IPHV8I1054X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .I
ISSN Online: 2319 – 9253
Print: 2319 – 9245

EFFICIENT SEARCH ENABLED USING RANKING BASED ON DISTANCE

[#1]CHEPURI DEEPTI - M.Tech Student
[#2] TUMMULAVENKATA SATYAVATHI - M.Tech Student
[#3]A.RAVI KUMAR- H.O.D
Dept of CSE,
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA, A.P,INDIA.

ABSTRACT: - Similarity search in multimedia databases requires an efficient support of nearest-neighbor search on a large set of high-dimensional points as a basic operation for query processing. As recent theoretical results show, state of the art approaches to nearest-neighbor search are not efficient in higher dimensions. In our new approach, we therefore pre compute the result of any nearest-neighbor search which corresponds results of page ranking and trust ranking. Page Rank is a proprietary mathematical formula (algorithm) that Google uses to calculate the importance of a particular web page/URL based on incoming links. The Page Rank algorithm assigns each web page a numeric value. That value is a particular URL's Page Rank. An extensive experimental evaluation of our tech-unique demonstrates the high efficiency for uniformly distributed as well as real data. We obtained a significant reduction of the search time compared to nearest neighbor search in the X-tree.

Keywords: - *Fast nearest-neighbor search, Information retrieval, spatial index, keyword search.*

I.INTRODUCTION

Keyword search in document performed with various approaches ranked retrieval results, clustering search results & identifying the nearest neighbor Keyword search on xml document categorized as two different approaches one is Keyword search on xml document which can be performed by ranking the searched results based on match or the answer to keyword & finding the nearest neighbor of keyword by using GST or by X path Query. In paper [2] the problem of returning clustered results for keyword search on documents the core of the semantics is the conceptually related relationship between keyword matches, which is based on the conceptual relationship between nodes in trees. Then, we propose a new clustering methodology for search results, which clusters results according to the way they match the given query.

A spatial database manages multidimensional objects (such as points, rectangles, etc.), and provides fast access to those objects based on different selection criteria. The importance of spatial databases is reflected by the convenience of modeling entities of reality in a geometric manner. For example, locations of restaurants, hotels, hospitals and so on are often represented as points in a map, while larger extents such as parks, lakes, and landscapes often as a combination of rectangles. Many functionalities of a spatial database are useful in various ways in specific contexts. For instance, in a geography information system, range search can be deployed to find all restaurants in a certain area, while nearest neighbor retrieval can discover the restaurant closest to a given address.

Today, the widespread use of search engines has made it realistic to write spatial queries in a brand new way. Conventionally, queries focus on objects' geometric properties only, such as whether a point is in a rectangle, or how close two points are from each other. We have seen some modern applications that call for the ability to select objects based on both of their geometric coordinates and their associated texts. For example, it would be fairly useful if a search engine can be used to find the nearest restaurant that offers "steak, spaghetti, and brandy" all at the same time. Note that this is not the "globally" nearest restaurant (which would have been returned by a traditional nearest neighbor query), but the nearest restaurant among only those providing all the demanded foods and drinks.

There are easy ways to support queries that combine spatial and text features. For example, for the above query, we could first fetch all the restaurants whose menus contain the set of keywords {steak, spaghetti, brandy}, and then from the retrieved restaurants, find the nearest one. Similarly, one could also do it reversely by targeting first the spatial conditions – browse all the restaurants in ascending order of their distances to the query point until encountering one whose menu has all the keywords. The major drawback of these straightforward approaches is that they will fail to provide real time answers on difficult inputs. A typical example is that the real nearest neighbor lies quite far away from the query point, while all the closer neighbors are missing at least one of the query keywords. Spatial queries with keywords have not been extensively explored. In the past years, the community has sparked enthusiasm in studying keyword search in relational databases. It is until recently that attention was diverted to multidimensional data.

The boom of Internet has given rise to an ever increasing amount of text data associated with multiple dimensions (attributes), for example, customer reviews in shopping websites (e.g., Amazon) are always associated with attributes

like price, model, and rate. A traditional OLAP data cube can be naturally extended to summarize and navigate structured data together with unstructured text data. Such a cube model is called text cube [1]. A cell in the text cube aggregates a set of documents/items with matching attribute values in a subset of dimensions. Keyword query, one of the most popular and easy-to-use ways to retrieve useful information from a collection of plain documents, is being extended to RDBMSs to retrieve information from text-rich attributes [2], [3]. Given a set of keywords, existing methods aim to find relevant items or joins of items (e.g., linked by foreign keys) that contain all or some of the keywords.

Traditional IR techniques can be used to rank documents according to the relevance. In a large text database, however, the number of relevant documents to a query could be large, and a user has to spend much time reading them. If a document is associated with attribute information, in a data cube model (a multidimensional space induced by the attributes), e.g., the text cube, a cell aggregates the documents with matching values in a subset of attributes. Such a collection of documents is associated with each cell, corresponding to an object that can be directly recommended to the user for the given query. This paper studies the problem of keyword-based top-k search in text cube, i.e., given a keyword query, find the top-k most relevant cells in a text cube. When users want to retrieve information from a text cube using keyword question, believe that relevant cells, rather than relevant documents, are preferred as the answers, because: 1) relevant cells are easy for users to browse; and 2) relevant cells provide users insights about the relationship between the values of relational attributes and the text data.

II. RELATED WORK

In the paper ‘fast nearest neighbor search with keywords’, there are methods like spatial index, inverted index, nearest neighbor search. The first method spatial index is used for creating indices because there is huge amount of data need to be stored for searching that data stored in the form of xml documents. If the data storage created in the form of indices then space required is less also time needed for searching the keyword is less.

Second method is inverted index. The inverted index data structure is a central component of a typical search engine indexing algorithm. A goal of a search engine performance is to optimize the speed of the query: find the documents where word occurs. Once an index is developed, which provisions lists of words per document; it is next inverted to develop an inverted index. Querying the index would require sequential iteration through each document and to each word to verify a matching document. The time memory and processing property to execute such a query are not always theoretically realistic. Instead of listing the words per article in the index, the inverted index data structure is developed which lists the documents per word. The inverted index produced, the query can now be determined by jumping to the word id in the inverted index. These were effectively inverted indexes with a small amount of supplementary explanation that required a implausible amount of attempt to produce.

Third method is nearest neighbor search. Nearest neighbor search (NNS), also identified as closeness search, parallel search is an optimization problem for finding closest points in metric spaces. In the paper ‘Efficient Keyword-Based Search for Top-K Cells in Text Cube’ methods used are inverted-index one-scan, document sorted-scan, bottom-up dynamic programming, and search-space ordering. In the top k cells, there is a searching of nearest key to the query. Cubes forms clusters of single unique group which shows its identity. Method like inverted index used for giving index rather than providing whole data which can be space consuming.

III. IMPLEMENTATION

In this paper, we suggest a new solution to sequential nearest neighbor search which is based on page ranking and trust based ranking instead of indexing the data. The solution space may be characterized by a complete and overlap-free partitioning of the data space into cells, each containing exactly one data point. Each cell consists of all potential query points which have the corresponding data point as a nearest neighbor. The cells therefore correspond to the d-dimensional Voronoi cells [PS 85]. Deter-mining the nearest neighbor of a query point now becomes equivalent to determining the Voronoi cell in which the query point is located. Since the Voronoi cells may be rather complex high-dimensional polyhedral which require too much disk space when stored explicitly, we approximate the cells by minimum bounding (hyper-) rectangles and store them in a multidimensional index structure such as the X-tree [BKK 96]. The nearest neighbor query now be-comes a simple point query which can be processed efficiently using the multidimensional index. In order to obtain a good approximation quality for high-dimensional cells, we additionally introduce a new decomposition technique for high-dimensional spatial objects.

Page Rank

We assume page A has pages T1...Tn which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. We usually set d to 0.85. There are more details about d in the next section. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:

$$PR(A) = (1-d) + d\,(PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))$$

Note that the PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one.

PageRank or PR(A) can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the

IPHV8I1054X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .I
ISSN Online: 2319 – 9253
Print: 2319 – 9245

normalized link matrix of the web.
An approximation determined from google.

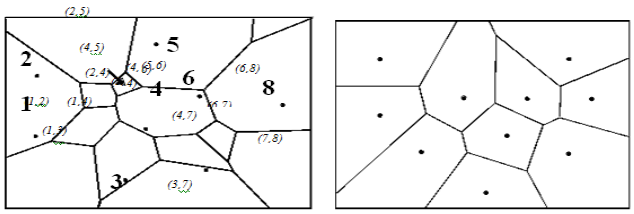
1. **PR(Tn)** - Each page has a notion of its own self-importance. That's "PR(T1)" for the first page in the web all the way up to "PR(Tn)" for the last page
2. **C(Tn)** - Each page spreads its vote out evenly amongst all of its outgoing links. The count, or number, of outgoing links for page 1 is "C(T1)", "C(Tn)" for page n, and so on for all pages.
3. **PR(Tn)/C(Tn)** - so if our page (page A) has a backlink from page "n" the share of the vote page A will get is "PR(Tn)/C(Tn)"
4. **d(...** - All these fractions of votes are added together but, to stop the other pages having too much influence, this total vote is "damped down" by multiplying it by 0.85 (the factor "d")
5. **(1 - d)** - The (1 – d) bit at the beginning is a bit of probability math magic so the "*sum of all web pages' PageRanks will be one*": it adds in the bit lost by the **d(...** It also means that if a page has no links to it (no backlinks) even then it will still get a small PR of 0.15 (i.e. 1 – 0.85).

Approximating the Solution Space

Our new approach to solving the nearest neighbor problem is based on ranking the solution space. Pre-calculating the solution space means determining the Voronoi diagram (cf. figure 1a) of the data points in the database. In the following, we recall the definition of Voronoi cells as provided in [Roo 91].

Definition 1.(Voronoi Cell, Voronoi Diagram)

Let DB be a database of points. For any subset $A \epsilon DB$ of size $m := |A|, 1 \leq m < n$, and a given distance function $d: \mathcal{R}^d \times \mathcal{R}^d \rightarrow \mathcal{R}_0^+$, the order-m Voronoi Cell of A is defined as
Voronoi Cell(A):=
$\{x \in \mathcal{R}^d \mid \forall (p_i \in A)\, \forall (P_j \in DB \backslash A): d(x, P_i) \leq d(x, P_j)\}$.
The order-m Voronoi diagram of DB is defined as
Voronoi Diagram$_m$ (DB):=
$(VoronoiCell(A) | A \subset DB \wedge |A| = m\}$.
Note that we are primarily inserted in the nearest neighbor query point, and that we assume the Voronoi cells to be bounded by the data space(DS).Therefore, in the following we only have to consider bounded Voronoi Cells of order 1, which are also called NN-cells (cf.figure 1b)



a. Voroni Diagram of order 2 (cf [PS 85]) b. NN-Diagram

Figure 1: Voronoi diagram and NN-diagram

Definition 2. (NN-cell, NN-Diagram)

For any point $P \in DB$ and a given distance function , $d: \mathcal{R}^d \times \mathcal{R}^d \rightarrow \mathcal{R}_0^+$ the NN-cell of P is defined as
NN-cell(P):=
$\{x \in DS | \forall (p' \in DB \backslash \{P\}): d(x, P) \leq d(x, P')\}$.
The NN-Diagram of a database of points DB is defined as
NN-Diagram (DB):=$\{NN - Cell(P) | P \in DB\}$.
According to Definition 2, the sum of volumes of all NN-Cells is the volume of the data space (cf .figure 1b):

$$\sum_{i=1}^{N} vol(NN - Cell_i) = vol(DS)\,.$$

If we are able to efficiently determine the NN-cells, to explicitly store them, and to directly find the NN-Cell which contains a given query point, then the costly nearest neighbor query could be executed by one access to the corresponding NN-cell. In general, however, determining the NN-cells is rather time consuming and requires (at least) $\Omega(NlogN)$ for $d \epsilon \{1,2\}$ and $\Omega(N^{\lceil d/2 \rceil})$ for d≥ 3 in the worst case [PS 85] for an Euclidean distance function. In addition, the space requirements for the NN-cells (number of k faces of the NN-diagram) are
$O(N^{min\,(d+1-k, \lceil d/2 \rceil)})$ for $0 \leq k \leq d-1$
in the worst case[Ede 87], making it impossible to store them explicitly. For a practicable solution, it is therefore necessary to use approximations of the NN-cells, which is a well-known technique that has been successfully used for improving the query processing in the context of geographical databases [BKS 93]. In principle, any approximation such as (hyper-) rectangles, rotated (hyper-)rectangles, spheres, ellipsoids, etc. may be employed. In our application, we use an approximation of the NN-cells by minimum bounding (hyper-)rectangles and store them in a multidimensional index structure such as the X-tree[BKK 96]. The nearest neighbor query can then be executed by a simple and very efficient point query on this index. In the following,,we define the approximation of the NN-cells.

Definition 3. (MBR-approximation of NN-cells)

The MBR approximation APPR$_{MBR}$ of a nearest neighbor cell (NNC) is the minimum bounding (hyper-) rectangle
MBR=(l_1,h_1,.....,l_d,h_d) of NNC, i.e. for i=1,.....,d:
$l_i = \min\{P_i | P \in NNC\}$ and $h_i = \max\{p_i | p \in NNC\}$.

Let us now consider examples of the NN-cells and their MBR-approximations for a number of different data distributions. Figure 2a and b show the NN-diagram and the corresponding approximation diagram for two independent uniformly distributed dimensions, figure 2c and d show the two diagrams for a regular multidimensional uniform distribution, and figure 2e and f show the diagrams for a sparse distribution. A uniform distribution is usually generated by using a random number generator to produce the data values for each of the dimensions independently. This

generation process produces a data set which –projected onto each of the dimension axes – provides a uniform distribution. It does not mean, however, that the data is distributed uniformly in multidimensional space, i.e. for a partitioning of the data space into cells of equal size that each

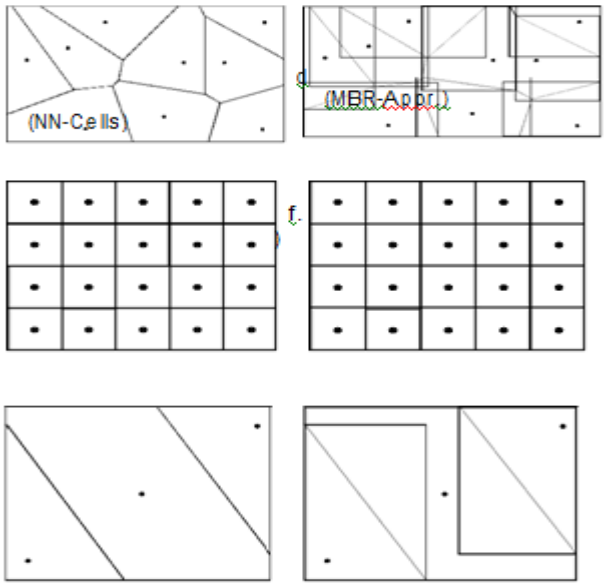


Figure 2: NN-cells and their MBR-approximations

of the cells contains an equal number of data points. A distribution which fulfills the latter requirement is called a multidimensional distribution.

Note that the regular multidimensional uniform distribution corresponds to the best case for our approach and the sparse distribution corresponds to the worst case: In case of the regular multidimensional uniform distribution, the MBR approximations are identical with the NN-cells, which means that the approximations do not have any over-lap and therefore a point query on the index accesses only one page. In contrast, in case of the sparse distribution al-most all approximations are identical with the DS which means that the approximations completely overlap and a point query on the index accesses most data pages.

IV.CONCLUSION

In this paper an optimized page rank algorithm based on normalization technique has been proposed. In this proposed scheme the page rank of all web pages are being normalized by using a mean value factor, which reduces the time complexity of the conventional page rank algorithm. Comparative study of the computational characteristics of the proposed scheme with the previous works signifies that the proposed page rank algorithm is a better alternative to the previously introduced page rank algorithm as seen from the prospect of time complexity and the computational savings. In the future, the researchers can plan to explore more on the page rank algorithm based on damping factor to enhance the performance of the proposed scheme. We finally showed in an experimental evaluation that our technique is efficient for various kinds of data and clearly outperforms the state of the art nearest-neighbor algorithms.

REFERENCES

[1] BHARAT, K., AND HENZINGER, M. Improved Algorithms for Topic Distillation in a Hyperlinked Environment. In Proceedings of ACM SIGIR'98 (Melbourne, Australia, 1998).

[2] A.K. Singh, Ravi Kumar and Alex Goh Kwang Leng "Efficient Algorithm for Handling Dangling Pages using Hypothetical node".

[3] Wenpu Xing and Ali Ghorbani, "Weighted PageRank Algorithm" Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR'04) 2004 IEEE.

[4] S. Al-Saffar and G. Heileman, Experimental bounds on the usefulness of personalized and topic-sensitive pagerank, International Conference on Web Intelligence, pp. 671-675, 2007.

[5] A. Rungsawang, K. Puntumapon, B. Manaskasemsak, Un-biasing the link farm effect in pagerank computation, 21th International Conference on Advanced Networking and Applications, pp. 924-931, 2007.

[6] Cooper, C. Frieze A., "A general model of Web graphs", In ESA, 2001, pp. 500-511. CERN Common Log Format, http://www.w3.org/Daemon/User/Config/Logging.html# common-log file-format.

[7] CALADO, P., RIBEIRO-NETO, B., ZIVIANI, N., MOURA, E., AND SILVA, I. Local Versus Global Link Information in the Web. ACM Transactions on Information Systems 21, 1 (January 2003), 42–63.

[8] CRASWELL, N., CRIMMINS, F., HAWKING, D., AND MOFFAT, A. Performance and cost tradeoffs in web search. In ADC'04 (Dunedin, New Zealand, January 2004), pp. 161–170. http://es.csiro.au/pubs/craswell adc04.pdf.

[9] A. L. Barabasi and R. Albert, Emergence of scaling in random networks, Science Magazine, Vol. 286. no. 5439, pp. 509-512, 1999.

[10] Kleinberg, J. M. Authoritative sources in a hyperlinked environment, Journal of the ACM, Vol.46 (5). (Sept. 1999). 604-632.

[11] Lerman, K., Getoor, L., Minton, S., and Knoblock, C.Using the Structure of Web Sites for Automatic Segmentation of Tables. SIGMOD (2004) 119-130.

[12] Eiron, N. McCurley, K., and Tomlin, J. Ranking the web frontier. Proceedings of the international conference on World Wide Web, (WWW'04). Pp.309-318, 2004.

[13] C. Guo and Z. Liang, An improved BA model based on the pagerank algorithm, 4th WiCOM International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1-4, 2008.

[14] NG, A. Y., ZHENG, A. X., AND JORDAN, M. I. Link analysis, eigenvectors, and stability. In Proceedings of IJCAI'01 (Seattle, USA, 2001), ACM Press.

[15] Deng Cai, Shipeng Yu, and et al. Block-based Web Search. In Proceedings of ACM SIGIR'04, July 25-29, 2004, Sheffield, South Yorkshire, UK. 465-463.

[16] HAVELIWALA, T. H. Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. In IEEE Transactions on Knowledge and Data Engineering (July 2003).

[17] Ziv Bar-Yossef and Sridhar Rajagopalan. Template Detection via Data Mining and its Applications. In: Proceedings of WWW2002, May 7- 11, 2002, Honolulu, Hawaii, USA. 580-591.

[18] L. Page, S. Brin, R. Motwani, and T. Winograd. The Pagerank citation ranking: Bring.

[19] [Roo 91] Roos T.: 'Dynamic Voronoi Diagrams', Ph.D. Thesis, University of Wiirzburg, Germany,1991.

[20] [PS 85] Preparata F. P., Shamos M.I.: 'Computational Geometry: An Introduction', Springer, 1985.

AUTHORS PROFILE:

[1]. CHEPURI DEEPTI - Pursuing M.Tech ,Dept of CSE, JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA, A.P,INDIA.

[2]. TUMMULAVENKATA SATYAVATHI, Pursuing M.Tech ,Dept of CSE, JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA, A.P,INDIA.

[3]. A.RAVI KUMAR working as H.O.D, Dept of CSE, JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA, A.P,INDIA.