HYBRID GEO-TEXTUAL INDEX STRUCTURE FOR SPATIAL RANGE KEYWORD SEARCH

Su Nandar Aung[1] and Myint Mint Sein[2]

[1]University of Computer Studies, Yangon, Myanmar
[2] Research and Development Department, University of Computer Studies, Yangon, Myanmar

ABSTRACT

Spatial database are becoming more and more popular in recent years. There is more and more commercial and research interest in location-based search from spatial database. Spatial keyword search has been well studied for years due to its importance to commercial search engines. Specially, a spatial keyword query takes a user location and user-supplied keywords as arguments and returns objects that are spatially and textually relevant to these arguments. Geo-textual index play an important role in spatial keyword querying. A number of geo-textual indices have been proposed in recent years which mainly combine the R-tree and its variants and the inverted file. This paper propose new index structure that combine K-d tree and inverted file for spatial range keyword query which are based on the most spatial and textual relevance to query point within given range.

KEYWORDS

Combination scheme, Scalable index structure, Spatial Keyword Queries, Boolean Range Keyword Queries, Range Keyword Search.

1. INTRODUCTION

Spatial data are data that have a location (spatial) and mainly required for Geographic Information Systems (GIS) whose information is related to geographic locations. GIS model supports spatial data types, such as point, line and polygon. A geospatial collections increase in size, the demand of efficient processing of spatial queries with text constraints becomes more prevalent. Spatial keyword search is an important tool in exploring useful information from a spatial database and has been studied for years. The query consists of a spatial location, a set of keywords and a parameter k and the answer is a list of objects ranked according to a combination of their distance to the query point and the relevance of their text description to the query keyword. *The spatial relevance* is measured by the distance between the location associated with the candidate document to the query location, and *the textual relevance* is said to be textually relevant to a query if object contains queried keywords. [1]

A *scalable index structure* should satisfy the following requirements:

i. The index shows effective storage utilization.
ii. The index answers queries efficiently.

DOI : 10.5121/cseij.2014.4603

Many index structures that have been proposed in recent years mainly use R-tree and then combine with inverted file, namely the family of IR-tree [4, 5, 6, 7, 8, 9, and 10]. All use R-tree for spatial (latitude/longitude) index and inverted file for textual index. They all created hybrid index structure according to the combination schemes: (1) *Text first loose combination scheme*, employs the inverted as the top-level index and then arrange the postings in each inverted list in a spatial structure. (2) *Spatial-first loose combination scheme* employs the spatial index as the top-level index and its leaf nodes contain inverted files or bitmaps for the text information of objects contained in the nodes. (3) *Tight combination index* combines a spatial and a text index tightly such that both types of information can be used to prune the search space simultaneously during query processing.

The construction of an efficient index structure should take into account overlaps between nodes and coverage of a node. Minimization of a node coverage leads to more precise searching within the tree and minimization of the overlap between nodes reduces the number of paths tested in the tree during a search that can reduce search time. As the data objects in the R-tree can be overlapping and covering each other, the search process in the R-tree might suffer from unnecessary node visits and higher IO cost [16]. Moreover, the IR-trees suffer from high update cost. Each node has to maintain an inverted index for all the keywords of documents associated with this node's MBR. When a node is full and split into two new nodes, all the textual information in the node has to be re-organized [1]. As the R-tree need to reorganized, it suffers from higher CUP costs.

This paper includes the following contributions:

1) The main contribution is to create index structure that combine K-d tree and inverted file for efficiently process spatial keyword queries within minimum time.
2) Range keyword search algorithm is developed using the proposed index structure to efficiently answer Boolean range queries and to explore useful and exact information that user required.
3) The own dataset is created for Yangon (Myanmar) region which contains latitude, longitude, name, description and category type of each object.

2. SPATIAL KEYWORD QUERIES

Standard spatial keyword queries involve different conditions on the spatial and textual aspects of places. In spatial databases, the arguably most fundamental queries are range queries and k nearest neighbour queries. In text retrieval, queries may be Boolean, requiring results to contain the query keywords, or ranking-based, returning the k places that rank the highest according to a text similarity function. [3]

Three types of spatial keyword queries are receiving particular attention. The *Boolean range query* $q = (\rho, \psi)$ where ρ is a spatial region and ψ is a set of keywords, returns all places that are located in region ρ and that contain all the keywords in ψ. Variations of this query may rank the qualifying places. The *Boolean kNN query* $q=(\lambda, \psi, k)$ takes three arguments, where λ is a point location, ψ is as above, and k is the number of places to return. The result consists of up to k places, each of which contains all the keywords in ψ, ranked in increasing spatial distance from λ. Next, the *top-k range query* $q = (\rho, \psi, k)$ where ρ, ψ, and k are as above, returns up to k places that are located in the query region ρ, now ranked according to their text relevance to ψ. Finally, the *top-k kNN query* takes the same arguments as the Boolean *k*NN query. It retrieves k objects ranked according to a score that takes into consideration spatial proximity and text relevance.

Among these queries, the latter two ones that perform textual ranking are the most similar to standard web querying, and the last one is the one that is most interesting and novel, as it integrates the spatial and textual aspects in the ranking. This query, also called the *top-k spatial keyword query.*

3. PROBLEM STATEMENT AND PROPOSED SYSTEM

Let D is a spatial database that contains D= $\{o_1,o_2,o_3\ldots..,o_n\}$ such that every object o in D has many attributes $<o_{id}, o_l, o_d>$ where o_{id} is an identifier of an object, o_l is a spatial location that contain latitude and longitude and o_d is an text document of each object for keyword querying.

***Boolean Range Keyword Queries*: ** Let $q=<q_k, q_r>$ be a Boolean keyword range query where q_k is user required keywords $w_1\ldots..w_m$ and q_r is the user desired range (km). A query q return all objects in D that contain all keywords $q_k=\{w_1, w_2,\ldots., w_m\}$ and belong to the range q_r .

$$\text{Ans}(q)=\begin{cases} o \in q_r \text{ , } o \textit{ is contained in } q_r \\ o \in q_k \text{ , } \forall_w \in q_{k'} \; w \in o_d \end{cases}$$

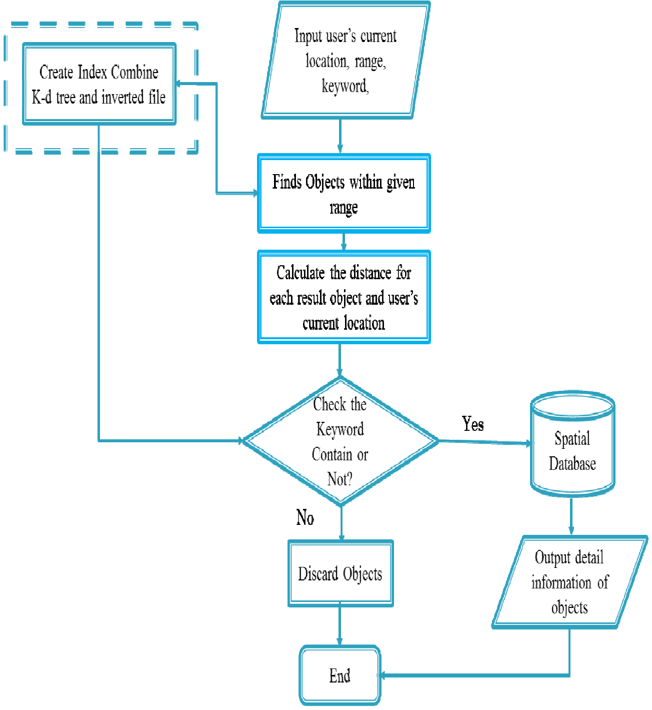

Create Index Combine K-d tree and inverted file
Input user's current location, range, keyword,
Finds Objects within given range
Calculate the distance for each result object and user's current location
Check the Keyword Contain or Not?
Yes
Spatial Database
No
Discard Objects
Output detail information of objects
End

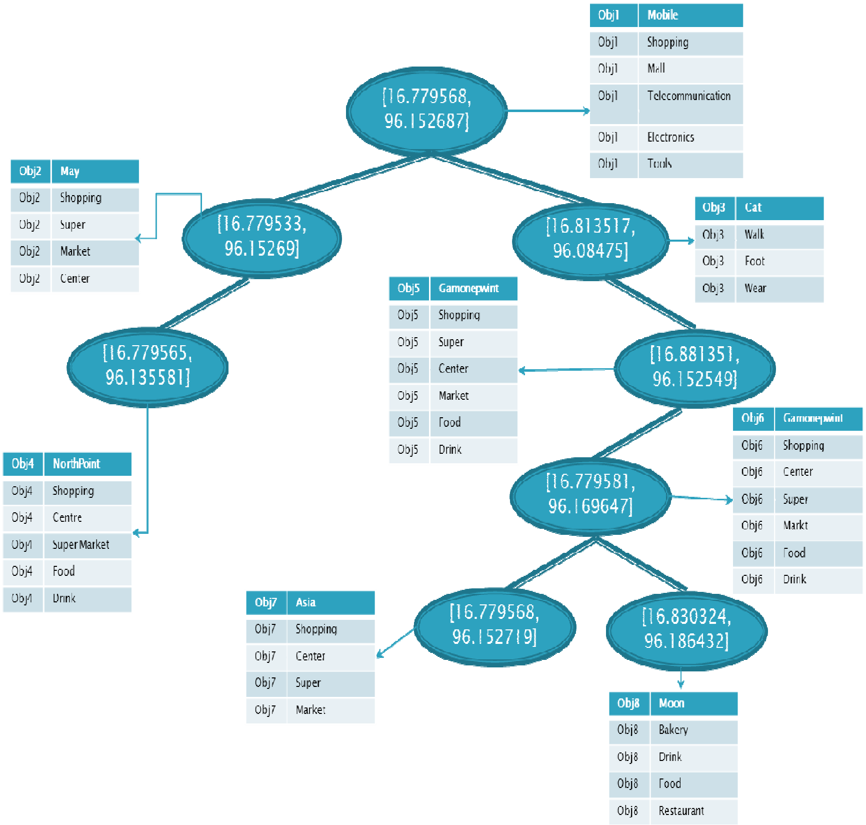
Figure1. Proposed System Flow Diagram

The proposed system creates hybrid geo-textual index structure that integrates location index and text index to process spatial keyword queries efficiently. In this proposed system K-d tree loosely combined with inverted file. K-d tree is used for spatial queries and inverted file is used for keywords information that is the most efficient index for text information retrieval. For each node of K-d tree, an inverted file is created for indexing the text components of objects contained in the node. As K-d trees represent a disjoint partition, the proposed system can't cause more IO costs and also K-d trees don't need to rebalance the textual information so the proposed can reduce update cost (CPU costs).

Most geo-textual indices use the inverted file for text indexing. Inverted file can be used to check the query keywords contain or not. K-d tree structure is known as point indexing structures as it is designed to index data objects which are points in a multi-dimensional space. It can be used efficiently for range queries and nearest neighbour queries.

Table1. Example Dataset

id	Latitude	Longitude	Keywords
Obj1	16.779568	96.152687	Mobile, Shopping, Mall, Telecommunication Electronics, Tools
Obj2	16.779533	96.15269	May, Shopping, Center, Super, Market
Obj3	16.813517	96.08475	Cat, Walk, Foot, Wear
Obj4	16.779565	96.135581	NorthPoint, Shopping, Center, Super, Market, Food, Drink
Obj5	16.881351	96.152549	Gamonpwint, Shopping, Center, Super, Market, Food, Drink
Obj6	16.779581	96.169647	Gamonpwint, Shopping, Center, Super, Market, Food, Drink
Obj7	16.779568	96.152719	Asia, Shopping, Center, Super, Market
Obj8	16.830324	96.186432	Moon, Bakery, Food, Drink



Obj1	Mobile
Obj1	Shopping
Obj1	Mall
Obj1	Telecommunication
Obj1	Electronics
Obj1	Tools

Obj2	May
Obj2	Shopping
Obj2	Super
Obj2	Market
Obj2	Center

Obj3	Cat
Obj3	Walk
Obj3	Foot
Obj3	Wear

Obj5	Gamonepwint
Obj5	Shopping
Obj5	Super
Obj5	Center
Obj5	Market
Obj5	Food
Obj5	Drink

Obj4	NorthPoint
Obj4	Shopping
Obj4	Centre
Obj4	Super Market
Obj4	Food
Obj4	Drink

Obj6	Gamonepwint
Obj6	Shopping
Obj6	Center
Obj6	Super
Obj6	Markt
Obj6	Food
Obj6	Drink

Obj7	Asia
Obj7	Shopping
Obj7	Center
Obj7	Super
Obj7	Market

Obj8	Moon
Obj8	Bakery
Obj8	Drink
Obj8	Food
Obj8	Restaurant

Figure2. Example Proposed Index Structure

4. RANGE KEYWORD SEARCH ALGORITHM

Algorithm1 is the proposed range keyword search procedure. The procedure RANGEKEYWORDSEARCH returns all points ‘p’ such that d (q,p) ≤ r and $keyword \in \sum_{i=1}^{n} p.word_i$. Use Boolean model to check required keywords contain or not in inverted file of each point such that,

$$result = \begin{cases} 1, & if\ keyword\ contain in inverted\ file \\ & o\ , otherwise \end{cases}$$

The procedure COMPUTEBOUNDINGBOXES ( ) calculates the bounding boxes lBB and rBB for the left and for the right sub tree, respectively. The procedure INTERSECTS (…) tells if the bounding box BB intersects with the region that satisfies the distance constraints. If the intersection is non-empty, the sub tree to be explored. The DISTANCE (…) procedure calculates the distance between two points using Euclidean distance $d(q,p) = \sqrt{(q_{lat} - p_{lat})^2 + (q_{lon} - p_{lon})^2}$.

Algorithm1. Range Keyword Search in Hybrid Index Structure

```
Input: user’s required keyword, K-d tree, query point, range, Max/Min BB
pq: priority queue
RANGEKEYWORDSEARCH (keyword,T,BB,q,r)
    if T=leaf then return
    p<−T.key;   i<−T.discr;
    distance <− DISTANCE(q,p);
    if distance ≤ r and $keyword \in \sum_{i=1}^{n} p.word_i$ then
        pq.PUSH (p,distance);
    COMPUTEBOUNDINGBOXES (lBB,rBB,p[i],i)
    if INTERSECTS (lBB,q,r) then
        RANGEKEYWORDSEARCH (keyword, T.left, lBB, c, radius)
    if INTERSECTS (rBB,q,r) then
        RANGEKEYWORDSEARCH(keyword, T.right, rBB, c, radius)
```

5. EXPERIMENTAL RESULT

As an example, we use {16.7858/96.14976} as a user’s current location and find all objects that contain the required keyword “bank” within 1,2,3,4 and 5 kilometres respectively. For this sample query, the proposed system reply all objects with the distance between the user’s current location and each result object that contain required keyword within given range. Table 2 and figure3 compare the searching time (second) between using proposed index structure and without using proposed index. Depending on the desired range (km), searching time is varied. Searching time using index structure is faster than without using index about 100-times in second. Figure4 shows the index construction time (second) depending on the size of datasets.

Table 2. Searching Time

Range	Using Proposed Index	Without Using Index (Direct from Database)
1-km	0.000337396	0.048688981
2-km	0.000544804	0.064471388
3-km	0.000772731	0.088644777
4-km	0.000815067	0.097669104
5-km	0.000852699	0.115346145

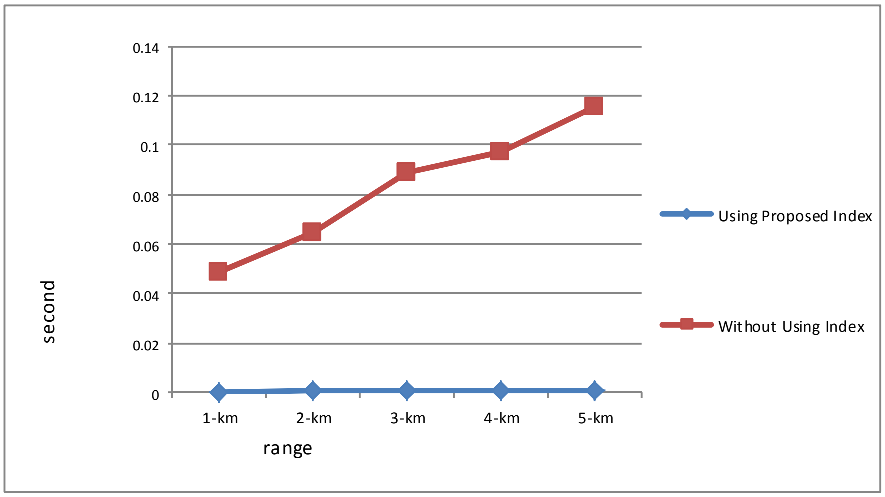


Figure3. Searching Time in range keyword search

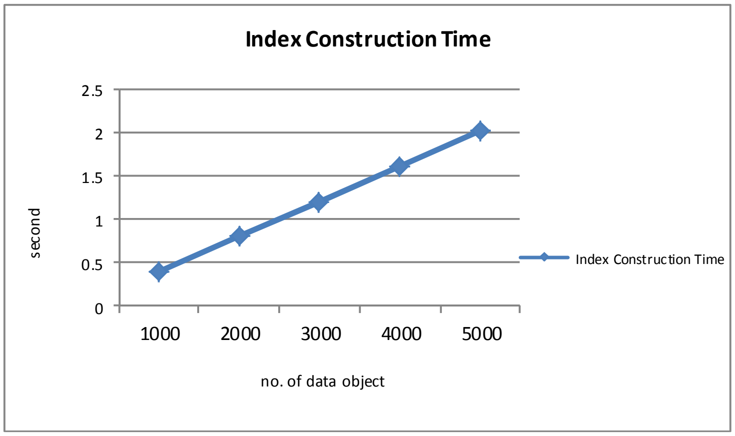


Figure4. Index Construction Time

3. CONCLUSIONS

This paper presented hybrid index structure for range keyword query searching with minimum IO costs and CPU costs. This index structure can avoid searching in overlapping area. So it can reduce searching time in overlap area. Moreover, it can't cause node overflow, so it doesn't need to re-organize the textual data and spatial data. Many Further extensions can be considered for efficient hybrid index structure for spatial database. As a further extension, we'll add an efficient nearest neighbour keyword search and approximate keyword search in this proposed index structure.

REFERENCES

[1] D. Zhang, K.L. Tan, Anthony K.H. Tung, (2013), "Scalable Top-K Spatial Keyword Search", EDBT/ICDT'13 March 18-22.

[2] L. Chen, G. Cong, C. S. Jensen, and D. Wu, (2013), "Spatial Keyword Query Processing: An Experimental Evaluation", in proceddings of the VLDB Endowment, Vol.6, No.3.

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

[4] J. B. Rocha-Junior, O. Gkorgkas, S. Jonassen, and K. Nørv°ag, (2011), "Efficient processing of top-k spatial keyword queries", in SSTD, pages 205–222.

[5] Z. Li, K. C. K. Lee, B. Zheng, W.-C. Lee, D. L. Lee, and X. Wang, (2011), "Ir-tree: An efficient index for geographic document search", IEEE TKDE, 23(4):585–599.

[6] A. Cary, O. Wolfson, and N. Rishe, (2010), "Efficient and scalable method for processing top-k spatial Boolean queries", in SSDBM, pages 87–95.

[7] G. Cong, C. S. Jensen, and D. Wu, (2009), "Efficient retrieval of the top-k most relevant spatial web objects", PVLDB, 2(1):337–348.

[8] R. G¨obel, A. Henrich, R. Niemann, and D. Blank, (2009), "A hybrid index structure for geo-textual searches", in CIKM, pages 1625–1628.

[9] I. D. Felipe, V. Hristidis, and N. Rishe, (2008), "Keyword search on spatial databases", in ICDE, pages 656–665.

[10] R. Hariharan, B. Hore, C. Li, and S. Mehrotra, (2007), "Processing spatial-keyword (sk) queries in geographic information retrieval (gir) systems", in SSDBM, page 16.

[11] Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma, (2005), "Hybrid index structures for location-based web search", in CIKM, pages 155–162.

[12] H.M. Kakde, (2005), "Range Searching using Kd Tree".

[13] A. Guttman, (1984), "R-trees: A dynamic index structure for spatial searching", in SIGMOD, pages 47–57.

[14] B.C. Ooi, R. Sacks-Davis, J.Han, "Indexing in Spatial Databases".

[15] X.Cao, G.Cong, Christian S. Jensen, Jun.J. Ng, Beng C.Ooi, N.T. Phan, D. Wu, "SWROS: A System for the Efficient Retrieval of Relevant Spatial Web Objects".

[16] . Theodoridis, T. Sellis, "Optimization Issues in R-tree Construction", Technical Report KDBSLAB-TR-93-08.

AUTHORS

Myint Myint Sein received the Ph.D in Electrical Engineering from the Graduate School of Engineering, Osaka City University, Osaka, Japan in 2001. She is presently serving as a professor in the Research and Development Department, University of Computer Studies, Yangon, Myanmar since 2005. Her research interests are Pattern Recognition, Image Processing, Soft computing, 3D reconstruction, 3D Image Retrieval and GIS Applications.

Su Nandar Aung received her B.C.Sc (Hons:) degree from the University of Computer Studies, Mandalay, Myanmar in 2004 and M.C.Sc degree from the University of Computer Studies, Yangon, Myanmar in 2009. She worked as a tutor in Computer University (Pinlon) from 2008 to current. She is currently a Ph.D student at the University of Computer Studies, Yangon, Myanmar. Her research interests currently include Geographic Information System and Spatial Database.