MCOPS-SPM: Multi-Constrained Optimized Path Selection Based Spatial Pattern Matching in Social Networks

Ying Guo[1(✉)], Lianzhen Zheng[1], Yuhan Zhang[1], and Guanfeng Liu[2]

[1] Qingdao University of Science and Technology, Qingdao 266061, Shandong, China
guoying@qust.edu.cn
[2] Macquarie University, Sydney, Australia

Abstract. In this paper, we study the multi-constrained optimized path selection based spatial pattern matching in Location-Based Social Network (MCOPS-SPM). Given a set D including spatial objects (each with a social identity and a social reputation) and social relationships (e.g., trust degree, social intimacy) between them. We aim at finding all connections (paths) of objects from D that match a user-specified multi-constraints spatial pattern P. A pattern P is a complex network where vertices represent spatial objects, and edges denote social relationships between them. The MCOPS-SPM query returns all the instances that satisfy P. Answering such queries is computationally intractable, and we propose algorithms to solve the multi-constrained optimized path matching problem and guide the join order of the paths in the query results. An extensive empirical study over real-world datasets has demonstrated the effectiveness and efficiency of our approach.

Keywords: Location-Based Social Network · Multiple constraints · Optimized path selection · Spatial Pattern Matching

1 Introduction

The emerging Location-Based Social Network (LBSN) attracts a large number of participants. Foursquare, Twitter and other applications make it easy for people to mark their locations. This has aroused widespread interests among researchers. In LBSN, Multi-Constrained Optimized Path Selection (MCOPS) has become a challenging problem. In this paper, we study a Spatial Pattern Matching (SPM) problem based on MCOPS, that is, in the LBSN, to find all the matches satisfying multi-constrained path pattern P in the space objects set D. The multi-constraints in P include keyword constraint, distance constraint, and social impact factors (trust, intimacy, social reputation) constraints. The related Spatial Keyword Query (SKQ) and Graph Pattern Matching (GPM) have been extensively studied.

In general, a spatial-keyword query (SKQ) [1–4] returns spatial objects that are spatially close to each other and match the keywords. For example, top-k keyword query [5], takes the spatial location and a set of keywords as parameters, returning k spatio-textual

X. Zhang et al. (Eds.): CloudComp 2019/SmartGift 2019, LNICST 322, pp. 3–19, 2020.
https://doi.org/10.1007/978-3-030-48513-9_1

objects that well meet the user's requirements, and the k objects are sorted according to the proximity of the query location and the relevance of the query keywords. Although SKQ plays an important role in geo-location-oriented applications (e.g., Google Maps), however, with the continuous emergence of new application scenarios, the requirements of users are increasingly diversified, which results in the returned query results not meeting the needs of users [6]. Moreover, various relationships between social network participants (e.g., location relationships, social relationships) are not well represented. Suppose that a user wishes to find a supplier, which is close to a bank and is far from a retailer and a whole-seller. SKQ will return objects that are spatially close to each other as shown in the dashed box in Fig. 1(a). However, users generally have restrictions on the spatial objects of the query (e.g., the supplier is at least 10 km away from the retailer in Fig. 1(b)). Therefore, the objects connected by the solid line in Fig. 1(a) are the answer to the query that the user really wants. Second, simply using spatial indexing structures (such as R-tree and their variants) requires dynamically creating an index for each vertex, which results in serious overhead [7]. Therefore, the solution to the SKQ problem cannot be directly applied to the MCOPS-SPM problem.

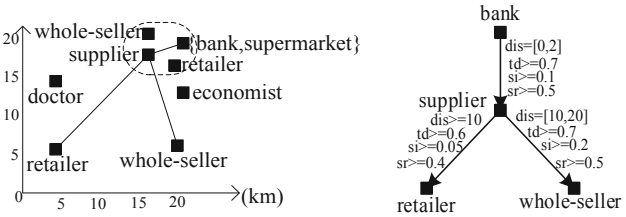

Fig. 1. MCOPS-SPM examples

In general, graph pattern matching (GPM) aims to find all matches of P in data graph G according to a given pattern graph P [8]. However, using GPM algorithms to solve MCOPS-SPM problem is not straightforward because (1) the solution to the GPM problem is primarily designed for the data structure of the graph, rather than spatio-textual objects indexed by an IR-tree index structure, (2) in the pattern graph, only a single constraint (such as hops or distances) among vertices is generally considered. Therefore, if GPM is used to solve the MCOPS-SPM problem, it is necessary to convert the spatial objects into graphs and extend the distance-based constraint to multi-constraints. The experiments in [7] prove that it is not efficient.

There are few studies on Spatial Pattern Matching (SPM). [9] proposed a database query language based on time mode, spatial mode and spatiotemporal mode. In [7], a SPM algorithm is proposed, but only a single constraint (different expression of distance constraint) is considered, which cannot meet the requirements of the user's multiple constraints.

We use an example in [10] to illustrate the importance of specifying multiple constraints for query keywords. In business activities, the pattern graph can be specified to look for keywords about suppliers, retailers, whole-sellers, banks in the social network.

The supplier directly or indirectly provides the retailer, whole-seller with products. Suppliers, retailers, whole-sellers obtain services directly or indirectly from the same bank. As shown in Fig. 1, the user specifies the supplier is at least 10 km from the retailer and the trust value is at least 0.6, the intimacy is at least 0.05, and the social reputation is at least 0.4 in the field. However, SKQ and GPM tend to ignore the distance constraint and social impact factor constraints of these problems, and the results of the query are often not of the most interest to the users.

In order to facilitate users to specify the spatial relationships between the keywords and the social relationships, we propose a multi-constrained optimized path selection based spatial pattern matching query (MCOPS-SPM). As shown in Fig. 1, given a set of spatial objects D (Fig. 1 (a)) and a spatial path pattern P (Fig. 1 (b)), MCOPS-SPM returns all matches about P in D. The pattern P is a complex network in which vertices represent spatial objects associated with keywords, edges represent spatial distance relationships between objects, and social relationships. For example, the distance between the user-specified bank and the supplier is [0, 2] (km), the trust value is at least 0.7, the intimacy is at least 0.1, and the social reputation is at least 0.5. In this example, the four objects connected by solid lines satisfy all constraints in P, which is a match of pattern P.

The main research contents of this paper are as follows:

1. We first proposed Multi-Constrained Optimized Path Matching algorithm (MCOPM). Different from the traditional path selection algorithms, MC-SPM based on MCOPS aims to find the matching of the spatial path pattern P in D. The vertices of the matching results satisfy the query keywords, and satisfy the spatial relationship and social relationship among the spatial objects. This will well support LBSN-based applications.
2. In general, participants and social relationships in social networks are relatively stable for a long period of time. We use IR-tree as an index structure, which is more effective in keyword search and distance pruning. It can effectively reduce search space and improve effectiveness.
3. Based on the IR-tree index structure, we propose an IR-tree based multi-constrained optimized path matching algorithm, namely IR-MCOPM. Matches of the multiple constrained paths of P in D can be found.
4. After mapping all the edges in P to corresponding paths in D, they need to be connected in a certain order to form a complete answer. Therefore, a sampling-based estimation algorithm is proposed to guide the connection order of the paths at the matching result in an efficient way, namely MCPJoin.

The rest of this paper is organized as follows. We first review the related work on SKQ and GPM in Sect. 2. Then we introduce the necessary concepts and formulate the focal problem of this paper in Sect. 3. Section 4 presents our solutions MCOPM, IR-MCOPM, MCPJoin algorithms. We report experimental results in Sect. 5 and conclude in Sect. 6.

2 Related Works

The most related works about spatial pattern matching are spatial keyword query and graph pattern matching. Below we analyze each of them in detail.

2.1 Spatial Keyword Query (SKQ)

In general, researches on SKQ issues can be roughly divided into two types. The first is m-Closest Keywords query [1, 11]. Given spatial data D, the mCK query returns a collection of objects that are spatially close to each other. The objects in the collection satisfy the m keywords specified by the user. [1] proposes an index structure of bR*-tree, and uses distance and keyword constraints to reduce the search space.

The second is the top-k SKQ. Top-k SKQ returns the objects with the highest sorting scores, determined by the sorting function, which takes into account both spatial proximity and text correlation [12]. In order to improve the efficiency of the query, some index structures are proposed, which can be divided into two types of structures depending on the method used. One is an R-Tree based index structure, such as: KR*-Tree [13], Hybrid Spatial-Keyword Indexing (SKI) [14], IR-tree [15], Spatial Inverted Index (S2I) [5], The other is a grid-based index structure, such as: Spatial-Keyword Inverted File (SKIF) [16]. In [17], the author used W-IBR-Tree (using keyword partitioning and inverted bitmaps to organize spatial objects) to solve the joint top-k SKQ problem. Recently, [18] proposed why the answer is why-not (SKQ), returning the result of the smallest modified query containing the expected answer. In [19], in order to improve the query efficiency in space, semantics and text dimensions, the NIQ-tree and LHQ-tree hybrid indexing methods are proposed, and the similarity in these three dimensions is used to prune the search space.

However, our proposed MCOPS-SPM can more clearly reflect users' needs. The MCOPS-SPM problem is also related to the multi-way spatial join [20]. In [21], the vertices in the data graph G are transformed into points in the vector space by the graph embedding method, which converts pattern matching queries into distance-based multi-paths-joining problems on vector space. However, in these solutions, the keyword attributes and multi-constraints between objects are not considered, so they cannot be directly applied to the MCOPS-SPM solution.

2.2 Graph Pattern Matching (GPM)

There are two types of GPM issues in the references. The first one is subgraph isomorphism [10, 22–24]. The subgraph isomorphism is an exact match between the data graph and the pattern graph. It aims to find the subgraph that exactly matches the pattern graph on the nodes and edges properties in the data graph [25]. [25] indicates that the subgraph isomorphism requires too strict conditions and poor scalability, and it is difficult to find a useful matching pattern in the data graph.

The second type of GPM problem is graph simulation. [26–28] pointed out that the graph simulation has more loose constraints, and it is convenient to search the data graph for the subgraph matching the specified pattern P. The returned subgraph can well meet the users' requirements, but it still needs to execute the edge to edge mapping, which is

still very harsh for applications that need to meet the specified path length. In order to solve the above problem, a bounded simulation is proposed in [8], in which each node has an attribute, each edge is associated with the path length (hops), and the value of each edge is One of len and *, len indicates that the path length cannot exceed len, and * indicates that there is no constraint on the path length. The bounded simulation maps the edges in pattern P to paths that meet the specified length in the data graph, further reducing the strict constraints, thereby better capturing users' intents.

There are some algorithms that connect the searched edges in a certain order [21]. In [21], an MDJ (Multi-Distance-Join) algorithm is proposed to guide the connection order of edges. In [10], the author proposed an R-join algorithm based on reachability conditions, using the B^+-tree structure as the index method.

However, the solutions above mentioned are based on a single constraint algorithm, and can not meet the requirements of multiple constraints specified by users, so these methods can not be directly applied to the MCOPS-SPM problem.

3 Problem Definition

3.1 Social Impact Factors

Social networks can be modeled as directed graphs, and we use $G = (V, E)$ to express social networks. Where $V = \{v_1, v_2, \ldots, v_n\}$ represents a collection of vertices, each vertex v_i represents a participant in a social network. $E = \{e_1, e_2, \ldots, e_m\}$ is a collection representing edges, each edge e_j represents an interaction or social relationships between two participants. However, the factors that influence social networks are always diverse, and we cannot draw precise conclusions based on a single condition. We propose four factors that influence social networks:

Social Identities. In a social network, each participant has his or her own identities, represented by keywords $k(v_i)$. Each participant may have multiple identities (for example, a digital blogger may be a student at a university), so the social identities help describe the social roles of participants.

Trust Degree. In a social network, trust degree refers to the level of trust that one participant forms in the interaction process with another participant. Let $td(v_i, v_j) \in [0, 1]$ indicate the trust degree that participant v_i evaluates v_j. If $td(v_i, v_j) = 1$, it represents that v_i completely trusts v_j while $td(v_i, v_j) = 0$ represents v_i wholly distrusts v_j.

Social Intimacy. In social networks, intimacy refers to the degree of intimacy formed between participants during the participation of social activities. It reflects the frequency of interaction between the two participants. $si(v_i, v_j) \in [0, 1]$ reflects the degree of intimacy between v_i and v_j. When $si(v_i, v_j) = 1$, it expresses that v_i and v_j are the most intimate. On the contrary, they have the least intimate social relationships.

Social Reputation. In social networks, social reputation refers to the extent to which a participant contributes in the field. It reflects the extent to which all participants rated this participant. Let $sr(v_i) \in [0, 1]$ denote social reputations in the domain. Here $sr(v_i) = 1$ represents that v_i has won the best reputations in the domain. Otherwise, there is no foothold for v_i in this field.

3.2 Social Impact Factor Aggregations

In order to meet the requirements of different conditions of users, the match of an edge in the pattern P may be a path in the data graph G. Then we need to aggregate the social impact factors along this path [29].

Trust Degree Aggregation. The aggregation of trust degree between source participants and target participants can be calculated according to the following formula [29]:

$$td(p) = \prod_{(v_i, v_j) \in p} td(v_i, v_j) \tag{1}$$

Where, p represents a complete path from the source participant to the target participant, (v_i, v_j) represents the path between v_i and v_j.

Social Intimacy Aggregation. The social intimacy aggregation between the source participant and the target participant is calculated according to the following formula [29]:

$$si(p) = \frac{\prod_{(v_i, v_j) \in p} si(v_i, v_j)}{\varepsilon^{\sigma}} \tag{2}$$

Where, ε denotes the length (hops) of the path from the source participant to the target participant, σ is called the attenuation factor that controls the decay rate.

Social Reputation Aggregation. The social reputation aggregation on the path p is calculated according to the following formula [29]:

$$sr(p) = \frac{\sum_{i=2}^{n-1} sr(v_i)}{n-2} \tag{3}$$

Here, in addition to the source and target participants, a weighted average of their social reputations is made to express the social reputation about the path p.

3.3 Multi-constrained Path Pattern

The multi-constrained path pattern P is a complex social network (for convenience of description, we will collectively refer to pattern P below). P = (V, E, KV, MCE) obeys the following constraints:

- V and E are the vertex and edge sets in P respectively;
- $KV(V_i)$ is a collection of keywords (social identities) that the user specifies on vertices v_i;
- For (u, v) $\in$ E, MCE(u, v) is the multi-constraints that users specified on edge (u, v). MCE(u, v) = {dis(u, v), td(u, v), si(u, v), sr(u, v)}, $\forall\varphi \in$ {td, si, sr}, $\varphi \in [0, 1]$. Where, dis(u, v) denotes their distance limits in real world. We assume the distance metric is Euclidean.

For example, in the pattern P of Fig. 1(b), the limit of distance between bank and supplier is at [0, 2] km, trust degree is no less than 0.7, social intimacy is not less than 0.1, social reputation is no less than 0.5.

3.4 Multi-constraints Path Match

In pattern P, for any two vertices v_i and v_j, for any two spatial objects o_i and o_j in spatial database D, when the following conditions are met:

- $k(o_i) \supseteq k(v_i)$, $k(o_j) \supseteq k(v_j)$;
- $td(o_i, o_j) \geq Q_{td}(v_i, v_j)$, $si(o_i, o_j) \geq Q_{si}(v_i, v_j)$, $sr(o_i, o_j) \geq Q_{sr}(v_i, v_j)$, where $Q_{td}(v_i, v_j)$, $Q_{si}(v_i, v_j)$, $Q_{sr}(v_i, v_j)$ represent trust degree, social intimacy, social reputation constraints on edge (v_i, v_j) respectively;
- The distance constraint between o_i and o_j holds one of the following conditions:

1) $dis(o_i, o_j) \geq distL(v_i, v_j)$
2) $0 \leq dis(o_i, o_j) \leq distU(v_i, v_j)$
3) $distL(v_i, v_j) \leq dis(o_i, o_j) \leq distU(v_i, v_j)$

Where, $distL(v_i, v_j)$ and $distU(v_i, v_j)$ are the lower distance limit and the upper distance limit on path (v_i, v_j) respectively;

We call o_i and o_j constitute a multi-constraints path match on path (v_i, v_j).

3.5 Match

Given a pattern P and a set O of objects, O is a match of P if: (1) for each path of P, there is a multi-constraints path match in O; (2) all objects in O are elements that match a certain edge in pattern P.

Problem 1 (Multi-Constrained Optimized Path Selection Based Spatial Pattern Matching, MCOPS-SPM). Given a pattern P, MCOPS-SPM returns all the matches of P in D.

In Fig. 1(a), for instance, the four objects connected in solid lines match the pattern in Fig. 1(b), and it is the answer of this MCOPS-SPM query.

Lemma 1 (Hardness). The MCOPS-SPM problem is NP-hard.

Proof. The MCOPS-SPM problem can be reduced to the spatial pattern matching problem. The SPM problem is NP-hard, which can be found in this paper [7].

4 Algorithm Design

4.1 Multi-Constrained Optimized Path Matching Algorithm (MCOPM)

To improve the quality of the solution, the MCOPM algorithm consists of two phases: reverse search (Reverse_Search_Dijkstra, RSD) and forward search (Forward_Search_Dijkstra, FSD). The RSD is executed in the direction from the target node to the source node to determine whether MCOPM exists (ie, from a path of pattern P to a mapping of D that satisfies multiple constraints). If there is a feasible solution, then the FSD is executed in the direction from the source node to the target node to determine whether there are better solutions. The detailed execution flow of the algorithm is as follows:

Step1: Perform RSD. From the target node to the source node, the Dijkstra algorithm is used to find the path with the smallest objective function value and record the aggregated values of each node from the target node to the intermediate node. The objective function [29] is defined as follows:

$$h(p) \triangleq \max\left(\frac{1-td(p)}{1-Q_{td}(p)}, \frac{1-si(p)}{1-Q_{si}(p)}, \frac{1-sr(p)}{1-Q_{sr}(p)}\right) \tag{4}$$

Where, td(p), si(p), sr(p) respectively represent the trust degree, social intimacy, and social reputation value of the aggregation on the path p. $Q_{td}(p)$, $Q_{si}(p)$, $Q_{sr}(p)$ respectively represent the corresponding constraints specified by users on the path p. It can be seen from Eq. (4) that only the aggregated values on the path all satisfy the constraints, we have $h(p) \leq 1$; if there is an aggregate value that does not satisfy the corresponding constraint, we have $h(p) > 1$.

Step2: If $h(p) \leq 1$, it means that there is a feasible path, that is, there is a match of multiple-constraints-edge, and go to Step 3 to continue execution; if $h(p) > 1$, it means that there is no feasible path, and the algorithm terminates.
Step3: Execute the FSD from the source node to the target node. According to the Dijkstra algorithm, we want to find the path that can maximize the path quality function [29]. The path quality function is defined as follows:

$$U(p) = \alpha \times td(p) + \beta \times si(p) + \gamma \times sr(p) \tag{5}$$

Here, α, β, γ are the weights of td(p), si(p), sr(p) respectively, $\forall\varphi \in \{\alpha, \beta, \gamma\}$, $0 < \varphi < 1$ and $\alpha + \beta + \gamma = 1$.

MCOPM is basically a two-round execution of the Dijkstra algorithm with particular metric functions. So it consumes twice the execution time of dijkstra. The time complexity of MCOPM is $O(|V|\log|V|+|E|)$, where |V| is the number of objects in D, |E| is the number of edges between any two objects.

Algorithm 1: MCOPM

Input: M, sn, tn, Q_{td}, Q_{si}, Q_{sr}
Output: $p_{s\to t}$
1 $P_{t\to s}=\emptyset$, $P_{s\to t}=\emptyset$
2 **Perform Reverse_Search_Dijkstra**(M, sn, tn, Q_{td}, Q_{si}, Q_{sr})
3 **if** $h(P_{t\to s})\leq 1$ **then**
4 **Perform Forward_Search_Dijkstra**(M, sn, tn, Q_{td}, Q_{si}, Q_{sr}, Agr_{td}, Agr_{si}, Agr_{sr})
5 **Return** $P_{s\to t}$
6 **else**
7 **Return** no feasible paths that satisfy constraints specified by users
8 **end**

4.2 Multi-Constrained Optimization Path Matching Algorithm Based on IR-Tree (IR-MCOPM)

In order to improve query efficiency and reduce search space in spatial data, we use IR-tree (Inverted File R-tree) [22] to organize spatial objects (social relationships in social networks are stable for a long period of time [30]), inverted files [31] are used for text relevance queries, and R-tree is used for spatial proximity queries. Specifically, each node in the R-tree is associated with a keywords list, and each leaf node is a data entity $<o_i, loc(x_i, y_i), (w_i, score_i)>$. Here, o_i represents an object identifier in a spatial database. $loc(x_i, y_i)$ represents the two-dimensional coordinates of a spatial object o_i. The $score_i$ of w_i can be calculated by the following expression [22]:

$$score = \lambda \times \frac{dis(q.loc, o.loc)}{maxdisD} + (1-\lambda) \times Krel(q.keywords, o.keywords) \quad (6)$$

Where, dis(q.loc, o.loc) denotes the Euclidean distance of query location q.loc and object location o.loc. maxdisD represents the maximum distance between any two objects in the spatial database D. Krel(q.keywords, o.keywords) indicates how similar the query keywords q.keywords is to the keywords of the objects o.keywords in D. $\lambda \in [0, 1]$ is used to weigh the importance of spatial proximity and keywords similarity. Each non-leaf node is an index entity $< id, mbr, IF >$. Here, id denotes the unique identifier of the node, mbr represents the minimum bounding rectangle (MBR) that covers all child nodes, each node points to an inverted file IF. The inverted file of a non-leaf node NF contains the following two parts: (1) list of all the different keywords contained in the sub-tree with NF as the root node, (2) a post list is associated with keywords, including a pointer to the child node and a text relevance in the sub-tree where the child node is the root node. The calculation of text relevance is given by the following formula [31]:

$$w_k = \frac{1 + \ln f_k}{\sqrt{\sum_k (1 + \ln f_k)^2}} \quad (7)$$

Where, f_k indicates the frequency of keyword k, w_k is the text relevance after normalization.

We need to create an IR-tree index for all nodes. The benefits of using IR-tree are as follows: (1) easy searching; (2) the spatial relationships between nodes can be quickly determined by their MBRs. Then we can quickly find the paths that satisfy the multi-constraints pattern according to Algorithm 2. The algorithm is shown in Algorithm 2, and the algorithm execution flow is as follows:

Step 1: Input the root of IR-tree irTree.root, the keywords of the source node and the target node of a edge(path) are respectively k_i and k_j, the trust degree, social intimacy, and social reputation user-specified are Q_{td}, Q_{si}, Q_{sr} respectively. The upper and lower limits of the distance are respectively distU, distL. Output is a set of paths that satisfy the multi-conditions Θ.

Step 2: Join the root node to the path set Ω, and add IR-tree root node to the key of Θ, and the value is the path set Ω.

Step 3: Look for objects x and y that respectively satisfy the keywords k_i and k_j in the IR-tree, and calculate the minimum distance and maximum distance between them based on the MBR of the objects x and y. If maxDist is less than distL, we skip this loop; Otherwise, and (x, y) is a matching pair, then we execute Algorithm 1 to find if there are paths satisfying multi-constraints between the objects x and y. If it exists, then we add it to the path set Ω. The matching objects and the corresponding path set are added into Θ. Finally we return Θ.

Algorithm 2: IR-MCOPM

Input: irTree.root, k_i, k_j, Q_{td}, Q_{si}, Q_{sr},distL, distU

Output: Θ

```
1  Ω.add(irTree.root), Θ←∅, Θ.add(irTree.root,Ω)
2  for x∈irTree.search(k_i) do
3        Ω←∅
4        for y∈irTree.search(k_j) do
5              mindist←min(x.mbr,y.mbr)
6              maxdist←max(x.mbr,y.mbr)
7              if maxdist<distL then break
8              else if mindist≤distU then
9                    p_{x→y}←MCOPM(M,x,y,Q_td, Q_si, Q_sr)
10                   if p_{x→y} ≠ ∅ then
11                         Ω.add(y), Θ.add(x,Ω)
12 Return Θ
```

The IR-MCOPM algorithm requires multiple executions of the MCOPM algorithm. So the time complexity of MCOPM is $O(mn|V|\log|V|+mn|E|)$. Where, m is the size of tuples satisfying one keyword, and the length of tuples satisfying another keyword is n.

4.3 The Join Order of the Paths in Results (MCPJoin)

In the above algorithm, we can find all the paths that match the multi-constraints pattern P in D. Then we need to connect these paths according to the pattern P, but [7, 24] show that the connection order of different paths has impact effect on the execution of the algorithm.

Example 1. Given a pattern P having vertices (e.g.,$\{v_1, v_2, v_3, v_4\}$) and edges (e.g., $\{v_1 - v_2, v_2 - v_3, v_3 - v_4, v_4 - v_1\}$). Assume that there are 10, 50, 100, 1000 matched paths for these edges respectively. We consider the following two situations:

Situation 1: We run IR-MCOPM for edges $v_1 - v_2$, $v_2 - v_3$, $v_3 - v_4$ respectively and get at most 50,000 tuples by linking their respective results to form $v_1 - v_2 - v_3 - v_4$. For $v_4 - v_1$, we do not run IR-MCOPM directly, because we just confirm whether the constraints between the fourth object and the first one is satisfying $v_4 - v_1$ in P.

Situation 2: If we follow the connection order of $v_1 - v_4 - v_3 - v_2$, we will get at most 5,000,000 tuples. Then, we will check whether the constraints between the first and second object match $v_1 - v_2$ in P. In this case, its computational complexity is 100 times that of the above situation.

As can be seen from the above example 1, the computational complexity of Situation 1 is much higher than the computational complexity of Situation 2. Therefore, a good connection order determines the efficiency of linking paths to form an answer to pattern P. In order to improve the efficiency of the algorithm, the number of executions of IR-MCOPM should be minimized, especially for the case where there is a large number of matching pairs about a path in pattern P. The amount of calculation is relatively large, at this time we should try to reduce the execution of IR-MCOPM. Thus, a sampling estimation algorithm is needed to estimate the number of matches for a certain path. We use the estimation method shown in [7].

$$\Pr(|Y - E[Y]| \geq \eta E[Y]) \leq \rho \tag{8}$$

Where, $0 < \eta < 1$, $\rho = e^{\left(\frac{-su^2}{8}\right)}$, $Y = \sum_{i=1}^{s} X_i$ represents the number of samples needed to find s matching pairs. In our experiment, we set $\eta = 0.5$, $\rho = 0.25$. Since the purpose of the estimation is to get the topology connection order of the path, the values of η and ρ are not necessarily large.

In order to improve the integrality of the Algorithm, we present Algorithm 3 to guide the join order of the paths that are returned from Algorithm 2.

Algorithm 3: MCPJoin

Input: irTree.root, P, η, ρ, distL, distU

Output: ϒ, the join order of IR-MCOPM

```
1  ϒ←∅, I←∅, U←∅, Q←∅
2  for (v_i,v_j)∈P do
3      if dist(v_i,v_j)≥distL(v_i,v_j) then
4          Θ←IR-MCOPM(v_i,v_j)
5          I.add((v_i,v_j),Θ)
6  Select u from P, U.add(u)
7  for v∈nbr(u) do
8      if 0≤dist(u,v)≤distU(u,v) or distL(u,v)≤dist(u,v)≤distU(u,v) then
9          Q.add((u,v),sample(u,v))
10     else Q.add((u,v),I.get(u,v).size)
11 while Q.size>0 do
12     (v_i,v_j)←Q.pop()
13     ϒ.add(v_i,v_j)
14     if v_i∈U and v_j∈U then continue
15     select u from P, and u∉U
16     for v∈nbr(u) and v∈U do ϒ.add(u,v)
17     for v∈nbr(u) and v∉U do
18         if 0≤dist(u,v)≤distU(u,v) or distL(u,v)≤dist(u,v)≤distU(u,v)
19         then
20             Q.add((u,v),sample(u,v))
21         else Q.add((u,v),I.get(u,v).size)
22     U.add(u)
23  Return ϒ
```

Step 1: Input the IR-tree, multi-constraints pattern P, estimation parameters η, ρ, the lower distance distL, the upper distance distU.

Step 2: For any path (v_i, v_j) in the pattern P, if the distance relationship between two objects satisfies $\text{dist}(v_i, v_j) \geq \text{distL}(v_i, v_j)$, we need to perform Algorithm 2 and save the match in I(I is a hashmap, key is a paths tuple, value is the matching results).

Step 3: Select a vertex u randomly from P and add it to the set of vertices U that have been accessed. Visit the neighbor nodes of u, if the distance between u and v is $0 \leq \text{dist}(u, v) \leq \text{distU}(u, v)$ or $\text{distL}(u, v) \leq \text{dist}(u, v) \leq \text{distU}(u, v)$, we do not directly carry out Algorithm 2 on all objects. We perform a sampling function to estimate the number of (u,v). Otherwise, we get the exact number of matched pairs based on the corresponding paths in I and arrange the corresponding paths in ascending order according to the number of matching pairs.

Step 4: We dequeue the fewer pairs of paths and add them to the path join order set ϒ. Then we consider another vertex and execute Step 3 until there is no path in the priority queue Q. Finally, we return path join order set ϒ.

For satisfying certain distance constraints, we need to perform estimation. The time complexity of it is O(μmn|V|log|V|+μmn|E|) for each edge in P. μ is the sampling rate and its value is between zero and one. For each while loop, we need perform enqueue and dequeue operations. The time complexity of it is O(hlogh). Where h is the size of priority queue. Thus, the overall complexity of MCPJoin is O(μmn|V|log|V|+μmn|E|+ hlogh).

5 Experiments

5.1 Dataset

We use PoI (Point of Interest) as a dataset, which is obtained from the OpenStreetMap (https://www.openstreetmap.org), including Barcelona, Dubai, Paris, Rome, Tuscany, Trondheim, etc., covering 103,028 spatial objects, about 232,520 keywords. Each record consists of a unique identifier for the object, including geographic coordinates of latitude and longitude, and a set of keywords. The social relationships between the objects are constructed according to the NW small world model, each object is connected with the remaining 3–5 objects, and an edge is added between the randomly selected pair of objects with a probability of 0.3.

The H_OSTP algorithm is the most promising algorithm for solving the optimal path selection problem in social networks [29]. In order to study the performance of our proposed sample-based algorithm, we compare it with the H_OSTP algorithm. The H_OSTP algorithm and the MCPJoin algorithm are implemented using Java IntelliJ IDEA. The operating environment is Intel(R) Core(TM) i5-8265U CPU @ 1.60 GHz 1.80 GHz processor, 8 GB of memory, 512 GB SSD, and operating system is Windows 10.

In our experiments, the trust degree, social intimacy, and social values were randomly generated based on a uniform distribution and ranged from 0–1. The attenuation factor σ is 1.5. The constraints of trust degree, social intimacy, and social reputation are 0.05, 0.001, 0.3 respectively. The estimation parameters η and ρ are respectively 0.5, 0.25. Sampling threshold is set as $0.7 \times |O_i| \times |O_j|$ (In order to avoid unlimited sampling when no match is found, stop sampling when the number of samples reaches the threshold.).

We use 8 different structures (as shown in Fig. 2). The black rectangles in the figure represent spatial objects. The edges between black rectangles represent the constraints and paths of spatial objects. For each structure, we generate 3–6 specific patterns. And there is at least one match in the dataset.

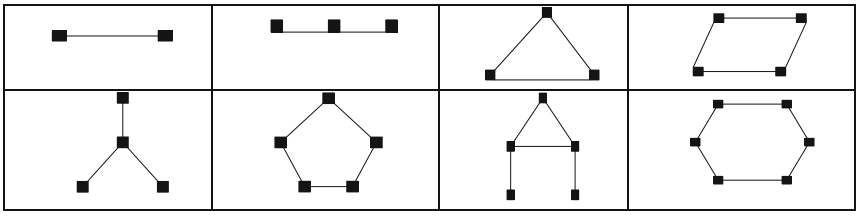

Fig. 2. Structures of path patterns.

We use the IR-tree index, where the fanout is 100, i.e., the maximum number of children of each node. The inverted object list of each keyword is stored in a single file on disk.

5.2 Performance

In order to evaluate the performance of our proposed algorithm, 150, 300, 450, 600, 750, 1050 objects are extracted from the PoI dataset as a subset. The details of the data set are shown in Table 1.

Table 1. The datasets used in experiments.

Datasets	Objects	Edges	Keywords
1	150	788	375
2	300	1,073	3,002
3	450	1,790	4,503
4	600	2,925	6,500
5	750	3,121	7,810
6	1,050	5,456	13,640

In each subset, the eight structures shown in Fig. 2 are independently executed for the MCPJoin algorithm and the H_OSTP algorithm, each of which is executed 3–6 times, and the results are averaged. To enhance comparability, the MCPJoin algorithm and the H_OSTP algorithm perform the same query path pattern each time. The comparison results are shown in Fig. 3.

As can be seen from Fig. 3, when the number of vertices in the pattern P is 2 and the number of objects in the spatial object is small, the H_OSTP algorithm can achieve better performance. There are two reasons for this: (1) The H_OSTP algorithm does not need to establish an IR-tree index for each vertex, thus reducing the building time of the index; (2) When there are only two objects in the pattern P, the H_OSTP algorithm does not involve the connection of the path problem. However, when the number of vertices in the pattern P exceeds 2 and the number of objects in the spatial object increases gradually, the performance of the H_OSTP algorithm decreases sharply. When the number of vertices in the pattern P is 3, the MCPJoin algorithm saves more than the H_OSTP algorithm 57.39% of the time, when the number of vertices in mode P is 4, the MCPJoin algorithm saves 65.28% of the time compared with the H_OSTP algorithm. When the number of vertices in P is 5, the MCPJoin algorithm saves 85.45% of the time compared with the H_OSTP algorithm. When the number of vertices in the pattern P is 6, the MCPJoin algorithm saves an average of 67.12% of the time compared to the H_OSTP algorithm. As can be seen from Table 2, if H_OSTP wants to achieve the same path search result as MCPJoin, it will cost more runtime. The main reasons are as follows: (1) The MCPJoin algorithm creates an IR-tree for spatial objects, which is beneficial to reduce the search space. (2) The MCPJoin algorithm uses a sampling-based estimation algorithm to limit the search time and ensure the efficiency of the path connection process.

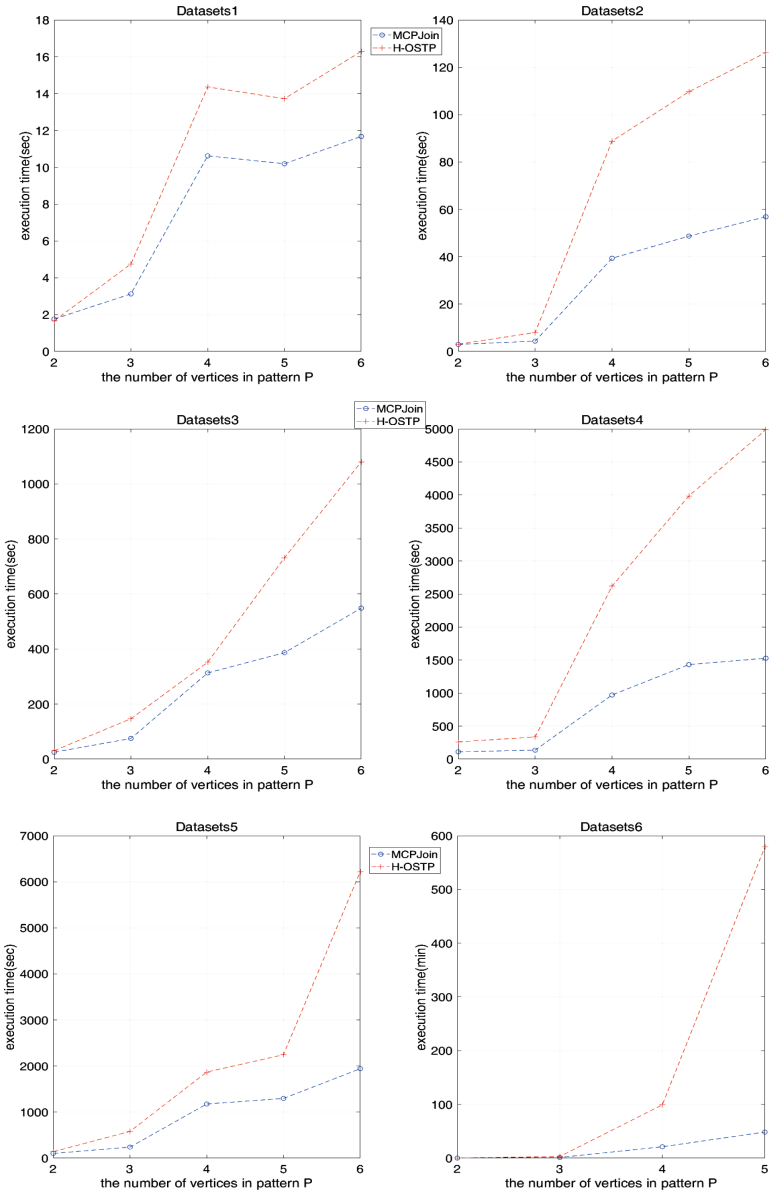

Fig. 3. The comparison of runtime about the number of vertices in P.

Therefore, based on the above analysis, the MCPJoin algorithm is nearly twice as efficient as the H_OSTP algorithm, especially when the number of spatial objects is large, the H_OSTP algorithm is difficult to give a matching result in a reasonable time.

Table 2. The comparison of average runtime about different vertices in P.

Vertices	2	3	4	5	6
MCPJoin	42.6814 (s)	90.0589 (s)	630.4801 (s)	1.0143e + 003 (s)	817.5096 (s)
H_OSTP	74.5253 (s)	211.3464 (s)	1.8159e + 003 (s)	6.9723e + 003 (s)	2.4860e + 003 (s)
Difference	42.73% less	57.39% less	65.28% less	85.45% less	67.12% less

6 Conclusion

In this paper, we propose a multi-constrained path selection problem based on spatial pattern matching. Based on the characteristics of spatial objects, IR-tree is used as the index structure, and IR-MCOPM is based on IR-tree to map paths from pattern P to social network. The MCPJoin algorithm based on IR-MCOPM guiding the path join order show that the proposed algorithms are more efficient than the H_OSTP algorithm.

In the future, we plan to obtain the weight of the path quality function through the machine learning method, and feed the obtained path back to the path search process through the neural network method, thereby improving the quality of the path search algorithm. Another research content is used to add time information to social networks, and extend this problem to time and space issues.

Acknowledgments. The work was supported by Natural Science Foundation of Shandong Province (No. ZR2016FQ10), National Natural Science Foundation of China (No. 6167126, No. 61802217), Key Research and Development Program of Shandong Province (No. 2016GGX101007).

References

1. Zhang, D., et al.: Keyword search in spatial databases: towards searching by document. In: ICDE, pp. 688–699. IEEE (2009)
2. Guo, T., Cao, X., Cong, G.: Efficient algorithms for answering the m-closest keywords query. In: SIGMOD, pp. 405–418. ACM (2015)
3. Deng, K., Li, X., Lu, J., Zhou, X.: Best keyword cover search. TKDE **27**(1), 61–73 (2015)
4. Choi, D., Pei, J., Lin, X.: Finding the minimum spatial keyword cover. In: ICDE, pp. 685–696. IEEE (2016)
5. Rocha-Junior, J.B., Gkorgkas, O., Jonassen, S., Nørvåg, K.: Efficient processing of top-k spatial keyword queries. In: Pfoser, D., et al. (eds.) SSTD 2011. LNCS, vol. 6849, pp. 205–222. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22922-0_13
6. Pengfei, Z.: Research on related issues of spatial keyword query. Zhejiang University (2018)
7. Fang, Y., Cheng, R., Cong, G., Mamoulis, N., Li, Y.: On spatial pattern matching, pp. 293–304 (2018). https://doi.org/10.1109/icde.2018.00035
8. Fan, W., Li, J., Ma, S., Tang, N., Wu, Y., Wu, Y.: Graph pattern matching: from intractable to polynomial time. In: VLDB 2010, pp. 264–275 (2010)
9. Cheng, T.S., Gadia, S.K.: A Pattern matching language for spatio-temporal databases (1994)
10. Cheng, J., Yu, J.X., Ding, B., Yu, P.S., Wang, H.: Fast graph pattern matching. In: ICDE (2008)
11. Carletti, V., et al.: Challenging the time complexity of exact subgraph isomorphism for huge and dense graphs with VF3. TPAMI (2017)

12. Zhang, P., Lin, H., Yao, B., et al.: Level-aware collective spatial keyword queries. Inf. Sci. **378**, 194–214 (2017)
13. Hariharan, R., Hore, B., Li, C., Mehrotra, S.: Processing spatial-keyword (SK) queries in geographic information retrieval (GIR) systems. In: SSDBM, p. 16 (2007)
14. Cary, A., Wolfson, O., Rishe, N.: Efficient and scalable method for processing top-k spatial Boolean queries. In: Gertz, M., Ludäscher, B. (eds.) SSDBM 2010. LNCS, vol. 6187, pp. 87–95. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13818-8_8
15. Cong, G., Jensen, C.S., Wu, D.: Efficient retrieval of the top-k most relevant spatial web objects. PVLDB **2**(1), 337–348 (2009)
16. Khodaei, A., Shahabi, C., Li, C.: Hybrid indexing and seamless ranking of spatial and textual features of web documents. In: Bringas, P.G., Hameurlain, A., Quirchmayr, G. (eds.) DEXA 2010. LNCS, vol. 6261, pp. 450–466. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15364-8_37
17. Wu, D., Yiu, M.L., Cong, G., et al.: Joint top-K spatial keyword query processing. IEEE Trans. Knowl. Data Eng. **24**, 1889–1903 (2012)
18. Zheng, B., Zheng, K., Jensen, C.S., et al.: Answering why-not group spatial keyword queries. IEEE Trans. Knowl. Data Eng. **32**, 26–39 (2018)
19. Qian, Z., Xu, J., Zheng, K., et al.: Semantic-aware top-k spatial keyword queries. World Wide Web Internet Web Inf. Syst. **21**(3), 573–594 (2018)
20. Mamoulis, N., Papadias, D.: Multiway spatial joins. TODS **26**(4), 424–475 (2001)
21. Zou, L., Chen, L., Ozsu, M.T.: Distance-join: pattern match query in a large graph database. PVLDB **2**(1), 886–897 (2009)
22. Chen, L., Cong, G., Jensen, C.S., Wu, D.: Spatial keyword query processing: an experimental evaluation. PVLDB, 217–228 (2013)
23. Tong, H., Faloutsos, C., Gallagher, B., Eliassi-Rad, T.: Fast best-effort pattern matching in large attributed graphs. In: KDD (2007)
24. Zou, L., Chen, L., Ozsu, M.T.: Distance-join: pattern match query in a large graph database. In: VLDB (2009)
25. Jing, Y., Yanbing, L., Zhang, Y., Mengya, L., Jianlong, T., Li, G.: Summary of large-scale graph data matching technology. Comput. Res. Dev. **52**(02), 391–409 (2015)
26. Henzinger, M.R., Henzinger, T., Kopke, P.: Computing simulations on fifinite and infifinite graphs. In: FOCS 1995 (1995)
27. Sokolsky, O., Kannan, S., Lee, I.: Simulation-based graph similarity. In: Tools and Algorithms for the Construction and Analysis of Systems (2006)
28. Liu, G., Zheng, K., Liu, A., et al.: MCS-GPM: multi-constrained simulation based graph pattern matching in contextual social graphs. IEEE Trans. Knowl. Data Eng. **30**, 1050–1064 (2017)
29. Liu, G., Wang, Y., Orgun, M.A., et al.: A heuristic algorithm for trust-oriented service provider selection in complex social networks. In: IEEE International Conference on Services Computing. IEEE Computer Society (2010)
30. Berger, P., Luckmann, T.: The Social Construction of Reality: A Treatise in the Sociology of Knowledge. Anchor Books, New York (1966)
31. Zobel, J., Moffat, A.: Inverted fifiles for text search engines. ACM Comput. Surv. **38**(2), 56 (2006)