# Multi-objective Spatial Keyword Query with Semantics

Jing Chen[1], Jiajie Xu[1(✉)], Chengfei Liu[2], Zhixu Li[1], An Liu[1], and Zhiming Ding[3]

[1] Department of Computer Science and Technology, Soochow University, Suzhou, China
20164227012@stu.suda.edu.cn, {xujj,zhixuli,anliu}@suda.edu.cn

[2] Faculty of SET, Swinbourne University of Technology, Melbourne, Australia
cliu@swin.edu.au

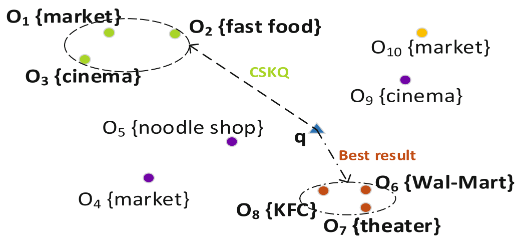
[3] Department of Computer Science and Technology, Beijing University of Technology, Beijing, China
zmding@bjut.edu.cn

**Abstract.** Multi-objective spatial keyword query finds broad applications in map services nowadays. It aims to find a set of objects that can cover all query objectives and are reasonably distributed in spatial. However, existing approaches mainly take the coverage of query keywords into account, while leaving the semantics behind the textual data to be largely ignored. This limits us to return those rational results that are synonyms but morphologically different. To address this problem, this paper studies the problem of multi-objective spatial keyword query with semantics. It targets to return the object set that is optimum regarding to both spatial proximity and semantic relevance. We propose an indexing structure called LIR-tree, as well as two advanced query processing approaches to achieve efficient query processing. Empirical study based on real dataset demonstrates the good effectiveness and efficiency of our proposed algorithms.

## 1 Introduction

Spatial keyword query is widely used in location based service (LBS) systems to recommend users the needed services or places to visit. The study on this topic has attracted a great deal of attention. Existing methodologies mainly study the efficient retrieval of spatial web objects that can best match the query in terms of both spatial and textual relevances. The spatial keyword query itself sometimes has multiple objectives, which may lead to none or few objects that can fully cover all keywords in query. To address this problem, [5] returns a group of objects that can cover all required keywords with reasonable spatial distribution. But the keyword match cannot help us to find out those objects with highly related semantics but low similarity in spellings, such as *market* and *Wal-Mart*. This limitation motivates us to investigate other approaches to capture the semantic relatedness to multi-objective spatial keyword queries.

S. Candan et al. (Eds.): DASFAA 2017, Part II, LNCS 10178, pp. 34–48, 2017.
DOI: 10.1007/978-3-319-55699-4_3



| Query Objectives | Textual Descriptions |
|---|---|
| $q.\Psi_1$ | market |
| $q.\Psi_2$ | fast food |
| $q.\Psi_3$ | cinema |

**Query Objectives Matrix**

**Fig. 1.** Distribution of Spatial Web Objects

*Example 1.* Figure 1 shows an example with ten spatial web objects, each has a geographical location and a set of keywords. A user issues a query with three objectives described by *market*, *fast food* and *cinema* respectively. By using traditional methods [7,9] to process each objective in query independently, the objects $\{O_4, O_5, O_9\}$ are returned because of the spatial and textual similarities to query. Alternatively by using the collective spatial keyword querying method [5], the search engine tends to return a more qualified result such as $\{O_1, O_2, O_3\}$, because they are coherent in spatial and been close to the query together. However if we check the semantics of query objectives more carefully, instead of $\{O_1, O_2, O_3\}$, we can easily observe that $\{O_6, O_7, O_8\}$ is the set of objects that should be returned, because they are best matched in spatial, and all objectives in the query can be fully matched in semantics. The key issue is how to take the semantics into account and process the query efficiently.

To represent the semantics of spatial web objects and query objectives, we can apply powerful tools in the field of machine learning, such as probabilistic topic model or word embedding. By using them on textual descriptions, query objectives (e.g., *market* in $q$ of Fig. 1) and spatial web objects are represented as high dimensional vectors called topic distributions in semantic space. A topic distribution indicates the semantic relevance between a textual description and a latent topic, and accordingly, the similarity between an object and an objective in query can be measured on top of their topic distributions. In this way it is possible to find the collective object set that can satisfy all query objectives while coherent in spatial and close to the query point.

While the incorporation of semantics helps us to return more meaningful feedbacks, the query processing becomes more challenging and time-consuming for three main reasons: firstly, finding the optimal result (the subset according to spatial and semantic similarity) is an NP complete problem, which cannot be solved in a polynomial time; secondly, existing spatial keyword indices, such as IR-tree [4], cannot be directly used to organize the information of spatial web objects because of its difficulties in representing their topic distributions regarding to semantics. Last but not the least, the high dimensionality of vector (topic distribution in semantics) deteriorates the pruning effectiveness in query processing due to the large dead space.

To address all above difficulties, we propose a novel query processing mechanism that has good efficiency and precision. To ensure the pruning effect in semantic space, we take advantage of the locality sensitive hashing (LSH) to hash the objects by their high dimensional topic distributions. Each bucket is understood as a semantic tag, and the LSH mechanism ensures that objects in the same bucket to have consistent semantic meanings. We design a candidate bucket set oriented searching mechanism to reduce the search space. It retrieves and compares local result for each candidate bucket set, and finally derives a result in global optimum. In addition, a more efficient approach is proposed to avoid checking all candidate bucket sets while ensuring high accuracy of the result. The main contributions of the paper can be briefly summarized as follows:

- We formalize a probabilistic topic model based similarity measure between a multi-objective query and a set of objects;
- We design a semantic hashing based algorithm by applying LSH index structure, so that collective objects can be derived by making use of the collective spatial keyword querying technologies.
- We propose a novel mechanism that can start from a good result directly, and then guide us to improve the result while ensuring the accuracy by distance based replacement strategy.
- We conduct an extensive experiment analysis based on real spatial databases and make the comparisons with baseline algorithm, and then demonstrate the efficiency of our method.

## 2 Preliminaries and Problem Definition

In this section, we introduce some preliminaries about probabilistic topic model and then formalize the problem of this paper.

### 2.1 Probabilistic Topic Model

Probabilistic topic model is a well-known technique on theme interpretation and document classification. In this paper, we apply one of the most frequently used probabilistic topic models, i.e. the *Latent Dirichlet Allocation* (LDA) model to understand the semantic meanings of textual descriptions. In LDA, each latent topic, or topic in short, is a feature that represents a semantic meaning. By carrying out statistical analysis on the large amount of textual descriptions, the LDA model automatically derives the semantic relevance of a textual description to all latent topics, known as topic distribution defined as follows:

**Definition 1** (*Topic Distribution*)**.** Given a textual description $W$, a topic distribution derived from LDA is a high dimensional vector that describes the semantic relevance between the textual description and each latent topic. We use $TD_W$ to denote the topic distribution of $W$ over finite latent topics, and a component $TD_W[i]$ indicates the relevance between $W$ and the $i_{th}$ latent topic.

**Table 1.** Topic distributions of textual descriptions

| Textual descriptions | Topics | | | | |
|---|---|---|---|---|---|
| | Exercise | Movie | Drink | Shop | Food |
| market (in $O_1$, $O_{10}$) | 0.09 | 0.09 | 0.09 | 0.64 | 0.09 |
| fast food (in $O_2$) | 0.04 | 0.04 | 0.16 | 0.04 | 0.72 |
| cinema (in $O_3$, $O_9$) | 0.07 | 0.72 | 0.07 | 0.07 | 0.07 |
| noodle shop (in $O_5$) | 0.07 | 0.07 | 0.07 | 0.07 | 0.72 |
| Wal-Mart (in $O_6$) | 0.07 | 0.07 | 0.07 | 0.72 | 0.07 |
| theater (in $O_7$) | 0.04 | 0.84 | 0.04 | 0.04 | 0.04 |
| KFC (in $O_8$) | 0.03 | 0.03 | 0.03 | 0.03 | 0.88 |

*Example 2.* Table 1 shows the LDA interpretation on all the spatial web objects in Fig. 1. Each tuple in Table 1 is a topic distribution over five topics. Each component is the relevance between the textual description and a specific topic, for example, $TD_{market}[1] = 0.09$ means the relevance between *market* and *exercise* is 0.09. We can learn from Table 1 that *cinema* has high coherence with *theater* due to $TD_{cinema}[2] = 0.72$ and $TD_{theater}[2] = 0.84$ while in contrast that $KFC$ is distinct to *cinema* because of $TD_{KFC}[2] = 0.03$.

## 2.2 Problem Definition

A spatial web object is a place of interest in LBS systems, and it is formalized as $o = (o.\lambda, o.\psi)$ where $o.\lambda$ is the position of $o$ and $o.\psi$ is the textual information for describing $o$. A user issues a multi-objective query $q = (q.\lambda, q.\Psi)$, where $q.\lambda$ represents a geographical location, and $q.\Psi$ is a set of query objectives which are textual descriptions for describing an activity intention. In the rest of this paper, we simply use *objects* to represent *spatial web objects*.

**Definition 2** (*Spatial Distance*)**.** The objects in the result set are supposed to be not only close to query, but also close with each other. We thus follow collective spatial keyword query [5] and measure the spatial distance $D_S$ to range [0,1] from a query $q$ to an object set $O$ as follow:

$$\mathcal{D}_S(q, O) = \beta \times max_{o_i \in O} \ (s_d(q, o_i)) \\ + (1-\beta) \times max_{o_i, o_j \in O}(s_d(o_i, o_j)) \quad (1)$$

where $\beta \in [0, 1]$ is a user-specified weight parameter, $s_d(q, o_i) = \frac{2}{1+e^{||q.\lambda, o_i.\lambda||}} - 1$ is the normalized spatial distance between query $q$ and object $o_i$ by sigmoid function. The spatial measure allows us to find a set of objects close to query and have spatial coherence with each other. That means, the objects are rationally distributed in spatial when $D_S(q, O)$ is small.

**Definition 3** (*Semantic Distance*)**.** Semantic distance $\mathcal{D}_T$ between a query $q$ and an object set $O$ can be measured on top of their topic distributions through

LDA. By calculating the distance between high dimensional vectors, we define $\mathcal{D}_T$ to range [0,1] by using the sigmoid function as follow:

$$\mathcal{D}_T(q, O) = \sum_{q.\Psi_i \in q.\Psi} min_{o_j \in O}(d_T(q.\Psi_i, o_j)) \tag{2}$$

such that,

$$d_T(q.\Psi_i, o_j) = \frac{2}{1 + e^{-\sqrt{\Sigma(TD_{q.\Psi_i}[z] - TD_{o_j}[z])^2}}} - 1 \tag{3}$$

where $\mathcal{D}_T(q, O) \in [0, 1]$. It is obvious that when semantic distance is smaller, the query $q$ and a correspond object $o$ are more relevant in semantics.

**Definition 4** (*Distance*)**.** By combining spatial distance $\mathcal{D}_S(q, O)$ and semantic distance $\mathcal{D}_T(q, O)$, we define the distance $\mathcal{D}ist(q, O)$ of query $q$ and object set $O$ in Equation below.

$$\mathcal{D}ist(q, O) = \alpha \times \mathcal{D}_S(q, O) + (1 - \alpha) \times \mathcal{D}_T(q, O) \tag{4}$$

where $\alpha \in [0, 1]$ is a user-specified weight parameter that balance spatial distance and semantic distance.

***Problem Statement.*** Given an object set $O$ and a query $q = (q.\lambda, q.\Psi)$, the multi-objective spatial keyword query (MoSKQ, in short) in this paper aims to return a subset $O'$ of objects $O(O' \subset O$, $|O'| \leqslant |q.\Psi|)$, such that $\forall O'' \subset O$, $\mathcal{D}ist(q, O') \leqslant \mathcal{D}ist(q, O'')$.

## 3 Baseline Algorithm

In this section, we propose a baseline algorithm which seeks to find the optimal result within a subspace incrementally. A lower bound and an upper bound are used to stop the searching process in the middle if possible.

Starting from a search region centered at the query $q$ with a radius $r$, we execute an exhaustive search to get the best object set $R(|R| \leqslant |q.\Psi|)$ which minimizes the distance to the query according to *Definition* 4. If needed, we enlarge the search radius $r = r + \Delta r$ and search all combinations of objects in this region to find a best object set $R'$. During this process, a set $S$ is used to store all the solutions found. In this process, we dynamically maintain an upper bound $\mathcal{UB} = min_{R \in O}(\mathcal{D}ist(q, R))$ and a lower bound $\mathcal{LB} = \alpha \times \beta \times (\frac{2}{1+e^{-r}} - 1)$ which equals to part of spatial distance.

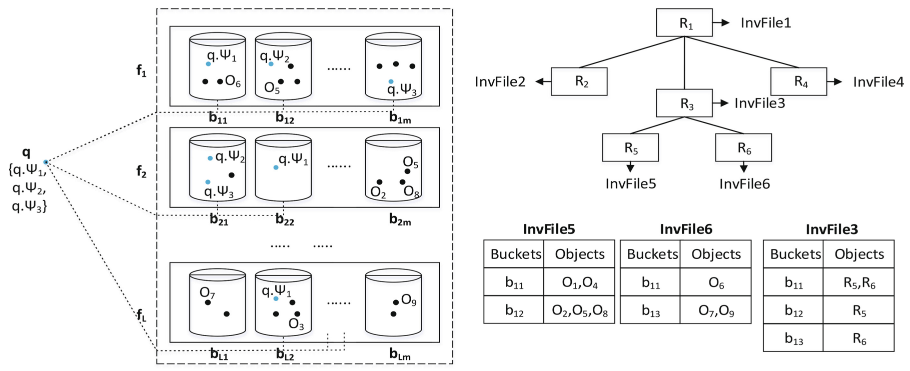
During the process, if $\mathcal{UB} \leqslant \mathcal{LB}$ or the search radius extends to the most distant object, the algorithm terminates and returns the best group in the solution set $S$, because the spatial distance of all unprocessed objects are no less than the distance to query of the found result object sets. However, this algorithm may require an exhaustive search sometimes because the bound is relative loose. Therefore more efficient approaches are required to find the results.

## 4 Semantic Hashing Based Algorithm

In this section, we propose a novel solution called semantic hashing based algorithm (SH-based algorithm in short) to speed up the querying process. In Sect. 4.1, we introduce the details of the LSH based indexing structure. Section 4.2 plots the search algorithm over the index.

### 4.1 Index Structure

In this subsection, we devise a new index, namely LIR-tree, based on LSH and IR-tree. As is known, LSH [3,8,17] is a method widely used for similarity search in high dimension. We first utilize LSH to preprocess all objects in the dataset, i.e., hash the object into buckets based on their topic distributions. Every bucket in LSH can be regarded as the tag of the semantic meanings of the objects in this bucket. That is to say, the objects in the same bucket are considered to be similar in semantics. In this way, all the objects in the dataset derive the bucket ids that they are hashed into, which makes the semantic similarity search possible based on the bucket ids. Then we use IR-tree [6,16,19] to organize the objects according to their geographical locations and corresponding bucket ids for a given query with specified location and bucket ids.



InvFile5

| Buckets | Objects |
|---|---|
| $b_{11}$ | $O_1,O_4$ |
| $b_{12}$ | $O_2,O_5,O_8$ |

InvFile6

| Buckets | Objects |
|---|---|
| $b_{11}$ | $O_6$ |
| $b_{13}$ | $O_7,O_9$ |

InvFile3

| Buckets | Objects |
|---|---|
| $b_{11}$ | $R_5,R_6$ |
| $b_{12}$ | $R_5$ |
| $b_{13}$ | $R_6$ |

**Fig. 2.** An example of LIR-tree

***LSH part.*** The LSH is a well-known index scheme for high-dimensional similarity search with the basic idea to use a family of locality-sensitive hashing functions to map the objects into the same buckets with high probability. LSH hash families have the property that objects close to each other have a higher colliding probability than those far apart, which is determined by different distance measure functions. In this paper, we use the hash family proposed by Datar et al. [8] based on $p$-stable distributions [12], which is defined as:

$$h(p) = \lfloor \frac{a \times p + b}{W} \rfloor \tag{5}$$

where $a$ is a random topic distribution vector, $W$ represents the width of the hash function, $b$ is a random variable belongs to $[0, W]$. All the objects in the dataset are divided into corresponding buckets based on their topic distributions. Each bucket can be considered as the semantic tag of the objects in this bucket and the objects in the same bucket have high proximity in semantics. We record the geographical location and the bucket ids that every object in dataset are hashed into.

***IR-tree part.*** The IR-tree part of LIR-tree is similar to the conventional inverted R-tree, except that we store the inverted list of the buckets of the objects derived by LSH, rather than the keywords that describe the objects. All the objects in the dataset are organized using the R-tree according to their geographical location. Since the objects also have the bucket ids that they are hashed into, we build the inverted list of the R-tree node in a bottom up fashion. The inverted list of both leaf node and non-leaf node includes the *buckets* and the *objects* that are hashed to this bucket.

## 4.2 Search Algorithm

In this subsection, we propose a search algorithm that prunes the search space effectively over the proposed index. The prune process is complished on topic layer and spatial layer respectively.

Let us consider how to match all query objectives first. Recall that all objects in the dataset have a topic distribution after applying the LDA model to interpret their textual descriptions. The objects are then hashed into buckets by LSH on top of their topic distributions. By using LSH, objects in a same bucket are supposed to be consistent in semantics, each bucket can thus be understood as a semantic tag. Given a query q, we can derive a topic distribution for each objective in q, and then hash the query objectives into the LSH buckets in the same way to objects. By taking advantage of the LSH structure, we can simply evaluate if an object can match a query objective if they share a same semantic tag (i.e. in a same LSH bucket), and accordingly, the semantic distance can be rewritten to:

$$d_T(q.\Psi_i, o) = \begin{cases} 0 & \exists B_{ij} : q.\Psi_i \in B_{ij} \land o \in B_{ij} \\ \infty & otherwise \end{cases} \tag{6}$$

where $q$ and $o$ are the query and the object respectively. Equation 6 means that an object can be matched to a query objective in semantics if they share at least one LSH bucket, and they have no semantic relevance otherwise.

On top of the LSH based semantic distance defined above, the next issue is to justify if a given object set is semantically relevant to all query objectives. Conceptually, the relevance requires us to find an object from the set to share a same bucket for each query objective (having a same semantic tag). To define the semantic relevance more clearly, we further define the concept of *candidate bucket set* as follows.

**Definition 5** (*Candidate bucket set*)**.** A candidate bucket set is a smallest unit of buckets to ensure the relevance to a query about its objectives. Given a query $q$, a candidate bucket set $cbs$ satisfy the following two requirement: (1) containment. A $cbs$ contains at least one bucket in bucket set of each query objective $q.\Psi_i \in q.\Psi$ such that $\mathrm{BS}(q.\Psi_i) \cap cbs \neq \varnothing$, where $\mathrm{BS}(q.\Psi_i)$ is the set of buckets which $q.\Psi_i$ is hashed to; (2) minimum. The above condition fails for each subset of these buckets, i.e., $cbs' \subsetneq cbs$.

The candidate bucket set ensures that all query objectives can be matched. Given a set of objects O, it can be returned if the union of buckets containing an object in O can cover a candidate bucket set of query q. In Fig. 2, $\{b_{1M}, b_{21}, ..., b_{L2}\}$ is a candidate bucket set which covers all query objectives, but $\{b_{12}, b_{22}, ..., b_{L2}\}$ is not a candidate bucket set which covers only part of query objectives $\{q.\Psi_1, q.\Psi_2\}$.

Here we describe the searching mechanism. The target of SH-based algorithm search is to find the object set such that: (1) its related bucket set covers at least one candidate bucket set to meet all query objectives; (2) the distance to query is the minimum. We use a candidate bucket set oriented searching mechanism. For each candidate bucket set, if each bucket is regarded as a keyword (denoting a semantic tag), our problem can be transfered to the well studied collective spatial keyword query [5]. We apply the Top-Down Search algorithm in [5] to obtain the best object set for the given candidate bucket set (line 7). The basic idea of the Top-Down Search algorithm is to perform a best-first search on the IR-tree to find the covering node sets, such that some objects from these nodes can constitute a group to cover all required buckets in the set. We process the covering node set with the lowest distance to query to find covering node sets from their child nodes. While reaching a covering node set consisting of leaf nodes, a group of objects with the lowest distance to query can be found by performing an exhaustive search (lines 6–10). The Top-Down Search algorithm return the exact result set and it is invoked $L^{|q.\Psi|}$ times according to Lemma 1.

**Lemma 1.** *There are* $\boldsymbol{L}$ *hash functions and query* $q$ *have* $|q.\Psi|$ *query objectives. The SH-based Algorithm would retrieve collective objects for* $L^{|q.\Psi|}$ *times.*

**Proof.** Assuming that there are $L$ hash functions, each query objective $q.\Psi_i$ can be put into $L$ buckets as a bucket set $BS(q.\Psi_i)$ after hashing. The quantity of this bucket set is L, namely $|BS(q.\Psi_i)| = L, i \in [1, |q.\Psi|]$. Then the $|q.\Psi|$ query objectives correspond to $|q.\Psi|$ bucket sets. All candidate bucket sets produced by Cartesian product $BS(q.\Psi_1) \times BS(q.\Psi_2) \times ... \times BS(q.\Psi_{|q.\Psi|})$ and obviously the size of these sets is $L^{|q.\Psi|}$. Therefore, this query step will be repeated $L^{|q.\Psi|}$ times. Lemma 1 can be proven. ■

This method avoids the worst situation in which the whole search region needs to be scrutinized. The search process is subject to at most $BS(q.\Psi_1) \times ... \times BS(q.\Psi_{|q.\Psi|})$ candidate bucket sets, each of them calls for a Top-Down Search whose time complexity is $|q.\Psi| - |N| + 1$, where $N$ represents the number of nodes which cover the query keywords and each node contributes at least one

**Algorithm 1.** *SH* based Search Algorithm

```
  Input: IR-tree $ir$, query $q$, $\lambda$
  Output: a set of objects $O$
1 $O \leftarrow \varnothing$;
2 $CBS \leftarrow BS(q.\Psi_1) \times BS(q.\Psi_2) ... \times BS(q.\Psi_{|q.\Psi|})$;
3 for each cbs in $CBS$ do
4     $O'$ and $Dist(q, O') \leftarrow$ TopDownSearch($q, cbs$);
5     if $Dist(q, O') < Dist(q, O)$ then
6         $Dist(q, O) = Dist(q, O')$;
7         $O = O'$;
8 return $O$ and $Dist(q, O)$;
```

object to the final result [5]. The time complexity of SH-based Algorithm is $L^{|q.\Psi|} \times (|q.\Psi| - |N| + 1)$ which cannot be solved in polynomial time either. The computational overhead will rapidly increase when the number of query objectives grows.

## 5 Distance Based Replacement Algorithm

This section presents a novel strategy called Distance Based Replacement (DBR) Algorithm, which starts at the SH-based algorithm but aims to find a high quality result more efficiently. Instead of taking every possible candidate bucket set as input, the DBR algorithm randomly sample a number of candidate bucket sets and derive a result based on SH-based algorithm. Then it aim to improve the result by replacement iteratively. Besides, the DBR Algorithm takes both spatial layer and topic layer into consideration, and LIR-tree is used to accelerate query processing as well.

We randomly sample some candidate bucket sets to obtain object sets by invoking SH-based Algorithm and choose the object set with the minimum distance to query from these object sets before replacement. Objects in this set are replaced individually and iteratively until a stable object set is found according to Lemma 2. A stable object set is obtained by iteratively replacing object until no reduction can be achieved on distance to query. The critical operation is that how to set the standard for replacement. In the procedure of replacement, an object in initial object set replaced by an object which beyond this set and has farther $\mathcal{MG}$ than others. The important concept *marginal gain* $\mathcal{MG}$ defined as

$$\mathcal{MG}(o_i, o_j) = Dist(q, O) - Dist(q, O') \tag{7}$$

where $O = \{o_1, .., o_i, .., o_n\}$ is the initial object set and $O' = \{o_1, .., o_j, .., o_n\}$ represents a replaced object set where $o_i$ is replaced by $o_j$. The object $o_i$ should be replaced by $o_j$ while $\max_{o_j \in O}(\mathcal{MG}(o_i, o_j))$ and $\mathcal{MG}(o_i, o_j) > 0$. The replacement strategy terminates when no object can be found to touch positive marginal gain, i.e., for each object $o'_i$ in object set, $\forall o'_j \in O, \mathcal{MG}(o'_i, o'_j) < 0$. The stable object set is final result of DBR Algorithm. And this result may have lower distance to query than the object set found by SH-based Algorithm.

**Lemma 2.** *Given dataset and bucket sets related to query objectives. The stable object set can be found through distance based replacement strategy.*

**Proof.** We assume that the object set after DBR Algorithm is not stable. Thus there must exist an object $o_i$ in this object set which can be replaced by another object $o_j$ and $\mathcal{MG}(o_i, o_j) > 0$. According to terminating condition, replacement will be continued until the stable object set is found. Therefore, the object set must be stable. Lemma 2 has been proved. ■

**Algorithm 2.** Distance based Replacement Search Algorithm

```
Input: IR-tree $ir$, query $q$, $\lambda$
Output: a set of objects $V$
1 $V \leftarrow \varnothing$;
2 $CBS \leftarrow BS(q.\Psi_1) \times BS(q.\Psi_2)... \times BS(q.\Psi_{|q.\Psi|})$;
3 $Somecbs \leftarrow$ randomSome($CBS$);
4 $V$ and $Dist(q, V) \leftarrow getMinimum(q, Somecbs)$;
5 for object $o_i$ in $V$ do
6     for object $o_j$ in $O$ do
7         $(o'_i, o'_j) = argmax(\mathcal{MG}(o_i, o_j))$;
8         V=Replace($o_i$,$o'_j$);
9 return V;
```

The pseudocode is shown in Algorithm 2. Firstly, we obtain all candidate bucket sets $CBS$ which is the cartesian product of bucket sets (line 2). Next, we randomly select some candidate bucket sets (line 3). Then we apply Top-Down Search Algorithm to query $q$ and each $cbs(cbs \in Somecbs)$ and obtain the object set $V$ with minimum distance $\mathcal{D}ist(q, V)$ (line 4). Then, we compute $\mathcal{MG}(o_i, o_j)$ to find an object $o_j$ with the maximum marginal gain and apply DBR strategy to replace it until finding a stable object set (line 5–8). Through DBR strategy, the problem of time consuming caused by messive candidate bucket sets has been solved. Assuming there are $n$ objects in datasets and the stable object set is found after $m$ replacement, the time complexity of DBR algorithm is $O(n \times m)$.

## 6 Experiment Study

In this section, we conduct extensive experiments on real datasets to evaluate the performance of our proposed algorithms.

## 6.1 Experiment Settings

We create the real datasets by using the online check-in records of Foursquare within the areas of New York City. Each record contains the user ID, venue with geographical location (place of interest) and the tips written in English. We put the records belonging to the same object to form textual descriptions of the objects, and the textual descriptions for each place are interpreted into a probabilistic topic distribution by the *LDA* model. The number of objects in this dataset is 206,097 in sum.

**Table 2.** Default values of parameters

| Parameter | Default value | Description |
|---|---|---|
| $\lvert q.\Psi \rvert$ | 3 | Number of query objectives |
| $\alpha$ | 0.5 | Weight factor for distance |
| $\beta$ | 0.5 | Weight for spatial distance |
| $L$ | 100 | Number of hash table |
| $M$ | 8 | Number of *LSH* function |
| $t$ | 50 | Number of latent topics |

We compare the query time cost of proposed algorithms respectively. The default values for parameters are given in Table 2. All algorithms are implemented in Java and run on a PC with a 2-core Intel CPU at 2.5 GHz and 4 GB memory.

## 6.2 Performance Evaluation

*(1) Comparisons of proposed methods*

We evaluate the efficiency and accuracy of the proposed methods by varying the parameters in Table 2.

**Effect of $\lvert q.\Psi \rvert$.** We investigate the effect of $\lvert q.\Psi \rvert$ on the efficiency and accuracy of the proposed algorithms. As shown in Fig. 3, the DBR algorithm has much less time consumption than both the baseline and the SH-based algorithms. The query time of the SH-based algorithm and the baseline algorithm are exponentially increasing with the growth of the number of query objectives, while in contrast the query time for the DBR algorithm increases smoothly. Within our expectations, the baseline and DBR algorithms have similar accuracy performance, better than the SH-based algorithm in all $\lvert q.\Psi \rvert$ settings.

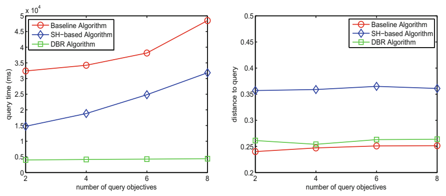
**Effect of $\alpha$.** We study the effect of weight factor $\alpha$ which varying from 0 to 1. As shown in Fig. 4, all algorithms including the baseline algorithm, the SH-based algorithm and the DBR algorithm have a marginal increase on query time when $\alpha$ grows. The DBR algorithm has the best time performance while the baseline algorithm always takes most of the time to complish search processing. The DBR algorithm and the baseline algorithm outperform the SH-based algorithm in accuracy. And the distances to query for these algorithms are smoothly increase with the grows of value of $\alpha$.

**Effect of $\beta$.** We investigate the performance of these algorithms when the threshold of weight factor $\beta$ for spatial distance is varying. Figure 5 shows the results of our experiment. With the increase of $\beta$, these algorithms have the same increasing trend on query time. On the other hand, the baseline algorithm and the DBR algorithm always outperform the SH-based algorithm in accuracy when $\beta$ varies from 0 to 1.

**Effect of $t$.** We proceed to examine the effect of number of latent topics by ploting query time and distance to query. As Fig. 6 shown, query time for these algorithms will increase as $t$ grows. Besides, the SH-based algorithm and the baseline algorithm need more query time than the DBR algorithm. The DBR algorithm achieves high accuracy compared with other algorithms and the distances to query for these algorithms are almost unaffected by the increase in the number of topics.

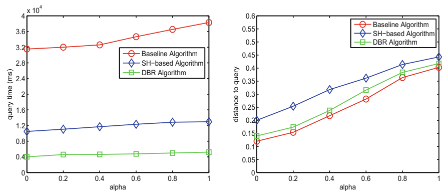
*(2) Evaluations of LSH parameters*

The performance of the SH-based Algorithm and the DBR Algorithm is mainly influenced by LIR-tree. LIR-tree can be measured by parameters hash tables $L$ and hash families $M$. We will tune these parameters to evaluate the performance of LIR-tree in sequence in this part.



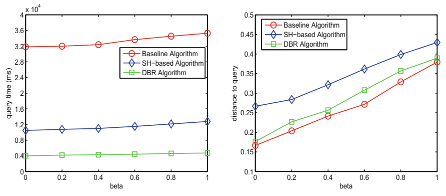
(a) Efficiency (b) Accurancy

**Fig. 3.** Effect of $|q.\Psi|$



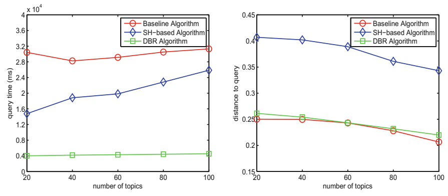
(a) Efficiency (b) Accurancy

**Fig. 4.** Effect of $\alpha$



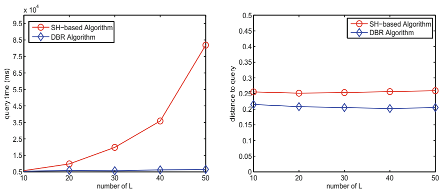
(a) Efficiency (b) Accurancy

**Fig. 5.** Effect of $\beta$



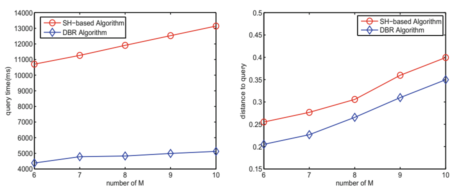
(a) Efficiency (b) Accurancy

**Fig. 6.** Effect of $t$



(a) Efficiency (b) Accurancy

**Fig. 7.** Effect of $L$



(a) Efficiency (b) Accurancy

**Fig. 8.** Effect of $M$

**Effect of $L$.** Parameter $L$ denotes the number of hash tables. Intuitively, a larger $L$ indidates more information provided by the LIR-tree, which facilitates the accuracy of object sets. To achieve higher quality with lower time, we vary $L$ from 10 to 50. As shown in Fig. 7, the DBR algorithm takes much less query time than the SH-based algorithm with the growth of the topics. This can be explained by the fact that the SH-based algorithm avoids computing for all candidate bucket set. Compared to the SH-based algorithm, the DBR algorithm always obtains result with less distance to query when $L$ varies from 10 to 50.

**Effect of $M$.** A group of experiments are conducted to evaluate the performance of LIR-tree under different $M$, which is a fundamental parameter for constructing a hash table. We vary $M$ from 6 to 10 with $L$ fixed as 30. Figure 8 indicates that the query time goes up as $M$ increases. And similar to Effect of $L$, the DBR algorithm transcends the SH-based algorithm in both efficiency and accuracy as we expected.

To sum up, compared the DBR algorithm with the SH-based algorithm and the baseline algorithm, the DBR algorithm can achieve a relatively high quality collective object set within short time in all settings.

## 7 Related Work

Spatial keyword query has been intensively studied in previous decades. Many contributions have already been made in the literature to support different types of spatio-textual querying. Some efforts are made to support the Spatial Keyword Boolean Query (SKBQ) [7,9,10,20,24] that requires exact keywords match, which may lead few or no results to be found. To overcome this problem, lots of work have been done to support the Spatial Keyword Approximate Query (SKAQ) [14,16,19,21–23], which ensures the query results are no longer sensitive to spelling errors and conventional spelling differences. Many novel indexing structures are proposed to support efficient processing on *SKBQ* and *SKAQ*, such as *IR-tree* [7], *IR*$^2$*-tree* [9], *MHR-tree* [19], *S2I* [16], etc. Numerous work studies the problem of spatial keyword query on *why-not questions* [18], *continuous querying* [1], *interactive querying* [23], *pub/sub system* [11,15], etc. Specifically, [20] addresses a more challenging problem on spatial keyword top-$k$ queries, where some known object is unexpectedly missing from a result; and [18] investigates a novel problem, namely, continuous top-$k$ spatial keyword queries on road networks; [13] eliminates the requirement of users to explicitly specify their preferences between spatial proximity and keyword relevance by enhancing the conventional queries with interaction; towards multi-objective query, [5] studies the $CSKQ$ problem to retrieve a group of objects that can collectively cover all keywords in query, while having the minimum distance cost.

But as far as we know, none of those existing approaches can retrieve spatial objects that are semantically relevant but morphologically different which are collectively cover all user-supplied keywords. The probabilistic topic models are statistical methods to analyze the words in documents and to discover the themes that run through them, how those themes are connected to each other, with no prior annotations or labeling of documents been required. Based on topic models, it is possible to measure the relevance of a testual description with regrad to a theme, as well as the relevance between different textual descriptions. The most classical topic models includes LDA [2], Dynamic Topic Model, etc. However, this method cannot find the object set that can satisfy all query objectives collectively and distributed in spatial rationally. Therefore, we investigate the topic model based collective spatial keyword querying to recommend users collective spatial objects that have both high spatial and semantic similarities to query.

## 8 Conclusion and Future Work

This paper committed to the problem of retrieving a group of spatial web objects more effectively and reasonably by converting keywords matching to topic distribution. The probabilistic topic model is utilized to interpret the textual descriptions attached to spatial objects and query objectives into topic distributions. To find the object set that can satisfy all query objectives collectively and distributed in spatial rationally, we propose an indexing structure to combine the spatial and semantic information effectively, as well as a searching algorithm to achieve efficient query processing. Extensive experimental results on real datasets demonstrate the efficiency of our proposed method. In the future, it will be interesting to consider the possible spatial dynamics of users and investigate the problem of continuous multi-objective spatial keyword querying with semantics.

**Acknowledgement.** This work was partially supported by Chinese NSFC project under grant numbers 61402312, 61232006, 61402313, 61572336, 61502324, 61572335, and Australia Research Council discovery projects under grant numbers DP140103499, DP160102412.

## References

1. Barbieri, D.F., Braga, D., Ceri, S., Valle, E.D., Grossniklaus, M.: C-SPARQL: SPARQL for continuous querying. In: WWW, pp. 1061–1062 (2009)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
3. Buhler, J.: Efficient large-scale sequence comparison by locality-sensitive hashing. Bioinformatics **17**(5), 419–428 (2001)

4. Cao, X., Cong, G., Jensen, C.S.: Retrieving top-k prestige-based relevant spatial web objects. PVLDB **3**(1), 373–384 (2010)
5. Cao, X., Cong, G., Jensen, C.S., Ooi, B.C.: Collective spatial keyword querying. In: SIGMOD, pp. 373–384 (2011)
6. Chen, Y.-Y., Suel, T., Markowetz, A.: Efficient query processing in geographic web search engines. In: SIGMOD Conference, pp. 277–288 (2006)
7. Cong, G., Jensen, C.S., Wu, D.: Efficient retrieval of the top-k most relevant spatial web objects. PVLDB **2**(1), 337–348 (2009)
8. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.S.: Locality-sensitive hashing scheme based on p-stable distributions. In: Symposium on Computational Geometry, pp. 253–262 (2004)
9. De Felipe, I., Hristidis, V., Rishe, N.: Keyword search on spatial databases. In: ICDE, pp. 656–665 (2008)
10. Jiang, H., Zhao, P., Sheng, V.S., Xu, J., Liu, A., Wu, J., Cui, Z.: An efficient location-aware top-k subscription matching for publish/subscribe with Boolean expressions. In: Navathe, S.B., Wu, W., Shekhar, S., Du, X., Wang, X.S., Xiong, H. (eds.) DASFAA 2016. LNCS, vol. 9643, pp. 335–350. Springer, Cham (2016). doi:10.1007/978-3-319-32049-6_21
11. Huiqi, H., Liu, Y., Li, G., Feng, J., Tan, K.-L.: A location-aware publish, subscribe framework for parameterized spatio-textual subscriptions. In: ICDE, pp. 711–722 (2015)
12. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: STOC, pp. 604–613 (1998)
13. Jin, J., Szekely, P.: Interactive querying of temporal data using a comic strip metaphor. In: IEEE VAST, pp. 163–170 (2010)
14. Li, F., Yao, B., Tang, M., Hadjieleftheriou, M.: Spatial approximate string search. IEEE Trans. Knowl. Data Eng. **25**(6), 1394–1409 (2013)
15. Li, G., Wang, Y., Wang, T., Feng, J.: Location-aware publish, subscribe. In: KDD, pp. 802–810 (2013)
16. Rocha-Junior, J.B., Gkorgkas, O., Jonassen, S., Nørvåg, K.: Efficient processing of top-k spatial keyword queries. In: Pfoser, D., Tao, Y., Mouratidis, K., Nascimento, M.A., Mokbel, M., Shekhar, S., Huang, Y. (eds.) SSTD 2011. LNCS, vol. 6849, pp. 205–222. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22922-0_13
17. Slaney, M., Casey, M.: Locality-sensitive hashing for finding nearest neighbors. IEEE Sig. Process. Mag. **25**(2), 128–131 (2008)
18. Tran, Q.T., Chan, C.-Y.: How to conquer why-not questions. In: SIGMOD, pp. 15–26 (2010)
19. Yao, B., Li, F., Hadjieleftheriou, M., Hou, K.: Approximate string search in spatial databases. In: ICDE, pp. 545–556 (2010)
20. Zhang, C., Zhang, Y., Zhang, W., Lin, X.: Inverted linear quadtree: efficient top k spatial keyword search. In: ICDE, pp. 901–912 (2013)
21. Zheng, B., Yuan, N.J., Zheng, K., Xie, X., Sadiq, S., Zhou, X.: Approximate keyword search in semantic trajectory database. In: ICDE 2015, pp. 975–986 (2015)
22. Zheng, K., Huang, Z., Zhou, A., Zhou, X.: Discovering the most influential sites over uncertain data: a rank-based approach. IEEE TKDE **24**(12), 2156–2169 (2012)
23. Zheng, K., Han, S., Zheng, B., Shang, S., Jiajie, X., Liu, J., Zhou, X.: Interactive top-k spatial keyword queries. In: ICDE 2015, pp. 423–434 (2015)
24. Ding, Z., Xu, J., Yang, Q.: SeaCloudDM: a database cluster framework for managing and querying massive heterogeneous sensor sampling data. J. Supercomput. **66**(3), 1260–1284 (2013)