valor calculado. Por fim, com os pontos com valor da função de ativação $\geq T$ (valor configurável) é recolhida a informação sobre quantos pontos de bifurcação existem nos

dados e onde. No caso de ser detetada uma bifurcação, ou mais, é feita a sugestão de partilha de
2 veículo.

Capítulo 5

Implementação

Esta dissertação foca-se essencialmente no desenvolvimento e aplicação de técnicas e algoritmos de forma a possibilitar a sugestão de partilha de veículo aos utilizadores. Embora aquando da apresentação da solução já tenham sido detalhados os algoritmos e a forma como foram usados, nesta secção são dados a conhecer outros detalhes técnicos.

Note-se que a plataforma de recolha de dados não é detalhada na medida em que já se encontrava desenvolvida quando foi iniciada a dissertação. A recolha de dados é feita para uma base de dados relacional MySQL (ver Secção 2.5.1.1 para mais detalhes) e não faz uso de nenhuma das funcionalidades atualmente existentes nos SGBDs de forma a obter uma melhoria de desempenho tendo em conta que o tipo de dados guardados são geográficos. Devido a esse entrave, os dados foram migrados para uma base de dados SQLServer onde é feito uso de tipos de dados adequados para o efeito (tipo de dados *SQLGeography*, por exemplo).

5.1 Servidor

O servidor é responsável pela migração dos dados da base de dados MySQL para a base de dados SQLServer. Para além disso, toda a computação inerente ao cálculo dos pontos de estadia, segmentos das viagens, locais significativos e padrões de viagem de cada utilizador e padrões de viagem vizinhos é também desempenhada pelo servidor. Para o seu desenvolvimento recorreu-se à tecnologia C#.

Na Figura 5.1 são visíveis três "áreas" de atuação por parte do servidor. Foi feita esta separação porque as tarefas desempenhadas pela caixa identificada com o número 1 seriam, idealmente, feitas aquando da recolha de dados com baixo custo computacional para o dispositivo móvel e minimizaria o do servidor como já foi referido nas Secções 3.1 e 3.3), as tarefas desempenhadas na caixa identificada com o número 2 são desempenhadas automaticamente com um intervalo de tempo definido (de 15 em 15 dias, por exemplo) e, por fim, as tarefas na caixa identificada com o número 3 são realizadas apenas e só se o utilizador o requisitar.

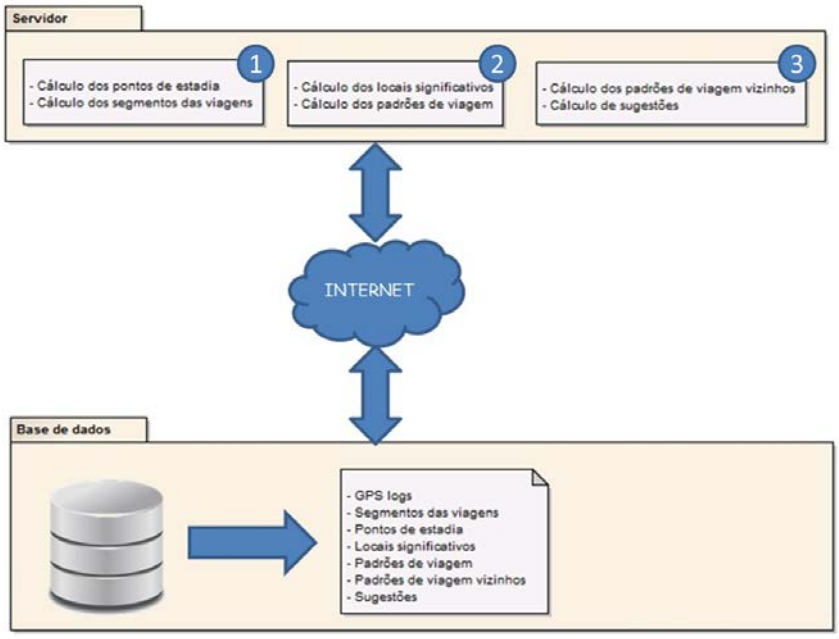


Figura 5.1: Ilustração simplificada do servidor

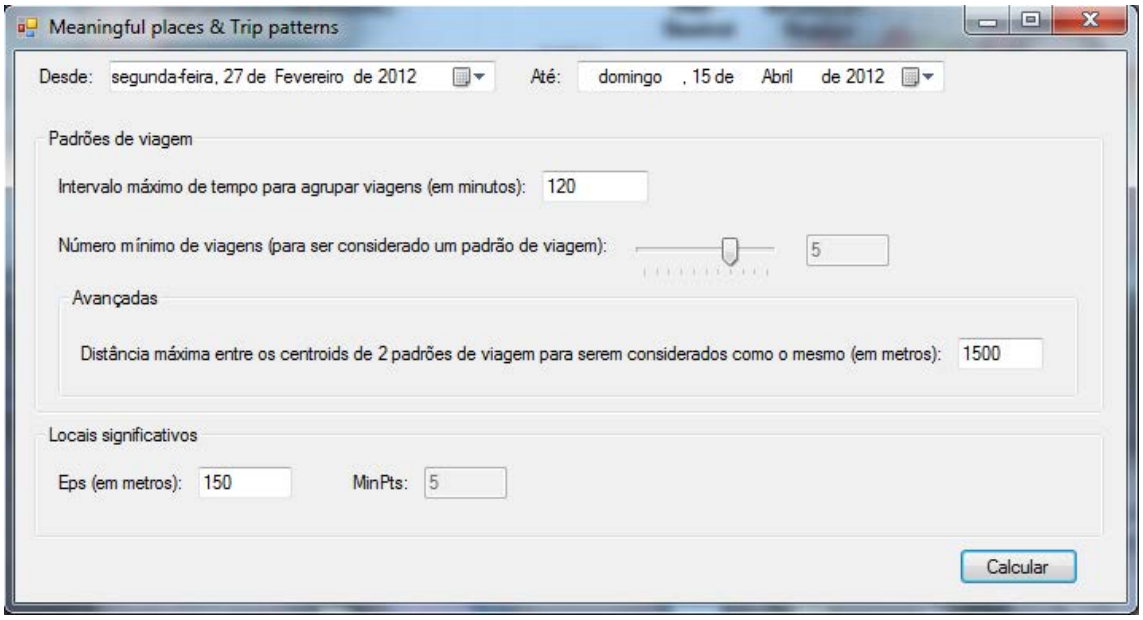


Figura 5.2: Interface gráfica da aplicação servidor

De forma a tornar mais fácil a alteração dos parâmetros dos algoritmos que executam no servidor foi desenvolvida uma interface gráfica (Figura 5.2).

Base de Dados A base de dados é o que suporta a computação do servidor pois é lá que estão todos os dados que ele necessita.

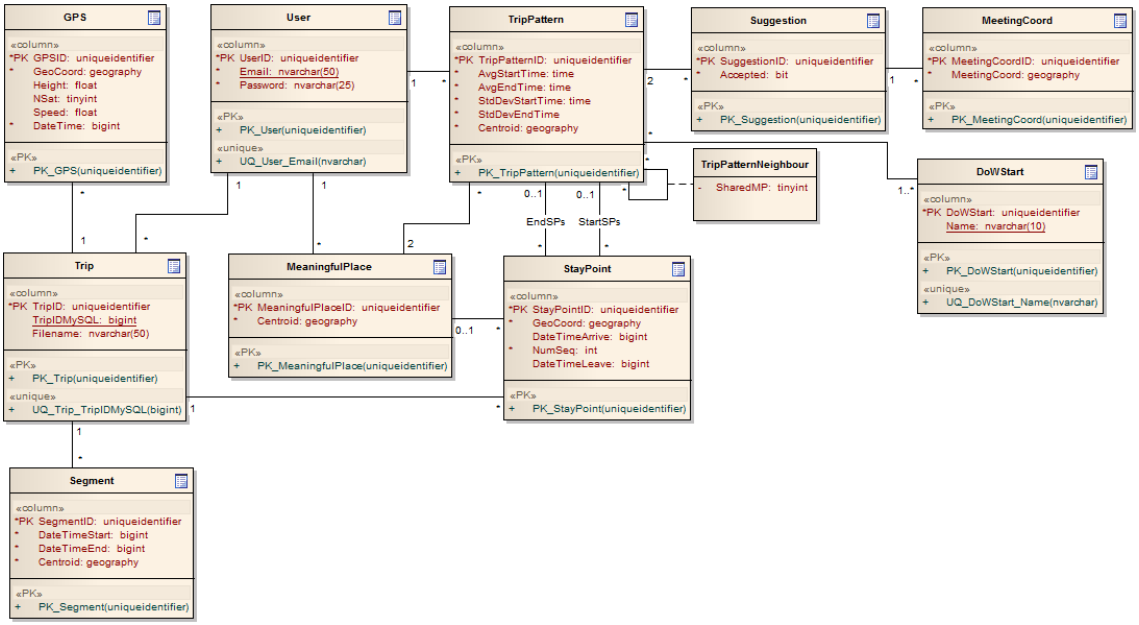


Figura 5.3: Esquema completo da base de dados

Devido à grande quantidade de dados inerente ao próprio problema, a base de dados foi sendo apetrechada ao longo do tempo com uma série de índices de forma a possibilitar um desempenho em termos de acesso aos dados aceitável (Tabela 5.1). Sem eles, seria, obviamente, impossível obter sugestões de partilha de veículo em tempo útil.

Tabela(s), campo(s)	***Clustered***	***Nonclustered***	**Composto**	***Spatial***	***Unique***
Todas, chave(s) primária(s)	X	-	-	-	-
Todas, chave(s) estrangeira(s)	-	X	-	-	-
GPS, DateTime	-	X	-	-	-
GPS, GeoCoord	-	-	-	X	-
MeaningfulPlace, Centroid	-	-	-	X	-
Segment, DateTimeStart e DateTimeEnd	-	X	X	-	-
Segment, Centroid	-	-	-	X	-
StayPoint, DateTimeArrive e DateTimeLeave	-	X	X	-	-
StayPoint, GeoCoord	-	-	-	X	-
TripPattern, Centroid	-	-	-	X	-
User, Email	-	-	-	-	X

Tabela 5.1: Índices criados na base de dados

A base de dados usada foi a Microsoft SQLServer 2008, versão R2. Esta escolha recaiu essencialmente pelo facto da mesma possibilitar o uso de índices espaciais para a indexação dos dados e pela familiarização com o SGBD em questão.

5.2 Cliente

A aplicação cliente serve essencialmente para o utilizador configurar o sistema de forma a adaptar aos seus interesses as sugestões que lhe são feitas. Serve também para a visualização dos padrões de viagem do próprio utilizador e das sugestões de partilha de veículo que o sistema faz.

A maior parte dos parâmetros descritos como configuráveis são-no na aplicação cliente (Figura 5.4).

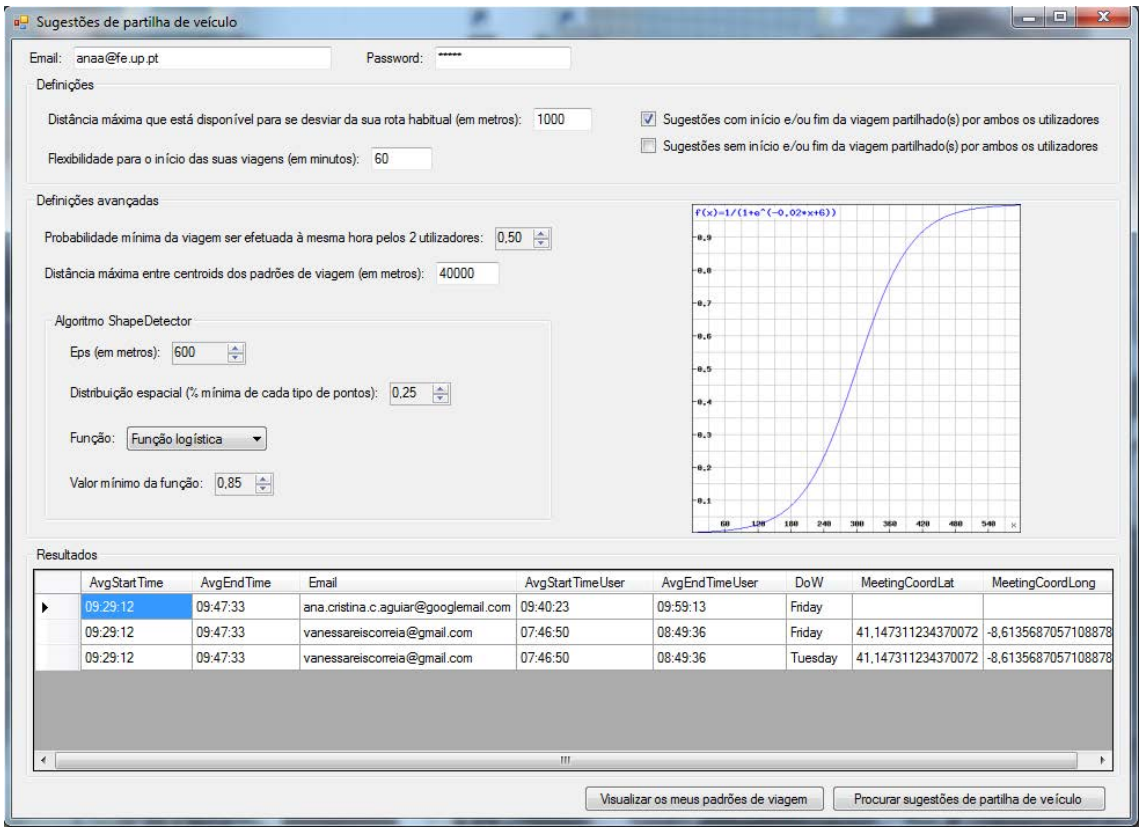


Figura 5.4: Interface gráfica da aplicação cliente

Na Figura 5.4 é visível toda a panóplia de opções que o utilizador tem para configurar o sistema. Todos os campos têm valores *default* que são valores viáveis para a esmagadora maioria dos utilizadores. No entanto, é expectável que principalmente os valores da *GroupBox* intitulada "Definições" sejam alterados por praticamente todos os utilizadores na medida em que são parâmetros que dizem respeito a cada um. Para o seu desenvolvimento recorreu-se também à tecnologia C#.

5.3 Configurações do Sistema

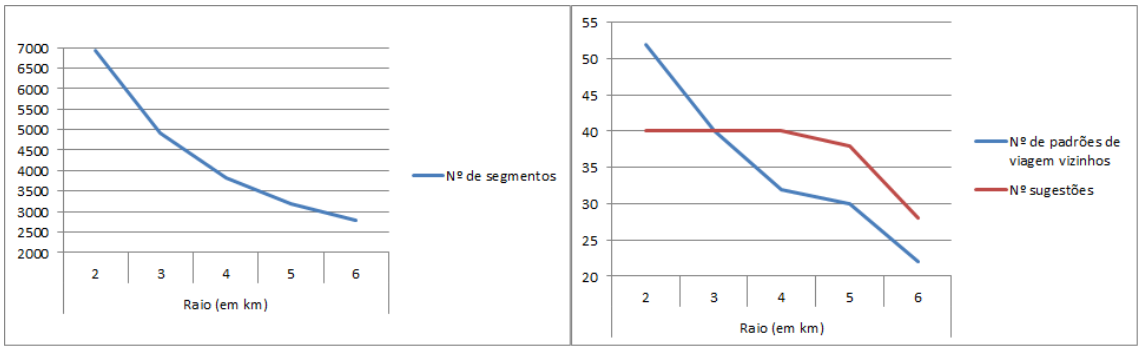
Tendo em conta que a deteção dos pontos de estadia e dos locais significativos são feitas com base nos mesmos valores para todos os utilizadores, foram feitas análises de sensibilidade de forma a perceber quais os valores que respondiam melhor ao *output* desejado.

Todos os valores considerados *default* para além de serem fundamentados com base nas análises de sensibilidade feitas, foram validados junto dos utilizadores que participaram na amostra que esteve encarregue de assegurar a recolha de dados inicial (consultar a Secção 6.1 para mais detalhes sobre a amostra).

Na análise de sensibilidade feita com vista à determinação da melhor configuração para a "Segmentação das viagens" foram usados os dados recolhidos entre 27 de Fevereiro de 2012 e 15 de Abril de 2012, mais concretamente 1.215 viagens e 1.113.294 coordenadas geográficas. Já para as restantes análises foram usados os dados recolhidos entre 06 de Fevereiro de 2012 e 01 de Abril de 2012, que correspondem a 876 viagens e 1.018.184 coordenadas geográficas (mais detalhes sobre os dados recolhidos na Secção 6.1).

5.3.1 Segmentação das viagens

Uma questão pertinente prende-se com a dimensão dos segmentos das viagens. Quanto menor a dimensão dos segmentos, maior granularidade têm os dados com que se trabalha para chegar às sugestões. Logo, mais e/ou melhores sugestões são feitas. No entanto, quanto maior a granularidade, maior a complexidade temporal. Uma dimensão demasiado pequena dos segmentos pode fazer com que a aplicação seja inviável do ponto de vista temporal.



(a) Evolução do número de segmentos variando a dimensão dos mesmos

(b) Evolução do número de sugestões e do número de padrões de viagem vizinhos variando a dimensão dos segmentos

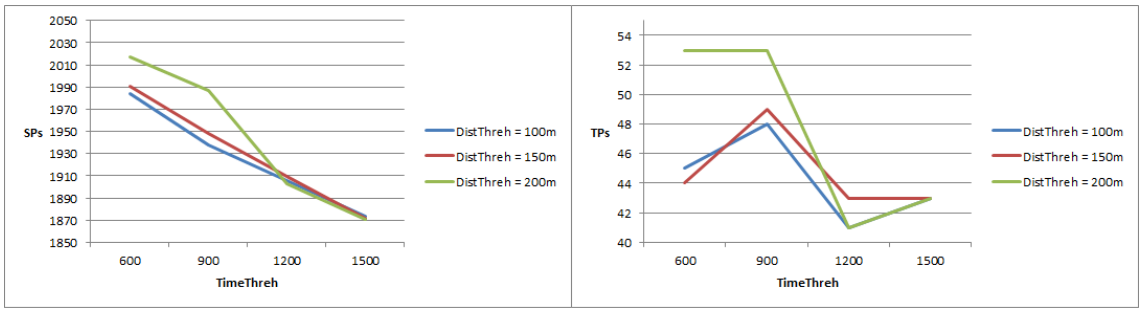
Figura 5.5: *Trade-off* a ter em conta com a dimensão dos segmentos das viagens

Na Figura 5.5b é percetível que o número de sugestões feitas pelo sistema é influenciado apenas a partir da dimensão 5km (inclusive) dos segmentos. Com uma dimensão de 4km, o número de segmentos criados (3832) é quase metade do que os criados quando a dimensão toma o valor de 2km (6929) (Figura 5.5a). Esta enorme redução causa um impacto muito grande no desempenho da aplicação quando está à procura dos padrões de viagem vizinhos pois o espaço de soluções é muito menor.

Com base nos dados apresentados, o valor escolhido para a dimensão dos segmentos de viagem foi 4km.

5.3.2 Pontos de Estadia

A deteção de pontos de estadia baseia-se em dois parâmetros: *DistThreh* e *TimeThreh*. A questão que se coloca é qual a melhor combinação dos dois.



(a) Evolução do número de pontos de estadia detetados variando os parâmetros *DistThreh* e *TimeThreh*

(b) Evolução do número de padrões de viagem detetados variando os parâmetros *DistThreh* e *TimeThreh*

Figura 5.6: Evolução do número de pontos de estadia e de padrões de viagem detetados variando os parâmetros *DistThreh* e *TimeThreh*

Fazer uma análise puramente gráfica nestes casos pode induzir-nos em erro. É muito mais importante a qualidade da solução apresentada do que número propriamente dito. Analisando o gráfico da Figura 5.6b não se percebe um padrão no gráfico que permita concluir algo. Um facto curioso é que o número de padrões de viagem detetados não é menor com valores de *TimeThreh* menores. O que acontece é que alguns padrões de viagem ficam divididos em vários.

O parâmetro *DistThreh* não se revelou significativo para a determinação de locais significativos. No entanto, é necessário ter em atenção que a *accuracy* dos recetores GPS dos dispositivos móveis em locais fechados é pior que 100 metros. Assim, o valor escolhido para o parâmetro *DistThreh* foi 150 metros. Pela mesma razão foi o valor escolhido para o parâmetro Eps do algoritmo DBSCAN na deteção dos locais significativos.

Já para o parâmetro *TimeThreh* foi escolhido o valor 900 segundos pois foi o que originou resultados que melhor satisfizeram as expectativas dos utilizadores.

Estas escolhas beneficiam também do conhecimento geral. À partida já era expectável que o parâmetro *TimeThreh* tomasse um valor entre os $\simeq$ 600 segundos e os $\simeq$ 1500 segundos. Um valor superior a 1500 segundos, por exemplo, começa a não fazer muito sentido na medida em que estamos a admitir que um utilizador pare mais de 25min a meio de um dos seus padrões de viagem. Muito provavelmente ninguém aceitará viajar com uma pessoa que esteja parada num local mais de 25min.

5.3.3 *Time Slots*

Quando há várias viagens do mesmo utilizador, no mesmo dia da semana e com a mesma origem e o mesmo destino é necessário agrupá-las. Idealmente são agrupadas por proximidade de horas, isto é, se há uma viagem às 10h e outra às 10h05m então pertencem ao mesmo *time slot*

(ver Secção 3.3 para mais detalhes). Recorreu-se então a uma análise de sensibilidade de forma a perceber qual o melhor limite máximo entre viagens do mesmo *time slot*.

Logicamente, quanto maior a dimensão dos *time slots*, mais padrões de viagem são detetados. Isto acontece porque se os *time slots* tiverem uma duração demasiado pequena não haverá viagens suficientes dentro do mesmo *time slot* para ser considerado um padrão de viagem.

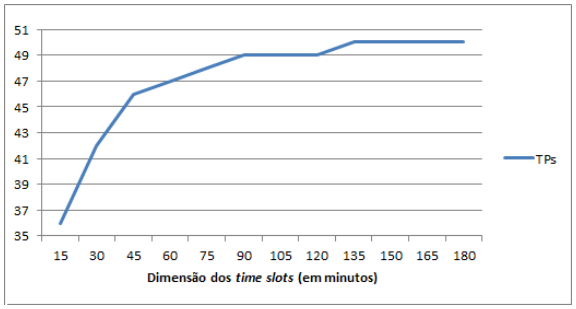


Figura 5.7: Evolução do número de padrões de viagem detetados variando a dimensão dos *time slots*

De acordo com a Figura 5.7 percebe-se que a subida do número de padrões de viagem é bastante acentuada até aos 45min. A partir daí abranda um pouco (continuando acentuada) até aos 90min, valor a partir do qual praticamente estagna.

Com base na análise apresentada percebe-se que o valor ideal situa-se entre os 90min e os 120min. O valor escolhido acabou por ser os 120min pois é bom ter em mente que os dados usados representam um pequena parte do universo e é preferível fazer sugestões que não sejam aceites do que não as fazer e se tivessem sido feitas teriam sido aceites.

5.4 Escalabilidade

Para além de ser capaz de conceber soluções viáveis, isto é, soluções que façam sentido com base nos dados recolhidos pelos utilizadores, um dos principais requisitos do sistema é a capacidade de resposta em tempo útil. A este tema também está inerente o conceito de escalabilidade.

De forma a classificar a solução obtida em termos de escalabilidade foi feita, aquando da apresentação dos diversos algoritmos, uma análise em termos de complexidade temporal a cada um deles. Existem algoritmos com uma complexidade temporal perfeitamente aceitável (por exemplo, os algoritmos de deteção de pontos de estadia e o de construção dos segmentos de viagem têm complexidade temporal linear, $O(n)$). Mas há especialmente dois algoritmos que apresentam um complexidade considerável, nomeadamente o algoritmo DBSCAN ($O(n * log\ n)$) e o algoritmo ShapeDetetctor ($O(n' * log\ n)$).

Contudo, estes são algoritmos que não são executados com muita frequência. Por exemplo, o algoritmo DBSCAN (o pior em termos de complexidade temporal) será executado de 15 em 15 dias. A ideia de executar de 15 em 15 dias prende-se com o facto de não haver necessidade de

o executar, por exemplo, diariamente. Aliás, o mínimo plausível para a sua execução é semanalmente pois é quando se pode ignorar a semana mais antiga que foi usada no cálculo anterior e passar a usar a mais recente. Se for executado de 15 em 15 dias garante-se que no máximo há uma semana de atraso na determinação de novos padrões de viagem de um utilizador.

Já o algoritmo ShapeDetector (que também apresenta uma complexidade temporal significativa) é executado apenas quando o utilizador o solicita. De forma a minimizar os cálculos desnecessários, sempre que o utilizador solicita o cálculo de sugestões de partilha de veículo são guardados os parâmetros dos algoritmos usados. Assim, no caso do utilizador voltar a pedir para calcular as sugestões e os dados em análise não terem sofrido alterações, a aplicação limita-se a retornar as sugestões que já tinham sido calculadas.

Posto isto, e de acordo com a análise da complexidade temporal feita a todos os algoritmos, pode dizer-se que o sistema escalará bem. A estrutura de dados espacial usada é a fornecida pelo SGBD, o que dá também garantias de que a escalabilidade está assegurada.

5.5 Sumário

A implementação limita-se a seguir o que já havia sido idealizado na conceção da solução.

Contudo, na conceção da solução nem sempre se definem os meios com que se quer chegar aos fins. Por outras palavras, por vezes, é possível variar os algoritmos de forma a obter a mesma solução (por vezes com diferente qualidade).

Os algoritmos DBCLASD e DJ-Cluster (mencionados na "Revisão da literatura", Secção 2.4.2.1) não foram testados de forma a comparar as suas soluções com as do algoritmo DBSCAN porque não foram encontradas implementações disponíveis e o tempo que restava para as suas implementações não era, de todo, suficiente. Não seria justo nem útil para a comparação dos vários algoritmos se estes dois algoritmos tivessem sido implementados de forma mais simples, nomeadamente não fazendo uso de estruturas de dados adequadas (seria deturpar a verdade na medida em que, na verdade, não seria possível concluir nada pois as circunstâncias em que ambos estavam a ser testados não eram as mesmas). Ainda foram contactados os autores de ambos os algoritmos (via email) no sentido de fornecerem uma implementação mas os autores do algoritmos DBCLASD não responderam e os do algoritmo DJ-Cluster não se disponibilizaram a fornecer o código fonte.

Embora existam várias implementações do algoritmo DBSCAN disponíveis, nenhuma (das encontradas) fornecia forma de considerar os pontos como pontos geográficos. Praticamente todas as implementações recorrem ao cálculo da distância euclediana para a determinação da distância entre quaisquer dois pontos (ou seja, consideram que os pontos estão num plano). Assim, enveredou-se pelo caminha da implementação do algoritmo e, desta feita, o cálculo da distância entre 2 quaisquer pontos é feito recorrendo à fórmula de Haversine (descrita na Secção 2.1.1.1).

Em relação à estrutura de dados espaciais, não foi implementada nenhuma na medida em que é usada a oferecida pelo SGBD. Esta decisão está relacionada com a qualidade da estrutura de dados oferecida (bastante elevada) e com a garantia de escalabilidade oferecida pela mesma. Para

além disso, é uma implementação altamente testada e melhorada, o que reduz a probabilidade de existirem problemas/erros.

Por fim, e abordando um pouco a escalabilidade do sistema, pode dizer-se que embora seja um sistema "pesado" (aliás, não era expectável que não o fosse), responde de forma satisfatória ao que lhe é exigido. Note-se que este não é um sistema onde seja suposto obter respostas em tempo real devido à enorme quantidade de dados com que trabalha.

Capítulo 6

Avaliação dos Algoritmos

Para se concluir o que quer que seja tem primeiro de se analisar e avaliar os resultados obtidos. Sem um conjunto de dados suficientemente grande não seria possível obter resultados fiáveis e, portanto, as conclusões estariam bastante condicionadas.

De forma a garantir um conjunto de dados rico foi constituída uma amostra que contribuiu no processo de recolha de dados (sendo este posteriormente alargado a toda a sociedade). Desta forma, para além de se garantir que existiriam dados para serem explorados, a aplicação de recolha de dados foi melhorada de acordo com o *feedback* recebido. Os dados entretanto recolhidos foram catalogados de forma a perceber-se, posteriormente, se os *outputs* dos algoritmos iam ao encontro do que realmente se tinha passado aquando da recolha de dados ou não.

6.1 Metodologia

A recolha de dados é feita com o uso da aplicação MyDrivingDroid (ver Secção 2.5.1.1 para mais detalhes), funcional no sistema operativo Android, versão 2.1 ou superior. No entanto, de forma a garantir o maior número de pessoas disponíveis para a recolha de dados foi disponibilizada informação sobre aplicações grátis para o iPhone e Windows Phone, versão 7.5 ou superior, disponíveis nos respetivos *markets*. Toda a informação sobre a recolha de dados, nomeadamente o objetivo, a forma como deve ser feita, os cuidados a ter, as aplicações a usar consoante o sistema operativo do dispositivo móvel, a forma como os dados podem ser sincronizados com o servidor (ou enviados diretamente para mim no caso de não ser usada a aplicação MyDrivingDroid) e um conjunto de FAQ estão acessíveis numa página *web* criada para o efeito[1].

Nos casos em que não é usada a aplicação MyDrivingDroid os *logs* são criados no formato GPX[2]. Foi então desenvolvido um *parser* que se encarrega de ler esses ficheiros e guardar a informação de lá extraída junto da informação providenciada pela aplicação MyDrivingDroid (na base

[1] http://paginas.fe.up.pt/~ei07084/data_collection_dissertation.html

[2] Formato XML para troca de dados GPS entre aplicações; http://www.topografix.com/gpx.asp

de dados). Foi também criado um grupo no *Google Groups*[3] para a eventualidade de surgirem dúvidas por parte dos utilizadores das aplicações. Este grupo serve também para notificar os utilizadores da aplicação MyDrivingDroid, que subscreveram o grupo, quando é lançada uma nova versão da aplicação, para divulgar o inquérito que foi levado a cabo no âmbito da dissertação e para enviar, normalmente ao fim de semana, um género de lembrete de forma a que os utilizadores não se esquecessem de utilizar a aplicação de recolha de dados.

A avaliação do *output* do sistema, pormenorizada na Secção 6.2, centra-se em 2 pilares: *precision* e *recall*. A avaliação é feita tanto aos padrões de viagem determinados como às sugestões de partilha de veículo feitas. No caso dos padrões de viagem, por exemplo, a *precision* consiste na percentagem de padrões de viagem verdadeiros entre todos os determinados e o *recall* significa a percentagem de padrões de viagem detetados entre todos os que o deviam ser. Quanto maiores estes valores, melhor. Para o caso das sugestões de partilha de veículo as definições são análogas. Note-se que a avaliação foi feita apenas nos dados recolhidos pelos elementos constituintes da amostra na medida em que apenas havia dados catalogados nos dados por eles recolhidos.

Caraterização da Amostra A recolha de dados foi inicialmente levada a cabo por um grupo restrito de 17 elementos, selecionados de forma a garantir que os dados recolhidos conteriam padrões de viagem e possibilidade de sugestões de partilha de veículo. Foram também selecionados elementos sem padrões de viagem e outros com padrões de viagem mas sem possibilidade de sugestão de partilha de veículo com nenhum outro elemento. Num período inicial (cerca de 1 mês) a recolha de dados foi levada a cabo apenas pelos elementos constituintes da amostra e depois, no dia 18 de Fevereiro de 2012, foi alargada à sociedade em geral com o envio de um *email* para toda a faculdade e divulgação em redes sociais.

De forma a catalogar os dados recolhidos foi criado um "diário" onde cada utilizador registava para cada viagem realizada, a origem, o destino, a data e a hora. Foi com base nos registos presentes nesse "diário" e num *feedback* constante dos elementos constituintes da amostra que o *output* dos algoritmos foi sendo validado. Naturalmente, a escolha dos parâmetros dos algoritmos também sofreu influência desse *feedback* constante.

Log book					
Origem	Destino	Data	Hora partida	Hora chegada	Observações
Casa	Trabalho	10/1/2012	8:00:00	9:00:00	-
Casa	Trabalho	11/1/2012	8:00:00	9:00:00	-
Trabalho	Casa	11/1/2012	18:30:00	19:20:00	-

Associação das origens e destinos a coordenadas GPS		
Nome	Google maps	Observações
Casa	41.335044,-8.562984	
Trabalho	41.455259,-8.548576	

Figura 6.1: Exemplo do "diário" preenchido

A amostra tem algumas particularidades. Dos 17 elementos, 12 estudam e/ou trabalham na FEUP e, portanto, a probabilidade de haver locais significativos lá e de haver padrões de viagem com origem e destino lá é bastante grande. Outra particularidade é que todos os elementos da amostra têm os seus padrões de viagem numa zona geográfica relativamente próxima (distritos do Porto e Braga).

[3] https://groups.google.com/

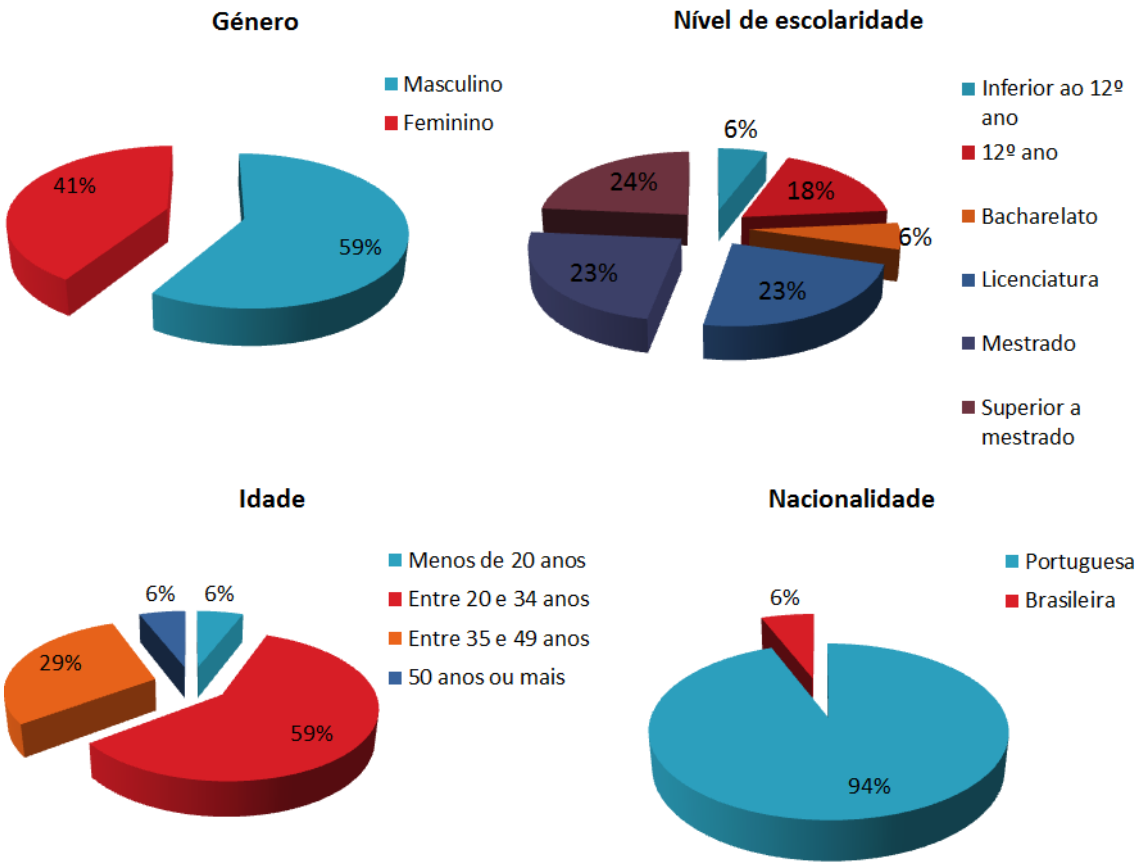


Figura 6.2: Caraterização da amostra

De forma a garantir que era possível determinar várias situações de sugestão de partilha de veículo foram analisadas a priori as rotas habituais de vários elementos da amostra. Dado que a maior parte dos elementos têm a sua atividade profissional na FEUP (professores, alunos e/ou investigadores), pode dizer-se que a amostra era relativamente tendenciosa em alguns aspetos e isto tem obviamente alguma influência nos dados recolhidos. Por exemplo, uma das caraterísticas que é visível nos dados recolhidos por esses elementos é que o desvio padrão da hora de partida dos seus padrões de viagem é tendencialmente maior do que o dos elementos que têm uma atividade profissional que lhes exige horários de entrada fixos.

6.2 Resultados Obtidos

Os resultados obtidos atingiram as expectativas. No entanto, existiram alguns problemas que serão debatidos de seguida (Secção 6.2.3).

A avaliação objetiva por parte de cada elemento da amostra consistiu na avaliação dos padrões de viagem detetados e das sugestões de partilha de veículo que o sistema lhes faz (segundo a configuração do próprio utilizador).

Nas secções seguintes serão analisados os padrões de viagem e as sugestões de partilha de veículo determinadas nos dados recolhidos entre 27 de Fevereiro de 2012 e 15 de Abril de 2012 (dados originados por 26 utilizadores).

6.2.1 Padrões de Viagem

Nos dados em análise foram determinados 60 padrões de viagem (por exemplo, no caso do utilizador ir às segundas e terças de casa para o ginásio são contabilizados 2 padrões de viagem). Se forem ignorados os dias da semana em que ocorrem os padrões de viagem, foram determinados 22 padrões de viagem. Dos 60 padrões de viagem determinados, 12 não pertencem a elementos que fazem parte da amostra, ou seja, pertencem a utilizadores que se interessaram pelo trabalho e participaram no recolha de dados depois da mesma ter sido divulgada para toda a sociedade.

Aquando da avaliação dos padrões de viagem determinados, os mesmos nunca sofreram contestação. Ou seja, a determinação dos padrões de viagem apresenta uma *precision* de 100%. No entanto, inicialmente houve alguma tendência para questionarem o porquê da não determinação de algumas movimentações como padrões de viagem. O problema que se verificou na maior parte dos casos é que os utilizadores tinham em mente que o trajeto $Casa \rightarrow Faculdade$, por exemplo, era feito diariamente e, portanto, devia ser considerado como um padrão de viagem. No entanto, muitas vezes essas viagens eram feitas a diferentes horas do dia (com um grande intervalo de tempo entre elas) e, portanto, não eram determinados padrões de viagem nesses trajetos. O facto do utilizador se esquecer de ligar a aplicação de recolha de dados constituiu também um grande impulsionador da não deteção de algumas movimentações como padrões de viagem por não atingirem o número mínimo de viagens requeridas para o serem considerado.

De todos os padrões de viagem que deviam ser detetados, de acordo com a análise do "diário" de cada elemento, era suposto serem detetados mais 12 do que os que efetivamente foram. A razão para a não deteção desses 12 padrões de viagem, de 1 único utilizador, está bem identificada e está relacionada com o funcionamento do recetor GPS do dispositivo móvel utilizado (ver Secção 6.2.3, intitulada "Principais Problemas", para mais detalhes). Assim, o *recall* estabelece-se nos 80% $\left(\frac{48}{60}\right) = 0,8$.

	É padrão de viagem	Não é padrão de viagem
Classificado como padrão de viagem	48	0
Não classificado como padrão de viagem	12	NA

Tabela 6.1: Tabela semelhante à matriz de confusão para os padrões de viagem

Da Tabela 6.1 (consultar o Anexo B para mais detalhes sobre a matriz de confusão) retira-se que:

Verdadeiros positivos = 48

Verdadeiros negativos = Não se Aplica

Falsos positivos = 12

Falsos negativos = 0

Precision = 100%

Recall = 80%

6.2.2 Sugestões de Partilha de Veículo

A determinação das sugestões de partilha de veículo apresenta melhores resultados que a determinação dos padrões de viagem. Foram determinadas 22 sugestões de partilha de veículo.

	Sugestão correta	Sugestão incorreta
Sugestão determinada	22	0
Sugestão não determinada	0	NA

Tabela 6.2: Tabela semelhante à matriz de confusão para a sugestão de partilha de veículo

Da Tabela 6.2 (consultar o Anexo B para mais detalhes sobre a matriz de confusão) retira-se que:

Verdadeiros positivos = 22

Verdadeiros negativos = Não se Aplica

Falsos positivos = 0

Falsos negativos = 0

Precision = 100%

Recall = 100%

Das seis situações detetáveis com a solução apresentada, apenas uma não se verificou nos dados recolhidos (Figura 6.3).

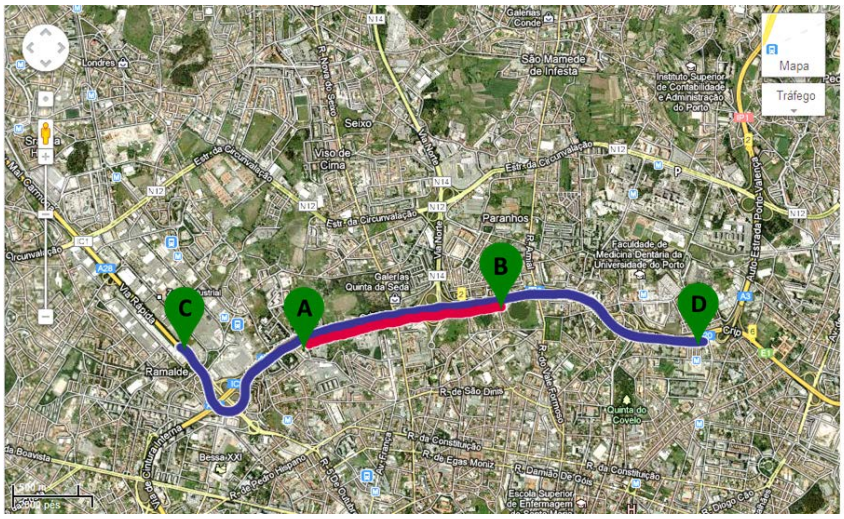


Figura 6.3: Situação não detetada nos dados recolhidos

No entanto, as duas soluções apresentadas na Figura 6.4 (que são muito similares à da Figura 6.3) foram ambas detetadas.

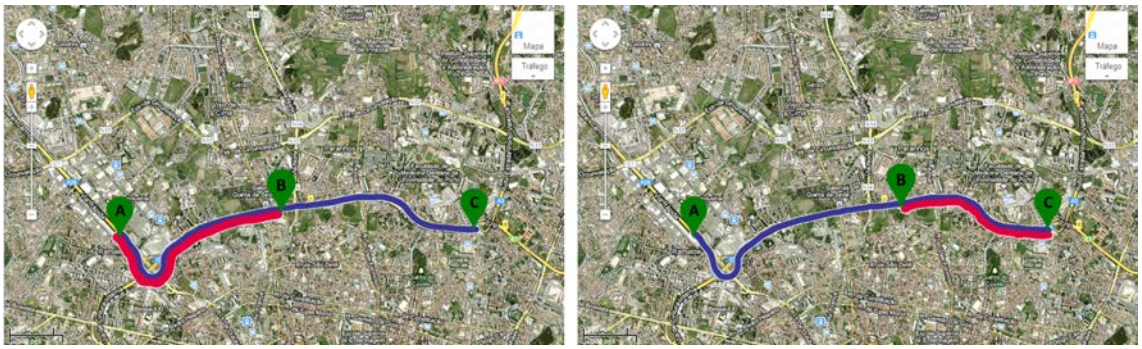


Figura 6.4: Situações detetadas nos dados recolhidos

Com base nessas deteções, e sabendo que a forma de deteção é a mesma nas três situações supracitadas, é perfeitamente plausível concluir-se que se existissem nos dados possibilidades de sugestão como a da Figura 6.3, esta teria sido detetada.

A maior parte das situações detetadas já eram expectáveis (de acordo com a pesquisa feita aquando da seleção dos elementos da amostra). No entanto, surgiram algumas surpresas agradáveis. Por exemplo, percebeu-se que um dos utilizadores que tem um padrão de viagem diário ($Casa \rightarrow Faculdade$) de $\simeq$ 28km pode partilhar veículo nos últimos $\simeq$ 6km com outro utilizador. Devido ao desconhecimento que cada um dos utilizadores em causa tinha sobre a rota do outro (os dois utilizadores não se conhecem sequer) acabou por ser uma surpresa agradável e que acaba por dar mais valor ao trabalho desenvolvido.

Nos dados recolhidos pela amostra era sabido, a priori, as situações que o sistema devia detetar. De acordo com os padrões de viagem determinados, todas as sugestões de partilha de veículo também o foram. Contudo, embora todos os padrões de viagem determinados fossem válidos, alguns não foram determinados. Esta falha deveu-se essencialmente a duas razões:

- Fraca qualidade do sensor GPS que demorava bastante tempo a iniciar a recolha de dados (aconteceu com 1 utilizador)
- Esquecimento de ligar a aplicação de recolha de dados.

Estes 2 problemas são abordados com mais detalhes na secção seguinte, Secção 6.2.3.

6.2.3 Principais Problemas

Os principais problemas encontrados prenderam-se com a falta de qualidade de alguns sensores GPS de alguns *smartphones* e com a dificuldade dos utilizadores, principalmente ao início, em ligarem a aplicação de recolha de dados pois era frequente esquecerem-se.

Figura 6.5: Problema do recetor GPS (formas azuis indicam os supostos 2 locais significativos do utilizador)

Na Figura 6.5 é visível o problema enunciado anteriormente e que está relacionado com a falta de qualidade de alguns recetores GPS de alguns *smartphones* (dos vários modelos usados aconteceu nos Samsung GT-I5500).

No caso apresentado, o utilizador tem 2 locais significativos relativamente próximos ($\simeq$ 1km de distância). O que realmente acontece é que o dispositivo móvel demora bastante a dar início à receção de dados (provavelmente o modelo de telemóvel em questão usa o arranque frio (consultar a Secção 2.1.2 para mais detalhes)). Assim, existem demasiados pontos de estadia "espalhados" no caminho entre os 2 supostos locais significativos do utilizador (circundados pelas formas azuis na Figura 6.5). Pelo próprio funcionamento do algoritmo DBSCAN, ao agrupar os pontos de estadia para determinar os locais significativos do utilizador existem sempre pontos de estadia vizinhos suficientes e suficientemente perto para alargar o *cluster* que está a ser criado e o algoritmo acaba por determinar um único *cluster*, ou seja, um único local significativo (que é a junção dos 2 que deviam ser determinados).

O outro problema enunciado, o esquecimento de dar início à recolha de dados, acabou por ser combatido com o preenchimento do "diário". A "obrigação" de o preencher acabou por servir muitas vezes de lembrete. Contudo, este é um problema que deve ser ultrapassado (iniciando a recolha de dados de forma automática, por exemplo) pois o facto de este ser um processo manual é um dos grandes entraves a uma recolha de dados maciça (consultar a secção intitulada "Trabalho futuro", Secção 7.2, para mais detalhes).

6.3 Análise aos Resultados do Inquérito

O inquérito surgiu para ajudar a perceber até que ponto a sociedade sente necessidade de sistemas como o que ia ser desenvolvido. Objetivamente, tinha como foco concluir se as pessoas

consideram que têm viagens de rotina (e quantas), se têm por hábito partilhar veículo e se têm por norma usar as plataformas existentes de auxílio à partilha de veículo. O grau de satisfação em relação a essas plataformas (para os inquiridos que já as experimentaram) foi também questionado. Um outro objetivo importante era perceber a recetividade de um sistema como o desenvolvido, mais concretamente saber se os inquiridos estariam disponíveis para participar num programa de recomendações automáticas de partilha de veículo.

A julgar pelo número de respostas ao inquérito (mais de 900) este é um tema apelativo para uma grande parte da comunidade. De uma forma em geral pode dizer-se que aplicações com este tipo de fim têm, por norma, uma boa recetividade.

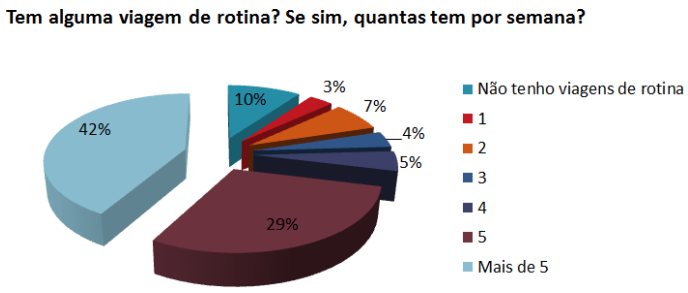


Figura 6.6: Número de viagens de rotina por semana

Para além da boa recetividade, é muito comum que as pessoas tenham padrões de mobilidade razoavelmente bem definidos. 41,9% dos inquiridos consideram mesmo que têm mais de 5 viagens de rotina por semana. Este é um bom indicador da utilidade de sistemas deste género.

No entanto, a esmagadora maioria dos inquiridos (97,2%) nunca experimentou aplicações de auxílio à partilha de veículo. Ou porque simplesmente não conhecem ou porque não se sentem seguros para partilhar veículo nessas condições, etc.

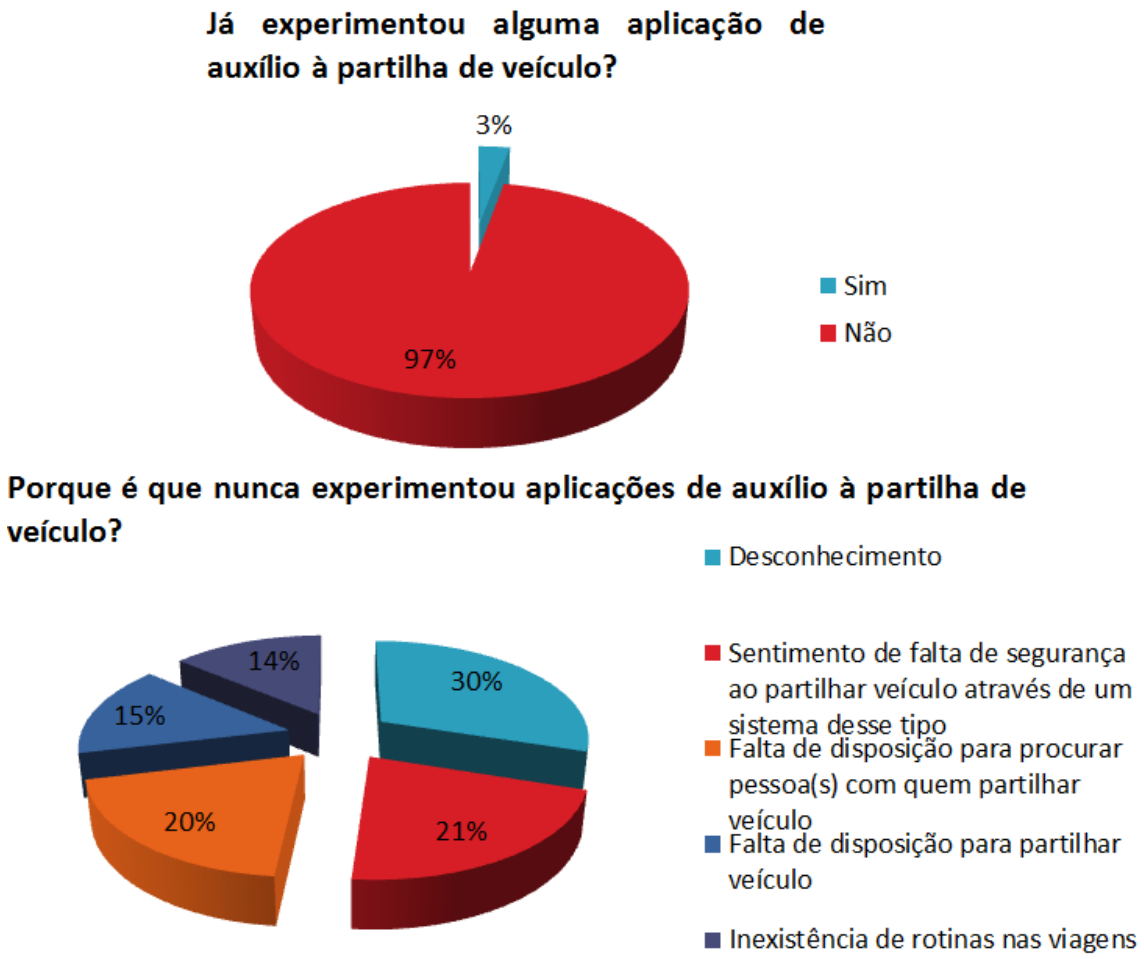


Figura 6.7: Questões sobre o uso de aplicações de auxílio à partilha de veículo

A necessidade de existência de mecanismos que zelem pela segurança dos utilizadores (sistema de avaliação dos utilizadores, possibilidade de comentários, etc) é também uma conclusão que se retira dos resultados deste inquérito.

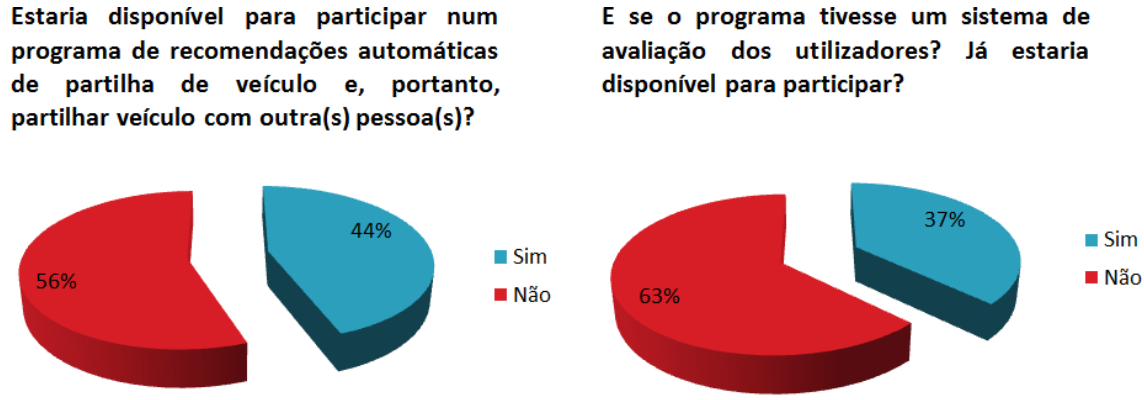


Figura 6.8: Respostas sobre a aceitação em participar num programa de recomendações automáticas de partilha de veículo

O tema segurança é mesmo considerado pelos inquiridos uma questão fundamental. Quando lhes é questionado se estariam disponíveis para participar num programa de recomendações automáticas de partilha de veículo apenas 44,3% responde afirmativamente. No entanto, ao saberem da existência de um sistema de avaliação dos utilizadores, a resposta "Sim" já "dispara" para os 65%. Note-se que a questão do lado direito da Figura 6.8 foi apresentada apenas aos inquiridos que responderam "Não" à questão do lado esquerdo da mesma figura. Assim, os 65% resultam de 37% dos 56% que responderam "Não", que são 20,72% $\left(\frac{37\% \times 56\%}{100\%}\right)$, mais 44% (que já aceitavam participar no estudo mesmo sem o sistema de avaliação), ou seja, 20,72% + 44% $\simeq$ 65%.

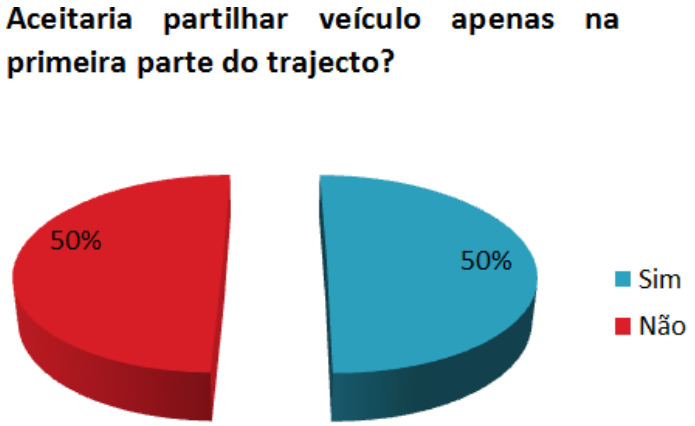


Figura 6.9: Resultado à questão "Aceitaria partilhar veículo apenas na primeira parte do trajeto?"

Curiosamente, ou não, os inquiridos dividem-se quando a questão é se estariam disponíveis para partilhar veículo apenas na primeira parte do trajeto. Ou seja, cerca de metade dos inquiridos (49,8%) não estaria disponível para fazer a ultima parte do trajeto a pé e/ou de transportes públicos.

Capítulo 7

Conclusão

Dos inquiridos que já experimentaram alguma aplicação de auxílio à partilha de veículo (apenas cerca de 3% de um universo de 905 inquiridos!), 72% indicam que a forma atual de pesquisa de utilizadores com quem partilhar veículo não é fácil. Outra conclusão clara é que os inquiridos estão ainda pouco recetivos à ideia de partilharem veículo com pessoas que não conhecem. No entanto, quando surge a hipótese de haver mecanismos para melhorar a segurança dos utilizadores do sistema (a existência de um sistema de avaliação dos utilizadores, por exemplo) a disponibilidade dos inquiridos para participar num programa do género sobe para 65%. A aplicação desenvolvida tenta colmatar a falta de soluções de apoio à partilha de veículo na área de suporte à mobilidade.

Os principais benefícios verificados estão relacionados com a facilidade que os utilizadores passam a ter para partilhar veículo se assim o desejarem. O sistema desenvolvido é altamente configurável o que permite uma customização das sugestões que são feitas à flexibilidade de cada utilizador.

7.1 Conclusões do Trabalho

Em suma, o trabalho desenvolvido consiste num sistema capaz de fazer sugestões de partilha de veículo de forma automática com base nos dados recolhidos por cada utilizador através do seu *smartphone*. Numa primeira fase são determinados os locais de e para onde é habitual cada utilizador viajar e depois são contabilizadas as viagens entre esses mesmos locais de forma a perceber as que são frequentes. Segue-se então a comparação das rotas das viagens frequentes de cada utilizador para se concluir quais as que são realizadas nos mesmos dias da semana, a uma hora semelhante e que se sobrepõem. Dessas comparações podem resultar sugestões de partilha de veículo em várias situações.

Os objetivos propostos foram atingidos na medida em que o sistema é capaz de fazer sugestões de partilha de veículo de forma automática em 6 situações diferentes.

Outro dos objetivos propostos consiste na resposta do sistema em tempo útil e na escalabilidade do mesmo. Esta matéria já foi abordada na Secção 5.4. Ao longo da conceção da solução foi sempre prestada grande atenção a este tema recorrendo constantemente à análise da complexidade temporal dos algoritmos usados, e fazendo uso de uma estrutura de dados espacial adequada, de modo a poder dizer-se que o sistema escalará bem. Note-se, no entanto, que com um volume de utilizadores grande não é de todo expectável que a resposta à procura de sugestões de partilha de veículo seja imediata.

Esta é uma dissertação focada nas técnicas e algoritmos para se chegar ao fim proposto e, por isso, a arquitetura do sistema e, essencialmente, a interface gráfica, embora não tenham sido completamente ignoradas, podem e devem sofrer melhorias significativas de forma a melhorar a experiência do utilizador.

7.2 Trabalho Futuro

Várias ideias para desenvolvimentos futuros foram emergindo ao longo da dissertação. Algumas delas já foram enunciadas, como passar a fazer-se o cálculo dos pontos de estadia e a criação dos segmentos das viagens no dispositivo móvel, aquando da própria recolha de dados, ao invés de os fazer a posteriori, ou seja, no servidor quando os dados são sincronizados. As vantagens destas abordagens já foram enunciadas nas Secções 3.1 e 3.3.

Como já foi mencionado, um grande entrave (provavelmente o maior) a uma recolha de dados verdadeiramente maciça é o facto do utilizador ter de ligar e desligar, pelo menos uma vez por dia, a aplicação de recolha de dados. Idealmente essa tarefa devia ser feita de forma completamente automática. Mesmo o ligar de manhã e desligar à noite a aplicação de recolha de dados não é solução na medida em que há dispositivos móveis que têm baterias que não suportam o sensor GPS tanto tempo ligado.

Em [TB] foi apresentado um sistema com aplicação no âmbito deste problema. A ideia passa pelo uso do acelerómetro para distinguir dois estados de mobilidade: parado e em movimento. Segundo as medidas presentes em [KLGT] o acelerómetro consome muito menos bateria do que o sensor GPS e, portanto, não há problema em estar sempre a ser usado.

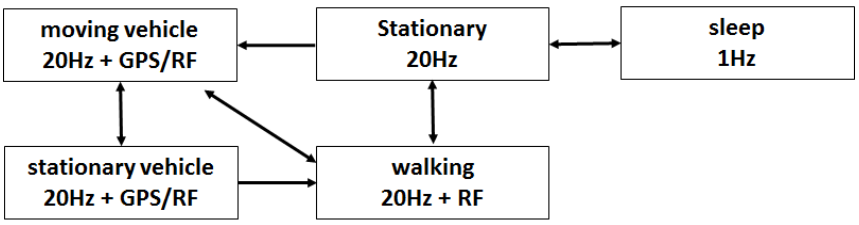


Figura 7.1: Máquina de estados finita do classificador de atividade (adaptado de [TB])

Em [TB] é usado um classificador de atividade que com base no estado atual da máquina de estados finita da Figura 7.1, ativa e/ou desativa sensores do *smartphone*. Na Figura 7.1, 1/20Hz

refere-se ao acelerómetro (mais concretamente à sua frequência), GPS refere-se ao sensor GPS e RF refere-se ao WiFi. A determinação se o utilizador está num veículo particular ou num transporte público (metro, comboio ou mesmo autocarro) é feito através do "*spatio-temporal route matching engine*". Este motor serve para fazer o *match* do percurso que está a ser realizado pelo utilizador com os percursos dos transportes públicos (que estão armazenados no servidor). As horas a que o utilizador passa nas paragens é usada de forma a detetar se o utilizador está a fazer a mesma rota que um dos transportes públicos mas num veículo particular. No caso de ser detetado que o utilizador não está num veículo particular, o sensor GPS fica desativado e, portanto, o uso da bateria é otimizado.

Uma outra implementação futura que se pode revelar bastante interessante está relacionada com as sugestões de partilha de veículo que são feitas. Se há dois padrões de viagem já com sugestão e há um terceiro ao qual é decidido fazer sugestão com apenas um deles, então pode ser boa ideia sugerir a partilha de veículo com o outro também. Embora este último par de padrões de viagem não obedeça a todas os requisitos necessários para que seja feita a sugestão de partilha de veículo, pode ser interessante fazer a sugestão aos três utilizadores na tentativa que cheguem a um acordo impulsionados pela possibilidade de reduzirem ainda mais os custos inerentes à deslocação que habitualmente fazem.

Outra sugestão para trabalho futuro, que constitui um melhoramento à forma como está a ser feito atualmente, está relacionada com o agrupamento dos padrões de viagem. É sabido que quando há mais do que um padrão de viagem em que apenas diferem no dia da semana em que se realizam é verificado se é possível o seu agrupamento, ou seja, é verificado se a rota seguida pelo utilizador é a mesma (ver Secção 4.1 para mais detalhes). Atualmente é apenas verificado se a origem e o destino dos dois padrões de viagem são os mesmos e se os centróides estão a uma distância inferior a um certo limite (configurável). E em caso afirmativo, os padrões de viagem são agrupados. Podia ser feita uma despistagem mais elaborada, nomeadamente dividir a rota em vários troços e verificar se nesses troços a maior parte dos centróides dos segmentos das viagens dos 2 padrões de viagem estão a uma distância relativamente pequena ou não.

Por fim, este é um melhoramento que também já foi abordado anteriormente (ver Secção 4.4 para mais detalhes) e prende-se com o armazenamento da informação sobre a orientação que o utilizador está a seguir em cada momento (Norte, Oeste, etc). Esta informação é fundamental para a determinação de sugestões em que não há partilha de nenhum local significativo nas duas rotas mas existe sobreposição das rotas a determinada altura. O que acontece atualmente é que se houver sobreposição das rotas mas os utilizadores estiverem a circular em sentidos contrários é determinado como uma possível sugestão pois não é tido em conta a orientação seguido pelos utilizadores. Esta melhoria permitiria a implementação da deteção de mais uma situação de sugestão de partilha de veículo.

Anexo A

Distribuição Normal e Métodos Numéricos para Integração

De seguida serão exploradas duas matérias abordadas ao longo da presente dissertação, nomeadamente a distribuição normal e alguns métodos numéricos para integração. A integração através de métodos numéricos foi extremamente útil para a integração da função densidade de probabilidade de forma a calcular, por exemplo, a probabilidade de um determinado utilizador iniciar as suas viagens de um padrão de viagem num determinado intervalo de tempo (que pode ser, por exemplo, o intervalo de tempo em que outro utilizador está disposto a iniciar as suas viagens).

A.1 Distribuição Normal

A distribuição normal (também conhecida como distribuição de Gauss) é uma das mais importantes distribuições da estatística. Descreve fenómenos que são determinados por múltiplas causas que em geral interagem entre si e possui grande uso na estatística inferencial. Esta é uma distribuição inteiramente descrita pelos seus parâmetros: média e desvio padrão. Ou seja, conhecendo-se estes 2 parâmetros, consegue-se determinar qualquer probabilidade.

De acordo com o teorema do Limite Central, a distribuição normal serve de aproximação para o cálculo de outras distribuições quando o número de observações é tendencialmente grande. Toda a soma de variáveis aleatórias independentes de média finita e variância limitada é aproximadamente Normal, desde que o número de termos da soma seja suficientemente grande[1].

Função Densidade de Probabilidade A função densidade de probabilidade é uma função que representa a distribuição de probabilidade caso a variável aleatória seja contínua. É uma função normalmente denotada por $F_X(x)$ e que satisfaz as seguintes condições

$$F_X(x) = P[a \leq X \leq b] = \int_a^b f(x)dx \tag{A.1}$$

[1] Fonte: http://pt.wikipedia.org/wiki/Distribui%C3%A7%C3%A3o_normal, acedido em: 15/05/2012

$$F_X(x) = P[X \leq x] = \int_{-\infty}^{x} f(X)dX \tag{A.2}$$

Na Equação A.1 está presente que uma variável aleatória contínua tem densidade $f(x)$ se f é uma função não negativa e integrável tal que a probabilidade no intervalo $[a,b]$ é dada por $\int_a^b f(x)dx$. Já na Equação A.2, mostra-se que a probabilidade da variável aleatória X assumir um valor $\leq x$ é dada pela integral $\int_{-\infty}^{x} f(X)dX$.

A função densidade de probabilidade da distribuição normal com média μ e variância σ é definida da seguinte forma

$$f(x,\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{(x-\sigma)^2}{2\sigma^2}\right)}, -\infty \leq x \leq +\infty, \sigma > 0 \tag{A.3}$$

Assim, a função densidade de probabilidade apresenta sempre uma forma sinusoidal. O quão acentuada é esta forma está relacionada com o desvio padrão da distribuição.

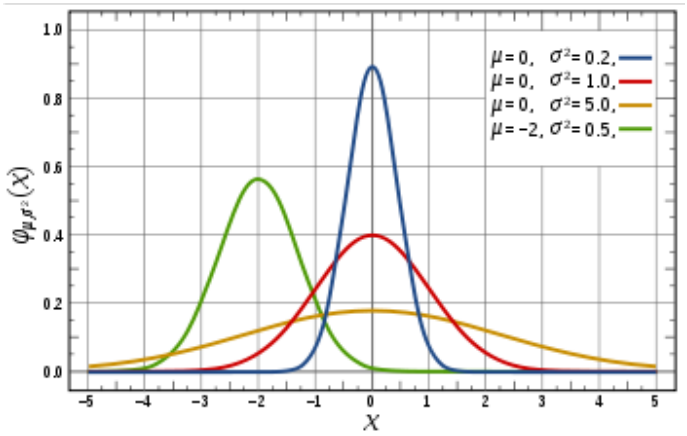


Figura A.1: Função densidade de probabilidade (http://en.wikipedia.org/wiki/Normal_distribution)

O conceito de função densidade de probabilidade é muito semelhante ao conceito de função de probabilidade, que serve para o caso de variáveis aleatórias discretas.

Uma variável aleatória discreta tem um número definido de possíveis ocorrências. Por exemplo, a variável aleatória "resultado de um dado" tem apenas 6 ocorrências possíveis: 1, 2, 3, 4, 5 e 6. Por isso, a função de probabilidade a ela associada também só pode assumir 6 valores, que necessariamente somarão 1. Contudo, uma variável aleatória contínua tem um número infinito de ocorrências. Por exemplo, a variável aleatória "idade de uma pessoa" pode assumir infinitos valores. Por isso, se simplesmente se tentar calcular $P(X = x)$ como faz uma função de probabilidade para uma variável aleatória discreta, chegaremos ao seguinte:

$$P(X = x) \leq P(x - \varepsilon \leq X \leq x) = P(X \leq x) - P(X \leq x - \varepsilon), \forall \varepsilon > 0 \tag{A.4}$$

Portanto,

$$0 \leq P(X = x) \leq \lim_{\varepsilon \to +\infty} [P(X \leq x) - P(X \leq x - \varepsilon)] = 0 \tag{A.5}$$

Ou seja, a probabilidade de a variável aleatória contínua X assumir um determinado valor x é zero. Por isso é que a função densidade de probabilidade não trabalha com valores pontuais, e sim com intervalos infinitesimais[2].

Regra 68-95-99.7 A distribuição normal tem uma característica (conhecida como a regra 68-95-99.7) que indica que entre $-\sigma$ e $+\sigma$ está cerca de 68% do conjunto.

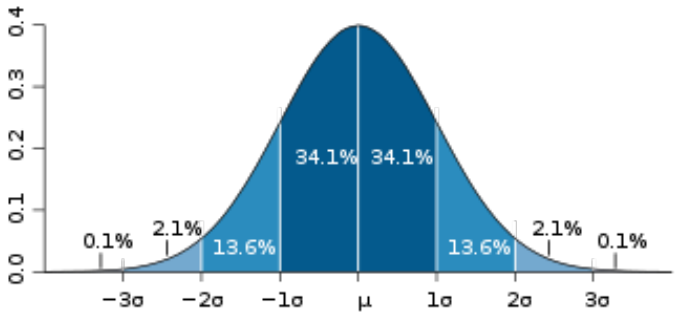


Figura A.2: Regra 68-95-99.7 (http://pt.wikipedia.org/wiki/Distribui%C3%A7%C3%A3o_normal)

Dois desvios padrão desde a média representam cerca de 95% e três desvios padrão cobrem cerca de 99.7%.

A.2 Métodos Numéricos

Por método numérico entende-se um método para calcular a solução de um problema realizando apenas uma sequência finita de operações aritméticas. Ao invés de serem usados métodos analíticos, que podem ser demasiado complexos, são usados métodos numéricos que conduzem a soluções aproximadas. A diferença entre o valor obtido (aproximado) e o valor exato é designado por erro.

No âmbito desta dissertação, será usado um método numérico para o cálculo de integrais definidas da função densidade de probabilidade.

[2]Fonte: http://pt.wikipedia.org/wiki/Fun%C3%A7%C3%A3o_densidade_de_probabilidade, acedido em: 15/05/2012

A.2.1 Regra dos Trapézios

Neste método, a ideia central consiste em substituir, em cada intervalo, o arco da curva pela sua corda, calculando em seguida a área sob a poligonal assim definida. Assim, para o primeiro trapézio temos

$$\int_{x_0}^{x_1} y dx = \frac{h}{2}[y_1 + y_0] \tag{A.6}$$

E para o segundo trapézio:

$$\int_{x_1}^{x_2} y dx = \frac{h}{2}[y_2 + y_1] \tag{A.7}$$

E assim sucessivamente até:

$$\int_{x_{n-1}}^{x_n} y dx = \frac{h}{2}[y_n + y_{n-1}] \tag{A.8}$$

Somando todos os termos, temos que

$$\int_{x_0}^{x_n} y dx = \frac{h}{2}[y_0 + 2y_1 + ... + 2y_{n-1} + y_n] \tag{A.9}$$

A.2.2 Regra de Simpson

A regra de Simpson é uma regra mais eficaz do que a regra dos trapézios. Nesta regra a curva é substituída pelas parábolas definidas por cada trio de pontos consecutivos (ao invés de substituir a curva pelas cordas definidas por cada par de pontos consecutivos).

Portanto, a parábola é da forma

$$y = y_i + \frac{x - x_i}{h}(y_{i+1} - y_i) + \frac{(x - x_i).(x - x_{i+1})}{2h^2}(y_{i+2} - 2y_{i+1} + y_i) \tag{A.10}$$

Daqui resulta que

$$\int_{x_i}^{x_{i+2}} y dx = \int_{x_i}^{x_{i+2}} \left[y_i + \frac{x - x_i}{h}(y_{i+1} - y_i) + \frac{(x - x_i).(x - x_{i+1})}{2h^2}(y_{i+2} - 2y_{i+1} + y_i) \right] dx \tag{A.11}$$

Que é igual a

$$2h \left[y_i + y_{i+1} - y_i + \frac{y_{i+2} - 2y_{i+1} + y_i}{6} \right] = \frac{h}{3}[y_i + 4y_{i+1} + y_{i+2}] \tag{A.12}$$

De modo que a soma dá

$$\int_{x_0}^{x_{2n}} y dx = \frac{h}{3}[y_0 + 4y_1 + 2y_2 + 4y_3 + ... + 4y_{2n-3} + 2y_{2n-2} + 4y_{2n-1} + y_{2n}] \tag{A.13}$$

A.2.3 Erro no Cálculo

O erro final pode ter tido origem em várias fontes, nomeadamente:

- Erros inerentes ao modelo matemático
- Erros inerentes aos dados
- Erros de arredondamento
- Erros de truncatura

Erros inerentes ao modelo matemático: Um modelo matemático raramente oferece uma representação exata dos fenómenos reais. Na grande maioria dos casos são apenas modelos idealizados, já que ao estudar os fenómenos da natureza vemo-nos forçados, regra geral, a aceitar certas condições que simplificam o problema por forma a torná-lo tratável.

Erros inerentes aos dados: Um modelo matemático não contém apenas equações e relações, também contém dados e parâmetros que, frequentemente, são medidos experimentalmente, e portanto, aproximados.

Erros de arredondamento: Quer os cálculos sejam efetuados manualmente quer obtidos por computador, somos conduzidos a utilizar uma aritmética de precisão finita, ou seja, apenas podemos ter em consideração um número finito de dígitos.

Erros de truncatura: Muitas equações têm soluções que apenas podem ser construídas no sentido que um processo infinito possa ser descrito como limite da solução em questão. Por definição, um processo infinito não pode ser completado, por isso tem de ser truncado após certo número finito de operações. Em muitos casos, o erro de truncatura é precisamente a diferença entre o modelo matemático e o modelo numérico.

Existem 2 tipos de erros associados ao uso de métodos numéricos para resolver um problema num computador: os erros de arredondamento e os erros de truncatura. Como consequência da ocorrência destes erros, as soluções numéricas obtidas são, em geral, soluções aproximadas.

A.3 Discussão

A distribuição normal será usada para descrever o fenómeno de partida/chegada de cada utilizador aos seus locais significativos. Esta assunção permitirá calcular para cada intervalo de tempo a probabilidade do utilizador partir/chegar a um determinado local.

O cálculo da probabilidade será feito através do método de Simpson. Este foi o método escolhido por se revelar mais eficaz que o método do trapézios. Um defeito óbvio da regra dos trapézios é o de cometer um erro sistemático em intervalos em que a segunda derivada da integranda mantém sinal constante. Para o evitar e para aumentar, na generalidade, a precisão da aproximação, a regra de Simpson em vez de substituir a curva pelas cordas definidas por cada par de pontos consecutivos, substitui-a pelas parábolas definidas por cada trio de pontos consecutivos.

Embora haja mais métodos numéricos que possibilitem o cálculo de integrais (como o método de Romberg, por exemplo), os dois apresentados apresentam uma qualidade apreciável (quer em termos de erro, quer em termos de eficiência temporal) para o âmbito desta dissertação.

Anexo B

Matriz de confusão

A matriz de confusão permite a visualização do desempenho de um algoritmo, normalmente um algoritmo de aprendizagem supervisionada. O nome da matriz deriva do facto de esta tornar fácil a visualização se o sistema está a confundir as classes das instâncias que classificou ou não.

Classe	predita C_+	predita C_-	Taxa de Erro da Classe	Taxa de Erro Total
verdadeira C_+	T_P	F_N	$\frac{F_N}{T_P + F_N}$	$\frac{F_P + F_N}{n}$
verdadeira C_-	F_P	T_N	$\frac{F_P}{F_P + T_N}$	

T_P = Verdadeiro Positivo (True Positive)
F_N = Falso Negativo (False Negative)
F_P = Falso Positivo (False Positive)
T_N = Verdadeiro Negativo (True Negative)
$n = (T_P+F_N+F_P+T_N)$

Figura B.1: Matriz de confusão para duas classes

Na Figura B.1 é visível a estrutura de uma matriz de confusão para duas classes. Contudo, é possível construir uma matriz de confusão com *n* classes.

Numa matriz de confusão:

- O número de acertos de cada classe localiza-se na diagonal principal da matriz
- Os restantes elementos (casos em que o número da linha é diferente do número da coluna) representam erros na classificação

Numa matriz de confusão é ainda possível calcular métricas como:

- Precisão: $\frac{\#T_P}{\#T_P+\#F_N}$
- *Recall*: $\frac{\#T_P}{\#T_P+\#F_P}$
- Taxa de falsos positivos: $\frac{\#F_P}{\#F_P+\#T_N}$

Existem, no entanto, outras métricas que se podem calcular. As métricas mais úteis dependem sempre do contexto do problema. Pode-se inclusive associar custos aos falsos positivos e negativos para avaliar a verdadeira *accuracy* do modelo.

Em aprendizagem não supervisionada a matriz de confusão é normalmente chamada de *match matrix*. Esta matriz é também usada noutras áreas que não a inteligência artificial e é normalmente conhecida por "matriz do erro" ou "tabela de contingência".

Anexo C

Visualizações Geográficas

Nesta secção são apresentadas algumas figuras que ilustram os dados recolhidos por dois utilizadores. Nas figuras está presente informação extraída dos dados recolhidos entre 27 de Fevereiro de 2012 e 15 de Abril de 2012.

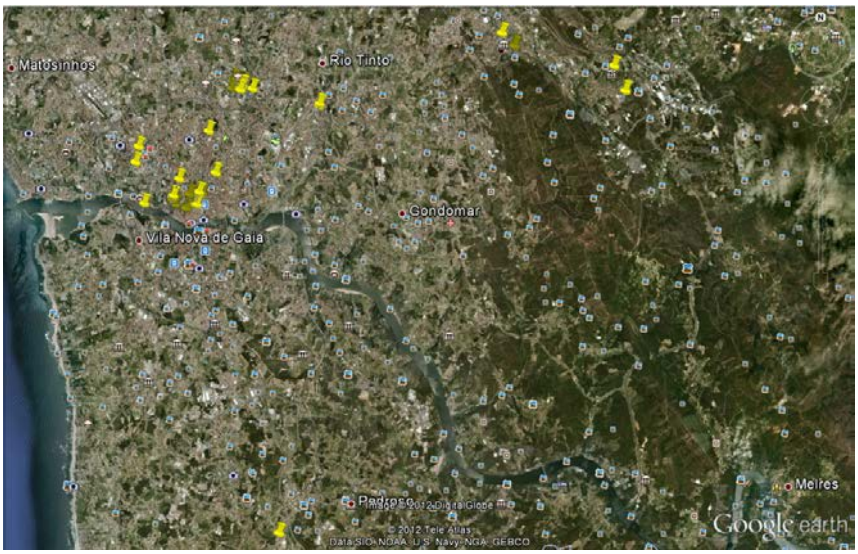


Figura C.1: Utilizador A: pontos de estadia do utilizador

Na Figura C.1 são visíveis todos os pontos de estadia do utilizador A. É percetível que é no centro da cidade do Porto que existe um maior aglomerado de pontos e, portanto, foi feito um *close up* a essa área (ver Figura C.2).

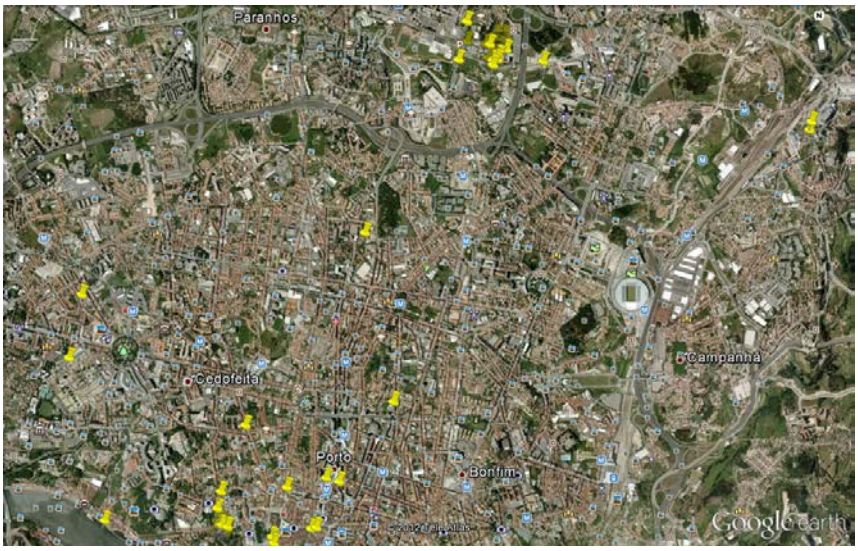


Figura C.2: Utilizador A: *close up* aos pontos de estadia do centro da cidade do Porto

Dos pontos de estadia da Figura C.1 resultaram os locais significativos presentes na Figura C.3.

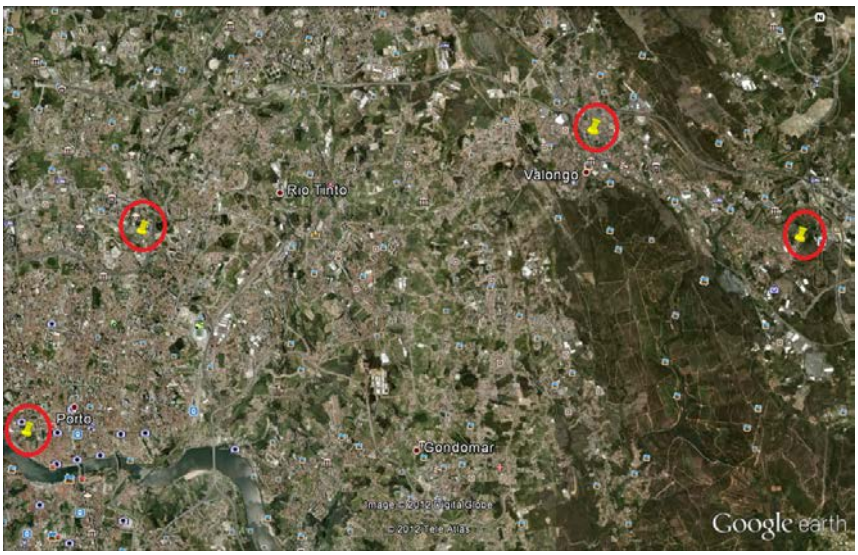


Figura C.3: Utilizador A: locais significativos do utilizador

Embora o utilizador tenha 4 locais significativos, apenas 2 deles têm padrões de viagem asso- 2
ciados (Figura C.4).

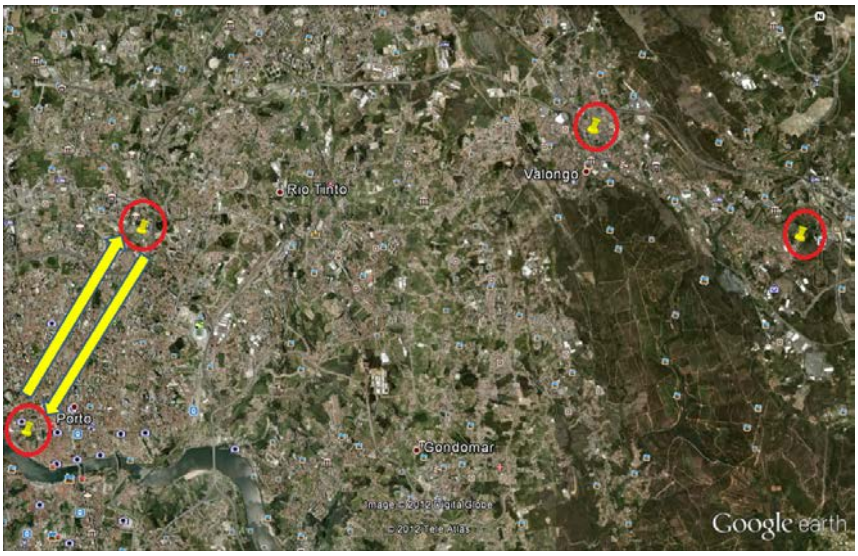


Figura C.4: Utilizador A: padrões de viagem do utilizador

Verificou-se que embora fossem realizadas viagens com alguma frequência para os outros 2 locais significativos, não existia um dia da semana específico nem um espaço temporal bem definido em que essas viagens fossem realizadas.

Um dos padrões de viagem do utilizador B é apresentado na Figura C.5. Este padrão de viagem é realizado mais ou menos à mesma hora que o padrão de viagem caraterizado pela seta que aponta para cima na Figura C.4 (que é um dos padrões de viagem do utilizador A).

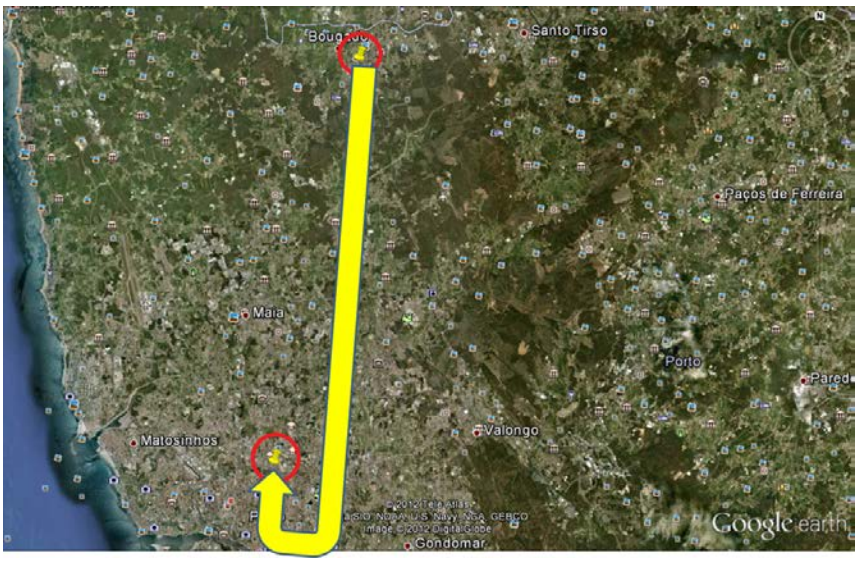


Figura C.5: Utilizador B: um dos padrões de viagem do utilizador

Destes 2 padrões de viagem resultou uma sugestão de partilha de veículo.

Figura C.6: Utilizadores A e B: padrões de viagem de ambos

A Figura C.6 acaba por representar a proximidade dos 2 padrões de viagem que, em conjunto com a hora de realização dos mesmos, possibilitou a sugestão de partilha de veículo. [2]

Anexo D

Inquérito e Resultados

Foi levado a cabo um inquérito de forma a perceber até que ponto a sociedade está satisfeita com as soluções atuais e de que forma se poderia melhorar as mesmas. O inquérito obteve 905 respostas e as perguntas e respostas do mesmo são apresentadas a seguir.

1. Género

- Masculino
- Feminino

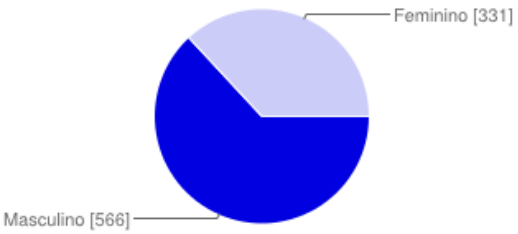


Figura D.1: Resultado das respostas à questão 1

2. Idade

- Menos de 20 anos
- Entre 20 e 34 anos
- Entre 35 e 49 anos
- 50 ou mais anos

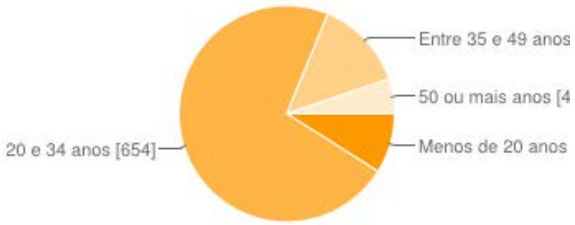


Figura D.2: Resultado das respostas à questão 2

3. Nível de escolaridade

- Inferior ao 12º ano
- 12º ano
- Bacharelato
- Licenciatura
- Mestrado
- Superior a mestrado

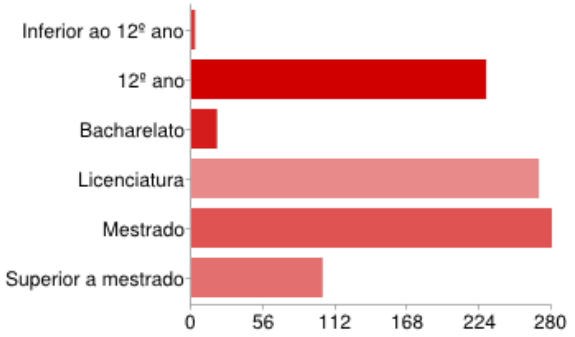


Figura D.3: Resultado das respostas à questão 3

4. Tem alguma viagem de rotina? Se sim, quantas tem por semana?

- Não tenho viagens de rotina
- 1
- 2
- 3

- 4
- 5
- Mais de 5

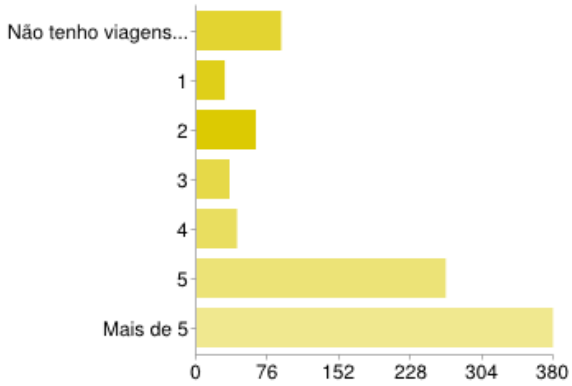


Figura D.4: Resultado das respostas à questão 4

5. Conhece alguém que partilhe alguma das suas viagens de rotina? Tem por hábito, sempre que possível, partilhar veículo com essa(s) pessoa(s)?

- Não conheço ninguém que partilhe alguma das minhas viagens de rotina
- Sim, tenho por hábito partilhar veículo sempre que possível
- Não, não tenho por hábito partilhar veículo nas minhas viagens de rotina

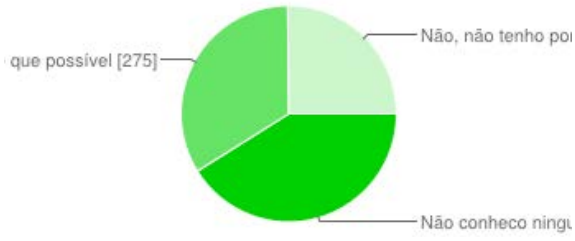


Figura D.5: Resultado das respostas à questão 5

6. Já experimentou alguma aplicação de auxílio à partilha de veículo?

- Sim
- Não

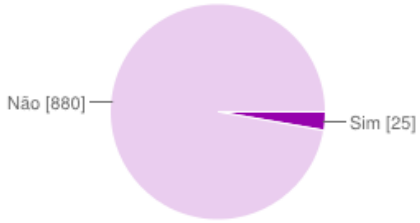


Figura D.6: Resultado das respostas à questão 6

7. Existem mecanismos para dar garantias em termos de segurança aos utilizadores que partilhem veículo?

- Sim
- Não, mas também não é uma característica fundamental neste tipo de aplicações
- Não, e esta seria uma melhoria substancial que valeria a pena ser implementada

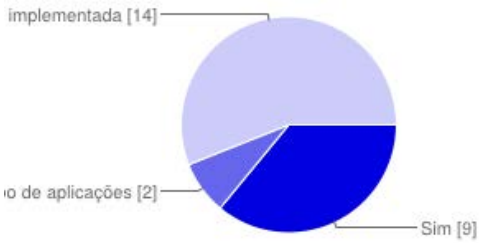


Figura D.7: Resultado das respostas à questão 7

8. Existe a possibilidade de visualização e/ou representação do trajecto das rotas num mapa?

- Sim
- Não, mas também não é uma característica fundamental neste tipo de aplicações
- Não, e esta seria uma melhoria substancial que valeria a pena ser implementada

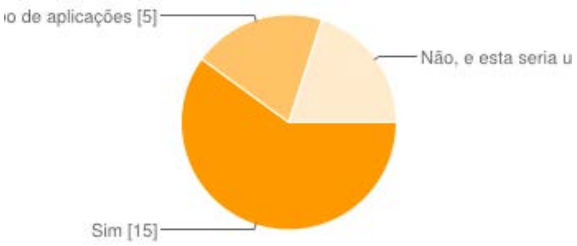


Figura D.8: Resultado das respostas à questão 8

9. É fácil a procura de utilizadores com quem partilhar viagem?

- Sim
- Não, mas a forma atual já é satisfatória
- Não, e esta seria uma melhoria substancial que valeria a pena ser implementada

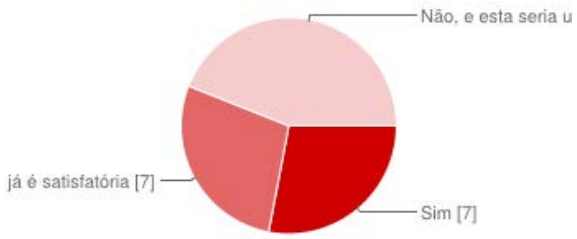


Figura D.9: Resultado das respostas à questão 9

10. Existe um sistema de avaliação dos utilizadores?

- Sim
- Não, mas também não é uma característica fundamental neste tipo de aplicações
- Não, e esta seria uma melhoria substancial que valeria a pena ser implementada

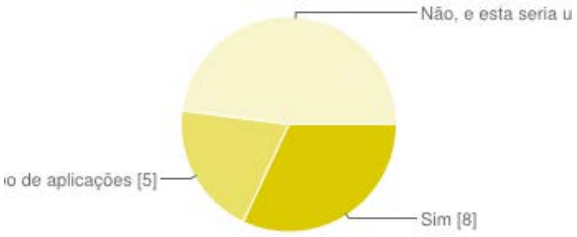


Figura D.10: Resultado das respostas à questão 10

11. Existe um sistema de recomendações automáticas de partilha de veículo?

- Sim
- Não, mas também não é uma característica fundamental neste tipo de aplicações
- Não, e esta seria uma melhoria substancial que valeria a pena ser implementada

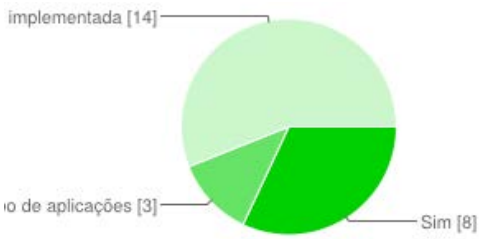


Figura D.11: Resultado das respostas à questão 11

12. Porque é que nunca experimentou aplicações de auxílio à partilha de veículo? (É dada a possibilidade do inquirido selecionar mais do que uma resposta)

- Desconhecimento
- Sentimento de falta de segurança ao partilhar veículo através de um sistema desse tipo
- Falta de disposição para procurar pessoa(s) com quem partilhar veículo
- Falta de disposição para partilhar veículo
- Inexistência de rotinas nas viagens

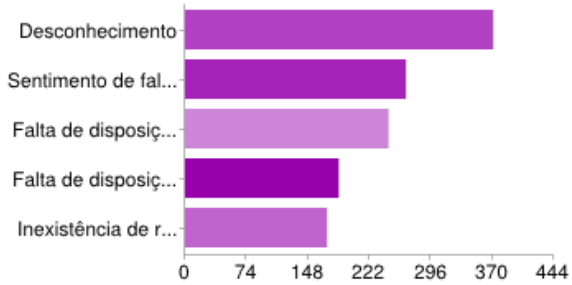


Figura D.12: Resultado das respostas à questão 12

13. Estaria disponível para participar num programa de recomendações automáticas de partilha de veículo e, portanto, partilhar veículo com outra(s) pessoa(s)?

- Sim
- Não

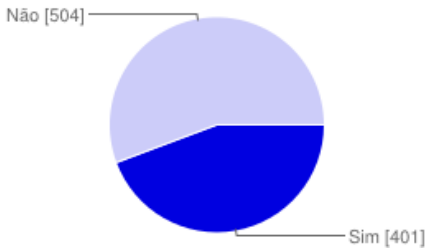


Figura D.13: Resultado das respostas à questão 13

14. E se o programa tivesse um sistema de avaliação dos utilizadores? Já estaria disponível para participar?

- Sim
- Não

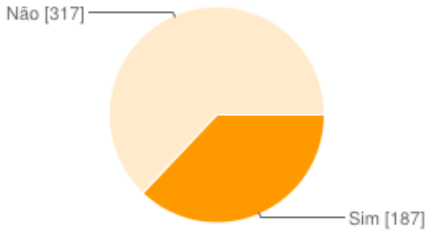


Figura D.14: Resultado das respostas à questão 14

15. Aceitaria partilhar veículo apenas na primeira parte do trajecto?

- Sim
- Não

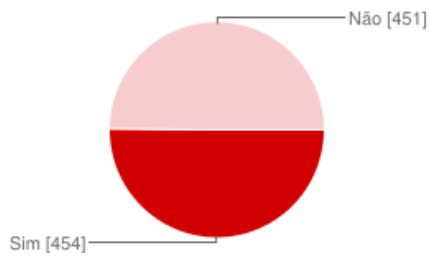


Figura D.15: Resultado das respostas à questão 15

Note-se que algumas questões são condicionais, isto é, só eram feitas se houvesse uma certa conjugação de respostas anteriormente.

Referências

[ANN09] G Agamennoni, J Nieto e E Nebot. Mining gps data for extracting significant places. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 855–862. IEEE, 2009.

[AS03] Daniel Ashbrook e Thad Starner. Using GPS to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing*, 7(5):275–286, October 2003.

[ASVR05] W ABRAHAMSE, L STEG, C VLEK e T ROTHENGATTER. A review of intervention studies aimed at household energy conservation. *Journal of Environmental Psychology*, 25(3):273–291, September 2005.

[BEH+06] J Burke, D Estrin, M Hansen, A Parker, N Ramanathan, S Reddy e M B Srivastava. Participatory Sensing. *American Journal of Public Health*, pages 1–5, 2006.

[BF79] J O N Louis Bentley e Jerome H Friedman. Data Structures for Range Searching. *Computing*, (4), 1979.

[BSS+90] Norbert Beckmann, Ralf Schneider, Bernhard Seeger, Praktuche Informatlk, Umversltaet Bremen, D Bremen e West Germany. The R * -tree : An Efficient and Robust Access Method for Points and Rectangles. *Work*, pages 322–331, 1990.

[DDK12] George Dimitrakopoulos, Panagiotis Demestichas e Vera Koutra. Intelligent Management Functionality for Improving Transportation Efficiency by Means of the Car Pooling Concept. *IEEE Transactions on Intelligent Transportation Systems*, 13(2):424–436, June 2012.

[DG11] Samuel Dário e Falcão Gabriel. Extração de Informação de Padrões Pessoais de Tempo e Espaço. 2011.

[EKSX] Martin Ester, Hans-peter Kriegel, Jörg Sander e Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Computer*.

[FDK+09] Jon Froehlich, Tawanna Dillahunt, Predrag Klasnja, Jennifer Mankoff, Sunny Consolvo, Beverly Harrison e James A Landay. UbiGreen : Investigating a Mobile Tool for Tracking and Supporting Green Transportation Habits. *Carbon*, pages 1043–1052, 2009.

[Fis08] Corinna Fischer. Feedback on household electricity consumption: a tool for saving energy? *Energy Efficiency*, 1(1):79–104, May 2008.

[Gut84] Antomn Guttman. Scientific Commons: R-trees: A Dynamic Index Structure for Spatial Searching. August 1984.

REFERÊNCIAS

[JGP+03] By John, P Snyder U S Geological, Survey Professional, U S Government, Printing Office e Excerpt Produced. Excerpts From : Map Projections A Working Manual. *Meridian*, 2003.

[JM07] Mitch J Duncan e W Kerry Mummery. GIS or GPS? A comparison of two methods for assessing route taken during active transport. *American Journal of Preventive Medicine*, 33(1):51–53, 2007.

[KLGT] Mikkel Baun Kjaergaard, Jakob Langdal, Torben Godsk e Thomas Toftkjaer. EnTracked - Energy-Efficient Robust Position Tracking for.pdf.

[KR90] Leonard Kaufman e Peter J. Rousseeuw, editors. *Finding Groups in Data*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Hoboken, NJ, USA, March 1990.

[KWSB04] Jong Hee Kang, William Welbourne, Benjamin Stewart e Gaetano Borriello. Extracting Places from Traces of Locations. *Work*, 2004.

[LZX+08] Quannan Li, Yu Zheng, Xing Xie, Yukun Chen, Wenyu Liu e Wei-ying Ma. Mining User Similarity Based on Location History. *Architecture*, (c), 2008.

[Mit97] Thomas M. Mitchell. Machine Learning. March 1997.

[MS00] Natalia Marmasse e Chris Schmandt. Location-Aware Information Delivery with ComMotion. pages 157–171, 2000.

[NHS02] Raymond T Ng, Jiawei Han e Ieee Computer Society. CLARANS : A Method for Clustering Objects for Spatial Data Mining. *Knowledge Creation Diffusion Utilization*, 14(5):1003–1016, 2002.

[RAV+11] Joao G. P. Rodrigues, Ana Aguiar, Fausto Vieira, Joao Barros e Joao P. Silva Cunha. A mobile sensing architecture for massive urban scanning. *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1132–1137, October 2011.

[Sub75] Annual Subscription. Survey review. *Survey Review*, xxm(176), 1975.

[TB] Arvind Thiagarajan e James Biagioni. Cooperative Transit Tracking using Smartphones. *Challenges*.

[WARS09] Charles William, Hawthorne Amick, David P Reed e Arthur C Smith. An Architecture for Socially Mobile Collaborative Sensing and its Implementation by by. *Source*, (2008), 2009.

[Wei99] Mark Weiser. The computer for the 21 st century. *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(3):3–11, July 1999.

[WHR] Rick Wash, Libby Hemphill e Paul Resnick. Design Decisions in the RideNow Project. *Design*, pages 1–4.

[Wil] Gregory Piatetsky-shapiro William J. Frawley. Knowledge Discovery in Databases: an Overview.

[WM08] C Weber e H Matthews. Quantifying the global and distributional aspects of American household carbon footprint. *Ecological Economics*, 66(2-3):379–391, June 2008.

[XEKS] Xiaowei Xu, Martin Ester, Hans-peter Kriegel e Jörg Sander. A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases. *History*.

[ZBST07] Changqing Zhou, Nupur Bhatnagar, Shashi Shekhar e Loren Terveen. Mining Personally Important Places from GPS Tracks. *Machine Learning*, pages 517–526, 2007.

[ZFL+04] Changqing Zhou, Dan Frankowski, Pamela Ludford, Shashi Shekhar e Loren Terveen. Discovering Personal Gazetteers : An Interactive Clustering Approach. *Human Factors*, 2004.