An Efficient Query Processing Search Using Textual and Spatial Relevances

Santhanakrishnan. C, Sivaprakasam. V, Rajasekar. R, Sudhakar. D & Lourdu Michael Antony. K

Department of Computer Science, SRM University, Chennai -603 203
E-mail : santhanakrishnan18@gmail.com, sivaprakasam98.6@gmail.com, rajasekar3388@gmail.com

***Abstract* – Geographic search engine query processing is different in that it requires a combination of textual and spatial data, it retrieves a document that is relevant to the query keywords and the location with respects to ranks the documents that are retrieved according to textual and spatial relevance to the query. The proposed IR-tree with a top-k document search algorithm for efficient query processing facilitates four major tasks in document searches: they are 1) spatial narrowing, 2) textual narrowing, 3) relevance computation and 4)rank the document in a integrated manner. The lack of an efficient index that can simultaneously handle both the textual and spatial aspects of the documents makes existing geographic search engines in efficient in answering geographic queries. IR-tree adopts different weights on textual and spatial relevance of documents search at the runtime. A set of comprehensive experiments over a wide range of scenarios has been conducted and the experiment results demonstrate that IR-tree outperforms the state-of-the art approaches for geographic document searches.**

Keywords – Geographic document search, index, search algorithm and IR-tree.

I. INTRODUCTION

An information retrieval process begins when a user enters a query into the system. Queries are formal statements of information needs. In information retrieval a query does not uniquely identify a single object in the collection. Instead, several objects may match the query, perhaps with different degrees of relevancy. Most IR systems compute a numeric score on how well each matches the query, and rank the objects according to this value. The top ranking objects are then shown to the user. The process may then be iterated if the user wishes to refine the query. Many different measures for evaluating the performance of information retrieval systems have been proposed. The measures require a collection of documents and a query. All common measures described here assume a ground truth notion of relevancy: every document is known to be either relevant or non-relevant to a particular query.

A geographic search engine is required to quickly return documents of high relevance in both textual and spatial aspects to a given geographic query. However, designing an efficient index structure for both textual and spatial information is not trivial, as four major challenges need to be overcome. First, each keyword in the documents is usually treated as one dimension in the document space. Indexes for document search need to cover a very large high-dimensional search space. Second, words and locations in geographic documents have different forms of representations and measurements of relevances to a query. A coherent index that can seamlessly integrate these two aspects of geographic documents is very desirable. Third, the words and location of a document have separate influences on the overall relevance of the document to a query, while the relative importance of textual and spatial relevance is very much subjective to the user. Various combinations of these two factors are necessary to accommodate diversified user needs. Thus, an ideal index should allow search algorithms to adapt to different weights between textual and spatial relevance of documents at the runtime. Last but not the least, the index structure together with an appropriate search algorithm has to facilitate efficient determination of both textual relevance and spatial relevance of the documents while performing document ranking in order to guarantee high search efficiency. However, existing approaches are inefficient in processing geographic document search. This motivates to design an efficient index structure, namely, IR-tree, for geographic search engines which effectively addresses all four challenges discussed above. The strength of IR-tree lays in its ability to perform document search, document relevance computation, and document

ranking in an integrated fashion. In brief, IR-tree indexes both the textual and spatial contents of documents that enable spatial pruning and textual filtering to be performed at the same time during query processing. A top-k document search algorithm based on IR-tree combines both the search and ranking processes, thus effectively reducing the number of documents examined.

II. IR TREE

IR tree is a tree data structure which is used as an index to handle location based queries. IR tree is designed such that it performs spatial clustering first and then textual filtering. Here first spatial filtering is done so that search space can be abridged because there may be many documents that are textually related but only very few of those are bounded within spatial scope. Now textual filtering is done so as to reduce search cost. Finally, the joint relevance and ranking is done simultaneously such that, as soon as top k (the number of documents to be retrieved) documents are obtained the search process stops.

Coming to the design issue, index structure must be designed in proper way as each textual word in documents is treated as a dimension. Document space need to cover many very high dimensional spaces. In addition to that spatial locations and textual words have their own representations and measurements. So index must integrate these two aspects so that they must be compatible.

Our IR Tree is designed to perform spatial filtering, textual filtering, relevance computation, and ranking simultaneously. Even storage and access overheads are considered.

2.1 IR Tree Structure

IR tree is designed in such a way that it clusters spatial documents and abstracts textual documents under various granularities [1]. All the spatially related documents are clustered so that any document that does not belong to that region requested by the user, can be pruned as and then as unrelated. All textual words are represented using inverted files. Each node has document précis such that if the query keyword is present in that node then it can traverse according to the nodes pointing it. IR tree is a collection of nodes.

It consists of a root node, few non leaf nodes, and few leaf nodes.

2.1.1. Leaf nodes

Each leaf node is linked to an inverted file. All the inverted files consist of list of words, such that each word is pointed to list of documents that contain the particular word. It can be represented as shown in fig 2

2.1.2. Non leaf nodes

All the non leaf nodes consist of document précis. Document précis is nothing but collection of information regarding nod's spatial region, number of documents that come under that particular node. It even contains the WF and IWF. It is shown in fig 3. In brief, let the non leaf node be node i, then will have many children nodes to node i. Document précis contains

1. Mi: It is the Minimal Bounding Box that covers all the locations of the documents under node i. It is nothing but a small rectangular region that covers all the locations in the document set under the node i.
2. |Wi|: It is the cardinality of the documents that come under the node i. i.e., the number of documents that come under node i.
3. WF and IWF pair values: WFt,w is the Word Frequency i.e.; it is the measure of frequency of a word t that occurs in a document w. IWFt,w is the Inverse Web page Frequency , the number of documents in the document set W that contain one or more occurrences of textual word t. This pair helps in computing the relevance just by checking the WF and IWF values as the node need not be considered if the pair value is low.

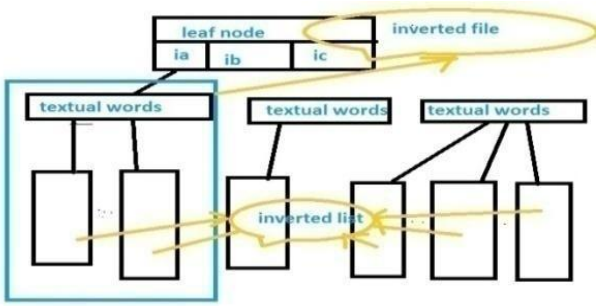

Fig. 1: Leaf node representation

2.2 IR Tree operations

The IR-tree can be manipulated with three operations, namely, bulk loading documents, inserting documents, and deleting documents. Given a set of documents, bulk loading creates an IR-tree from scratch. The pseudocode is depicted in Algorithm 1. As a brief description, it first clusters documents based on their spatial locations into leaf-level entries, and then groups the formed entries as nodes in a bottom-up fashion repeatedly until the root is formed.

ALGORITHM 1: IR TREE CONSTRUCTION

INPUT: a document set, D; minimal node fan-out, min; maximal node fan-out, max;

OUTPUT: the root of an IR-tree

PROCEDURE

1. $N_e \leftarrow Ø$
2. *for* each d Є D *do*
3. geocode d and represent L_d with MBB $m_{d;}$
4. *if* $\exists_e$ Є N_e , $m_e = m_d$ then
5. add d to e's document set $D_{e;}$
6. *else*
7. create a new entry e;
8. set $m_e \leftarrow m_d$ and $D_e \leftarrow \{d\}$;
9. $N_e \leftarrow N_e$ U {e};
10. *end if*
11. *end for*
12. for each e Є N_e do
13. build inverted file with each list l_w w.r.t. every word w in at least one document d Є D_e
14. *end for*
15. *while*| N_e| > n_{max} *do*
16. cluster N_e according to min/max into nodes, represent as new entries N'_e; form document summary for e in N'_e;
17. $N_e \leftarrow N'_e$;
18. *end while*
19. create the root node to cover N_e and their document summaries;
20. *output* the root node;

III. TEXT RETRIEVAL

Information retrieval (e.g. Web search engines) concerns essentially with two main activities: indexing and searching. Indexing refers to representing data for the purpose of efficient retrieval, and is done after pre-processing operations have taken care of extracting appropriate items (i.e. tokenizing text). Various text indexing methods have been developed. Inverted indexes are the most popular technique, consisting of a set of inverted lists, one for each occurring word or index term. The inverted list for a term is a sorted list of positions, or hits, where the term appears in the collection. A hit consists of a document identifier and the position of the term within it, often containing additional information useful for ranking (e.g. HTML markup). Figure 1 shows a forward index (usually created as a first step in making an inverted index) and an inverted index for two example documents.

Searching involves the use of the structure built in the indexing stage for processing queries. A typical query contains terms and operators (i.e. disjunction, conjunction and filters). The indexes are examined to find matching documents, and a similarity score is computed between the query and each document. A ranked list is finally computed according to the similarity scores. The term weighting and document ranking function known as Okapi BM25 is the state-of-the-art in ranking results for text IR, and extensions to HTML documents have also been proposed.

In Web IR, citations and hypertext links are commonly combined with document content to improve ranked retrieval. Page Rank is the most popular link-based ranking algorithand researchers have evaluated different techniques for combining standard text-based techniques with link-based ranking scores.

IV. RETRIEVAL WITH GEOSCOPES

Currently, two types of approaches are used by existing geographic search engines, namely, Approach I that uses separated indexes for spatial information and textual information, and Approach II that uses a combined index. However, they both are not efficient. Approach I logically extends conventional textual search engines with spatial filtering capability of Quad-tree, R-tree, and Grid index as suggested in respectively.

Step 1:Retrieving textually relevant documents with respect to query keywords via a conventional textual index.

Step 2: Filtering out the documents obtained from Step 1 that are not covered by the query spatial scope.

Step 3: Ranking the documents from Step 2 based on the joint textual and spatial relevances in order to return the ranked results to the user.

Approach I is inefficient. First of all, a keyword-based search may retrieve a large number of textually relevant documents that are outside the spatial scope. Take our evaluation as an example. More than 90 percent of the textually relevant documents are outside the query spatial scopes. Although it is possible to reorder Steps 1 and 2 based on their selectivity, performance improvement is rather limited if the selectivity in Steps 1 and 2 are both high. Besides, the

ranking process is not incremental, i.e., it has to sort all of the candidate documents based on the joint textual and spatial relevances in Step 3 in order to find the top-k documents.

In addition to having geo-scopes associated with the documents, and similarly to text IR, retrieving documents with basis on geographical. The relevance of a location can hypothesize with respect to a query region increases with decreasing Euclidean distance between them. The extent of overlap can also be used to measure spatial relevance. For instance, the greater the overlap between the two regions, the greater the assumed relevance. Besides spatial distance, we can define notions of topological distance between locations. Hierarchical measures can, for instance, use the number of non-common parents between a pair of places within the hierarchies to which they belong or the minimum number of direct relationships separating both places at an ontology . Besides edge-counting, semantic similarity Measures can also take into consideration hierarchy depth, or even things like language, population, and non-geographical relations. The problem of measuring similarity in hierarchical semantic structures has in fact been extensively studied. Combinations of semantic and spatial methods can also be used to create hybrid metrics, which in turn can be further combined with thematic similarity to create an integrated Geo-IR relevance ranking metric. A good motivation for using semantic similarity is that Euclidean space has been noted as unsuitable for modeling geographical proximity. The concept of proximity is asymmetric, as people can consider *A* is near *B* while considering *B* is not near *A*. This asymmetry is related to the sizes and importance of geographical objects (e.g. total population or economic relevance), and the existing relationships with other geographical objects.

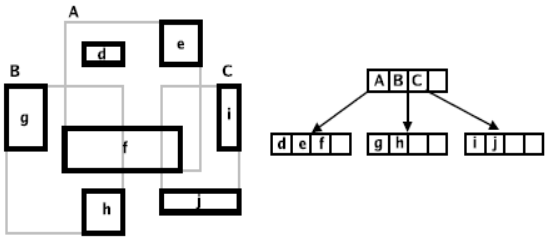

Fig. 2: Rectangles arranged in R-tree hierarchy

Different multi-dimensional indexes have been proposed for managing spatial data, including grid indexes, quad-trees, R-trees, kd- trees, and space filling curves such as Z-order . Since geoscopes can be seen as spatial footprints, these schemes can be used for document retrieval in a Geo-IR system. A geo-retrieval algorithm can follow this general guideline:

1. Transform the location and the spatial operators in the query into a geo-scope or more, if the query cannot be disambiguated into a single geo-scope.
2. Rank each geo-scope in the set according to how relevant they are to the query location.
3. Get the ranked list of documents matching the set of geoscopes. Ranking is based on the relevance of the documents to their corresponding scopes, obtained from the (pre-ordered) index, combined with the relevance score assigned to each scope from the query (e.g. a linear combination).

V. TOP K-DOCUMENT RETRIEVAL

After buffer B containing candidate IR-tree nodes is returned by the IDF Calculation algorithm, Top-k Document Retrieval algorithm as the second step of the search runs to identify the result documents. As the candidate set might contain far more documents than k, this step tries to avoid examining nonresult documents. Our strategy is to evaluate the documents based on their joint spatial and textual relevances with respect to a given query q and to terminate the process once the top-k result documents are obtained. Algorithm 4 lists the pseudocode of top-k document retrieval. It maintains a priority queue Q that orders the pending entries (either nodes or documents) in descending order of their relevance with respect to q (lines 1-3).

ALGORITHM 2: TOP K DOCUMENT RETRIEVAL

INPUT: a set of idf values $\{idf_{w,D,Sq}, w \in W_q\}$; a candidate set, B; query keyword,W_q; query spatial scope S_q;a ratio between textual and spatial relevance,α; the number of returned document, k

OUTPUT: the k most relevant document, R;

PROCEDURE:

1. *MACRO* :$\psi(\epsilon) = \alpha \cdot \sum_{w \epsilon Wq}(tf^{max}_{w,\epsilon}.idf_{w,D,Sq})+(1-\alpha)/dist()A_\epsilon,S_q$;
2. *for* each entry $\epsilon \in$ B *do*
3. enqueue (ϵ ,$\psi(\epsilon)$) to Q; //initialize Q with entries in B
4. *end for*
5. *while* Q is not empty *do*
6. dequeue an entry ϵ from Q;
7. *if* ϵ is a document *then*
8. R $\leftarrow$ R U {ϵ};
9. *if* |R|= k *then*

10. *goto* 22;
11. *end if*
12. *else if* ϵ is a leaf node *then*
13. *for* each document d in ϵ's inverted list l_w, $\forall w \in W_q$ do
14. enqueue $(d,\psi(d))$ to Q;
15. *end for*
16. *else*
17. *for* each child c of ϵ *do*
18. enqueue $(c, \psi(c))$ to Q;
19. *end for*
20. *end if*
21. *end while*
22. *output* R;

VI. CONCLUSION

This approach is to present the textual information along with the location information by giving the query keyword. The previous result will focus on the efficiency issues of geographic document search. At present try to improve the high search efficiency with geographical information and also plan to further enhance the IR-tree index based on various access patterns.

VII. REFERENCES

[1] D. Hiemstra, "A Probabilistic Justification for Using TF x IDF Term Weighting in Information Retrieval," Int'l J. Digital Libraries, vol. 3, no. 2, pp. 131-139, 2000.

[2] A. Markowetz, Y.-Y. Chen, T. Suel, X. Long, and B. Seeger, "Design and Implementation of a Geographic Search Engine," Proc. Eighth Int'l Workshop Web and Databases (WebDB), pp. 19-24, 2005.

[3] Y.-Y. Chen, T. Suel, and A. Markowetz, "Efficient Query Processing in Geographic Web Search Engines," Proc. ACM SIGMOD '06, pp. 277-288, 2006.

[4] Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma, "Hybrid Index Structures for Location-Based Web Search," Proc. 14th ACM Int'l Conf. Information and Knowledge Management (CIKM '05), pp. 155- 162, 2005.

[5] V.N. Anh and A. Moffat, "Pruned Query Evaluation Using Pre- Computed Impacts," Proc. ACM SIGIR '06, pp. 372-379, 2006.

[6] I.D. Felipe, V. Hristidis, and N. Rishe, "Keyword Search on Spatial Databases," Proc. IEEE 24th Int'l Conf. Data Eng. (ICDE '08), pp. 656-665, 2008.

[7] R. Hariharan, B. Hore, C. Li, and S. Mehrotra, "Processing Spatial- Keyword (SK) Queries in Geographic Information Retrieval (GIR) Systems," Proc. 19th Int'l Conf. Scientific and Statistical Database Management (SSDBM '07), pp. 16-25, 2007.

[8] E. Amitay, N. Har'El, R. Sivan, and A. Soffer, "Web-a- Where: Geotagging Web Content," Proc. ACM SIGIR '04, pp. 273-280, 2004.

◈◈◈