Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Departamento de Informática

Pesquisa de Eventos Geográficos Semelhantes: Trajectórias de Objectos em Movimento

Fábio Augusto Grabulho Afonso (30218)

Dissertação para obtenção do Grau de Mestre em Engenharia Informática

Orientadora
Prof. Doutora Fernanda Maria Barquinha Tavares Vieira Barbosa

Co-Orientadora
Prof. Doutora Maria Armanda Simenta Rodrigues Grueau

Composição do Júri
Prof. Doutor Pedro Abílio Duarte de Medeiros (Presidente)
Prof. Doutor Luis Filipe Fernandes Silva Marcelino (Arguente)
Prof. Doutora Fernanda Maria Barquinha Tavares Vieira Barbosa (Vogal)

Março 2011

Copyright

Agradecimentos

Em primeiro lugar, gostaria de agradecer à Professora Fernanda Barbosa, pelo seu apoio, disponibilidade, motivação, críticas e sugestões dadas durante a orientação desta dissertação, que enriqueceram, sem dúvida, este documento. Gostaria ainda de lhe agradecer pela influência na minha vida não académica devido à sua exigência, organização, excelência e companheirismo.

Ao corpo docente do Departamento de Informática da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, especialmente à Professora Armanda Rodrigues, por todo o seu apoio.

Ao Centro de Informática e Tecnologia de Informação (*CITI*), pela atribuição de uma Bolsa de Investigação Cientifica que possibilitou a realização deste trabalho.

À equipa Transactional Systems do *CITI*, nomeadamente ao Professor João Lourenço, que gentilmente disponibilizou uma máquina para a realização de testes experimentais, sem a qual seria extremamente difícil concluir esta dissertação nos prazos estabelecidos.

Ao colega Ângelo Sarmento que, amavelmente, contribuiu de forma importantíssima para a concretização deste trabalho.

Aos meus amigos mais próximos, pelo companheirismo, amizade, apoio e motivação demonstrados ao longo do tempo.

À minha família mais próxima, em especial aos meus pais, Benjamim Afonso e Maria Vitória Afonso, e à minha irmã, Sónia Afonso, por tudo o que sempre fizeram e continuam a fazer por mim.

Conseguir agradecer a todos aqueles que ajudaram a realizar esta dissertação não é tarefa fácil. O maior risco que se corre ao agradecer individualmente não é decidir quem incluir, mas quem não mencionar. Assim sendo, a todos os que contribuíram de forma directa ou indirecta para a realização deste trabalho, gostaria de lhe revelar a minha enorme gratidão.

Resumo

Nos dias de hoje rara é a pessoa que não possui um aparelho de Geo-Posicionamento por Satélite (*GPS*), esteja este no seu automóvel ou no seu bolso, visto que os componentes necessários para o seu correcto funcionamento são agora adquiridos a um preço bastante convidativo.

Devido à existência de um enorme volume de dados georreferenciados, sendo alguns deles referentes a movimentações de pessoas e/ou eventos ambientais/sociais, tais como furacões, migrações, tráfego, transportes, tornou-se necessário descobrir e aperfeiçoar processos que agilizem o tratamento eficaz e eficiente de pesquisas por semelhança neste tipo de dados de modo a se poder prever/analisar possíveis catástrofes, assim como ajudar na tomada de decisões referente a estratégias.

Neste trabalho foi realizada uma avaliação das técnicas existentes na pesquisa por semelhança de eventos geográficos, nomeadamente trajectórias. Para tal foi realizado um estudo de todas as técnicas existentes que estão envolvidas neste tipo de pesquisa, em particular as funções de semelhança e os métodos de indexação mais relevantes utilizados nesta área de investigação. Foi realizada uma avaliação das pesquisas por semelhança em diferentes espaços métricos de trajectórias com as estruturas de dados métricas *Recursive Lists of Clusters 2* (*RLC2*) e *Metric--Tree* (*M-Tree*). Com base nesta avaliação, foi proposto um mecanismo de indexação para armazenamento de trajectórias que agiliza a pesquisa dos *k* mais semelhantes num espaço métrico de trajectórias, denominado *SimTraj*.

Palavras-Chave: Sistemas de Informação Geográfica (SIG), Eventos Geográficos, Indexação de Objectos em Movimento, Pesquisa por Semelhança, Funções de Similaridade

Abstract

It is rapidly becoming difficult to find someone who does not own Global Positioning System (*GPS*) device in their car or pocket, since the components needed for its proper functioning can now be purchased at an inviting price.

Due to the existence of a vast volume of geo-referenced data, some of which related to movements of people and/or environmental/social events, such as hurricanes, migration, trade, transportation, it became necessary to discover and refine processes that streamline the search process in this kind of data effectively and efficiently, in order to predict/analyze possible catastrophes, as well as help in making decisions regarding strategies.

In this work, an evaluation of the existing techniques in the similarity search of geographic events, such as trajectories, was realized. For such, an exhaustive study of existing techniques and indexing methods that are involved in this type of research, in particular the most important similarity functions used in this research area and an evaluation of the similarity search in different metric spaces, using two metric data structures, *Recursive Lists of Clusters 2* (*RLC2*) and *Metric-Tree* (*M-Tree*). Based on the evaluation performed, an indexing method for the trajectory storage that streamlines the *k* most similar in a metric space of trajectories, named *SimTraj*, was proposed.

Keywords: Geographic Information Systems (GIS), Geographic Events, Indexing of Moving Objects, Similarity Search, Similarity Functions

Índice

Lista de figuras

Lista de tabelas

1. Introdução

O tema central desta dissertação diz respeito às técnicas usadas na pesquisa por semelhança em eventos geográficos, nomeadamente em trajectórias.

1.1. Motivação

O avanço tecnológico em sistemas de Geo-Posicionamento por Satélite (*GPS*), levou a um aumento no volume de dados georreferenciados disponíveis nos Sistemas de Informação Geográfica (*SIG*) e, por conseguinte, a uma maior exigência nas técnicas utilizadas na sua manipulação de modo a garantir um melhor desempenho nas suas funcionalidades.

Com esta tecnologia é possível manter grandes volumes de dados referentes a colecções geográficas (localizações) recolhidas ao longo do tempo (trajectórias). Estes dados são essenciais na análise de situações de emergência, diagnóstico e planeamento, ao se ter como base o estudo de padrões semelhantes. São exemplos disso aplicações na área do ambiente, desporto, urbanismo, comércio ou mesmo militar.

No ambiente, o estudo/análise de movimentos de animais (migrações) e/ou eventos com origem na natureza, tais como furacões, é essencial para prever catástrofes. Nestas aplicações existe a necessidade de pesquisar trajectórias semelhantes de modo a se poder analisar/prevenir situações futuras (Barbosa e Rodrigues 2009).

Abordando o desporto, estas técnicas são extremamente úteis para quem faz a recolha de dados, sejam estes de equipas adversárias ou da sua. Caso sejam de equipas adversárias,

tal análise poderá ter como objectivo a descoberta das trajectórias mais frequentes, em campo, dos jogadores tidos como mais perigosos durante as partidas até então realizadas, de forma a engendrar um estratégia para os parar. Se for da sua própria equipa, então uma finalidade poderá ser a correcção de erros posicionais dos jogadores, pesquisando as suas trajectórias habituais e mostrando-lhe quais as que ele deveria realizar ou mesmo para substituições. A abordagem deste tema poderá ser observada em (Yanagisawa, Akahani, e Satoh 2003).

No urbanismo, as análises de padrões revelam-se uma enorme ajuda no que diz respeito a análise de tráfego e transporte. Ao analisar trajectórias provenientes de vários *GPS*, dos condutores que circulam pelos diversos pontos da cidade, é possível prever quais as estradas mais utilizadas ou mesmo os congestionamentos que sucedem ao longo do dia, podendo assim ser tomadas medidas preventivas que visam evitar ou mesmo resolver situações complicadas que surgem no tráfego no dia a dia. Na criação da rede de transportes públicos, graças a uma análise cuidada dos percursos existentes e face às necessidades desejadas, serão escolhidos quais os caminhos a percorrer, de forma a, não serem sobrepostas as rotas. Ainda relativamente a este assunto, podemos destacar o auxílio prestado na realização de horários pois, ao comparar as trajectórias realizadas pelos transportes na rede poder-se-ão suprimir carreiras num sítio e colocá-las num outro para melhor servir a população. Esta será, por ventura, a motivação mais referida pelos autores em inúmeros trabalhos de investigação, tais como (Pelekis et al. 2009; Ding, Trajcevski, e Scheuermann 2008; Giannotti et al. 2009; Trajcevski et al. 2007; Tiesyte e Jensen 2008).

Na área militar e comercial, no que a este tema diz respeito, tudo é idêntico. Enquanto que na primeira interessam os padrões provenientes das rotas inimigas, de forma a evitá-las ou atacá-las, no segundo privilegia-se a semelhança dos movimentos das pessoas, por exemplo a forma como estas percorrem os corredores das superfícies comerciais. Assim, os produtos que se pretendam escoar de forma mais rápida, ou que se dirijam a um

determinado público alvo, poderão ser dispostos em pontos estratégicos. Uma abordagem à área comercial é realizada em (Keogh e Pazzani 1999).

Face às exigências destas aplicações é necessário estudar e analisar técnicas que agilizem o processo de reconhecimento de padrões semelhantes em grandes colecções de trajectórias.

1.2. Enquadramento e Contexto

Esta dissertação está inserida num projecto de investigação realizado na equipa Geographic Information and Simulation Systems (*GISS*) do Centro de Informática e Tecnologias da Informação (*CITI*), que pretende estudar mecanismos de representação das características espaço-temporais de eventos geográficos de modo a tornar mais eficaz e eficiente o processo de pesquisas que permitam uma melhor análise dos dados georreferenciados. Um dos tópicos desta investigação é o estudo e avaliação das técnicas usadas em pesquisa por semelhança sobre as trajectórias, o qual é foco desta dissertação.

Para poder medir a semelhança entre trajectórias é necessário definir uma função de semelhança (denominada de distância). Quanto menor for o valor da distância, maior será a semelhança existente entre estas. Quando esta função é métrica, tem-se um espaço métrico de trajectórias

De entre todas as possíveis pesquisas por similaridade, duas assumem um papel de maior preponderância. Desta forma ter-se-à, obrigatoriamente, de se destacar as pesquisas por alcance (*range query*), e as pesquisas dos *k* vizinhos mais próximos (*k-nearest neighbor query*).

A pesquisa por alcance, tem como objectivo encontrar todos os objectos existentes cuja distância em relação ao objecto da consulta não excede o raio de pesquisa, tendo este sido previamente fornecido.

Já a pesquisa dos *k* vizinhos mais próximos pretende encontrar os *k* objectos que sejam mais similares ao objecto dado, ou seja, a sua distância em relação ao objecto da consulta seja menor.

De modo a tornar estas pesquisas mais eficientes, existem diferentes mecanismos de indexação baseados na localização ou na distância (Samet 2006).

1.3. Objectivo

Nesta dissertação pretende-se estudar a aplicabilidade e eficiência de um conjunto de técnicas utilizadas na pesquisa por semelhança em dados georreferenciados (trajectórias), de modo a propor um mecanismo de indexação para trajectórias que agilize a pesquisa por semelhança.

Este estudo envolve uma análise detalhada de dois tópicos de interesse nesta área, as funções de semelhança e os mecanismos de indexação baseados na localização e na distância.

1.4. Contribuições previstas

Esta dissertação pretende contribuir para tornar mais eficiente o processo de pesquisa de trajectórias semelhantes e, se possível, propor um mecanismo de indexação específico para aplicações na área de Sistemas de Informação Geográficos (*SIG*) que manipulem trajectórias. Para além disso, é previsto que seja um contributo na área dos *SIG*, nomeadamente em dados espacio-temporais, uma vez que será feito:

- Um estudo das medidas de semelhança existentes em aplicações *SIG* que permitem comparar trajectórias;
- Um estudo dos mecanismos de indexação baseadas em localização e em distância que podem ser usadas em trajectórias;

- Uma avaliação de mecanismos de indexação baseados em distância em espaços métricos de trajectórias.

1.5. Metodologia

Com o objectivo de propor um mecanismo de indexação para trajectórias que agilize a pesquisa por semelhança, nesta dissertação são elaborados dois estudos, que dizem respeito à aplicabilidade das técnicas existentes e à eficiência dos mecanismos de indexação.

De forma a estudar a aplicabilidade das técnicas em aplicações de semelhança de trajectórias é necessário realizar:

- Estudo e avaliação das funções de semelhança utilizadas em trajectórias;
- Estudo das principais estruturas de dados baseadas na localização que são específicas para armazenamento de trajectórias;
- Estudo das principais estruturas de dados baseadas em distância que já foram utilizadas para o armazenamento de trajectórias.

Na fase de preparação desta dissertação, foram realizados todos estes estudos associados às funções de semelhança e mecanismos de indexação. O único aspecto que não foi totalmente tratado nesta dissertação diz respeito à avaliação das funções de semelhança.

De forma a avaliar a eficiência dos mecanismos de indexação, é necessário realizar um conjunto de pesquisas por semelhança, com diferentes funções de semelhança em diferentes conjuntos de dados.

Nesta dissertação, só foi possível avaliar os mecanismos de indexação baseados em distância em espaços métricos de trajectórias. Esta avaliação envolveu dois conjuntos de dados, um que diz respeito a um conjunto de trajectórias de autocarros e outro a trajectórias de furacões, previamente utilizados em trabalhos de investigação relacionados

(Barbosa e Rodrigues 2009; Gao et al. 2009; Frentzos, Gratsias, Pelekis, et al. 2007). Utilizando estes dados foram definidos quatro espaços de dados métricos. As pesquisas dos *k* vizinhos mais próximos (kNN) foram efectuadas sobre estes dados, utilizando duas estruturas de dados métricas, *Recursive Lists of Clusters 2* (*RLC2*) e *Metric-Tree* (*M-Tree*). A avaliação de eficiência das pesquisas é realizada com base no número de cálculos de distância realizados e no seu tempo de execução.

Com base nestes dois estudos foi proposto um mecanismo de indexação para trajectórias que agilize a pesquisa dos *k* mais próximos em espaços métricos de trajectórias.

1.6. Organização do documento

Este documento encontra-se organizado em sete capítulos. Neste primeiro capítulo é feita a apresentação do tema deste trabalho, nomeadamente no que diz respeito à sua motivação, enquadramento, objectivos e as principais contribuições previstas.

No capítulo dois será apresentado o trabalho relacionado, nomeadamente no que diz respeito às medidas de semelhança e métodos de indexação utilizados em trabalhos de investigação relacionados.

No terceiro capítulo, será apresentada a pesquisa por semelhança em espaços métricos de trajectórias que foi o foco central deste trabalho. Para além de uma descrição geral do tópico, apresenta-se, os espaços métricos e mecanismos de indexação baseados em distância utilizadas neste trabalho.

O capítulo quatro será preenchido com a avaliação realizada nas estruturas de dados baseadas em distância no que diz respeito à pesquisa dos *k* vizinhos mais próximos em espaços métricos de trajectórias.

No capítulo cinco será feita a definição do novo método de indexação para trajectórias *SimTraj*, o qual agiliza as pesquisas dos k mais próximos em espaços métricos de trajectórias.

Finalmente, no capítulo seis, apresentam-se as conclusões extraídas da elaboração deste trabalho. Uma apreciação crítica ao trabalho desenvolvido e algumas sugestões que podem ser tidas em conta no trabalho a realizar no futuro.

2. Trabalho relacionado

Em *SIG* existem muitas aplicações onde os elementos são descritos como uma sequência de localizações feitas ao longo do tempo (trajectórias) (Pelekis et al. 2009; Vlachos, Kollios, e Gunopulos 2002; Trajcevski et al. 2007; Laurinen, Siirtola, e Roning 2006; Ding et al. 2008; Frentzos, Gratsias, e Theodoridis 2007; Barbosa e Rodrigues 2009; Pelekis et al. 2007; Giannotti et al. 2009; 2007; Panagiotakis, Pelekis, e Kopanakis 2009; Monreale et al. 2009; Tiesyte e Jensen 2008). Face ao avanço tecnológico do *GPS*, estas aplicações lidam com um grande volume de dados logo, analisar os elementos destes sistemas tem-se tornado uma tarefa difícil face à dimensão dos problemas.

Nestas aplicações, o estudo de padrões semelhantes é crucial para a realização tanto de diagnósticos como de afinação de estratégias. Como exemplo podem-se referir os seguintes trabalhos: (Keogh e Pazzani 1999; L. Chen, Ozsu, e Oria 2005; Tiesyte e Jensen 2008; Giannotti et al. 2009; 2007; Panagiotakis et al. 2009; Monreale et al. 2009).

No que respeita à pesquisa de trajectórias semelhantes pode-se identificar dois aspectos fundamentais. O primeiro tem que ver com a forma de medir o grau de semelhança entre as trajectórias. Já o segundo, refere-se à pesquisa por semelhança propriamente dita, ou seja, como é que esta pode ser realizada em grandes volumes de dados.

A maioria dos trabalhos existentes nesta área dizem respeito às funções usadas para medir o grau de semelhança entre trajectórias (Keogh e Pazzani 1999; Pelekis et al. 2009; Vlachos et al. 2002; Trajcevski et al. 2007; Laurinen et al. 2006; Ding et al. 2008; Frentzos, Gratsias, e Theodoridis 2007; Quannan Li et al. 2008; Wang et al. 2008; L. Chen et al. 2005; Pelekis et al. 2007; Tiesyte e Jensen 2008; Giannotti et al. 2009; 2007;

Panagiotakis et al. 2009; Monreale et al. 2009) e propostas de métodos de indexação para pesquisa (Chakka, Everspaugh, e Patel 2003; Pfoser, Jensen, e Theodoridis 2000; Song e Roussopoulos 2003; Mokbel, Ghanem, e Aref 2003; Cudre-Mauroux, Wu, e Madden 2010; Barbosa e Rodrigues 2009; Samet 2006). No entanto, poucos integram estes dois aspectos (Barbosa e Rodrigues 2009).

De seguida são apresentadas duas secções onde serão apresentados os trabalhos relacionados, no que diz respeito às funções de semelhança e aos métodos de indexação.

2.1. Funções de semelhança

Muitos dos trabalhos que se referem a métodos de avaliação de semelhança de trajectórias (função de semelhança) propõem novas funções e alguns fazem uma avaliação com respeito às existentes. Nesta área de investigação, é importante salientar que cada função de similaridade considera diferentes características da trajectória, que poderão passar somente pelo suporte a alterações temporais (Pelekis et al. 2007; Laurinen et al. 2006; Trajcevski et al. 2007), espaciais (Pelekis et al. 2007) ou serem mais robustas e considerar características espacio-temporais (Pelekis et al. 2007; Frentzos, Gratsias, e Theodoridis 2007; Ding et al. 2008; Vlachos et al. 2002).

Para além destas restrições, ainda serão tomadas decisões sobre a incidência, ou seja, será preferível considerar a totalidade da trajectória do objecto em questão (Pelekis et al. 2007; Frentzos, Gratsias, e Theodoridis 2007; Pelekis et al. 2009; Quannan Li et al. 2008; Ding et al. 2008; Vlachos et al. 2002; L. Chen et al. 2005) ou somente porções desta (Pelekis et al. 2009; Laurinen et al. 2006; Trajcevski et al. 2007; Keogh e Pazzani 1999; Panagiotakis et al. 2009).

Uma outra vertente, completamente distinta desta, é ter em conta, somente, as regiões de maior afluxo (Giannotti et al. 2009; 2007; Tiesyte e Jensen 2008; Monreale et al. 2009), ignorando a trajectória em si.

As funções de semelhança mais usadas em trajectórias são:

- Dissim;
- L_p-norm
  - L_1-norm (Distância de Manhattan);
  - L_2-norm (Distância Euclidiana);
  - L_∞-norm (Norma Máxima);
- Dynamic Time Warping (*DTW*);
- Longest Common SubSequences (*LCSS*);
- Edit Distance (*ED*)
  - Edit Distance on Real Sequences (*EDR*);
  - Edit Distance with Real Penalty (*ERP*).

A eficácia de uma função de semelhança é medida com base nos resultados desta função quando são realizadas pesquisas por semelhança, nomeadamente com os falsos positivos (incorrectamente classificados) e verdadeiros positivos (correctamente classificados). De notar que a avaliação destas funções de semelhança é uma tarefa complicada face às características espacio-temporais envolvidas e depende, como é óbvio, da aplicação em si.

Trabalhos existentes (Frentzos, Gratsias, e Theodoridis 2007; Tiesyte e Jensen 2008; Wang et al. 2008) apresentam uma avaliação destas funções, onde se pode concluir que as funções *LCSS*, *EDR*, e *ERP* são muito semelhantes, em termos de eficácia, e que a *LCSS* supera as L_p- *-norm* e que a função *Dissim* supera a *LCSS* e a *EDR (Frentzos, Gratsias, e Theodoridis 2007)*.

De notar que a maior parte dos trabalhos existentes nesta área são mais de carácter teórico do que prático, isto é, poucos têm uma aplicação real, onde seja realizada a pesquisa de trajectórias semelhantes e por conseguinte a escolha da melhor função para uma dada aplicação é uma tarefa difícil e pouco analisada nos trabalhos existentes.

Uma apresentação mais detalhada das funções e da sua avaliação encontra-se no anexo A.1 .

2.2. Métodos de indexação

A representação dos dados multi-dimensionais é um aspecto essencial em SIG. Quando os dados são localizações no espaço, é tradicional utilizar estruturas de dados como a *Quad-Tree*, *k-d-Tree* e *R-Tree* ou variantes destas (Google 2010; Krafft, Harris, e Bougerel 2009; Greenheck 2009; Theodoridis 2005). Muitos são os trabalhos que tratam de propor estas variantes com o intuito de agregar a dimensão tempo a estes dados tendo, como objectivo, pesquisas referentes ao passado, presente e futuro, ou seja, métodos de indexação espacio-temporais. Para um *overview* deste métodos consultar (Mokbel et al. 2003), sendo que, em todos, os dados considerados são uma localização num dado momento.

No entanto, os dados utilizados neste trabalho de investigação são trajectórias, as quais são sequências de localizações, onde cada uma ocorre num dado momento (tempo). Uma trajectória encontra-se representada por uma sequência de triplos $<d_i, lat_i, lon_i>$, onde d_i é a data, lat_i a latitude e lon_i a longitude. Na Ilustração 2.1, retirada de (Pfoser et al. 2000), pode-se ver um conjunto de trajectórias no espaço de três dimensões, onde t é a data e x e y são as coordenadas associadas às respectivas latitudes e longitudes.

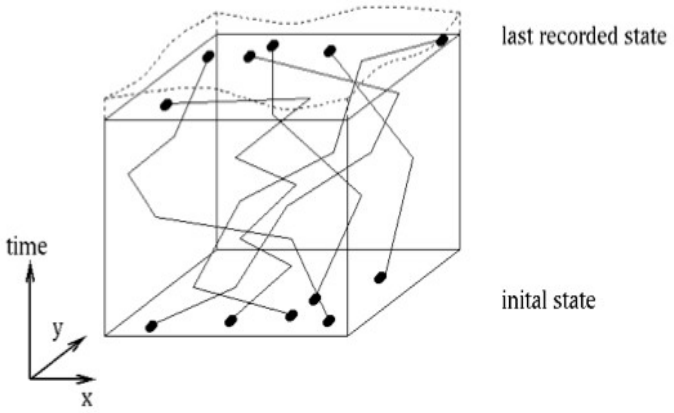

Ilustração 2.1: Trajectórias de objectos em movimento num ambiente espacio-temporal

Estes dados espacio-temporais têm sido, em muitos trabalhos de investigação, guardados em *R-Tree* (Ding et al. 2008; Trajcevski et al. 2007; L. Chen et al. 2005; Tiesyte e Jensen 2008). De referir, que uma dada trajectória está contida em diferentes regiões, pelo que na estrutura de dados aparece em partições distintas. Assim sendo, qualquer processo de pesquisa referente a estas terá uma dificuldade acrescida, visto não existir a noção de "trajectória", mas sim de localizações num dado momento ou segmentos de trajectórias. Nos trabalhos de investigação que usam estes mecanismos, a ênfase é no actual momento do objecto e não no seu histórico de posições passadas (Pfoser et al. 2000). De modo a resolver este problema, têm surgido propostas de alguns métodos de indexação específicos para trajectórias como *Trajectory-Bundle Tree* (*TB-Tree*) (Pfoser et al. 2000), *Spatial-Temporal Tree* (*STR-Tree*) (Pfoser et al. 2000), *Start/End time stamp B-tree* (*SEB-Tree*) (Song e Roussopoulos 2003), *Scalable and Efficient Trajectory Index* (*SETI*) (Chakka et al. 2003) e *TrajStore* (Cudre-Mauroux et al. 2010).

Analisando os resultados provenientes dos trabalhos existentes (Song e Roussopoulos 2003; Chakka et al. 2003; Cudre-Mauroux et al. 2010; Pfoser et al. 2000) pode-se concluir que a *R-Tree* é superada por todas as outras estruturas de dados e que, o *SETI* supera a *TB-Tree*, que por sua vez supera a *STR-Tree*. Com a *TrajStore* não é realizada nenhuma comparação com as outras estruturas de dados. Uma análise mais detalhada

destas estruturas de dados baseadas em localização específicas para trajectórias pode ser vista no anexo A.2 .

No entanto, em todos estes trabalhos, a pesquisa por semelhança é realizada tendo como base uma região e/ou um intervalo de tempo e a função de semelhança utilizada é a função L_2-*norm*, a qual mede a proximidade em termos de localização no espaço.

Com base nesta observação, é evidente que existe pouca integração entre os trabalhos de investigação referentes às funções de semelhança e aos métodos de indexação para pesquisa.

Em (Barbosa e Rodrigues 2009), pela primeira vez surge um trabalho que utiliza a função de semelhança *ERP* para a pesquisa de trajectórias semelhantes. Neste trabalho é utilizado um mecanismo de indexação baseado em distância, *Recursive Lists of Clusters* (*RLC2*) (Mamede 2005), para o armazenamento e pesquisa de trajectórias semelhantes. No entanto, não é realizada nenhuma avaliação desse método com respeito a outras estruturas de dados.

Existem muitos outros mecanismos de indexação baseados em distância (Samet 2006), tais como: *Geometric Near-Neighbor Access Tree* (*GNAT)*; *Vp-Tree*: *Metric-Tree* (*M-Tree*); *Sa-Tree*; *Linear Approximating Eliminating Search Algorithm* (*LAESA)*; e *Slim--Tree*. Neste tipo de estruturas de dados, os elementos da base de dados são particionados tendo em conta a distância entre um conjunto de elementos seleccionados e os restantes. Pelo facto desta distância ser métrica, no momento das pesquisas, muitas partições são descartadas utilizando as propriedades da função distância, de modo a agilizar o processo de pesquisa.

2.3. Conclusões e contributos

Na área de pesquisa de trajectórias semelhantes é evidente que existe, ainda, muito trabalho a realizar, a começar pela identificação de qual a melhor medida de semelhança a utilizar para certas aplicações, passando à integração destas com métodos de indexação, culminando na avaliação desses métodos, principalmente nos baseados em distância e em localização.

Neste trabalho, tentou-se minimizar as lacunas acima indicadas, ao se integrar as funções de semelhança, nomeadamente a L_2-*norm* e a *ERP* (secção 3.1.), com métodos de indexação baseados em distância (secção 3.2.), do qual resultou uma proposta de um mecanismo de indexação para pesquisa de trajectórias semelhantes num espaço métrico (secção 5.). Esta proposta teve como base a avaliação realizada sobre pesquisas dos *k* vizinhos mais próximos em espaços métricos de trajectórias (secção 3.).

3. Pesquisa de semelhança em espaços métricos de trajectórias

Um espaço métrico é caracterizado por um par (*U*,*d*), onde *U* é um conjunto de objectos (denominado universo), e $d : U \times U \Rightarrow \mathbb{R}_0^+$ é uma função distância métrica. Uma função é métrica se verificar as seguintes condições:

- Positividade: $\forall_{x,y \in U} \; d(x,y) \geqslant 0$;
- Reflexividade: $\forall_{x \in U} \; d(x,x) = 0$;
- Simetria: $\forall_{x,y \in U} \; d(x,y) = d(y,x)$;
- Desigualdade triangular: $\forall_{x,y \in U} \; d(x,y) \leqslant d(x,z) + d(z,y)$.

Uma base de dados definida sobre um espaço métrico (*U*,*d*) é um subconjunto finito *B* do universo *U* $(B \subseteq U)$.

Alguns trabalhos de investigação, (Yianilos 1993; Bring 1995; Chávez et al. 2001; G. Navarro 2002; Micó, Oncina, e Vidal 1994), propõem mecanismos de indexação que agilizam as pesquisas por semelhança, sendo algumas delas específicas para espaços métricos.

Dependendo da dimensão do espaço métrico, a pesquisa por semelhança pode ser, ou não, uma tarefa mais "difícil". Na verdade, a dificuldade da pesquisa cresce com a dimensão do espaço (Chávez et al. 2001).

O cálculo da dimensão de um espaço métrico que utiliza dados reais continua a ser um problema em aberto. Em (Chávez et al. 2001), a dimensionalidade intrínseca de um espaço métrico encontra-se definida pela função (1), onde μ e σ^2 são respectivamente, a média e a variância do histograma das distâncias entre os objectos da base de dados.

$$\mu^2/(2\times\sigma^2) \qquad (1)$$

As pesquisas por semelhança podem ser de dois tipos: por alcance, ou pesquisa dos *k* mais semelhantes (kNN). Numa pesquisa por alcance, pretende-se obter todos os objectos que se encontram a uma distância de um dado objecto (objecto de pesquisa), que seja menor ou igual a um dado valor (raio de pesquisa). Por outro lado, numa pesquisa dos *k* mais semelhantes, apenas se pretende encontrar os *k* objectos mais semelhantes a um dado objecto.

Neste trabalho, decidiu-se dar primazia às pesquisas do tipo kNN em detrimento das pesquisas por alcance. A escolha deste tipo de pesquisa deve-se, principalmente, por ser mais relevante neste domínio de aplicação. Para além disso, como poderá ser visto mais adiante (secção 5.4.2.), esta pesquisa baseia-se na pesquisa por alcance, logo pode-se dizer que a avaliação realizada envolve ambas as pesquisas.

Formalmente, a pesquisa dos kNN pode ser definida da seguinte forma:

- Dada uma base de dados B definida sobre um espaço métrico (U,d);
- Um objecto de pesquisa q, $q \in U$;
- Um valor *k*, $k \in \mathbb{R}^+$.

Iremos obter como resposta para a pesquisa dos kNN $(q)_k$, um sub-conjunto de objectos da base de dados *B* (*X*), tal que $X = \{x \in B \,\forall_u \in B - X, d(x,q) \leqslant d(u,q)\}$ e a cardinalidade de X é igual a *k*.

De seguida, serão apresentados os espaços métricos de trajectórias e os mecanismos de indexação utilizados na avaliação realizada nesta dissertação.

3.1. Espaços métricos

Para medir a semelhança entre trajectórias foram escolhidas as funções *L_2-norm* e *Edit Distance with Real Penalty* (*ERP*). Esta escolha foi realizada no conjunto de todas as

funções de semelhança métricas apresentadas em A.1 . A escolha da *L_2-norm* baseou-se em ser a mais usada nos trabalhos existentes, nomeadamente nas estruturas de dados específicas para trajectórias. Já a escolha da *ERP* teve por base os resultados obtidos em (Tiesyte e Jensen 2008). De notar, que todas as funções apresentadas em A.1 foram implementadas e integradas nos protótipos elaborados para a avaliação de funções.

Nesta dissertação, foram utilizados dois conjuntos de dados, nomeadamente:

- As trajectórias das deslocações efectuados por autocarros escolares, a quando da tomada e largada de passageiros, na cidade de Atenas e arredores da sua área metropolitana[1]. Este conjunto de dados é constituído por 148 trajectórias, tendo a maior trajectória 1018 localizações, a menor 16 e a média 449;
- As trajectórias de furacões[2]. Este conjunto de dados possui, igualmente, 148 trajectórias. Aqui, a maior trajectória possuí 619 localizações, a menor 2 e a média 85.

Estes conjuntos de dados já tinham sido usados em outros trabalhos de investigação (Barbosa e Rodrigues 2009; Gao et al. 2009; Frentzos, Gratsias, Pelekis, et al. 2007).

Passa-se agora a descrever as funções de semelhança utilizadas e a caracterização dos espaços métricos utilizados, no que diz respeito à sua dimensão.

3.1.1. Funções de semelhança

As funções *L_2-norm* e *Edit Distance with Real Penalty* (*ERP*) são funções métricas. A primeira é uma variante da *Lp-norm* com $p = 2$. A *ERP* é uma variante da função de edição que poderá ser vista como sendo uma evolução da combinatória da *Longest Common SubSequences (LCSS)* com a *Dynamic Time Warping* (*DTW)*, sendo que difere

1 http://www.rtreeportal.org/datasets/trajectories/buses.zip
2 http://weather.unisys.com/hurricane/atlantic/

da primeira ao não existir o parâmetro de tolerância ε, e da segunda ao não proceder à replicação dos elementos anteriores.

Sendo *R* e *S* duas trajectórias de tamanho respectivamente *M* e *N*, os elementos r_i e s_i definem o i-ésimo elemento da respectiva trajectória, cujos valores dizem respeito às coordenadas espaciais. Considere- -se ainda r_i-s_i como a distância Euclidiana entre dois pontos e *Rest(R)* a trajectória resultante de retirar o primeiro elemento da trajectória *R*. Tem-se que as fórmulas (2) e (3) definem, respectivamente, a L_2-*norm* e a *ERP*.

$$L_2-norm(R,S)=\sqrt{\sum_{i=1}^{N}(r_i-s_i)^2} \qquad (2)$$

$$ERP(R,S)=\sum_{i=1}^{M}|r_i-g| \quad se \quad N=0;$$

$$ERP(R,S)=\sum_{i=1}^{N}|s_i-g| \quad se \quad M=0;$$

$$ERP(R,S)=\min\{ERP(Rest(R),Rest(S))+dist_{erp}(r_1,s_1),$$
$$ERP(Rest(R),S)+dist_{erp}(r_1,g),$$
$$ERP(R,Rest(S))+dist_{erp}(s_1,g)\} \quad caso\ contrátio$$

onde (3)

$$dist_{erp}(r_i,s_i)=|r_i-s_i| \quad se \quad não \quad considerada \quad falha;$$

$$dist_{erp}(r_i,s_i)=|r_i-g| \quad se \quad s_i \quad considerado \quad como \quad falha;$$

$$dist_{erp}(r_i,s_i)=|s_i-g| \quad se \quad r_i \quad considerado \quad como \quad falha.$$

e sendo *g* um ponto constante associada à falha.

Como foi dito anteriormente, a escolha destas funções de semelhança teve por base dois critérios. O primeiro diz respeito a estas serem métricas, pois ao se escolher métodos de

indexação baseados em distância, estas têm de, obrigatoriamente, ser métricas. A segunda tem a ver com os resultados provenientes dos trabalhos de investigação, onde se pode destacar:

(1) De todas as variantes da L_p-*norm*, a L_2-*norm* é a mais utilizada em trabalhos de investigação, seja sozinha (Trajcevski et al. 2007; Laurinen et al. 2006) ou como auxiliar de outras normas, tais como a *DTW* (Ding et al. 2008);

(2) A eficácia da *ERP* é muitíssimo semelhante à da *LCSS*, *DTW* e *ED*, como podemos observar pela avaliação realizada em (Wang et al. 2008).

3.1.2. Caracterização dos espaços métricos

De modo a poder avaliar o grau de dificuldade das pesquisas nos espaços métricos utilizados, foram elaborados os histogramas de distâncias nos diferentes espaços métricos, com o objectivo de os poder caracterizar, tendo como base a fórmula (1). Nas tabelas 3.1 e 3.2, são apresentados os valores correspondentes a essa análise nos dois conjuntos de dados utilizados, respectivamente.

Função de Semelhança	μ	σ^2	$\mu^2/(2\times\sigma^2)$
ERP	1526202480	1227843647621422336	0.95
L_2-norm	233022.39	293250148941.05	0.09

Tabela 3.1: Dimensão dos espaços métricos dos autocarros

Função de Semelhança	μ	σ^2	$\mu^2/(2\times\sigma^2)$
ERP	401.63	193657.08	0.42
L_2-norm	26.77	1136.81	0.32

Tabela 3.2: Dimensão dos espaços métricos dos furacões

Analisando os resultados da Tabela 3.1, referentes aos dados dos autocarros, pode-se afirmar que os espaços métricos definidos possuem diferentes dimensões, sendo a mais

elevada obtida através da função *ERP*, onde o quociente é de 0.95, e a menos pela L_2-*-norm*.

Referente aos dados dos furacões, presentes na Tabela 3.2, ambos os espaços métricos possuem dimensões muito semelhantes, ainda que o de maior dimensão seja aquele onde se utiliza a função *ERP*, onde o quociente é de 0.42.

Na Tabela 3.3 apresentam-se os espaços métricos utilizados, ordenados por ordem crescente da dimensão.

Espaço métrico	*Dimensão*
Autocarros + L_2-*norm*	0.09
Furacões + L_2-*norm*	0.32
Furacões + *ERP*	0.42
Autocarros + *ERP*	0.95

Tabela 3.3: Espaço de dados métricos ordenados por dimensão

De notar, que os espaços métricos definidos com base na L_2-*norm* têm uma dimensão menor dos definidos com a função *ERP*. Sendo o espaço métrico dos autocarros com a função *ERP* o de maior dimensão, destacando-se com uma dimensão de 0.95.

3.2. Métodos de indexação baseados em distância

Os mecanismos baseado em distância, também denominados de estruturas de dados métricas, nesta dissertação, necessitam que a função de semelhança (distância) seja métrica e particionam os elementos da base de dados, com base nas distâncias entre um conjunto de elementos seleccionados e os restantes. Estes mecanismos de indexação podem ser classificados em baseados em *pivots* ou em agrupamentos. Para tal, tem-se em conta a forma como os dados são armazenados de acordo com a distância entre:

- O novo objecto e o centro do agrupamento (*cluster based*);

- O novo objecto e os objectos *pivots* (*pivot based*).

Estes mecanismos de indexação, ao particionarem os elementos no espaço de acordo com a função de distância permitem, no momento da pesquisa, descartar varias partições do espaço, agilizando este tipo de de pesquisas. As partições são descartadas com base nas propriedades da função métrica, nomeadamente, a desigualdade triangular.

Pode-se, ainda, classificar estas estruturas de dados de acordo com o seu dinamismo, podendo ser estáticas ou dinâmicas. As estruturas de dados métricas estáticas são aquelas que, depois da sua construção, não admitem inserções nem remoções de objectos, sem que seja necessária uma reconstrução completa. As estruturas de dados métricas dinâmicas permitem a inserção e/ou a remoção de objectos *a posteriori* da sua construção.

Das estruturas de dados métricas existentes, é de notar que a *RLC* já foi utilizada na representação de informação geográfica (Barbosa e Rodrigues 2009), tal como a *M-Tree* (Tiakas et al. 2009; G.-P. Roh et al. 2010). Por tal motivo, foram escolhidas, na nossa avaliação, a *M-Tree* e uma variante da *RLC* (*RLC2*) (Sarmento 2010).

De seguida, será feita uma descrição destas estruturas de dados métricas, as quais são ambas dinâmicas.

3.2.1. Recursive Lists of Clusters 2 (*RLC2*)

A *RLC2* (Sarmento 2010), é baseada na *RLC* descrita em (Mamede 2005), e particiona os elementos em agrupamentos com base na distância entre um elemento e os centros dos agrupamentos, sendo por isso uma estrutura baseada em agrupamentos.

A *RLC2* é uma sequência de agrupamentos. Nesta sequência, os agrupamentos são disjuntos dois a dois, e um dado elemento pertence ao primeiro agrupamento que o pode conter, como será visto mais a diante. Cada agrupamento da *RLC2* esta caracterizado por:

- Centro (*c*): um objecto da base de dados;
- Raio (*r*): um valor real positivo que define a maior distância possível entre os elementos no agrupamento e o centro;
- Nível (*n*): um valor inteiro que indica o nível do agrupamento na estrutura de dados;
- Interior (*I*): constituído por objectos cuja distância ao centro é menor ou igual ao raio do agrupamento.

Dependendo do número de elementos no interior de um dado agrupamento de nível *i*, este pode ser uma folha ou uma lista de agrupamentos de nível superior (*i*+1). Para tal, a *RLC2* tem um parâmetro, denominado de capacidade da folha, que vai definir quando um interior é ou não folha.

Para além deste parâmetro, a *RLC2* tem um outro associado ao raio dos agrupamentos de nível zero, o qual é denominado de raio da *RLC2*.

Na Ilustração 3.1, baseada em (Mamede 2005), pode-se observar uma *RLC2* com raio ρ e com capacidade da folha 5. Na figura, *c* representa o centro, *r* o raio, *l* o nível, *s* o tamanho do interior e *I* o próprio interior do agrupamento.

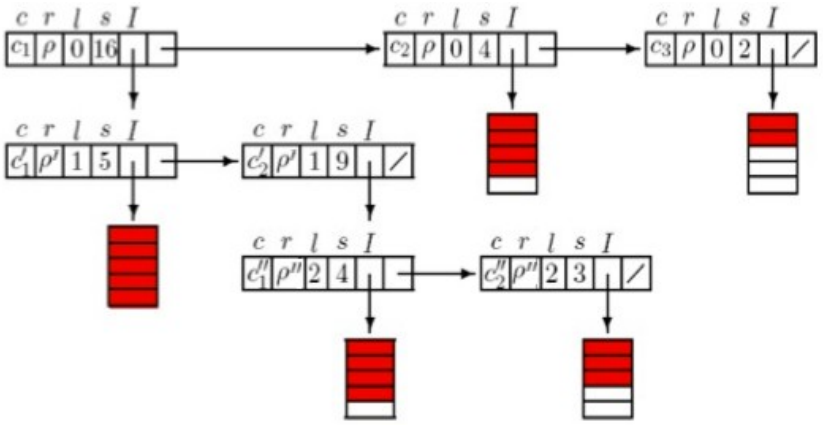

Ilustração 3.1: Exemplo da RLC2 com capacidade cinco

Como se observa na Ilustração 3.1, os raios dos agrupamentos são variáveis, dependendo do seu nível. O cálculo do raio para um dado agrupamento é feito com base na formula (4), sendo *n* o nível do agrupamento e *r* o raio da *RLC2*.

$$r = \frac{r'}{(n-1)} \qquad (4)$$

Na versão original da *RLC* (Mamede 2005), o raio dos agrupamentos é constante. No entanto, a variante da *RLC* com raio variável (*RLC2*) demonstrou ter melhor desempenho, pois permite uma organização mais eficaz dos elementos. Na Ilustração 3.2, retirada de (Sarmento 2010), é fácil de constatar uma melhor distribuição dos elementos pelos agrupamentos contidos num dado agrupamento da *RLC2* (de cor cinzenta na ilustração). Nesta ilustração a parte (a) representa a distribuição de uma *RLC* de raio fixo e (b) a distribuição com raio variável (*RLC2*).

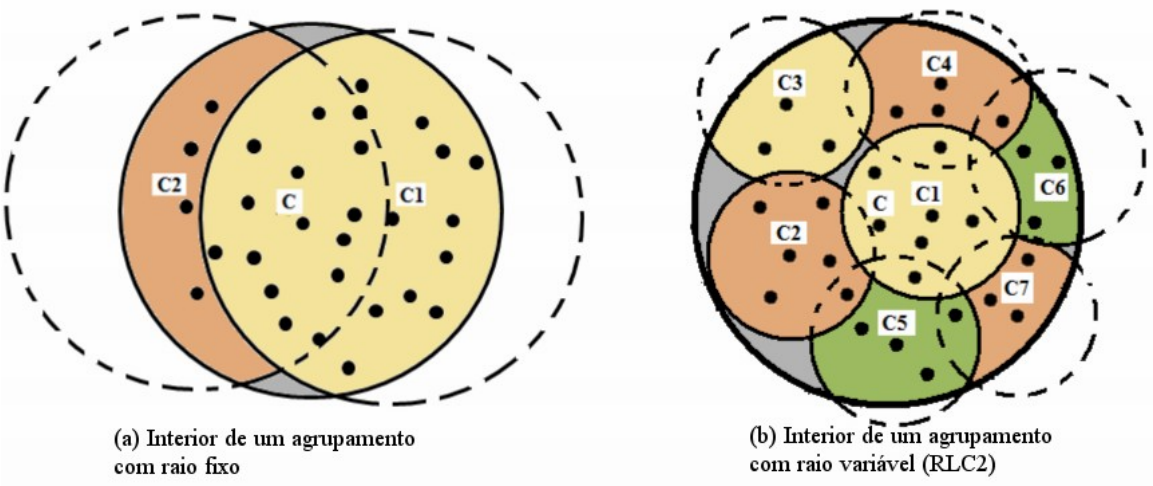

Ilustração 3.2: Exemplo de distribuição de elementos num agrupamento

Na *RLC2*, um elemento pertence a um agrupamento (c, r, n, I) se for satisfeita a restrição, baseada na distância d, $d(c, e) \leq r$. Cada elemento na *RLC2* pode pertencer a vários agrupamentos em diferentes níveis. Logo, por eficiência nas operações, cada elemento numa folha tem associada uma lista onde se encontram alocadas as distâncias a cada centro dos agrupamentos a que este pertence. Esta lista está ordenada de forma descendente, ou seja, do agrupamento com maior nível para o de nível zero.

A organização dos elementos em agrupamentos resulta da forma como estes são inseridos na estrutura de dados. Para se inserir um novo elemento na *RLC2* ter-se-à de percorrer a lista de agrupamentos de nível zero e proceder-se à verificação da restrição da distância, ou seja, a distância entre o elemento e o centro do agrupamento é calculada para se verificar se esta é inferior ou igual ao raio do agrupamento. Caso não seja encontrado nenhum agrupamento que cumpra a condição, é criado, no fim da lista, um novo agrupamento, sendo o novo elemento o seu centro. Caso contrário, o elemento será inserido no primeiro agrupamento encontrado. A inserção do elemento no agrupamento poderá cair num de dois casos:

(1) O interior deste é uma folha;

(2) O interior é uma lista de agrupamentos.

Em (1), é verificada a existência de espaço para o novo elemento, o qual não pode exceder a capacidade da folha. Em caso afirmativo, procede-se à sua inserção ordenada na folha, tendo como base a sua distância ao centro do agrupamento. Caso contrário, o interior do agrupamento sofrerá uma reorganização, ou seja, uma lista de agrupamentos será criada e serão inseridos nesta lista o novo elemento e todos os elementos que estavam no interior. Em (2), a inserção do elemento é aplicada recursivamente nessa lista de agrupamentos até que se crie um novo agrupamento ou se insira o elemento num nó folha.

A *RLC2* usada nesta dissertação foi gentilmente cedida pelo colega Ângelo Sarmento e está implementada em *C++*, e em memória secundária. Como a pesquisa dos *k* mais próximos não estava definida, foi necessário implementá-la. A implementação realizada baseia-se na pesquisa por alcance da *RLC2* (explicada na secção 5.4.2.).

Para criar uma *RLC2* é necessário parametrizar a capacidade da folha, para além de todos dos outros (tamanho da folha e raio) a cima referidos.

A estrutura de dados *RLC*, a qual serviu de base à *RLC2*, já foi utilizada em diferentes domínios de aplicação, tais como imagens de rosto (Silva 2009; Chambel e Barbosa 2009), música (F. Costa e Barbosa 2009; F. M. R. Costa 2009) e dicionários (Barbosa e Mamede 2007), demonstrado ser competitiva com outras estruturas de dados métricas, tendo inclusivamente superado muitas das testadas.

3.2.2. Metric-Tree (*M-Tree*)

A estrutura de dados *Metric-Tree* (*M-Tree*) (GBDI 2009; Ciaccia, Patella, e Zezula 1997; Samet 2006) é uma estrutura de dados métrica que integra os conceitos de *pivots* e centros. Esta estrutura de dados apresenta enumeras semelhanças com as árvores B+ (Comer 1979), pois, guarda todos os objectos nas suas folhas e estas encontram-se todas

ao mesmo nível. Os nós intermédios desta árvore, denominados de nós de encaminhamento, são caracterizados por:

- Pivot (*p*): um objecto da base de dados;
- Raio (*r*): um valor real que define a maior distância possível entre os elementos na sub-árvore e o *pivot*;
- Apontador para uma sub-árvore (*T*): a sub-árvore que contém objectos cuja distância ao *pivot* (*p*) é menor ou igual ao raio (*r*);
- Distância entre nós de encaminhamento (*D*): distância entre o *pivot* (*p*) e o *pivot* do nó de encaminhamento onde este está contido.

De modo a organizar os elementos na *M-Tree*, a inserção realiza-se de forma descendente e recursiva com o intuito de descobrir a folha mais adequada para inserir o novo elemento. Para localizar essa folha descer-se-á cada nível da árvore através dos nós de encaminhamento, escolhendo um que não necessite que o seu raio seja aumentado para poder alocar o elemento. Nesta escolha, pode acontecer que exista mais do que um nó ou não se encontre nenhum. No primeiro caso, será escolhido o nó cujo *pivot* esteja mais próximo do elemento. No segundo caso, o novo objecto será inserido no nó cujo raio sofra o menor aumento. Ao inserir-se um elemento na folha ideal, um de dois casos irá ocorrer:

(1) A folha não necessita de ser dividida;
(2) A folha necessita de ser dividida.

Em (1) o elemento é alocado nessa mesma folha e a inserção encontra-se concluída. Em (2) a resolução não é tão simples, pois este obriga ao particionamento do nó de encaminhamento associado à folha, e, por conseguinte, à aplicação de um método de promoção de modo a equilibrar a árvore. Este particionamento, consiste da divisão de uma folha em duas e na distribuição dos objectos que se encontravam na folha juntamente com o novo. Uma vez feita esta distribuição é obrigatório ter dois objectos, um de cada nova folha criada, que irão desempenhar a função de *pivot* nos novos nós de encaminhamento. Estes novos nós terão de ser organizados na árvore, ou seja, será

necessário escolher os nós de encaminhamento onde irão ser alocados. Este processo poderá induzir um particionamento de nós intermédios, de forma a garantir as características da árvore. Note-se que, caso a raiz da árvore seja particionada, a árvore verá a sua profundidade aumentada.

Na Ilustração 3.3, retirada de (Zezula et al. 2006), podemos ver uma *M-Tree*. Em (a) é possível observar as partições de objectos no espaço e em (b) a organização desses mesmos objectos numa *M-Tree*.

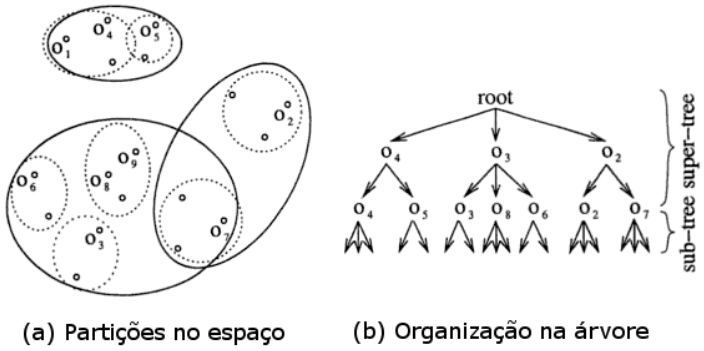

Ilustração 3.3: Organização dos elementos numa M-Tree

O código, implementado em *C++*, da *M-Tree* utilizado encontra-se em (GBDI 2009). Para parametrizar a *M-Tree*, é necessário definir os valores para o tamanho da folha e o método a utilizar no particionamento.

De notar que o código da *M-Tree* foi utilizado sem ter sido realizada, por nós, qualquer alteração ao mesmo.

3.3. Conclusões e contributos

Para uma avaliação da pesquisa de trajectórias semelhantes em espaços métricos, é necessário estudar e avaliar as diferentes técnicas no que diz respeito à métrica, mecanismos de indexação e grau de dificuldade da pesquisa.

Nesta secção foram apresentadas as funções de semelhança e as estruturas de dados métricas utilizadas na nossa avaliação (secção 4.). Para além disso foram caracterizados os espaços métricos utilizados, de modo a se poder realizar uma melhor análise, tendo por base a dificuldade da pesquisa nos espaços métricos usados.

4. Avaliação experimental

A avaliação realizada nesta dissertação diz respeito à eficiência da pesquisa dos k mais próximos (com $k = 1$ e $k = 5$) em espaços métricos de trajectórias. Esta avaliação envolveu dois mecanismos de indexação baseados em distância (*RLC2* e *M-Tree*), os quais foram usados em quatro espaços métricos. Como dito anteriormente, estes espaços métricos estão definidos sobre dois conjuntos de dados (autocarros e furacões), utilizando duas funções de semelhança (L_2-*norm* e *ERP*).

De modo a poder avaliar estes mecanismos de indexação, em cada pesquisa realizada foram calculados os seguintes valores:

- SD – Distâncias calculadas durante a pesquisa;
- SR – Leituras efectuadas em disco durante a pesquisa;
- ET – Tempo de execução da pesquisa (em segundos).

Para cada espaço métrico, o número de pesquisas realizadas de um dado tipo ($k = 1$ ou $k = 5$) foi de vinte e cinco por cento do tamanho da base de dados. Dado que ambos os conjuntos de dados são constituídos por cento e quarenta e oito elementos, o número de pesquisas realizadas em cada um é de trinta e sete. A escolha do conjunto de trajectórias utilizados nas pesquisas foi feito de forma aleatória.

Como ambas as estruturas de dados são parametrizáveis, foi necessário refinar os valores da parametrização de modo a obter uma avaliação justa nestes mecanismos, ou seja, foram escolhidos os valores que tornam mais eficientes as pesquisas dos kNN realizadas.

Para parametrizar a *RLC2* foram realizados diferentes testes (anexo A.3) e foi escolhida, para cada espaço métrico, a seguinte parametrização do raio da *RLC2*:

- Autocarros + *ERP*: raio 25% da distância média do espaço;
- Autocarros + L_2-*norm*: raio 25% da distância média do espaço;
- Furacões + *ERP*: raio 75% da distância média do espaço;
- Furacões + L_2-*norm*: raio 50% da distância média do espaço.

Em todos os espaços métricos, a capacidade da folha da *RLC2* foi de 75 e o tamanho da página de 4096.

Para se parametrizar a *M-Tree*, recorreu-se aos resultados descritos em (Ciaccia et al. 1997), e foi escolhido o método *Minimum Maximum Radius Policy* (*MIN_RAD*) para particionamento, tendo como base a estratégia *Generalize Hyperplane Partition Strategy* (*G_HYPERPL*), e 5020 para tamanho da página.

Como a organização dos dados, nestas duas estruturas de dados, é dependente da ordem em que os elementos são inseridos, foi necessário gerar aleatoriamente três permutações de ambos os conjuntos de dados, para tornar este estudo mais preciso.

Após serem escolhidas as parametrizações das estruturas de dados, geradas as três permutações e os sub-conjuntos de trajectórias que servem para pesquisa, estava-se em condições de se proceder à realização, para cada espaço métrico, das pesquisas em seis "instâncias" de estruturas de dados (três da *RLC2* e três da *M-Tree*). Os valores presentes nas tabelas deste capítulo resultam das médias dos resultados obtidos nas três "instâncias" para cada estrutura de dados.

De seguida serão apresentados os valores das pesquisas 1NN e 5NN nos espaços métricos de trajectórias. Por último, será realizada uma análise global da avaliação efectuada.

4.1. Pesquisa dos *k* vizinhos mais próximos (kNN)

Foram realizadas as pesquisas para os *k* vizinho mais próximos *(k* = 1 e *k* = 5*)* em ambas as estruturas de dados, utilizando os 4 espaços métricos escolhidos.

Os resultados apresentados, ao longo desta secção, espelharam a média aritmética dos valores obtidos através das três permutações, bem como o valor máximo. É de notar que os valores dizem respeito a cada pesquisa realizada.

4.1.1. Recursive Lists of Clusters 2 (*RLC2*)

Nas tabelas 4.1, 4.2, 4.3 e 4.4 estão os valores obtidos nas pesquisas (*k* = 1 e *k* = 5) realizadas na *RLC2* nos espaços métricos dos autocarros e furacões.

Funções de Semelhança	*SD*		*SR*		*ET*	
	Média	Max	Média	Max	Média	Max
ERP	47.67	63	4.97	13.67	180.53	437.33
L_2-norm	113.15	123.33	14	15	1.72	2.67

Tabela 4.1: 1NN na RLC2 (autocarros)

Funções de Semelhança	*SD*		*SR*		*ET*	
	Média	Max	Média	Max	Média	Max
ERP	40.22	92.67	7.77	11.33	8.52	37
L_2-norm	45.38	98	6.29	9	0.25	1

Tabela 4.2: 1NN na RLC2 (furacões)

Funções de Semelhança	*SD*		*SR*		*ET*	
	Média	Max	Média	Max	Média	Max
ERP	15.80	37.67	6.17	8	186.54	428.67
L_2-norm	110.80	120.67	14.72	18	1.87	3

Tabela 4.3: 5NN na RLC2 (autocarros)

Funções de Semelhança	*SD*		*SR*		*ET*	
	Média	Max	Média	Max	Média	Max
ERP	40.22	92.67	8.70	11.67	10.51	41
L_2-norm	27.92	82.33	4.64	5.33	0.23	0.67

Tabela 4.4: 5NN na RLC2 (furacões)

Tendo por base a média do número de distâncias calculadas na pesquisa, é de salientar que para todos os espaços métricos existe uma redução muito significativa face a uma pesquisa exaustiva. Nas tabelas 4.5 e 4.6 é possível observar a percentagem dos elementos da base de dados comparados em média e no pior caso, para cada pesquisa (k = 1 e k = 5) nos quatro espaços métricos, respectivamente.

Espaço métrico	*Percentagem dos elementos comparados*	
	Média	*Máximo*
Autocarros + *L_2-norm*	76.5%	83.3%
Autocarros + *ERP*	32.2%	42.6%
Furacões + *L_2-norm*	30.7%	66.2%
Furacões + *ERP*	27.2%	62.6%

Tabela 4.5: Percentagem de elementos comparados nas pesquisas 1NN

Espaço métrico	*Percentagem dos elementos comparados*	
	Média	*Máximo*
Autocarros + L_2-*norm*	74.9%	81.5%
Autocarros + *ERP*	10.7%	25.5%
Furacões + L_2-*norm*	18.9%	55.6%
Furacões + *ERP*	27.2%	62.7%

Tabela 4.6: Percentagem dos elementos comparados nas pesquisas 5NN

Tendo por base estes valores, pode-se concluir que o número médio de elementos, comparados em todos os casos, nunca é de 100%, logo, nunca acontece uma pesquisa exaustiva. Nesta avaliação, o pior caso é de 83.3% e de 81.5% de elementos comparados, para $k = 1$ e $k = 5$, respectivamente, ambos no espaço métrico dos autocarros com a função L_2-*norm*.

Relembrando as dimensões dos espaços métricos (Tabela 3.3), podemos concluir que a *RLC2* tem um bom desempenho em espaços métricos de maior dimensão, com a *ERP*, no qual temos entre 18.9% e 30.7% de elementos comparados, ou seja, descartamos em média 69.3% a 82.1% elementos na pesquisa. Se observarmos os valores máximos de comparações, temos o pior resultado a ser obtido no espaço métrico de menor dimensão (Autocarros + L_2-*norm*).

Tendo por base o número médio de leituras realizadas em cada pesquisa, pode-se observar que:

- O valor médio varia entre 4.97 e 14, sendo o valor máximo, outra vez, obtido no espaço métrico de menor dimensão (Autocarros + L_2-*norm*), na pesquisa 1NN;
- O valor médio varia entre 4.64 e 14.72, sendo novamente no espaço métrico de menor dimensão onde se obtém o máximo valor, na pesquisa 5NN.

Se observarmos o tempo médio de execução de cada pesquisa, é fácil verificar que, em todos os espaços métricos, excepto os autocarros com *ERP*, este valor é muito "bom" e varia entre:

- 0.25 e 8.52 segundos, na pesquisa 1NN;
- 0.23 e 10.51 segundos, na pesquisa 5NN.

No entanto, no espaço métrico dos autocarros com *ERP* este valor é de 180.53 segundos (cerca de 3.0 minutos), para $k = 1$ e de 186.54 segundos (3.1 minutos), para $k = 5$. Estes valores são tempos de resposta muito deficientes, podendo chegar a um valor máximo de 437.33 segundos (cerca de 7.3 minutos), em $k = 1$ e 428.67 segundos (cerca de 7.1 minutos), em $k = 5$. Numa primeira análise, poderia-se dizer que este valor era pouco esperado, dado que a redução no número médio de elementos comparados na pesquisa foi grande (30.7%, para $k = 1$ e 18.9%, para $k = 5$), no entanto esta função de semelhança (*ERP*) é mais complexa e implica maior tempo de execução. Este facto leva, também, a um tempo de execução médio de 8.52 segundos, para $k = 1$ e 10.51, para $k = 5$ no espaço métrico de furacões definido com a função *ERP*, o segundo maior valor de tempo de execução.

É de salientar um facto muito peculiar. Enquanto que os tempos de execução de 1NN são menores que os de 5NN para o conjunto de dados dos autocarros, o oposto sucede no conjunto de dados dos furacões. Após uma análise a este resultado, verificou-se que nas pesquisas dos k mais semelhantes realizadas nestes dados, a pesquisa por alcance efectuada era a mesma para $k = 1$ e $k = 5$. Isto é, este facto ocorre devido à forma como os kNN são implementados (baseia-se na pesquisa por alcance) e à organização dos dados na estrutura[3]. No entanto, quer nos parecer que este facto foi uma excepção e não uma regra para as pesquisas kNN.

3 Quando este facto foi detectado não foi possível, por limitação de tempo, proceder à geração de mais permutações com o intuito de realizar novamente mais pesquisas.

No global, podemos concluir que a *RLC2* teve um desempenho muito bom em qualquer um dos espaços métricos definidos nos dados referentes a furacões, os quais, como sabemos, têm dimensão muito similar (Tabela 3.3). Nos espaços métricos definidos nos dados dos autocarros, é evidente que os resultados já não são consistentes, já que o tempo de execução é bastante pior quando se usa a função *ERP* e o número de cálculos de distância é muito pior com a *L_2-norm*. Isto ocorre devido às características temporais dos dados, os quais são mais dispersos nos autocarros que nos furacões (tempos mais uniformes).

4.1.2. Metric-Tree (M-Tree)

Nas tabelas 4.7, 4.8, 4.9 e 4.10 estão os valores obtidos nas pesquisas (k = 1 e k = 5) realizadas na *M-Tree* nos espaços métricos dos autocarros e furacões. Aqui não foram apresentados os valores mínimo e máximo de SD e SR por serem iguais ao valor médio.

Funções de Semelhança	*SD*	*SR*	*ET*	
	Média	Média	Média	Max
ERP	148	1	209.49	405.67
L_2-norm	148	1	1.93	3.33

Tabela 4.7: 1NN na M-Tree (autocarros)

Funções de Semelhança	*SD*	*SR*	*ET*	
	Média	Média	Média	Max
ERP	148	1	10.63	36
L_2-norm	148	1	0.24	1

Tabela 4.8: 1NN na M-Tree (furacões)

Funções de Semelhança	*SD*	*SR*	*ET*	
	Média	Média	Média	Max
ERP	148	1	210.10	410
L_2-norm	148	1	1.98	3.33

Tabela 4.9: 5NN na M-Tree (autocarros)

Funções de Semelhança	*SD*	*SR*	*ET*	
	Média	Média	Média	Max
ERP	148	1	10.53	36
L_2-norm	148	1	0.28	1

Tabela 4.10: 5NN na M-Tree (furacões)

Analisados os resultados obtidos, é de notar que não existe redução de cálculos de distância, face à pesquisa exaustiva. No entanto, é feita sempre uma leitura a disco para cada pesquisa. O tempo médio de execução de uma pesquisa varia entre:

- 0.24 e 209.49 segundos, com $k = 1$;
- 0.28 e 210.10 segundos, com $k = 5$.

O pior tempo de execução foi de 405.67 segundos (cerca de 6.8 minutos) em $k = 1$ e 410 segundos (cerca de 6.8 minutos) em $k = 5$ no espaço métrico dos autocarros com a *ERP*.

Tendo como base a dimensão do espaço métrico, a *M-Tree* não mostra nenhuma alteração do seu desempenho, isto é, são percorridos sempre todos os seus elementos (pesquisa exaustiva).

Quando confrontados os autores desta estrutura de dados com estes resultados, foi-nos indicado que, em espaços de dados pouco esparsos, a *M-Tree* não tem tido bons resultados.

Os tempos de execução são, ligeiramente, maiores na pesquisa dos 5NN do que a de 1NN para todos os espaços métricos, excepto o dos furações com a *ERP*. Pensa-se que este facto atípico poderá estar relacionado com a forma como os dados estão organizados na estrutura de dados, ou seja, uma vez mais com as permutações utilizadas.

4.2. Análise dos resultados

Nas ilustrações 4.1, 4.2, 4.3 e 4.4 encontram-se os valores médios das distâncias calculadas (SD), das leituras em disco (SR) e do tempo de execução (ET) para cada conjunto de dados, autocarros e furacões, utilizando as funções de distância, *ERP* e *L_2-norm*, para as estrutura de dados, *RLC2* e *M-Tree*.

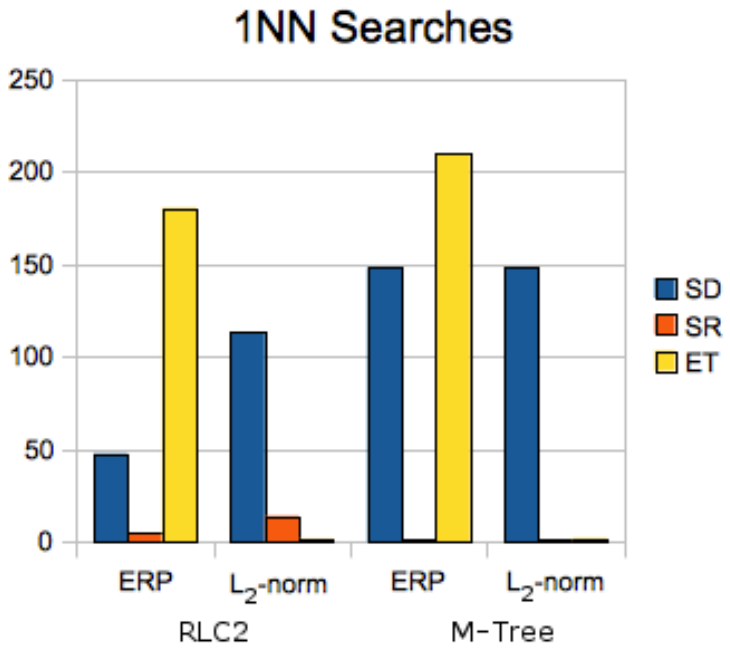

Ilustração 4.1: 1NN (autocarros)

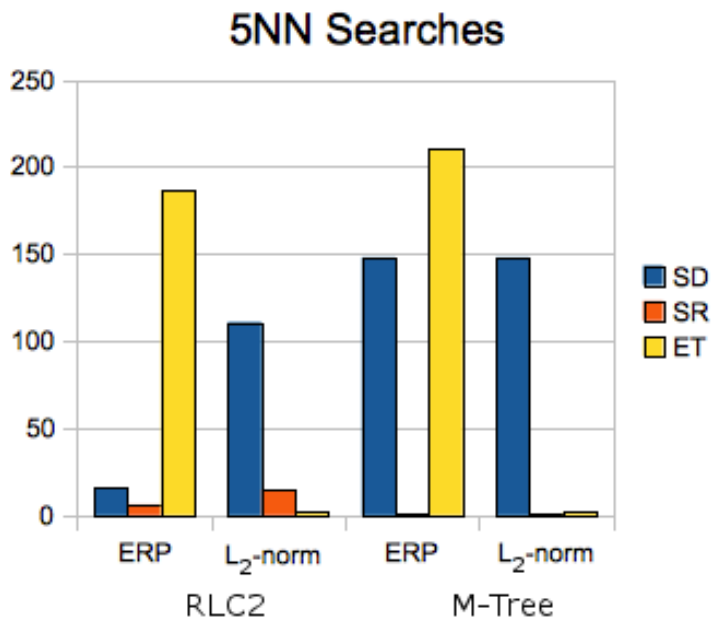

Ilustração 4.2: 5NN (autocarros)

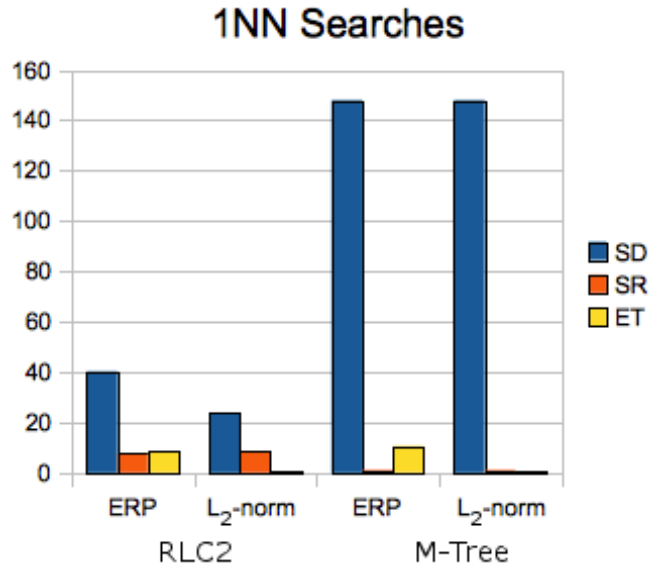

Ilustração 4.3: 1NN (furacões)

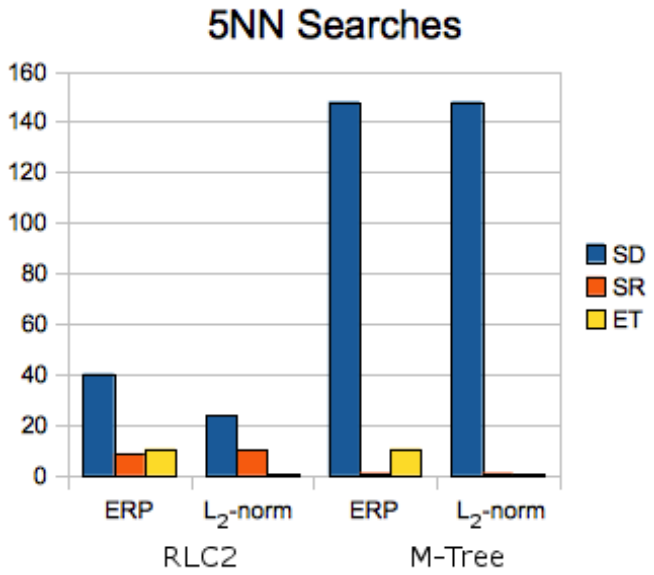

Ilustração 4.4: 5NN (furacões)

Analisando os resultados obtidos, pode-se concluir que a *RLC2* e a *M-Tree* são bastantes competitivas, no que se refere ao tempo de execução das pesquisas. No entanto, a *RLC2* apresenta sempre melhores resultados, ainda que pouco significativos.

A *RLC2* destaca-se na *M-Tree* no número médio de trajectórias comparadas em cada pesquisa, descartando sempre um número bastante elevado de trajectórias para comparação. A *M-Tree* neste aspecto tem um desempenho muito "mau", pois é comparada a uma pesquisa exaustiva na base de dados.

Tendo por base o número de acesso a disco, a *M-Tree* é imbatível e, seria de esperar que o tempo de resposta a uma dada pesquisa fosse excelente. No entanto, isto não acontece. Este facto deve-se ao número de elementos comparados em cada pesquisa, o qual aumenta significativamente o processamento nas pesquisas, já que é necessário calcular muitas distâncias e este processamento, normalmente, é complexo.

4.3. Conclusões e contributos

No global, pode-se concluir que ambas as estruturas de dados (*RLC2* e *M-Tree*) são bastante competitivas. Enquanto que a *RLC2* calcula um menor número de distâncias durante as pesquisas, a *M-Tree* realiza menos leituras em disco. A nível de desempenho no tempo de resposta ao utilizador (tempo de execução), a *RLC2* consegue sempre obter melhores resultados que a *M-Tree*, apesar das diferenças serem pouco significativas.

Constata-se que a *RLC2* tem um melhor desempenho em espaços métricos de maior dimensão, o qual já tinha acontecido em outros trabalhos de investigação (Barbosa e Mamede 2007; Barbosa 2009).

Dado o tipo de aplicações em pesquisas de trajectórias semelhantes, gostaríamos de enfatizar que, por norma, o conjunto de dados, é relativamente, pouco esparso. Logo, face ao desempenho da *M-Tree*, parece evidente que esta não consegue minimizar em grande escala o número de elementos comparados por pesquisa, pelo menos nos dados utilizados (furacões e autocarros). Assim sendo, a *RLC2* parece ser a escolha correcta neste tipo de dados, em que a organização dos dados na estrutura possibilita diminuir, em muito, o número de elementos comparados.

Como principais contribuições pode-se destacar a análise comparativa entre as duas estruturas de dados (*RLC2* e *M-Tree*) em quatro espaços métricos (Autocarros + *ERP*; Autocarros + L_2-*norm*; Furacões + *ERP*; Furacões + L_2-*norm*).

5. SimTraj

Face à necessidade de um mecanismo para o armazenamento de trajectórias que agilize a pesquisa por semelhança e face aos resultados obtidos na nossa avaliação (secção 4.), é proposto um mecanismo de indexação específico para trajectórias, denominado de *SimTraj*. Este mecanismo agiliza as pesquisas de trajectórias semelhantes (SIMilar TRAJectories) em espaços métricos. Neste mecanismo, é necessário que a função de semelhança seja métrica.

SimTraj é uma estrutura de dados dinâmica e implementada em memória secundária, a qual permite operações de inserção, remoção, consulta e actualização de trajectórias. As pesquisas por semelhança permitidas são os *k* vizinhos mais próximos e por alcance.

Na Ilustração 5.1, pode-se observar a arquitectura da *SimTraj*. Este mecanismo de indexação é composto por duas componentes: base de dados de trajectórias (*frontline*) e partição baseada em distâncias (Distance Based Partition).

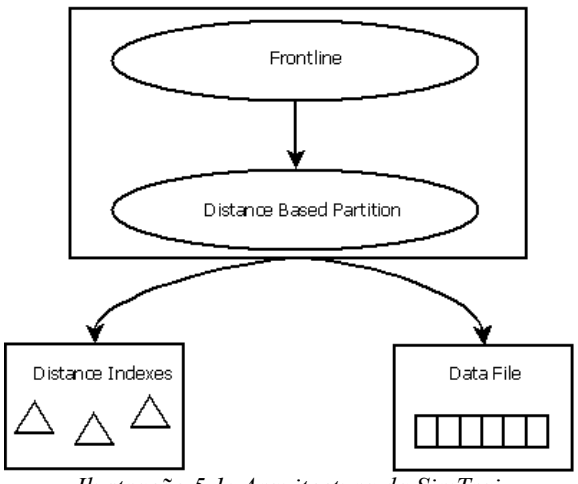

Ilustração 5.1: Arquitectura da SimTraj

A componente associada à partição tem, como principal objectivo, a agilização das operações de pesquisa de trajectórias semelhantes e foi implementada utilizando uma *RLC2*. Nesta componente, as trajectórias estão organizadas em agrupamentos, sendo possível, na execução das pesquisas, descartar muitos destes agrupamentos tendo como base as propriedades das funções de distância, nomeadamente a desigualdade triangular.

A componente associada propriamente à base de dados serve para agilizar as operações referentes a actualizações da base de dados, como são inserções, remoções e actualizações. Nesta primeira versão da *SimTraj*, esta componente está modelada como um dicionário. Este está implementado através de uma tabela de dispersão, sendo a chave o identificador da trajectória, e o elemento o par composto pela trajectória e uma ligação directa ao local onde se encontra o elemento na *RLC2* (apontador). Esta ligação directa à estrutura de dados *RLC2* serve para agilizar as actualizações e remoções, sendo efectivamente o apontador do agrupamento de nível superior ao qual o objecto pertence.

De seguida, serão apresentadas as operações principais da *SimTraj* e é realizada uma análise a este mecanismo de indexação no que diz respeito à sua extensibilidade.

Gostaríamos de salientar que, no anexo A.4 , estão presentes alguns valores calculados no momento da criação da *SimTraj*, aquando da realização da avaliação[4].

Nas secções seguintes associadas às operações, assume-se sempre que estamos a trabalhar numa *SimTraj*(*rlc*,*frontline*), onde *rlc* é a estrutura de dados *RLC2* e *frontline* é um dicionário. Assume-se ainda, que o quarteto (*c*,*r*,*I*,*n)* denota um agrupamento da *RLC2* e o par (*e*,*ap_rlc*) o elemento guardado na *frontline*.

4 Estes valores foram documentados nesta dissertação, pois poderão ser úteis para trabalhos futuros.

5.1. Inserção de uma trajectória

Para se realizar a inserção é necessário fornecer os dados da trajectória. A inserção será realizada em dois passos, como se pode observar na Ilustração 5.2. Primeiro, é inserido o objecto trajectória na *RLC2* (linha 3), após se verificar na *frontline* que o identificador ainda não existe. Depois essa trajectória é inserida, juntamente com o apontador para a *RLC2* (resultado da inserção na *RLC2*) na *frontline* (linha 4).

```
   Name: insetSimTraj
   Input: object – a trajectory object.
1. let pointer_rlc, initially empty;
2. if frontline.get(object.getID()) is null then
3.      pointer_rlc = rlc.insert(object);
4.      frontline.add(object.getID(),pair(object,pointer_rlc));
5. else
6.      raises_error("Trajectory Already Exists");
7. end if
```

Ilustração 5.2: Algoritmo de inserção da SimTraj

A inserção na *SimTraj* necessita do algoritmo de inserção da *RLC2* numa *SimTraj*, o qual invoca o algoritmo apresentado na Ilustração 5.3, sendo o segundo parâmetro a lista de agrupamentos de nível zero da *RLC2,* o terceiro o seu raio e o quarto o valor zero. Neste algoritmo temos quatro parâmetros: (1) o objecto a inserir; (2) uma lista de agrupamentos; (3) o raio dos agrupamentos na lista; e (4) o nível dos agrupamentos na lista.

```
    Name: insertRLC2
    Input: object – a RLC2 object;
         l – the list of clusters;
         r – the cluster radius;
         n – the cluster level.
    Output: a apointer to the place where the object is alocated.
 1. let pointer_rlc, initially empty;
 2. let double distance = 0;
 3. let double newRadius = 0;
 4. if l is empty then
 5.      pointer_rlc = newCluster(object,r,emptyList,n);
 6.      l.insert(pointer_rlc);
 7.      return pointer_rlc;
 8. else
 9.      let l = <(c,r',I,n')|l'>; // r is equal to r'
10.      distance = d(object,c);
11.      if distance ≤ r then
12.           if I is leaf and I.getSize() ≤ LEAFCAPACITY then
13.                I.insert(object); // leaf insertion
14.                return pointerTo((c,r',I,n'));
15.           else if I is leaf then
16.                     let l'' initially empty;
17.                     foreach e in I do
18.                          newRadius = getRadius(r',n');
19.                          pointer_rlc = insertRLC2(e,l'',newRadius,n'+1);
20.                          frontline.set(e.getID(),pair(e,pointer_rlc));
21.                     end for
22.                     pointer_rlc = insertRLC2(object,l'',newRadius,n'+1);
23.                     I = l'';
24.                     return pointer_rlc;
25.                else
26.                     newRadius = getRadius(r',n');
27.                     return insertRLC2(object,I,newRadius,n'+1);
28.                end if
29.           end if
30.      else
31.           return insertRLC2(object,l',r,n);
32.      end if
33. end if
```

Ilustração 5.3: Algoritmo de inserção da RLC2 na SimTraj

A inserção na *RLC2* numa *SimTraj* consiste em procurar o agrupamento no qual se vai inserir o novo objecto. Esse agrupamento (*c*,*r*,*I*,*n*) será o primeiro que se encontre na lista que satisfaça a condição $d(c, o) \leqslant r$, sendo *o* o objecto a inserir (linhas 10 e 11). A lista de agrupamentos da *RLC2* é iterada, e, uma vez conseguido o agrupamento, é realizada a inserção nesse. Se esse agrupamento for uma folha, onde o número de elementos em *I* é inferior à capacidade da folha (*LEAFCAPACITY*), o elemento será inserido nessa folha (linhas 13 e 14). Caso o agrupamento seja uma folha totalmente ocupada, é necessário colocar os elementos do seu interior, juntamente com o novo elemento, numa lista de agrupamentos (linhas 16 a 24). Neste caso, é de salientar que é necessário realizar uma actualização no *frontline* da *SimTraj* para todos os elementos que foram reorganizados na *RLC2* (linha 20). Por último, se o interior do agrupamento for uma lista de agrupamentos, procede-se à invocação recursiva deste algoritmo (linha 27). Caso não se consiga nenhum agrupamento, um novo será criado (linhas 5 a 7).

5.2. Remoção de uma trajectória

Para realizar a remoção de uma dada trajectória na *SimTraj* é necessário fornecer o identificador. O algoritmo de remoção na *SimTraj*, pode ser visto na Ilustração 5.4.

```
   Name: removeSimTraj
   Input: id – a RLC2 object id.
1. let boolean result = false;
2. pair(o,pointer_rlc) = frontline.getPointer(id);
3. if pair(o,pointer_rlc) is not null then
4.      rlc.remove(pair(o,pointer_rlc));
5.      frontline.remove(id);
6. else
7.      raises_error("Trajectory not found");
8. end if
```

Ilustração 5.4: Algoritmo de remoção da SimTraj

Na remoção, a primeira acção a realizar é consultar a *frontline* para se obter o par constituído pela trajectória e pelo apontador para a *RLC2*, onde o objecto a remover se

encontra (linha 2). Uma vez obtido esse apontador, invoca-se a remoção no agrupamento associado na *RLC2* numa *SimTraj* (linha 4) e retira-se o elemento da *frontline*.

O algoritmo de remoção da *RLC2* numa *SimTraj* está descrito na Ilustração 5.5.

```
    Name: removeRLC2
    Input: pair(object,(c,r,I,n)) – an frontline element.
 1. let pointer_rlc, initially empty;
 2. if object is not c then
 3.      I.remove(object); // leaf deletion
 4. else
 5.      let l = (c,r,I,n).getList();
 6.      l.remove((c,r,I,n));
 7.      foreach e in I do
 8.           pointer_rlc = insertRLC2(e,l,r,n);
 9.           frontline.set(e.getID(),pair(e,pointer_rlc));
10.      end for
11. end if
```

Ilustração 5.5: Algoritmo de remoção da RLC2 na SimTraj

Este algoritmo tem como parâmetro o par composto pelo objecto e o apontador ao agrupamento, que contém o objecto. Como este agrupamento é o de maior nível a que o elemento pertence, um de dois casos poderá ocorrer:

(1) O elemento a remover é o centro de um agrupamento;

(2) O elemento a remover está na folha (interior do agrupamento).

Em (1) todo o agrupamento é removido, sendo novamente inseridos, na lista de agrupamentos que continha este agrupamento, todos os objectos, com excepção do elemento que se pretende remover (linhas 5 a 10). Esta inserção leva a uma actualização dos apontadores dos objectos inseridos na *frontline* (linha 9). Em (2) o processo é menos complexo, já que aqui o objecto é, simplesmente, removido na folha (linha 3).

5.3. Actualização de uma localização na trajectória

As actualizações permitidas na *SimTraj* dizem respeito a uma nova localização, num dado momento, numa dada trajectória. Ao incorporar essa nova coordenada espacio-temporal na trajectória, a distância (semelhança) dessa trajectória às outras na base de dados pode mudar e, por conseguinte, a sua localização na estrutura de dados *RLC2* poderá mudar. Isto acontece porque a inserção de um dado elemento na *RLC2* é realizada com base na função de distância, como foi visto anteriormente.

Face a isto, uma actualização é vista como uma remoção seguida de uma inserção na *SimTraj*. O algoritmo de actualização da *SimTraj* está apresentado na Ilustração 5.7 e recebe, como entrada, o identificador e a nova localização espacio-temporal da trajectória a actualizar.

```
   Name: updateSimTraj
   Input: id – a RLC2 trajectory object id;
        (t,x,y) – spatial-temporal coordinate.
1. let newObject be a RLC2 trajectory object;
2. pair(object,pointer_rlc) = frontline.get(id);
3. if pair(object,pointer_rlc) is not null then
4.      newObject = object.set((t,x,y));
5.      removeSimTraj(id);
6.      insertSimTraj(newObject);
7. else
8.      raises_error("Trajectory not found");
9. end if
```

Ilustração 5.6: Algoritmo de actualização na SimTraj

5.4. Pesquisa de trajectórias semelhantes

As pesquisas por semelhança permitidas na *SimTraj* são duas: pesquisa por alcance e pesquisa dos *k* vizinhos mais próximos.

5.4.1. Pesquisa por alcance

Na pesquisa por alcance tem de se fornecer o objecto trajectória (*o*) bem como o raio (*r*) que delimita a pesquisa, ou seja, somente serão retornados objectos que se encontrem a uma distância de *o* menor ou igual a *r*.

Esta pesquisa na *SimTraj* é realizada usando a operação de pesquisa por alcance da *RLC2*. O seu algoritmo pode ser consultado em Ilustração 5.7.

```
    Name: rangeSimTraj
    Input: radius – the radius search;
         object – a trajectory object.
    Output: a set with all the objects which distance is lesser or equal to r.
1.  return rlc.searchRQ(object,radius);
```

Ilustração 5.7: Algoritmo das pesquisas por alcance da SimTraj

Uma pesquisa por alcance na *RLC2* consiste em se procurar todos os elementos, nesta estrutura de dados, que distam, do objecto dado, um valor inferior ou igual ao raio dado. Esta procura na *RLC2* é realizada invocando o algoritmo apresentado na Ilustração 5.8, sendo *l* a lista de agrupamentos de nível zero.

```
    Name: rangeRLC2
    Input: radius – the search radius;
         object – a RLC2 trajectory object;
         let l be a list of clusters;
    Output: a set with all the objects which distance is lesser or equal to r.
 1. let result be the result set, initially empty;
 2. let double distance = 0;
 3. if l is empty then
 4.      return result;
 5. else
 6.      let l = <(c,r,I,n)|l'>
 7.      distance = d(object,c);
 8.      if distance ≤ radius then // cases (a); (b); (c)
 9.           result.add(c);
10.           if distance + radius ≤ r then // case (a)
11.                if I is leaf then
12.                     return result.addAll(I.searchRQ(object,radius)); // Leaf Range Query
13.                else
14.                     return result.addAll(rangeRLC2(radius,object,I));
15.                end if
16.           else if distance + r ≤ radius then // case (b)
17.                     result.addAll(I);
18.                     return result.addAll(rangeRLC2(radius,object,l'));
19.                else // case (c)
20.                     if I is leaf then
21.                          result.addAll(I.searchRQ(object,radius));
22.                     else
23.                          result.addAll(rangeRLC2(radius,object,I));
24.                          return result.addAll(rangeRLC2(radius,object,l'));
25.                     end if
26.                end if
27.           end if
28.      else // cases (d); (e); (f)
29.           if distance + radius ≤ r then // case (d)
30.                if I is leaf then
31.                     return result.addAll(I.searchRQ(object,radius));
32.                else
33.                     return result.addAll(rangeRLC2(radius,object,I));
34.                end if
35.           else if distance > radius + r then // case (f)
36.                     return rangeRLC2(radius,object,l');
```

```
37.               else // case (e)
38.                    if I is leaf then
39.                         result.addAll(I.searchRQ(object,radius));
40.                    else
41.                         result.addAll(rangeRLC2(radius,object,I));
42.                         return result.addAll(rangeRLC2(radius,object,l'));
43.                    end if
44.               end if
45.          end if
46.     end if
47. end if
```

Ilustração 5.8: Algoritmo das pesquisas por alcance da RLC2

Como se pode ver no algoritmo, é necessário iterar por todos os agrupamentos e procurar aqueles onde possam existir elementos candidatos à resposta. Na pesquisa por alcance existe uma região que interessa, denominada de região de pesquisa, a qual pode ser vista como um agrupamento com centro no objecto pergunta e raio igual ao valor dado. No momento da pesquisa, quando queremos verificar se um dado agrupamento pode conter candidatos à resposta, temos que verificar se existe ou não alguma intersecção entre a região de pesquisa e o agrupamento. Um de seis possíveis casos (Ilustração 5.9) pode acontecer:

(a) A região de pesquisa contém o centro do agrupamento e está contida neste;
(b) A região de pesquisa contém estritamente o agrupamento;
(c) A região de pesquisa contém o centro do agrupamento e intersectá-lo sem o conter e sem estar contida neste;
(d) A região de pesquisa não contém o centro do agrupamento mas está contida neste;
(e) A região de pesquisa não contém o centro do agrupamento mas intersecta-o sem estar contida neste;
(f) A região de pesquisa é disjunta do agrupamento.

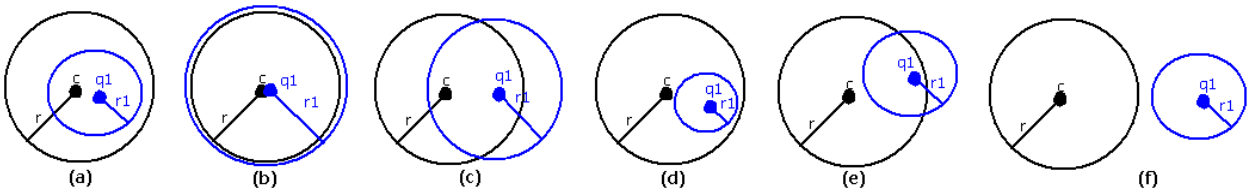

Ilustração 5.9: Exemplos de situações de pesquisa na RLC2

Identificada a intersecção entre o agrupamento da *RLC2* e a região de pesquisa, poderá ser necessário analisar o interior do agrupamento, de forma a se encontrar os elementos desejados na pesquisa por alcance. No caso (f), não existe nenhum elemento no agrupamento que seja relevante para a pesquisa e no caso (b), todos os elementos do agrupamento são imediatamente incluídos no resultado da pesquisa (linha 17). Para todos os outros casos é necessário prosseguir o processamento da pesquisa no agrupamento da *RLC2*. Caso o interior do agrupamento seja uma lista de agrupamentos, o algoritmo é invocado recursivamente com essa lista (linhas 14, 23, 33 e 41). No entanto, se o interior é uma folha, então a pesquisa é realizada na folha (linhas 12, 21, 21 e 39) e, para tal, existe um vector auxiliar, denominado *minDist*, que foi construido incrementalmente na pesquisa e contém as distâncias entre o objecto da pesquisa e os centros dos agrupamentos já analisados de nível inferior. Este vector é útil, pois pode permitir descartar ou incluir elementos no resultado sem cálculos adicionais de distância. Por legibilidade no algoritmo presente na Ilustração 5.8, este pormenor técnico não foi apresentado.

De notar que, para todos os casos, excepto (a) e (d), é necessário prosseguir a pesquisa pelos restantes agrupamentos da *RLC2* (linhas 18, 24, 36 e 42).

5.4.2. Pesquisa dos *k* mais próximos

A pesquisa dos kNN não se encontrava implementada na *RLC2*, assim sendo foi necessário realizar essa tarefa. Na implementação desta pesquisa usa-se a pesquisa por alcance, acima referida, e um iterador existente na *RLC2*, o qual itera todos os elementos desta estrutura de dados sem nenhuma ordem definida.

Na pesquisa dos kNN é necessário fornecer o objecto trajectória (*o*) e o número (*k*) de objectos similares que se pretende encontrar. O algoritmo da pesquisa dos kNN está apresentado na Ilustração 5.10.

```
    Name: knnSimTraj
    Input: k – the nubmber of neighbours;
         object – a RLC2 trajectory object.
    Output: a ordered set containing the k nearest neighbours.
 1. let element be a RLC2 trajectory object;
 2. let pQueue be a priority queue of pairs <distance,element>, initially empty;
 3. let resultRQ be a set with the results of a RLC2 range query;
 4. let rlc2Iterator be the RLC2 iterator;
 5. let double maxDistance = 0;
 6. let double distance = 0;
 7. while k > pQueue.size() then
 8.      element = rlc2Iterator.next();
 9.      distance = distanceTo(object,element);
10.      pQueue.push(<distance,element>);
11. end while
12. maxDistance = pQueue.top().first;
13. resultRQ = rangeSimTraj(maxDistance,object);
14. return resultRQ.sortByDistance();
```

Ilustração 5.10: Algoritmo dos k vizinhos mais próximos da SimTraj

Esta pesquisa envolve duas fases. Na primeira, utiliza-se o iterador da *RLC2* para se obter *k* trajectórias existentes na estrutura de dados, sendo estas ordenadas de forma descendente pela sua distância à trajectória da pesquisa *o* (linhas 7 a 11). Para se obter esta ordenação usou-se uma fila de prioridades. Numa segunda fase, efectua-se uma pesquisa por alcance na *SimTraj* (linha 13), sendo *o* o objecto da pesquisa e sendo o raio a maior distância encontrada entre as trajectórias dadas pelo iterador e o objecto da pesquisa (linha 12). O resultado desta pesquisa por alcance contém os *k* elementos procurados, os quais serão ordenados, de forma descendente tendo por base a distância a *o*. Deste modo são dadas as *k* trajectórias mais semelhantes ao objecto da pesquisa.

5.5. Extensibilidade da *SimTraj*

Ao longo do trabalho desenvolvido, foram ponderadas várias formas de proceder ao armazenamento das trajectórias na *SimTraj*. No entanto, devido ao tempo estipulado para apresentar esta dissertação foi impossível realizar mais actividades, mas parece-nos ser algo bastante robusto e de largo interesse para esta área de investigação, pelo que é apresentada nesta secção de modo a poder ajudar no trabalho futuro.

Existem dois aspectos relevantes que gostaríamos de mencionar e que dizem respeito à extensibilidade da *SimTraj*. O primeiro está associado à componente *frontline*, a qual, neste momento, está implementada como sendo uma tabela de dispersão mas que podia ser integrada com uma base de dados de objectos em movimento de uma qualquer aplicação *SIG*.

O segundo aspecto, tem a ver com a partição baseada em distâncias da *SimTraj*, a qual neste momento está implementada com a *RLC2* e armazena trajectórias completas. Existem muitas aplicações em que a pesquisa por semelhança não é realizada tendo em conta a trajectória completa mas sim parte dela. Se for possível, à partida saber o comprimento dos segmentos de interesse numa trajectória, então poder-se-à usar uma *RLC2*, onde se armazena todos os segmentos com esse comprimento de todas as trajectórias da base de dados. Neste caso, a *frontline*, para poder ter informação referente à trajectória completa, também necessita de alguns ajustes. Era necessário guardar na *frontline*, para cada trajectória, a trajectória em si e uma sequência de apontadores para a estrutura de dados *RLC2*, sendo cada um associado a cada segmento envolvido.

Estes dois aspectos permitem uma maios adaptabilidade da *SimTraj* às aplicações nesta área de investigação.

5.6. Conclusões e contributos

O mecanismo proposto, *SimTraj*, permite o armazenamento e a pesquisa de trajectórias semelhantes em espaços métricos e integra as técnicas de semelhança de trajectórias com os mecanismos de indexação baseados em distância. Face à nossa avaliação (secção 4.) podemos concluir que a *SimTraj* tem um bom desempenho nas pesquisas de trajectórias semelhantes.

Com a proposta deste mecanismo específico para trajectórias em espaços métricos, minimiza-se a lacuna existente nos trabalhos de investigação existentes, os quais não integravam as funções de semelhança com mecanismos de indexação. Para além disso, é de salientar como principal contribuição a proposta e implementação da *SimTraj*. Destaca-se, ainda, as alterações realizadas à *RLC2*, nomeadamente a implementação dos *k* vizinhos mais próximos.

6. Conclusões

Neste capítulo será realizada uma análise ao trabalho desenvolvido, nomeadamente no que diz respeito às conclusões finais, às limitações e às contribuições. Para além disso, serão propostas algumas direcções, que poderão ser seguidas, para trabalho futuro.

6.1. Apreciação crítica do trabalho desenvolvido

Neste trabalho foi realizado um estudo das medidas de semelhança existentes em aplicações de Informação Geográfica que permitem comparar trajectórias, bem como dos principais mecanismos de indexação para trajectórias. Realizou-se ainda uma análise comparativa entre duas estruturas de dados baseadas em distância (*M-Tree* e *RLC2*) em espaços métricos de trajectórias. Finalmente, com base nestes estudos e na avaliação, foi proposto um mecanismo de indexação específico para trajectórias que agiliza a pesquisa do *k* mais semelhantes em espaços métricos de trajectórias (*SimTraj*).

Na fase de preparação desta dissertação realizaram-se as seguinte actividades:

- Estudo e implementação das oito funções de semelhança em JAVA;
- Estudo das principais estruturas de dados específicas para trajectórias, quer as baseadas em distância (*RLC2* e *M-Tree*), como as baseadas em localização (*STR--Tree*, *TB-Tree*, *SEB-Tree*, *SETI* e *TrajStore*);
- Elaboração de um questionário para avaliar as funções de semelhança.

Após esta preparação estava-se em condição de integrar as funções de semelhança com os mecanismos de indexação. Tendo como base o trabalho da equipa GISS, optou-se por primeiro avaliar os mecanismos de indexação baseados em distância (*RLC2* e *M-Tree*).

Visto que estas estruturas de dados se encontram implementadas na linguagem *C++*, todas as funções de semelhança foram também implementadas nesta linguagem e passou-se a integrar estes mecanismos num protótipo que nos serviu para a avaliação.

Nesta fase de integração foram realizadas as seguintes actividades:

- Familiarização com a implementação da *RLC2*;
- Implementação dos *k* mais próximos na *RLC2*;
- Familiarização com a implementação da *M-Tree*.

De seguida, foi realizada a avaliação das pesquisas por semelhança, em particular dos *k* mais próximos neste protótipo experimental. Para tal efectuaram-se as seguintes tarefas:

- Escolha dos espaços métricos a utilizar nos testes. Foram escolhidos quatros espaços métricos utilizando dados reais de trajectórias de autocarros e furacões e as funções de semelhança *ERP* e L_2*-norm*. Quando se escolheram as funções de semelhança estava previsto fazê-lo com base no questionário elaborado. No entanto, isto não foi assim visto não se ter conseguido uma equipa de especialistas nesta área que respondessem ao questionário;
- Parametrização de ambas as estruturas de dados disponíveis. Essa parametrização consistiu na consulta de bibliografia para a *M-Tree* e na comparação de resultados experimentais para a *RLC2*;
- Análise dos resultados experimentais. Foram realizados testes experimentais para comparar o desempenho da pesquisa dos kNN em ambas as estruturas de dados métricas, usando diferentes métricas e conjuntos de dados. Desta avaliação, foi possível concluir-se que a *RLC2* e a *M-Tree* são bastantes competitivas. No entanto, a primeira demonstra ser ligeiramente superior no número de distâncias calculadas e tempo de execução;

Na avaliação realizada, foi notório que a estrutura de dados *M-Tree* não conseguiu minimizar, de todo, o número de elementos comparados. Este facto foi discutido com

Willian Oliveira, Mestre em Ciências Matemáticas e Computação (ICMC/USP), que faz parte do grupo GBDI, o qual indicou que a *M-Tree* não tem tido um bom desempenho em dados pouco esparsos. Tratou-se de se conseguir um novo conjunto de dados que possibilitasse um melhor desempenho na *M-Tree*. No entanto, não foi possível obter os dados neste domínio de aplicação.

Tendo por base a avaliação realizada, foi proposto um novo mecanismo de indexação específico para trajectórias, que agilize a pesquisa das *k* trajectórias mais semelhantes em espaços métricos de trajectórias. Este mecanismo foi baptizado de *SimTraj*.

Era intenção inicial deste trabalho integrar o mecanismo com um visualizador que estava a ser proposto por um aluno de Mestrado. No entanto, o aluno não concluiu este trabalho e, por conseguinte, não houve qualquer possibilidade de se proceder à integração.

Tendo a implementação da *SimTraj*, tratou-se de fazer uma avaliação entre este mecanismo e alguns mecanismos de indexação baseados em localização, nomeadamente a *TB-Tree*. No entanto, estes mecanismos estavam implementados em *VisualBasic* e, face ao tempo restante desta dissertação, optou-se para deixar esta actividade para trabalho futuro.

6.2. Contribuições

Durante a realização desta dissertação foram alcançadas todas as principais contribuições previstas no seu inicio (secção 1.4.):

- O estudo das medidas de semelhança usadas em aplicações de Sistemas de Informação Geográfica (*SIG*) que permitem comparar trajectórias (anexo A.1);
- O estudo dos principais mecanismos de indexação baseados em localização específicos para trajectórias (anexo A.2);
- A análise comparativa dos mecanismos de indexação baseados em distância (*M--Tree* e *RLC2*) em espaços métricos.

Para além destas contribuições relevantes, é importante realçar que o estudo realizado permitiu ainda oferecer à comunidade cientifica:

- Um protótipo experimental, implementado em *C++*, que permite as pesquisas das *k* trajectórias mais semelhantes em duas estruturas de dados baseadas em distância (*RLC2* e *M-Tree*). Neste protótipo encontram-se implementadas todas as funções de semelhança estudadas;
- A proposta de um mecanismo de indexação específico para trajectórias que agiliza a pesquisa das *k* trajectórias mais semelhantes em espaços métricos de trajectórias, denominado *SimTraj*;
- A submissão de dois artigos científico (Afonso, Barbosa, e Rodrigues 2011b; 2011a) que servirão de teste para uma avaliação a nível internacional. No primeiro, avalia-se a pesquisa dos *k* vizinhos mais próximos (para $k = 1$ e $k = 5$) em ambas as estruturas de dados (*M-Tree* e *RLC2*), utilizando o espaço métrico de dados dos autocarros, com as funções de semelhança *ERP* e L_2*-norm*. No segundo artigo, formaliza-se a proposta do mecanismo de indexação específico para trajectórias, utilizando, para avaliá-lo, o espaço métrico dos furacões com as mesmas funções de semelhança.

6.3. Trabalho Futuro

Como trabalho futuro apontam-se três pontos que parecem merecer bastante atenção e uma profunda reflexão.

O primeiro, diz respeito à avaliação do desempenho da *SimTraj* em pesquisas por semelhança, nomeadamente a sua comparação com mecanismos de indexação baseados em localização. Seria bastante interessante perceber o desempenho destes dois tipos de indexação com dados reais e, se possível, caracterizar em que situações se deve utilizar cada um dos mecanismos.

O segundo, é a utilização de mais conjuntos de dados para se efectuarem mais testes com as estruturas de dados *RLC2* e *M-Tree*, de preferência mais esparsos. Apesar de pensarmos que, nesta área, a maioria dos conjuntos de dados não são muito dispersos em termos de coordenadas, seria interessante investir um pouco mais nesta área.

Um último aspecto é a avaliação do desempenho da *SimTraj* no que diz respeito ao seu dinamismo e, se possível, o seu uso numa aplicação concreta.

7. Bibliografia

Afonso, Fábio, Fernanda Barbosa, e Armanda Rodrigues. 2011a. "SimTraj: An Approach to Similar Queries over Metric Trajectory Space." em *Proceedings of the 11th International Conference of Geocomputation*. Londres, Inglaterra.

Afonso, Fábio, Fernanda Barbosa, e Armanda Rodrigues. 2011b. "Trajectory Data Similarity with Metric Data Structures." em *Proceedings of 19th annual Geographic Information Systems Research UK Conference (GISRUK 2011)*. Portsmouth, Inglaterra.

Aurenhamme, Franz. 1991. "Voronoi diagrams - a survey of a fundamentalgeometric data structure." *ACM Computing Surveys* 23(3):345-405.

Barbosa, Fernanda. 2009. "Similarity-Based Retrieval in High Dimensional Data with Recursive Lists of Clusters: A Study Case with Natural Language Dictionaries." em *Submetido à International Conference on Information Management and Engineering (ICIME 2009)*.

Barbosa, Fernanda, e Margarida Mamede. 2007. "Queries in Natural Language Dictionaries with Recursive Lists of Clusters." em *Proceedings of the 22th International Symposium on Computer and Information Sciences (ISCIS 2007)*. Ancara, Turquia: IEEE Xplore.

Barbosa, Fernanda, e Armanda Rodrigues. 2009. "Range Queries over Trajectory Data with Recursive Lists of Clusters: a case study with Hurricanes data." Pp. 369-376 em *Proceedings of the 2009 Geographic Information Systems Research UK (GISRUK'09)*. Universidade de Durham, Reino Unido: GISRUK.

Bring, Sergey. 1995. "Near Neighbor Search in Large Metric Spaces." Pp. 574-584 em *Proceedings 21th International Conference on Very Large Data Bases (VLDB 1995)*. São Francisco, EUA.

Chávez, Edgar, R Navarro, Ricardo Baeza-Yates, e José Luis Marroquín. 2001. "Searching in Metric Spaces." *ACM Computing Surveys* 33(3):273-321.

Chakka, V. Prasad, Adam C. Everspaugh, e Jignesh M. Patel. 2003. "Indexing Large Trajectory Data Sets With SETI." em *Proceedings of the First Biennial Conference on Innovation Data Systems Research (CIDR'03)*. Asilomar, Calfórnia, EUA: ACM.

Chambel, Pedro, e Fernanda Barbosa. 2009. "Improving Similarity Search in Face-Images Data." em *Proceedings of the Second Workshop on Very Large Digital Libraries (VLDL 2009),*

13th European Conference on Research and Advanced Technologies on Digital Libraries (ECDL 2009). Corfu, Grécia.

Chen, L. 2005. "Similarity Search Over Time Series and Trajectory Data." Doctor of Philosophy in Computer Science, Waterloo, Ontário, Canadá: Universidade de Waterloo.

Chen, L, M.Tamer Ozsu, e Vincent Oria. 2005. "Robust and Fast Similarity Search for Moving Object Trajectories." Pp. 491 - 502 em *Proceedings of the 2005 ACM SIGMOD international conference on Management of data (SIGMOD'05)*. Baltimore, Maryland, EUA: ACM.

Ciaccia, Paolo, Macro Patella, e Pavel Zezula. 1997. "M-tree: An Efficient Access Method for Similarity Search in Metric Spaces." Pp. 426-435 em *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB 1997)*. Atenas, Grécia: Morgan Kaufmann Publishers.

Comer, Douglas. 1979. "The Ubiquitous B-Tree." *ACM Computing Surveys* 11(2):121-137.

Costa, FMR. 2009. "Geração automática de "playlists" de músicas semelhantes." Dissertação de Mestrado em Engenharia Informática, Caparica, Portugal: Universidade Nova de Lisboa Faculdade de Ciências e Tecnologia Departamento de Informática.

Costa, F, e Fernanda Barbosa. 2009. "Timbre Similarity Search with Metric Data Structures." em *Proceedings of the Workshop on Exploring Musical Information Spaces (WEMIS 2009), 13th European Conference on Research and Advanced Technologies on Digital Libraries (ECDL 2009)*. Corfu, Grécia.

Cudre-Mauroux, Philippe, Eugene Wu, e Samuel Madden. 2010. "TrajStore: An adaptive Storage System for Very Large Trajectory Data Sets." Pp. 109-120 em *Proceedings of the 26th Internacional Conference on Data Engineering (ICDE'10)*. Long Beach, Califórnia, EUA: IEEE Computer Society.

Ding, Hui, Goce Trajcevski, e Peter Scheuermann. 2008. "Efficient Similarity Join of Large Sets of Moving Object Trajectories." em *Proceedings of the 2008 15th International Symposium on Temporal Representation and Reasoning (TIME'08)*, vol. 79-87. Monteéal, Canadá: IEEE Computer Society.

Frentzos, Elias, Kostas Gratsias, Nikos Pelekis, e Yannis Theodoridis. 2007. "Algorithms for Nearest Neighbor Search on Moving Object Trajectories." *GeoInformatica* 11(2):159-193.

Frentzos, Elias, Kostas Gratsias, e Yannis Theodoridis. 2007. "Index-based Most Similar Trajectory Search." Pp. 816 - 825 em *IEEE 23rd International Conference on Data Enginering (ICDE'07)*. Istambul, Turquia: IEEE Computer Society.

Gao, Yunjun, B Zheng, G Chen, e Qing Li. 2009. "Algorithms for constrained k-nearest neighbor queries over moving object trajectories." *GeoInformatica* 14(2):241-276.

GBDI. 2009. "M-Tree." *Databases and Images Group*. http://www.gbdi.icmc.usp.br/en/home.

Giannotti, Fosca, Mirco Nanni, Dino Pedreschi, e Fabio Pinelli. 2009. "Trajectory Pattern Analysis for Urban Traffic." Pp. 43-47 em *Proceedings of the Second International Workshop on Computational Transportation Science (IWCTS'09)*. Seattle, Washington, EUA: ACM.

Giannotti, Fosca, Mirco Nanni, Fabio Pinelli, e Dino Pedreschi. 2007. "Trajectory Pattern Mining." Pp. 330-339 em *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'07)*. San Jose, Califórnia, EUA: ACM.

Google. 2010. "k-d-tree." *kdtree*. http://code.google.com/p/kdtree/.

Greenheck, Daniel. 2009. "Quad-tree." *QuadTree*. http://digitseven.com/QuadTree.aspx.

Keogh, Eamonn J., e Michael J. Pazzani. 1999. "An Indexing Scheme for Fast Similarity Search in Large Time Series Databases." Pp. 56-67 em *Proceedings of the 11th International Conference on Scientific and Statistical Database Management (SSDBM'99)*. Cleveland, Ohio, EUA: IEEE Computer Society.

Krafft, Martin F., Paul Harris, e Sylvain Bougerel. 2009. "k-d-tree." *libkdtree++*. http://libkdtree.alioth.debian.org/.

Laurinen, Perttu, Pekka Siirtola, e Juha Roning. 2006. "Efficient algorithm for calculating similarity between trajectories containing an increasing dimension." Pp. 392-399 em *Proceedings of the 24th IASTED international conference on Artificial intelligence and applications*. Innsbruck, Áustria: ACTA Press.

Li, Quannan et al. 2008. "Mining User Similarity Based on Location History." P. 3 em *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems (GIS'08)*. Irvine, Califórnia, EUA: ACM.

Mamede, Margarida. 2005. "Recursive Lists of Clusters: A Dynamic Data Structure for Range Queries in Metric Spaces." Pp. 843-853 em *Proceedings of the 20th International Symposium on Computer and Information Sciences (ISCIS'05)*. Istambul, Turquia: Springer-Verlag.

Micó, María Luisa, José Oncina, e Enrique Vidal. 1994. "A new version of the nearest-neighbour approximating and eliminating search algorithm." *Pattern Recognition Letters* 15 (1):9-17.

Mokbel, Mohamed F., Thanaa M. Ghanem, e Walid G. Aref. 2003. "Spatial-temporal Access Methods." *IEEE Data Engineering Bulletin*, 40-49.

Monreale, Anna, Fabio Pinelli, Roberto Trasarti, e Fosca Giannotti. 2009. "WhereNext: a Location Predictor on Trajectory Pattern Mining." Pp. 637-646 em *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'09)*. Paris, França: ACM.

Navarro, G. 2002. "Searching in Metric Spaces by Spatial Approximation." *Very Large Data Bases Journal* 11(1):28-46.

Panagiotakis, Costas, Nikos Pelekis, e Ioannis Kopanakis. 2009. "Trajectory Voting and Classification based on Spatiotemporal Similarity in Moving Object Databases." Pp. 131-142 em *Proceedings of the 8th International Symposium on Intelligent Data Analysis: Advances in Intelligent Data Analysis VIII*. Lyon, França: Springer-Verlag.

Pelekis, Nikos, Ioannis Kopanakis, Evangelos E. Kotsifakos, Elias Frentzos, e Yannis Theodoridis. 2009. "Clustering Trajectories of Moving Objects in an Uncertain World." Pp. 417-427 em *Ninth IEEE International Conference on Data Mining (ICDM'09)*. Miami, Flórida, EUA: IEEE Computer Society.

Pelekis, Nikos et al. 2007. "Similarity Search in Trajectory Databases." Pp. 129-140 em *Proceedings of 14th International Symposium on Temporal Representation and Reasoning (TIME'07)*. Alicante, Espanha: IEEE Computer Society.

Pfoser, Dieter, Christian S. Jensen, e Yannis Theodoridis. 2000. "Novel Approach to the Indexing of Moving Objects Trajectories." Pp. 395-406 em *Proceedings of the 26th Internacional Conference on Very Large Database (VLDB'00)*. Cairo, Egito: Endowment.

Roh, G-P, J-W Roh, Seung-Won Hwang, e Byoung-Kee Yi. 2010. "Supporting Pattern Matching Queries over Trajectories on Road Networks." *IEEE Transactions on Knowledge and Data Engineering*.

Samet, Hanan. 2006. "Foundations of Multidimensional and Metric Data Structures." Pp. 50-89; 270-311; 356-357; 566; 608 em *Foundations of Multidimensional and Metric Data Structures*. Estados Unidos da América: Morgan Kaufmann Publishers Inc.

Sarmento, Ângelo Miguel Loureiro. 2010. "Estruturas de Dados Métricas Genéricas Memória Secundária." Dissertação de Mestrado em Engenharia Informática, Caparica, Portugal: Universidade Nova de Lisboa Faculdade de Ciências e Tecnologia Departamento de Informática.

Silva, Pedro Miguel Chambel. 2009. "Pesquisa de Imagens de Rosto." Dissertação de Mestrado em Engenharia Informática, Caparica, Portugal: Universidade Nova de Lisboa Faculdade de Ciências e Tecnologia Departamento de Informática.

Song, Zhexuan, e Nick Roussopoulos. 2003. "SEB-tree: An Approach to Index Continuously Moving Objects." Pp. 340-344 em *Proceedings of 4th International Conference on Mobile Data Management (MDM'03)*. Melbourne, Austrália: Springer-Verlag.

Theodoridis, Yannis. 2005. "R-tree." *R-tree Portal - Home*. http://www.rtreeportal.org/.

Tiakas, Eleftherios et al. 2009. "Searching for similar trajectories in spatial networks." *Information Systems* 34(3):328-352.

Tiesyte, Dalia, e Christian S. Jensen. 2008. "Similarity-Based Prediction of Travel Times for Vehicles Traveling on Known Routes." P. 14 em *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems (GIS'08)*. Irvine, Califórnia, EUA: ACM.

Trajcevski, Goce, Hui Ding, Peter Scheuermann, Roberto Tamassia, e Dennis Vaccaro. 2007. "Dynamics-Aware Similarity of Moving ObjectsTrajectories." P. 11 em *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems (ACMGIS'07)*. Seattle, Washington, EUA: ACM.

Vlachos, Michail, George Kollios, e Dimitrios Gunopulos. 2002. "Discovering Similar Multidimensional Trajectories." P. 673 em *Proceedings of the 18th International Conference on Data Engineering (ICDE'02)*. São Jose, Califórnia, EUA: IEEE Computer Society.

Wang, Xiaoyue, Hui Ding, Goce Trajcevski, Peter Scheuermann, e Eamonn J. Keogh. 2008. "Querying and mining of time series data: experimental comparison of representations and distance measures." Pp. 1542-1552 em *Proceedings of the VLDB Endowment (VLDB'08)*. Auckland, Nova Zelândia: VLDB Endowment.

Yanagisawa, Yutaka, Jun-ichi Akahani, e Tetsuji Satoh. 2003. "Shape-based Similarity Query for Trajectory of Mobile Objects." Pp. 63-77 em *Proceedings of the 4th International Conference on Mobile Data Management (MDM'03)*. Melbourne, Austrália: Springer-Verlag.

Yianilos, Peter. 1993. "Data Structures and Algorithms for Nearest Neighbor Search in General Metric Spaces." Pp. 312-321 em *Proceedings of the 4th Annual SIAM Symposium on Discrete Algorithms*. Filadélfia, EUA: ACM.

Zezula, Pavel, Giuseppe Amato, Vlastislav Dohnal, e Michal Batko. 2006. "Similarity Search: The Metric Space Approach." Pp. 101-112 em *Similarity Search: The Metric Space Approach*, vol. 32, *Advances in Database Systems*. Springer.

A Anexos

A.1 Funções de semelhança

A.1.1 Dissim

Esta função de semelhança pode ser calculada de forma a retornar o valor exacto ou apenas uma aproximação. Normalmente é calculado o valor aproximado, pois este cálculo é realizado mais rapidamente e com uma eficácia em tudo semelhante à outra possibilidade. A grande diferença entre ambos é o calculo do integral da área onde se encontra a trajectória no caso do valor exacto.

Para se proceder ao cálculo da distância em si utilizam-se todos os pontos da trajectória, excepto o primeiro, sendo adicionado uma penalização caso o tempo seja diferente nos pontos que estão a ser processados das respectivas trajectórias.
A *Dissim* é uma função de semelhança espaço-temporal que considera a trajectória como um todo. Sejam Q e T duas trajectórias definidas no intervalo de tempo $[t_1,t_n]$. A função *DISSIM* está definida na formula (5).

$$DISSIM(Q,T)=\sum_{k=1}^{n-1}\int_{t_k}^{t_{k+1}} D_{Q,T}(t)\,dt \qquad (5)$$

De onde, t_k representa o tempo registados por ambas as trajectórias no instante k e a função $D_{Q,T}(t)$ representa a distância Euclidiana, calculada tendo por base as respectivas coordenadas x e y nesse instante.

Uma aplicação desta função poderá ser encontrada em (Frentzos, Gratsias, e Theodoridis 2007), onde são definidas mais duas variantes desta função que poderão, ou não, ser dependentes da velocidade, sendo estas caracterizadas como sendo uma função optimista ou pessimista, *OPTDISSIM* e *PESDISSIM*, respectivamente.

No que à avaliação da qualidade diz respeito, ao longo da leitura de (Frentzos, Gratsias, e Theodoridis 2007), torna-se evidente que esta é uma boa estratégia, pois nos testes a sua eficácia foi bastante melhor do que a das funções de semelhança *LCSS* e *EDR* (apresentadas na secção A.1.4 e secção A.1.5.1 , respectivamente).

A.1.2 L_p-norm

A família de funções L_p-norm é vastíssima, pois *p* compreende os valores entre um e infinito. No entanto, de entre todas, apenas três (*L_1-norm*, *L_2-norm* e *L_∞-norm*) merecem uma maior atenção neste campo de investigação, pois poderão ser utilizadas por si só ou por funções mais poderosas que necessitam de uma função auxiliar no calculo da distância.

Esta função é uma das mais utilizadas para se calcular a similaridade entre trajectórias, pois sendo métricas poderão tomar vantagem na utilização de métodos de indexação para pesquisas por semelhança em espaços métricos (Samet 2006; Chávez et al. 2001).
Nesta função, as trajectórias poderão ser consideradas como um todo ou somente porções isoladas num determinado período de tempo. No entanto, estas terão de ter tamanho igual, ou seja, o instante inicial e final de ambas as trajectórias tem de ser coincidente. Para tal poderá ser necessário estender uma delas.

Sejam duas trajectórias *R* e *S*, ambas de comprimento *N*, esta função será definida na formula (6).

$$L_p - norm(R,S) = \sqrt[p]{\sum_{i=1}^{n} (r_i - s_i)^p} \qquad (6)$$

Apesar das vantagens a cima descritas, esta função, por norma, não é utilizada por si só nos trabalhos de investigação, mas sim combinada com funções mais poderosas que necessitam, para os seus cálculos, de outras complementares.

Uma variante desta função também bastante interessante é a L_p-*norm* com peso, a qual para cada ponto da trajectória terá um peso específico, podendo então dar-se mais ênfase a locais de maior interesse ou, no caso de previsões, será lógico, dar um peso maior aos pontos que surgiram mais recentemente do que aos mais antigos. Esta está definida na formula (7).

$$L_p - norm(R,S,W) = \sqrt[p]{\sum_{i=1}^{N} w_i (r_i - s_i)^p} \qquad (7)$$

Pode-se ver a sua aplicação em (Tiesyte e Jensen 2008). Neste mesmo artigo é concluído que a função LCSS (secção A.1.4) em termos de eficiência leva uma boa vantagem sobre a L_p-norm, quer nos resultados, quer no uso de memória e CPU utilizado.

A.1.2.1 L_1-norm (Distância de Manhattan)

A distância de Manhattan é a mais simplista desta família, pois é o calculo da distância no plano. Em verdade, o que se obtém é o valor absoluto da distância.

Particularmente às trajectórias, tal como na *Dissim*, uma pequena penalização é aplicada ao cálculo da distância de cada conjunto de pontos, caso estes tenham um valor de tempo diferente.

Trata-se de um caso particular da L_p-*norm* com $p=1$ e está definida na formula 8:

$$L_1 - norm(R, S) = \sum_{i=1}^{N} (r_i - s_i) \quad (8)$$

Consultando (Wang et al. 2008), torna-se claro o porquê de não ser utilizada como único ponto para medir a similaridade entre trajectórias, visto que a sua eficácia fica muito aquém da oferecida por funções pertencentes à mesma família, nomeadamente a L_2-*norm* e a L_∞-*norm*.

A.1.2.2 L_2-norm (Distância Euclidiana)

Pode-se dizer que esta, de todas as variantes da L_p-*norm* é a mais utilizada em trabalhos de investigação, seja sozinha (Trajcevski et al. 2007; Laurinen et al. 2006) ou como auxiliar de outras normas, tais como a *DTW* (Ding et al. 2008).

Tomando a fórmula generalizada e aplicando $p=2$ obtém-se a formula definida em (9).

$$L_2 - norm(R, S) = \sqrt{\sum_{i=1}^{N} (r_i - s_i)^2} \quad (9)$$

A.1.2.3 L_∞-norm (Norma Máxima)

A distância máxima, ou norma máxima, não é mais do que o cálculo da distância (no caso deste trabalho aplica-se a L_2-*norm* para tal) entre dois pontos da trajectória, obtendo-se como resultado a maior distância em valor absoluto.

Como todas as outras funções definidas, caso o tempo dos pontos seja diferente será aplicada uma pequena penalização ao resultado da distância.

Esta também não é uma das funções mais utilizadas, no entanto torna-se interessante a sua comparação com outras desta mesma família L_p, como nos é descrito em (Wang et al. 2008). Ao se comparar as três variantes aqui apresentadas, torna-se claro que a L_2-*norm*, em termos de eficácia, supera qualquer uma das outras duas.
Partindo-se da formula geral, ao igualar-se p a ∞ obter-se-à a formula definida em (10).

$$L_\infty - norm(R, S) = max_{i=1}^{N}(|r_i - s_i|) \quad (10)$$

A.1.3 Dynamic Time Warping (*DTW*)

Esta função permite medir a similaridade entre duas sequências, nas quais o tempo e/ou velocidade possam variar. Um exemplo clássico é ter um atleta a realizar duas voltas à mesma pista, uma a andar e outra a correr, esta função permitiria reconhecer padrões semelhantes em ambas as trajectórias. Ao conseguir lidar com desvios temporais, poderá ser necessário estender trajectórias.

Enumeras vezes é utilizada pois permite a correspondência exacta de sequências mesmo com falta de informação, desde que hajam segmentos suficientemente grandes para ocorrerem as correspondências.
Nesta função é considerada a "forma" da trajectória, ou seja, quanto mais as "formas" destas se assemelharem mais semelhantes são as trajectórias como um todo. No entanto, esta-se na presença de uma função não métrica, visto esta não obedecer à desigualdade triangular.

Sejam duas trajectórias R e S de comprimentos M e N respectivamente, a função *DTW* esta definida em (11).

$$DTW(R,S)=\infty \quad se\, M=0 \vee N=0;$$

$$DTW(R,S)=|r_1-s_1|+\min\{DTW(Rest(R),Rest(S)),\\ DTW(Rest(R),S),DTW(R,Rest(S))\} \quad caso \quad contrário \tag{11}$$

Quanto à sua aplicação em trabalhos, podemos ver que esta função foi aplicada com resultados extremamente positivos em (Ding et al. 2008; Vlachos et al. 2002; Wang et al. 2008). Neste último trabalho é tornado bem claro o domínio da *DTW* sobre todas as outras funções do teste, somente sendo superada pela *EDR* (sendo esta função apresentada na secção A.1.5.1), mas por uma margem mínima.

A.1.4 Longest Common SubSequences (*LCSS*)

Um dos objectivos desta função é a localização da maior sub-sequência comum entre as trajectórias. No entanto, esta foi desenhada para ser robusta à possível existência de ruído, ou seja, poderão ter sido introduzidos erros na definição da trajectória a quando da sua construção pelo hardware ou mesmo pelo canal de transmissão. Assim sendo, a *LCSS* propõem-se a remover este problema ao contar somente o número de elementos com correspondência perfeita entre as trajectórias.

Infelizmente, não se está na presença de uma função métrica, no entanto tem as virtudes de poder ser aplicada a trajectórias de tamanho variável, bem como a de lidar bem com variações temporais (L. Chen 2005).

Sendo *R* e *S* duas trajectórias, tendo a primeira cumprimento *M* e a segunda *N*, a função *LCSS* está definida na formula (12).

$$
\begin{aligned}
&LCSS(R,S)=0 \quad se \quad M=0 \vee N=0;\\
&LCSS(R,S)=LCSS(Rest(R),Rest(S))+1 \quad se \quad dist(r_{1,}s_1)\leqslant \varepsilon;\\
&LCSS(R,S)=\min\{LCSS(Rest(R),S),\\
&\qquad\qquad LCSS(R,Rest(S))\} \quad caso \quad contrário
\end{aligned}
\tag{12}
$$

onde a função *dist(r_1,s_1)* poderá ser qualquer uma das L_p-*norm* e ε representa o limiar da distância entre dois elementos, ou seja, caso não a excedam estes são vistos como iguais. Ter-se-à no entanto de ter presente um pormenor de vital importância, ao contrário de todas as outras funções esta não tem em conta a distância propriamente dita, mas sim um resultado (*score*). Este pode, no entanto, ser convertido para distância ao ser aplicada a fórmula defina em (13).

$$
LCSS_{dist}(R,S)=1-\frac{LCSS(R,S)}{min(|R|,|S|)} \tag{13}
$$

Somente é possível superar a questão da introdução de ruído devido à existência de um limiar (ε) que quantifica os elementos com um de dois valores possíveis (zero ou um), sendo removido posteriormente os efeitos nefastos causados por este na maior distância.

Analisando os trabalhos de investigação, pode-se comprovar que esta função por norma é utilizada em trajectórias vistas como um todo (Vlachos et al. 2002; Tiesyte e Jensen 2008). Avaliando os resultados obtidos nestes trabalhos, juntamente com os presentes em (Wang et al. 2008), são evidenciadas aplicações em que a *LCSS* é vista como sendo a melhor solução (primeira referência), e outras em que a sua acuidade é em tudo semelhante a funções como a *EDR* ou *ERP* (segunda referência).

A.1.5 Edit Distance (*ED*)

A família de funções de edição de distância têm uma particularidade interessante, pois aqui a semelhança é calculada de uma forma diferente. Não se comparam directamente pontos, mas sim o número de transformações necessárias para transformar um objecto no outro.

Define-se tendo por base a função L_2-*norm* e é vulgarmente aplicada em trabalhos de investigação, visto que têm uma eficácia muitíssimo boa, sendo normalmente utilizadas em strings. Uma outra vantagem, igualmente, proporcionada por esta família de funções é a de satisfazerem as propriedades referentes à métrica, e por conseguinte tem assim, a possibilidade de utilizar métodos de indexação que agilizam a pesquisa por semelhança (Samet 2006; Chávez et al. 2001). A função *ERP*, variante da *ED*, já foi utilizada numa avaliação de pesquisa por semelhança em base de dados de furações (Barbosa e Rodrigues 2009).

Podem ainda ser destacadas outras duas vantagens nesta família de funções. A primeira, refere-se ao tratamento eficaz de trajectórias que sofrem desvios temporais e a segunda, ao suporte à introdução de ruído (L. Chen 2005). Desta forma, pode-se afirmar que esta gama de funções são reais concorrentes, no verdadeiro sentido da palavra, às funções *LCSS* e *DTW*.

Passa-se a apresentar duas variantes da *ED*, que se aplicam a valores reais, podendo ou não considerar penalizações na diferença. Nas definições abaixo ir-se-à considerar duas trajectórias S e R de tamanhos M e N respectivamente.

A.1.5.1 Edit Distance on Real Sequences (EDR)

A avaliação desta função é feita de uma maneira um pouco peculiar, visto que a distância é calculada com base no número de transformações que um objecto tem de sofrer para ser igual ao outro. A *EDR* suporta algum desvio dos valores, ou seja, os pontos não têm de ser coincidentes para ser considerados como iguais.

Esta é a função de edição aplicada a valores reais está definida na formula (14).

$$
\begin{aligned}
&EDR(R,S)=M \quad se \quad N=0;\\
&EDR(R,S)=N \quad se \quad M=0;\\
&ERD(R,S)=\min\{EDR(Rest(R),Rest(S))+subcost,\\
&\qquad EDR(Rest(R),S)+1, EDR(R,Rest(S))+1\} \quad caso \quad contrário
\end{aligned}
\tag{14}
$$

onde o *subcost* valerá zero se e só se $|r_i - s_i| \leqslant \varepsilon$, onde ε é o valor máximo da distância entre dois pontos para que este sejam considerados como iguais, caso contrário o seu valor será um.

Devido ao resultado obtido advir das transformações realizadas, estas terão um custo unitário e poderão passar por inserções, remoções ou substituições.

Com base nos trabalhos de investigação existentes, é fácil de concluir que esta função se insere no grupo das mais eficazes, e será, talvez, a que tem mesmo melhores resultados, tal com é dito em (Wang et al. 2008). Uma outra ilação que pode ser retirada diz respeito ao ruído, a *EDR* quando comparada com a *DTW* ou *LCSS* apresenta melhores resultados,

no entanto, caso as trajectórias não tenham ruído introduzido, como as apresentadas em (L. Chen et al. 2005), as três funções produzem resultados muito semelhantes.

A.1.5.2 Edit Distance with Real Penalty (ERP)

Quanto a esta variante poderá ser vista como sendo uma evolução da combinatória da *LCSS* com a *DTW*, sendo que difere da primeira ao não existir o parâmetro de tolerância ε, e da segunda ao não procede à replicação dos elementos anteriores.

Com base nestas características esta função poderá ser formalizada como ilustrado na formula (14).

$$ERP(R,S)=\sum_{i=1}^{M}|r_i-g| \quad se \quad N=0;$$

$$ERP(R,S)=\sum_{i=1}^{N}|s_i-g| \quad se \quad M=0;$$

$$\begin{aligned} ERP(R,S)=\min\{ & ERP(Rest(R),Rest(S))+dist_{erp}(r_1,s_1), \\ & ERP(Rest(R),S)+dist_{erp}(r_1,g), \\ & ERP(R,Rest(S))+dist_{erp}(s_1,g)\} \quad caso\,contrário \end{aligned} \tag{15}$$

onde

$$dist_{erp}(r_i,s_i)=|r_i-s_i| \quad se \quad não \quad considerada \quad falha;$$

$$dist_{erp}(r_i,s_i)=|r_i-g| \quad se \quad s_i \quad considerado \quad como \quad falha;$$

$$dist_{erp}(r_i,s_i)=|s_i-g| \quad se \quad r_i \quad considerado \quad como \quad falha.$$

e sendo *g* um ponto constante associada à falha.

Recorrendo-se aos dados provenientes da investigação para se avaliar convenientemente esta função, em (Wang et al. 2008) é dito que a eficácia da *ERP* é muitíssimo semelhante à da *LCSS*, *DTW* e *EDR*, sendo que em alguns casos é demonstrado que a *EDR* supera as outras três.

Apesar da semelhança existente nos resultados, a *ERP* tem a vantagem de ser métrica o que não se verifica com a *LCSS*, nem com a *DTW*. No entanto, ambas as funções da família da *ED*, apresentadas, têm características semelhantes, sendo que, em alguns casos, a *ERD* supera a *ERP*, o que sugere uma séria ponderação sobre qual utilizar.

A.2 Métodos de indexação específicos de trajectórias

A.2.1 Spatial-Temporal Tree (*STR-Tree*)

A estrutura de dados *STR-Tree* (Pfoser et al. 2000) é uma extensão da *R-Tree* e, por conseguinte, é uma árvore balanceada e, as suas folhas contêm os segmentos de trajectórias. Cada segmento é da forma *(id, #traj, MBB, orientation)*, com *id* a representar o identificador do objecto (segmento), *#traj* o número da trajectória, *MBB (Minimum Boundig Box)* representa uma região num espaço de n-dimensões e *orientation ε {1,2,3,4}*, representando a orientação do segmento *MBB*. Na Ilustração A.1, retirada de (Pfoser et al. 2000), é possível ver a aproximação de trajectória usando o *MBB* num espaço tridimensional e as orientações do segmento na região. Os nós internos da árvore são da forma *(ptr, MBB)* onde *ptr* é um apontador para um filho e *MBB* é a região coberta.

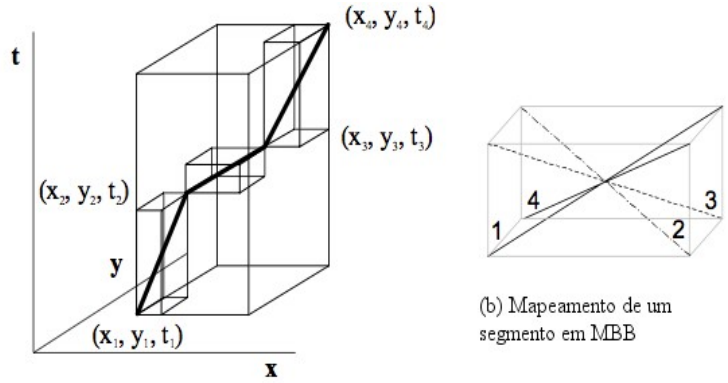

Ilustração A.1: Aproximação de trajectória e mapeamento de segmento na SRT-Tree

Nesta estrutura de dados, um dos objectivos é tratar de manter os segmentos pertencentes à mesma trajectória o mais juntos possíveis na *STR-Tree*. Para isso, a estrutura de dados tem um parâmetro *p* que indica o número de níveis "reservados" para garantir a proximidade de todos os segmentos de uma dada trajectória. Assim sendo, no momento de inserção de um novo segmento numa trajectória é procurado o nó onde se encontra o seu segmento anterior. Se existir lugar neste nó, ou em algum parente até ao nível *p-1,* este é colocado. Caso contrário é necessário reorganizar a estrutura do nó, ou seja, será necessário dividi-lo.

Para se proceder à divisão ter-se-à de ter em conta quais os tipos de segmentos nele contidos. Assim sendo, (1) caso todos os segmentos no nó não se encontrem conectados será invocado o algoritmo *QuadraticSplit* (Ilustração A.2 *(a)*), (2) se existir pelo menos um segmento desconectado, os que se encontram conectados são colocados num novo nó (Ilustração A.2 *(b)*) e (3) se não existir nenhum segmento desconectado, o segmento com "maior tempo" é colocado num novo nó (Ilustração A.2 *(c)*).

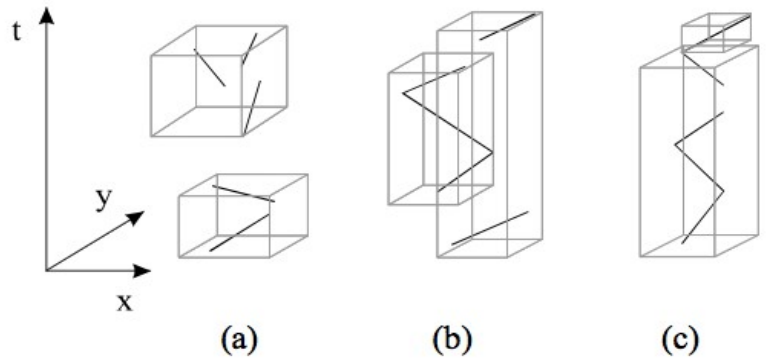

Ilustração A.2: Diferentes cenários de particionamento[5]

No artigo (Pfoser et al. 2000), esta estrutura de dados foi usada para pesquisar quais as trajectórias que se encontram num dado período de tempo $[t_1,t_2]$ numa dada região *R*. Esta pesquisa é realizada em dois passos.

5 Retirada de (Pfoser et al. 2000)

Foi realizada uma avaliação neste artigo que comparou as estruturas de dados *STR-Tree*, *TB-Tree* e *R-Tree*, sendo que a *TB-Tree* mostrou um melhor desempenho neste tipo de pesquisas.

A.2.2 Trajectory-Bundle Tree (*TB-Tree*)

A estrutura de dados *TB-Tree* proposta em (Pfoser et al. 2000) assenta igualmente na *R-Tree*, mas esta garante que numa dada folha da árvore apenas estejam segmentos referentes à mesma trajectória. Assim sendo, na folha está guardado o número da trajectória e os elementos da folha são da forma *(id, MBB, orientation)* com *id* a representar o identificador do objecto (segmento), *MBB (Minimum Boundig Box)* é uma região no espaço e *orientation ε {1,2,3,4}*, representando a orientação do segmento *MBB*.

No momento da inserção de uma trajectória na *TB-Tree* esta é dividida em *M* segmentos. Assim, para inserir um novo nó apenas se tem de encontrar o nó folha que contém o seu predecessor. Poder-se-à obter uma de duas situações:

- Nó suporta a imediata alocação do segmento;
- Nó atingiu a capacidade máxima.

O segundo caso obrigará à criação de um novo nó.

Na Ilustração A.3 retirada de (Pfoser et al. 2000) é possível ver uma parte da *TB-Tree*. A trajectória aí mostrada está distribuída por seis nós e estes estão conectados por uma lista ligada.

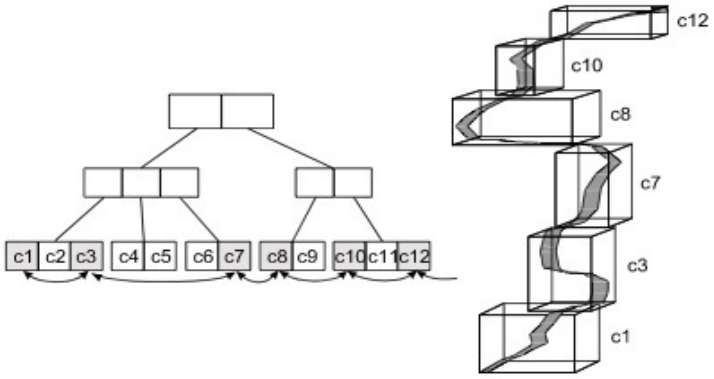

Ilustração A.3: Estrutura da TB-Tree

Desta forma, os segmentos de uma trajectória estão distribuídos por folhas da árvore não ligadas. De modo a agilizar a pesquisa e a preservar a informação total referente à trajectória, estes nós estão relacionados usando uma lista ligada.

No artigo (Pfoser et al. 2000), a pesquisa realizada na *TB-Tree* é muito semelhante à realizada na *STR-Tree*, tendo esta, como objectivo, encontrar as trajectórias, numa dada região, num intervalo de tempo.

Como foi dito na secção referente à *STR-Tree*, a *TB-Tree* obteve melhor desempenho na avaliação realizada em (Pfoser et al. 2000).

A.2.3 Start/End time stamp B-tree (*SEB-tree*)

A *SEB-Tree* é uma estrutura específica para trajectórias. Esta estrutura de dados foi proposta em (Song e Roussopoulos 2003) e está integrada com uma base de dados *MOD* (*Moving object Database*), tendo como principal objectivo o de efectuar pesquisas em que, dada uma região *R* e um intervalo de tempo $[t_1,t_2]$ com $t_1 \leq t_2 \leq now$, se pretende encontrar os elementos que se encontram, nesse período de tempo, nessa região.

O espaço é particionado em zonas utilizando diagramas de Voronoi (Aurenhamme 1991). Cada objecto na estrutura conhece o identificador da zona em que se encontra, e sempre que ultrapassa o limite da zona, é transferido para outra zona.

A cada zona do espaço está associada uma *SEB-Tree*. Cada elemento na estrutura é da forma *(id, t_s, t_e)*, sendo *id* o identificador do objecto, t_s o tempo inicial e t_e o tempo final, com $t_s \leq t_e \leq now$. Na estrutura correspondente à zona existem tuplos que representam registos históricos dos objectos, isto é, objectos que já estiveram na zona. Existe ainda um tuplo *(id, t_s, now)* para cada elemento que está na zona neste momento (após a última actualização).

Na Ilustração A.4, retirada de (Song e Roussopoulos 2003), é possível ver o mecanismo de inserção de um novo ponto com *B=3*, sendo *B* o número máximo de elementos que podem ser guardados numa página. A *SEB-Tree* guarda os registos históricos ordenando--os pelo tempo final (t_e), isto é, o registo com tempos *(t_{s1}, t_{e1})* está primeiro que o registo com tempos *(t_{s2}, t_{e2})* se $t_{e1} \leq t_{e2}$. O mecanismo de indexação destes registos históricos está implementado construindo um espaço de duas dimensões, onde o eixo *x* é o tempo inicial (t_s) e o eixo *y* é o tempo final (t_e). No início, a árvore apenas contém um nó e os pontos são inseridos neste. Quando este atinge o seu tamanho máximo (ver Ilustração A.4 *(b)*), são desenhadas duas linhas, uma vertical e outra horizontal, representando estas o valores máximos de t_s e t_e, respectivamente. Após tal acontecer, uma de duas situações poderá suceder aquando da inserção de novos pontos: o seu tempo inicial é menor que a linha e este é inserido à esquerda (ver Ilustração A.4 *(c)*); ou é maior e então será inserido à direita desta (ver Ilustração A.4 *(d*)).

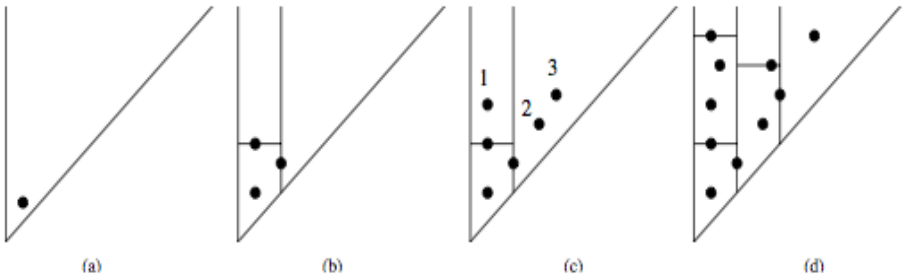

Ilustração A.4: Procedimento de Inserção para B=3

No artigo (Song e Roussopoulos 2003), são apresentados resultados de uma avaliação entre a *R-Tree*, *TB-Tree* e *SEB-Tree*, referentes a inserções e pesquisas de trajectórias semelhantes, dada uma região num dado intervalo de tempo. A *SEB-Tree* mostra um desempenho superior a todas as outras estruturas de dados, no que às inserções diz respeito. No entanto, em pesquisas, os desempenhos são muito semelhantes. Na secção quatro deste mesmo artigo, os autores manifestam o desejo de procederem à avaliação de pesquisas do tipo *k* vizinhos mais próximos, mas não se conseguiu obter referências posteriores desta estrutura de dados.

A.2.4 Scalabel and Efficient Trajectory Index (*SETI*)

O mecanismo de indexação *SETI* foi proposto em (Chakka et al. 2003) sendo este composto por dois níveis de indexação que separam a dimensão espacial da temporal. Neste, são utilizadas duas estruturas de dados, *R-Tree* (Theodoridis 2005) e *R*-Tree* (Theodoridis 2005). Procede, ainda, ao particionamento da dimensão do espaço em regiões estáticas e disjuntas, sendo que, para cada uma destas regiões, existe um mecanismo de indexação sobre a dimensão tempo. A motivação para este particionamento do espaço prende-se com o facto de a dimensão que mais varia (aumenta) num dado movimento é a dimensão do tempo, visto que as alterações referentes às localizações são mínimas, isto é, normalmente estão na mesma região. Se um dado segmento da trajectória cruzar duas ou mais regiões, então esse segmento é dividido e cada sub-segmento colocado nas regiões respectivas.

Uma trajectória em *SETI* é representada por $(t_{id}, <\mu_0, \mu_1,..., \mu_n>)$, onde t_{id} é o identificador da trajectória e μ_i é uma sequência de posições do objecto em movimento. Cada posição μ_i é um triplo (x_i, y_i, t_i), onde t_i representa o tempo e x_i e y_i representam as posições espaciais do objecto no plano *xy*. Um segmento de trajectória tem a forma $(t_{id}, i, \mu_{i-1}, \mu_i)$.

Para a indexação da dimensão espacial é usada uma *R-Tree* em que cada folha mantém os segmentos de trajectórias que pertencem ao *MBB* associado à folha. A cada folha está associado um tempo de vida, o qual é representado por um intervalo de tempo que cobre todos os segmentos lá inseridos. Estes índices temporais são mantidos numa R^*-*Tree* (*R-Tree* com *MBB* disjuntos). Para além disso, existe uma estrutura de dados, por exemplo uma *hash table*, em memória principal, que mantém a última posição para todos os objectos em movimento, para facilitar o processamento da inserção.
A inserção é uma "actualização" numa trajectória e implica:

1. A pesquisa na estrutura de dados que mantém as ultimas actualizações para cada trajectória, ou seja, procurar o triplo $(t_{i-1}, X_{i-1}, Y_{i-1})$ associado à trajectória;
2. A construção do novo segmento, com base no último e na nova posição;
3. A sua inserção no mecanismo *SETI*, ou seja na *R-Tree* (região do espaço), tal como na R^*-*Tree* (tempo).

No artigo (Chakka et al. 2003) o mecanismo *SETI* foi comparado com a *TB-Tree* e mostrou ter um melhor desempenho nas pesquisas das trajectórias, numa dada região num dado intervalo de tempo.

A.2.5 TrajStore

TrajStore é um mecanismo de armazenamento dinâmico que está optimizado para realizar eficientemente pesquisas numa região espacio-temporal. Este foi proposto em (Cudre-Mauroux et al. 2010).

As trajectórias são divididas em segmentos que são alocados nas regiões espacio--temporais definidas. A estrutura de dados é primariamente organizada de acordo com a indexação espacial e só aplicado o temporal aos elementos na região espacial. Para particionar o espaço pode-se usar uma *Quad-Tree*, onde cada célula corresponde a uma colecção de páginas em disco que contém os segmentos localizados nesta região. A cada uma destas é atribuído um índice temporal, que simplesmente guarda o intervalo de tempo que contém todos os segmentos para cada página. Para além disso, é mantido um mecanismo de indexação para cada trajectória que associa o conjunto de células que contém segmentos da trajectória. Na Ilustração A.5, retirada de (Cudre-Mauroux et al. 2010) é visível o mecanismo de armazenamento usado na *TrajStore*.

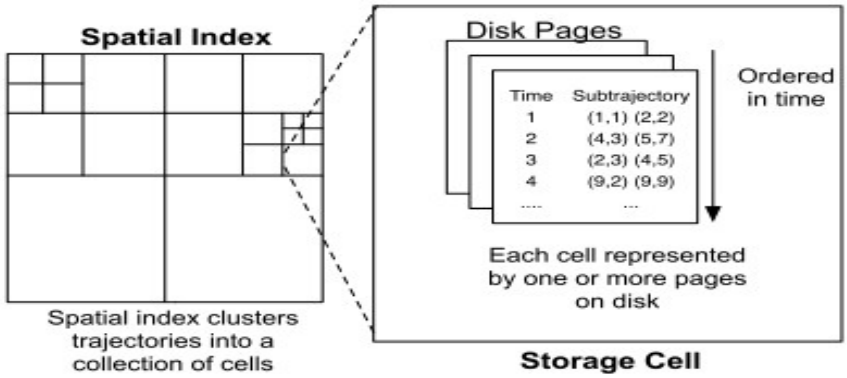

Ilustração A.5: Arquitectura da TrajStore

No artigo (Cudre-Mauroux et al. 2010) são apresentados alguns mecanismos para a escolha óptima do tamanho das células, baseados na densidade dos elementos de cada uma, nas actualizações e pesquisas efectuadas. Alguns foram testados e obtiveram bons resultados relativamente ao seu desempenho.

A.3 Parametrização da *RLC2*

A.3.1 Pesquisa do vizinho mais próximo (1NN)

A.3.1.1 Espaço métrico dos autocarros

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	38.33	47.67	63	7	4.97	13.67	11.33	180.53	437.33
L_2-norm	109	113.15	123.33	8	14	15	1	1.72	2.67

Tabela A.1: 1NN na RLC2 (autocarros com raio de 25% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	58.33	65.23	88.67	1	2.71	3.67	17	210.96	488
L_2-norm	81.33	105.45	138	6.67	17.14	19	0.67	1.72	3.33

Tabela A.2: 1NN na RLC2 (autocarros com raio de 50% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	70.67	77.23	108	1	2.46	3.33	20.67	219.87	481.67
L_2-norm	79	125.41	144.33	6	15.71	17.33	0.67	1.89	3

Tabela A.3: 1NN na RLC2 (autocarros com raio de 75% do valor médio)

A.3.1.2 Espaço métrico dos furacões

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	31.33	58.70	100.67	2.67	6.71	9.33	3	40.95	172.67
L_2-norm	25.33	48.67	79	3	7.22	9	0	0.23	1

Tabela A.4: 1NN na RLC2 (furacões com raio de 25% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	31	57.33	113.33	3.67	7.02	9.33	2.67	14.86	67
L_2-norm	26	45.38	98	3	6.29	9	0.25	0.25	1

Tabela A.5: 1NN na RLC2 (furacões com raio de 50% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	16.33	40.22	92.67	4	7.77	11.33	2	8.52	37
L_2-norm	8.33	23.85	100.67	3.33	8.61	11.67	0	0.26	1

Tabela A.6: 1NN na RLC2 (furacões com raio de 75% do valor médio)

A.3.2 Pesquisa dos cinco vizinhos mais próximos (5NN)

A.3.2.1 Espaço métrico dos autocarros

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	5.67	15.80	37.67	1.67	6.17	8	14.67	186.54	428.67
L_2-norm	108	110.80	120.67	9.67	14.72	18	1	1.87	3

Tabela A.7: 5NN na RLC2 (autocarros com raio de 25% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	3.67	43.59	88	1.67	3.25	4	18	207.09	441.67
L_2-norm	70	92.68	124.33	8.33	17.83	19	1	1.98	3

Tabela A.8: 5NN na RLC2 (autocarros com raio de 50% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	3.67	52.79	100.67	1	3.67	3.67	22.67	223.85	484.33
L_2-norm	50	123.90	143.67	7	17.33	17.33	1	2	1

Tabela A.9: 5NN na RLC2 (autocarros com raio de 75% do valor médio)

A.3.2.2 Espaço métrico dos furacões

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	31.33	57.51	97.67	3.33	7.05	9.33	2.67	11.28	44.67
L_2-norm	20	29.37	50.67	7.67	9.48	10	0	0.33	1

Tabela A.10: 5NN na RLC2 (furacões com raio de 25% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	11	31.33	71.67	3.67	6.61	7.67	2.67	7.59	30.67
L_2-norm	14	27.92	82.33	3	4.64	5.33	0	0.23	0.67

Tabela A.11: 5NN na RLC2 (furacões com raio de 50% do valor médio)

Funções de Semelhança	*SD*			*SR*			*ET*		
	Min	Média	Max	Min	Média	Max	Min	Média	Max
ERP	16.33	40.22	92.67	7	8.70	11.67	2.33	10.51	41
L_2-norm	8.33	23.85	100.67	7.33	9.86	11.67	0	0.29	1

Tabela A.12: 5NN na RLC2 (furacões com raio de 75% do valor médio)

A.3.3 Análise dos resultados experimentais

Analisando os valores das distâncias calculadas, das leituras em disco e dos tempos de execução, tendo como intuito minimiza-los em ambos os espaços métricos, é necessário inquirir qual o melhor valor a atribuir ao raio da *RLC2*.

Assim sendo, no espaço métrico dos autocarros, para ambas as funções (*ERP* e *L_2--norm*) e pesquisas (1NN e 5NN), o valor do raio da *RLC2*, que permite tal feito, é de 25% da distância média do espaço.

No espaço métrico dos furacões não existe uma concordância tão evidente pois, o valor do raio óptimo é diferente para cada função, nas pesquisas 1NN, sendo o seu valor de 25% e 75%, da distância média do espaço, para a *L_2-norm* e *ERP*, respectivamente. Já nas pesquisas de 5NN, o valor óptimo do raio é de 50%, da distância média do espaço, para ambas as funções.

A.4 Criação da RLC2

Nas tabelas A.13 e A.14 encontram-se os valores referentes ao número de distâncias calculadas, leituras e escritas efectuadas em disco, bem como o tempo médio da alocação de cada objecto, em segundos, a quando da criação das "instâncias" referentes aos espaços métricos de dados dos autocarros e furacões, respectivamente. De notar, que este valores são provenientes da média aritmética dos valores obtidos a quando da alocação das três permutações na *RLC2*.

Função de Semelhança	*Distâncias*	*Leituras*	*Escritas*	*Tempo*
ERP	496	130.33	139	5.95
L_2-norm	6700	24.67	39.67	0.38

Tabela A.13: Criação da RLC2 (autocarros)

Função de Semelhança	*Distâncias*	*Leituras*	*Escritas*	*Tempo*
ERP	942.67	176.67	186.67	0.81
L_2-norm	674.33	171	178.67	0.12

Tabela A.14: Criação da RLC2 (furacões)

Recordando as dimensões dos espaços métricos (Tabela 3.3), podemos ver que no espaço de maior dimensão (Autocarros + *ERP*), obtém-se um maior valor para a média de tempo que cada objecto precisa para ser alocado, no entanto é neste espaço que se obtém o menor valor para o cálculo de distâncias. Já o menor número de leituras e escritas é obtido no espaço métrico de menor dimensão (Autocarros + L_2-*norm*).

Curiosamente, o melhor tempo de execução é obtido num espaço métrico em que a sua dimensão é muito próximo de 0.5, ou seja, no espaço métrico criado com recurso aos dados dos furacões e à função de semelhança L_2-*norm*, onde se obteve o valor de 0.42.