Universidade Estadual de Feira de Santana

Programa de Pós-Graduação em Computação Aplicada

Encontrando os locais de interesse com maior concentração de objetos relevantes para um conjunto de palavras-chave

Claudio Moisés Valiense de Andrade

Feira de Santana

2019

Universidade Estadual de Feira de Santana

Programa de Pós-Graduação em Computação Aplicada

Claudio Moisés Valiense de Andrade

Encontrando os locais de interesse com maior concentração de objetos relevantes para um conjunto de palavras-chave

Dissertação apresentada à Universidade Estadual de Feira de Santana como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada.

Orientador: Prof. João B. Rocha-Junior

Feira de Santana

2019

Ficha Catalográfica - Biblioteca Central Julieta Carteado - UEFS

A566 Andrade, Cláudio Moisés Valiense de
Encontrando os locais de interesse com maior concentração de objetos relevantes para um conjunto de palavras-chave / Cláudio Moisés Valiense de Andrade. - 2019.
68 f.: il.

Orientador: João Batista da Rocha Junior
Dissertação (mestrado) - Universidade Estadual de Feira de Santana, Programa de Pós-Graduação em Computação Aplicada, Feira de Santana, 2019.

1. Locais de interesse. 2. Objetos de referência. 3. Banco de dados espaciais. 4. Consulta espaço-textual preferencial por popularidade (CETPP). 5. Algoritmos. I. Rocha Junior, João Batista da, orient. II. Universidade Estadual de Feira de Santana. III. Título

CDU: 004.78:528

Luis Ricardo Andrade da Silva - Bibliotecário - CRB-5/1790

Cláudio Moisés Valiense de Andrade

Encontrando os locais de interesse com maior concentração de objetos relevantes para um conjunto de palavras-chave

Dissertação apresentada à Universidade Estadual de Feira de Santana como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada.

Feira de Santana, 29 de março de 2019

BANCA EXAMINADORA

João Batista da Rocha Junior

Dr. João Batista da Rocha Junior (Orientador)
Universidade Estadual de Feira de Santana

Dr. José Amâncio Macedo Santos
Universidade Estadual de Feira de Santana

Carlos Alberto Rodrigues

Dr. Carlos Alberto Rodrigues
Universidade Estadual de Feira de Santana

Abstract

Spatial data is increasingly present in our daily lives. We use various applications that use this data, such as Google Maps and Uber. There are a huge number of interesting questions that can be performed on these data. For example, a tourist maybe interested in hotels that have a lot of restaurants in its vicinity. This project proposes a new query type named Popularity Based Spatio-Textual Preference Query (PSTPQ), whose main contribution, that can select the best spatial objects taking into account the number of relevant spatio-textual objects, for a given set of query keywords, in its neighborhood. We present algorithms to process this query efficiently and evaluate the algorithms proposed in real datasets. Our experiments show that it is more efficient to use spatial indices (e.g. R-Tree) for distances less than 5 km in relation to textual indexes (e.g. Inverted File). In our experiments, the hybrid index processed the PSTPQ query with better performance. The PSTPQ query has as a differential take into account the number of reference objects in the spatial neighborhood, in addition to selecting the reference objects from the textual description.

Keywords: Spatial Database, Spatio-Textual Query, Preference queries.

Resumo

Dados espaciais estão cada vez mais presentes em nosso dia a dia. Usamos diversas aplicações que utilizam esses dados, como o Google Maps e Uber. Há um grande número de perguntas interessantes que podem ser realizadas com base nestes dados. Por exemplo, um turista talvez esteja interessado em hotéis que têm muitos restaurantes na sua vizinhança. Este projeto propõe um novo tipo de consulta denominada Consulta Espaço-Textual Preferencial por Popularidade (CETPP), cuja principal contribuição, pode selecionar os objetos espaciais com maior escore levando em conta o número de objetos espaço-textuais relevantes, para um determinado conjunto de palavras-chave de consulta, em sua vizinhança. Apresentamos algoritmos para processar essa consulta de forma eficiente e avaliar os algoritmos propostos em conjuntos de dados reais. Nossos experimentos mostram que tem melhor desempenho utilizar índices espaciais (e.g. R-Tree) para distâncias menores de 5 km em relação a índices textuais (e.g. Inverted File). Em nossos experimentos, o índice híbrido processou com melhor desempenho a consulta *CETPP*. A consulta CETPP tem como diferencial levar em consideração a quantidade de objetos de referência na vizinhança espacial, além de selecionar os objetos de referência à partir da descrição textual.

Palavras-chave: Banco de Dados Espacial, Consulta Espaço-textual, Consulta preferencial.

Prefácio

Esta dissertação de mestrado foi submetida a Universidade Estadual de Feira de Santana (UEFS) como requisito parcial para obtenção do grau de Mestre em Computação Aplicada.

A dissertação foi desenvolvido dentro do Programa de Pós-Graduação em Computação Aplicada (PGCA) tendo como orientador o Dr. **João B. Rocha-Junior**.

Agradecimentos

Primeiramente agradeço a Deus por permitir ter saúde para estudar e realizar as atividades exigidas nesta pesquisa.

Agradeço a minha mãe Tânia e toda a minha família pelos devidos ensinamentos por toda a vida.

Agradeço ao meu orientador João B. Rocha-Junior pela paciência e ensinamentos que levarei por toda a minha vida.

Agradeço a todo o corpo docente do programa de pós-graduação de computação aplicada.

Agradeço a todos os amigos que fiz através do programa PGCA, em especial a Felipe, Reuder, Naan, Gabriel, Wanderson, Elivelton, Lucas, Jeziel, Reginaldo, Neto, Marcondes, Priscila...

Agradeço o grupo ADaM por me acolher como um membro da família.

Agradeço a minha companheira Gessica pelo comprometimento em resolver os problemas ao meu lado.

Agradeço a minha filha Eliza, que me incentiva em ser uma pessoa mais responsável e dedicada.

Sumário

Lista de Publicações

- Encontrando os melhores locais a partir da popularidade na vizinhança espacial: uma proposta. Claudio Moisés Valiense de Andrade, João B. Rocha-Junior. (2018). In WPOS/ERBASE.
- Encontrando os locais de interesse com maior popularidade a partir do critério espacial e textual. Claudio Moisés Valiense de Andrade, João B. Rocha-Junior. (2018). In Sistemas e Computação ISSN 2237-2903 [Valiense de Andrade e B Rocha-Junior 2018].

Lista de Tabelas

Lista de Figuras

Lista de Abrebiações

Abreviação	**Descrição**
CETPP	Consulta Espaço-Textual Preferencial Por Popularidade
P	Conjunto de objetos de interesse
F	Conjunto de objetos de referência
Q.x	Latitude do ponto da consulta
Q.y	Longitude do ponto da consulta
Q.D	Conjunto de palavras-chave
Q.r	Limiar do raio
Q.k	Número de resultados esperados
$Q.\sigma$	Limiar de similaridade
W_d	Peso do documento d
W_q	Peso da consulta q
$S_{q,d}$	Similaridade textual dos termos da consulta com o documento.
S2I	Spatial Inverted Index
NN	Nearest Neighbor
RNN	Reverse Nearest Neighbor
L1	Distância Manhattan
L2	Distância Euclidiana
B-Tree	Árvore B
R-Tree	Árvore de estrutura de dados espaciais
I/O	Acesso a entrada e saída de dados
φ	Função que converte o ângulo para radiano
θ	Função para o cálculo de similaridade textual

Capítulo 1

Introdução

"Não acredito em sorte, acredito em trabalho duro."
– Autor desconhecido

Com o passar do tempo as máquinas estão realizando uma quantidade maior de tarefas que auxiliam atividades do cotidiano, como exibir a rota entre um local de origem e destino. O *Waze*[1], por exemplo, informa sinalizações e buracos existentes em um trajeto. Para realizar essas tarefas, o *Waze* utiliza dados espaciais.

Com a popularidade de dispositivos móveis com *GPS* (Global Positioning System), a quantidade de dados produzidos com referências geográficas (latitude e longitude) tem crescido rapidamente. Um objeto espacial é um ponto que contém uma referência geográfica. Além da localização, outras informações podem estar associadas aos objetos (e.g., nome, tamanho, tipo, preço, mensagem) [Yiu et al. 2007]. Objetos que possuem localização espacial (referência geográfica) e texto são chamados de objetos espaço-textuais.

Existem diversos tipos de consultas capazes de selecionar dados espaciais, denominadas consultas espaciais [Zhang et al. 2006, Rocha-Junior et al. 2010, Yiu et al. 2011, Rocha-Junior et al. 2011, de Almeida e Rocha-Junior 2016, Gao et al. 2016]. Estas consultas permitem selecionar objetos espaciais de interesse, a partir de um raio ou vizinhança.

Uma consulta que vem sendo bastante estudada é a consulta espacial preferencial [Yiu et al. 2007, Yiu et al. 2011, Rocha-Junior et al. 2010, de Almeida e Rocha-Junior 2016, de Almeida e Durão 2018]. Esta consulta funciona da seguinte forma, dado um conjunto de objetos espaciais de interesse (*locais*) e um conjunto de objetos espaciais de referência, esta consulta retorna os k melhores

[1] `www.waze.com`

objetos de interesse, levando-se em consideração o maior escore entre os objetos de referência dentro da região espacial de interesse fornecida pelo usuário (e.g. 100m). Nesta consulta, cada objeto de referência tem um escore que é definido por um provedor de classificação específico (e.g. *ZAGAT*[2], *IFOOD*[3]).

Exemplo. A Figura 1.1, os objetos de interesse p (hotéis) são representados por círculo preto preenchido e os objetos de referência f (bares) por quadrados. O círculo ao redor dos objetos espaciais de interesse delimita a região espacial (raio). É assumido que um usuário deseja ficar hospedado em um hotel que tenha o bar com maior classificação, considerando o entorno do hotel (e.g. raio de $100m$*). Ao utilizar a consulta espacial preferencial para retornar os dois hotéis com maior escore (*k*=2), a consulta retorna os hotéis* p_1 *em primeiro e* p_3 *em segundo. O hotel* p_1 *é retornado em primeiro, porque no seu entorno tem um bar* f_1 *com escore* 0.9*, enquanto que o bar com maior escore no entorno de* p_3 *é* f_5 *com escore* 0.8*. Esta consulta considera apenas o objeto de referência com o maior escore dentro da região espacial de interesse.*

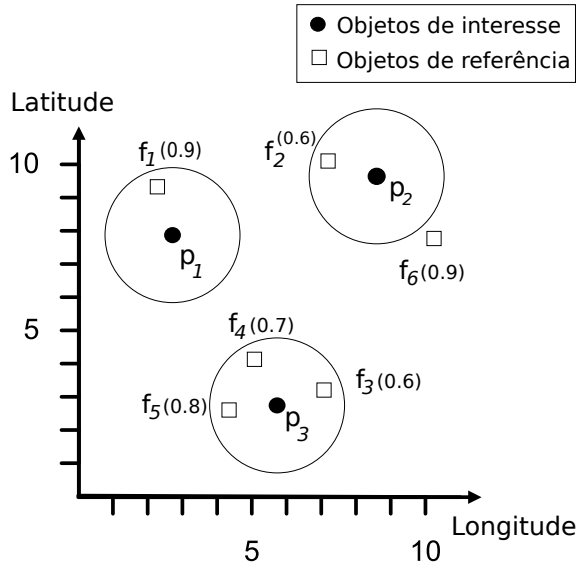


Figura 1.1: Representação hotéis e pontos de referência. Fonte: Próprio Autor.

A consulta espacial preferencial tradicional tem três problemas principais: 1) não leva em consideração a quantidade de objetos de referência na vizinhança espacial, apenas o objeto de referência com maior escore; 2) não é possível selecionar os objetos de referência à partir da descrição textual dos objetos e 3) a consulta espacial preferencial tradicional assume que o escore dos objetos de referência são números estáticos. Entretanto, para a maioria das aplicações, estes objetos estão associados a um texto descritivo. Neste caso, o escore de um objeto ou a sua relevância pode ser computada, levando-se em consideração a similaridade textual destes objetos

[2] `www.zagat.com`
[3] `www.ifood.com.br`

com palavras-chave de busca. Os k objetos de interesse com os maiores escores são retornados.

A consulta proposta nesta pesquisa é denominada de Consulta Espaço-Textual Preferencial Por Popularidade (CETPP). Diferente da consulta tradicional de Yiu et al. (2007) que retorna o objeto de interesse analisando o objeto de referência com maior escore no seu entorno, a *CETPP* leva em consideração todos os objetos espaço-textuais de referência que tem relevância textual maior que um limiar mínimo e que estão na região espacial de interesse definida pelo usuário. Na consulta *CETPP*, o escore do objeto de referência é calculado a partir da similaridade do texto do objeto de referência e do conjunto de palavras-chave. A partir disso, o escore dos objetos de interesse é calculado somando todos os objetos de referência cuja a relevância textual fique acima do limiar e que estejam presentes na região de interesse.

Até onde sabemos, nenhum dos trabalhos anteriores cobriu o problema de computar a pontuação dos objetos de interesse, levando em consideração o número de objetos espaço-textuais que são textualmente relevantes na vizinhança espacial de interesse. Em todos os trabalhos anteriores, o escore de um único objeto de referência define o escore dos objetos de interesse, ou seja, o objeto espacial de referência com a maior pontuação na vizinhança espacial de interesse. No entanto, para muitas aplicações reais, isso é uma limitação enorme. Por exemplo, um usuário pode estar interessado em uma rua (objeto espacial de interesse) onde há muitas lojas de sapatos (objetos espaço-textuais de referência) para que ela possa comparar os preços ou encontrar diferentes modelos de sapatos. Nesse caso, a rua com mais lojas de sapatos deve ter uma pontuação melhor do que uma rua com uma única loja.

Exemplo. A Figura 1.2 é composta por objetos espaço-textuais de interesse p (e.g. apartamentos) e objetos de referência f (e.g. objeto espaço-textual de qualquer tipo tal como restaurantes, hotéis, bares e escolas). Assumindo que um cliente deseja comprar um apartamento (objeto de interesse) que tenha muitas escolas (objetos de referência) na sua proximidade, ele pode então fornecer uma palavra-chave de interesse "escola", indicar a região que considera próxima (e.g. $1km$*) e definir um limiar de semelhança textual entre o texto do objeto de referência e as palavras-chaves de busca (e.g.* 0.5*). Ao realizar esta consulta na base de dados da Figura 1.2, ele tem como resposta o objeto p_3 com maior escore, visto que p_3 possui* 2 *objetos de referência f_5 e f_6 que satisfazem a condição espacial e são relevantes com a palavra-chave de busca "escola", seguido de p_2 que tem apenas* 1 *objeto de referência f_7 que satisfez a condição espacial e textual.*

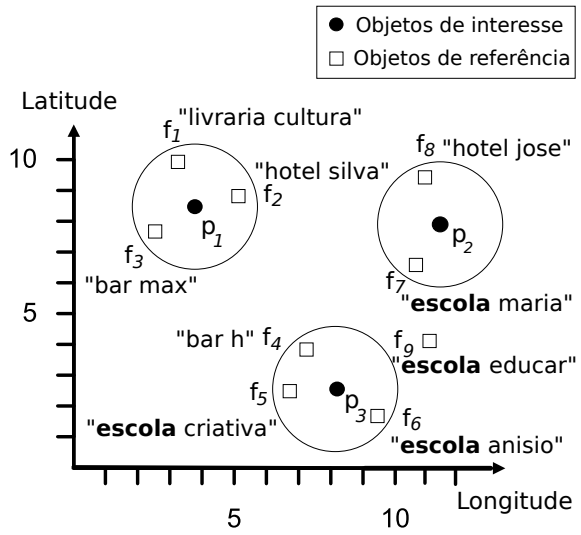


Figura 1.2: Objetos de interesse (apartamento) e objetos espaço-textuais de referência. Fonte: Próprio Autor.

1.1 Aplicações

Existem várias aplicações que podem se beneficiar desta nova consulta. A seguir são apresentados alguns exemplos.

1.1.1 Identificar Local com Maior Variedade de Serviços

A consulta *CETPP* beneficia aplicações que tem o objetivo de encontrar um local com uma maior variedade de produtos ou serviços.

Exemplo. Assumindo que um pai deseja comprar para sua filha um remédio difícil de ser encontrado. Para aumentar a chance dele encontrar o remédio, ele precisa localizar a estação de metrô (objeto de interesse) que tem o maior número de objetos espaço-textuais relevantes para as palavras-chave "farmácia", aumentando a chance de encontrar o remédio. Na Figura 1.3, o conjunto de objetos de interesse p, representa as estações de metrô. A consulta CETPP irá retornar p_2 como a estação de metrô mais relevante, porque tem o maior número de objetos de referência com o termo "farmácia".

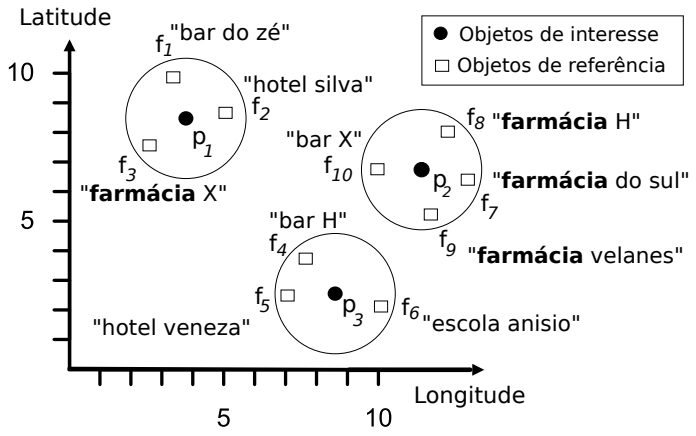


Figura 1.3: Objetos de interesse (estação de metrô) e objetos espaço-textuais de referência. Fonte: Próprio Autor.

1.1.2 Identificar o Local Ideal para um Novo Estabelecimento

Escolher o local ideal na abertura de um novo negócio é importante para aumentar a possibilidade de ter sucesso. É possível utilizar a consulta *CETPP* para auxiliar nessa tomada de decisão.

Exemplo. Um empreendedor deseja ter sucesso ao abrir sua loja de material escolar. Pensando nisso, ele deseja encontrar um local (objeto de interesse) que tenha uma maior quantidade de objetos espaço-textuais relevantes para a palavra-chave "escola" (objetos de referência).

A base de dados do *Twitter* também pode ser utilizada como objetos de referência. No *Twitter*, uma porcentagem das mensagens dos usuários enviadas pelo *smartphone* contém informações geográficas. Assim, é possível saber o local dos usuários cujo as postagens são mais relevantes para um conjunto de palavras-chave.

Exemplo. Um grupo de empresários pretende construir um shopping, é de interesse do grupo que seja construído no local (objetos de interesse) que atenda uma maior quantidade de clientes (objetos de referência), cujo as postagens sejam relevantes para a palavra-chave "shopping".

1.2 Objetivo Geral

Identificar os locais de interesse com maior concentração (popularidade) de objetos espaço-textuais relevantes para um conjunto de palavras-chave.

1.3 Objetivos Específicos

- Especificar a nova consulta "Espaço-Textual Preferencial Por Popularidade" (CETPP);
- Desenvolver algoritmos para processar essa nova consulta com um maior desempenho;
- Avaliar os algoritmos propostos utilizando bases de dados reais. Com o objetivo de simular consultas reais produzidas por usuários.

1.4 Publicações

Nesta seção são listadas todas as publicações originadas desta pesquisa de mestrado, acompanhadas por uma breve descrição.

- Encontrando os melhores locais a partir da popularidade na vizinhança espacial: uma proposta. Cláudio Moisés Valiense de Andrade, João B. Rocha-Junior. (2018). In WPOS/ERBASE, Sergipe, Aracaju, Agosto de 2018.

  O artigo foi apresentado no *Workshop* de Pós-Graduação da Escola Regional de Computação Bahia-Alagoas-Sergipe (ERBASE). Neste artigo foi apresentado a definição da consulta CETPP, dois algoritmos para processar esta consulta de forma eficaz e experimentos em bases de dados reais. Foram conduzidos experimentos para analisar o tempo de resposta. Durante o WPOS, o artigo foi analisado por uma banca de doutores que ofereceram sugestões para melhorias do trabalho. Este artigo foi escolhido como o melhor artigo do WPOS 2018, como premiação, foi convidado para publicação de um artigo estendido na revista de Sistemas e Computação.

- Encontrando os locais de interesse com maior popularidade a partir do critério espacial e textual. Claudio Moisés Valiense de Andrade, João B. Rocha-Junior. (2018). In Sistemas e Computação ISSN 2237-2903 [Valiense de Andrade e B Rocha-Junior 2018].

1.5 Organização do Trabalho

O restante do documento está divido da seguinte forma, no Capítulo 2 é apresentada a fundamentação teórica. No Capítulo 3 são apresentados os trabalhos relacionados. No Capítulo 4 é apresentada a metodologia aplicada. No Capítulo 5 são apresentados os algoritmos propostos para processar a consulta. No Capítulo 6 é avaliado os resultados dessa pesquisa. No Capítulo 7 as considerações finais deste trabalho.

Capítulo 2

Fundamentação Teórica

"Se eu vi mais longe, foi por estar de pé sobre ombros de gigantes."
– Isaac Newton

Neste capítulo está presente a fundamentação teórica desta pesquisa. Seção 2.1 contém conceitos relacionados à consultas textuais. Seção 2.2 contém consultas espaciais, métricas de distância e índices. Seção 2.3 contém consultas espaço-textuais.

2.1 Consultas Textuais

Quando utilizamos uma máquina de busca como o *Google*[1], estamos realizando uma consulta textual. Os mecanismos de pesquisa são ferramentas para encontrar os documentos em uma coleção que correspondam às consultas dos usuários [Zobel e Moffat 2006].

Exemplo. A partir de uma base textual, como a base apresentada na Figura 2.1, é possível encontrar os itens (documentos) mais relevantes para um conjunto de palavras-chave (consulta textual). Assim, um usuário interessado em hospital, pode utilizar o termo "hospital" como palavra-chave de busca, obtendo como resposta os documento 2 *e* 6. *Estes itens tem uma maior relevância textual.*

[1] `www.google.com`

1. Academia da mulher
2. Hospital do câncer
3. Padaria Moura
4. Bar do Zé
5. Escola da criança
6. Hospital Manoel Novais
7. Escola criativa

Figura 2.1: Uma base de dados textual. Fonte: Adaptada [Zobel e Moffat 2006].

Existem atualmente vários modelos para avaliar a relevância textual entre as palavras-chave de busca e o texto dos documentos (e.g. modelo booleano, probabilístico, vetorial) [Robertson e Jones 1976, Salton 1973]. Um destes modelos é o modelo vetorial, que determina o grau de relevância de cada documento a partir da similaridade com as palavras-chave de busca. Este modelo representa documentos e palavras-chave como vetores. A relevância de um documento é dada pela similaridade entre os termos que descrevem o documento e as palavras-chave de busca. Aos termos das consultas e documentos são atribuídos pesos que especificam o tamanho e a direção de seu vetor de representação. A similaridade é baseada no *cosseno* entre os vetores que representam o documento e a consulta [Salton e Buckley 1988].

Zobel et al. (2006) utiliza as equações abaixo para calcular a similaridade textual entre o texto presente nos documentos e os termos de buscas. A Equação 2.1 corresponde ao peso do termo da consulta em relação a coleção, onde N corresponde ao total de documentos e f_t é o número de documentos que contém o termo. A Equação 2.2 representa o peso do termo no documento, onde $f_{D,t}$ corresponde a frequência do termo no documento. A Equação 2.3 representa o peso do documento: é constituído pela raiz quadrada do somatório do peso de todos os termos t que aparecem no documento D. A Equação 2.4 representa o peso dos termos da consulta q, é constituído pela raiz quadrada do somatório dos pesos dos termos presente na consulta q. A Equação 2.5, corresponde ao valor de similaridade textual do texto do documento com as palavras-chave da consulta. O intervalo de similaridade textual varia entre 0 e 1, quanto mais próximo de 1, o documento é mais similar as palavras-chave da consulta.

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \tag{2.1}$$

$$w_{D,t} = 1 + \ln f_{D,t} \tag{2.2}$$

$$W_D = \sqrt{\sum_t w_{D,t}^2} \tag{2.3}$$

$$W_q = \sqrt{\sum_t w_{q,t}^2} \tag{2.4}$$

$$S_{q,D} = \frac{\sum_t W_{D,t} \cdot W_{q,t}}{W_D \cdot W_q} \tag{2.5}$$

A partir do valor da similaridade de um documento com as palavras-chave ($S_{q,d}$), é possível utilizar um limite mínimo de relevância textual para auxiliar na filtragem de resultados, eliminando os itens com pouca ou nenhuma relevância do conjunto resposta.

Índices textuais como Arquivos Invertidos (Inverted Files) [Zobel e Moffat 2006] são utilizados para processar consultas textuais de forma eficiente. No arquivo invertido, o mapeamento de cada termo está relacionado a lista de documentos ao qual o termo é encontrado e a sua frequência, a partir de um termo, é possível encontrar rapidamente todos os documentos que contém ao menos uma ocorrência do termo.

Exemplo. A Figura 2.2 exemplifica um arquivo invertido criado a partir do pré-processamento da lista de documentos da Figura 2.1. Nesta figura, o $Termo_{(t)}$ *representa o termo presente no vocabulário,* $Documento_{(id)}$ *o identificador do documento,* f_t *a frequência do termo na coleção,* $f_{t,D}$ *a frequência do termo no documento. Analisando o termo "academia", sua frequência na coleção tem valor* 1*, por esta contido apenas no primeiro documento. Analisando o termo "hospital", sua frequência é* 2*, este termo está presente no documento* 2 *e* 6 *[Zobel e Moffat 2006].*

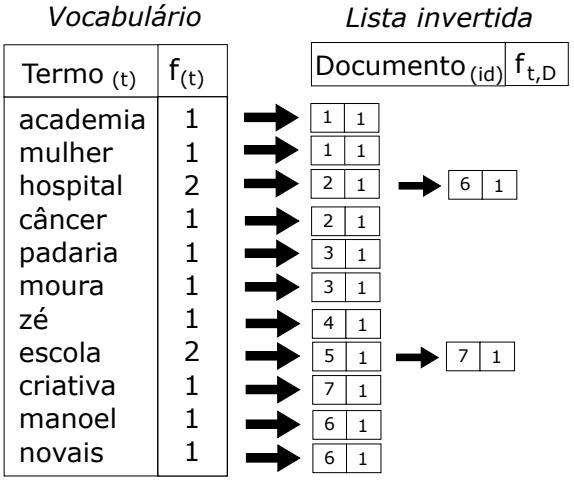


Figura 2.2: Exemplo de arquivo invertido. Fonte: Próprio autor.

2.2 Consultas Espaciais

A principais consultas espaciais são do tipo de seleção ou range. Esta consulta retorna um conjunto de objetos espaciais que satisfazem um critério definido na consulta. Por exemplo, "buscar todas as escolas em um raio de 100 metros da minha localização".

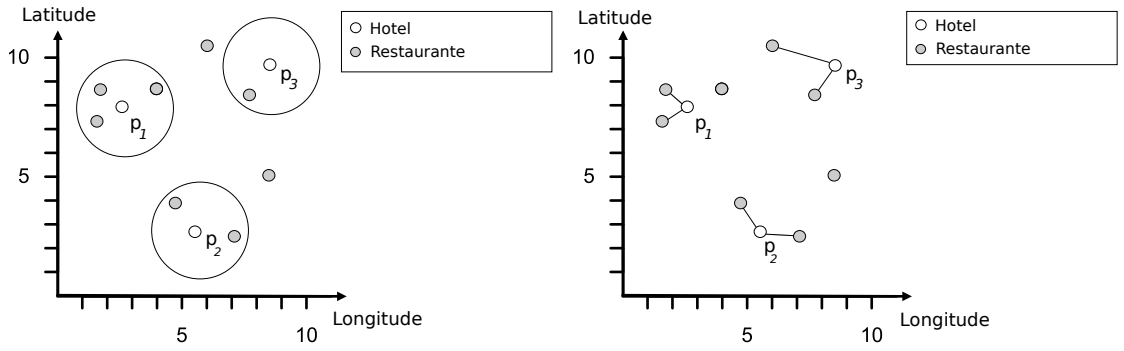
Uma consulta espacial bastante referenciada na literatura é a *Nearest Neighbor* [Korn e Muthukrishnan 2000]. Este consulta retorna o objeto espacial mais próximo de um dado local q. Uma variante desta consulta é a *kNN* que retorna os k objetos mais próximos em ordem crescente de distância. A Equação (2.6) é uma definição da consulta espacial *Nearest Neighbor*, onde S é o conjunto de pontos no plano, q o ponto de consulta, d a função de distância, p os pontos que estão sendo analisados dentro de S, e m os pontos com maior proximidade de q.

$$NN(q) = \{m \in S | \forall p \in S : d(q, m) \leq d(q, p)\} \tag{2.6}$$

Outra abordagem que pode ser aplicada, a utilização da consulta de seleção por *Range*, dado um ponto de consulta q, e uma distância r, esta consulta retorna todos os pontos p cujas as distâncias são menores ou iguais a r. Da mesma forma que na Equação 2.6, considere S como o conjunto de pontos no plano. A Equação 2.7 representa essa consulta.

$$Range(q) = \{p \in S : d(p, q) \leq r\} \tag{2.7}$$

Exemplo. Dado um conjunto de objetos de interesse p (hotéis) e o conjunto de objetos de referência (restaurante). A Figura 2.3(a) exemplifica uma consulta por range, onde cada p tem o raio de 0.2km. A Figura 2.3(b) apresenta uma consulta por knn, onde k=2. Note que dependendo da consulta espacial utilizada, range (Figura 2.3(a)) ou NN (Figura 2.3(b)), os elementos analisados são diferentes, como consequência obtendo resultados diferentes. Por exemplo, na Figura 2.3(a), p_1 *possui* 3 *restaurantes que atendem o limite espacial, enquanto que na Figura 2.3(b),* p_1 *possui apenas* 2 *restaurantes que atendem ao critério espacial.*



(a) Consulta por *range*. (b) Consulta por vizinho mais próximo (2NN).

Figura 2.3: Consultas espaciais. Fonte: Adaptado [Yiu et al. 2007].

2.2.1 Distância Espacial

As métricas de distância servem para computar a diferença entre pontos no espaço. Para as Equações (2.8), (2.9) e (2.10), x_1 representa a latitude e y_1 a longitude do ponto 1, enquanto que de forma similar x_2 e y_2 do ponto 2.

A métrica *Manhattan*(L1) utiliza quadrados para delimitar o raio até o vizinho mais próximo [Korn e Muthukrishnan 2000]. A Equação (2.8) define a métrica *Manhattan*:

$$distManhattan = |x_1 - x_2| + |y_1 - y_2| \tag{2.8}$$

A distância *Euclidiana* [Allman 1889], computa a distância em linha reta entre dois pontos. A Equação (2.9) representa a distância Euclidiana bidimensional:

$$distEuclidiana = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2.9}$$

Por razão da curvatura do planeta Terra, a distância *Havesine* tem maior precisão em relação a *Euclidiana*, por considerar o ângulo da curvatura da Terra [Sinnott 1984]. A Equação (2.10) apresenta a definição matemática da distância de *Haversine*:

$$\begin{aligned} a = \sin^2(\frac{\varphi(x_2 - x_1)}{2}) + \cos(\varphi(x_1))\cos(\varphi(x_2))\sin^2(\frac{\varphi(y_2 - y_1)}{2}) \\ c = 2 * atan2(\sqrt{a}, \sqrt{1-a}) \\ distHaversine = R * c \end{aligned} \tag{2.10}$$

Na Equação (2.10), R representa o raio da Terra (média = 6,371km), φ a função que converte o ângulo para radiano e *atan2* é a função para calcular o arco tangente do intervalo $\sqrt{a}, \sqrt{1-a}$. Note que o ângulo da latitude e longitude para esta equação estão em radiano.

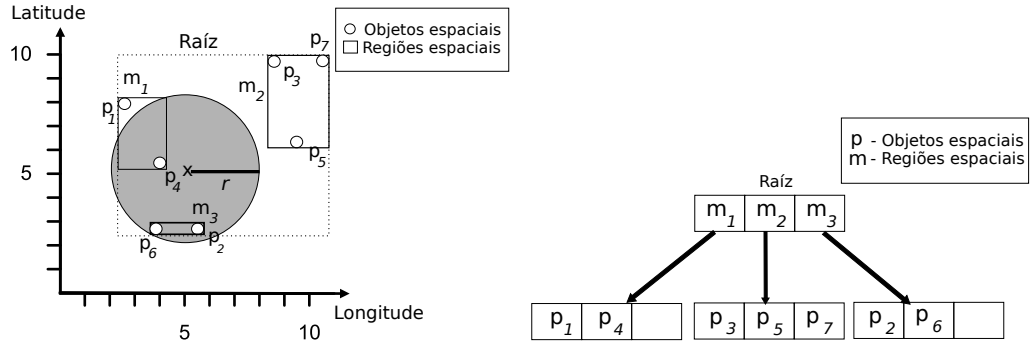
2.2.2 Índices Espaciais

Os índices espaciais permitem realizar consultas em grandes bases de dados espaciais de forma eficiente. Esta Seção contém a descrição de dois índices espaciais bastantes referenciados na literatura: *R-Tree* e *Aggregated R-tree*.

R-tree

Para trabalhar com dados espaciais como localizações dos objetos que contém por exemplo, latitude e longitude, pode ser utilizado o índice *R-tree* [Guttman 1984]. Este índice é similar a árvore B [Bayer e Mccreight 1972] com relação aos blocos e balanceamento, porém apresenta características diferentes. Os *nós* folhas da árvore são os objetos espaciais que contém as coordenadas (X, Y), e os *nós* não folhas representam as regiões. A região é o delimitador de menor retângulo (*MBR*) que engloba todos os objetos da suas folhas.

Exemplo. A Figura 2.4 exemplifica uma R-tree. Na Figura 2.4(a) é apresentada uma consulta espacial, a região m1 engloba os objetos espaciais p1 e p4, m2 contém p3, p5 e p7, enquanto que m3 contém p2 e p6. Onde x é o ponto de interesse da consulta do usuário com a condição que a distância seja menor que o raio r. As MBRs que satisfazem a condição são m1 e m3, sendo que m3 contém p2 e p6, estes objetos satisfazem a condição da consulta de r, m1 contém p1 e p4, onde apenas p4 satisfaz a condição da consulta de r. Portanto, nesta consulta são retornados os objetos espaciais p2, p4 e p6. Na Figura 2.4(b), é representado como a R-tree armazena esses dados espaciais, note que a região m2 não é visitada porque não faz interseção com o ponto de consulta x, portanto, são analisados apenas as regiões m1 e m3, por não analisar a sub-árvore de m2, a consulta tem uma maior eficiência.



(a) Consulta espacial em uma R-tree. (b) Estrutura de uma R-tree.

Figura 2.4: Exemplificando uma R-tree. Fonte: Próprio autor.

Aggregate R-tree

A *Aggregate R-tree* (aR-tree) [Papadias et al. 2001] é uma variante da *R-tree*. A principal diferença em relação a *R-tree* é que a *aR-tree* armazena informações extras nos *nós* que representam uma região, por exemplo, a contagem de quantos objetos existem na sua sub-árvore. Na Figura 2.5 é adicionado no *nó* a informação de quantos objetos tem na região, esta é a diferença em relação a Figura 2.4(b). Muitas consultas podem se beneficiar dessa informação por não precisar percorrer todos os objetos para realizar a contagem.

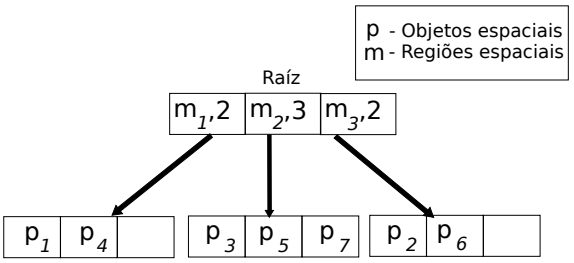


Figura 2.5: Estrutura de uma aR-tree. Fonte: Próprio autor.

2.3 Consultas Espaço-Textuais

A consulta espaço textual por palavras-chave é definida por [Chen et al. 2006, Ganeshan et al. 2010]. Esta consulta consiste em utilizar critérios espaciais e textuais. Chen et al. (2006) utiliza a estrutura de arquivos invertidos e R-Tree para processar a consulta. De Felipe et al. (2008) cria o índice espaço textual IR^2-Tree para processar essa consulta de forma eficiente. Estas pesquisas utilizam o conceito de *Boolean Keyword Query*, que significa que para um objeto ser considerado relevante, é preciso que contenha em seu texto todas as palavras-chave de busca [Ganeshan et al. 2010, Wu et al. 2012b].

2.3.1 Índices Espaço-Textuais

Índices espaço-textuais são índices híbridos que podem ser utilizados para realizar consultas eficientes em bases de dados contendo objetos espaço-textuais, estes índices utilizam dados espaciais e textuais dos objetos simultaneamente. Algumas pesquisas estão relacionadas a realizar processamento eficiente de índices híbridos [Chen et al. 2013, Cong et al. 2009, Rocha-Junior et al. 2011]. Este tipo de índice utiliza métodos de acesso espacial como *R-tree* [Guttman 1984] combinados com métodos de acesso textuais, como Arquivos Invertidos [Zobel e Moffat 2006]. Estes índices podem ser aplicados para restringir o conjunto de objetos a partir do escore textual ou a distância espacial. A seguir alguns índices espaço-textuais.

IR-Tree

Inverted File R-tree (IR-tree) desenvolvido por [Cong et al. 2009, Li et al. 2011], é uma estrutura híbrida que combina a *R-tree* com o arquivo invertido. Cada *nó* intermediário da árvore possui um arquivo invertido correspondente aos documentos indexados na sub-árvore do *nó*. A Figura 2.6 exemplifica uma *IR-tree*, a região *m4* por ser a raiz da árvore contém todos os objetos espaciais, portanto *IF 1* contém todos os termos de todos os objetos espaciais, enquanto que *IF 4* contém apenas os termos da região *m3*.

Exemplo. Assumindo que um usuário deseja encontrar um hospital que esteja no máximo 100 *metros da sua localização atual. Suponha que apenas a região* m_3 *satisfaça a condição espacial dos* 100 *metros, então, utilizando a IR-tree será necessário analisar apenas o arquivo invertido IF 4 buscando os objetos que contenha o termo "hospital".*

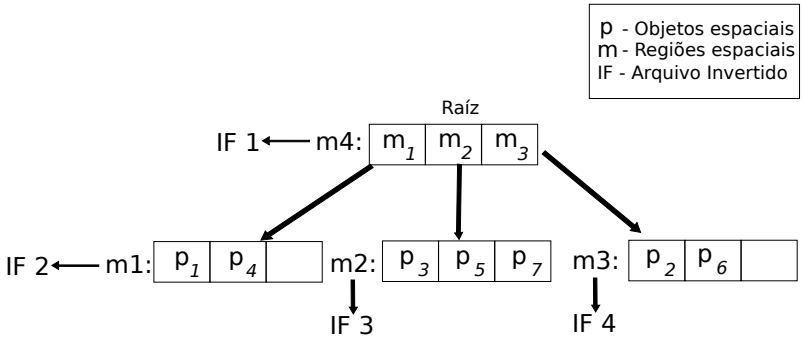


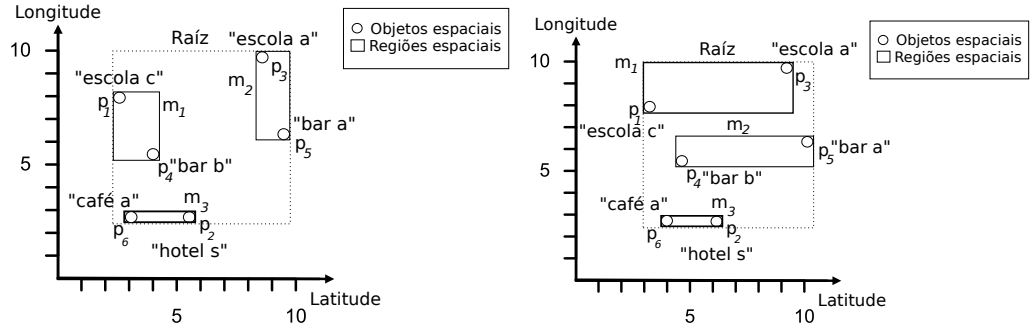
Figura 2.6: Estrutura de uma IR-tree. Fonte: Adaptado de [Cong et al. 2009].

Dir-tree

Outro índice espaço-textual é o *Document similarity enhanced Inverted file R-tree* (Dir-Tree) também criado por [Cong et al. 2009], a diferença em relação a *IR-tree* esta na criação das *MBR*, na *Dir-tree* as *MBRs* são criadas com base na similaridade textual dos documentos. Na Figura 2.7(a), a distribuição dos *MBR* está sendo feita apenas com base na localização espacial do objetos. Note que a região *m1* contém o objeto *p1* com o texto "escola c" e *p4* com texto "bar b". Na Figura 2.7(b) a formação do *MBR* leva em consideração a similaridade textual dos objetos espaciais, a região *m1* contém objetos com o termo "escola" e *m2* com o termo "bar".

S2I

Spatial Inverted Index (S2I) criado por [Rocha-Junior et al. 2011], é um índice que utiliza a frequência de um termo do arquivo invertido para determinar se armazena



(a) MBR com R-tree. (b) MBR com DIR-tree.

Figura 2.7: Diferença entre MBRs presentes nos índices R-tree e DIR-tree. Fonte: Próprio autor.

na forma de *aR-tree* ou em blocos. Caso um termo tenha uma frequência alta, a lista de objetos que contém aquele termo é armazenado em uma *aR-tree*, caso não seja tão frequente é armazenado em bloco de arquivo. Para determinar se um item é frequente ou não, é criado um arquivo em bloco que vai sendo inserido até preencher o bloco, caso o bloco chegue a capacidade máxima, o termo é considerado frequente e é armazenado em uma *aR-tree*. A Figura 2.8 exemplifica o índice *S2I*.

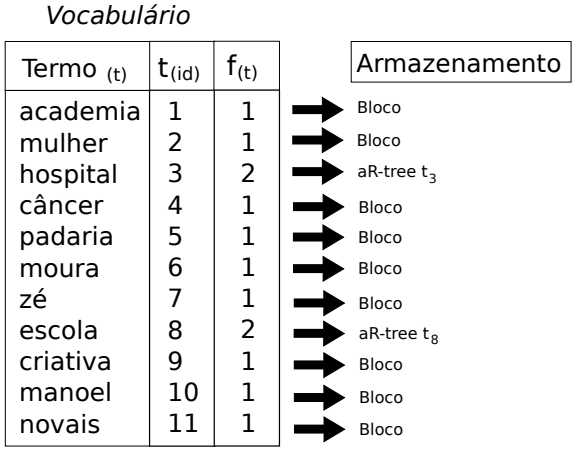


Figura 2.8: Armazenamento através do S2I. Fonte: Adaptado de [Rocha-Junior et al. 2011].

Capítulo 3

Trabalhos Relacionados

"Nunca tenha certeza de nada, porque a sabedoria começa com a dúvida."

– Sigmund Freud

Definido por [Du et al. 2005, Xia et al. 2005], a *Maximum Influence Query* busca o ponto de máxima influência ao adicionar um novo objeto de interesse (candidato). Este novo objeto de interesse precisa atrair a maior quantidade de objetos de referência. Um objeto de referência é atraído por um objeto de interesse se a distância entre os dois objetos é menor que a distância para qualquer outro objeto existente. A consulta é definida pela Equação (3.1).

$$p = argmax_{p \in P} \sum_{c \in C_p} w(c), \tag{3.1}$$

onde $C_p = \{c | c \in C \wedge \forall p' \in P, d(c,p) \leq d(c,p')\}$.

Nesta definição, C é o conjunto que representa os clientes(objetos de referência), p o local candidato (objeto de interesse) que esta sendo calculado o escore, p' o local candidato diferente de p, e w o peso do objeto (cliente).

Exemplo. Assumindo que a empresa McDonald's tem interesse em abrir uma nova unidade, na Figura 3.1 os objetos espaciais c representam as casas de clientes, enquanto que p são os locais possíveis para instalação da nova unidade do McDonald's. Dada esta distribuição dos objetos c e p, o local que atende a maior quantidade de clientes é o p_1*. Este é o local de máxima influência.*

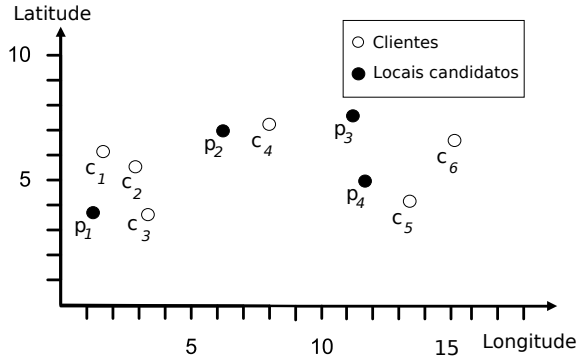


Figura 3.1: Exemplo de localização de máxima influência. Fonte: Adaptada [Xia et al. 2005].

De forma análoga a consulta proposta neste trabalho, a consulta de *Maximum Influence Query* utiliza a contagem dos objetos de referência para determinar o escore do objeto de interesse. Diferente desta consulta, a *CETPP* tem duas diferenças principais: 1) nós consideramos uma área de vizinhança em torno dos objetos de interesse definida pelo raio, enquanto que a *Maximum Influence Query* usa o conceito de atração (não importa a distância) e 2) a *Maximum Influence Query*, o escore dos objetos de referência é um número predefinido, enquanto que em nossa consulta nós os computamos dinamicamente usando a similaridade textual entre o texto dos objetos de referência e as palavras-chave de consulta.

Wu et al. (2012) definem dois tipos de consultas preferencias, uma que leva em consideração a localização do usuário e outra a região de interesse: *Location-aware top-k Text retrieval (LkT)*, dado um ponto de uma localização espacial, são retornados os k objetos de interesse ranqueados de acordo com a função de distância e relevância textual; *Region-aware top-k Text retrieval (RkT)*, dado uma região R, são retornados os k objetos que estão dentro do limite espacial da região e são ranqueados pela sua relevância textual com as palavras-chave [Wu et al. 2012a]. A consulta preferencial *LkT* tem similaridade com a consulta *CETPP*, porque é utilizada uma função de distância e relevância textual para determinar se o objeto de referência é relevante. Entretanto, diferente da consulta *LkT*, a principal diferença em relação a *CETPP* é que não temos definido apenas um ponto de consulta. Cada objeto de interesse é um ponto de consulta, além de que o escore de um objeto de interesse é definido de acordo com o número de objetos de referência que atendem os critérios espacial e textual.

Há outro tipo de consulta de seleção de local, com o objetivo de encontrar regiões espaciais baseadas em uma determinada função de classificação, como o peso do vizinho mais próximo. Dada uma base de dados de objetos de interesse e um retângulo de tamanho r, um dos objetivos da pesquisa de Choi et al. (2012) é encontrar o retângulo r que tem a maior soma dos pesos dos objetos de referência [Choi et al. 2012]. De forma similar a consulta *CETPP*, Choi et al. está utilizando uma função de classificação e utiliza os pesos de objetos de referência para determinar o escore

do objeto de interesse. Caso seja definido o mesmo peso para todos os objetos de referência, será similar ao cálculo de escore utilizado pela *CETPP*. As diferenças principais em relação a consulta *CETPP*: 1) a utilização de cada objeto de interesse como centro do raio da consulta definido pelo usuário e 2) determinar se um objeto de referência é relevante utilizando o critério de similaridade textual.

Com o mesmo objetivo de Choi et al. (2012), Cao et al. (2014) busca encontrar o retângulo que tenha a maior soma dos pesos dos objetos de interesse. Cao et al. (2014) utiliza palavras-chave para definir peso em objetos textualmente relevantes e além disso, utiliza a estrutura de estradas de redes no cálculo da distância espacial [Cao et al. 2014]. De forma similar ao objetivo do nosso trabalho, Cao et al. (2014) está buscando a região mais popular que atenda aos critérios espacial e textual. A *CETPP* difere em alguns ponto como: 1) cálculo do escore do objeto de interesse é baseado na contagem dos objetos de referência que atendem ao critério espacial e textual; 2) ao invés de considerar um retângulo fixo, a consulta proposta neste trabalho utiliza cada objeto de interesse como centro do raio da consulta, analisando a vizinhança espacial desses objetos; 3) utiliza um limiar de relevância textual para definir se o objeto de referência é relevante; 4) nos trabalhos de Choi et al. (2012) e Cao et al. (2014), para encontrar o retângulo mais popular, é preciso analisar todas as possibilidade de r, o que transforma este problema com a complexidade *NP-Completo*.

Algumas consultas têm o objetivo de relacionar dados espaciais e temporais, Cho e Chung (2007) propuseram uma consulta espaço temporal para responder questões como, "Qual é o número total de acidentes no raio de 1km de cada escola em Junho de 2005?" [Cho e Chung 2007]. As principais diferenças em relação a consulta *CETPP*: 1) utiliza palavras-chave para verificar a relevância textual dos objetos de referência e 2) determinamos um limiar de similaridade textual para considerar o objeto de referência como relevante.

Em *Preference-Aware Top-k Spatio-Texutal Query* [Gao et al. 2016] computa o escore de um objeto de interesse usando dado espacial, texutal e informações extra (e.g. classificação de um hotel). Além disso, ele também emprega uma limiar de similaridade textual para considerar a relevância dos objetos de referência. Diferente de *Preference-Aware Top-k Spatio-Texutal Query*, a consulta *CETPP* calcula a pontuação do objeto de interesse com base no número de objetos de referência que atendem aos critérios espaciais e textuais, o que é mais um desafio, pois mais objetos de referência textuais espaciais devem ser verificados para encontrar o resultado final.

Na consulta proposta por de Almeida e Rocha-Junior (2016) [de Almeida e Rocha-Junior 2016], o peso dos objetos de referência espaço-textuais é calculado dinamicamente com base nas palavras-chave de consulta colocadas pelo usuário. O processamento desta consulta é mais desafiador do que processar a Consulta de Preferência Espacial tradicional [Yiu et al. 2007, Yiu et al. 2011], porque as palavras-chave de consulta alteram o peso dos objetos de referência, que

são calculados com base na similaridade textual entre as palavras-chave de consulta e o texto dos objetos de referência.

Exemplo. Na Figura 3.2, supondo que um usuário está interessado em alugar um apartamento próximo a um objeto relevante para as palavras-chave "escola" e "infantil", e indicar a região que considera próxima (e.g. 1km). Ao realizar esta consulta na base de dados da Figura 3.2, ele tem como resposta o objeto p_2 e p_3, sendo que p_3 é o objeto mais relevante, pois possui na sua vizinhança espacial f_4 que é mais relevante para suas palavras-chave de busca.

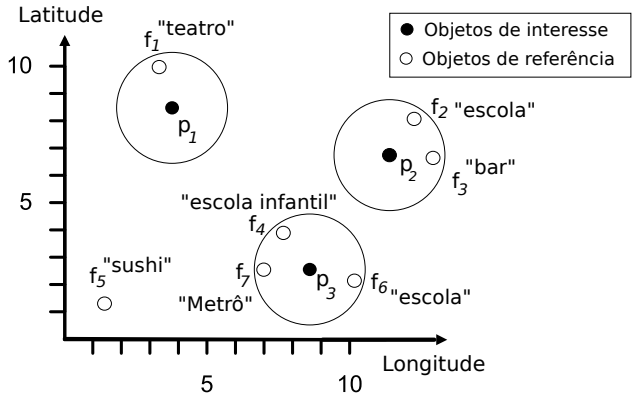


Figura 3.2: Objetos de interesse (apartamento) e objetos espaço-textuais de referência. Fonte: Próprio Autor.

Assim como a consulta espacial preferencial tradicional, a consulta proposta por [de Almeida e Rocha-Junior 2016] também só considera o objeto de referência com o maior escore. Nesse caso, o escore do objeto de referência é computado, levando em consideração a similaridade textual entre o texto dos objetos de referência e as palavras-chave da consulta. Diferente desta consulta, a consulta *CETPP* utiliza um limiar de relevância textual e um critério de vizinhança espacial para selecionar os objetos de referência e computa o escore dos objetos de interesse de acordo com o número de objetos de referência que atendem os dois critérios. A pesquisa desenvolvida por Almeida et al. [de Almeida e Rocha-Junior 2016] foi continuada e publicada em eventos de área, recentemente publicado em *Webmedia* 2018 [de Almeida e Durão 2018], que mostra que a pesquisa nesta área continua ativa.

Capítulo 4

Metodologia

"Prova-se tudo o que se quer, e a verdadeira dificuldade está em saber o que se quer provar."

– Émile-Auguste Chartier

A metodologia deste trabalho é dividida nas seguintes etapas: i) coletar bases de dados; ii) especificar consulta; iii) desenvolver algoritmos; iv) avaliar resultados; v) publicar resultados.

4.1 Coletar Bases de Dados

Neste trabalho são utilizadas bases de dados reais, com o objetivo de avaliar os algoritmos desenvolvidos. Este projeto utiliza dados de duas ferramentas online: o *Quot*[1] e o *Open Street Maps*[2] com o propósito de simular consultas de usuários reais.

Uma das bases de dados utilizada nessa pesquisa, é a base *Quot*, que disponibiliza os dados de imóveis para compra/venda ou aluguel. O *Quot* é um sistema web de apoio à tomada de decisão, na área de imóveis. Uma das funcionalidades do sistema é avaliar o preço de imóveis, levando diversos fatores para o calcular o preço, por exemplo, a localização geográfica do imóvel. Foram disponibilizados dados de imóveis referentes as cidades de Recife, Belo Horizonte, Rio de Janeiro e São Paulo. Nesta pesquisa utilizamos essa base para representar objetos de interesse.

Outra base de dados obtida nessa pesquisa foi através do sistema *Open Street Maps* (OSM), um projeto de mapeamento para representar objetos espaciais. O *OSM* é

[1] `http://www.quot.com.br/`

[2] `https://www.openstreetmap.org`

construído de forma colaborativa com dados abertos sob a licença *ODbL*[3]. É possível exportar informação de uma área de interesse, por exemplo, informações de objetos espaciais como restaurantes, bares, hotéis, etc. Os objetos desta base podem ser utilizados na forma de objetos de interesse ou de referência, mas nos experimentos realizados neste trabalho utilizamos como objetos de referência.

4.2 Especificar a Consulta

Esta seção consiste em descreve de forma precisa os parâmetros e resultados retornados pela consulta. Dado um conjunto de objetos de interesse P, onde cada objeto $p \in P$ possui uma coordenada espacial $p = (p.x, p.y)$; e um conjunto de objetos espaço-textuais de referência F, onde cada objeto $f \in F$ possui uma coordenada espacial $(f.x, f.y)$ e um texto $f.D, f = (f.x, f.y, f.D)$.

A consulta *CETPP* Q tem 6 parâmetros, $Q = \{Q.x, Q.y, Q.D, Q.r, Q.k, Q.\sigma\}$, onde $Q.x$ e $Q.y$ são os pontos de longitude e latitude do objeto de interesse, $Q.D$ é o conjunto de palavras-chave de interesse, $Q.r$ é o raio que delimita a vizinhança espacial de interesse, $Q.k$ é o número de resultados esperados e $Q.\sigma$ é o limiar que define o valor mínimo de similaridade textual entre o texto da consulta $Q.D$ e o texto dos objetos de referência $f.D$ para que um objeto de referência seja considerado textualmente relevante.

A consulta Q retorna o $Q.k$ objetos em P com os maiores escores. O escore de um objeto p, representado por $\tau(p)$, é a quantidade dos objetos de referência $f \in F$ presentes na vizinhança espacial de interesse p e que são textualmente relevantes para as palavras-chave de busca. Equação (4.1) apresenta a equação para o obter o escore:

$$\tau(p) = \sum \{f \in F \mid dist(p, f) \leq Q.r : \theta(f.D, Q.D) \geq Q.\sigma\} \tag{4.1}$$

onde $\theta(f.D, Q.D)$ é a relevância textual (similaridade textual) entre o texto do objeto espaço-textual de referência $f.D$ e a palavras-chave de consulta $Q.D$. Nesta pesquisa, nós computamos a relevância textual usando o cosseno [Zobel e Moffat 2006], enquanto que a distância espacial $dist(p, f)$ entre um objeto p e um objeto espaço-textual de referência f está em *Haversine* [Sinnott 1984].

4.3 Desenvolver Algoritmos

Os algoritmos desta pesquisa, processam a consulta *CETPP*, especificada na Seção 4.2. O primeiro algoritmo a ser desenvolvido é o *Baseline*. O algoritmo *Baseline*

[3] `https://opendatacommons.org/licenses/odbl/`

serve para comparar o desempenho com os demais algoritmos que buscam um menor tempo de resposta ao processar a consulta *CETPP*, estes algoritmos são apresentados a seguir. O primeiro algoritmo avançado é o *Text First Algorithm*, ele utiliza um filtro textual para otimizar o processamento da consulta. O segundo algoritmo é o *IF Text First Algorithm*, que utiliza um arquivo invertido para realizar o filtro textual. O terceiro algoritmo é o *Spatial First Algorithm*, que utiliza uma *R-tree* para otimizar o filtro espacial. O último algoritmo é o *Hybrid Algorithm*, que utiliza o índice *S2I* para realizar o filtro espacial e textual de forma eficiente.

4.4 Avaliar Resultados

Este trabalho realiza uma pesquisa de caráter quantitativo, que "significa traduzir em números opiniões e informações para classificá-las e analisá-las. Requer o uso de recursos e de técnicas estatísticas (percentagem, média, moda, mediana, desvio-padrão, coeficiente de correlação, análise de regressão, etc.)" [Prodanov e de Freitas 2013].

Essa pesquisa utiliza uma avaliação quantitativa para investigar o impacto dos algoritmos no tempo de resposta ao variar alguns parâmetros da consulta e na base de dados. A seguir, os parâmetros que são avaliados:

1. Variação do tamanho do conjunto de objetos de interesse. Obtemos da base de dados *Quot* informações imobiliárias de 4 cidades brasileiras, a variação da quantidade dos objetos de interesse esta relacionado com a cidade.

2. Tamanho do conjunto de objetos de referência. Obtemos da base de dados *OSM* objetos espaciais diversos (e.g. hotéis, cafeterias) para representar os objetos de referência, a variação da quantidade dos objetos de referência esta relacionado com a cidade selecionada.

3. Número de objetos retornados k. Tem como objetivo disponibilizar o custo que uma aplicação tem em retornar os k objetos de interesse.

4. Raio da consulta. O objetivo da variação desse parâmetro, surge de simular valores que são frequentemente utilizado por usuários em mecanismos de pesquisa. É esperado que o tempo de resposta dos algoritmos seja mais lento a medida que o raio é incrementado, porque quanto maior o raio, uma quantidade maior de objetos de referência são analisados.

5. Número de palavras-chave. A variação desse parâmetro tem o objetivo de simular a consulta *CETPP* com uma quantidade variável de palavras-chave $Q.D$ (e.g. "Farmácia", "Farmácia Popular", "hospital infantil criança"). Para realizar este experimento, inicialmente é criado um vocabulário com os termos encontrados nos objetos de referência com suas respectivas frequências, após este processo, é realizado um ordenamento decrescente pela frequência

dos termos, finalmente é realizado um sorteio no grupo dos 10% dos termos mais frequentes e a depender do parâmetro quantidade de palavras-chave, é retornado de 1 a 5 termos que são utilizados na consulta.

6. Limiar de similaridade textual dos objetos de referência. O objetivo deste parâmetro é variar o filtro textual realizado pela similaridade das palavras-chave de busca $Q.D$ com o texto presente nos objetos de referência. A depender do valor do limiar, implica na quantidade de objetos de referência que são considerados como relevantes. Espera-se que quanto maior o valor do limiar de similaridade textual da consulta, os algoritmos apresentem um menor tempo de resposta, porque uma quantidade menor de objetos de referência são analisados.

Cada consulta *CETPP* é executada com 20 repetições, com o objetivo de evitar problemas de performance por abertura de algum processo inesperado enquanto o experimento é executado. Após esta etapa, calcula-se o valor médio do tempo de resposta dessas repetições, este é o tempo de resposta utilizado como resultado. Além disso, foram realizados testes em dias diferentes, com o objetivo de verificar se ocorria diferença significativa na média do tempo de resposta dos algoritmos para identificar problemas relacionados ao *hardware*.

4.5 Publicar Resultados

No final da pesquisa, foi submetido um artigo para que as ideias sejam avaliadas por pesquisadores e especialistas, um artigo completo foi enviado para uma conferência de prestígio. Além disso, está sendo mantido uma aplicação com a possibilidade de realizar a consulta espacial proposta neste trabalho. As aplicações deste trabalho podem ser encontrada através do site[4] de um dos autores.

4.5.1 Exemplo de Aplicação

Uma aplicação web foi criada para exemplificar a consulta *CETPP*. A escolha web tem a finalidade de abranger o maior número de dispositivos. Além disso, esta aplicação foi desenvolvida de forma responsiva com o objetivo de funcionar em diversos dispositivos de tamanho de telas variadas, por exemplo: *desktop*, *smartphone*, *tablets*, etc. Para alcançar este objetivo foi utilizado o *framework Bootstrap*[5].

A Figura 4.1 apresenta a tela inicial da aplicação. No campo de selecionar os objetos de interesse é possível selecionar apartamento (*Quot*), restaurante (*OSM*) e farmácia *(OSM)*. No campo dos objetos de referência é possível selecionar 4 cidades que foram exportadas do *OSM* e utilizadas no experimento desta pesquisa. Os parâmetros de

[4] `http://www.claudiovaliense.com`
[5] `https://getbootstrap.com/`

palavra-chave, raio, limite de resultado k e limite de escore textual são inseridos pelo usuário no momento da pesquisa. No campo algoritmo, é preciso que seja selecionado um dos algoritmos desenvolvidos para processar a consulta *CETPP*. Na Figura 4.1 utilizamos a base dos apartamentos (objetos de interesse) do *Quot* para responder a pergunta: "Qual o apartamento a venda no Rio de Janeiro que tem mais locais (objetos de referência) com o termo "padaria", em uma distância de no máximo 200 metros?".

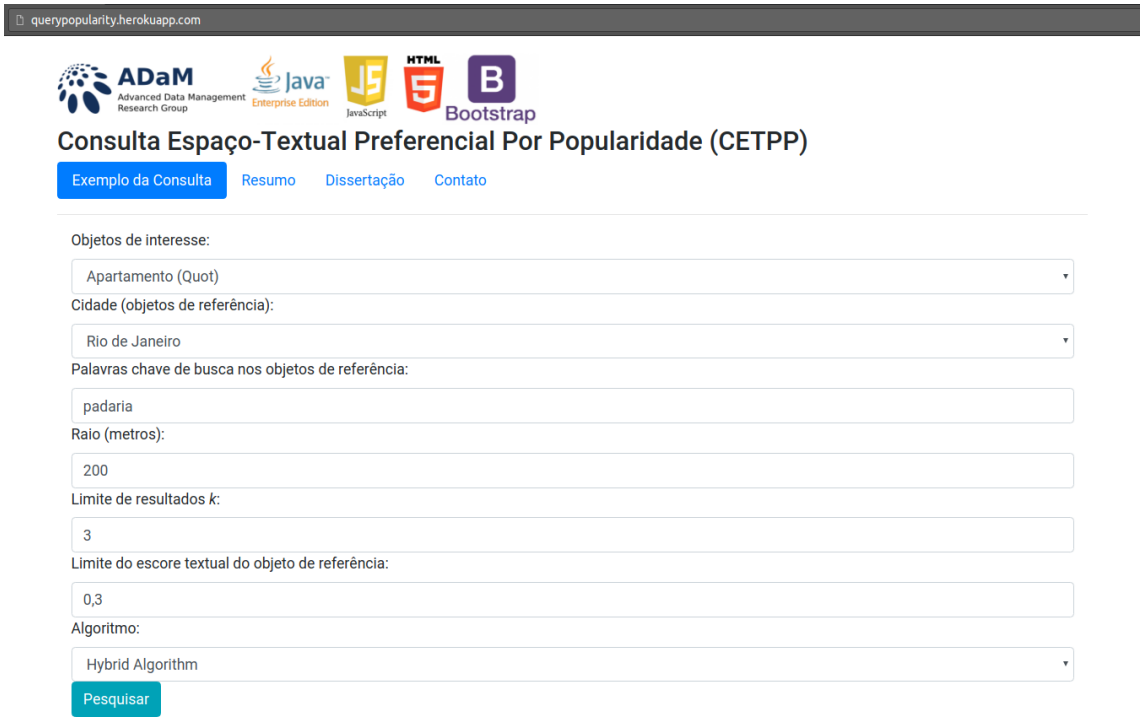


Figura 4.1: Tela inicial da aplicação da consulta CETPP. Fonte: Próprio Autor.

A Figura 4.2, apresenta o resultado da consulta *CETPP*, a partir dos parâmetros da Figura 4.1. Inicialmente é apresentada uma tabela que contém os objetos de interesse com sua devida classificação e ordenado pelo escore, além dessa informação, é apresentado a localização (latitude, longitude) e uma informação associada ao objeto de interesse (e.g. preço do imóvel). Utilizamos a biblioteca *Leaflet*[6] para manipular o mapa do *OSM*, por exemplo, adicionando ícones.

Além da aplicação da Figura 4.1, foi desenvolvido uma aplicação *jar* para que outros pesquisadores avaliarem a consulta *CETPP*. Com este *jar* é possível processar a consulta *CETPP* na base fornecida no parâmetro de entrada, além de permitir que o pesquisador avalie a consulta variando os parâmetros especificados em 4.2. A Figura 4.3 apresenta a execução da consulta CETPP em um terminal.

[6] `https://leafletjs.com/`

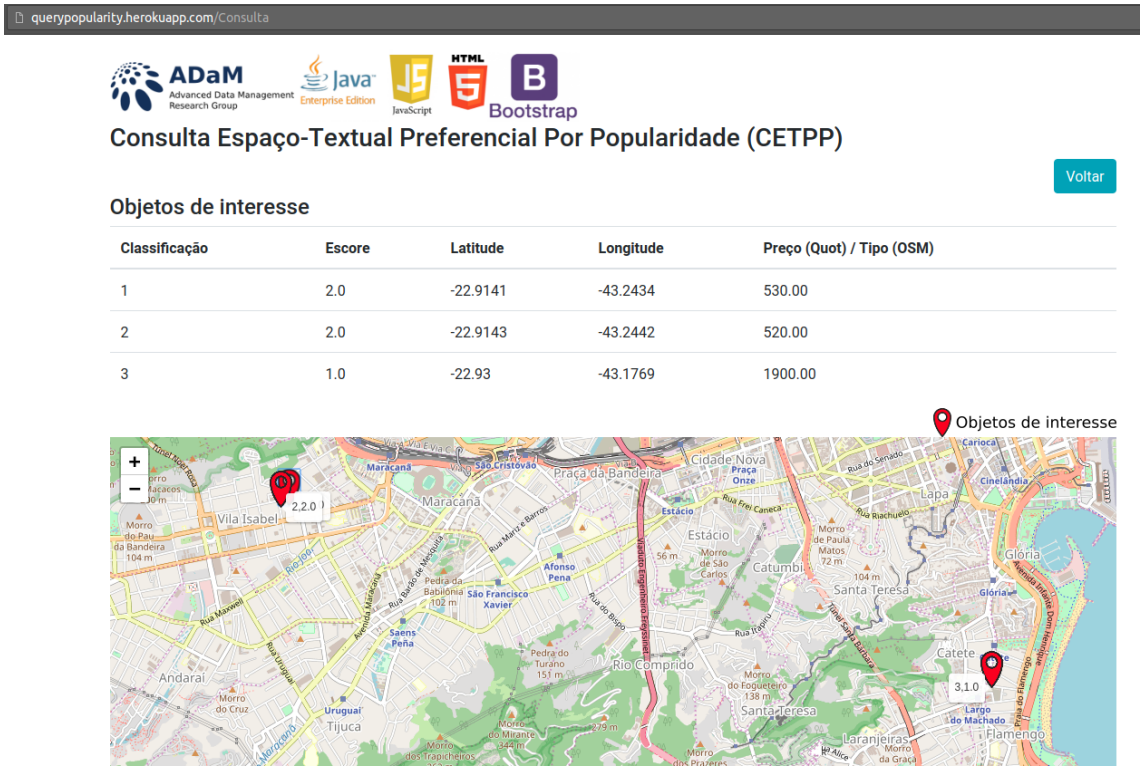


Figura 4.2: Resultado da consulta CETPP. Fonte: Próprio Autor.

```
pc@pc-pc:~/Documentos/teste$ java -jar CETPP.jar P.csv F.csv 'escola' 1000 3 0.3 hybrid
Total de objetos espaciais em P.csv: 31022
Total de objetos espaciais em F.csv: 2573
Tempo para criar o índice R-tree: 455 milissegundos
Tempo para criar o índice Inverted File: 188 milissegundos
Tempo para criar o índice S2I: 415 milissegundos
Quantidade total de termos no conjunto dos objetos de referência: 6398
Quantidade de termos único no conjunto dos objetos de referência: 2700
Tempo para realizar a consulta com o algoritmo hybrid: 2607 milissegundos
Classificação: 1, Escore: 16.0, ID: 643727, Latitude: -22.9143, Longitude: -43.1832
Classificação: 2, Escore: 16.0, ID: 46802, Latitude: -22.9146, Longitude: -43.1826
Classificação: 3, Escore: 16.0, ID: 645958, Latitude: -22.9143, Longitude: -43.1832
pc@pc-pc:~/Documentos/teste$
```

Figura 4.3: CETPP no formato jar. Fonte: Próprio Autor.

Capítulo 5

Algoritmos Propostos

"A vida é um algoritmo novo todos os dias."
– Bruno de Souza Almeida

Neste capítulo contém cinco algoritmos propostos para processar a consulta *CETPP* e o tempo de complexidade dos algoritmos. O *BaseLine Algorithm* que percorre todos os objetos de ambos os conjuntos de interesse e referência; *Text First Algorithm* que utiliza o cálculo na similaridade no conjunto dos objetos de referência inicialmente; *IF Text First Algorithm* que utiliza o Arquivo Invertido [Zobel e Moffat 2006] para filtrar os elementos que contém a palavra-chave de busca; *Spatial First Algorithm* que utiliza a *R*-Tree* [Beckmann et al. 1990] para selecionar o conjunto de objetos de referência pela condição espacial e *Hybrid Algorithm* que utiliza o índice *S2I* [Rocha-Junior et al. 2011] para buscar os objetos de referência que atendem ao critério espacial e textual.

Em todo o texto, é utilizado P para representar o conjunto de objetos de interesse, F o conjunto de objetos de referência, $Q.D$ o conjunto de palavras-chave, $Q.r$ que representa a distância máxima (raio) para um objeto de referência, $Q.k$ a quantidade de objetos de interesse retornados e $Q.\sigma$ que representa o escore textual mínimo que um objeto de referência precisa ter para ser considerado. Ao final dos algoritmos, é retornado uma lista L que contém k objetos de interesse ordenados, em ordem decrescente, por escore.

5.1 Algoritmo Baseline

Esta seção apresenta o algoritmo Baseline (Algoritmo 1). Esse algoritmo, como o nome sugere, é usado para medir o desempenho dos outros algoritmos. Além

disso, facilita a apresentação, porque os outros algoritmos são construídos de forma incremental a partir dele. Este algoritmo calcula o escore de cada objeto de interesse $p \in P$ a partir dos objetos de referência que atendem aos critérios espacial e textual. Para cada objeto de interesse p, todos os objetos de referência $f \in F$ são percorridos. Este algoritmo é capaz de processar a consulta *CETPP* e serve como parâmetro de comparação para os demais algoritmos.

O algoritmo Baseline (Algoritmo 1) utiliza uma *MinHeap* H para armazenar os elementos de acordo com o seu escore (linha 1). Os elementos com menor escore ficam no topo da *MinHeap*, visto que são removidos primeiro. Ao final do algoritmo, a lista L vai conter os k objetos de interesse com os maiores escores (linha 2). O algoritmo percorre o conjunto de objetos de interesse P (linhas $3-12$), e para cada objeto $p \in P$, ele verifica os objetos $f \in F$ que satisfazem o critério espacial (linha 6) e textual (linha 7). O algoritmo incrementa o escore de cada objeto p, a medida que encontra um objeto de referência f que atende ambos os critérios (linha 8). O objeto p com escore maior que 0 (linha 9) é adicionado a *MinHeap* H (linha 10). O algoritmo só precisa manter na *MinHeap* H os k objetos de interesse com os maiores escores (linhas 11-12). Para retornar os objetos de interesse ordenado pelo escore de forma decrescente, é retirado os elementos do topo da *MinHeap* H e adicionado cada elemento no inicio da lista L (linha $13-14$).

Algorithm 1: Baseline Algorithm

Input: P (Objects of Interest), F (Objects of Reference), $Q.D$ (Keywords), $Q.r$ (Radius), $Q.k$ (Number o results), $Q.\sigma$ (Limit Score text)
Output: List that maintains the k best objects of interest

```
1  MinHeap H ← ∅;
2  List L ← ∅;
3  forall p ∈ P do
4      p.score = 0;
5      forall f ∈ F do
6          if dist(p,f) ≤ Q.r then
7              if θ(f.D, Q.D) ≥ Q.σ then
8                  p.score++;
9      if p.score > 0 then
10         H.add(p);
11         if |H| > Q.k then
12             H.remove();
13 while |H| > 0 do
14     L.addFirst(H.remove());
15 return L;
```

5.2 Algoritmos Aplicando Filtro Textual

Nesta seção, nós apresentamos dois algoritmos para deixar a consulta *CETPP* mais eficiente. Na Subseção 5.2.1 contém o Algoritmo 2 que utiliza a estratégia de filtrar todo o conjunto dos objetos de referência para posteriormente percorrer este conjunto reduzido. Na Subseção 5.2.2 utilizamos a estrutura de dados Arquivo Invertido [Zobel e Moffat 2006] para reduzir o conjunto dos objetos de referência, esta estrutura foi descrita com maiores detalhes na Seção 2.1.

5.2.1 Algoritmo Aplicando Filtro Textual

Esta seção apresenta o algoritmo *Text First* (Algoritmo 2). Este algoritmo calcula o escore de cada objeto de interesse a partir dos objetos de referência que atendem aos critérios espacial e textual. Inicialmente, este algoritmo percorre uma vez o conjunto $f \in F$, para cada f que ultrapasse o critério textual, é adicionado no conjunto F' que é composto por todos os objetos de referência que são relevante no critério textual. Para cada objeto de interesse $p \in P$, todos os objetos de referência $f \in F'$ são percorridos, é verificado se f atende ao critério espacial, caso atenda, incrementa o escore de p. A ideia por trás deste algoritmo, quando comparado ao *Baseline* (Algoritmo 1), é que este algoritmo percorre uma única vez o conjunto F para selecionar os objetos que são textualmente relevante primeiro, reduzindo assim a etapa de percorrer todo o conjunto F para cada $p \in P$, reduzindo assim o processamento da consulta.

O algoritmo *Text First* (Algoritmo 2) utiliza uma *MinHeap* H para armazenar os elementos de acordo com o seu escore (linha 1). Os elementos com menor escore ficam no topo da *MinHeap*, visto que são removidos primeiro. Ao final do algoritmo, a lista L vai conter os k objetos de interesse com os maiores escores (linha 2). O algoritmo 2 cria F' para armazenar um novo conjunto dos objetos de referências (linha 3). Em F' são adicionados os objetos de referência que são textualmente relevantes com as palavras-chave $Q.D$ (linhas $3-6$). O restante do algoritmo é processado de forma análoga ao Algoritmo 1, com exceção de que o conjunto F' é percorrido (linha 9) ao invés de F. Além disso, não é necessário verificar se os objetos $f \in F'$ atendem ao critério textual, visto que todos objetos f que pertencem

a F' atendem.

Algorithm 2: Text First Algorithm

Input: P (Objects of Interest), F (Objects of Reference), $Q.D$ (Keywords), $Q.r$ (Radius), $Q.k$ (Number o results), $Q.\sigma$ (Limit Score text)
Output: List that maintains the k best objects of interest

```
1  MinHeap H ← ∅;
2  List L ← ∅;
3  F' ← ∅;
4  forall f ∈ F do
5      if θ(f.D, Q.D) ≥ Q.σ then
6          F'.add(f);
7  forall p ∈ P do
8      p.score = 0;
9      forall f ∈ F' do
10         if dist(p,f) ≤ Q.r then
11             p.score++;
12     Lines 9-12 of the Baseline algorithm (Algorithm 1)
13 while |H| > 0 do
14     L.addFirst(H.remove());
15 return L;
```

5.2.2 Algoritmo utilizando Arquivo Invertido

Esta seção apresenta o algoritmo *Inverted File* (Algoritmo 3). Neste algoritmo é utilizado o arquivo invertido (IF) para reduzir o conjunto dos objetos de referência. Utilizando o *IF* é retornado apenas os objetos de referência em F que contém alguns dos termos da consulta $Q.D$, facilitando o cálculo da similaridade textual.

Assim como o Algoritmo *Fist Text* (Algoritmo 2), este algoritmo cria um novo conjunto de objetos de referência F' e adiciona neste conjunto apenas os objetos de referência que atendem ao critério textual. Para cada objeto de interesse $p \in P$, todos os objetos de referência de $f \in F'$ são percorridos, é verificado se f atende ao critério espacial, caso atenda, incrementa o f no escore de p.

Assim como os algoritmos 1 e 2, o algoritmo *Inverted File* (Algoritmo 3) utiliza uma *MinHeap* H para armazenar os objetos de interesse p de acordo com o seu escore (linha 1). Os elementos com menor escore ficam no topo da *MinHeap*, visto que são removidos primeiro. Ao final do algoritmo, a lista L vai conter os k objetos de interesse com os maiores escores (linha 2). O Algoritmo 3 cria F' para representar um novo conjunto dos objetos de referências que atendem ao critério textual (linha 3). É utilizado uma *HashMap* para armazenar o identificador do objeto de referência

e o seu respectivo escore de similaridade textual (linha 4). Para cada termo q da consulta $Q.D$, é retornado o conjunto de objetos que tem o termo, acompanhado da sua correspondente frequência, o escore textual do objeto de referência é a soma do escore de cada termo presente no objeto (linhas 5-7). O algoritmo verifica se o objeto de referência supera o liminar textual, caso sim, é buscado em F as informações do objeto referência e adicionado no conjunto de objetos de referência F' (linhas 8-9). O conjunto F' contém todos os objetos de referência que atendem ao filtro textual. As demais linhas são análogas as linha 7-14 do Algoritmo 2.

Algorithm 3: IF Text First Algorithm

Input: P (Objects of Interest), *IF* (Inverter File of reference objects), $Q.D$ (Keywords), $Q.r$ (Radius), $Q.k$ (Number o results), $Q.\sigma$ (Limit Score text)

Output: List that maintains the k best objects of interest

```
1  MinHeap H ← ∅;
2  List L ← ∅;
3  F' ← ∅;
4  HashMap<Long, Double> M;
5  foreach q ∈ Q.D do
6      foreach t ∈ IF.list(q) do
7          M.put(t.id, M.get(t.id) + termScore(t, t.fq));
8          if M.get(t.id) ≥ Q.σ then
9              F'.add(F.get(t.id));
10 Lines 7-14 of the Text First Algorithm (Algorithm 2)
11 return L;
```

5.3 Algoritmo Aplicando Primeiro Filtro Espacial

Esta seção apresenta o algoritmo *Spatial First* (Algoritmo 4). Este algoritmo utiliza a estrutura *R*-Tree* com o objetivo de retornar de forma eficiente o conjunto de objetos de referência F' que atendem ao critério espacial $Q.r$. Para cada $p \in P$, é executado a consulta na *R*-Tree* dos objetos de referência, sendo p um dos parâmetros da consulta na *R*-Tree*, é retornado o conjunto de objetos de referência F' que atendem ao critério espacial $Q.r$. Após obter o conjunto F', o algoritmo verifica os objetos $f \in F'$ que satisfazem o critério textual. Este algoritmo utiliza a estrutura *R*-Tree*, que foi obtida utilizando a biblioteca XXL[1], está biblioteca foi publicada em importantes eventos da área [Bercken et al. 2001, Cammert et al. 2003].

Como nos algoritmos anteriores, o algoritmo *Spatial First* (Algoritmo 4) utiliza uma *MinHeap* H (linha 1) e a lista L (linha 2). O algoritmo percorre o conjunto de objetos de interesse P (linhas 3-9), e para cada objeto $p \in P$, é pesquisado em F (

[1] https://github.com/umr-dbs/xxl

R-Tree_F*) o conjunto de objetos de referência F' que atendem ao critério espacial de p ($dist(p, f) < Q.r$) (linha 5). Para cada objeto de referência $f \in F'$ (linha 6), checa se f é textualmente relevante (linha 7), incrementando o escore de p (linha 8). O restante do algoritmo é processado de forma análoga ao algoritmo *Baseline* (Algoritmo 1).

Algorithm 4: Spatial First Algorithm

Input: P (Objects of Interest), R*-Tree_F (R*-Tree of reference objects), $Q.D$ (Keywords), $Q.r$ (Radius), $Q.k$ (Number o results), $Q.\sigma$ (Limit Score text)

Output: List that maintains the k best objects of interest

```
1  MinHeap H ← ∅;
2  List L ← ∅;
3  forall p ∈ P do
4      p.score = 0;
5      F' ←R*-Tree_F.search(p.x,p.y,Q.r);
6      foreach f ∈ F' do
7          if θ(f.D, Q.D) ≥ Q.σ then
8              p.score++;
9      Lines 9-12 of the Baseline algorithm (Algorithm 1)
10 while |H| > 0 do
11     L.addFirst(H.remove());
12 return L;
```

5.4 Algoritmo Híbrido

Esta seção apresenta o algoritmo *Hybrid* (Algoritmo 5). Este algoritmo utiliza o índice *S2I* (Seção 2.3.1) para indexar o conjunto de objetos de referência. Com o objetivo de calcular de forma mais eficiente o escore dos objetos de interesse, é utilizado o índice *S2I* para filtrar os objetos de referência que atendem ao critério espacial e textual para cada objeto de interesse $p \in P$ ($dist(p, f) < Q.r$ e $\theta(f.D, Q.D) \geq Q.\sigma$)).

O índice *S2I* (Seção 2.3.1) foi proposto para processar consultas *Top-k spatial-keyword Queries*. Dada uma localização espacial (latitude e longitude) e um conjunto de palavras-chave de consulta; essa consulta retorna os k melhores objetos espaço-textuais em ambos os termos: distância ao local da consulta e semelhança textual com as palavras-chave da consulta. Os dois algoritmos SKA (*Single Keyword Algorithm*) e MKA (*Multiple Keyword Algorithm*) propostos por Rocha-Junior et al. (2011) não podem ser aplicados para processar a consulta *CETPP*, por causa de duas restrições principais: por causa de duas restrições principais: 1) eles não retornam os objetos em uma região espacial definida por um raio, eles retornam os

melhores objetos k em um vizinho mais próximo, e 2) eles não empregam um filtro de limiar para selecionar melhores objetos, eles calculam o escore e selecionam os melhores objetos k. Portanto, nós tivemos que criar um novo algoritmo chamado *RangeSearch*.

O algoritmo *RangeSearch* recebe como parâmetro um local de consulta (latitude, longitude), um raio ($Q.r$), um conjunto de palavras chave de consulta ($Q.D$) e um limiar de similaridade ($Q.\sigma$) e retorna todos os objetos espaço-textuais na região espacial de interesse, delimitada pelo local de consulta e raio, cuja relevância textual para as palavras-chave de consulta $Q.D$ estão acima do limite de similaridade da textual ($Q.\sigma$). Primeiro, o algoritmo acessa as listas invertidas das palavras-chave presentes na descrição textual dos objetos espaço-textuais de referência, recuperando todos os objetos que são textualmente relevantes e chega se a distância para p é menor que $Q.r$. Finalmente, emprega o impacto [Rocha-Junior et al. 2011] de uma palavra-chave (termo) no texto do objeto espaço-textual para calcular a relevância textual de cada objeto espaço-textual, removendo aqueles cuja similaridade textual está abaixo do limiar de consulta ($Q.\sigma$).

Neste algoritmo, para cada objeto de interesse $p \in P$, é executada a consulta no índice *S2I* dos objetos de referência, sendo a localização do objeto p um dos parâmetros da consulta do índice. Além da localização do objeto p, as palavras-chave da consulta Q, o limiar espacia $Q.r$, e o limite de similaridade textual $Q.\sigma$ são utilizados na consulta do índice *S2I*. Com o processamento da consulta no índice *S2I*, são retornados todos os objetos de referência F' que atendem ao critério espacial e textual. O escore do objeto de interesse p é baseado na quantidade de objetos de referência retornado pelo índice *S2I*.

Algoritmo 5 contém uma descrição do Algoritmo Híbrido que usa o *RangeSearch* e *S2I* (índice híbrido. Como os algoritmos anteriores, ele usa um MinHeap H (linha 1) e uma lista L (linha 2). O algoritmo atravessa o conjunto de objetos de interesse P (linhas 3-6), e para cada objeto $p \in P$, usa o algoritmo *RangeSearch* (linha 4) para encontrar o espaço objetos textuais de referência $F' \subseteq F$ que são espacialmente e textualmente relevantes (linha 4): $dist(p, f) < Q.r$ e $\theta(f.D, Q.D) \geq Q.\sigma$. O escore do objeto de interesse p é definido pela cardinalidade (tamanho) do conjunto F'

($|F'|$) (linha 5).

Algorithm 5: Hybrid Algorithm

Input: P (Objects of Interest), S2I_F (S2I of reference objects), $Q.D$ (Keywords), $Q.r$ (Radius), $Q.k$ (Number o results), $Q.\sigma$ (Limit Score text)

Output: List that maintains the k best objects of interest

```
1 MinHeap H ← ∅;
2 List L ← ∅;
3 forall p ∈ P do
4     F' = S2I_F.RangeSearch(p.x,p.y,Q.r,Q.D,Q.σ);
5     p.score = |F'|;
6     Lines 9-12 of the Baseline algorithm (Algorithm 1)
7 while |H| > 0 do
8     L.addFirst(H.remove());
9 return L;
```

5.5 Tempo de Complexidade dos Algoritmos

Nesta seção nós analisamos o tempo de complexidade de cada algoritmo proposto. Para cada algoritmo, nós apresentamos o pior, médio e melhor tempo possível. Em todos os casos, usamos a notação Theta $\Theta(n)$, que é a maneira formal de expressar a complexidade média do tempo limite de um algoritmo [Cormen et al. 2009]. Tabela 5.1 resume a complexidade de tempo dos algoritmos. P é o conjunto de objetos de interesse e F é o conjunto de objetos espaço-textuais de referência. $|P|$ e $|F|$ representam a cardinalidade (tamanho) de cada conjunto, respectivamente.

O tempo de complexidade do algoritmo *Baseline* (Algoritmo 1) é $\Theta(|P| * |F|)$ para o pior, médio e melhor caso. Para cada objeto de interesse $p \in P$, ele visita todos os objetos de referência $f \in F$.

O algoritmo *Text First* (Algoritmo 2), percorre uma única vez o conjunto F com o objetivo de identificar os objetos de referência $F' \subseteq F$ que satisfizeram o critério textual. A partir disto, percorre para cada objeto $p \in P$, todos os objetos de referência no conjunto $f \in F'$. No pior caso, todos os objetos de referência atenderam ao critério textual, ou seja, F' será o mesmo conjunto F. Portanto, a complexidade do pior caso deste algoritmo é $\Theta(|F| + (|P| * |F|))$, que é pior do que o tempo de complexidade do algoritmo *Baseline*. No caso do meio, o tamanho de F' depende da seletividade das palavras-chave de consulta $Q.D$. Assumindo que metade dos objetos são textualmente relevantes $|F'| = |F|/2$ (na prática, o tamanho é muito menor), a complexidade de tempo no caso médio é $\Theta(|F| + (|P| * |F|/2))$. Na melhor das hipóteses, o tamanho de $|F|$ é uma pequena constante. Portanto, a complexidade do melhor caso corresponde a $\Theta(|F| + |P|)$.

A única diferença entre o algoritmo *Text First* (Algoritmo 2) e o *IF Text First* (Algoritmo 3) é que *IF Text First* encontra o conjunto de objetos espaço-textuais que são textualmente relevantes F' mais rápido, já que não precisa percorrer todo o conjunto F para encontrar os objetos, ele usa o índice *IF*. Portanto, a complexidade de tempo diminui de acordo com a seletividade das palavras-chave de consulta. No pior dos casos, a complexidade de tempo é a mesma do algoritmo *Text First*. No entanto, no caso médio, a complexidade de tempo é $\Theta(|F|/2 + |P| * |F|/2)$, assumindo que metade dos objetos em F são textualmente relevantes (na prática, é muito melhor); e a complexidade de tempo do melhor caso é $\Theta(1 + |P|)$, assumindo que alguns objetos (pequena constante) são textualmente relevantes.

O algoritmo Spatial First (Algoritmo 4) usa um *R*-Tree* [Beckmann et al. 1990] para encontrar o conjunto F' de objetos espaço-textuais de referência que são espacialmente relevantes para $p \in P$. A média e o melhor tempo de complexidade para executar uma pesquisa de intervalo em um *R*-Tree* é $\Theta(\log n)$ [Arge et al. 2008], enquanto o pior caso é $\Theta(n)$, pois todos os objetos são retornados. Portanto, a complexidade de tempo do pior caso para o algoritmo *Spatial First* é $\Theta(|P| * (|F| + |F|))$, onde o primeiro $|F|$ se refere ao tempo de pesquisa no *R*-Tree*. A complexidade média do tempo do caso é $\Theta(|P| * (\log(|F|) + \log(|F|)))$, assumindo que $\log(F)$ satisfaz a restrição espacial. Nós não consideramos que $|F|/2$ são espacialmente relevantes como fizemos para os algoritmos *Text First*, porque a busca espacial é executada para cada localização de objeto p, portanto menos objetos atenderiam aos critérios espaciais, enquanto a busca textual retorna todos os objetos que são textualmente relevantes, não importando a localização de p. Finalmente, o tempo de complexidade no melhor caso é $\Theta(|P| * \log(F))$, assumindo que um pequeno número constante de objetos é espacialmente relevante.

Finalmente, o algoritmo híbrido (Algoritmo 5) usa o índice *S2I* para encontrar os objetos espaço-textuais que são textualmente e espacialmente relevantes. No pior dos caso, todos os objetos espaço-textuais de referência $f \in F$ podem ser espacialmente (dependendo do tamanho do raio) e textualmente (dependendo das palavras-chave de consulta) relevantes. Portanto, obter os objetos relevantes no *S2I* custaria $\Theta(|F|)$ e o tempo de complexidade do algoritmo é $\Theta(|P| * (|F| + |F|))$. No caso médio, supondo que $\log(|F|)$ objetos sejam textualmente e espacialmente relevantes, a complexidade de tempo médio é $\Theta(|P| * \log(|F|)$. Não é necessário percorrer o conjunto $\log(|F|)$ para calcular a escore de p, porque todos os objetos são espacialmente e textualmente relevantes. Por fim, no melhor dos caso, o tempo de complexidade é a mesma que no caso médio, pois o custo para realizar a pesquisa permanece $\Theta(\log(n))$.

Tabela 5.1: Tempo de complexidade dos algoritmos.

Algorithms	**Worst case** (Θ)	**Average case** (Θ)	**Best case** (Θ)
Baseline	$\|P\| * \|F\|$	$\|P\| * \|F\|$	$\|P\| * \|F\|$
Text First	$\|F\| + (\|P\| * \|F\|)$	$\|F\| + (\|P\| * \|F\|/2)$	$\|F\| + \|P\|$
IF Text First	$\|F\| + (\|P\| * \|F\|)$	$\|F\|/2 + (\|P\| * \|F\|/2)$	$1 + \|P\|$
Spatial First	$\|P\| * (\|F\| + \|F\|)$	$\|P\| * (\log(F) + \log(F))$	$\|P\| * \log(F)$
Hybrid	$\|P\| * (\|F\| + \|F\|)$	$\|P\| * \log(F)$	$\|P\| * \log(F)$

Capítulo 6

Avaliação Experimental

"Aa primeira coisa que um bom cientista faz quando está diante de uma descoberta importante, é tentar provar que ela esta errada."
– Albert Camus

Este capítulo contém uma descrição do estudo experimental realizado. A primeira seção apresenta as bases de dados utilizadas nos experimentos (Seção 6.1). A segunda seção apresenta a configuração do ambiente utilizado no experimento (Seção 6.2). A terceira seção apresenta o tempo de criação dos índices utilizados nos experimentos (Seção 6.3). A quarta seção apresenta a análise do experimento variando a quantidade de resultados k (Seção 6.4). A quinta seção apresenta a análise do experimento variando a quantidade de palavras-chave de busca (Seção 6.5). A sexta seção apresenta a variação do raio na consulta (Seção 6.6). A sétima seção apresenta a análise do experimento variando o limiar de similaridade textual da consulta (Seção 6.7). A oitava seção apresenta o resultado da consulta *CETPP* variando as bases de dados (Seção 6.8). A nona seção apresenta a variação da quantidade dos objetos de referência na cidade do Rio de Janeiro (Seção 6.9). A décima seção apresenta o tamanho do índice em relação as bases de dados (Seção 6.10). Na avaliação dos experimentos, são avaliados o desempenho dos algoritmos ao processar a consulta *CETPP*.

6.1 Bases de Dados

Para representar os objetos de interesse, utilizamos uma base de dados com informações sobre imóveis fornecida pelo *Quot*[1]. Nesta pesquisa utilizamos essa base para

[1] `http://www.quot.com.br/`

representar objetos de interesse. As informações existentes nesta base são: preço, latitude, longitude, nome da cidade. Destas informações, utilizou-se a localização exata de cada imóvel (latitude, longitude) e o nome da cidade na qual ele está localizado. A Tabela 6.1 apresenta o tamanho desta base, agrupando os dados de acordo com as cidades de Recife (RC), Belo Horizonte (BH), Rio de Janeiro (RJ) e São Paulo (SP).

Tabela 6.1: Base de dados do *Quot*.

Base de dados	**P**
Recife (RC)	10917
Belo Horizonte (BH)	23434
Rio de Janeiro (RJ)	31022
São Paulo (SP)	113877

Para representar os objetos de referência, utilizamos a base de dados do *OpenStreetMap*[2](OSM). O *OSM* é um projeto de mapeamento para representar objetos espaciais (e.g. lojas, escolas). Todo o sistema é desenvolvido pela comunidade e usuários de todo o mundo podem fornecer informações de forma cooperativa para melhorar o sistema. Através do sistema é possível selecionar uma área de interesse e exportar as informações no formato *Extensible Markup Language (XML)*.

A Figura 6.1 apresenta um trecho extraído do *OpenStreetMap*. A tag bounds (<bounds>) delimita a área espacial selecionada. A tag node (<node>) contém a latitude (lan) e longitude (lon) dos objetos espaciais. Os elementos node podem possuir outros elementos como no exemplo apresentado na Figura 6.1, onde o node possui duas tags (<tag>) com mais informações sobre o nó (node). No exemplo, uma tag contém a informação do tipo do objeto (amenity) e a outra contém a informação do nome do objeto (name).

Assim como na base do *Quot*, extraímos da base *OSM* os dados de acordo com as cidades de Recife (RC), Belo Horizonte (BH), Rio de Janeiro (RJ) e São Paulo (SP). Das informações existentes, utilizou-se a localização exata de cada objeto de referência (latitude, longitude) e o texto que descreve o nome do objeto. Este objetos espaciais que contém informações textuais associadas, são denominamos de objetos espaço-textuais.

Para representar cada cidade na base *OSM*, foi selecionado um retângulo com quatro pontos, latitude mínima, latitude máxima, longitude mínima e longitude máxima conforme apresentado da Tabela 6.2.

Na Tabela 6.3 são apresentados os dados dos objetos de referência contidos nas regiões da cidades delimitadas pelos pontos da Tabela 6.2. A coluna F representa a quantidade de objetos de referência contidos na cidade exportada. Cada objeto

[2]`https://www.openstreetmap.org`

```
<?xml version="1.0" encoding="UTF-8"?>
<bounds minlat="-12.2142000" minlon="-38.9754000" maxlat="-12.2024000"
   maxlon="-38.9657000"/>
<node id="2517880272" visible="true" version="1" changeset="18683637"
   timestamp="2013-11-02T21:40:18Z" user="luanrc" uid="1793684" lat="
   -12.2045987" lon="-38.9717530">
       <tag k="amenity" v="restaurant"/>
       <tag k="name" v="Fifó"/>
</node>
...
```

Figura 6.1: Trecho do arquivo XML extraído do *OpenStreetMap*. Fonte: Próprio Autor.

Tabela 6.2: Seleção da área de cidades do *OpenStreetMap*.

Cidade	**Lat min**	**Lat max**	**Long min**	**Long max**
Recife (RC)	-8.1649000	-7.9198000	-35.1532000	-34.7240000
Belo Horizonte (BH)	-20.0253000	-19.8084000	-44.1609000	-43.7318000
Rio de Janeiro (RJ)	-22.9452000	-22.8921000	-43.2617000	-43.1577000
São Paulo (SP)	-23.8230000	-23.4683000	-46.7482000	-46.3857000

de referência tem um texto associado, o texto do objeto contém termos. A coluna Termos Total representa o total de termos presentes no conjunto de objetos de referência F. A coluna Termos Únicos representa a quantidade de termos sem repetição em todo o conjunto F. A informação referente aos termos é necessária para calcular a similaridade textual da Equação 2.5 (Seção 2.1) utilizada nesta pesquisa.

Tabela 6.3: Base de dados do *OpenStreetMap*.

Base de dados	**F**	**Termos Total**	**Termos Únicos**
Recife (RC)	1292	3059	1433
Belo Horizonte (BH)	2335	5215	2218
Rio de Janeiro (RJ)	2573	5590	2622
São Paulo (SP)	5551	12011	4404

Em cada experimento, foram utilizados os imóveis da base *Quot* para representar o conjunto de interesse P e os objetos do *OSM* (e.g. farmácia, hotel, escola) para representar o conjunto dos objetos de referência F.

6.2 Configuração

Um ambiente de teste foi montado para rodar os experimentos, que é constituído por um computador com a seguinte configuração: Processador Core $i3 - 6100$ 3.7 GHz; HD 500 GB; Memória 4 GB. Em todos os experimentos, nós medimos o tempo de resposta, variando os valores de uma variável. Os valores das outras variáveis permanecem fixos.

A Tabela 6.4 apresenta os parâmetros que são estudados nos experimentos. O valor padrão está em negrito e esses valores são frequentemente utilizados em pesquisas semelhantes na área [Rocha-Junior et al. 2010, Chen et al. 2013, Zheng et al. 2015].

Tabela 6.4: Parâmetros utilizados na consulta com os valores padrões destacados em negrito.

Parâmetros	**Valores**
Número de resultados (k)	1, **3**, 5, 10
Número de palavras-chave	1, 2, **3**, 4, 5
Valores do raio quilômetros	0.5, **1**, 2, 3, 5, 7
Limite de similaridade textual	0.1, 0.2, **0.3**, 0.4, 0.5
Bases de dados	RC, BH, **RJ**, SP
Tamanho da base	25%, 50%, 75%, **100%**

6.3 Criação dos Índices

Os resultados apresentados neste trabalho nas Seções 6.4, 6.5, 6.6, 6.7 e 6.8 não consideram o tempo de criação dos índices para realizar a consulta espacial, porque com a criação do índice não é preciso recriá-lo ao processar uma nova consulta. Por exemplo, ao analisar o desempenho do algoritmo *IF Text Algorithm*, não considera o tempo para criar o índice *IF*. A Tabela 6.5 apresenta o tempo em milissegundos para criação de cada índice, variando a base de dados.

Tabela 6.5: Tempo para criação do índice em milissegundos.

Base de dados	**IF**	**R-tree**	**S2I**
Recife (RC)	120	340	332
Belo Horizonte (BH)	180	430	610
Rio de Janeiro (RJ)	220	465	651
São Paulo (SP)	376	643	589

6.4 Variando o Número de Resultados k

Esta seção avalia o tempo de resposta, variando o número de resultados k. O tempo de resposta é o tempo decorrido entre a submissão da consulta e o retorno do algoritmo. Ao variar o número k, espera-se que o processamento seja mais lento, visto que um número maior de objetos precisa ser retornado. O tempo de resposta está sendo medido em segundos (Figura 6.2). Na Figura 6.2, é apresentado o tempo de resposta em escala logarítmica, devido a grande diferença entre os resultados dos algoritmos.

Apesar de ser esperado aumento do tempo de resposta com o incremento de k, os tempos dos algoritmos não sofreram variação significativa ao alterar a quantidade de resultados retornados (k). Isto ocorre porque todos os algoritmos precisam percorrer todo o conjunto de objetos de interesse independente do valor de k. Os algoritmos diferenciam um do outro apenas na forma em que eliminam dados do conjunto de referência.

O Algoritmo *Baseline* apresenta o pior tempo de resposta, visto que precisa percorrer, todo o conjunto dos objetos de referência para cada objeto de interesse. O Algoritmo *Hybrid* apresenta melhor desempenho porque utiliza o filtro espacial e textual de forma simultânea, usa o índice *S2I* para retornar o conjunto dos objetos de referência que atendem aos dois critérios.

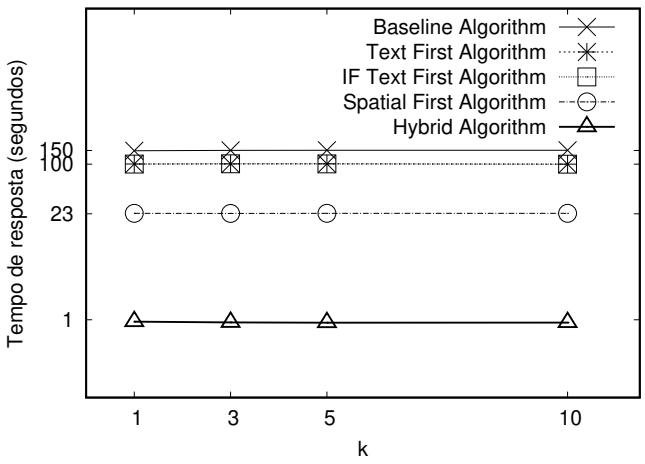


Figura 6.2: Variando o número de resultados k. Fonte: Próprio Autor.

6.5 Variando a Quantidade de Palavras-Chave

Esta seção avalia o tempo de resposta, variando a quantidade de palavras-chave na consulta. O tempo de resposta está sendo medido em segundos (Figura 6.3).

Na Figura 6.3, é apresentado o tempo de resposta em escala logarítmica, devido a grande diferença entre os resultados dos algoritmos.

Analisando a Figura 6.3, o Algoritmo *Baseline* não tem variação significativa na sua resposta em relação ao número de palavras-chave de busca, isto ocorre porque este algoritmo percorre o mesmo conjunto de objetos de referência independente do número de palavras-chave.

Os Algoritmos *Text First Algorithm, IF Text First Algorithm* e *Hybrid Algorithm*, esperam-se que apresentem piores tempos de resposta conforme aumenta a quantidade de palavras-chave de busca. Para 1 palavra-chave, o algoritmo *Text First Algorithm* processou a consulta em 96 segundos, enquanto que para 5 palavras-chave precisou de 98 segundos. Um acréscimo pouco significativo, por razão das características da base de dado utilizada no experimento. A base *OSM* contém em média poucos termos para descrever seus objetos espaciais, diferente de alguns bases que contém uma média maior de termos para descrever seus objetos espaciais (e.g. *Twitter*).

Quanto maior a quantidade de palavras-chave na busca, é maior a possibilidade que o objeto de referência tenha na sua descrição as palavras-chave, implicando em um escore de similaridade diferente de 0. Como estes algoritmos reduzem o conjunto F identificando os objetos que são textualmente relevantes, com o incremento da quantidade de palavras-chave, espera-se aumento do conjunto dos objetos relevantes.

O Algoritmo *Spatial First Algorithm* apresenta crescimento pouco significativo no tempo de resposta. O aumento esta relacionado à quantidade de comparações com um maior número de palavras-chave nos objetos de referência.

Os algoritmos *Text First* e *IF Text First* apresentam quase o mesmo resultado em todos os experimentos. A principal razão dessa semelhança é que, a otimização obtida na busca dos objetos espaço-textuais que são textualmente relevantes não é suficiente para reduzir o tempo total de resposta, já que o maior custo está na segunda etapa, onde a pontuação dos objetos espaciais de interesse são calculados.

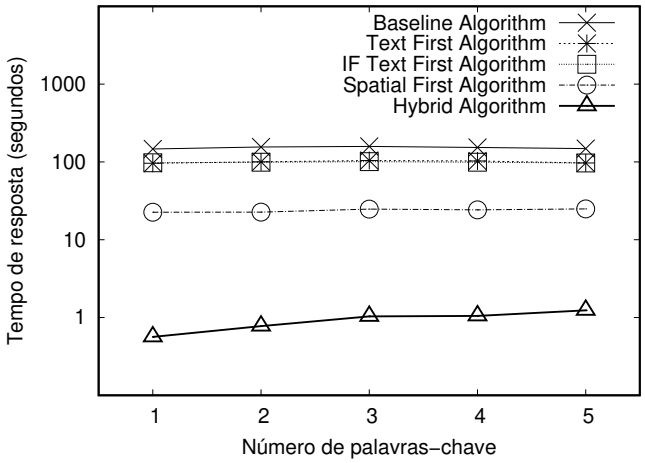


Figura 6.3: Variando a quantidade de palavras-chave na consulta. Fonte: Próprio Autor.

6.6 Variando o Tamanho da Área de Interesse

Esta seção avalia o tempo de resposta variando o raio da consulta. O tempo de resposta está sendo medido em segundos (Figura 6.4). Na Figura 6.4, é apresentado o tempo de resposta em escala logarítmica, devido a grande diferença entre os resultados dos algoritmos.

Na Figura 6.4, o aumento do tempo de resposta nos algoritmos *Baseline Algorithm, Spatial First Algorithm* e *Hybrid Algorithm* ocorre porque com o aumento do limiar da distância espacial, há o crescimento da quantidade de objetos de referência que atendem à condição espacial, isto implica em uma quantidade maior de objetos de referência. Assim, é necessário realizar mais cálculos de similaridade textual.

Os algoritmos *Text First Algorithm* e *IF Text First Algorithm* não tem impacto significativo com relação ao limiar da distância espacial, porque estes algoritmos utilizam apenas o critério de similaridade textual para reduzir o número de objetos de referência visitados durante o processamento da consulta. Portanto, para estes algoritmos, independente do limiar espacial, a mesma quantidade de objetos é percorrida.

Neste experimento verificamos que os algoritmos *Text First Algorithm* e *IF Text First Algorithm* apresentam melhor tempo de resposta para distâncias acima de 5km, comparado ao algoritmo *Spatial First Algorithm* que utiliza o índice *R-tree*. O algoritmo *Hybrid* apresenta melhor desempenho quando comparado com todos os outros algoritmos. Para valores pequenos do raio, é ainda melhor.

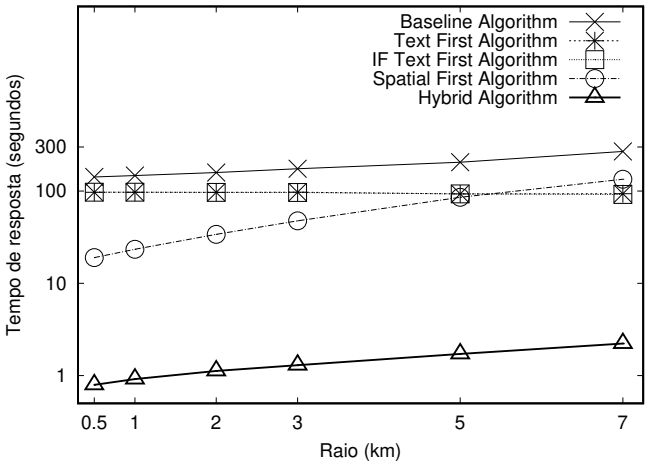


Figura 6.4: Variando o raio na consulta. Fonte: Próprio Autor.

6.7 Variando o Limiar da Similaridade Textual

Esta seção avalia o tempo de resposta variando o limiar de similaridade textual da consulta. O tempo de resposta está sendo medido em segundos (Figura 6.5). Na Figura 6.5, é apresentado o tempo de resposta em escala logarítmica, devido a grande diferença entre os resultados dos algoritmos.

Na Figura 6.5, o Algoritmo *Baseline Algorithm* e *Spatial First Algorithm* não tem variação significativa no seu tempo de resposta, isso ocorre porque é necessário percorrer o mesmo conjunto de objetos de referência independente da variação do limiar de similaridade textual.

Os algoritmos *Text First Algorithm, IF Text First Algorithm* e *Hybrid Algorithm* apresentam um melhor tempo de resposta conforme é aumentado o limiar de similaridade. Isto acontece porque quanto maior o limiar de similaridade textual, é mais difícil que o objeto de referência alcance essa condição, com isso diminuindo o tamanho do conjunto dos objetos de referência que alcançaram esse limiar. Como estes algoritmos reduzem o conjunto dos objetos de referência F identificando os objetos textualmente relevantes, se beneficia com este acréscimo por analisar um conjunto reduzido.

Como os algoritmos *Text First* e *IF Text First* possuem apenas o filtro textual, eles são mais afetados que o *Hybrid*. Especialmente, quando o limite é maior que 0.4, quando poucos objetos espaço-textuais têm uma semelhança textual acima do limite. Novamente, para todos os cenários avaliados, o algoritmo *Hybrid* é uma ordem de magnitude melhor.

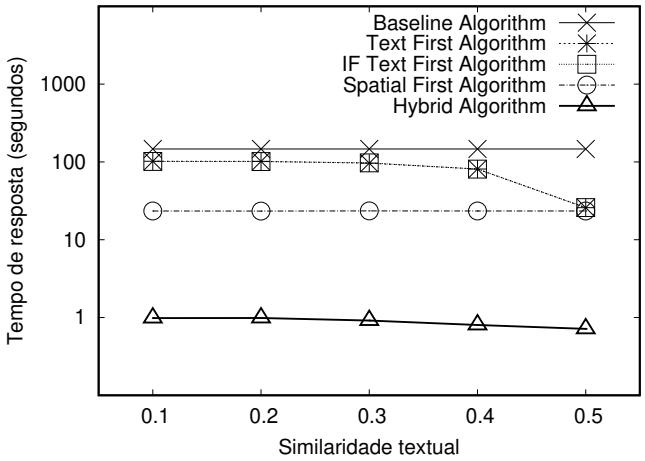


Figura 6.5: Variando a similaridade textual da consulta. Fonte: Próprio Autor.

6.8 Variando a Base de Dados

Neste experimento, nós estudamos o impacto no tempo de resposta dos algoritmos nas cidades de Recife (RC), Belo Horizonte (BH), Rio de Janeiro (RJ) e São Paulo (SP). Na Figura 6.6, novamente é apresentado o tempo de resposta em escala logarítmica, devido a grande diferença entre os resultados dos algoritmos nas bases de dados.

Na Figura 6.6, os tamanhos dos conjuntos de dados aumentam da esquerda para a direita. O menor conjunto de dados é Recife e o maior São Paulo. Todos os algoritmos são impactados pelo aumento nos tamanhos dos conjuntos de dados, porque todos eles são muito dependentes do tamanho de P (objetos de interesse) e F (objetos espaço-textuais de referência) como demonstrado na Seção Tempo de complexidade dos algoritmos. (Seção 5.5). O algoritmo *Hybrid* apresenta melhores resultados que os outros algoritmos, seguidos pelos algoritmos *Spatial First* para todos os conjuntos de dados. Esta figura comprova a eficiência dos algoritmos nas diferentes cidades e demonstra que quanto maior a base, maior o ganho destas abordagens.

6.9 Variando o Tamanho dos Objetos Espaço-Textuais de Referência

Neste experimento fixamos a quantidade de objetos de interesse no conjunto P e variamos a quantidade de objetos de referência (F) na cidade do Rio de Janeiro (Figura 6.7). Nesta figura, o eixo y não está na escala de logaritmos para apresentar melhor os resultados de cada algoritmo.

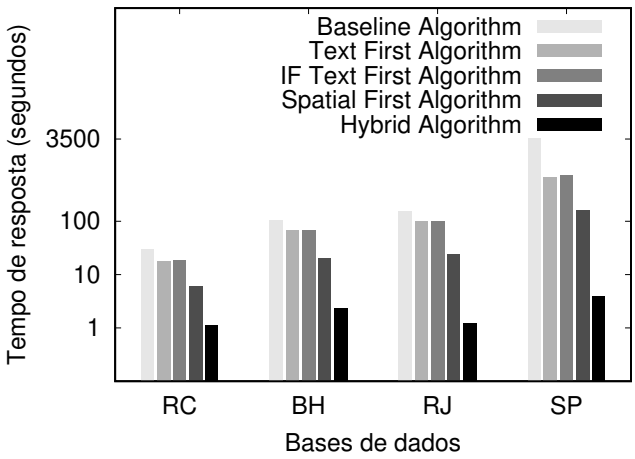


Figura 6.6: Variando a base de dados. Fonte: Próprio Autor.

Todos os algoritmos são impactados pelo aumento do número de objetos espaço-textuais de referência $|F|$. A principal razão é que todos os algoritmos são dependentes do tamanho dos objetos espaço-textuais de referência F (Tabela 5.1). No entanto, esta figura mostra claramente qual algoritmo é mais impactado negativamente por essa variação, que é o algoritmo *Baseline*. O algoritmo *Hybrid* é o mais robusto e mantém o bom tempo de resposta.

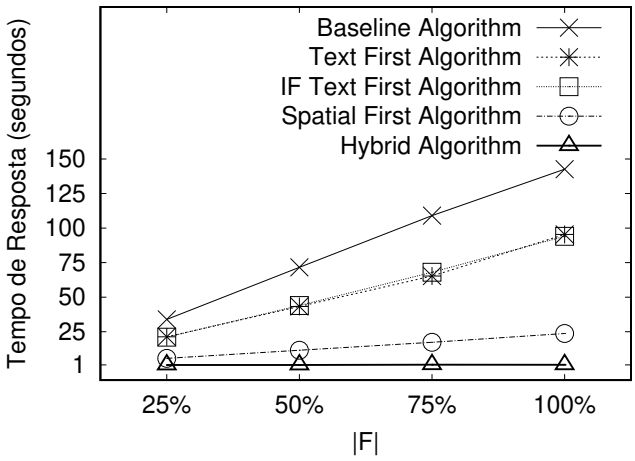


Figura 6.7: Tempo de resposta para diferentes valores de $|F|$ (tamanho de F). Fonte: Próprio Autor.

6.10 Tamanho dos Índices

Três algoritmos utilizam índices: *IF Text First* que usa o Arquivo Invertido [Zobel e Moffat 2006], *Spatial First* que usa o índice espacial [Beckmann et al. 1990] e o algoritmo *Hybrid* que usa o índice espaço-textual [Rocha-Junior et al. 2011]. Em todos esses algoritmos, é utilizado o índice para indexar os objetos de referência F. A Figura 6.8 mostra o tamanho de cada índice em *Kilobyte* para diferentes conjuntos de dados.

A Figura 6.8 apresenta o tamanho do índice em *Kilobyte* variando a base de dados. Para todos os conjuntos de dados, o índice *S2I* requer mais espaço. A principal razão é que mantém informações textuais e espaciais. O Arquivo Invertido e a *R*-Tree* têm tamanhos comparáveis. Além do espaço extra requerido pelo *S2I*, o desempenho alcançado pelos algoritmos usando este índice mostra a vantagem de usar um índice híbrido para este tipo de consulta.

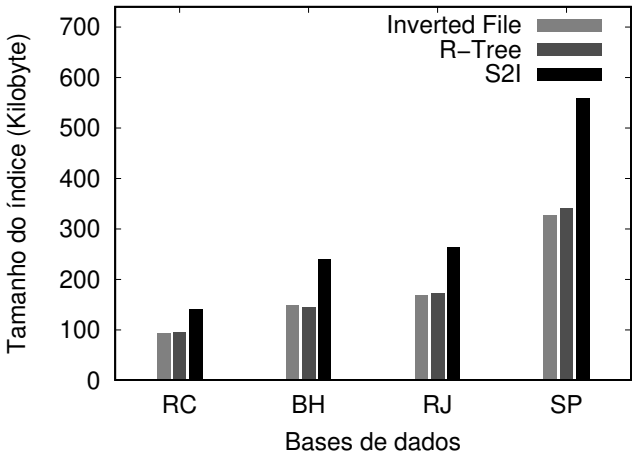


Figura 6.8: Tamanho de cada índice. Fonte: Próprio Autor.

Capítulo 7

Considerações Finais

"A imaginação é mais importante que o conhecimento."
– Albert Einstein

A Consulta Espaço-Textual Preferencial por Popularidade (CETPP) tem grande aplicabilidade e pode ser utilizada em futuros produtos inovadores. A verificação da relevância textual dos objetos de referência no momento da consulta permite que o usuário utilize quaisquer termos para selecionar os objetos espaço-textuais de interesse. A utilização de um limite mínimo para relevância textual auxilia na filtragem de resultados, eliminando objetos de referência com pouca ou nenhuma relevância do conjunto resposta. Novos algoritmos para processar essa consulta foram propostos e implementados, além disso, foram validados utilizando bases de dados reais.

7.1 Principais Contribuições

As principais contribuições desta pesquisa estão na especificação da consulta e nos cinco algoritmos desenvolvidos para processar a consulta *CETPP*.

A consulta *CETPP* trás a novidade de oferecer o escore de um objeto espacial de interesse baseado na contagem de objetos espaciais de referência relevantes. Esta é uma novidade em pesquisas que envolvem dados espaço-textuais. Muitas aplicações podem surgir utilizando a consulta *CETPP*, por exemplo, identificação de locais com maior variedade de serviços ou identificação do local ideal para um novo estabelecimento.

Propomos alguns algoritmos para processar a consulta *CETPP*. O algoritmo *Baseline* (Algoritmo 1), realiza a consulta *CETPP* de forma trivial, percorrendo para

cada objeto de interesse $p \in P$, todos os objetos de referência $f \in F$. Para cada objeto f é verificado se atende ao critério espacial e textual, caso atenda, incrementa o escore de p. Os demais algoritmos aplicam técnicas para reduzir a quantidade de acessos ao conjunto dos objetos de referência F. O algoritmo *First Text* (Algoritmo 2), percorre uma vez o conjunto F para verificar os objetos f que são textualmente relevantes F', a partir disso, percorre para cada $p \in P$, o conjunto de objetos de referência reduzido F', não sendo necessário percorrer todo o conjunto F. O algoritmo *IF Text First* (Algoritmo 3), utiliza o arquivo invertido para retornar os objetos de referência $f \in F$ que contém os termos da palavras-chave de busca $Q.D$, a partir disso verificar se estes objetos são relevantes textualmente criando F', da mesma forma que o Algoritmo 2, percorrendo para cada $p \in P$, o conjunto de objetos de referência reduzido F'. O algoritmo *Spatial First* (Algoritmo 4) utiliza o índice *R-tree* para verificar eficientemente os objetos de referência que ultrapassaram o limiar espacial. O algoritmo *Hybrid* (Algoritmo 5), para cada $p \in P$, o escore de p é baseado na quantidade de objetos retornado pelo índice *S2I*, o índice filtra os objetos que atendem ao critério espacial e textual.

7.2 Trabalhos Futuros

Nesta dissertação nós apresentamos a consulta *CETPP*. Algumas pesquisas podem surgir a partir dessa dissertação. A seguir, nós apresentamos algumas possibilidades para estender esta pesquisa:

Consulta CETPP em outras bases de dados

Nesta pesquisa, utilizamos a base de dados do *Quot* para representar os objetos de interesse e a base de dados *OSM* para representar os objetos de referência. Entretanto, não foram realizadas avaliações em bases de dados que em média, tenha uma quantidade maior de termos para descrever um objeto de referência.

A base de dados do *Twitter* pode ser utilizada para representar os objetos de referência. Uma porcentagem das mensagens dos usuários contém informações geográficas, ou seja, uma postagem é um objeto espaço-textual que pode ser utilizado como objeto de referência. Uma base de dados que em média tenha uma maior quantidade de termos para descrever um objeto espacial, pode beneficiar o algoritmo *Inverted File* (Algoritmo 3), porque a partir das palavras-chave de busca são encontrados os objetos de referência que contém os termos, não sendo necessário percorrer todo o conjunto de objetos de referência. Quanto maior a quantidade de termos na base de dados, é mais custoso realizar o cálculo de similaridade textual entre os objetos de referência e as palavras-chave de busca, portanto, o algoritmo que utiliza menos o cálculo de similaridade textual terá um maior desempenho ao processar a consulta *CETPP*.

Consulta CETPP com GPU

You et al. (2013) utiliza a *Graphics Processing Unit* (GPU) com o objetivo de processar de forma eficiente consultas que utilizam estruturas *R-tree*. Além disso, You et al. (2013) realiza experimentos com consultas espaciais no formato do processamento serial utilizando 1 core, vários cores de uma mesma *CPU* e utilizando uma *GPU* [You et al. 2013]. Podem ser estudados e implementados algoritmos para processar a consulta *CETPP* com o objetivo de ter beneficio na utilização de vários cores. Pode ser usada a estratégia de dividir a responsabilidade de processar o conjunto de objetos de interesse P por cada *core*.

Consulta CETPP em Sistemas Distribuídos

Um sistema centralizado ao mesmo tempo em que é crucial para o funcionamento do sistema, também é seu principal ponto fraco, já que se trata de um ponto único de falha e um problema ocasionado com o hardware que presta este serviço põe a perder todo o funcionando do sistema. Alguns benefícios podem ser alcançados em processar a consulta *CETPP* em sistemas distribuídos: 1) aumento da eficiência da consulta; 2) distribuição da base de dados entre os computadores que compõe a rede; 3) atender de forma simultânea várias consultas *CETPP*. Para a implementação do algoritmo para processar a consulta *CETPP* em sistemas distribuídos, pode ser utilizada a estratégia de dividir a responsabilidade de processar o conjunto de objetos de interesse P pelas máquinas presentes no sistema distribuído que forma o sistema.

Consulta CETPP em estradas de rede

Na consulta *CETPP*, a distância espacial entre um objeto de interesse e um objeto de referência é dada pela equação de *Haversine* (Equação 2.10), que considera a curvatura da terra para calcular a distância espacial. Mas na realidade, é comum que a distância entre dois objetos espaciais sejam separadas por ruas ou estradas. A consulta *CETPP* recebe o parâmetro espacial que corresponde ao raio de interesse da consulta, sendo necessário que seja desenvolvido um algoritmo que obtenha uma valor numérico a partir da rota entre um objeto de interesse e referência, posteriormente verifique se este valor atende ao critério do raio de interesse da consulta. Cao et al. (2014) utiliza técnicas para obter um valor a partir da rota entre os objetos que será utilizado no cálculo da distância *Euclidiana* [Cao et al. 2014].

Consulta CETPP em SGBDs

Grande parte das bases de dados estão armazenadas em gerenciadores de banco de dados (SGBDs). Com o objetivo de processar a consulta *CETPP* em diferentes cenários, podem ser estudadas as modificações necessárias para processar a consulta *CETPP* em *SGBDs*. Utilizar SGBDs para processar a consulta *CETPP* traz consigo os benefícios de utilizar estruturas que são amplamente pesquisadas no meio científico. Os *SGBDS* implementam mecanismos para armazenar e manipular informações de forma eficiente.

O *SQBD PostgreSQL*[1] permite armazenar objetos espaciais com o tipo de dados *point*. O tipo *point* armazena dados bidimensional (e.g. latitude e longitude). Ar-

[1] `https://www.postgresql.org/`

mazenados os objetos espaciais é possível aplicar funções geométricas presentes no próprio *PostgreSQL*. Como trabalho futuro, poderia verificar a implementação de um algoritmo para processar a consulta *CETPP* utilizando instruções *SQL*, com o objetivo de analisar o desempenho da consulta *CETPP* em *SGBDs*.

.

Referências Bibliográficas

[cla] Aplicacoess cetpp.

[Allman 1889] Allman, G. (1889). *Greek Geometry from Thales to Euclid*. Dublin University Press series. Hodges, Figgis, & Company.

[Arge et al. 2008] Arge, L., Berg, M. D., Haverkort, H., e Yi, K. (2008). The priority r-tree: A practically efficient and worst-case optimal r-tree. *ACM Trans. Algorithms (TALG)*, 4(1):9:1–9:30.

[Bayer e Mccreight 1972] Bayer, R. e Mccreight, E. M. (1972). Organization and maintenance of large ordered indexes. *Acta Inf.*, 1(3):173–189.

[Beckmann et al. 1990] Beckmann, N., Kriegel, H.-P., Schneider, R., e Seeger, B. (1990). The r*-tree: An efficient and robust access method for points and rectangles. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 322–331.

[Bercken et al. 2001] Bercken, J. V. d., Blohsfeld, B., Dittrich, J.-P., Krämer, J., Schäfer, T., Schneider, M., e Seeger, B. (2001). Xxl - a library approach to supporting efficient implementations of advanced database queries. In *Proceedings of the 27th International Conference on Very Large Data Bases*, VLDB '01, pp. 39–48, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Cammert et al. 2003] Cammert, M., Heinz, C., Krämer, J., Schneider, M., e Seeger, B. (2003). A status report on xxl - a software infrastructure for efficient query processing. *Bulletin of the Technical Committee on Data Engineering (IEEE-CS)*, 26(2):12–18.

[Cao et al. 2014] Cao, X., Cong, G., Jensen, C. S., e Yiu, M. L. (2014). Retrieving regions of interest for user exploration. *Proceedings of the VLDB Endowment (PVLDB)*, 7(9):733–744.

[Chen et al. 2013] Chen, L., Cong, G., Jensen, C. S., e Wu, D. (2013). Spatial keyword query processing: an experimental evaluation. In *Proceedings of the 39th international conference on Very Large Data Bases*, PVLDB'13, pp. 217–228. VLDB Endowment.

[Chen et al. 2006] Chen, Y.-Y., Suel, T., e Markowetz, A. (2006). Efficient query processing in geographic web search engines. In *Proceedings of the 2006 ACM*

SIGMOD International Conference on Management of Data, SIGMOD '06, pp. 277–288, New York, NY, USA. ACM.

[Cho e Chung 2007] Cho, H.-J. e Chung, C.-W. (2007). Indexing range sum queries in spatio-temporal databases. *Information and Software Technology (IST)*, 49(4):324–331.

[Choi et al. 2012] Choi, D.-W., Chung, C.-W., e Tao, Y. (2012). A scalable algorithm for maximizing range sum in spatial databases. *Proceedings of the VLDB Endowment (PVLDB)*, 5(11):1088–1099.

[Cong et al. 2009] Cong, G., Jensen, C. S., e Wu, D. (2009). Efficient retrieval of the top-k most relevant spatial web objects. *Proceedings of the VLDB Endowment (PVLDB)*, 2(1):337–348.

[Cormen et al. 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2009). *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition.

[de Almeida e Durão 2018] de Almeida, J. P. D. e Durão, F. A. (2018). Improving the spatial keyword preference query with linked open data. In *Brazilian Symposium on Multimedia and the Web (WebMedia)*, pp. 19–24.

[de Almeida e Rocha-Junior 2016] de Almeida, J. P. D. e Rocha-Junior, J. B. (2016). Top-k spatial keyword preference query. *Journal of Information and Data Management (JIDM)*, 6(3):162.

[Du et al. 2005] Du, Y., Zhang, D., e Xia, T. (2005). The optimal-location query. In *Proceedings of the 9th International Conference on Advances in Spatial and Temporal Databases*, SSTD'05, pp. 163–180, Berlin, Heidelberg. Springer-Verlag.

[Ganeshan et al. 2010] Ganeshan, K. V. V., Sarda, N. L., e Gupta, S. (2010). Keyword search in geospatial database. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pp. 538–539, New York, NY, USA. ACM.

[Gao et al. 2016] Gao, Y., Wang, Y., e Yi, S. (2016). Preference-aware top-k spatio-textual queries. In Song, S. e Tong, Y., editors, *Web-Age Information Management*, pp. 186–197, Cham. Springer International Publishing.

[Guttman 1984] Guttman, A. (1984). R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, SIGMOD '84, pp. 47–57, New York, NY, USA. ACM.

[Korn e Muthukrishnan 2000] Korn, F. e Muthukrishnan, S. (2000). Influence sets based on reverse nearest neighbor queries. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, pp. 201–212, New York, NY, USA. ACM.

[Li et al. 2011] Li, Z., Lee, K. C. K., Zheng, B., Lee, W.-C., Lee, D., e Wang, X. (2011). Ir-tree: An efficient index for geographic document search. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 23(4):585–599.

[Papadias et al. 2001] Papadias, D., Kalnis, P., Zhang, J., e Tao, Y. (2001). Efficient olap operations in spatial data warehouses. In *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases*, SSTD '01, pp. 443–459, London, UK, UK. Springer-Verlag.

[Prodanov e de Freitas 2013] Prodanov, C. C. e de Freitas, E. C. (2013). *Metodologia do Trabalho Científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico-2ª Edição*. Editora Feevale.

[Robertson e Jones 1976] Robertson, S. E. e Jones, K. S. (1976). Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146.

[Rocha-Junior et al. 2011] Rocha-Junior, J. a. B., Gkorgkas, O., Jonassen, S., e Nørvåg, K. (2011). Efficient processing of top-k spatial keyword queries. In *Proceedings of the 12th International Conference on Advances in Spatial and Temporal Databases*, SSTD'11, pp. 205–222, Berlin, Heidelberg. Springer-Verlag.

[Rocha-Junior et al. 2010] Rocha-Junior, J. a. B., Vlachou, A., Doulkeridis, C., e Nørvåg, K. (2010). Efficient processing of top-k spatial preference queries. *Proc. VLDB Endow.*, 4(2):93–104.

[Salton 1973] Salton, G. (1973). Recent studies in automatic text analysis and document retrieval. *Journal of the ACM (JACM)*, 20(2):258–278.

[Salton e Buckley 1988] Salton, G. e Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

[Sinnott 1984] Sinnott, R. W. (1984). Virtues of the haversine. *Sky and Telescope*, 68:159.

[Valiense de Andrade e B Rocha-Junior 2018] Valiense de Andrade, C. M. e B Rocha-Junior, J. (2018). Encontrando os locais de interesse com maior popularidade a partir do critério espacial e textual. *Revista de Sistemas e Computação-RSC*, 8(2).

[Wu et al. 2012a] Wu, D., Cong, G., e Jensen, C. S. (2012a). A framework for efficient spatial web object retrieval. *The VLDB Journal*, 21(6):797–822.

[Wu et al. 2012b] Wu, D., Yiu, M. L., Cong, G., e Jensen, C. S. (2012b). Joint top-k spatial keyword query processing. *IEEE Transactions on Knowledge and Data Engineering*, 24(10):1889–1903.

[Xia et al. 2005] Xia, T., Zhang, D., Kanoulas, E., e Du, Y. (2005). On computing top-t most influential spatial sites. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, pp. 946–957. VLDB Endowment.

[Yiu et al. 2007] Yiu, M. L., Dai, X., Mamoulis, N., e Vaitis, M. (2007). Top-k spatial preference queries. In *2007 IEEE 23rd International Conference on Data Engineering*, pp. 1076–1085.

[Yiu et al. 2011] Yiu, M. L., Lu, H., Mamoulis, N., e Vaitis, M. (2011). Ranking spatial data by quality preferences. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 23(3):433–446.

[You et al. 2013] You, S., Zhang, J., e Gruenwald, L. (2013). Parallel spatial query processing on gpus using r-trees. In *Proceedings of the 2Nd ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data*, BigSpatial '13, pp. 23–31, New York, NY, USA. ACM.

[Zhang et al. 2006] Zhang, D., Du, Y., Xia, T., e Tao, Y. (2006). Progressive computation of the min-dist optimal-location query. In *Proceedings of the 32Nd International Conference on Very Large Data Bases*, VLDB '06, pp. 643–654. VLDB Endowment.

[Zheng et al. 2015] Zheng, K., Su, H., Zheng, B., Shang, S., Xu, J., Liu, J., e Zhou, X. (2015). Interactive top-k spatial keyword queries. In *2015 IEEE 31st International Conference on Data Engineering*, pp. 423–434.

[Zobel e Moffat 2006] Zobel, J. e Moffat, A. (2006). Inverted files for text search engines. *ACM Computing Surveys (CSUR)*, 38(2).