



# QQESPM: spatial keyword search based on qualitative and quantitative spatial patterns

Carlos Vinicius Alves Minervino Pontes   [ Universidade Federal de Campina Grande | [carlosminervino@copin.ufcg.edu.br](mailto:carlosminervino@copin.ufcg.edu.br) ]

Claudio E. C. Campelo  [ Universidade Federal de Campina Grande | [campelo@dsc.ufcg.edu.br](mailto:campelo@dsc.ufcg.edu.br) ]

Maxwell Guimarães de Oliveira  [ Universidade Federal de Campina Grande | [maxwell@computacao.ufcg.edu.br](mailto:maxwell@computacao.ufcg.edu.br) ]

Salatiel Dantas Silva  [ Universidade Federal de Campina Grande | [salatiel@copin.ufcg.edu.br](mailto:salatiel@copin.ufcg.edu.br) ]

 Systems and Computing Department, Federal University of Campina Grande (UFCG), Av. Aprigio Veloso 882, Campina Grande, PB, 58.429-140, Brazil.

Received: 1 March 2024 • Published: 27 January 2025

**Abstract** The search for Points of Interest (POIs) based on keywords and user preferences is a daily need for many people. One way of representing this kind of search is the Spatial Pattern Matching (SPM) query, which allows for the retrieval of geo-textual objects based on spatial patterns defined by keywords and distance criteria. However, SPM is not able to represent qualitative requirements, such as connectivity relations between the searched objects. In this context, this work proposes the Qualitative and Quantitative Spatial Pattern Matching (QQ-SPM) query, which allows searches with qualitative connectivity constraints in addition to keywords and distance criteria. We also propose the QQESPM algorithm which combines an efficient use of geo-textual indexing with a top-down search strategy inspired in the Efficient Spatial Pattern Matching (ESPM) algorithm. Through a performance evaluation, QQESPM algorithm proved to be more than 1,000 times faster in average execution time than simple approaches for the QQ-SPM search. Furthermore, it achieved slower execution time growth when facing the increase of the dataset size, showcasing its efficiency and efficacy for handling geo-textual searches in a quantitative and qualitative setting.

**Keywords:** Geo-textual Object Retrieval, Spatial Keyword Search, Spatial Pattern Matching, POI Search, Topological Relations

## 1 Introduction

The daily generation and use of geo-textual data (data that has textual and geospatial attributes) has increased with the popularity of mobile devices, geolocation technologies and location-based services such as Google Maps<sup>1</sup> and Foursquare<sup>2</sup> [Chen *et al.*, 2022, 2021, 2019]. The need to process such data brings challenges with respect to processing time and storage efficiency, demanding attention to the design and implementation of efficient algorithms and indexing systems capable of handling big geo-textual data.

Points of Interest (POIs) are places with associated keywords and geospatial attributes in a spatial dataset. Generally, users need to find POIs whether through recommendation systems or keyword-based search systems [Fang *et al.*, 2019, 2018a]. One way of representing a POI group search is through a Spatial Pattern Matching (SPM) query, which is designed to identify all combinations of POIs that conform to a user-defined spatial pattern established by keywords and distance requirements [Fang *et al.*, 2018a, 2019, 2018b; Li *et al.*, 2019; Chen *et al.*, 2019]. Figure 1 (A) illustrates an example of a search scenario where a user seeks an apartment near a school and a hospital, while maintaining a certain distance from the hospital for hygiene reasons. The user's cri-

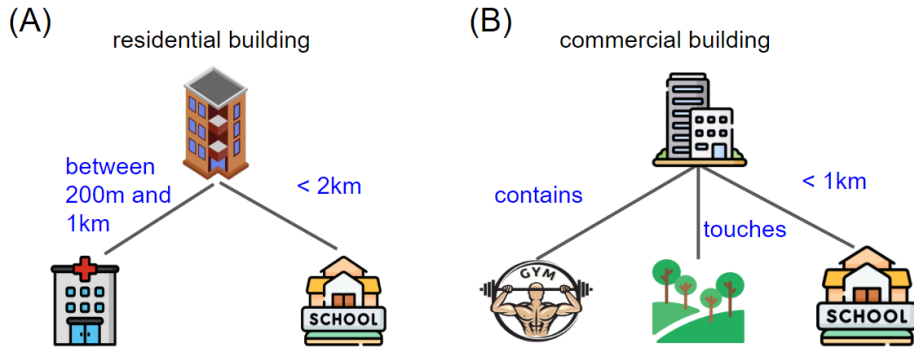
teria stipulate that the apartment should be situated between 200m and 1km away from a hospital and at most 2km away from a school.

While the SPM search methodology proves to be highly effective in scenarios necessitating distance constraints among queried POIs, it lacks the capability to address qualitative connectivity constraints between these entities. There are situations in which the user needs a more qualitative search, which cannot be fully modeled by an SPM query. For example, queries such as finding a school adjacent to a wooded area. To illustrate a more intricate search scenario, consider an individual seeking a rental space within a commercial building for their small business. Additionally, they want to have an onsite gym and a green area touching the building, which should be located within 1km away from an elementary school for the convenience of their child's enrollment and pickup. This specific scenario can be modeled using a spatial pattern graph that incorporates both quantitative (distance) and qualitative (connectivity) constraints, as shown in Figure 1 (B).

Figure 2 shows a region for the scope of a spatial pattern search containing three commercial buildings (CB1, CB2, CB3), a residential building (RB1) and two schools (S1, S2). Considering the example illustrated in Figure 1 (B), there are only two possible solutions (matches) for the given spatial pattern represented in the search graph, which are formed by

<sup>1</sup><https://www.google.com/maps/>

<sup>2</sup><https://foursquare.com/>



**Figure 1.** Example of a distance-based spatial pattern (A) and a qualitative and quantitative spatial pattern (B)

the POIs circled in red, namely (CB1, S1) and (CB1, S2). Note that POI CB2 cannot constitute a search solution as it does not have a training gym inside, while CB3 also cannot be a solution candidate as it is not adjacent to any green area. Thus, there are only two solutions for the user's search pattern in this scope region.

In response to the outlined problem scenarios, this article formalizes the Qualitative and Quantitative Spatial Pattern Matching (QQ-SPM) problem. Such formalization is built upon the SPM search from prior research [Fang *et al.*, 2018a, 2019, 2018b; Li *et al.*, 2019; Chen *et al.*, 2019] to include qualitative connectivity requirements among spatial objects, offering a comprehensive solution for group search scenarios, such as Points of Interest (POI) retrieval.

The main objective of this work is the design of the Qualitative and Quantitative Efficient Spatial Pattern Matching (QQESPM) algorithm as a solution to the QQ-SPM search problem, optimizing efficiency through early stopping conditions and computation reuse. We highlight two primary contributions: a detailed algorithmic overview coupled with a thorough mathematical formalization.

This article is an extended version of [Minervino *et al.*, 2023], presented in XVII Brazilian Symposium on GeoInformatics (GEOINFO 2023). In this manuscript we address limitations from the previous work, providing a broader experimentation scope, the proposal of enhanced algorithm features, and a more comprehensive formalization.

The structure of this article is as follows: Section 2 re-

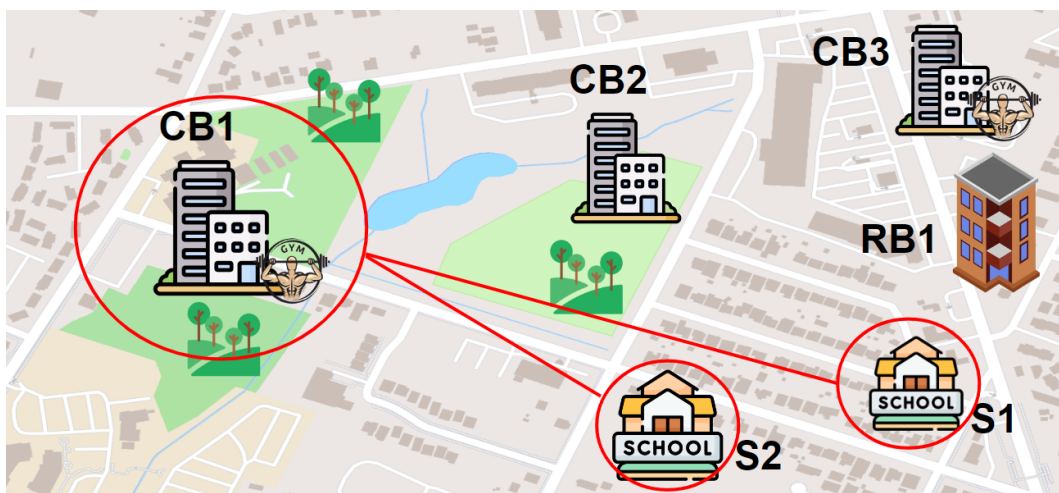
views related work. Section 3 outlines the indexing and topological relation model central to QQESPM. Section 4 provides formal definitions of QQ-SPM concepts and proofs of Lemmas useful for the designed of the QQESPM efficient search. Section 5 details the QQESPM algorithm. Section 6 presents performance experiments. Finally, Section 7 concludes and outlines future directions.

## 2 Related Work

In this section we mention some of the main types of spatial keyword queries related to this work.

The first type of search aims at finding spatial objects (e.g.: POIs) such that each of them individually meets a set of keywords and are in close proximity to a designated center point or within a search area. For instance, the top- $k$  spatial keyword search aims to identify geo-textual objects (e.g., POIs) using a set of keywords and an initial search location. The goal is to locate the top- $k$  closest objects to the starting point satisfying all search keywords. Studies in this field include those by [Cao *et al.*, 2011; Hermoso *et al.*, 2019; Zhang *et al.*, 2013].

The second type of search focuses on finding a group of geo-textual objects that collectively satisfy a set of keywords and are in close proximity to each other. For instance,  $m$ -Closest spatial keyword search seeks groups of closely located geo-textual objects that collectively satisfy a user-



**Figure 2.** Finding matches for a spatial pattern quantitative and qualitative search

defined set of  $m$  keywords. Studies in this category include those conducted by [Choi *et al.*, 2016, 2020; Guo *et al.*, 2015]. However, the first two search types are not able to represent more intricate patterns, such as specifying a minimum distance between two returned POIs, which is essential when users want to avoid close proximity to certain types of POIs.

The third search type is the SPM search, which is parameterized by a graph-based spatial pattern. In a SPM spatial pattern graph, the vertices are associated with search keywords, and the edges represent the desired distance constraints. SPM search is more powerful for certain search scenarios since it enables searches with both minimum and maximum distance restrictions between pairs of geo-textual objects. Studies in this category include works by [Fang *et al.*, 2018a; Li *et al.*, 2019; Fang *et al.*, 2018b, 2019; Chen *et al.*, 2019, 2022]. However, the SPM search lacks the capability to model qualitative restrictions, such as connectivity limitations.

In [Long *et al.*, 2016], an efficient mechanism for indexing qualitative relations is proposed, aiming to reduce the time required for calculating the qualitative relation between two geometries. The core concept involves initially computing the qualitative relation between the Minimal Bounding Rectangles (MBRs) of the spatial objects. In cases where it is not possible to determine the topological relation between the geometries of the POIs solely based on the topological relation between their MBRs, their topological relation will be previously indexed. However, this approach primarily focuses on efficiently determining the qualitative relation between two existing geospatial objects within a dataset, rather than identifying the subset of objects satisfying a specific qualitative relation among numerous objects.

The concept of a spatial pattern defined by solely qualitative connectivity constraints is used in [Rafael, 2021]. The work presents the Qualitative Spatial Pattern Search (QSPM) and an algorithm called the Topo-MSJ algorithm for addressing this type of search. However, the author does not explore the potential of combining quantitative restrictions with qualitative ones in this search context. To the best of our knowledge, our introductory work [Minervino *et al.*, 2023] in this field is the first work to formally generalize the SPM query proposed in [Fang *et al.*, 2018a] by combining quantitative and qualitative constraints.

### 3 Background

In this Section we give a brief review of the core concepts used in the QQESPM algorithm, including the geo-textual indexing and the topological relation model.

#### 3.1 Inverted Linear Quadtree Indexing

The Inverted Linear Quadtree (IL-Quadtree), a geo-textual indexing structure introduced in [Zhang *et al.*, 2016], serves as a fundamental component in the QQESPM algorithm. This index functions as a map, associating each unique spatial keyword in the dataset with its respective quadtree index structure. Each of these quadtrees contains a set of spatial objects (e.g., POIs) related to the specific keyword under consideration.

The bidimensional quadtree, as proposed in [Finkel and Bentley, 1974], divides a two-dimensional spatial domain into four quadrants recursively. Each quadrant can be further subdivided into four subquadrants, and this subdivision is represented by a tree structure. Each rectangular subspace represents a node in the tree, and a node's children correspond to its subquadrants. Subdivision occurs when the number of spatial entities (POIs) in a node exceeds a specified threshold, which can be adjusted during quadtree construction. Subspace division employs directional codes (00, 01, 10, and 11) to signify southwest, southeast, northwest, and northeast quadrants, respectively. Concatenating these codes recursively provides a unique identifier for each node, indicating its position in the quadtree hierarchy. Figure 3 illustrates the spatial partitioning in the quadtree and its associated tree structure. The IL-Quadtree's architecture efficiently retrieves geo-textual objects during geo-textual queries, as indicated in [Zhang *et al.*, 2016] and [Chen *et al.*, 2019]. The QQESPM algorithm relies on the IL-Quadtree indexing method to perform queries.

#### 3.2 Topological Relations

Topological relations are generally binary relations between two geometric objects. These relations need to be formally defined in order to accurately represent the connectivity between geometric entities. Two of the main formal methods of topological relation representation are the Region Connection Calculus (RCC) [Randell *et al.*, 1992; Cohn *et al.*, 1997] and the Dimensionally Extended Nine-Intersection Model

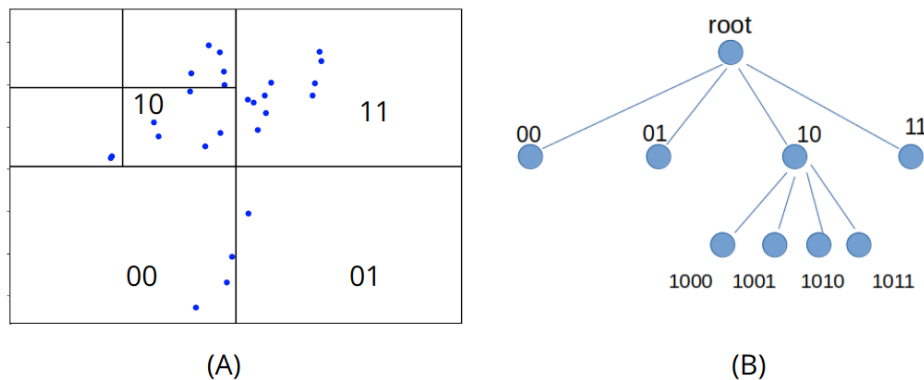


Figure 3. Example of a quadtree space subdivision (A), and its associated tree structure (B)

Table 1. Topological Predicates

Topological Predicate	Meaning
Equals	The Geometries share the same points
Disjoint	The Geometries have no point in common
Intersects	The Geometries have at least one point in common (the inverse of Disjoint)
Touches	The Geometries have at least one boundary point in common, but no interior points
Partially Overlaps	The Geometries share some but not all points in common, and the intersection has the same dimension as the Geometries themselves
Covers	Geometry A covers all the points in Geometry B
Covered By	All points of Geometry A are in Geometry B (the inverse of Covers)

(DE-9IM) [Egenhofer and Herring, 1990; Clementini *et al.*, 1993, 1994]. These representation models provide a structured framework for formally defining spatial predicates that describe the connectivity between spatial objects. Using such a formalization, it is possible to define relations such as “equals”, “touches” and “contains”. A simple description of some topological relations can be found in Table 1. The relation “covers” is a variation of “contains”, allowing the geometries to have intersecting boundaries [Clementini *et al.*, 1994].

## 4 Problem Formalization

Within this section, we give a formal definition for the fundamental terms in the QQ-SPM search problem.

**Definition 1 (spatial pattern)** A Spatial Pattern is a graph  $G(V, E)$  with a set of  $n$  vertices  $V = v_1, \dots, v_n$  and a set of  $m$  edges  $E$ , satisfying the following constraints:

1. each vertex  $v_i \in V$  has an associated spatial keyword  $w_i$
2. each edge  $e(v_i, v_j) \in E$  is labelled with at least one of the following types of description:
  - a connectivity spatial predicate  $\mathcal{R}_{ij}$ , among the following: {“equals”, “touches”, “covers”, “covered by”, “partially overlaps”, “disjoint”}
  - a distance interval  $[l_{ij}, u_{ij}]$ , and a sign  $\tau \in \{“\leftarrow”, “\rightarrow”, “\leftrightarrow”, “-”\}$

Each possible spatial pattern graph specifies a QQ-SPM query. For example, in the context of a POI group search, the vertices specify the POIs desired keywords, the connectivity predicates indicate the desired connectivity relations between the searched POIs. The distances between the POIs are restricted by the lower ( $l_{ij}$ ) and upper ( $u_{ij}$ ) bounds associated with the edge. This query representation can be generalized to any other geo-textual objects search scenarios. The meanings of the possible signs for an edge are described below:

- $v_i \rightarrow v_j$  [ $v_i$  excludes  $v_j$ ]: No geo-textual object with keyword  $w_j$  in the dataset should be found within a distance less than  $l_{ij}$  from the object corresponding to  $v_i$ .
- $v_i \leftarrow v_j$  [ $v_j$  excludes  $v_i$ ]: No geo-textual object with keyword  $w_i$  in the dataset should be found within a distance less than  $l_{ij}$  from the object corresponding to  $v_j$ .

- $v_i \leftrightarrow v_j$  [mutual exclusion]: The two-way restriction, i.e.,  $v_i$  excludes  $v_j$  and  $v_j$  excludes  $v_i$ .
- $v_i v_j$  [mutual inclusion]: The occurrence of geo-textual objects with keywords  $w_i$  and  $w_j$  in the dataset with distance shorter than  $l_{ij}$  from the objects corresponding to  $v_i, v_j$  is allowed.

Edges with the distance interval information are called quantitative edges, and edges with the connectivity predicate are called qualitative edges. Edges may or may not be simultaneously quantitative and qualitative. If a quantitative edge has the mutual inclusion sign, it is called an inclusive edge, otherwise, it is called an exclusive edge. Also note that, since the relation “covered by” is the inverse of “covers”, it could be discarded, but once edges are directional, i.e., have specific starting and ending vertices, we keep the relation “covered by”.

Notice that the attributes of an edge for the QQ-SPM query is a generalization of the attributes of an edge for the SPM query allowing qualitative connectivity constraints. In this way, the spatial pattern definition for the QQ-SPM query is also a generalization of the spatial pattern definition for the SPM query.

**Definition 2 (qq-e-match)** A pair of geo-textual objects  $(p_i, p_j)$  from a dataset  $D$  is called a qq-e-match for the edge  $e(v_i, v_j)$  if they respectively have the keywords  $w_i, w_j$  from the vertices  $v_i, v_j$ , and satisfy the distance and connectivity constraints of the edge  $e$ .

**Definition 3 (match)** A tuple of  $n$  geo-textual objects  $S = (p_1, p_2, \dots, p_n)$  from a dataset  $D$  is called a match for a spatial pattern  $G(V, E)$  when  $|V| = n$  and for each  $1 \leq i \leq n$ ,  $p_i$  has the keyword  $w_i$  from the vertex  $v_i$ , and for each edge  $e(v_i, v_j)$  of  $G$ , the pair of geo-textual objects  $(p_i, p_j)$  is a qq-e-match for the edge  $e$ .

Note that the order of POIs in the tuple corresponds to the order of vertices in the pattern  $G$ , so the  $i$ th POI  $p_i$  in the tuple corresponds to the  $i$ th vertex ( $v_i$ ) in the pattern  $G$ .

**Problem 1 (QQ-SPM search problem)** The QQ-SPM search problem or QQ-SPM query consists of finding all the matches of a spatial pattern  $G(V, E)$  in a dataset  $D$  of geo-textual objects, i.e., finding all combinations of objects from  $D$  that match the requirements represented in the given spatial pattern graph.

In order to find the qq-e-matches efficiently, our QQESPM algorithm uses the qq-n-match concept, which is formally defined below.

**Definition 4 (qq-n-match)** *Given a dataset  $D$  of geo-textual objects indexed in an IL-Quadtree  $ILQ$ , and a spatial pattern  $G(V, E)$ , let  $e(v_i, v_j)$  be an edge of the spatial pattern  $G$ , let  $ILQ_i$  and  $ILQ_j$  be the quadtrees for the keywords  $w_i$  and  $w_j$  of the vertices  $v_i$  and  $v_j$ , respectively, in the IL-Quadtree  $ILQ$ , and let  $n_i, n_j$  be two nodes from  $ILQ_i$  and  $ILQ_j$ , respectively, and  $b_i, b_j$  the MBRs for these nodes. We say that the node pair  $(n_i, n_j)$  is a qq-n-match for the edge  $e(v_i, v_j)$  if  $d_{min}(b_i, b_j) \leq u_{ij}$  and  $d_{max}(b_i, b_j) \geq l_{ij}$ , where  $d_{min}$  and  $d_{max}$  represent the minimum and maximum distance between the MBRs, and additionally:*

1. Case  $v_i \rightarrow v_j$ : there is no object  $x_j$  in  $ILQ_j$  such that  $d_{max}(b_i, x_j) < l_{ij}$
2. Case  $v_i \leftarrow v_j$ : there is no object  $x_i$  in  $ILQ_i$  such that  $d_{max}(x_i, b_j) < l_{ij}$
3. Case  $v_i \leftrightarrow v_j$ : (1) and (2) hold
4. Case  $e$  is qualitative with  $\mathbb{R}_{ij} \neq \text{"disjoint"}: b_i \cap b_j \neq \emptyset$

Intuitively, a pair of nodes  $n_i, n_j$  is a qq-n-match for the edge  $e$  if they consist of a pair of bounding boxes whose interior potentially contains solutions for the edge  $e$ . More specifically, if by checking the minimum and maximum distance between their MBRs  $b_i, b_j$ , it is not possible to eliminate the possibility of existing geo-textual objects  $p_i, p_j$  inside these nodes, such that  $(p_i, p_j)$  is a qq-e-match for the edge  $e$ , so the children nodes or leaves of  $n_i, n_j$  need to be further examined, and are candidates for finding qq-e-matches for the edge  $e$  in context.

Next, we introduce two lemmas on which the QQESPM algorithm is based.

**Lemma 1** *Let  $e(v_i, v_j)$  be an edge from a spatial pattern graph  $G(V, E)$ , let  $w_i, w_j$  be the keywords of the vertices  $v_i, v_j$ , and let  $ILQ_i, ILQ_j$  be the quadtrees for the keywords  $w_i, w_j$  respectively. Consider a pair  $o_i, o_j$  of geo-textual objects in the dataset  $D$ . If  $(o_i, o_j)$  is a qq-e-match for the edge  $e$ , then for any nodes  $n_i, n_j$  from the quadtrees  $ILQ_i, ILQ_j$ , respectively, if  $o_i$  is inside  $n_i$  and  $o_j$  is inside  $n_j$ , then the node pair  $(n_i, n_j)$  is a qq-n-match of  $e$ .*

**Proof.** Since  $o_i, o_j$  is a qq-e-match, then this pair satisfies the constraints of the edge  $e$ , in particular, we have  $d_{max}(b_i, b_j) \geq d(o_i, o_j) \geq l_{ij}$ , and  $d_{min}(b_i, b_j) \leq d(o_i, o_j) \leq u_{ij}$ . Also, if  $e$  is qualitative with  $\mathbb{R}_{ij} \neq \text{"disjoint"}$ , then  $o_i, o_j$  are intersecting, and  $n_i, n_j$  will be intersecting too. Now let us analyse the possible signs of the edge  $e$ . In case  $v_i \rightarrow v_j$ , suppose there is an object  $x_j$  in  $ILQ_j$  such that  $d_{max}(b_i, x_j) < l_{ij}$ , but since  $o_i$  is inside the bounding box  $b_i$  of the node  $n_i$ , then  $l_{ij} > d_{max}(b_i, x_j) \geq d(o_i, x_j)$  which is contradictory to  $(o_i, o_j)$  being a qq-e-match. So such  $x_j$  does not exist. The prove for the other possible signs are analogous, and thus, all the conditions for  $n_i, n_j$  to be a qq-n-match are satisfied.  $\square$

By using Lemma 1, in a QQ-SPM search procedure, we only need to find the qq-n-matches for the edges in order to

find the matching objects, since a pair of objects  $o_i, o_j$  outside any qq-n-match of an edge  $e$  will never be a qq-e-match for  $e$ . In other words, there will not exist any solutions for an edge  $e$  outside its qq-n-matches, thus, by only looking inside these qq-n-matches, an algorithm finds all the matches of a spatial pattern, without having to analyse the entire dataset. The following lemma brings the final piece to design an efficient search of qq-n-matches for the QQESPM algorithm.

**Lemma 2** *Suppose the node pair  $(n_i, n_j)$  is a qq-n-match of the edge  $e(v_i, v_j)$ . Let  $n_i^f$  and  $n_j^f$  be the father nodes of  $n_i$  and  $n_j$  respectively. Then, the node pair  $(n_i^f, n_j^f)$  is also a qq-n-match of  $(v_i, v_j)$ .*

**Proof.** The proof for the quantitative restrictions of the edges is provided in [Chen et al., 2019]. Regarding the additional proposed criterion to qualify as a qq-n-match, which is related to the connectivity constraint of the edge, it's important to note that if the node pair  $(n_i, n_j)$  constitutes a qq-n-match for an edge with a qualitative constraint other than "disjoint", they will possess intersecting bounding boxes. Since their father nodes encompass them, they too will be intersecting, thus ensuring that the condition for a node match persists for the father nodes.  $\square$

By using Lemma 2, in a QQ-SPM search procedure, to find the next level's qq-n-matches, we do not need to compare the bounding boxes of each node in that current depth level of the quadtrees, but we only need to test the children node pairs of the previous level's qq-n-matches, since for a node pair in the current level to be a qq-n-match it necessarily needs to have its parents node pair as a qq-n-match in the previous level. By using Lemmas 1 and 2, we can efficiently find the qq-n-matches for edges level by level, and at last, verify the inner objects of the qq-n-matches in the last level to find the matching objects. This process is described in the next Section.

## 5 QQESPM algorithm

This section presents the QQESPM algorithm, designed to handle QQ-SPM queries. QQESPM considers six possible topological relations between geo-textual objects, namely "equals", "touches", "covers", "covered by", "partially overlaps", and "disjoint". The overview of the search procedure is shown in Algorithm 1 (QQESPM), which delineates the high-level sequential steps for query execution, with an emphasis on achieving efficient execution by using the qq-n-matches concept. QQESPM takes as input a spatial pattern modelled as a graph  $G(V, E)$ , and a dataset of geo-textual objects indexed in an IL-Quadtree  $ILQ$  containing each quadtree  $ILQ_i$  for each keyword  $w_i$  in the dataset, and returns all the matches of the given spatial graph  $G$  in the dataset.

The maximum depth level of the quadtrees at which QQESPM will operate is chosen as the depth of the deepest quadtree among the quadtrees of the keywords in the search pattern. QQESPM starts by reordering the list of edges in order to first process the exclusive edges and then the inclusive edges, since the first one, as being more restrictive, tend



**Algorithm 1:** QQESPM

---

**Input:** IL-Quadtree  $ILQ$ , spatial pattern  $G$   
**Output:**  $\psi$ : all the matches of  $P$

```

1  $L = \max(\text{depth}(ILQ_i), 1 \leq i \leq n)$ 
2 for  $level = 1$  to  $L$  do
3   | derive the order of computing qq-n-matches for this
   | level
4   for each edge  $e$  do
5   | | compute the qq-n-matches for  $e$  in the current level
6   |
7   | derive the order of computing qq-e-matches
7   | identify skip-edges
8   for each non-skip edge  $e$  do
9   | | compute the qq-e-matches for  $e$ 
10  |
10  | derive the order of joining qq-e-matches
11   $\psi \leftarrow \text{join\_qq\_e\_matches}()$ 
12 return  $\psi$ 

```

---

to accumulate less candidate solutions during the algorithm execution steps. Then, the first qq-n-match for each edge is set as the pair of roots of the quadtrees of the keywords from the extreme vertices of each edge, since the entire space is the only candidate bounding region for finding solution in the first depth level of the quadtrees.

The QQESPM algorithm then iterates over the levels of the quadtrees, and at each, stores temporary sets of candidate nodes that potentially contain solutions for each vertex in the search pattern. Inside an iteration over a specific level  $l$ , the algorithm finds the qq-n-matches of the next level for each edge, by analysing the children of the qq-n-matches of the same edge in the previous level (Following Lemma 2), but also restricted to qq-n-matches whose nodes are among the candidate nodes for the vertices. For example, taking edges  $e_{12}$  and  $e_{13}$ , as they have vertex  $v_1$  in common, then at each level, the node for keyword  $w_1$  in  $e_{13}$  should be among the nodes for  $w_1$  in the qq-n-matches of  $e_{12}$ , so the first processed edges always restrict the candidate solutions for the next edges to be processes, and this gives intuition about why it is useful to process the more restrictive edges first. Guided by the same rationale, after ending the computation of qq-n-matches for each edge in a specific level of the quadtrees, QQESPM reorder the edges list for the next level, putting edges with fewer qq-n-matches in the previous level first.

Upon reaching the final depth level, the algorithm selected a set of edges, called skip edges, that do not need to have their qq-e-matches computed. This happens to inclusive edges that share the same extreme vertices with other edges that will already have the computation of qq-e-matches, and thus, to find the solutions for these edges it is sufficient to only join the candidate solutions of their adjacent edges, and keep only the ones satisfying the constraints of the skip edge. Proceeding, QQESPM computes the qq-e-matches for each non-skip edge by testing the distance and connectivity between the objects inside the qq-n-matches in the last level. At this step, for qualitative relations other than “disjoint”, it is useful to only test if the bounding box of the objects are intersecting, since this computation is much lighter than computing the exact connectivity relation, and since many candidate qq-e-match will still be discarded as they will not match with many other qq-e-matches of adjacent edges in the joining phase,

this strategy avoids unnecessary computation of connectivity relations for objects that will be discarded later on in the joining phase. QQESPM also keeps temporary sets of candidate objects for each vertex, since the computation of qq-e-matches for each edge restricts the possible qq-e-matches for its adjacent edges as they must share an equal object for their common ending vertex, and this discards early some objects serving as a pre-joining mechanism.

During the computation of qq-n-matches and also the qq-e-matches, the algorithm repeatedly needs to test if a pair of nodes (or objects)  $A$  and  $B$  are qq-n-match (or qq-e-match) for a given exclusive edge, which involves checking if there is an object sufficiently close to  $A$  with the same keyword of  $B$ . In case there is, this disqualify the pair  $A, B$ , and more generally, disqualify  $A, X$  for any node (or object)  $X$  as a qq-n-match (or qq-e-match) for that edge. So for these cases, QQESPM reuses the computation of the previous search and keeps in temporary memory the radius search for the existence of an object close to  $A$ , avoiding redundant tests during its execution.

After having the candidate qq-e-matches for each non-skip edge, the QQESPM algorithm then joins the qq-e-matches for adjacent edges, eliminating non-matching ones, as edges sharing a common vertex must have as solution the same geo-textual object for that vertex. After joining the solutions for all non-skip edge, the algorithm tests the obtained partial solutions and eliminate those not satisfying the skip edges constraints, and then verifies the actual connectivity relation between the objects of the final candidate solutions, and get the final solutions.

The structural framework of ESPM occurs in the QQESPM algorithm. Although the qq-n-match and qq-e-match criteria is different from the n-match and e-match criteria of ESPM, thus QQESPM promote distinct computations at each level, as qq-n-matches are computed level by level from the root to the maximum depth level of the keyword’s quadtrees. More specifically, as QQESPM treats connectivity from the beginning through the qq-n-matches concept, it is able to eliminate unpromising regions or objects much earlier with respect to qualitative requirements, and efficiently reduce the cost of the joining phase.

## 6 Experiments and Results

In this Section, we evaluate the performance of the proposed algorithm QQESPM in terms of execution time by comparing with a trivial algorithm for solving the QQ-SPM query that we call QQ-simple.

### 6.1 Experiments Description

The ESPM algorithm was implemented in Python, following the description provided in [Chen *et al.*, 2019]. This implementation was further adapted to include the qq-n-matches and qq-e-matches verification stages to accommodate qualitative connectivity constraints.<sup>3</sup> Additionally, a more straightforward approach, referred to as QQ-simple,

<sup>3</sup>The code for this implementation can be found in <https://github.com/pyqqespm/qqespm>

was also implemented, serving as baseline for QQESPM. This approach works as follows. The QQ-simple algorithm initially receives a spatial pattern graph and removes its qualitative requirements, sticking with a only-quantitative spatial pattern. This pattern is then forwarded to the ESPM only-quantitative algorithm, which then performs the search and return its results back to QQ-simple. These results are finally checked, to filter out the candidate solutions that do not meet the qualitative requirements. A comparative performance analysis was conducted, in order to compare the efficiency of these two QQ-SPM resolution algorithms, in terms of execution time and memory usage.

Matching based on spatial pattern graphs poses a computationally intractable problem, as outlined in [Fang et al., 2018a]. However, exact algorithmic solutions are pertinent given the typically small size of search patterns. Due to the computational complexity involved, our experimental evaluation primarily centers on execution time analysis, with supplementary assessment of memory usage in a simplified manner.

We have produced an implementation of the QQESPM and QQ-simple algorithms with parallelization for some independent steps. Specifically, we parallelized the finding of all qq-n-matches children of a given qq-n-match, for each level of depth, and also in the step of generating the candidate qe-matches children of a given qq-n-match in the final level.

Experiments were executed on a machine equipped with Intel Core i7-12700F CPU 4.90 GHz, coupled with 32GB of memory, operating on the Ubuntu OS.

We used a dataset comprising 40,410 POI geometries (13,073 polygons and 27,337 points) extracted from OpenStreetMap<sup>4</sup> filtered by the following bounding box: {min\_lat: -9.254, min\_long: -41.418, max\_lat: -4.544, max\_long: -32.794}, thereby predominantly spanning the Paraíba state, Brazil. The dataset comprises the tags “amenity”, “shop”, and “tourism”, which represent three of the most important POIs tags in the OSM dataset. Our extraction contains 40,604 POIs keywords among 343 distinct keywords.

In an effort to construct resource-intensive search spatial patterns, mirroring real-world conditions, we have selected only the most frequent keywords to compose the search spatial patterns with the following methodology: the candidate keywords for composing the spatial patterns were always chosen among the keywords with at least 100 occurrences in the dataset, which led to a set of 66 most frequent keywords.

We have used 12 distinct graph architectures for creating the spatial patterns, varying from graphs with only 2 vertices and 1 edge up to 6 vertices and 7 edges. These graph architectures, which are depicted in Figure 4, are the same used in the experiments of [Chen et al., 2019]. For each architecture, 10 distinct spatial patterns were generated with randomly selected keywords, totalizing 120 spatial patterns, of which 60 were generated using a probability of 1/2 for an edge to have qualitative requirements, and 60 using a probability of 1/3. As the dataset does not have too many intersecting POI geometries, patterns created with a smaller qualitative probability (i.e. more quantitative patterns) tend to have more solu-

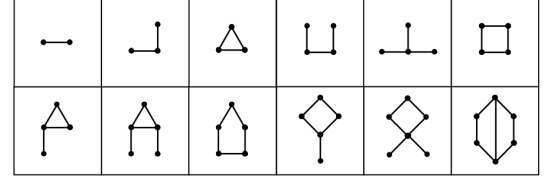


Figure 4. Structure of Search Spatial Patterns [Chen et al., 2019]

tions, whereas patterns created with greater qualitative probability tend to have less.

The dataset of POIs was randomly shuffled and divided into segments representing 20%, 40%, 60%, 80%, and 100% of its total size. For each of these dataset subsets, searches were conducted 3 times for each of the 120 generated spatial patterns, for both algorithms QQESPM and QQ-simple. Thus, the total number of executions was  $120 \cdot 5 \cdot 3 \cdot 2 = 3,600$ , and each of the two algorithms answered 1,800 queries.

For this implementation we have used the geodesic distance in meters as the default distance for satisfying the requirements in the search patterns. This distance metric is much more realist when compared to the euclidian distance between the latitude and longitude coordinates (used in [Minervino et al., 2023]), which is not in meters, but in degrees, making it harder for users to interpret when using a real world search system.

To construct the search patterns, the parameter  $l_{ij}$ , representing the minimum inter-POI distance, was randomly chosen between 0 and 1,000 (in meters), while the parameter  $u_{ij}$ , representing the maximum inter-POI distance, was randomized within  $l_{ij} + 1$  and  $l_{ij} + 10,001$  (in meters). We used a uniformly distributed random selection. The connectivity relations were introduced in the edges randomly from the set {“equals”, “touches”, “covers”, “covered by”, “partially overlaps”, “disjoint”} following a qualitative probability between 1/2 and 1/3 that determines the chance of an edge of the graph to have a qualitative requirement.

## 6.2 Results

We now present the performance analysis of the algorithm’s executions.

### Overall Execution Time Assessment

Figure 5 shows a boxplot for the execution times of the QQESPM and QQ-simple algorithms. The whisker parameter determines the minimum and maximum value marks considered non-outliers, and was set to 10, indicating that the min-

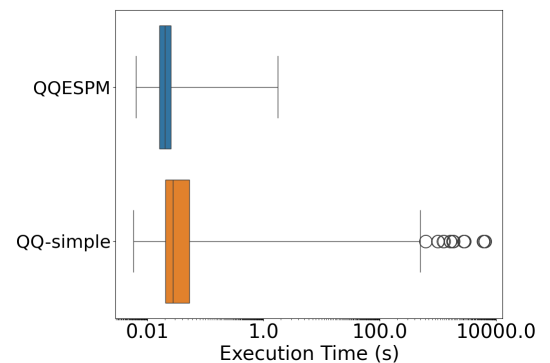


Figure 5. Boxplot of execution times per algorithm

<sup>4</sup><https://www.openstreetmap.org/>

**Table 2.** Execution Time Statistics for QQESPM and QQ-simple

Algorithm	#Executions	Avg. Exec. Time	Std	Median Exec. Time	Best Exec. Time	Worst Exec. Time
QQ-simple	1,800	43.93s	413.27s	0.03s	0.01s	6,528.77s
QQESPM	1,800	0.04s	0.10s	0.02s	0.01s	1.77s

imum and maximum marks in the boxplot will be expanded by up to 10 times the interquartile range units of distance in relation to the 25% and 75% percentiles. A generally accepted standard for the whisker in a boxplot is 1.5, but we chose a higher one so that we could generate a visualization that makes it easier to compare algorithm execution times. Note that even so, QQ-simple can achieve higher execution times considered outliers. Table 2 shows that for the dataset used, the QQ-simple algorithm had a worst execution time of 6,528.77s, which is 3,688 times slower than the worst execution time of QQESPM. Also note that the median execution times are around 50% higher for the QQ-simple algorithm, which also has a very high variance in execution times. For the average, the QQESPM algorithm obtained an execution time more than 1,000 times lower than the QQ-simple average.

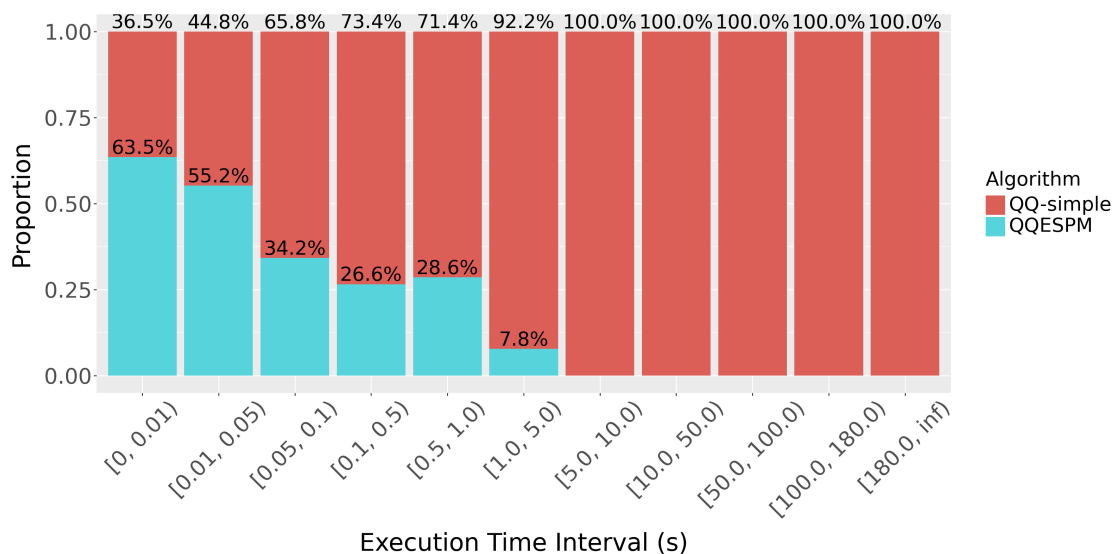
Figure 6 presents the proportion of executions of each algorithm within each execution time interval. Note that the QQESPM executions are entirely within the first six intervals (bars on the left), so that the execution times greater than 5s are all from the QQ-simple algorithm. Table 3 shows the cumulative values of the percentage of executions that fell in each execution time interval for both algorithms. Note that only the first two intervals, which include executions of less than 0.05s, contain 92.27% of the QQESPM executions but only 74.28% of the QQ-simple executions, which has its execution times divided much more widely, some executions being extremely costly, while the maximum execution time of QQESPM was 1.77s as shown in Table 2.

### Scalability Assessment

The average execution time by dataset size was measured for each algorithm. The results are shown in Figure 7 (A). Notice that the average execution time difference between QQESM and QQ-simple becomes larger as the dataset size increases. Clearly, QQESPM demonstrates significantly better scalability compared to QQ-simple. To better visualize the comparison, we applied a logarithmic scale to the y-axis. Shaded areas represent a 95% confidence interval for the average execution time.

### Number of Vertices Assessment

The average execution time by each number of vertices (number of keywords in the search pattern) was measured for each algorithm. The results, illustrated in Figure 7 (B), consistently demonstrate QQESPM's superior runtime performance over QQ-simple solution, regardless of the number of vertices. Figure 7 (B) shows that there is no direct correlation between the number of vertices and the execution time, as depending on the keywords and search distance and connectivity restrictions, the algorithms may or may not have an early stop during their executions, either due to the lack of qq-n-matches at some depth level of the search or due to the lack of qq-e-matches after scanning the last level. Furthermore, spatial patterns that have more solutions also tend to be more costly than patterns with no or few solutions (matches), as suggested by the graph in Figure 8. However, this analysis still allows us to verify that regardless of the number

**Figure 6.** Percentual of executions of each algorithm for each execution time interval**Table 3.** Percentage of Executions fallen within each Execution Time Range for QQESPM and QQ-simple

algorithm	[0, 0.01)	[0.01, 0.05)	[0.05, 0.1)	[0.1, 0.5)	[0.5, 1.0)	[1.0, 5.0)	[5.0, 10.0)	[10.0, 50.0)	[50.0, 100.0)	[100.0, 180.0)	180.0+
QQ-simple	1.28	73.00	6.83	9.67	1.11	2.61	1.17	1.33	0.50	0.33	2.17
QQESPM	2.22	90.05	3.56	3.50	0.44	0.22	0.00	0.00	0.00	0.00	0.00



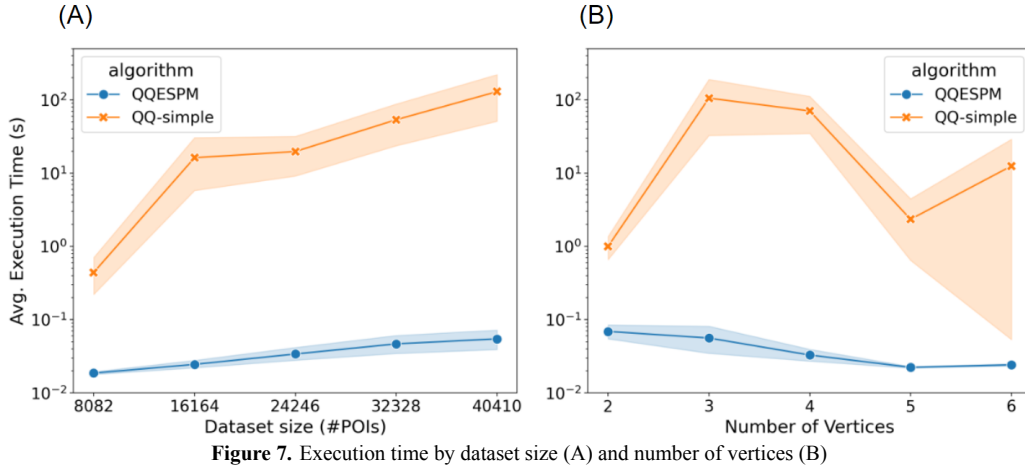


Figure 7. Execution time by dataset size (A) and number of vertices (B)

of vertices in the pattern of search, our proposed algorithm QQESPM is consistently more efficient than QQ-simple.

### Memory Usage Assessment

The average memory allocation by QQESPM queries was also consistently lower for all dataset sizes and number of vertices evaluated, compared to the QQ-simple executions. The overall average memory allocation during queries was 236.24MB for QQESPM executions and 317.56MB for QQ-simple executions, highlighting the memory efficiency advantage of QQESPM over the QQ-simple trivial approach. Figure 9 shows that this average difference also increases with the grow in the dataset size. It is important to note that in practice, part of the temporary runtime data can be virtualized in secondary memory, and these values were not measured here, however the numbers presented already provide an understanding of the efficiency of the algorithms.

### Statistical Test

We took the executions performed on the full dataset grouping by algorithm, which correspond to 360 executions for each algorithm, to perform a bootstrapping selecting 1,000 random samples for each algorithm, where each sample is the selection of 10% of the measured executions times of one of the algorithms. Then, we performed a Z hypothesis test comparing the average execution time between the QQESPM samples and the QQ-simple samples. The obtained p-value  $8.536 \cdot 10^{-179}$  clearly confirms a statistically signifi-

cant difference in execution time between the algorithms, where the proposed QQESPM has the advantage over the trivial algorithm QQ-simple.

### Generalizing the solution

Although we propose the QQ-SPM search as an example of only POI search scenarios, it can actually be generalized to any search context for geo-textual objects that have some geometry that delimits the scopes or boundaries of the objects. For example, you could use the QQ-SPM query to search entire cities, web documents, or any other type of data that has keywords and delimiting polygons. Furthermore, we implemented the geodesic distance to satisfy the distance requirements of the search patterns. However, the proposed solution is independent of implementation, and therefore, the distance metric between objects can be modified to any other, such as the distance of network, which considers the minimum path along existing lines (e.g.: through streets and avenues). The indexing strategy implemented in the QQESPM algorithm was IL-quadtrees. However, other geo-textual indexing strategies can also be used at this point.

## 7 Conclusion

The primary aim of this study is to introduce and formally define the QQ-SPM search as a geo-textual search problem centered on keywords, distance, and qualitative connectivity requirements among geo-textual objects within a dataset.

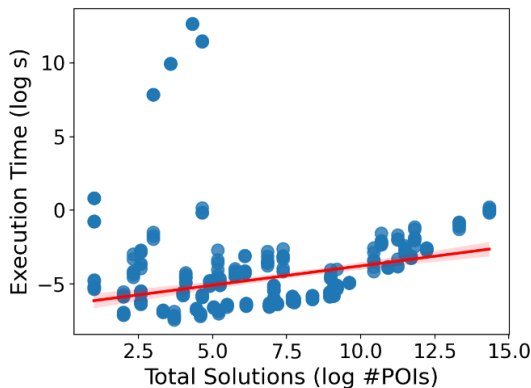


Figure 8. Execution time by total solutions (log)

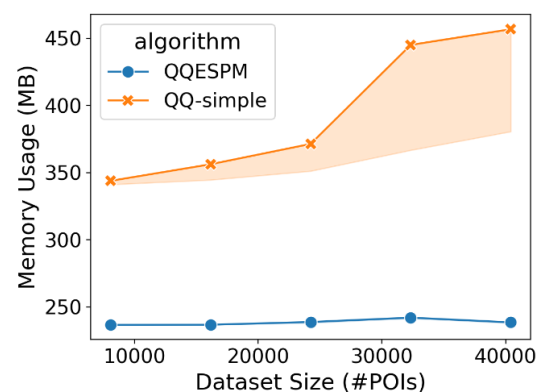


Figure 9. Memory Usage by dataset size

Additionally, we present an efficient algorithmic solution, the QQESPM algorithm, derived from the ESPM algorithm which was only based on quantitative distance constraints.

Our investigation involves a rigorous formalization. By leveraging mathematically proven lemmas the QQESPM algorithm optimizes early filtering and search termination conditions. Through performance analysis, we demonstrate the superior efficiency of this algorithm compared to simplistic solutions for the QQ-SPM search problem. Notably, QQESPM exhibits favorable scalability, with marginal increases in execution time as dataset size grows. The proposed algorithm holds practical utility across various scenarios, such as Points of Interest (POI) search or geolocated web document retrieval.

Integration of QQESPM into backend APIs for POI search systems or GUI web/mobile applications can offer enhanced search capabilities. Furthermore, extensions for spatial databases could incorporate QQESPM's filtering strategy, broadening its applicability. Importantly, our theoretical formalization and the algorithm are adaptable beyond POI searches, potentially serving diverse applications with minimal conceptual and implementation modifications. Future directions include the implementation of the above examples, the refinement of parallelization strategies, the experimentation of alternative indexing structures, and the flexibility expansion for the qualitative requirements for spatial patterns represented on QQ-SPM searches.

## References

- Cao, X., Cong, G., Jensen, C. S., and Ooi, B. C. (2011). Collective spatial keyword querying. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 373–384.
- Chen, H., Fang, Y., Zhang, Y., Zhang, W., and Wang, L. (2019). Espm: Efficient spatial pattern matching. *IEEE Transactions on Knowledge and Data Engineering*, 32(6):1227–1233.
- Chen, Y., Feng, K., Cong, G., and Kiah, H. M. (2022). Example-based spatial pattern matching. *Proceedings of the VLDB Endowment*, 15(11):2572–2584.
- Chen, Z., Chen, L., Cong, G., and Jensen, C. S. (2021). Location-and-keyword-based querying of geo-textual data: a survey. *The VLDB Journal*, 30:603–640.
- Choi, D.-W., Pei, J., and Lin, X. (2016). Finding the minimum spatial keyword cover. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 685–696. IEEE.
- Choi, D.-W., Pei, J., and Lin, X. (2020). On spatial keyword covering. *Knowledge and Information Systems*, 62(7):2577–2612.
- Clementini, E., Di Felice, P., and Van Oosterom, P. (1993). A small set of formal topological relationships suitable for end-user interaction. In *International symposium on spatial databases*, pages 277–295. Springer.
- Clementini, E., Sharma, J., and Egenhofer, M. J. (1994). Modelling topological spatial relations: Strategies for query processing. *Computers & graphics*, 18(6):815–822.
- Cohn, A. G., Bennett, B., Gooday, J., and Gotts, N. M. (1997). Qualitative spatial representation and reasoning with the region connection calculus. *geoinformatica*, 1:275–316.
- Egenhofer, M. J. and Herring, J. (1990). Categorizing binary topological relations between regions, lines, and points in geographic databases. *The*, 9(94-1):76.
- Fang, Y., Cheng, R., Cong, G., Mamoulis, N., and Li, Y. (2018a). On spatial pattern matching. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 293–304. IEEE.
- Fang, Y., Cheng, R., Wang, J., Budiman, L., Cong, G., and Mamoulis, N. (2018b). Spacekey: Exploring patterns in spatial databases. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 1577–1580. DOI: 10.1109/ICDE.2018.00180.
- Fang, Y., Li, Y., Cheng, R., Mamoulis, N., and Cong, G. (2019). Evaluating pattern matching queries for spatial databases. *The VLDB Journal*, 28:649–673.
- Finkel, R. A. and Bentley, J. L. (1974). Quad trees a data structure for retrieval on composite keys. *Acta informatica*, 4:1–9.
- Guo, T., Cao, X., and Cong, G. (2015). Efficient algorithms for answering the m-closest keywords query. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 405–418.
- Hermoso, R., Trillo-Lado, R., and Ilarri, S. (2019). Rescoskq: Towards pois recommendation using collective spatial keyword queries. In *CEUR workshop proc.*, number ART-2019-114041.
- Li, Y., Fang, Y., Cheng, R., and Zhang, W. (2019). Spatial pattern matching: A new direction for finding spatial objects. *SIGSPATIAL Special*, 11(1):3–12. DOI: 10.1145/3355491.3355493.
- Long, Z., Duckham, M., Li, S., and Schockaert, S. (2016). Indexing large geographic datasets with compact qualitative representation. *International Journal of Geographical Information Science*, 30(6):1072–1094.
- Minervino, C., Campelo, C., Oliveira, M., and Silva, S. (2023). Qqespm: A quantitative and qualitative spatial pattern matching algorithm. *arXiv preprint arXiv:2312.08992*.
- Rafael, G. J. R. (2021). Busca por grupos de pontos de interesse usando processamento qualitativo de regiões espaciais. Master's thesis, Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, Programa de Pós-Graduação em Ciência da Computação, Campina Grande, Paraíba, Brasil.
- Randell, D. A., Cui, Z., and Cohn, A. G. (1992). A spatial logic based on regions and connection. *KR*, 92:165–176.
- Zhang, C., Zhang, Y., Zhang, W., and Lin, X. (2016). Inverted linear quadtree: Efficient top k spatial keyword search. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1706–1721.
- Zhang, D., Tan, K.-L., and Tung, A. K. (2013). Scalable top-k spatial keyword search. In *Proceedings of the 16th international conference on extending database technology*, pages 359–370.