Universidade Federal da Bahia
Instituto de Matemática

Programa de Pós-Graduação em Ciência da Computação

EXPLOITING OPEN DATA FOR IMPROVING SPATIAL KEYWORD QUERIES APPLICATIONS

João Paulo Dias de Almeida

QUALIFICAÇÃO DE DOUTORADO

Salvador
1 de julho de 2019

JOÃO PAULO DIAS DE ALMEIDA

EXPLOITING OPEN DATA FOR IMPROVING SPATIAL KEYWORD QUERIES APPLICATIONS

Esta Qualificação de Doutorado foi apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

Orientador: Frederico Araújo Durão

Salvador
1 de julho de 2019

RESUMO

to do

Palavras-chave: consulta espacial, linked data, LOD, personalização

ABSTRACT

to do

Keywords: spatial query, linked data, LOD, personalization

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTA DE SIGLAS

Chapter 1

INTRODUCTION

Chapter 2

DATABASE QUERIES

This chapter introduces core concepts to understand the topics discussed in this research. We discuss textual queries followed by spatial queries and indexes. After that, we present preference query and its extensions.

2.1 TEXTUAL QUERIES

Textual query is a key technology to search engines (ZOBEL; MOFFAT, 2006). A user can type one or more keywords in a textual query to describe the document she wants to retrieve (MANNING et al., 2008). This query searches and retrieve information from textual collections, returning to the user the documents relevant to her query keywords (SALMINEN; TOMPA, 1994). Web search engines and desktop search systems are examples of daily applications which employ textual queries.

A textual database is a collection of textual data like Web pages, encyclopedias, academic publications, or e-mails. Each element from a textual database is called *Document*. Accordingly to Manning et al. (MANNING et al., 2008), Document is any unit which is chosen to build a Retrieval System. In a typical Textual Information Retrieval System, a user describes the document she desires using a set of keywords (also known as "bag of words") (ZOBEL; MOFFAT, 2006).

Example. Based on the textual database described in Figure 2.1, a user can identify a document she is interested in using textual queries. In this scenario, the system considers each line as a Document. Therefore, when a user types a set of keywords in a textual query, this query returns all documents inside the textual database that are relevant to the query keywords. As an example, whether the user types the keyword big*, the textual query returns the Documents 2 and 3, because they contain the keyword.*

Inverted Files (IF) are commonly used to process textual queries efficiently (ZOBEL; MOFFAT, 2006). Create an IF requires to extract the terms from each document in a textual database. For this purpose, a pre-processing stage named *parsing* is applied.

Parsing is realized in two stages: *casefold* and *stop words* removal. The casefold converts every letter in a document to lowercase letters. Applying the casefold in Document 1, one obtains "*the old night keeper keeps the keep in the town*" as a result.

1 The old night keeper keeps the keep in the town
2 In the big old house in the big old gown.
3 The house in the town had the big old keep
4 Where the old night keeper never did sleep.
5 The night keeper keeps the keep in the night
6 And keeps in the dark and sleeps in the light.

Figure 2.1 Textual database *Keeper*. Each text line represents a document. Source: Zobel and Moffat (ZOBEL; MOFFAT, 2006).

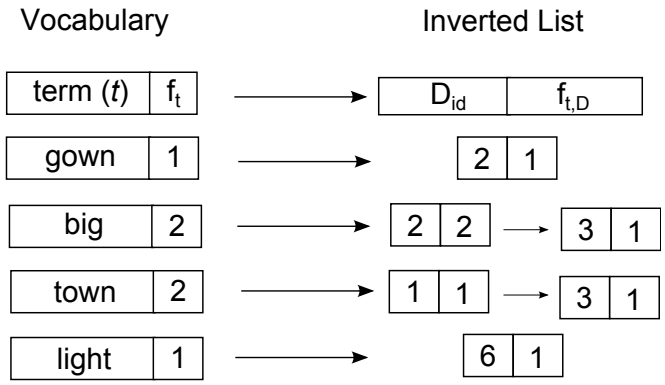

Figure 2.2 Inverted File example using existing terms in textual database *keeper*.

All languages have words that frequently occur in texts produced for that language, or words whose function only is to identify a grammar relationship. These words are known as *stop words*. Removing the stop words, and applying the casefold, one obtains the following terms from Document 1: "*old night keeper keeps keep town*" (ZOBEL; MOFFAT, 2006). Observe that parsing a document reduces the document size considerably, facilitating the storage and organization process.

The IF is composed of vocabulary (also known as dictionary of terms) and a set of inverted lists (referred as postings list too). Moreover, each term t in a collection has a corresponding inverted list. This list contains an identifier (D_{id}) for each document (D[1]) that contains the term t in its textual description. D_{id} is followed by a integer value representing the frequency $f_{t,D}$ of a term t occurs in a Document's textual description. The vocabulary stores a number f_t of documents which contain the term t and a pointer to the inverted list correspondent to the term t (ZOBEL; MOFFAT, 2006).

Example. Figure 2.2 illustrates a part of a Inverted File (IF) generated from the textual database keeper *(Figure 2.1). This IF contains the terms* gown, big. town, *and* light. *The vocabulary stores terms, the number of documents containing the stored terms, and a pointer to the inverted list related to the term (represented by the unidirectional arrow). The inverted list stores one tuple for each document which contains a term t, this tuple is composed by the document identifier (D_{id}) and the occurrence frequency($f_{t,D}$) of*

[1]Each line in Figure 2.1 represents a document D while the text inside the line represents the textual description of D.

the term t in D.

A Textual Information Retrieval System, which employs textual relevance to retrieve documents, uses a *ranking* to order the possible documents to be presented to the user. In order to create a *ranking*, a similarity measure, or heuristic, is applied to indicate the similarity between the document and the query keywords defined by the user (ZOBEL; MOFFAT, 2006).

The Information Retrieval community widely use the cosine similarity as an effective formulation of the similarity between a document and a set of query keywords (COHEN; RAVIKUMAR; FIENBERG, 2003; ZHU et al., 2011; ZOBEL; MOFFAT, 2006). Given a textual query T, composed by a set of terms t ($t \in T$), the cosine similarity $\theta(T, D)$ defines the cosine angle, in a n-dimensional space, between the weight vector[2] and the textual description of document D. As a result, a document D is a possible answer to the user only when exists at least one term $t \in T$ which exists in D too ($\exists t \in T : t \in D$).

Zobel and Mofat (ZOBEL; MOFFAT, 2006) propose the following metrics to calculate the cosine between a document and a query:

- the frequency $f_{t,D}$ of term t in the textual description of D
- the frequency $f_{t,T}$ of term t in the query T
- the number f_t of documents containing the term t
- the total number N of documents in the collection

There are many variations of the cosine similarity formulation (MANNING et al., 2008; ROCHA-JUNIOR, 2012). In this thesis, we use the formulation proposed by Zobel and Mofat (ZOBEL; MOFFAT, 2006), presented in Equation 2.1 which employs the metrics described early.

$$\theta(T, D) = \frac{\sum_{t \in T} w_{t,D} \cdot w_{t,T}}{\sqrt{\sum_{t \in D} (w_{t,D})^2 \cdot \sum_{t \in T} (w_{t,T})^2}} \tag{2.1}$$

The term weight t in document D ($w_{t,D}$) is defined by $w_{t,D} = 1 + ln f_{t,D}$, while the weight $w_{t,T}$ of term t in a query T is $w_{t,T} = ln\left(1 + \frac{N}{f_t}\right)$. The greater the value of $\theta(T, D)$, the greater is the textual relevance between the document D and the query T. Consequently, $\theta(T, D)$ is also known as the textual score of D related to the query T.

Additionally, $w_{t,T}$ represents a property usually described as *inverse document frequency* (IDF), while $w_{t,D}$ is the *term frequency* (TF). For this reason, the formulation described by the Equation 2.1 is also described in the literature as TFxIDF (ZOBEL; MOFFAT, 2006).

A textual query must generate a *ranking* containing the documents to return to the user. Figure 2.3 exemplifies a textual query processing. Initially, each document has textual score equals zero while a sum array A, of size N, is created to sum the partial

[2]The weight vector is formed by the terms weight t in T.

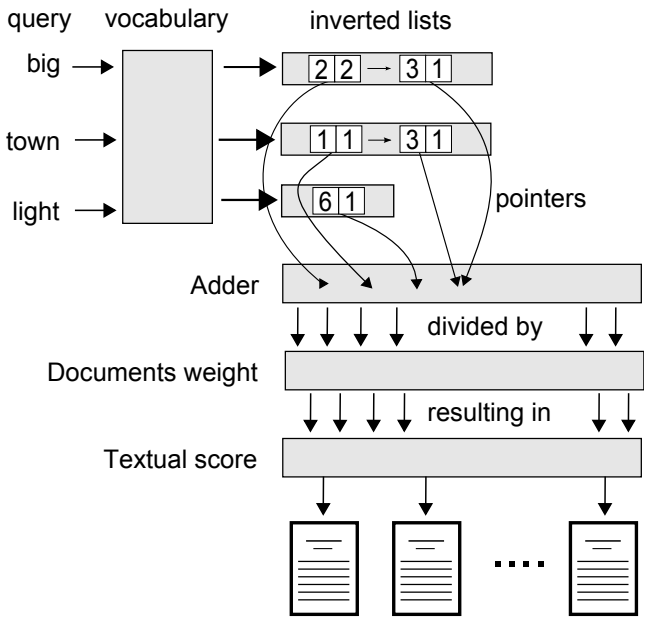

Figure 2.3 Textual query execution example using an Inverted File (IF). Source: Adapted of Zobel and Mofat (ZOBEL; MOFFAT, 2006).

textual score of each document. Each array position A_D stores the partial textual score of a document D.

Provided those definitions, each term $t \in T$ contributes $w_{t,D} \cdot w_{t,T}$ (Equation 2.1) to the similarity between a query T and a document D. The A_D position of the sum array (represented by "Adder" in Figure 2.3) stores the summation value obtained from $\sum_{t \in T} w_{t,D} \cdot w_{t,T}$.

The textual score of a document D is the division of its partial score in A_D by the document weight (W_D), $W_D = \sqrt{\sum_{t \in D} (w_{t,D})^2 \cdot \sum_{t \in T} (w_{t,T})^2}$ (Equation 2.1). Lastly, the documents are ordered by their respective textual scores and then presented to the user.

2.2 SPATIAL QUERIES

Databases adapted themselves to store and organize different types of data efficiently. The spatial data availability associated with technologies advancement made possible a scenario where spatial data is the core of many applications (RIGAUX; SCHOLL; VOISARD, 2001). Today, any individual using a smartphone, is a potential spatial data provider due to the popularization of the Global Positioning System (GPS).

Satellite images, medical equipment, or Geographic Information System (GIS) are other sources of spatial data which provide a large amount of data. Unfortunately, manipulate this volume of data is expensive and impractical to users who don't have proper computational tools. This task becomes even more difficult when is required to analyze the data in details.

A spatial database system offers support to spatial objects like points, lines, and polygons (GÜTING, 1994; RIGAUX; SCHOLL; VOISARD, 2001). This system provides

additional support to spatial data modeling and spatial queries description. In order to process spatial queries efficiently, the spatial database system employs spatial indexes (GÜTING, 1994).

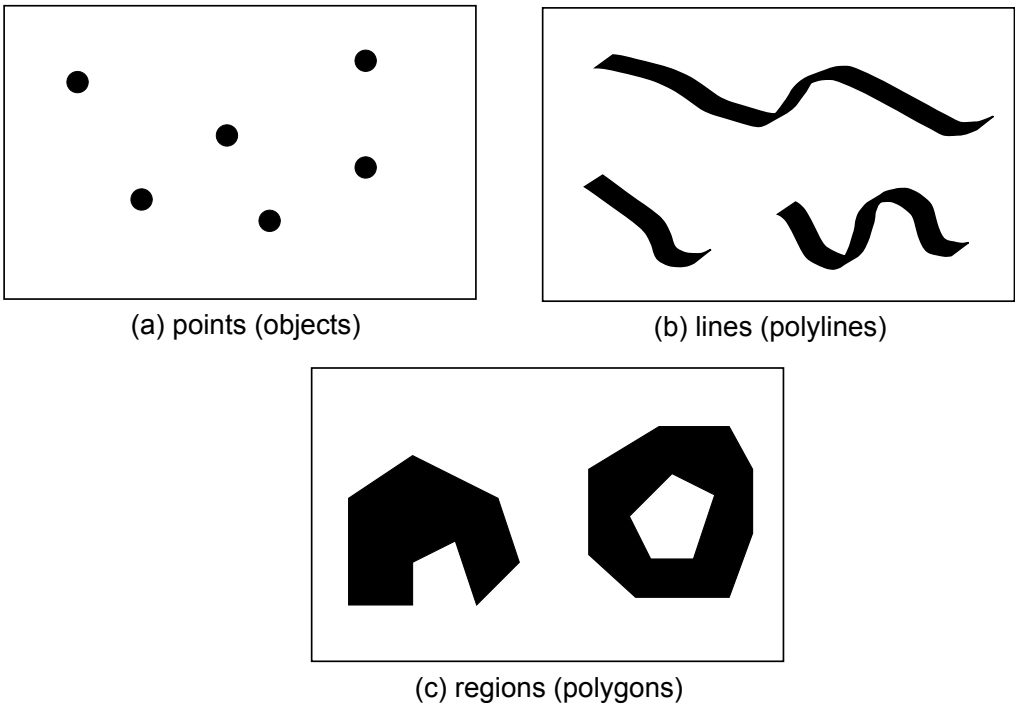

Figure 2.4 Basic spatial objects. Source: Adapted of Rocha-Junior (ROCHA-JUNIOR, 2012).

Spatial objects. Figure 2.4 presents some of the basic spatial objects: points (objects), lines, and regions (polygons). A point (Figure 2.4(a)) represents an object whose area is not relevant, only its spatial location. For instance, a point can represent a reference object location (i.e. restaurant) or a person location. A line (Figure 2.4(b)) can represent a river, a road, or power lines. Besides, it is important to notice that lines can intersect other lines. At long last, a region (Figure 2.4(c)) usually is modeled like a polygon and can describe spatial objects whose spatial area is relevant like a farm or a forest. Regions are disjoint; however, they can have holes or can be composed of many disjoint pieces (GÜTING, 1994; ROCHA-JUNIOR, 2012).

Because of the large volume of spatial data available for search, the popularity of spatial queries increase. Among the most important types of spatial queries employed in spatial databases are the spatial selections based on predicates (GÜTING, 1994; ROCHA-JUNIOR, 2012). Given a database, a spatial selection returns the set of objects which satisfies the predicate. This predicate can be represented by one or more spatial relationships - the most significant operation provided by the spatial algebra (GÜTING, 1994). These spatial relationships can be topological (i.e. adjacency, disjunction), directional (i.e. above, below, to the left), and metric (i.e. distance), among others. The sentence "find all restaurant in a 100m radius from my actual location" is an example of spatial selection.

In this work, we direct our focus to one spatial selection in particular: the *range*.

Range. Given the query location $q.l$ and the distance $dist(p, q.l)$ (euclidian distance between $q.l$ and an object p) , the spatial query *range* retrieve all p objects whose distance values are smaller than the radius r, $dist(p, q.l) \leq r$ (ROCHA-JUNIOR, 2012; YIU et al., 2007). Therefore, r defines the query's spatial neighborhood.

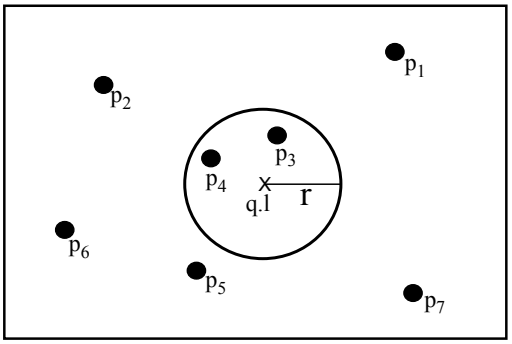

Figure 2.5 Spatial selection example: range. Source: Rocha-Junior (ROCHA-JUNIOR, 2012).

Figure 2.5 illustrates a range query where $q.l$ is the query location and r is the interest radius. Processing this query on the spatial area illustrated in Figure 2.5 returns the points p_3 and p_4 as result.

2.2.1 Spatial Indexes

A spatial database system requires a mechanism to improve the spatial objects retrievement, taking in consideration their locations and the user's need (GUTTMAN, 1984). In order to assist in this task, several researchers proposed many spatial indexes (BECKMANN et al., 1990; GUTTMAN, 1984; PAPADIAS et al., 2001; SAMET, 1984). We present some of these spatial indexes in this subsection.

The R-tree is a balanced tree, almost identical to a B-tree (BARUFFOLO, 1999; BAYER; MCCREIGHT, 1970; COMER, 1979) whose leaves have pointers to space-textual objects. R-tree is dynamic; hence, insertion and removal of elements can be performed in conjunction with queries without having to reorganize the tree periodically (GUTTMAN, 1984). In addition, R-tree nodes are generally the size of a disk page, and their structure is designed to search only a small number of nodes. Thus, each node of the R-tree has a minimum and a maximum number of entries (GUTTMAN, 1984; ROCHA-JUNIOR, 2012).

There are two types of nodes in an R-tree: intermediate nodes and leaf nodes. The intermediate node contains pointers to the descendant nodes, while the leaf nodes have pointers to the indexed objects. The entries of an R-tree are formed by (MBR, *id*). Minimum Bounding Rectangle (MBR) is an n-dimensional rectangle surrounding the indexed object, and *id* is a number that identifies the input. The *id* of an intermediate node is a pointer (address) to another node in the tree (descendant node), while the MBR of an intermediate entry involves the MBRs of all entries in the child node. In the

input of a leaf node, *id* is the identification of the object in the database and the MBR is the smallest possible n-dimensional rectangle that can wrap the indexed spatial object (ROCHA-JUNIOR, 2012).

Figure 2.6(a) is the representation of a spatial area where objects (p) are indexed in a R-tree. Under those circumstances, $q.l$ is the query location, r defines the spatial neighborhood of $q.l$, and m_1, m_2, m_3, and *root* are the MBRs. On the side, in Figure 2.6(b), the root is a intermediate node that has three intermediate entries m_1, m_2, m_3 which point to the leaf nodes n_1, n_2, n_3, respectively. The intermediate entry $(m_1, *n_1)$ contains the MBR m_1 that involves all stored objects in node n_1, and a pointer $*n_1$ pointing to the node n_1. The leaf node n_1 contains two leaf entries: $(m_{p1}, *p_1)$ and $(m_{p3}, *p_3)$, where m_{p1} is the MBR involving the spatial object p_1 and $*p_1$ is the pointer (identifier) to object p_1 in the database.

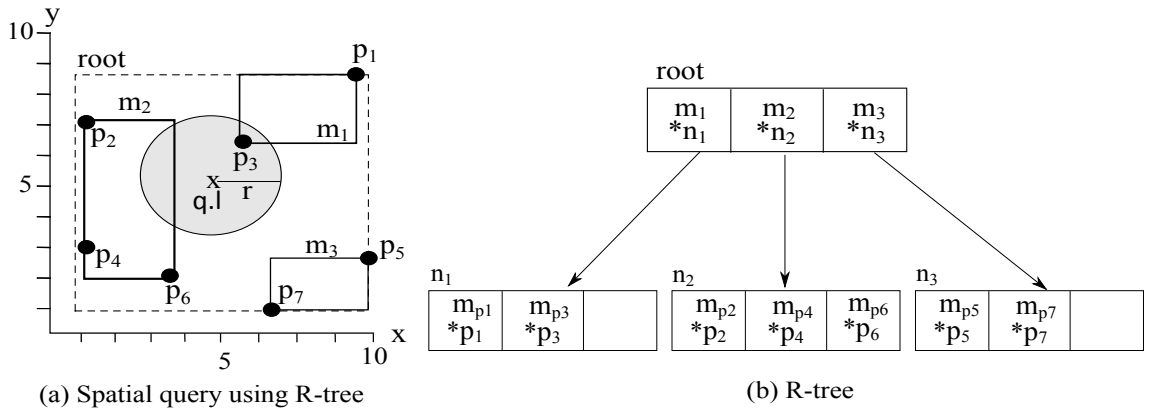

Figure 2.6 R-tree examples. Adapted of Rocha-Junior (ROCHA-JUNIOR, 2012).

Example. Figure 2.6(a) presents a range query processed with a R-tree. The query searches for spatial objects inside the spatial neighborhood defined by r. In other words, the query searches for objects inside the circumference which has 'x' as the center and r as the radius. The range query starts the search in the root and then searches the entries, verifying which entry has a MBR distance to $q.l$ smaller than the size of r. Knowing that p is the nearest point in a MBR to $q.l$, $dist(p, q.l)$ defines the shortest distance of a MBR to $q.l$, this way $dist(p, q.l)$ have to be lower than r to the entry be visited. In Figure 2.6(a), two entries satisfy this condition: $(m_1, *n_1)$ e $(m_2, *n_2)$. As a result, the leaf nodes n_1 and n_2 are accessed to search for the leaf entries whose MBR[3] is inside the spatial neighborhood defined by r, returning object p_3 as consequence.

The R-tree is based on a heuristic optimization, consisting in minimize the MBR area of each intermediate node. However, this criterion proved not to be the best (BECKMANN et al., 1990). One of the most well-known variations of the R-tree is the R*-tree (CHEN et al., 2013; HARIHARAN et al., 2007; WU et al., 2012; ZHOU et al., 2005). The R*-tree is superior to the R-tree in query processing and in the algorithm that defines the MBR of the nodes (BECKMANN et al., 1990).

[3]In this case, the leaf entries are bi-dimensional points. Thereby, the upper right vertex is identical to the lower left vertex.

The R*-tree reduces the coverage area of the MBRs involving intermediate nodes. Thus, fewer tree branches are used during query processing, resulting in less access to disk pages. In addition, R*-tree reduces the overlap between MBRs, reducing the probability of having more than one MBR covering the same area and increasing the efficiency of the query (ROCHA-JUNIOR, 2012).

Another widely used variation of R-tree is the aggregate R-tree (aR-tree), proposed by (PAPADIAS et al., 2001). The main feature of aR-tree is to use pre-aggregated non-spatial data to optimize query processing. In other words, each node of an aR-tree has a non-spatial data (eg, a numeric value) added.

Example. Assume that each object p in Figure 2.6 has a non-spatial score (numeric value). In this context, a query can be made to search for objects in the spatial neighborhood defined by r and that have a score greater than 0.7. In a traditional R-tree, this query needs to be performed in two steps. Initially, all objects that are in the spatial neighborhood of q.l are selected. Then the score of each selected object is checked, and only those that have a score greater than 0.7 are returned (ROCHA-JUNIOR, 2012; PAPADIAS et al., 2001).

In order to optimize this process, each intermediate node of aR-tree stores a value that is obtained by an aggregated function applied to the child node inputs. Under those circumstances, the Max aR-tree is used because it employs the aggregated function max(). Thus, the maximum value of the score on the child nodes is added to their respective intermediate node (ROCHA-JUNIOR, 2012; PAPADIAS et al., 2001).

Figure 2.7 represents a Max aR-tree where the aggregated function max() was applied. Therefore, it is observed that the score stored at the intermediate input $(m_1, 0, 9, *n_1)$ is 0.9 because this is the highest score value between the entries of the node n_1: $(m_{p1}, 0, 5, *p_1)$ e $(m_{p3}, 0, 9, *p_3)$. The structure and the way the aR-tree query is executed are similar to that of the R-tree. However, only entries that satisfy the spatial and non-spatial conditions are visited. For example, to find objects that are in the spatial neighborhood of q.l and have a score greater than 0.7, the root is accessed for the input that satisfies these two conditions (neighborhood criterion and score). Thus, only the input $(m_1, 0, 9, *n_1)$ is visited, and the object p_3 is returned because it is the only one that has a score greater than 0.7 and has a distance to q.l lower than the size of r.

2.3 CONSULTAS PREFERENCIAIS

Consultas tradicionais realizadas em bancos de dados fornecem uma maneira rígida para definir as características dos dados que necessitam ser recuperados (LACROIX; LAVENCY, 1987); resultando em um conjunto resposta muito grande, ou muito pequeno, de dados recuperados. Por isto, o tratamento das preferências do usuário tem se tornado uma ferramenta importante nos Sistemas de Informação atuais (**??**). Estas preferências são utilizadas para filtrar a informação, reduzindo o volume de dados apresentado ao usuário (**??**).

Exemplo. A Tabela 2.1 representa um conjunto de dados H que contém informações sobre hotéis, os seus respectivos valores de diárias, e a distância destes hotéis para a praia. Assuma um usuário que não conheça o conjunto de dados e deseje localizar um

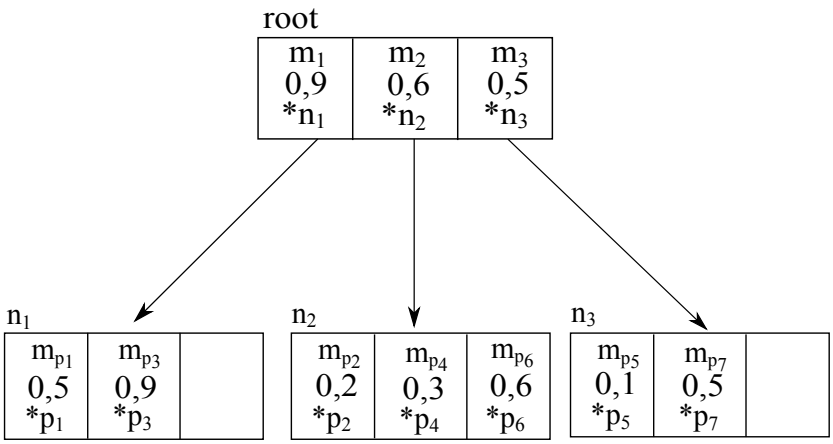

Figure 2.7 AR-tree example.

hotel barato. Utilizando consultas tradicionais, o usuário pode solicitar a lista de hotéis com valores de diária inferiores a 60. Neste caso, nenhum hotel será retornado. Em contrapartida, caso o usuário solicite a lista de hotéis com diárias superiores a 60, toda base será retornada. Desta forma, o usuário terá que percorrer toda base até encontrar o hotel que deseja, dificultando o processo de localizar o hotel mais barato.

Table 2.1 Exemplo de conjunto de dados contendo os valores das diárias dos hotéis e a distância destes hotéis para a praia. Fonte: Tabela adaptada de Rocha-Junior (ROCHA-JUNIOR, 2012).

Hotel	Diária (R$)	Distância (m)
h_1	300	50
h_2	100	100
h_3	500	100
h_4	90	300
h_5	250	500

As Consultas Preferenciais (LACROIX; LAVENCY, 1987) permitem que o usuário possa expressar as suas preferências de forma mais clara e precisa[4]. É possível solucionar o problema descrito anteriormente definindo uma consulta da seguinte forma: "selecione os hotéis com os menores valores de diária, pare depois de k" (ROCHA-JUNIOR, 2012). Assim, considerando $k = 3$ e utilizando os dados da Tabela 2.1, esta consulta Preferencial retorna os hotéis h_2, h_4, h_5.

As consultas Preferenciais podem ser classificadas pela forma como elas expressam a necessidade por informação de um usuário. A consulta Preferencial qualitativa especifica a preferência do usuário diretamente entre pares de objetos (tuplas) existentes na base de

[4]Borzsony, Kossmann e Stocker (**??**) demonstram como uma consulta preferencial pode ser implementada utilizando SQL (sem realizar modificações no sistema de banco de dados), e os motivos pelos quais tal implementação apresenta baixo desempenho quando comparado a uma consulta preferencial (implementada a partir da extensão de um sistema de banco de dados com um novo operador lógico, que representa a consulta preferencial).

dados, utilizando para isto, uma fórmula preferencial $f(h, q)$. Dado dois objetos h e q em um conjunto de dados H, a fórmula preferencial $f(h, q)$ determina se um objeto atende a necessidade do usuário. A formula preferencial $f(h, q)$ é uma operação binária entre os objetos h e q. Sendo assim, quando o resultado desta fórmula é verdadeiro (*true*), entende-se que o objeto h atende melhor às necessidades do usuário do que o objeto q. A fórmula preferencial é definida utilizando operadores lógicos (**????**).

Exemplo. Considere a base de dados representada pela Tabela 2.1 e um usuário interessado pelo hotel mais barato e mais próximo da praia. Este interesse do usuário pode ser descrito através da fórmula preferencial $f_1(h, q) = [(h[diária] \leq q[diária]) \wedge (h[distância] \leq q[distância]$. *O objeto* h_2 *satisfaz melhor a necessidade do usuário do que o objeto* h_3. *Ambos os objetos possuem a mesma distância para a praia, porém o hotel* h_2 *é mais barato. Neste contexto, é dito que* h_3 *é dominado por* h_2 *pois* $f_1(h_2, h_3) = true$. *Todos os objetos que não são dominados são respostas válidas para a consulta descrita pela fórmula* f_1.

Por outro lado, a consulta Preferencial quantitativa especifica as preferências indiretamente para cada objeto existente no conjunto de dados. Uma função escore avalia os atributos de um objeto, e produz um valor numérico (escore) que representa a importância deste objeto para as necessidades do usuário. Consultas quantitativas são frequentemente denominadas de consultas top-k. Neste tipo de consulta é especificado a função para calcular o escore dos objetos e a quantidade (k) de objetos que são recuperados da base de dados (ROCHA-JUNIOR, 2012).

Exemplo. Na base de dados H apresentada na Tabela 2.1, o hotel h_1 *pode ser representado por* $h_1 = \{300, 50\}$, *sendo que o valor 300 está posicionado na coluna (dimensão) 1 da tabela e o valor 50 na coluna 2. Pode-se utilizar a consulta Preferencial quantitativa para encontrar os 3 hotéis mais baratos e mais próximos da praia. Supondo uma função escore* $f(h) = 0,5 * h[diária] + 0,5 * h[distância]$, *onde* $h \in H$, *os objetos com menores escores são os que se aproximam mais da necessidade do usuário. Sendo assim, a função escore retorna os valores de escore* $f(h_2) = 100$, $f(h_1)$=175 *e* $f(h_3)$=195 *para os os objetos* h_2, h_1 *e* h_3, *respectivamente. Portanto, a consulta Preferencial quantitativa retorna os objetos* h_2, h_1 *e* h_3 *como resposta. Esta função escore seria considerada monótona se para todo objeto* $h_x, h_y \in H$, $f(h_x) \leq f(h_y)$ *quando* $h_x[i] \leq h_y[i]$. *Como* $f(h_2) \leq f(h_1)$ *mas* $h_2[2] > h_1[2]$ *(100 ¿ 50), esta função de escore não é considerada monótona.*

A maioria das técnicas de processamento de consultas top-k utilizam funções escore denominadas de funções *ranking* monótonas, pois estas funções possuem propriedades especiais que permitem o processamento eficiente da consulta top-k (**??**). Sendo um objeto $h \in H$ representado por $h = h[1], ..., h[n]$, onde $h[i]$ é um valor numérico na dimensão i. Uma função f_h definida sobre os atributos (dimensões) de um objeto h é monótona, se para todos os objetos $h, q \in H$, $f_h \leq f_q$ quando $h[i] \leq q[i]$ para todo i (ROCHA-JUNIOR, 2012).

2.4 CONSULTAS ESPACIAIS PREFERENCIAIS TRADICIONAIS

Os bancos de dados espaciais gerenciam grandes coleções de entidades geográficas. Cada entidade tem como característica principal as coordenadas geográficas, que indicam a

posição do objeto no espaço. Porém, em conjunto com o atributo espacial é comum existir informações não-espaciais como descrição textual, nome do objeto, tamanho ou preço deste objeto (YIU et al., 2007).

Consultas espaciais top-k retornam um conjunto de objetos espaciais que podem atender a necessidade do usuário. Entretanto, cada consulta define seu próprio conjunto de parâmetros para representar a preferência do usuário. Yiu et al. (YIU et al., 2007) apresenta um novo tipo de consulta Top-k, a consulta Espacial Preferencial Tradicional. Nesta consulta, os k melhores objetos de interesse para um usuário são definidos através da qualidade dos objetos de referência[5] (*features*) existentes na vizinhança espacial de cada objeto de interesse.

Desta forma, dado um conjunto P de objetos de interesse, a consulta Espacial Preferencial Tradicional retorna os k objetos existentes em P com os maiores escores. O escore de um objeto de interesse é definido pela qualidade dos objetos de referência (ex: cafés, restaurantes, hospitais) existentes na sua vizinhança. Nesta consulta, a qualidade de um objeto de referência (*feature*) pode ser obtida através de um sistema de classificação online, como o Booking[6] ou o Foursquare[7], onde os usuários avaliam diversos tipos de objetos de referência (YIU et al., 2007).

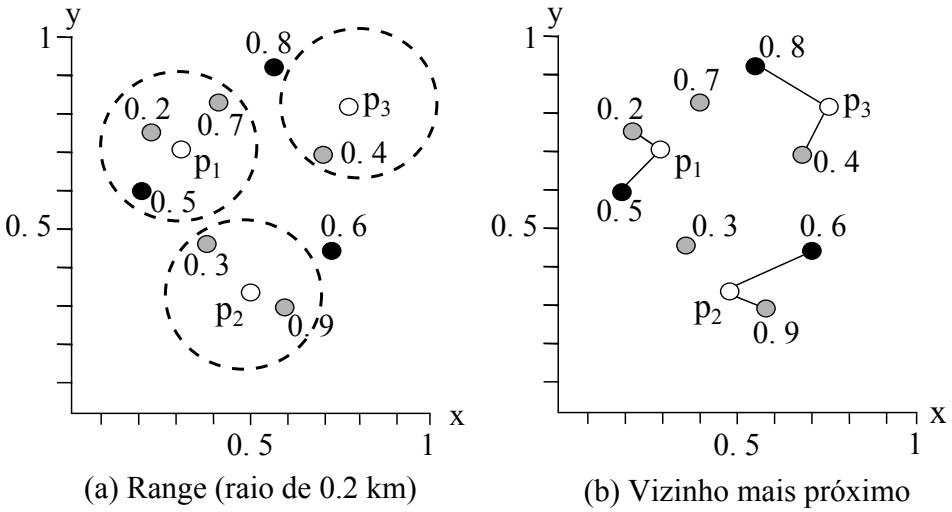

Figure 2.8 Exemplos de consultas Espaciais Preferenciais utilizando diferentes maneiras de definir a vizinhança espacial do objeto de interesse. Fonte: Yiu et al. (YIU et al., 2007).

Exemplo. Os pontos brancos p na Figura 2.8 representam objetos espaciais de interesse. Além destes, os pontos cinza representam restaurantes, enquanto os pontos pretos representam cafeterias. Cada objeto existente no conjunto de restaurantes e cafeterias (pontos pretos e cinza) possuem um valor de escore pré-definido, representado pelo número real posicionado ao redor de cada um destes pontos.

[5]Considere "objeto de referência" como uma classe de objetos em um mapa espacial, como uma instalação específica ou um serviço (YIU et al., 2007). Cada objeto de referência é associado a um escore que é pré-definido por um sistema de classificação.

[6]www.booking.com

[7]www.foursquare.com

Assumindo que o usuário deseja obter os melhores hotéis em termos de cafeterias e restaurantes, a consulta Espacial Preferencial Tradicional retorna os objetos de interesse (hotéis) com os maiores escores. Neste caso, o escore do objeto de interesse é computado pela soma do escore do melhor restaurante existente na vizinhança do objeto de interesse, com o valor da melhor cafeteria existente nesta mesma vizinhança. Sendo assim, os valores dos objetos de interesse para a consulta do tipo range são $\tau(p_1) = 0,7 + 0,5 = 1,2$ *,* $\tau(p_2) = 0,9 + 0 = 0,9$ *e* $\tau(p_3) = 0,4 + 0 = 0,4$*. Objetos de interesse que não possuem cafeterias ou restaurantes na sua vizinhança recebem o valor zero como escore deste objeto de referência ausente, como aconteceu nos casos dos objetos de interesse* p_2 *e* p_3*. Através dos resultados extraídos da Figura 2.8, obtemos o objeto* p_1 *como o resultado top-1 da consulta Espacial Preferencial Tradicional do tipo range (YIU et al., 2007).*

De forma semelhante, é calculado o escore para a consulta Espacial Preferencial Tradicional do tipo vizinho mais próximo (Figura 2.8 (b)). Portanto, nesta consulta, o escore do objeto de interesse é o escore do objeto de referência mais próximo. Assim, temos $\tau(p_1) = 0,2 + 0,5 = 0,7$*,* $\tau(p_2) = 0,9 + 0,6 = 1,5$ *,* $\tau(p_3) = 0,4 + 0,8 = 1,2$*, resultando em* p_2 *como o melhor hotel (YIU et al., 2007).*

Deste modo, a consulta Espacial Preferencial Tradicional utiliza duas etapas para selecionar seus objetos de interesse. Primeiro, calcula a distância do objeto de interesse para um determinado objeto de referência (*feature*). Em seguida, ordena os objetos de interesse a partir de sua qualidade (YIU et al., 2007).

Além desta consulta Top-k, muitas pesquisas estão sendo desenvolvidas nesta área (**??**), (**??**), (CAO et al., 2012) e (ROCHA-JUNIOR et al., 2010), demonstrando a popularidade das consultas Top-k.

2.5 CONSULTAS ESPACIAIS PREFERENCIAIS TEXTUAIS

Entre as consultas espaciais existem aquelas que utilizam palavras-chave para expressar o que o usuário deseja obter da base de dados. Nesta seção serão descritas algumas consultas que utilizam este modelo para recuperar a informação desejada, além de índices híbridos capazes de indexar dados espaciais e textuais simultaneamente. Estes índices espaço-textuais visam dar suporte a um processamento eficiente de consultas espaço-textuais.

Uma *Top-k Spatial Keyword Query* (SK) (CAO et al., 2012; CHEN et al., 2013) utiliza a localização do usuário e um conjunto de palavras-chave, fornecidas por ele, como parâmetros. A consulta identifica objetos que são espacialmente próximos a localização do usuário, e textualmente relevantes às palavras-chave; retornando os k objetos que possuem estas duas características (proximidade do usuário e relevância textual). Uma função escore avalia a proximidade espacial entre um objeto e o usuário, além da relevância textual da descrição do objeto considerando o conjunto de palavras-chave. A resposta desta consulta é ordenada a partir dos valores de escore gerados para cada objeto pela função escore.

Supondo um usuário que deseja encontrar um bar onde tenha apresentação de samba, utilizando a área espacial descrita pela Figura 2.9. Este usuário forma uma consulta q top-3 com as seguintes palavras-chave: “samba” e “bar”. A localização do usuário é a localização da consulta q na Figura 2.9.

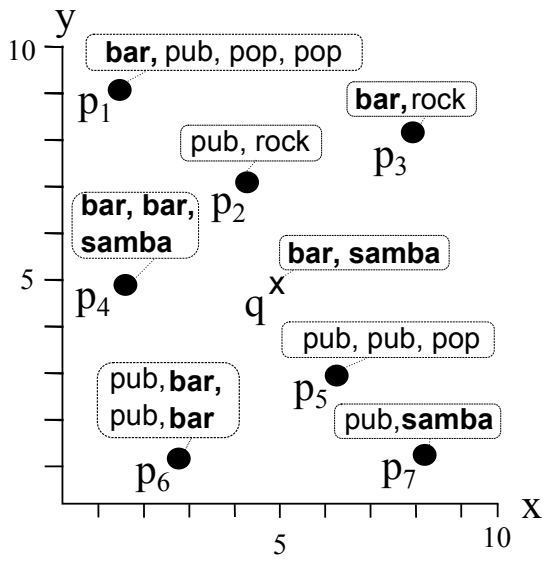

Figure 2.9 Área espacial contendo bares e pubs. Fonte: Rocha-Junior et al. (ROCHA-JUNIOR et al., 2011).

A *Top-k Spatial Keyword Query* retorna um conjunto resposta formado pelos objetos p_4, p_6, p_7. O objeto p_4 é o resultado top-1 pois a sua descrição textual é semelhante às palavras chaves fornecidas pelo usuário, e é o objeto mais próximo da localização da consulta q. p_6 possui uma relevância maior do que p_7, pois a descrição textual do p_6 é mais relevante e este objeto também fica mais próximo de q do que p_7.

2.5.1 Índices espaço-textuais

Atualmente, muitas aplicações utilizam uma grande quantidade de dados geo-referenciados, como o Twitter[8] e o Flickr[9]. Estas aplicações podem se beneficiar da *Top-k Spatial Keyword Query* (SK) e de outras consultas espaço-textuais, porém o custo do processamento destas consultas é alto (ROCHA-JUNIOR et al., 2011). Por isto, índices espaço-textuais desempenham um papel importante para o processamento destas consultas. Ao indexar dados que contém descrições textuais e geo-referenciamentos, estes índices híbridos possibilitam um processamento eficiente das consultas espaço-textuais (CHEN et al., 2013).

O índice espaço-textual Spatial-Keyword Inverted File (SKIF) proposto por Khodaei, Shahabi e Li (**??**) é um Arquivo Invertido (IF) capaz de indexar e buscar por dados espaciais e textuais de forma integrada, utilizando apenas uma estrutura para gerenciar os dois tipos de dados. Para isto, o espaço é particionado em células (Figura 2.10), onde cada célula é tratada como uma palavra-chave textual.

De forma semelhante ao IF, o SKIF é composto por um vocabulário e por um conjunto de listas invertidas. O vocabulário contém todos os termos existentes nas descrições textuais dos objetos, e identificadores para as células que formam o grid sobre a área espacial de interesse do usuário. Para cada termo distinto, os seguintes valores são armazenados

[8] www.twitter.com
[9] www.flickr.com

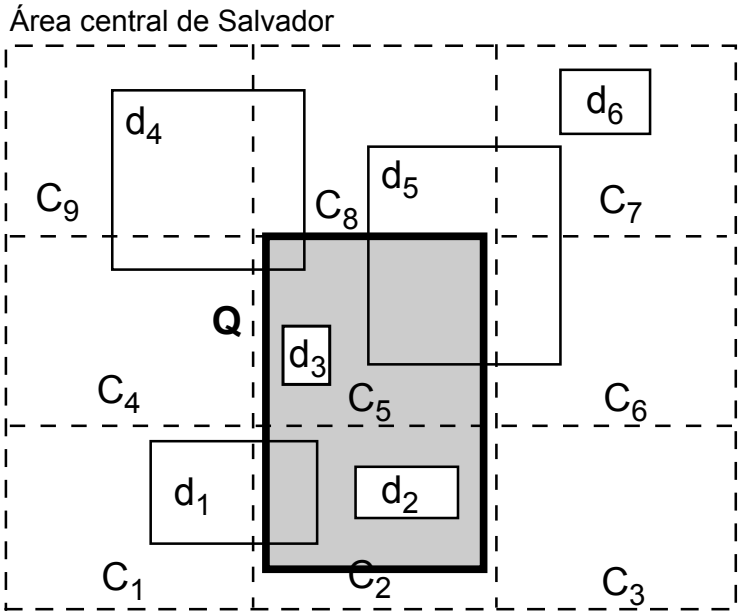

Figure 2.10 A área de busca é restrita pelo usuário (ex: "Área central de Salvador") e dividida em grid (C_i) pelo sistema para utilizar o SKIF. O retângulo escuro corresponde ao local onde a consulta foi realizada. Retângulos claros (d_i) correspondem a documentos próximos a área de interesse do usuário. Fonte: Imagem adaptada de (**??**).

no vocabulário: o número de documentos f_t que possuem o termo t, um ponteiro para lista invertida correspondente, e o tipo de termo indexado. Cada termo t possui uma lista invertida correspondente, cada uma destas listas armazenam os identificadores dos documentos que possuem o termo t e a frequência normalizada com o que termo t aparece em cada documento D (**??**).

O SKIF foi idealizado para processar uma consulta capaz de retornar os k objetos que possuem os maiores escores textuais e espaciais concomitantemente. Para o SKIF, a localização de cada objeto é tratada como uma região ao invés de um ponto. A relevância espacial é expressada pela sobreposição entre a região da consulta (região escura) e a região de um objeto (d_i) (CHEN et al., 2013). Por isto, apesar de ser um índice hibrido, o SKIF não é capaz de processar uma consulta *Top-k Spatial Keyword Query*. Recentemente, Chen et al. (CHEN et al., 2013) modificou o SKIF para processar consultas do tipo *Boolean Range Query* (BRQ).

Assim como o SKIF, muitas outras estruturas baseadas em células já foram propostas para processar objetos espaço-textuais (**????**). A seguir são discutidos outros índices híbridos capazes de indexar objetos espaço-textuais, estes índices são baseados em árvores como a R-tree proposta por Guttman (GUTTMAN, 1984) e a R*-tree proposta por Beckmann et al. (BECKMANN et al., 1990).

Cong, Jensen e Wu (CONG; JENSEN; WU, 2009) incorporam a similaridade entre documentos para produzir um novo índice espaço-textual denominado DIR-tree (*Document similarity enhanced Inverted file R-tree*). A DIR-tree combina informações espaciais e textuais durante a construção do seu índice, mantendo no mesmo nível da árvore, os ob-

jetos próximos espacialmente e maximizando a semelhança textual entre os documentos que fazem parte de uma mesma MBR (CONG; JENSEN; WU, 2009; **??**). Um parâmetro β é introduzido para determinar o peso entre a característica espacial e textual. Por exemplo, a depender do valor de β os objetos de uma mesma MBR podem ser muito próximos um do outro e possuir pouca relevância textual entre si.

Cada nó de uma DIR-tree é associado a um Arquivo Invertido (IF). Além do IF, cada nó folha da DIR-tree é associado a um sumário de documentos que disponibiliza informações textuais sobre os documentos (**??**). A Figura 2.11 apresenta objetos espaciais O_i e suas respectivas MBRs R_i. A Figura 2.12 exemplifica a organização de uma DIR-tree para indexar os objetos existentes na Figura 2.11.

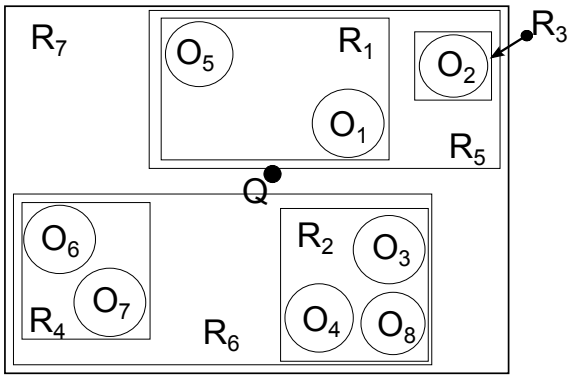

Figure 2.11 Objetos espaciais O_i e suas respectivas MBRs R_i. Fonte: (CONG; JENSEN; WU, 2009).

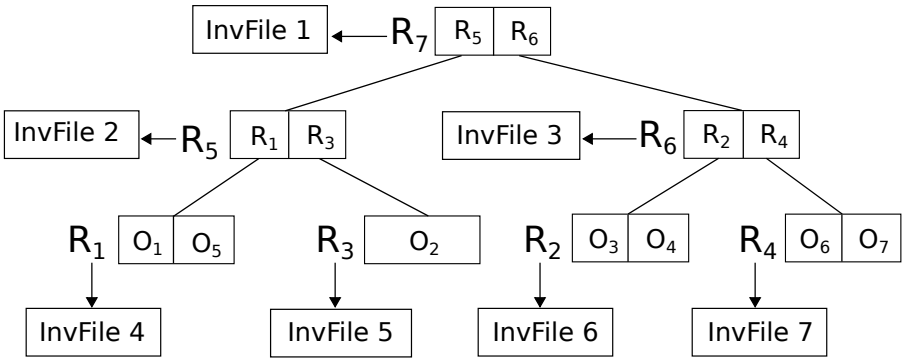

Figure 2.12 Estrutura de uma DIR-tree. Fonte: (CONG; JENSEN; WU, 2009).

Exemplo. Considere uma consulta espaço-textual que receba um conjunto de palavras-chaves T. Assuma que o objeto O_1 (Figura 2.11) é o mais relevante textualmente para este conjunto de palavras-chave. Sendo assim, para processar esta consulta Q, é necessário percorrer apenas os nós R_7, R_5 e R_1, da árvore ilustrada na Figura 2.12, para chegar ao objeto O_1 e retorná-lo como resposta.

Em alternativa, Rocha-Junior et al. (ROCHA-JUNIOR et al., 2011) propõe uma nova estrutura para indexar os dados espaço-textuais a fim de otimizar o processamento da consulta *Top-k Spatial Keyword Query* (SK). O *Spatial Inverted Index* (S2I) é semelhante ao Arquivo Invertido (ZOBEL; MOFFAT, 2006; ROCHA-JUNIOR et al., 2011), porém o armazenamento dos termos mais frequentes da coleção é feito de forma diferenciada. O S2I mapeia os termos mais frequentes da coleção para uma *aggregated R-tree* (aR-tree) (PAPADIAS et al., 2001), sendo que cada árvore armazena apenas os objetos que possuem o mesmo termo t. Enquanto os termos menos frequentes são armazenados em blocos, dentro de um arquivo; sendo que cada bloco armazena os objetos que possuem o mesmo termo t.

termo	id	df_t	*flag*	armazenamento
escola	t_1	4	árvore	aR^{t_1}
creche	t_2	3	árvore	aR^{t_2}
infantil	t_3	3	árvore	aR^{t_3}
idioma	t_4	1	bloco	(p_5)

Figure 2.13 *Spatial Inverted Index.*

O S2I (exemplificado na Figura 2.13) é composto por vocabulário, blocos de arquivo (p_i) e aR-tree's (aR^{t_i}). O vocabulário armazena cada termo distinto existente na base de dados (ex: "escola", "creche"). Para cada termo t_i, é armazenado a quantidade de documentos em que este termo ocorre df_t, um *flag* indicando em que tipo de estrutura o termo está armazenado (bloco ou árvore) e um ponteiro apontando para o bloco ou árvore que contém o termo (representado pela seta unidirecional na Figura 2.13).

Cada bloco de arquivo armazena um conjunto de objetos. Para cada objeto é armazenado a identificação deste objeto $p.id$, a localização do objeto $p.l$ e a frequência $f_{p,t}$ com que o termo t ocorre na descrição textual deste objeto.

Os nós folha da aR-tree armazenam as mesmas informações de um objeto espacial que o bloco de arquivo: $p.id$, $p.l$ e $f_{p,t}$. Os nós intermediários armazenam um *minimum bounding rectangle* (MBR) que envolve a localização espacial de todos os objetos que estão na sub-árvore. Os nós intermediários de uma aR-tree (PAPADIAS et al., 2001) são capazes de armazenar também um valor não espacial. O valor agregado aos nós intermediários de uma aR-tree é o valor máximo de $f_{p,t}$ que um objeto p pode possuir entre os objetos existentes na sub-árvore deste nó intermediário. Assim, os objetos podem ser acessados decrescentemente pelos valores de impacto do termo t na descrição textual de seu objeto ($f_{p,t}$), e da proximidade espacial (ROCHA-JUNIOR et al., 2011).

Segundo Rocha-Junior et al. (ROCHA-JUNIOR et al., 2011), os resultados obtidos utilizando o S2I demonstram que a nova estrutura é capaz de otimizar o custo da consulta, assim como o custo para atualizar um termo existente na coleção. Para consultas com apenas um termo, o S2I percorre apenas uma pequena árvore, ou bloco de arquivo. Quando a consulta possui vários termos, é necessário percorrer apenas um conjunto de

pequenas arvores, ou blocos de arquivos; dispensando o acesso a um índice invertido externo.

Chapter

3

LINKED OPEN DATA

Chapter 4

PRELIMINARY EXPERIMENTAL EVALUATION

Chapter 5

This chapter presents the research stages until now. Classes completed and publications are described, as well as, the schedule.

FINAL REMARKS

5.1 ACADEMIC LIFE

In the first two years of the Ph.D. several activities took place in order to define the research topic of the research. The activities include conferences presentations and submission of journal articles. In the early stages the focus was on the Ph.D. classes and curricular components to consolidate the theoretical background. Then, the teaching internship and the workshop presentations to the research group contributed to the communication skill development.

5.1.1 Completed Classes

Each Ph.D. class gave a unique contribution to the research. In the sequence, it is described each of these contributions.

- **MATE64 - Scientific Seminars -** Professor Christina Von Flach organized a series of seminars where several professors or students in the late stage of their research presented interesting topics in computer science. The discussion after these presentations stimulate many ideas and inspired great solutions.
- **MATD74 - Algorithms and Graphs -** This class gave an overview of algorithms complexity and techniques. Professor Tiago de Oliveira Januário presented a series of challenging computational problems which students had to solve using techniques like dynamic programming, recurrences, or greedy algorithms. In addition, a paper entitled "Robot Routing: Genetic Algorithm Applied to Travelling Salesman Problem" was written to demonstrate how to solve an NP-hard problem.
- **MATE66 - Computer Science Research Fundamentals II -** The scientific methodology is one of the most important knowledge to develop solid research. This class was lectured by Luciano Rebouças de Oliveira who motivated the students

to write scientific papers objectively and efficiently. Read and writing methods were studied based on the book "Style - Toward Clarity and Grace" by Joseph M. Williams. As a result, the students produced a paper where the teacher evaluated the students' writing skills. He simulated a journal submission process, thus all papers were evaluated using a blind review method. The paper wrote in this class was the start of the first paper published as a product of this research.

- **MATE32 - Topics on Computer Intelligence II -** Overview of machine learning algorithms focusing on automatic knowledge retrieval from datasets. Professors Ricardo Araújo Rios and Tatiane Nogueira Rios explained how to pre-process data properly and to analyze data using predictive probabilistic methods based on optimization. This class was focused on supervised learning algorithms and how to evaluate them.

- **MATE33 - Topics on Computer Intelligence III -** Professors Ricardo Araújo Rios and Tatiane Nogueira Rios taught about use clustering methods and unsupervised learning algorithms. The students were challenged with clustering problems and led to solving these problems using the main techniques available in the related literature. This class was essential to improve the skill of pattern identification on datasets.

- **MATE85 - Topics on Information Systems and Web I** - This class lectured by the advisor presented core topics on Linked Open Data and Web Semantic. The students developed projects using technologies related to this topic, like Jena and Protégé. This class did a substantial contribution to the research and software development.

- **MATA31 - Oriented Research -** This curricular component is used by the advisor to evaluate the research progress. Periodical feedbacks were given to the advisor who led the research to the correct path with suggestions and contributions.

- **MATA32 - Oriented Teaching Internship -** The advisor offered to the Ph.D. student the experience of teaching to undergraduate students. Additionally, the student gave support for lectures preparation and assisted the class on projects.

- **Classes Dispensed -** Some classes were valuable to the research but they were completed during the master degree on Programa de Pós-Graduação em Computação Aplicada (PGCA). For this reason, the following classes were dispensed by the Programa de Pós-Graduação em Ciência da Computação (PGCOMP):

  – MATE04 - Topics on Databases I

  – MATE10 - Topics on Computer Intelligence I

5.1.2 Publications and Participation in Scientific Conferences

It is important to the Ph.D. student to participate in scientific conferences and to publish the research results on relevant journals. Conferences are a channel to disclose and discuss

the research, exchanging information with other researchers from related areas. Following are described the published papers and the conferences attended.

- **WebMedia 2018 - Brazilian Symposium on Multimedia and the Web -** it is the main event of the theme in Brazil and an excellent opportunity for scientific and technical exchange among students, researchers and professionals in the areas of Multimedia, Hypermedia and Web. We published the paper (ALMEIDA; DURÃO, 2018) presenting the preliminary results of our method to enhance the SKPQ accuracy using Linked Open Data.
- **J.UCS 2018 - Journal of Universal Computer Science -** this journal is run by the J.UCS consortium consisting of research institutions from Austria, Germany, Guatemala, USA, and Pakistan. We published the paper (ALMEIDA; DURÃO; COSTA, 2018) detailing our approach to enhance the SKPQ and discussed all obtained results in the experimental evaluation.
- **AMCIS 2019 - Annual Americas Conference on Information Systems -** AMCIS is viewed as one of the leading conferences for presenting the broadest variety of research done by and for information technology academicians. Every year its papers and panel presentations are selected from over 700 submissions, and the AMCIS proceedings are in the permanent collections of libraries throughout the world. From a collaborative work we did a paper about exploiting Web features for relevance feedback. This paper has been accepted and it is in the publishing process.
- **WE.PGCOMP 2018 -** this conference is a curricular component too. After the second year as a Ph.D. student, once per year, the student has to present his research progress to three professors of the doctoral program and to an audience. The research progress is evaluated by the professors who ask questions and make great suggestions for the research.
- **ESWA 2019 - Expert Systems With Applications -** is a refereed international journal whose focus is on exchanging information relating to expert and intelligent systems applied in industry, government, and universities worldwide. The personalization approach we described to improve the SKPQ will be reported in a paper which will be submitted to this journal.
- **Data Mining and Knowledge Discovery 2020 -** the premier technical publication in the field, it is a resource collecting relevant common methods and techniques and a forum for unifying the diverse constituent research communities. We plan to submit a paper to this journal in 2020 describing new techniques and experimental evaluations.

5.2 SCHEDULE

Until now we have proposed two techniques to improve spatial keyword queries accuracy. There were several experimental evaluations demonstrating query improvement. How-

ever, there is room to further improve our work. Under those circumstances, Figure 5.1 describes the activities of this research. Activities A01 to A10 was developed before the qualification exam, while A10 to A17 will be developed before the thesis defense.

- **Activity A01 - Academic Classes -** describes the time consumed to complete all classes demanded by the doctoral program. During this time the theoretical background was consolidated. All classes have been described in Section 5.1.1
- **Activity A02 - Literature Review -** studies related to the research topic definition. Many papers were read and drafts were written during this process.
- **Activity A03 - Topic discussion -** meetings with the advisor to discuss research topics and possible solutions to these topics.
- **Activity A04 - Workshops -** meetings involving the research members of the RecSys group. Each meeting consists of members presentation about their research topics followed by practices exercises. There was a workshop about SPARQL queries and another about Jena and Web Semantic.
- **Activity A05 - First proposal: LOD enhancement to spatial queries -** studies related to the research topic definition, writing the proposal definition and define the strategies to solve the problem.
- **Activity A06 - Implementation of the first proposal -** development of the strategies or algorithms defined in the early stage.
- **Activity A07 - Preliminary experiments -** the first rounds of experiments on our approach based on LOD to improve spatial keyword queries.
- **Activity A08 - Evaluation of preliminary results -** analysis of the results with graphs and proper metrics.
- **Activity A09 - Submission to J.UCS 2018 -** the first publication inside the doctoral program. This activity consumed much time because it was the first related to this topic. In this paper, we talked about the benefits of using LOD to enhance textual descriptions of objects and how it improves spatial keyword queries. Several evaluation experiments were described and analyzed.
- **Activity A10 - Submission to WebMedia 2018 -** the first paper submitted to a conference during the doctoral research. Motivated by the late review response on the J.UCS article, in this paper, we described some of the experiments on the LOD enhancement. The conference was a pleasurable experience where was possible to discuss the research topic with experienced researchers.
- **Activity A11 - Extension of the first proposal: query personalization -** after the first publication, the work on the second proposal started. In this activity we worked on the personalization of spatial keyword queries. The experimental evaluation indicates a relevant improvement on these queries accuracy.

- **Activity A12 - Implementation of the proposal extension -** implementation of the strategies or algorithms defined in the early stage.
- **Activity A13 - Preliminary experiments of the proposal extension -** initial experiments on our approach to personalize spatial keyword queries.
- **Activity A14 - Submission to AMCIS 2019 -** in this activity a collaborative work was done, revising the article and attending the demands of the journal's reviewers.
- **Activity A15 - Submission to ESWA 2019 -** the second paper describing the personalization approach is submitted and we are waiting for the journal response.
- **Activity A16 - Qualification -** write qualification text and prepare the qualification presentation.
- **Activity A17 - New optimization research -** explore new methods to improve spatial keyword queries. New algorithms or hybrid solutions can be proposed in this stage.
- **Activity A18 - Implementation of the new optimization -** this activity will implement the strategies or algorithms defined in the previous activity.
- **Activity A19 - Experiment results -** after the development stage it is necessary to evaluate the algorithms or method developed.
- **Activity A20 - Experiment analysis -** the experimental results are analyzed and compared with previous results, as well as, with the results reported in the literature.
- **Activity A21 - Preliminary results paper -** another paper will be published introducing the new approach and the results obtained so far. It is possible to submit on paper to a journal and another to a conference in this stage.
- **Activity A22 - Thesis -** activity defined to write the thesis.
- **Activity A23 - Thesis defense -** designated to thesis presentation and research end.

The schedule of activities executed since the course enrollment together with the ones that will be executed after the qualification exam are presented in Figure 5.1.

5.3 SUMMARY

The aforementioned chapter described the student life academy until now. It was depicted the classes completed and how they contributed to the research. In addition, it is possible to see the timeline between the publications and the stages before them. On the other hand, the schedule illustrates the time left to the thesis defense and future works followed by each scheduled date.

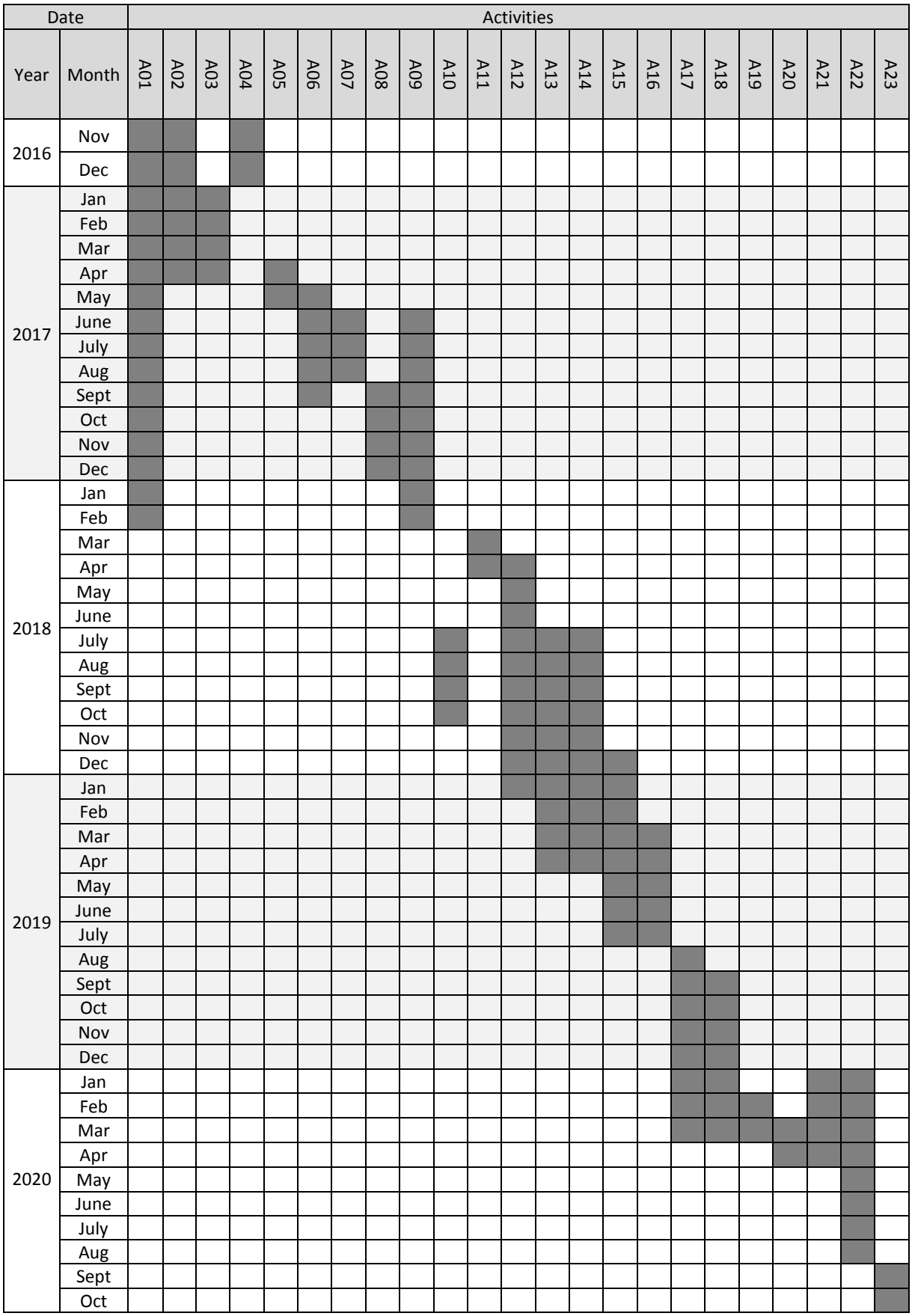

Figure 5.1 Ph.D. activities since the course enrollment until thesis defense.

BIBLIOGRAPHY

ALMEIDA, J. P. D. de; DURÃO, F. A. Improving the spatial keyword preference query with linked open data. In: SBC. *Anais Estendidos do XXIV Simpósio Brasileiro de Sistemas Multimídia e Web*. [S.l.], 2018. p. 19–24.

ALMEIDA, J. P. D. de; DURÃO, F. A.; COSTA, A. F. da. Enhancing spatial keyword preference query with linked open data. v. 24, n. 11, p. 1561–1581, nov 2018.

BARUFFOLO, A. R-trees for astronomical data indexing. In: *Astronomical Data Analysis Software and Systems VIII*. [S.l.: s.n.], 1999. v. 172, p. 375.

BAYER, R.; MCCREIGHT, E. Organization and maintenance of large ordered indexes. In: ACM-SIGFIDET. *Workshop on Data Description and Access*. Houston, Texas, 1970.

BECKMANN, N. et al. The R*-tree: An efficient and robust access method for points and rectangles. In: *SIGMOD*. [S.l.]: ACM, 1990. p. 322–331.

CAO, X. et al. Spatial keyword querying. In: *ER*. [S.l.]: Springer, 2012. p. 16–29.

CHEN, L. et al. Spatial keyword query processing: an experimental evaluation. In: . [S.l.]: VLDB Endowment, 2013. p. 217–228.

COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. E. A comparison of string distance metrics for name-matching tasks. *Workshop on Data Cleaning and Object Consolidation*, p. 73–78, 2003.

COMER, D. Ubiquitous b-tree. *ACM Computing Surveys (CSUR)*, ACM, v. 11, n. 2, p. 121–137, 1979.

CONG, G.; JENSEN, C. S.; WU, D. Efficient retrieval of the top-k most relevant spatial web objects. In: . [S.l.]: VLDB Endowment, 2009. v. 2, n. 1, p. 337–348.

GÜTING, R. H. An introduction to spatial database systems. *The VLDB Journal–The International Journal on Very Large Data Bases*, Springer-Verlag New York, Inc., v. 3, n. 4, 1994.

GUTTMAN, A. *R-trees: a dynamic index structure for spatial searching*. [S.l.]: ACM, 1984. v. 14.

HARIHARAN, R. et al. Processing spatial-keyword (sk) queries in geographic information retrieval (gir) systems. In: IEEE. *Scientific and Statistical Database Management, 2007. SSBDM'07. 19th International Conference on*. [S.l.], 2007. p. 16–16.

LACROIX, M.; LAVENCY, P. Preferences; putting more knowledge into queries. In: *VLDB*. [S.l.: s.n.], 1987. v. 87, p. 1–4.

MANNING, C. D. et al. *Introduction to information retrieval*. [S.l.]: Cambridge university press, 2008. v. 1.

PAPADIAS, D. et al. Efficient OLAP operations in spatial data warehouses. In: *SSTD*. [S.l.]: Springer, 2001. p. 443–459.

RIGAUX, P.; SCHOLL, M.; VOISARD, A. *Spatial databases: with application to GIS*. [S.l.]: Morgan Kaufmann, 2001.

ROCHA-JUNIOR, J. B. *Efficient processing of preference queries in distributed and spatial databases*. Tese (Doutorado) — Norwegian University of Science and Technology, 2012.

ROCHA-JUNIOR, J. B. et al. Efficient processing of top-k spatial keyword queries. In: *SSTD*. [S.l.]: Springer, 2011. p. 205–222.

ROCHA-JUNIOR, J. B. et al. Efficient processing of top-k spatial preference queries. In: . [S.l.]: VLDB Endowment, 2010. p. 93–104.

SALMINEN, A.; TOMPA, F. W. Pat expressions: an algebra for text search. *Acta Linguistica Hungarica*, v. 41, n. 1, p. 277–306, 1994.

SAMET, H. The quadtree and related hierarchical data structures. *ACM Computing Surveys (CSUR)*, ACM, v. 16, n. 2, p. 187–260, 1984.

WU, D. et al. Joint top-k spatial keyword query processing. *Knowledge and Data Engineering, IEEE Transactions on*, IEEE, v. 24, n. 10, p. 1889–1903, 2012.

YIU, M. L. et al. Top-k spatial preference queries. In: IEEE. *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*. [S.l.], 2007. p. 1076–1085.

ZHOU, Y. et al. Hybrid index structures for location-based web search. In: ACM. *Proceedings of the 14th ACM international conference on Information and knowledge management*. [S.l.], 2005. p. 155–162.

ZHU, S. et al. Scaling up top-K cosine similarity search. *Data & Knowledge Engineering*, Elsevier, p. 60–83, 2011.

ZOBEL, J.; MOFFAT, A. Inverted files for text search engines. In: . [S.l.]: ACM, 2006. p. 6.