Efficient Batch Top-k Spatial Keyword Search (BTOPK-SK) Using R* Tree

Anita Sofia Liz.D.R
Department of Computer Science and Engineering
New Prince Shri Bhavani College of Engineering and Technology
Chennai-73,TamilNadu,India
dranitasofializ@gmail.com

R.M.Beula and K.Saranya
Department of Computer Science and Engineering
New Prince Shri Bhavani College of Engineering and Technology
Chennai-73,TamilNadu,India
rmbeula@gmail.com and sana17saran@gmail.com

Abstract*--With advances in geographical-position technologies and geographical-location services, there are a quickly growing amount of spatiotextual objects collected in many applications such as position based services and communal networks, in which an object is described by its spatial location and a set of keywords (terms). Consequently, the study of spatial keyword search which explores both position and word description of the objects has attracted great attention from the commercial organizations and research communities.A novel index structure R* Tree is implemented by replacing Inverted Linear QuadTree (IL-Quadtree). To enhance the effectiveness we are using collaborative filter and Point Of Interest (POI) based process. An efficient algorithm is then developed to handle top k spatial keyword search. From the existing techniques, Top K Spatial Keyword Search (TOPK-SK) and Batch Top K Spatial Keyword Search (BTOPK-SK) are used for spatial keyword related queries.

Index Terms – R* Tree, Collaborative filter, POI, TOPK-SK, BTOPK-SK

I. INTRODUCTION

With the increasing pervasiveness of the geographical-positioning technologies and geographical-location services, there are an enormous amount of spatio-textual objects available in many applications. For example, in the local search service, yellow pages provide the location information as well as short descriptions of the businesses (e.g., hotels, restaurants). In the Global Position System (GPS) navigation system, POI (point of interest) is a geographically attaching pushpin that someone may find useful or interesting, which is usually annotated with texture information. Moreover, in many communal network services (e.g., Facebook, Flickr), a huge number of geographically-tagged photographs are accumulated everyday, which can be geographically-tagged by users, GPS enabled smartphones or cameras with a built-in GPS receiver.These uploaded photographs are usually associated with multiple text labels. As a result, in recent years various spatial keyword query prototype and techniques have appeared such that users can effectively exploit both spatial and word information of these spatio textual objects. In the paper, we investigate the problem of conducting top k spatial keyword search (TOPK-SK); that is, given a set of spatio-textual objects, a query location q and a set of keywords, we aim to retrieve the k closest objects each of which contains all keywords in the query. The top k spatial keyword search is elemental in spatial keyword queries and has a wide scoop of applications. Below are two motivating examples.

In suppose there are a set of businesses whose locations (represented by squares) and service lists (a set of keywords) are registered in the online yellow pages of a local search service provider. When a GPS-enabled smartphone user wants to find a nearby restaurant to have a piece of donut and a cup of tea, she may send the local search server two keywords, tea and donut.

In many real applications, the query workload may vary from time to time, and the system may encounter a burst of queries (e.g., queries invoked by a particular event). In this scenario, the system throughout is poor if a large number of queries are processed one by one. Motivated by this, a large body of existing work have been devoted to investigate how to improve the system throughout with the batch query processing techniques such that a large number of queries in the queue can be processed with a reasonable delay. Meanwhile, nowadays, a large volume of spatial keyword queries may be generated in a short time period. For instance, a large number of queries may be imposed to seek nearby restaurants during the dinner and lunch time. An important local event may incur a large number of relevant queries. As shown in, a large number of fake spatial keyword searches may be issued in order to protect the users privacy. This may lead to dramatic degrade of the system throughout if queries are processed individually. To alleviate this issue, we also investigate the problem of batch spatial keyword query (BTOPK-SK) which aims to efficiently support a large number of spatial keyword queries at the same time.

II. Related Work

[1]In this paper, With advances in geographical-positioning technologies and geographical-location services, there are a quickly growing amount of spatiotextual objects collected in many applications such as position based services and communal networks, in which an object is described by its spatial location and a set of terms. Consequently, the study of spatial keyword search which explores both location and word description of the objects has attracted great attention from the commercial organizations and research communities. In the paper, we study two elemental issues in the spatial keyword queries: top k spatial keyword search (TOPK-SK), and batch top k spatial keyword search (BTOPK-SK).

[2]The problems of nearest neighbor search on spatial data and keyword search on text data have been extensively studied separately. However to the best of our knowledge there is no systematic method to answer spatial keyword queries that is queries that specify both a position and a set of terms. In this work we present an efficient method to answer top-k spatial keyword queries. To do so we introduce an indexing structure called Information Retrieval R-Tree which combines an R-Tree with lay over text signatures. We present algorithms that construct and maintain an Information Retrieval R-Tree and use it to solve top-k spatial keyword queries. Our algorithms are innovatively compared to current methods and are shown to have greater performance and excellent scalability.

[3]Web users and content are increasingly being geopositioned, and increased center of attraction by being given to serving local content in response to web queries. Such joint processing is attractive during high query loads and also occurs when multiple queries are used to confuse a user's true query. We propose a algorithm and index structure for the joint processing of top-k spatial keyword queries. Factual studies show that the proposed solution is efficient on real data sets. We also offer logical studies on synthetic data sets to demonstrate the efficiency of the proposed solution.

[4]The information entities of disaster response, national infrastructure protection, crime analysis, and numerous other databases have both spatial and word descriptions. For example Geographical Information Retrieval system that we address in this paper is a search engine built on top of hundreds of thousands of publicly available Geographic Information System databases. Building a search engine over such large repositories is a challenge. We propose a framework for Geographical Information Retrieval systems and focus on indexing strategies that can process SK queries efficiently. We show through investigation that our indexing strategies lead to significant improvement in efficiency of answering Spatial Keyword queries over existing techniques.

[5]In this paper, we propose an efficient index, called Intermediate Representation-tree, which together with a top-k document search algorithm facilitates four major tasks in document searches, namely, spatial filtering, textual filtering, relevance computation, and document ranking in a fully integrated manner. In addition, Intermediate Representation-tree allows searches to embrace different weights on word and spatial relevance of documents at the runtime and thus caters for a broad diversity of applications. A set of comprehensive experiments over a wide range of scenarios has been conducted and the experiment results demonstrate that Intermediate Representation-tree outperforms the state-of-the-art approaches for geographic document searches.

[6]Location-based services have been widely accepted by mobile users. Many Location-based services users have direction-aware search needed that answers must be in the search direction. We propose a direction-conscious spatial keyword search method which inherently supports direction-aware search. We formulate novel direction-aware indexing structures to prune needless directions. We develop effective pruning techniques and search algorithms to efficiently answer a direction-conscious query. As users may dynamically change their search directions, we propose to incrementally answer a query. Exploratory results on real datasets show that our method attains high performance and outperforms existing methods significantly.

III. Existing

The Existing Techniques for the problem of TOPK-SK query as well as some other variants of top k spatial keyword search. Then other spatial keyword related queries are introduced. Considering the indexing scheme used in existing works, we classify the indexes into two categories, namely Keyword First Index and Spatial First Index. We describe the shortcomings of the existing indexing approaches. The system throughout is poor if a large number of queries are processed one by one. Motivated by this, a large body of existing work has been devoted to investigate how to improve the system throughout with the batch query processing techniques such that a large number of queries in the queue can be processed with a reasonable delay.

A. IL Quadtree

A quadtree is a space partitioning tree data structure in which a d-dimensional space is recursively subdivided into 2d regions. Due to its simplicity and regularity, the quadtree technique has been widely applied in many applications. As an efficient implementation of the disk-based quadtree, the linear quadtree is proposed to keep the non-empty leaf node of the quadtree in an auxiliary disk-based one dimensional structure (e.g., B+ tree), where each node can be encoded by the space filling curve techniques

B. Btopk-sk

The system throughout may be deteriorated if we separately process each individual query in the BTOPK-SK, especially when the volume of the queries is large. Therefore, it is desirable to partition the queries into groups so that the queries in the same group

may share computation and hence significantly reduce the overall costs. Moreover, novel query processing techniques are necessary to cope with groups of queries based on the IL-Quadtree proposed in this paper.
Query partition plays an important role in batch query processing because it can significantly reduce the processing time by grouping similar queries so that the CPU and I/O costs can be shared between queries in the same group. This strategy has been widely adopted in many research papers such as [4]. However, this method only considers spatial information, and not explicitly considers textual composition. Thus, we take both spatial feature and textual information into consideration to further improve the effectiveness of query partition.

C. TOPK-SK

We introduce an efficient TOPK-SK query algorithm assuming that objects are organized by an ILQuadtrees. Same as other inverted index based approaches, we also conduct incremental nearest neighbor search on the IL-Quadtrees. The main difference is that we can make use of the space partition based signatures (i.e., quadtree structures) to eliminate non-promising objects

IV. PROPOSED ARCHITECTURE

We propose a completely unique index structure, specifically R * tree, to prepare the spatio-textual objects. Associate degree economical algorithmic program is developed to support the highest k abstraction keyword search by taking advantage of the R * tree. We have a tendency to any propose a partition primarily based technique to reinforce the effectiveness of the signature of R * tree. To facilitate an oversized quantity of abstraction keyword queries, we have a tendency to propose a BTOPK-SK algorithmic program moreover as a question cluster algorithmic program to reinforce the performance of the system.

A. R*TREE

R * tree is a space partitioning tree data structure in which a d-dimensional space is recursively subdivided into 2d regions. Due to its simplicity and regularity, the R * tree technique has been widely applied in many applications. As an efficient implementation of the disk-based R * tree, the linear R * tree is proposed to keep the non-empty leaf node of the R * tree in an auxiliary disk-based one dimensional structure (e.g., B+ tree), where each node can be encoded by the space filling curve techniques

B. EFFECTIVE QUERY PARTITION

Query partition plays an important role in batch query processing because it can significantly reduce the processing time by grouping similar queries so that the CPU and I/O costs can be shared between queries in the same group. This strategy has been widely adopted in many research papers such as. However, this method only considers spatial information, and not explicitly considers textual composition.

C. EFFICIENT BATCH QUERY PROCESSING

After dividing the batch query into a set of groups, it is essential to devise new query processing techniques based on R * tree structure proposed in this paper. The key is the computation sharing strategies. We carefully maintain the objects loaded during the query processing so that the location information of these objects can be used to facilitate the computation.

D. USER MODULE

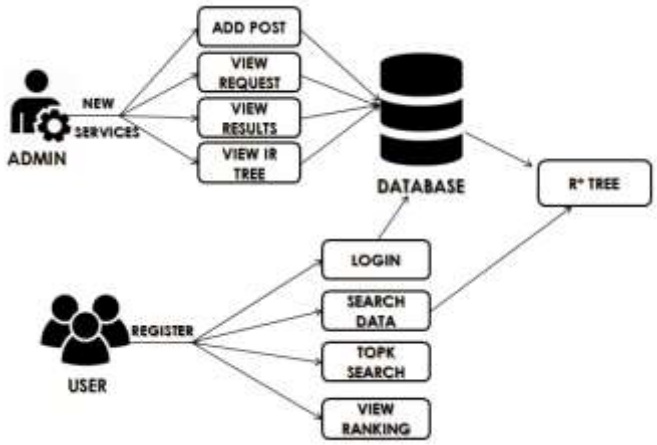
All users are travel on the underlying network and they also hold privacy policies which specify the privacy requirements of each user. In this module, Users are having authentication and security to access the detail which is presented in the authentication system. Before accessing or searching the details user should have the account in that otherwise they should register first.

E. COLLABORATING FILTERS

The increasing use of the Internet has made it much more difficult to effectively detect useful information from all the available online information. The enormous amount of data necessitates mechanisms for efficient information filtering. Collaborative filtering algorithms often require users' active participation, an easy way to represent users' interests, and algorithms that are able to match people with similar interests.

Typically, the advancement of a collaborative filtering system is:A user expresses the priority by rating items of the system. These ratings can be viewed as an approximate representation of the user's liking in the related domain. The system matches the user's ratings unique from other users' and finds the people with most "alike" tastes. The system suggest items that the similar users have rated highly but not yet being rated by the user (undoubtedly the absence of rating is often considered as the unknowingly of an item).

F.ARCHITECTURE DESIGN



TOPK-SK

Algorithm we introduce an efficient TOPK-SK query algorithm assuming that objects are organized by an IL-Quadtrees. Same as other inverted index based approaches, we also conduct incremental nearest neighbor search on the IL-Quadtrees. The main difference is that we can make use of the space partition based signatures(i.e., quadtree structures) to eliminate non promising objects.

Algorithm 1. TOPK-SK (*Q, q, k*)
Input: $\mathcal{LQ}$: the IL-Quadtree, *q*: the spatio-textual query
k: number of objects returned,
Output: $\mathcal{R}$: TOPK-SK query result.

1 $\mathcal{R} = \theta$; $\mathcal{H} = \theta$; $\lambda = \infty$;
2 for each $\mathcal{LQ}_i$ where $i \in I(q.\mathcal{T})$ **do**
3 Push root node of $\mathcal{LQ}_i$ into H;
4 **while** $\mathcal{H} \neq \theta$; **do**
5 e ← the quadtree node popped from $\mathcal{H}$;
6 **if** e is a black leaf node
7 Suppose *e* is from $\mathcal{LQ}i$;
8 **for** each $\mathcal{LQ}j$ where $j \neq i$ and $j \in I(q.\mathcal{T})$ **do**
9 **CheckSignature**$(e, \mathcal{LQ}_j)$;
10 if *e* passed the signature test **then**
11 Load objects contained by *e*;
12 **for** each object o contained by *e* **do**
13 $o_{hit} := o_{hit} + 1$;
14 **if** $o_{hit} = \ell$ **then**
15 Compute $\delta(o,q)$;
16 **if** $\delta(o,q) > \lambda$ **then**
17 Break;
18 **else**
19 Update λ and $\mathcal{R}$;
20 **else if** *e* is a non-leaf node **then**
21 **for** each child node *e'* of *e* **do**
22 **if** e is **not** a *white* leaf node **and** $\delta_{min}(e',q) \leq \lambda$ **then**
23 Push *e'* into $\mathcal{H}$;
24 **return** $\mathcal{R}$

Partition Algorithm

In number theory and computer science, the partition problem is the task of deciding whether a given multiset *S* of positive integers can be partitioned into two subsets and such that the sum of the numbers in equals the sum of the numbers. Although the partition problem is NP-complete, there is a pseudo-polynomial time dynamic solution, and there are probing that solve the problem in many instances,either optimally. For this reason, it has been called "the easiest NP-hard problem".There is an optimization version of the partition problem, which is to partition the multiset *S* into two subsets such that the difference between the sum of elements and the sum of elements is minimized. The optimization version is NP-Hard, but can be solved efficiently in practice.

Algorithm 2. Partition (O, Q, D)

INPUT: A list of integers "S"

OUTPUT: True if "S" can be partitioned into two subsets that have equal sum

```
1   " function " " find_partition" ("S"):
2    n ← |S|
3    "K" ← "sum(S)"
4     "P" ← empty boolean table of size ‘‘(<math>\lfloor K/2    \rfloor </math> + 1 ) " by ‘‘(n + 1)"
5    " initialize " top row (‘‘P (0, x)") of "P" to True
6     " initialize " leftmost column ("P(x, 0)") of "P", except for ‘‘P (0, 0) " to False
7     " for ‘‘‘‘i’’ "'from " 1 " to’’ <math>\lfloor K/2 \rfloor  </math>
8       "'for"' "j" "'from"' 1 "'to"' n
9          "'if"' ‘‘(i-S [j-1])" ">=" 0
10            ‘‘P (i, j)" ← ‘‘P (i, j-1)" "'or"' ‘‘P (i-S [j-1], j-1)"
11          " else "
12            ‘‘P (i, j)" ← ‘‘P (i, j-1)"
13   " return " ‘‘P (<math>\ lfloor K/2 \rfloor </math>, n)"
```

R*-Tree

This data structure is a popular method to localize rectangular objects in two or more dimensional space. In VRR, it is used for effective localization of geometrical objects near to a given mouse-click or to find all objects in a given rectangular area.R*-Tree is one of the variants to data structure called R-Tree. Each leaf node represents one item within a given bounding box. Internal nodes contain information about the smallest common bounding boxes of all their children. This arrangement allows us to walk the tree from root to leaves while ignoring large unimportant plane areas subtrees.

R*-Tree tree is based on a heuristic optimization and uses combination of several optimization criteria to arrange objects to groups and build a balanced tree over them. The data structure is fully dynamic. Insertions, deletions, updates and queries can be mixed and no periodic global reorganization is required.

```
/* R*-Tree main structure */
    struct geom_rtree {
      struct geom_rtree_node *root;
        /* pointer to root node (NULL if tree is empty) */
      clist lquery;
        /* list of dynamic rectangular queries */
    };

    /* R*-Tree leaf node */
    struct geom_rtree_obj {
      struct geom_rectangle bbox;
        /* rectangle enclosing entire geometrical object */
      struct geom_rtree_node *parent;
        /* pointer to parent tree node */
    };
```

```
/* R*-Tree internal node */
struct geom_rtree_node {
  struct geom_rectangle bbox;
    /* smallest common bounding box of node children */
  byte count;
    /* number of children */
  byte height;
    /* tree level (0 for internal nodes containing leaves) */
  struct geom_rtree_node *parent;
    /* parent node (NULL in root node) */
  struct geom_rtree_node *child[GEOM_RTREE_MAX +   1];
    /* child-pointers (internal nodes or leaves) */
};
```

CONCLUSION

The problem of top k spatial keyword search is important due to the increasing amount of spatio-textual objects collected in a wide spectrum of applications. In the paper, we propose a novel index structure, namely R*tree, to organize the spatio-textual objects. An efficient algorithm is developed to support the top k spatial keyword search by taking advantage of the R*tree. We further propose a partition based method to enhance the effectiveness of the signature of linear quadtree. To facilitate a large amount of spatial keyword queries, we propose a BTOPK-SK algorithm as well as a query group algorithm to enhance the performance of the system. Our comprehensive experiments convincingly demonstrate the efficiency of our techniques.

REFERENCE

1. Chengyuan Zhang, Ying Zhang, Wenjie Zhang, and Xuemin Lin, “Inverted Linear Quadtree: Efficient Top K Spatial Keyword Search “ IEEE Tran.Conf.Data Eng.,,

2. I. D. Felipe, V. Hristidis and N. Rishe, "Keyword search on spatial databases", Proc. IEEE 24th Int. Conf. Data Eng., pp.
3. D. Wu, M. L. Yiu, G. Cong and C. S. Jensen, "Joint top k spatial keyword query processing", IEEE Tran. Knowl. Data Eng., vol. 24, no. 10.
4. R. Hariharan, B. Hore, C. Li and S. Mehrotra, "Processing spatial-keyword (sk) queries in geographic information retrieval (gir) systems", Proc. 19th Int. Conf. Sci. Statist. Database Manage.
5. Z. Li, K. C. K. Lee, B. Zheng, W.-C. Lee, D. L. Lee and X. Wang, "Ir-tree: An efficient index for geographic document search", IEEE Trans. Knowl. Data Eng.,
6. G. Li, J. Feng and J. Xu, "Desks: Direction-aware spatial keyword search", Proc. IEEE 28th Int. Conf. Data Eng.