ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

An Efficient Approach to Optimize the Keyword Cover Search

Ms. Supriya Kumar More
M.Tech student
Marri Laxman Reddy Institute of Technology & Management
Hyderabad.

Mr. K. Siva Ram Prasad
Assistant Professor
Marri Laxman Reddy Institute of Technology & Management
Hyderabad.

Abstract

Spatial database stores the information about the spatial objects which are associated with the keywords to indicate the information such as its business/services/features. None of the individual objects is associated with all query keywords, this motivates studies to retrieve multiple objects, called keyword cover, which together cover all query keywords and are close to each other. In m closest keyword search, it covers a set of query keywords and minimum distance between objects. From last few years, keyword rating increases its availability and importance in object evaluation for the decision making. This is the main reason for developing the new algorithm called best keyword cover which is consider inter- distance as well as the keyword rating provided by the customers through the online business. m closest keyword search algorithm combines the objects from different query keywords to generate candidate keyword covers. Baseline algorithm and keyword nearest neighbor expansion algorithms are used to find the best keyword cover. The performance of the m closest keyword algorithm drops dramatically, when the number of query keyword increases. This work proposes to solve generic version problem of the existing algorithm called keyword nearest neighbor expansion which reduces the resulted candidate keyword covers.

Keywords: ***Spatial database, point of interests, keywords, keyword rating, keyword cover***

INTRODUCTION

An increasing number of applications require the efficient execution of nearest neighbor (NN) queries constrained by the properties of the spatial objects. Due to the popularity of keyword search, particularly on the Internet, many of these applications allow the user to provide a list of keywords that the spatial objects (henceforth referred to simply as objects) should contain, in their description or other attribute [1, 2]. For example, online yellow pages allow users to specify an address and a set of keywords and produce results which have description to these keywords, ordered by their distance to the specified address location. As another example, real estate web sites allow users to search for properties with specific keywords in their description and rank them according to their distance from a specified location. We call such queries spatial keyword queries. A spatial keyword query consists of a query area and a set of keywords. The answer is a list of objects ranked according to a combination of their distance to the query area and the relevance of their text description to the query keywords. A simple popular variant, which is used in our running example, is the distance-first spatial keyword query, where objects are ranked by distance and keywords are applied as a conjunctive filter to eliminate objects that do not contain them. Which is our running example, displays a dataset of fictitious hotels with their spatial coordinates and a set of descriptive attributes (name, amenities)? An example of a spatial keyword query is “find the nearest

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

hotels to point that contain keywords internet and pool". The top result of this query is the hotel object. Unfortunately there is no efficient support for top-k spatial keyword queries, where a prefix of the results list is required. Instead, current systems use ad-hoc combinations of nearest neighbor (NN) and keyword search techniques to tackle the problem. For instance, an R- Tree is used to find the nearest neighbors and for each neighbor an inverted index is used to check if the query keywords are contained. We show that such two-phase approaches are inefficient [3–5].

EXISTING SYSTEM:

- Some existing works focus on retrieving individual objects by specifying a query consisting of a query location and a set of query keywords (or known as document in some context). Each retrieved object is associated with keywords relevant to the query keywords and is close to the query location.
- The approaches proposed by Cong et al. and Li et al. employ a hybrid index that augments nodes in non-leaf nodes of an R/R*-tree with inverted indexes.
- In virtual bR*-tree based method, an R*-tree is used to index locations of objects and an inverted index is used to label the leaf nodes in the R*-tree associated with each keyword. Since only leaf nodes have keyword information the mCK query is processed by browsing index bottom-up.

DISADVANTAGES OF EXISTING SYSTEM:

- When the number of query keywords increases, the performance drops dramatically as a result of massive candidate keyword covers generated.
- The inverted index at each node refers to a pseudo-document that represents the keywords under the node. Therefore, in order to verify if a node is relevant to a set of query keywords, the inverted index is accessed at each node to evaluate the matching between the query keywords and the pseudo-document associated with the node.

PROPOSED SYSTEM:

- This paper investigates a generic version of mCK query, called Best Keyword Cover (BKC) query, which considers inter-objects distance as well as keyword rating. It is motivated by the observation of increasing availability and importance of keyword rating in decision making. Millions of businesses/services/features around the world have been rated by customers through online business review sites such as Yelp, Citysearch, ZAGAT and Dianping, etc.
- This work develops two BKC query processing algorithms, baseline and keyword-NNE. The baseline algorithm is inspired by the mCK query processing methods. Both the baseline algorithm and keyword-NNE algorithm are supported by indexing the objects with an R*-tree like index, called KRR*-tree.
- We developed much scalable keyword nearest neighbor expansion (keyword-NNE) algorithm which applies a different strategy. Keyword-NNE selects one query keyword as principal query keyword. The objects associated with the principal query keyword are principal objects. For each principal object, the local best solution (known as local best keyword cover lbkc) is computed. Among them, the lbkc with the highest evaluation is the solution of BKC query. Given a principal object, its lbkc can be identified by simply retrieving a few nearby and highly rated objects in each non-principal query keyword (two-four objects in average as illustrated in experiments).

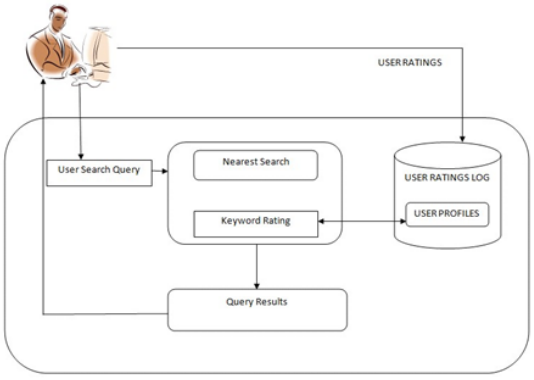
ADVANTAGES OF PROPOSED SYSTEM:

- Compared to the baseline algorithm, the number of candidate keyword covers generated in keyword-NNE algorithm is significantly reduced. The in-depth analysis reveals that the number of candidate keyword covers further processed in keyword-NNE algorithm is optimal, and each keyword candidate cover processing generates much less new candidate keyword covers than that in the baseline algorithm.
- The proposed keyword-NNE algorithm applies a different processing strategy, i.e., searching local best solution for each object in a certain query keyword. As a consequence, the number of candidate keyword covers generated is significantly reduced.
- The analysis reveals that the number of candidate keyword covers which need to be further processed

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

inkeyword-NNE algorithm is optimal and processing each keyword candidate cover typically generates much less new candidate keyword covers in keyword-NNE algorithm than in the baseline algorithm.

SYSTEM ARCHITECTURE:



PROPOSED ALGORITHM

Baseline Algorithm

Baseline algorithm is not feasible in practice. The main reason is that baseline algorithm requires maintaining H in memory. The peak size of H can be very large because of the exhaustive combination until the first current best solution best keyword cover (bkc) is obtained. To release the memory bottleneck, the depth-first browsing strategy is applied in the baseline algorithm such that the current best solution is obtained as soon as possible. Compared to the best-first browsing strategy which is global optimal, the depth-first browsing strategy is a kind of greedy algorithm which is local optimal. As a consequence, if a candidate keyword cover (kc) has kc.score > bkc.score, kc is further processed by retrieving the child nodes of kc and combining them to generate more candidates. Note that bkc.score increases from 0 to BKC.score in the baseline algorithm. Therefore, the candidate keyword covers which are further processed in the baseline algorithm can be much more than that in baseline algorithm. Given a candidate keyword cover kc, it is further processed in the same way in both the baseline algorithm and baseline algorithm, i.e., retrieving the child nodes of kc and combines them to generate more candidates using Generate Candidate function in Algorithm. Since the candidate keyword covers further processed in the baseline algorithm can be much more than that in baseline algorithm, the total candidate keyword covers generated in the baseline algorithm can be much more than that in baseline algorithm. Note that the analysis captures the key characters of the baseline algorithm in BKC query processing which are inherited from the methods for mCK query processing.

Keyword-NNE Algorithm

In keyword-NNE algorithm, the best-first browsing strategy is applied like baseline but large memory requirement is avoided. For the better explanation, we can imagine all candidate keyword covers generated in baseline algorithm are grouped into independent groups. Each group is associated with one principal node (or object). That is, the candidate keyword covers fall in the same group if they have the same principal node (or object). Given a principal node Nk, let GNk be the associated group. The example in Figure shows GNk where some keyword covers such as kc1, kc2 have score greater than BKC.score, denoted as G1Nk, and some keyword covers such as kc3, kc4 have score not greater than BKC.score, denoted as G2Nk. In baseline algorithm, GNk is maintained in H before the first current best solution is obtained and every keyword cover in G1Nk needs to be further processed. In keyword-NNE algorithm, the keyword cover in GNk with the highest score, i.e., lbkcNk, is identified and maintained in memory. That is, each principal node (or object) keeps its lbkc only.

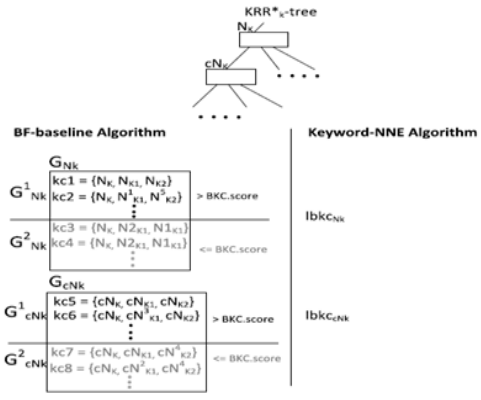


Fig. 1: Baseline vs. Keyword-NNE.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

CONCLUSION

The proposed system provides more sensible decision making than the mCK query. Baseline algorithm is inspired by the mCK query. The main problem of baseline algorithm is that, it reduces the performance when number of query keyword increases. Keyword-NNE algorithm applies a different strategy that searches the best solution in query keyword for each object. It reduces the generated candidate keyword covers. Baseline keyword covers are passed to keyword-NNE algorithm for further processing which is optimal and generates less new candidate keyword covers than the baseline algorithm.

REFERENCES

1.Ke Deng, Xin Li, Jiaheng Lu et al. Best keyword cover search. IEEE Transactions on Knowledge and Data Engineering. 2015; 27(1).

2.X. Cao, G. Cong, C. Jensen. Retrieving top-k prestige-based relevant spatial web objects. Proc. VLDB Endowment. 2010; 3(1/2): 373–384p.

3.X. Cao, G. Cong, C. Jensen et al. Collective spatial keyword querying. In Proc. ACM SIGMOD Int. Conf. Manage. Data. 2011; 373–384p.

4.G. Cong, C. Jensen, D. Wu. Efficient retrieval of the top-k most relevant spatial web objects. Proc. VLDB Endowment. 2009; 2(1): 337–348p.

5.I. D. Felipe, V. Hristidis, N. Rishe. Keyword search on spatial databases. In Proc. IEEE 24th Int. Conf. Data Eng. 2008; 656–665p.

6.R. Hariharan, B. Hore, C. Li et al. Processing spatial keyword (SK) queries in geographic information retrieval (GIR) systems. In Proc. 19th Int. Conf. Sci. Statist. Database Manage. 2007; 16–23p.

7.Z. Li, K. C. Lee, B. Zheng et al. IRTree: An efficient index for geographic document search. IEEE Trans. Knowl. Data Eng. 2010; 99(4): 585–599p.

8.J. Rocha-Junior, O. Gkorgkas, S. Jonassen et al. Efficient processing of top-k spatial keyword queries. In Proc.12th Int. Conf. Adv. Spatial Temporal Databases. 2011; 205–222p.

9.S. B. Roy, K. Chakrabarti. Location- aware type ahead search on spatial databases: Semantics and efficiency. In Proc. ACM SIGMOD Int. Conf. Manage. Data. 2011; 361–372p.

10.D. Zhang, Y. Chee, A. Mondal et al. Keyword search in spatial databases: Towards searching by document. In Proc. IEEE Int. Conf. Data Eng. 2009; 688–699p