Information Sciences 378 (2017) 194–214

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Level-aware collective spatial keyword queries☆

Pengfei Zhang[a,1], Huaizhong Lin[a,1,*], Bin Yao[b,1], Dongming Lu[a,1]

[a] College of Computer Science and Technology, Zhejiang University, Hangzhou, China
[b] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

ARTICLE INFO

Article history:
Received 15 May 2015
Revised 17 October 2016
Accepted 21 October 2016
Available online 21 October 2016

Keywords:
Collective spatial keyword query
Keyword level
Branch and bound strategy
Triggered update strategy

ABSTRACT

The collective spatial keyword query (CoSKQ), which takes a location and a set of keywords as arguments, finds a group of objects that collectively satisfy the query and achieve the smallest cost. However, few studies concern the *keyword level* (e.g., the level of hotels), which is of critical importance for decision support. Motivated by this, we study a novel query paradigm, namely *Level-aware Collective Spatial Keyword* (LCSK) query. The LCSK query asks for a group of objects that cover the query keywords collectively with a threshold constraint and minimize the cost function, which takes into account both the cost of objects and the spatial distance. In our settings, each keyword that appears in the textual description of objects is associated with a level for capturing the feature of keyword.

We prove the LCSK query is NP-hard, and devise exact algorithm as well as approximate algorithm with provable approximation bound to this problem. The proposed exact algorithm, namely MergeList, explores the candidate space progressively with several pruning strategies, which is based on the keyword hash table index structure. Unfortunately, this approach is not scalable to large datasets. We thus develop an approximate algorithm called MaxMargin. It finds the answer by traversing the proposed LIR-tree in the best-first fashion. Moreover, two optimizing strategies are used to improve the query performance. The experiments on real and synthetic datasets verify that the proposed approximate algorithm runs much faster than the competitor with desired accuracy.

1. Introduction

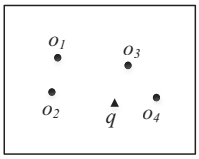
Spatial database has been studied for decades as it supports many applications from people's daily life to scientific research [2,3,10,23,26,34,42,47]. Recently, the keyword search has been combined with spatial queries to enhance location-based services such as Baidu Lvyou and Google Earth. Previous works on spatial keyword queries can be roughly classified into two categories based on the answer granularity: (1) some proposals find individual objects. Typically, given a location and a set of keywords as arguments, this type of query [14,15,17,39] retrieves individual objects that each can cover all query keywords, and (2) others ask for a group of objects. In a wide spectrum of applications, whereas multiple objects are required to satisfy the user's needs (expressed by keywords) collectively. Toward this goal, *m*CK [51,52], CoSKQ [7,35], BKC

☆ This manuscript is the authors' original work and has not been published nor has it been submitted simultaneously elsewhere.

* Corresponding author.
E-mail addresses: zpf_2013zb@zju.edu.cn (P. Zhang), linhz@zju.edu.cn (H. Lin), yaobin@cs.sjtu.edu.cn (B. Yao), ldm@zju.edu.cn (D. Lu).

[1] All authors have checked the manuscript and have agreed to the submission.

http://dx.doi.org/10.1016/j.ins.2016.10.033
0020-0255/



object	keywords	level vector	cost	distance
o_1	t_1, t_2, t_3	4, 5, 1	0.1	10
o_2	t_1, t_2, t_4	5, 4, 1	0.1	8
o_3	t_1, t_2, t_3, t_4	3, 2, 5, 5	0.15	6
o_4	t_1, t_2, t_3, t_5	2, 3, 5, 5	0.15	7

Fig. 1. Example of the LCSK query.

[18] and SGK [6] are investigated. To the best of our knowledge, few studies consider the *keyword level*. In real applications, we can use the keyword level to capture the level of tourist attractions, hotels or the rescue ability of equipments, which is increasingly important for users to make decisions.

In this work, we study a novel query paradigm called Level-aware Collective Spatial Keyword (LCSK) query. We enhance the collective spatial keyword queries (CoSKQ) from the following two aspects. First, we introduce the *level vector* for the objects in the database O. Each object $o \in O$ is associated with an integer level vector, denoted by $o.v$. The ith element of a level vector, i.e., $o.v^i$, represents the level of ith keyword in $o.\omega$, i.e., $o.\omega^i$. We denote by $o.\omega$ the associated keywords with o. Second, we introduce a normalized *weight vector* into the query definition for capturing the user-specified weights assigned to different levels. Similar to [12], we define our cost function, namely *cost distance*, as a combination of the spatial distance and the cost of objects (will be explained later). The LCSK query has numerous real applications such as resource scheduling and emergency rescue. Next, we present an example of the emergency rescue task.

Example 1. As illustrated in Fig. 1, we assume that the query point q is an earthquake point. There are four rescue teams o_1, o_2, o_3 and o_4 having necessary rescue equipments, namely $t_1, t_2, \ldots, t_5$. The *level vector* captures the level of equipments that are associated with teams, which can be used to measure the rescue ability. Usually, the higher the level, the higher the rescue ability. The *cost* denotes the overhead of performing the rescue by the corresponding team, and *distance* denotes the Euclidean distance between q and the rescue team. In such a scenario, we aim to find multiple teams that can together achieve the required rescue ability and have the smallest cost.

To address the above problem, we issue a query $q = (\ell, \omega, W, \theta)$, where ℓ denotes the location of q and $\omega = \{t_1, t_2\}$ captures the required rescue equipments. The normalized weight vector $W = (0.1, 0.15, 0.2, 0.25, 0.3)$ indicates the rescue ability of equipments with different levels. As an example, the level of t_1 w.r.t. o_1 is 4, and thus the corresponding rescue ability is $W[4] = 0.25$. Then, $\theta = 0.5$ denotes the desired rescue ability for each required equipment. That is, a group of teams whose rescue abilities are not less than 0.5 for each required equipment are called for. In this case, we deliver the group $\{o_1, o_2\}$ as the answer, because it offers the desired rescue ability for each required equipment and has the smallest cost distance. Specifically, the rescue ability of t_1 and t_2 contributed by $\{o_1, o_2\}$ are $W[4] + W[5] = 0.55$ and $W[5] + W[4] = 0.55$, which are larger than the given threshold 0.5. Moreover, the cost distance of $\{o_1, o_2\}$ is $0.1 * 10 + 0.1 * 8 = 1.8$.

More formally, given a spatial database O, and an LCSK query $q = (\ell, \omega, W, \theta)$, where ℓ is the query location and ω is the set of query keywords. W is a normalized weight vector and θ is a threshold. The LCSK query is to retrieve a group G of objects such that satisfy the following two conditions:

- $\forall t \in q.\omega$, the coverage weight of t by G is not less than $q.\theta$;
- G has the smallest cost distance among those groups that meet the above condition.

We prove the LCSK query is NP-hard by the reduction from the weighted set cover (WSC) problem. We devise both the exact algorithm and approximate algorithm to this problem. The proposed exact algorithm, namely MergeList, performs the query by searching the candidate space progressively, which is based on the keyword hash table index structure. Specifically, the candidate space contains all promising answers, and we use several strategies to prune the candidate space. Though equipped with several pruning strategies, MergeList is not scalable to large datasets due to the complexity of our problem. We thus develop an approximate algorithm called MaxMargin with provable approximation bound. MaxMargin finds the answer by traversing the LIR-tree in the best-first fashion. In particular, the LIR-tree augments each inverted file of IR-tree with additional information, i.e., the level of keywords and the cost of objects. Two effective optimizing strategies, namely the *branch and bound strategy* (BBS) and the *triggered update strategy* (TUS) are proposed to further improve the performance of MaxMargin.

To summarize, we make the following contributions:

- We formally define the LCSK query, which is to find a group of objects such that collectively cover the query keywords with a threshold constraint and have the smallest cost distance. We then theoretically prove this problem is NP-hard.
- We develop an exact algorithm called MergeList on the top of keyword hash table index structure. Furthermore, we propose the LIR-tree, which is extended from IR-tree. Based on the LIR-tree, we devise an approximate algorithm called MaxMargin that finds the answer using the best-first strategy. To further facilitate the query processing, two optimizing strategies, namely BBS and TUS, are adopted by MaxMargin.
- We conduct comprehensive experiments on real and synthetic datasets to verify the performance of our proposed algorithms.