IJRSET JUNE 2017 Volume 4, Issue 6 Pages: 6-11
ISSN 2394-739X
International Journal for Research in Science Engineering and Technology

BEST KEYWORD SEARCH ON SPATIAL DATABASE WITH FRQUENT ITEM SEARCH

[1] S. VISHALAKSHI, [2] Dr. R. SUKUMAR
[1] PG SCHOLAR [2] PROFESSOR
[1, 2] KIT- KALAIGNAR KARNANIDHI INSTITUTE OF TECHNOLOGY

ABSTRACT: Databases nowadays contain large volumes of data, and they are accessed by numerous users on a daily basis. The large volume of data poses challenges to both users accessing the databases and the companies or organizations managing them. Enterprises on the other hand, need to make their content visible and accessible to the users and identify which objects in their database (e.g. products) have a significant impact on the user basis and use this information for promoting their products. The original target of increasing the visibility of the available products is thus hindered by the abundance of products contained in the database. It is therefore necessary to develop data exploration techniques that will enable users to explore large databases and provide them with a wide, yet coherent overview of objects that fit their preferences. In this paper, we propose exploratory algorithms that return to the user a small number of results, which at the same time provide a wide overview of the available content. In addition, we present algorithms that identify items that are appealing to users and can be exploited for offering users an insight of the available items and motivating them to explore the database. We also propose analysis techniques using FP growth algorithm for identifying frequent search objects that are attractive to the users. This algorithm is a more efficient algorithm that achieves results of comparable quality, but with significantly lower processing cost.

1. INTRODUCTION

Most companies today invest significant resources on making their content visible on the Web and enabling users to browse the offered products and services. Often, companies provide a plethora of different alternatives, which overwhelm the user and make it extremely difficult for them to find the products they are interested in.When users are searching in a database, they are usually unaware of the exact database content. Quite commonly, they do not have a concrete idea of the objects' properties they are searching for but only certain preferences about them. Consequently, they need to explore the database contents to find the objects that best fit their preferences. For instance, if someone wishes to buy a laptop, one may have a general idea about the desired characteristics, but an exact description of the laptop is difficult to be strictly determined. Traditional database queries are hard constraint queries, which return either exact matches or nothing. In addition, hard constraint queries are in general quite complex, and in order to produce useful results, they require the user to be aware of the database content. They also often require the knowledge of a specific query language and

the structure of the queried database. Moreover, hard constraints are quite likely to produce very small or extremely large result sets that provide little insight of the available data. As a result, users are led to pose repeatedly new queries until they retrieve a satisfying result set. Therefore, they are inappropriate for exploratory search as they pose significant difficulties to users searching the database.

Users experience frustration when they are not able to easily find the information they need. In an attempt to make database content easily accessible, several approaches have been proposed, which allow users to express their needs by posing preference queries using either sets of keywords or by indicating their interest on the objects' attributes they are searching for. The query result is typically a list of objects, usually ranked according to a function that measures the relevance or the performance of each object with respect to the query.

A key aspect that preference queries fail to capture in its entirety is the fact that users performing exploratory search are generally unfamiliar with the domain of the data they are searching, and they are possibly unclear about their wishes. A flat list of results provides little insight to the user about the available information. In addition, the relaxation of constraints induced by preference queries introduces ambiguity to the search, as each keyword query could be associated with a large number of database queries. As a result queries can produce a large number of redundant results, which the user has to filter out.

In this paper, we propose exploratory algorithms that return to the user a small number of results, which at the same time provide a wide overview of the available content. In addition, we present algorithms that identify items that are appealing to users and can be exploited for offering users an insight of the available items and motivating them to explore the database. We also propose analysis techniques using FP growth algorithm for identifying frequent search objects that are attractive to the users.

2. RELATED WORKS

The 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 similarity between documents are applied to measure the relevance between two sets of keywords.

Efficient Processing of Direction Joins Using R-trees [1] presents an efficient method for processing direction joins using R-trees. The quad-tuples model is defined to represent direction relations between the minimum bounding rectangles of spatial objects. An algorithm of processing the filter step of joins using R-trees is given and the refinement step processing is further decomposed into three different operations.Answering Why-Not Spatial Keyword Top-k Queries via Keyword Adaption [2] A spatial keyword top-k query takes a user location and a set of keywords as arguments and retrieves the k objects that are ranked the highest according to a scoring function that considers both spatial distance and textual similarity. Efficient Collective Spatial Keyword Query Processing on Road Networks [3] We study the problem of collective spatial keyword queries on road networks (i.e., CSKQ on road networks), which retrieves a set of POIs (Point of Interests) that collectively cover the queried keywords and have the lowest cost, measured by their shortest path distances to a specified query position, and the inter-POI distances between POIs in the set. Efficient Top-k Spatial Locality Search for Co-located Spatial Web Objects [4] Locality Search, a query that returns top-k sets of spatial web objects and integrates spatial distance and textual relevance in one ranking function. Keyword Search on Spatial Databases [5] we introduce an indexing structure called IR2-Tree

(Information Retrieval R-Tree) which combines an R-Tree with superimposed text signatures. We present algorithms that construct and maintain an IR2-Tree, and use it to answer top-k spatial keyword queries. Challenges in the Design and Implementation of Wireless Sensor Networks: A Holistic Approach- Development and Planning Tools, Middleware, Power Efficiency, Interoperability[6] WSN challenges by developing an integrated platform for smart environments with built-in user friendliness, practicality and efficiency. This platform will enable the user to evaluate his design by identifying critical features and application requirements. Location Aware Keyword Query Suggestion Based on Document Proximity[7] weighted keyword-document graph, which captures both the semantic relevance between keyword queries and the spatial distance between the resulting documents and the user location. The graph is browsed in a random-walk-with-restart fashion, to select the keyword queries with the highest scores as suggestions.

The number of candidate keyword covers generated is significantly reduced.
Mining user's availability based on their interest.

Compared to the baseline algorithm, the number of candidate keyword covers generated in this algorithm is significantly reduced.

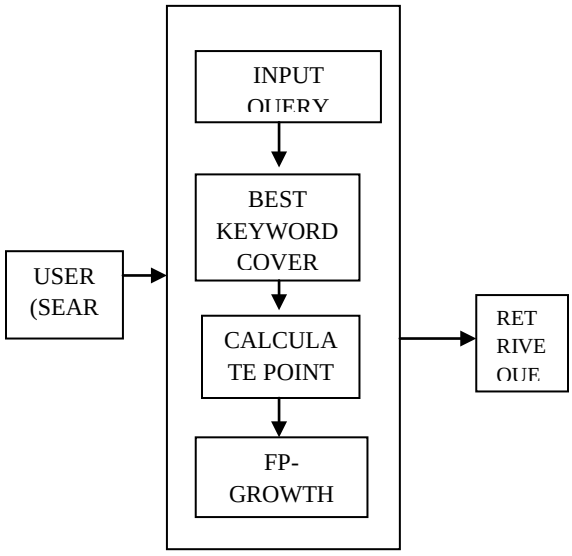

Figure 1: Architecture Diagram

3. PROPOSED WORK

It is motivated by the observation of increasing availability and importance of keyword rating in decision making. We presented algorithms for the identification of objects that are constantly attractive for a large number of users over a specified period of time. FP algorithm is used for identify the frequently searched items in a spatial database through this we improve the best keyword search for the web users.Searching local best solution for each object in a certain query keyword.

3.1 Spatial Database module

The spatial data model thus consists of a set of abstract concepts that can be used to describe any object that has a spatial extent in one or more dimensions. Spatial data, in turn, consist of spatial objects comprised of points, lines, regions, rectangles, surfaces, volumes, and more abstract dimensions such as time Examples of such objects include roads, rivers, cities, forests, mountains, or other geographical landmarks and areas. In a regular relational database, objects are stored as a collection of tuples , where each tuple consists of several fields belonging to different data types. A spatial object can be stored in such a database by naively creating a field for each of the spatial dimensions of the object, or any of the other spatial properties it is desirable to store.

3.2 Getting User Interest and Activity

A point of interest, or POI for short, is a specific point location that someone may find useful or interesting. POIs can be used in navigation, characterization of a place, sociological studies, city dynamics analysis, geo-reference of texts, etc. Such a simple information structure can be used and

enriched such that context-aware systems behave more intelligently. In spite of their importance, the production of POIs is scattered across a myriad of different websites, systems and devices, thus making it extremely difficult to obtain an exhaustive database of such a wealthy information.

3.3 Frequent Search Item Mining

Frequent-pattern growth or simply FP-growth, which mines the complete set of frequent item sets without candidate generation. This method adopts a divide-and-conquer strategy as follows: first it compresses the database representing frequent items into frequent-pattern tree, or FP-tree, which retains the item set association information. It then divides the compressed database into a set of conditional database , each associated with one frequent item or pattern fragment, and mines each such database separately.

3.4 Search Result

User enters search criteria and executes search. The system shall display results which match the search criteria entered. The view shall default to that selected by the user (i.e. Suspect/Case). If the User chooses to change the view on the results page, The system shall switch between views depending on the view (Suspect/Case) selected.The search space is exponential in the number of items occurring in the database and the targeted databases tend to be massive, containing millions of transactions. Both these characteristics make it a worthwhile effort to seek the most efficient techniques to solve this task.

FP growth Algorithm

The FP-growth method transforms the problem of finding long frequent patterns to searching for shorter ones recursively and then concatenating the suffix. It uses the least frequent items as a suffix, offering good selectivity. The method substantially reduces the search costs.

In general when considering the branch to be added for a transaction, the count of each node along a common prefix is incremented by 1 and nodes for the items following the prefix are created and linked accordingly. 12 To facilitate tree traversal, an item header table is built so that each item points to its occurrences in the tree via a chain of node-links. In this way the problem of mining frequent pattern in database is transformed to that of mining the FP-tree. The FP-tree is mined as follows: Start from each frequent length-1 pattern, as an initial suffix pattern, construct its conditional pattern base, a sub-database, which consists of the set of prefix paths in the FP-tree co-occurring with the suffix pattern, then construct its conditional FP-tree and perform mining recursively on such a tree. The pattern growth is achieved by the concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-tree.

The FP-growth algorithm: mine frequent item sets using an FP-tree by pattern fragment growth.

Input:

§ D, a transaction database

§ min_sup, the minimum support count threshold

Output:

the complete set of frequent patterns.

Method:

(1) the FP-tree is constructed

(2) The FP-tree is mined by calling FP-growth(FP_tree, null):

procedure FP_growth(Tree, α) if Tree contains a single path P then for each combination (denoted as β) ofthe nodes in the path P generate pattern βUα with support_count = minimum support count of nodes in β; else for each ai in the header of Tree{ generate pattern β=aiUα with support_count = ai .support_count construct β's conditional

pattern base and then β's conditional FP_tree Treeβ ;
if Treeβ != 0 then
call FP_growth(Treeβ , β); }

4. EXPERIMENTAL RESULT AND DISCUSSION

In this round of evaluation, we first conduct an experiment varying the number of objects to evaluate the scalability of the indices for each type of query. In addition, to evaluate the effect of the text size of each object, we conduct an experiment varying the average number of words per object for the best search.
keywords is large or a space limitation has to be strictly satisfied. Grid based indices are not attractive for the BKS compared with the other indices. We do not find dramatic difference in relative performance among the indexing techniques.

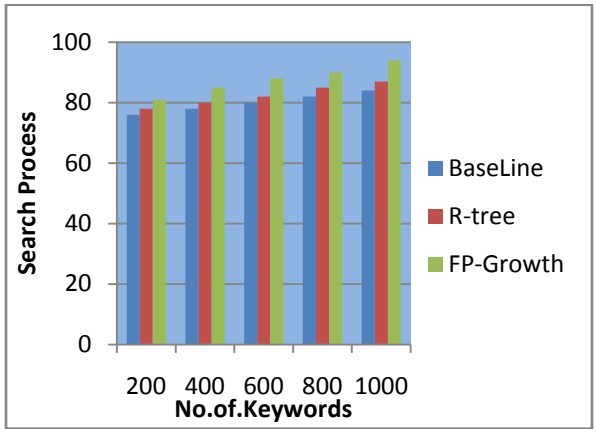

Fig 2 : Experimental Chart

The query processing of the indices scales linearly with the number of objects and text length per object. Text-first indices are more sensitive to text length than the other types of indices. - The FP-tree page size for all FP-tree based indices is experimentally shown to be a factor that makes a great difference on query performance.

CONCLUSION

The major problem with frequent set mining methods is the explosion of the number of results. It is difficult to find the most interesting frequent item sets in the set mining methods.In this paper we propose exploratory algorithms that return to the user a small number of results, which at the same time provide a wide overview of the available content. In addition, we present algorithms that identify items that are appealing to users and can be exploited for offering users an insight of the available items and motivating them to explore the database. We also propose analysis techniques using FP growth algorithm for identifying frequent search objects that are attractive to the users.

REFERENCES

[1]. Rakesh Agrawal and Ramakrishnan Srikant. "Fast algorithms for mining association rules in large databases". In: VLDB. 1994, pp. 487–499.
[2]. T. Brinkhoff, H. Kriegel, and B. Seeger. "Efficient processing of spatial joins using R-trees". In: SIGMOD (1993), pp. 237–246.
[3]. Xin Cao, Gao Cong, and Christian S. Jensen. "Retrieving top-k prestige-based relevant spatial web objects". In: Proc. VLDB Endow. 3.1-2 (2010), pp. 373–384.
[4]. Xin Cao et al. "Collective spatial keyword querying".In: ACM SIGMOD. 2011.
[5]. G. Cong, C. Jensen, and D. Wu. "Efficient retrievalof the top-k most relevant spatial web objects". In: Proc. VLDB Endow. 2.1 (2009), pp. 337–348.
[6]. Ian De Felipe, Vagelis Hristidis, and Naphtali Rishe. "Keyword Search on Spatial Databases". In: ICDE. 2008, pp. 656–665.
[7]. R. Fagin, A. Lotem, and M. Naor. "Optimal Aggregation Algorithms for Middleware". In: Journal of Computer and System Sciences 66 (2003), pp. 614–656.
[8]. Ramaswamy Hariharan et al. "Processing Spatial-Keyword (SK) Queries in Geographic Information Retrieval (GIR) Systems". In: Proceedings of the 19th International Conference on Scientific and Statistical Database Management. 2007, pp. 16–23.3

[9]. G. R. Hjaltason and H. Samet. “Distance browsing in spatial databases”. In: TODS 2 (1999), pp. 256–318.
[10]. Z. Li et al. “IR-tree: An efficient index for geographic document search”. In: TKDE 99.4 (2010), pp. 585–599.
[11]. N. Mamoulis and D. Papadias. “Multiway spatial joins”. In: TODS 26.4 (2001), pp. 424–475.
[12]. D. Papadias, N. Mamoulis, and B. Delis. “Algorithms for querying by spatial structure”. In: VLDB (1998), p. 546.
[13]. D. Papadias, N. Mamoulis, and Y. Theodoridis. “Processing and optimization of multiway spatial joins using R-trees”. In: PODS (1999), pp. 44
[14]. J. M. Ponte and W. B. Croft. “A language modeling approach to information retrieval”. In: SIGIR (1998), pp. 275–281.
[15]. Jo˜ao B. Rocha-Junior et al. “Efficient processing of top-k spatial keyword queries”. In: Proceedings of the 12th international conference on Advances in spatial and temporal databases. 2011, pp. 205–222.
[16]. S. B. Roy and K. Chakrabarti. “Location-Aware Type Ahead Search on Spatial Databases: Semantics and Efficiency”. In: SIGMOD (2011).