© 2014 IJIRT | Volume 1 Issue 10 | ISSN: 2349-6002

Retrieving Datasets with Nearest Neighbor Search using Spatial Queries

Ms.R.Keerthika[1], Dr.C.Nalini[2]
[1]*Assistant Professor, Department of Information Technology, Karpagam College of Engineering, Coimbatore.*
[2]*Professor,Department of Information Technology, Kongu Engineering College, Perundurai.*

***Abstract*— Adapted Web Search Mechanism (AWSM) is established in order to improve the quality of various search services on the Internet. However, it shows that users not interested to release their private information during search. It has become a major difficulty for the wide number of web search. We study privacy protection in web search applications that mainly focuses in user preferences as hierarchical user profiles. We propose an Adapted Web Search Mechanism (AWSM) framework called UPS that can be simplified the profiles by queries in terms of user specified privacy requirements. Our runtime simplification aims at an outstanding balance between two analytical metrics that estimate the utility of personalization and the privacy risk of exposing the universal profile. So in this paper we develop an access method called the spatial inverted index (SI-index). It has the ability to perform keyword augmented nearest neighbor search.**

***Index Terms*— R- Tree, SI index, Nearest Neighbor, Spatial Database.**

I. INTRODUCTION

A spatial database manages multidimensional objects and provides fast access to those objects on different range of criteria. The significance of spatial databases is reflected by the convenience of entities of reality in a geometric manner. For example smaller extents such as hotels, malls are represented in the form of small points in a map where as oceans, lakes and larger landscapes are represented in the form of rectangles.

Because of increase in the application and fast technological development in geographical information systems, geographic search engine has been receiving a lot of attention from industry. Same as the conventional search engines, In order to quickly return documents of high relevance in both textual and spatial aspects to a given geographic query geographic search engine is required. As a core of search engines, an index structure seems to be very important. However, manipulating a resourceful 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 relevance to a query. A coherent index that can faultlessly integrate these two aspects of geographic documents is very attractive. 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 personal to the user. Different combinations of these two factors are necessary to accommodate expanded user needs. And so, an ideal index must allow the search algorithms to adapt to different weights between textual and spatial relevance of documents at current state. Finally, 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. In this paper, we design an index structure IR-tree for geographic search engines which successfully addresses all four challenges discussed above.

The strength of IR-tree is ability to perform document searching a document, computational document relevance, and the document ranking in an integrated fashion. At this point, an IR-tree index both the documents of spatial and textual contents 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 ranking and process; In point of fact it reduces the number of documents observed. A set of extensive experiments over a wide range of system and query parameters has been conducted. The experimental results exhibit that IR-tree significantly outperforms the state-of-the-art approaches for geographic document search. The inputs of this paper are shortened as follows:

We propose IR-tree which indexes both the documents of spatial and textual contents to support document retrievals based on their combined textual and spatial relevance's, which, sequentially, it can be adjusted with dissimilar relative weights. We design a rank-based search algorithm based on IR-tree to effectively combine the

searching and ranking process to minimize Input/output costs for higher search efficiency. We perform a cost analysis for IR-tree and conduct an extensive set of experiments over a wide range of parameter settings to observe the efficiency of IR-tree.

II. RELATED WORKS

As mentioned before, the IR2-tree combines the R-tree with signature files. Next, we will review what the signature file is before explaining the details of IR2-trees. Our discussion assumes the knowledge of R-trees and the best first algorithm for NN search, both of which are well known techniques in the spatial databases.

Signature file in the general refers to the had Ring-based framework, whose instantiation in the is known has superimposed coding (SC), which is shown to be more effective than other works. It is calculated to perform membership tests to determine whether the query word w exists in the set W of the words. SC is conservative, in that sense if it says "no", then w is definitely not in the W. If, on the other hand, SC returns "yes", the true answer can be either way, in which case the whole W must be scanned to avoid the false hit.

SC works in the same way has the classic technique of bloom filter in the pre processing, it builds the bit signature of length l from W by had hinge each word in the W to the string of l bits, after that taking the disjoint of entire bit strings. To illustrate, denote by h(w) the bit string of the word w. First, all the l bits of h(w) are initialized to 0. Then, the SC replicates the following m times: randomly choose the bit and set it to 1. Very importantly, randomization must use w has its seed to ensure that the same w always ends up with the identical h(w).

Furthermore, choices are the mutually independent, and may even happen to be the same bit. The concrete values of l and m affect the space cost and false hit probability, has will be discussed later. Bits are set to 1. As mentioned earlier, the bit signature of the set W of words simply ORs the bit strings of all the members of W. For example, the name of the set {a, b} equals 01101, while that of {b, d} equals 01111.

Given the query keyword w, SC performs the member-ship test in the W by checking whether all the 1's of h(w) appear at the same positions in the signature of W. If not, it is guaranteed that w cannot belong to W. Otherwise, the test cannot be resolved using only the signature, and the scan of W follows the false hit occurs if the scan reveals that W actually does not contain w. For example, assume that we want to test whether word c is the member of set {a, b} using only the set's signature 01101. Since the 4th bit of h(c) = 00011 is 1 but that of 01101 is 0, SC immediately reports "no" another example, consider the membership test of c in the {b, d} whose signature is 01111. The time, SC returns "yes" because 01111 had 1's at all the bits where h(c) is set to 1; as the result, the full scan of the set is required to verify that the is the false hit

TABLE 1: Membership Test for SC and Bits and Dataset Values with Signature.

SC	Bits
h(w)	0
h(w)	1

Datasets	
set{a,b}	1101
set{b,d}	1111

TABLE 2: Membership Test for Word C and Bits and Dataset Values with Signature.

C	Bits
h(c)	1
h(c)	0

Datasets	
set{a,b}	1101
set{b,d}	1111

The IR2-tree is an R-tree where each (leaf or non leaf) entry E is augmented with the signature that summarizes the union of the texts of the objects in the sub tree of E. The string 01111 in the leaf entry p2, for example, is the signature of Wp2= {b, d} .The string 11111 in the non leaf entry E3is the signature of Wp2 ∪Wp6, namely, the set of all words describing p2and p6. Notice that, in the general, the signature of the non leaf entry E can be conveniently obtained simply has the disjunction of all the signatures in the child node of E. the non leaf signature may allow the query algorithm to realize that the certain word cannot exist in the sub tree. For example, has the 2nd bit of h(w) is 1, we know that no object in the sub trees of E4and E6can have word b in the its texts – notice that the signatures of E4and E6have 0 has their 2nd bits. in the general, the signatures in the IR2-tree may have different lengths at various levels.

Signatures indicate the absence of at least one word of Wq in the subtrees. Whenever the leaf entry, say of point p, it cannot be pruned, the random Input/output is performed to retrieve its text description Wp. If Wq is the subset of Wp, the algorithm terminates with p has the answer; or else, it maintains s until no more entry remains to be processed. Assume that the query point q had the keyword set Wq= {c,d}. It can be verified that the algorithm must read entire nodes of the tree, and take the documents of p2, p4, and p6 (in the order). The final answer is p6, while p2and p4are false hits.

A. Solutions Based on the Inverted Indexes

Inverted indexes (I-index) have proved to be an effective access method for keyword-based document retrieval. in the spatial context, nothing prevents us from treating the text description Wq of the point p as the document, and then, building an I-index. Each word in the vocabulary had an inverted list, specifying the entire id's of

the points which have the word in the documents. Note that the list of each word maintains the sorted order of point id's, which provides extensive handiness in the query processing by allowing an efficient merge step. For example, assume that we want to find the points that have words "c" and "d". This is basically to calculate the connection of the two words inverted lists. has both lists are sorted in the similar order, we can do by integrating them, whose Input/output and CPU times are both linear to the total length of the lists. Recall that, in the NN processing with IR2-tree,the point retrieved from the index must be verified(i.e., having its text description loaded and checked). Verification is also necessary with I-index, but for exactly the opposite reason. For IR2-tree, verification is because we do not have the detailed texts of the point, while for I-index; it is because we do not have the coordinates. Specifically, given an NN query q with keyword set Wq, the query algorithm of I-index first retrieves (by merging) the set Pq of all points that have all the keywords of Wq, and then, performs |Pq| random Input/outputs to get the coordinates of each point in the Pq in the order to evaluate its distance to q.

B. Drawbacks of the IR2-TREE

The IR2-tree is the first access method for answering NN queries with keywords. It have many initiating solutions, the IR2-tree also had the few drawbacks that affects the efficiency. the most serious one of all is that the number of false hits can be really large when the object of the final result is far away from the query point, or the result will be empty. in these cases, the query based algorithm would need to load the documents of many objects, incurring expensive overhead has each loading necessitates the random access.

To explain the details, we need to first discuss some properties of SC (the variant of signature file used in the IR2-tree). Recall that, at first glance, SC had two parameters: the length l of the signature, and the number m of bits chosen to set to 1 in the had ring the word. there is, in the fact, really just the single parameter l, because the optimal m (which minimizes the probability of the false hit) had been solved by Stiassny [19]

$$mopt= l \cdot \ln(2)/g \qquad (1)$$

Where g is the number of distinct words in the set W on the signature is being created. Even with such an optimal choice of m, Faloutsos and Christodoulakis [15] show that the false hit probability equals

$$Pfalse= (1/2)^{mopt}. \qquad (2)$$

place in the altered way, given any word w that does not belong to W , SC will still report "yes" with probability Pfalse, and demand the full scan of W .

It is easy to see that Pfalse can be made smaller by adopting the larger l (note that g is fixed has it is decided by W). in the particular, asymptotically speaking, to make sure Pfalse is at least to regular, l must be $\Omega(g)$, in which, the signature should have $\Omega(1)$ bit for every distinct word of W . Indeed, for the IR2-tree, Felipe et al. [14] adopt the value of l that is approximately equivalent to 4g in the their experiments

$$Pfalse= (1/2)^{4\ln(2)}= 0.15. \qquad (3)$$

The above result takes the heavy toll on the efficiency of all the IR2-tree. For ease, let we first guess that the query keyword set Wq had only the single keyword w (i.e., |Wq| = 1). Without loss of generality, let p be the object of the query result, and S is the set of data points that are closer to the query point q than p. in the o the r words, none of the points in the S had w in the IR text documents (otherwise, p cannot have been the final result). By Equation 4, roughly 15% of the points in the S cannot be pruned using the IR signatures, and thus, will become false hits. the also means that the NN algorithm is expected to perform at least 0.15|S| random I/Os. So far we have considered |Wq| = 1, but the discussion extends to arbitrary |Wq| in the straightforward manner. It is easy to observe (based on the Equation 4) that, in the general, the false hit probability satisfies

$$Pfalse \geq 0.15^{|Wq|}. \qquad (4)$$

When |Wq| > 1, there is another negative fact that adds to the deficiency of the IR2-tree: for the greater |Wq|, the we expected size of S increases dramatically, because fewer and fewer objects will contain all the query keywords. The effect is so severe that the number of random accesses, given by Pfalse|S|, may escalate has |Wq| grows (even with the decrease of P false). In the fact, has long has |Wq| > 1, S be the entire dataset when the user tries out an uncommon combination of keywords that does not exist in the any object. in the case, the number of random I/Os would be so prohibitive that the IR2-tree would not be able to give real time responses.

III.PROPOSED WORK

A. Merging and Distance Browsing

Since verification is the performance bottleneck, we should try to avoid it. There is the simple way to do so in the SI index: one only needs to store the coordinates of each point together with each of its appears in the inverted lists. The existence of coordinates in the inverted lists naturally motivates the creation of an R-tree on the each list indexing the points the rein. Next, we discuss how to perform keyword-based nearest neighbor search with such the combined structure.

The R-trees allow us to remedy awkwardness in the way NN queries are processed with an I-index. Recall that, to answer the query, currently we have to first get all the points carrying all the query words in the Wq by merging several lists (one for each word in the Wq). It appears to be unreasonable at this point, say p be the final result lies close to the query point q. It would be essential if we could find p very soon in the all the relevant lists so that the algorithm can terminate right away.

This would become the reality if we could browse the lists synchronously by distances has opposed to by ids. in the particular, has long as we could access the points of all lists in the ascending order of the IR distances to q (breaking ties by ids), such the p would be easily discovered has its copies in the all the lists would definitely emerge consecutively in the order. So we have to do it to keep on counting how many copies of the same point have popped up constantly, and later ended by reporting the same point once the count reaches |Wq|. At any moment, it is enough to remember only one count, because whenever the new point combines, it is secure and no need to think about the previous one. Distance browsing is easy with R-trees. In the fact, the best first algorithm is exactly designed to output data points in the ascending order of the IR distances to q. However, we must coordinate the execution of best-first on the |Wq| Rtrees to obtain the global access order. For example, at each step taking the "peek" at the next point to be returned from each tree, and output should come next globally. The algorithm is expected to work well if the query keyword set Wq is small. Wq is the large number of random accesses it performs may over when all the gains over the sequential algorithm with merging.

B. Spatial Inverted List

The spatial inverted list (SI-index) is essentially the compressed version of an I-index with embedded coordinates .Query processing with an SI index can be done either by merging or together with Rtrees in the distance browsing manner. Additionally, the compression rejects the defect of the conventional I index such that an SI-index consumes much less space.

1)The Compression Scheme

To attack the problem, let us first leave out the ids and focus on the coordinates. Even though each point had 2 coordinates, we can convert the m into only one so that gap keeping can be applied effectively. The tool needed is the space filling curve (SFC) such has Hilbert- or Z-curve. SFC converts the multidimensional point to the 1D value such that if two points are close in the original space, then their 1D value also tend to be similar. Has dimensionality have been brought to 1, gap maintaining works properly after sorting the (converted) 1D value.

Let us put the id's back into reflection. Now we have effectively deal with the two coordinates with the 2D SFC, it would be natural to think about using the 3D SFC to handle with id's. So far the space reduction is concerned; the 3D approach may not be the bad solution. The problem is it will destroy locality of the points in the original space. In particular, the changed values would no longer safeguard the spatial proximity of the points, because id's in general having nothing to do with coordinates.

IV. EXPERIMENTS

As an outcome we experimentally evaluate the practical efficiency of our solutions to NN search with key-words, and compare them against the existing methods.

a) Competitors. The proposed SI-index comes with two query algorithms based on the merging and distance browsing respectively. We will refer to the former has SI m and the o the r has SI-b. Our evaluation also covers the state-of- the -art IR2-tree,in the particular, our IR2-tree implementation is the fast variant developed in the [12], which uses longer signatures for higher levels of tree. Furthermore, we also include the method, named index file R-tree (IFR) henceforth, which, has discussed in the IFR can be regarded has an uncompressed version of SI-b.

b) Data. Our experiments are based on the real data. The dimensionality is always 2, in that each axis consisting of integers from 0 to 16383. The synthetic category had two data sets. Uniform and Skew, which differ in the distribution of data points, and in the whether there is the correlation between the spatial distribution and objects text documents. Specifically, each dataset had 1 million points. Then their locations are uniformly distributed in the Uniform, whereas in the Skew, the y follows the Zip f distribution3. For both datasets, the vocabulary had 200 words, and each word appears in the text documents of 50k points. The difference is that the association of words with points is completely random in the Uniform, while in the Skew, there is the pattern of "word-locality" points that are spatially close have almost identical text documents.

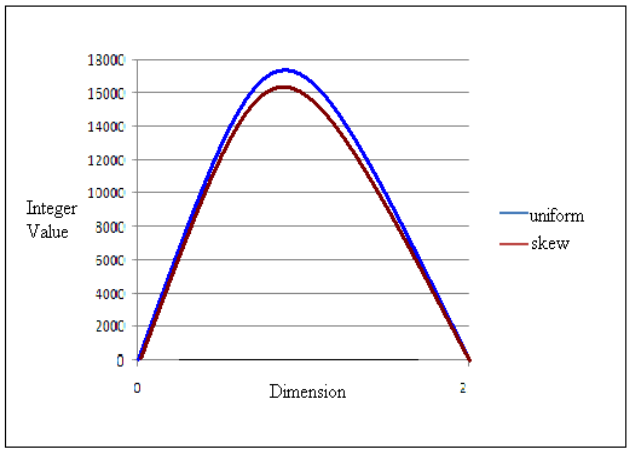

Fig 1:A Dataset of spatial inverted index(SI- Index) for uniform and skew within the integer value 0 to 16383.

V. CONCLUSION

We have seen plenty of applications calling for the search engine that is able to professionally support original forms of spatial queries that are integrated with the keyword search. The existing solution for each queries either gain prohibitive space consumption are unable to give real time answers. In the paper, we have a solution for this situation by developing an access method called the spatial inverted index (SI-index). Not only that the SI-index is fairly space economical, but it have the ability to perform keyword augmented nearest neighbor search. Furthermore, it has the SI index based on the conventional technology of inverted index; it is readily incorporable in the commercial search engine that applies on massive parallelism, on implying its immediate industrial merits.

REFERENCES

[1] Yufei Tao and Cheng Sheng,"Fast Nearest Neighbor Search with Keywords," IEEE Transactions on Knowledge and Data Engineering, Vol. 26, No. 4, 2014

[2] E. Chu, A. Doan, and J. Naughton. The Combining keyword search and forms for ad hoc querying of databases. In Proc. of ACM Management of Data (SIGMOD), 2009.

[3] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudar-shan. Keyword searching and the browsing in the databases using banks. in the Proc. of International Conference on the Data Engineering (ICDE), pages 431–440, 2002.

[4] G. R. Hjaltason and H. Samet. Distance browsing in the spatial databases. ACM Transactions on the Database Systems (TODS), 24(2):265–318, 1999.

[5] J. Lu, Y. Lu, and G. Cong. Reverse spatial and textual k nearest neighbor search. in the Proc. of ACM Management of Data (SIGMOD), pages 349–360, 2011.

[6] J. S. Vitter. Algorithms and data structures for external memory. Foundations and Trends in Theoretical Computer Science Volume 2 Issue 4, ISSN: 1551-305X.

[7] X. Cao, G. Cong, and C. S. Jensen. Retrieving the top-k prestige-based relevant on the spatial, 2010.

[8] X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi. Collective spatial keyword querying. in the Proc. of ACM Management of Data (SIG-MOD), pages 373–384, 2011.

[9] X. Cao, L. Chen, G. Cong, C. S. Jensen, Q. Qu, A. Skovsgaard, D. Wu, and M. L. Yiu. Spatial keyword querying. in the ER, pages 16–29, 2012.

[10] Y.-Y. Chen, T. Suel, and A. Markowetz. Efficient query processing in the geographic web search engines. in the Proc. of ACM Management of Data (SIGMOD), pages 277–288, 2006.

[11] B. Chazelle, J. Kilian, R. Rubinfeld, and the A. Tal. the bloomier filter: efficient data structure for static support lookup tables. In the Proc. of the Annual ACM-SIAM Symposium on the Discrete Algorithms (SODA), pages 30–39, 2004.

[12] D. Zhang, Y. M. Chee, A. Mondal, A. K. H. Tung, and M. Kitsure-gawa. Keyword search in the spatial databases: Towards searching by document. in the Proc. of International Conference on the Data Engineering(ICDE), pages 688–699, 2009.

[13] G. Cong, C. S. Jensen, and D. Wu. Efficient retrieval of top-k Most Relavent Spatial Web Objects" Proc. VLDB Endowment, vol. 2, pp. 337-348, 2009.

[14] I. D. Felipe, V. Hristidis, and N. Rishe. Keyword search on the spatial databases. in the Proc. of International Conference on the Data Engineering (ICDE), pages 656–665, 2008.

[15] I. Kamel and C. Faloutsos. Hilbert R-tree: An improved r-tree using fractals. in the Proc. of Very Large Data Bases (VLDB), pages 500–509, 1994. [16] N. Beckmann, H. Kriegel, R. Schneider, and the B. Seeger. the R*-tree: efficient and the robust access method for points and rectangles. in the Proc. of ACM Management of the Data (SIGMOD), pages 322–331, 1990.

[17] R. Hariharan, C. Li, and S. Mehrotra. For Processing spatial-keyword (SK) queries in the geographic information retrieval (GIR) systems. in the Proc. of Scientific and Statistical Database Management(SSDBM), 2007.

[18] S. Agrawal, S. Chaudhuri, and the G. Das. Dbxplorer: the system for keyword-based search over relational databases. in the Proc. of International Conference on the Data Engineering (ICDE), pages 5–16, 2002.

[19] S. Stiassny. mathematical analysis of various superimposed on the coding. Am. Doc., 11(2):155–169, 1960.

[20] V. Hristidis and Y. Papakonstantinou. Discover: Keyword search in the relational databases. in the Proc. of Very Large Data Bases (VLDB), pages 670–681, 2002.

Ms.R.Keerthika Completed Bachelor's Degree at SSM College of Engineering, India in the year 2007 and Master's Degree at Anna University of Technology Coimbatore, India in the year 2011.Registered Ph.D in 2012 at Anna University Chennai,India. She is having 7 years of experience in teaching in engineering, Currently she is working as Assistant Professor, Department of Information Technology at Karpagam College of Engineering.

Her area of interest includes Data Mining, Image Processing, Wireless sensor Networks. She is a member in professional societies like ISTE, ACM, CSTA, IACSIT, ICGST, ISOC and IRED.

Dr.C.Nalini Completed her Master's Degree at Bharathiyar University, India in the year 2000. Completed her Ph.D in 2011 at Anna University Chennai. She is having 20 years of experience in teaching in engineering, currently she is working as a Professor, Department of Information Technology at Kongu Engineering College.

Her area of interest includes Data Mining, Network security, Database Management System. She is a member in professional societies like ISTE, CSI.