On Nearby-Fit Spatial Keyword Queries (Extended Abstract)

Victor Junqiu Wei[#], Raymond Chi-Wing Wong[*], Cheng Long[†], Pan Hui[*‡]
[#]*Noah's Ark Lab, Huawei Technologies*, [*]*The Hong Kong University of Science and Technology*
[†]*Nanyang Technological University*, [‡]*University of Helsinki*
[#]weijunqiu@huawei.com, [*]{raywong,panhui}@cse.ust.hk, [†]c.long@ntu.edu.sg, [‡]panhui@cs.helsinki.fi

Abstract—**Geo-textual data is ubiquitous nowadays, where each object has a location and is associated with some keywords. Many types of queries based on geo-textual data, termed as *spatial keyword queries*, have been proposed, and are to find optimal object(s) in terms of both its (their) location(s) and keywords. In this paper, we propose a new type of query called *nearby-fit spatial keyword query* (NSKQ), where an optimal object is defined based not only on the location and the keywords of the object itself, but also on *those of the objects nearby*. For example, in an application of finding a hotel, not only the location of a hotel but also the objects near the hotel (e.g., shopping malls, restaurants and bus stops nearby) might need to be taken into consideration. The query is proved to be NP-hard, and in order to perform the query efficiently, we developed two approximate algorithms with small constant approximation factors equal to 1.155 and 1.79. We conducted extensive experiments based on both real and synthetic datasets, which verified our algorithms.**

Index Terms—**spatial keyword queries, proximity queries, spatial database, location-based services**

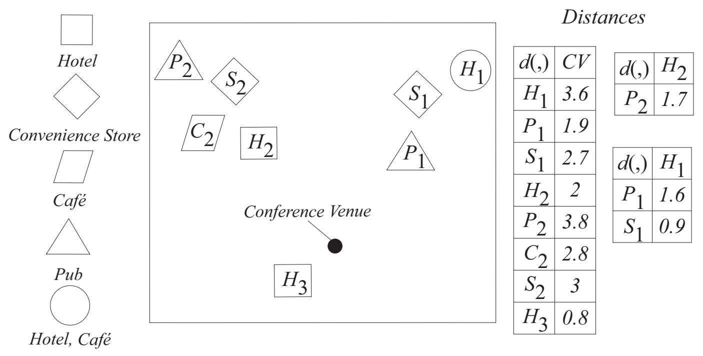
1. Introduction

Nowadays, geo-textual data which refers to data with both spatial and textual information is ubiquitous (e.g., webpages and photos at Flicker). The geo-textual data could be represented by a set of spatial objects each of which is associated with a set of (textual) keywords. Consider the following problem based on the geo-textual data presented in Figure 1.

`A hotel-finding problem: Tom is going to attend an international conference and would like to find a hotel such that (1) the location of the hotel is near to the conference venue (marked by the black dot in the figure) and (2) there are at least a convenience store, a café and a pub nearby the hotel.`

It is easy to see that H_2 is the best selection for Tom since it is surrounded by a pub (P_2), a café (C_2) and a convenience store (S_2) and is in the vicinity of the conference venue. Besides, H_1 and H_3 are bad choices since there are nothing in the vicinity of H_3 and H_1 is far away from the conference venue.

Although many different types of queries based on geo-textual data which are termed as *spatial keyword queries* have been proposed in the literature, none of them could be used to capture the above hotel-finding problem well. The relevant existing studies fall into the following 3 categories. The first category is the *spatial keyword kNN query* (SK*k*NNQ) [1] which ranks each POI by its own location and keywords and fails to consider its vicinity. Thus, it is not applicable to the aforementioned problem. The second and the third category are the *collective spatial keyword query* (CoSKQ) [2]–[4] and *m-Closest Keyword query* (*m*CK) [5], respectively. They aim to find a group of objects collectively covering all query keywords with their cost minimized, where the cost involves the distances among the POIs and the query location. But, they rank each group of objects collectively rather than each target object (in the above example, a hotel). Thus, the target object found in the returned group could be a bad one although the returned group has the lowest cost (see the detailed explanation in [6] for better understanding).



$d(,)$	CV
H_1	3.6
P_1	1.9
S_1	2.7
H_2	2
P_2	3.8
C_2	2.8
S_2	3
H_3	0.8

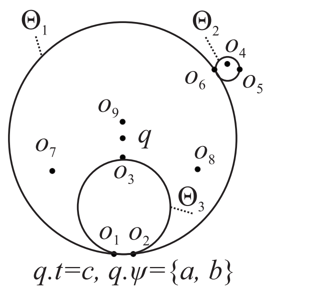
$d(,)$	H_2
P_2	1.7

$d(,)$	H_1
P_1	1.6
S_1	0.9

Fig. 1: A motivating example (CV stands for "Conference Venue")

Motivated by this, in this paper, we formalize the problems like the hotel-finding problem as a query called n̲earby-fit s̲patial k̲eyword q̲uery (NSKQ) as follows.

Problem statement. Let D be a set of spatial objects each of which is associated with a set of keywords. Given a query q which consists of a location $q.\lambda$ called the *query location*, a keyword $q.t$ called the *target keyword*, and a set $q.\psi$ of keywords called the *nearby keywords*, the **n̲earby-fit s̲patial k̲eyword q̲uery** (NSKQ) is to find the object in D which has the *smallest* cost wrt q, denoted by $Cost(o|q)$, among all objects containing the target keyword. Here, $Cost(o|q)$ is defined to be a linear combination of two



$q.t$=c, $q.\psi$=$\{a, b\}$

Fig. 2: An example of NSKQ

Obj	ψ
o_1	a
o_2	b
o_3	c
o_4	a
o_5	b,c
o_6	b,c
o_7	a,c
o_8	b,c
o_9	c

Fig. 3: Keyword set of each object

components: (1) the distance from the query location to o and (2) the *smallest* distance from a set of objects covering the set of nearby keywords to o i.e., $Cost(o|q) = \alpha \cdot d(q,o) + (1-\alpha) \cdot \min\{Dist(o,O)|O \subseteq D, O$ covers $q.\psi\}$, where $\alpha \in [0,1]$ is a tuning parameter and $Dist(o,O)$ denotes the maximum pairwise distance of the set $O \cup \{o\}$, i.e., $Dist(o,O) = \max_{o_1,o_2 \in O \cup \{o\}} d(o_1,o_2)$.

Besides, in [6], we prove that the NSKQ problem is NP-hard, and to solve it efficiently, we developed two approximate algorithms, both with small constant approximation factors and empirically verified the efficiency and the effectiveness of the approximate algorithms.

2. Processing NSKQ

We proposed two approximate algorithms for NSKQ, namely *Appro1* and *Appro2*. In *Appro1*, we adopted a concept called "minimal covering disk" which is a disk *compactly* covering $q.\psi$ and $q.t$. Given a disk Θ, Θ is said to be a *covering disk* if there exists a set O of objects in Θ such that O covers $q.\psi$ and $q.t$. Given a covering disk Θ, Θ is said to be a *minimal covering disk* if there does not exist a smaller disk Θ' such that Θ' is a covering disk and Θ' does not contain any object outside Θ.

For example, Figure 2 shows 9 objects, namely $o_1, o_2, ..., o_9$, together with their keywords shown in Figure 3. In Figure 2, we also show three disks, namely Θ_1, Θ_2 and Θ_3, where Θ_1 centers at q and its radius is $d(q, o_6)$, Θ_2 (Θ_3) is a disk whose boundary contains o_5 and o_6 (o_1, o_2 and o_3, resp.). Consider that $q.t = c$ and $q.\psi = \{a, b\}$. Θ_1, Θ_2 and Θ_3 are covering disks. Neither Θ_1 nor Θ_2 is a minimal covering disk (since there exists a smaller covering disk of Θ_1 and one smaller covering disk of Θ_2) but Θ_3 is a minimal covering disk.

Based on this concept, we develop the *Appro1* algorithm. Roughly speaking, instead of enumerating all possible combinations of objects covering $q.\psi$ and $q.t$, *Appro1* (1) first enumerates all possible *candidates* of minimal covering disks, (2) for each enumerated disk, finds within the disk the nearest object o'_q containing the target keyword from query q and a set O' of objects covering all nearby keywords, and computes the cost value $Cost(o'_q|q, O')$ which is an estimated value of $Cost(o'_q|q)$, and (3) returns the object o'_q with the smallest estimated cost. It is shown in [6] that the approximate ratio of *Appro1* is 1.155 and its time complexity is $O(|q.\psi|^3 \cdot |D| \log |D|)$.

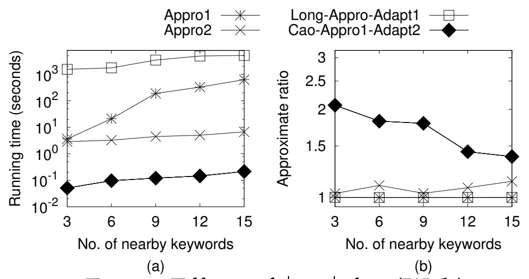


Fig. 4: Effect of $|q.\psi|$ for (Web)

Besides, we propose *Appro2* which is faster than *Appro1* but has a slightly larger approximation factor. *Appro2* uses two new concepts, namely "center-centric nearest neighbor set" and "axis-parallel square", which are "looser" concepts compared with "minimum covering disk". Since finding "center-centric nearest neighbor set" and "axis-parallel square" are computationally cheaper than finding "minimum covering disk" [6], *Appro2* is faster than *Appro1*. The time complexity of *Appro2* is $O(|q.\psi||D| \log |D| + |D| \log^2 |D|)$. Besides, the approximate factor of *Appro2* is 1.79, a little bit larger than the approximate factor of *Appro1*.

3. Experiment

We conducted experiments on a 2.2GHz machine with 4GB RAM. We compared our proposed algorithms, namely *Appro1* and *Appro2*, with several baselines adapted from existing algorithms as shown in [6] and present only two baselines, namely *Long-Appro-Adapt1* and *Cao-Appro1-Adapt2*, since each of the other baselines is dominated by one of the four algorithms considered in the paper. All algorithms were implemented in C++. We use the same datasets as in [2], namely Web, GN and Hotel. Figure 4 shows the results on the Web dataset (which contains 1,777,598 objects). Each considered baseline either has an unacceptable running time (up to more than 2 orders of magnitude than ours) or has an unacceptable approximate ratio (more than 2). Remarkably, the approximate ratio of *Appro1* is nearly 1, which means that its accuracy is significantly high in practice.

4. Acknowledgments

We are grateful to the anonymous reviewers for their constructive comments on this paper. The research of Victor Junqiu Wei and Raymond ChiWing Wong is supported by HKRGC GRF 16219816. The research of Cheng Long was supported by the NTU Start-Up Grant and Singapore MOE Tier 1 Grant RG20/19 (S).

References

[1] I. D. Felipe, V. Hristidis, and N. Rishe, "Keyword search on spatial databases," in *ICDE*, 2008.

[2] X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi, "Collective spatial keyword querying," in *SIGMOD*, 2011.

[3] C. Long, R. C.-W. Wong, K. Wang, and A. W.-C. Fu, "Collective spatial keyword queries:a distance owner-driven approach," in *SIGMOD*, 2013.

[4] X. Cao, G. Cong, T. Guo, C. S. Jensen, and B. C. Ooi, "Efficient processing of spatial group keyword queries," *TODS*, 2015.

[5] T. Guo, X. Cao, and G. Cong, "Efficient algorithms for answering the m-closest keywords query," in *SIGMOD*, 2015.

[6] J. Wei, R. C.-W. Wong, C. Long, and P. Hui, "On nearby-fit spatial keyword queries," *TKDE*, 2019.