International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 5, Issue 6, November - December 2016 **ISSN 2278-6856**

Efficient Spatial Database Keyword Search

S.Sajida[1], S.Ramakrishna[2]

[1] Research Scholar, Dept. of Computer Science, Sri Venkateswara University, Tirupati

[2]Professor, Dept.of Computer science Sri Venkateswara University, Tirupati

Abstract

Generally, Spatial Database objects such as Universities, Cities, Hotels and Temples and so on are associated with corresponding keywords. In order to execute services such as University Admissions, Bank services, purchasing, Courier services, research and so on. Already existing spatial keyword search algorithm is called "Keyword Cover" which is completely based on minimum inter-object distance during spatial query execution for finding query objects. Nowadays, the Keyword rating plays an important role in spatial objects evaluation for taking best decisions in many business situations. Present study proposes a fast scalable Keyword search algorithm for Efficient Spatial Database keyword search. New method uses both inter-objects distance as well as Keyword Rating.

Index terms: Spatial Database, Keyword rating, Advanced spatial database, search, query tree

I.INTRODUCTION

Nowadays, Spatial Keywords search problem is gaining its momentum with high speed to achieve fast query answering results. General applications of spatial keyword search are:

1) Mobile computing
2) Weather Simulation
3) Network Maintenance
4) Global positioning system
5) Finding location of Spatial objects
6) Satellite image management
7) Digital maps controlling System

Spatial database represents spatial tuples and each tuple is used to represent one spatial object. Each spatial object is associated with one Keyword that indicates the function of a spatial object. Spatial keywords search problem finds spatial query objects for the given spatial query based on keywords of objects with some desirable spatial relationships among the spatial objects. In many spatial database applications it is common that users are often expressed their requirements in the form of queries that contains

multiple keywords. for example: A students wishes to select University with 'A' grade, with Hostel Accommodation, Job Placement and which is nearer to India. As another example, real estate web sites allow users to search for properties with specific keywords in their description and rank them according to their distance from a specified location. We call such queries spatial keyword queries. A spatial keyword query consists of a query area and a set of keywords. The answer is a list of objects ranked according to a combination of their distance to the query area and the relevance of their text description to the query keywords, where objects are ranked by distance and keywords are applied as a conjunctive filter to eliminate objects that do not contain them [1].

In the literature, many spatial keywords search problems have been studied because of the importance of spatial keyword search one existing spatial keyword for retrieving search multiple objects against the spatial query is called "Keyword Cover". Already existing spatial keyword search problem is known as m-closest keyword (mck) query search.

Present study studies a more generalized version of mck keyword query search and this new algorithm is called "Efficient Spatial Database Keyword Search and it considers both the measures inter-object distance as well as keyword rating of objects. Nowadays in many real time situations many business services are taking place based on keyword rating in addition to the inter object distance of spatial objects. For example, Amazon, Flip cart, jabong, Zagat are some of the examples for services based on rating. Some of the possible spatial object rating scales is: 1..6 for engineering colleges, A..Z for railway stations, 1-5 for Hotels, 1..20 for bus stations, 1-100 for universities, 1..1000 for poverty specifications, 1..70 for share details, 1.100 for company ratings, 1-10 for Airports, 1..100 for temples, 1-50 for cities and so on.

Proposed algorithm uses a new indexing method for indexing the spatial database objects. The new method uses an R*-tree indexing structure. The R-tree indexing data structure is a height-balanced multi-way tree and each node of the tree corresponds to a disk page in the secondary memory. The root is at level h-1, where h is the height of the tree, and the leaf.EachR- tree node except from the root always should contain at least a number of entries, called minimum R-tree least a number of entries, called minimum degree of R-tree node. This new indexing method is named as KRR*-tree.n-number of KRR*-tree are constructed corresponding to n-number of distinct keywords. Based on the given number of query keywords the corresponding KRR*-trees are combined for obtaining query results. During the processing of combining KRR*-

trees, nodes from different KRR*-trees are combined from top to bottom level by level of trees. In this process candidate keywords are generated at each level. Here main task is to find candidate keyword cover that produces high score. For simplicity assume that each spatial object is associated with a single keyword. a spatial object is identified with the following form:<Sno,X,Y,Keyword,rating> where X,Y define the location of the spatial objects in a two dimensional geographical space.A mathematical model is constructed for computing Score that is based on both inter-object distance and also keyword rating measures. it is a linear interpolation function of distance and rating measures. For obtaining accurate results this measures are normalized.

Obj is a set of spatial objects

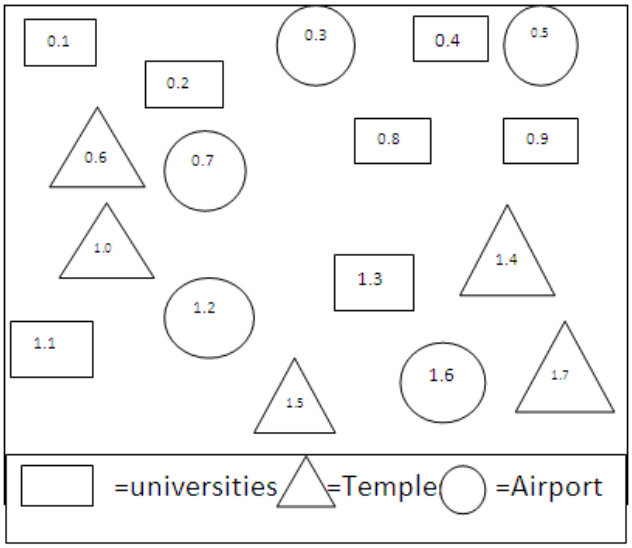
$$Diam(Obj)=\max_{ai,aj \in 0} dist(o_i, o_j).$$

$$Obj.Score=Score(A, B)$$

$$=\alpha\left(1-\frac{A}{\max\_dist}\right)+(1-\alpha)\frac{B}{\max\_rating}$$

$$A=diam(Obj).$$

$$B=\min_{o\in O}(o.rating),$$



minimum keyword rating of spatial objects in Obj and $\alpha(0 \leq \propto \leq 1)$ is an application specific parameter if $\alpha = 1$,the score of Obj is solely determined by the diameter of Obj.

II. LITERATURE SURVEY

In the given spatial object database each spatial object may be associated with one or more keywords. The spatial objects are located at the same location, each with a different search keyword such as (Sno, x, y, keyword, rating) where x, y define the location of the spatial object in a tree.

Finding m-closest objects corresponding to the given query keywords is a very old problem. Main requirement of this problem is that among all the objects only the desired objects are selected such that the inter-object distances is minimum. Many spatial database applications require finding the desired objects with the desired criteria.

In general, m-closest keyword query search problem finds m desired and distinct objects whose inter-object distances is minimum. That is, Keyword cover is a method for retrieving multiple objects. All the desired multiple objects must be retrieved corresponding to the specified and given nearest location of the query location. Main function is to consider inter-object distances only but in many of the real life applications such as businesses, services, organizations, require not only the inter-object distances but also keyword rating, query location, cost of the products and many other service levels. Many of the existing query search keyword algorithms require more memory. Existing query keyword search finding methods are not efficient, fast, robust, accurate, reliable and scalable. Keyword query search time complexity is very high because the spatial dataset size increases with the time complexity of many existing query keyword finding algorithms and it approaches to exponential time complexity. When considered measures such as speed, accuracy, performance, reliability and response times of existing algorithms are not reasonably good.

Many existing algorithms are possible and in practice and useful. Also memory bottle necks are common in many of the existing algorithms and generate all possible search keyword covers unnecessarily. as for as indexing structure is considered the R*-tree is the predominant spatial data indexing tree structure for many existing spatial data query keyword search handling algorithms. Retrieval objects are always associated with keywords relevant to the query keywords and are very close to the actual query location. also note that many large number of different object combinations satisfy the currently specified desirable spatial relationship. It is almost impossible to satisfy all the following three conditions at the same time using existing spatial query keyword search handling algorithms.

1. Covering all query search keywords
2. All objects having minimum inter-object distance
3. All objects are close to a query location

Many of the algorithms are not based on a particular query location.

III. EXISTING SYSTEM

Existing problem is not scalable for large number of keywords and spatial keyword query search produces very lengthy intermediate results and computational complexity of these intermediate results is very high. Existing method

is called m-closest keywords query search. The problem of satisfying keyword cover property is called m-closest keywords (MCK) problem. the m-closest query requires to satisfy minimum inter object distance among the queried spatial database objects associated with query keywords. Keyword covers means retrieving multiple objects corresponding to the given query keywords. The existing system retrieves to find all the spatial objects which are very close to each other such that it minimizes the whose inter-objects distance.

IV. ALGORITHMS

FIRST BASELINE ALGORITHM

The following Baseline algorithm is an advanced version of m-closest keywords (MCK) search query method. The baseline algorithm uses both top-down and bottom-up indexing techniques a. Assume that there are n query keywords and n KRR*-trees have been constructed. That is there exist one to one mapping between keywords and KRR*-trees. Suppose T={kw_1,kw_2,kw_3,… kw_n} is the set of n query keywords. The child nodes of the root of KRR*-tree ($1 \leq i \leq n$) are retrieved and they are merged in order to generate candidate keyword covers. Suppose for a given candidate keyword cover o = {}, Where o is a node of-tree.

O.score = score (A, B)

A= $\max_{Ni,Nj \in o} distance(N_i, N_j)$ →(4)

B = $\min_{N \in O}(N.maxrating)$

Where N.maxrating is the maximum value of objects under N in the keyword rating dimension and the value dist(is the minimum Euclidean distance between nodes in the two dimensional geographical space represented by x and y dimensions.

Pseudo-code of the baseline algorithm is given in algorithm1. T is a set of query keywords. The baseline algorithm first generates candidate keyword covers after executing the function generate candidate. This function combines all the child nodes of all the root of all the KRR*-trees for all the keywords T. These candidate keyword covers are stored and maintained in the heap H. After this, the candidate with the highest score in H is selected and its function in order to generate more candidates. tree nodes are accessed in the depth first search order to access leaf nodes as fast as possible because the number of candidate keyword covers that are generated are very large. The first candidate solution consisting of objects other than the nodes of KRR*-tree is taken as the current best solution and it is denoted as BKC. BKC actually an intermediate solution sometimes pruning may also be applied to H when the score is less than BKC score, BKC will be continuously updated as long as all the remaining candidates are continue to process. If H contains no candidate then the algorithm terminates after returning current BKC.

Sometimes there is a chance of generating all possible keyword covers by the function Generate-candidate function. A better way is to generate keyword covers incrementally by combining individual nodes. The following diagram shows how nodes are incrementally generated by combining individual nodes in the bottom up fashion. K1, K2, K3 are three keywords and each keyword has two nodes. Highest score will be given first priority. Each time a new input node is combined in order to cover a keyword. We must observe one thing that if a combination has a score less than BKC score, then any super set of it must have a score less than BKC score. In such cases it is not necessary to generate the superset.

Algorithm 1- Baseline (T, Root)

Input:

1. Q is a set of given query keywords and
2. Root is the list of all the root nodes of all KRR*-trees

Output:

Optimal set of keyword cover

step 1:okw
step2:HGenerate-Candidatelist(Q,Root,okw)
step3:While(H≠null) repeat
step4:Cand is the candidate in H with the highest rating
step5:Remove cand from H
step6:Depth-first-search(H,Q,cand,okw)
step7:For each cand H do
step8:If(cand. Score <okw. Score) then
step9: Remove candidate from H
step10:Return okw

Algorithm 2- Depth-first-search (H<Q, cand okw)

Input:

1. A set of query keywords Q
2. cand is the candidate key
3. H is the set of candidate and
4. The present best solution okw

Output:

step1:If(cand⊆ leaf nodes) then
step2:s the set of objects in cand
step3:okw' generated ind keyword cover with the highest score in S
step4:if(okw. Score≤ okw'.score) then
step5:okw= okw'
step6: Else
step7: New-cand
step8:call(Q,cand,okw)
step9: swap cand with new-cand in H
step10:cand=new-cand with the highest score
step11: Depth-first-search (H,Q,cand,okw)

Algorithm 3: Generate –candidate –list (Q,cand,okw)
Input:

1. Q is the set of query keywords
2. cand is the candidate key
3. The present best solution okw

Output:

1. New-can∅
2. comvaluecombining child nodes of cand to generate keyword covers
3. foreach (comval ∈ comvalue) do
4. if(comval.score > okw.score) then
5. New-can =comval
6. End if
7. End for
8. Return New-can

Proposed Algorithm

Proposed system is highly scalable and reliable for finding answers to spatial object queries. Proposed system is based on both inter object distances as well as keyword rating of business or service objects. New method is generalization of the existing m-closest keywords search query system and it can be transformed either to consider only inter object distance or keyword rating only. Proposed system uses a new and simple technique before processing of spatial keyword training data sets. Proposed method is more generalized version of m- closest Keywords cover query search . Many real time business organizations are turning their business functions based on the keyword ratings of spatial objects. Decision making for purchasing new spatial objects is directly proportional to the the rating of the spatial objects. nowadays the rating based purchase transaction number is increasing rapidly. Therefore proposed method is mai inly depends on keyword rating of spatial objects.

The proposed method is having the Time Complexity is O (n logn) and O (n logn) for the corresponding memory requirements. The proposed method uses multiple tree structure. Each keyword has its own indexing structure called KRR-Trees.

Spatial Object can be represented in the form of (Sno,X,Y,Keyword,rating) where x and y are location details of objects in the two dimensional geographical space. Two important points that must be considered in the proposed method are:

1. Inter-object distance and
2. Keyword rating

Online business transactions are increasing very fast, using keyword rating of spatial objects.

KEYWORD NEAREST NEIGHBOR DETAILS

Baseline algorithm produces the correct result for the given query, but only thing is its computational cost is very high. It references large number of objects. These objects are represented as minimum bounding rectangles. Baseline algorithm is good but its performance decreases very fastly. When the number of query keywords increases because keyword covers generated increases very fastly.

Authors have proposed a new algorithm called keyword nearest neighbor algorithm to improve the performance of baseline algorithm dramatically.

This new algorithm is mainly based on a new concept called principle query keyword. All the objects that are mapped with the principle query keyword are called principle objects. Assume that the principle query keyword is denoted by k and the set of principle objects that are associated with k are denoted by

T is a set of query keywords and the principle query keyword k∈T. The local best keyword cover of a principle object , is denoted as

$$lbkc_{ok} = \left\{ kc_{ok} \mid kc_{ok} \in KC_{ok} \; kc_{ok}.score = \max_{kc \in KC_{ok}} kc.score \right\} \rightarrow (5)$$

Where is the set of keyword covers in such a way that the principle object is a member of each keyword cover. First for each principle object , must be identified. Check all the principle objects, the with highest score is called global best keyword cover (GBK).

Obviously any query keyword can be selected as the principle query keyword. For each principle object local best keyword cover must be computed. In general, the query keyword with the minimum number of query keyword objects will be selected as the principle query keyword for improving the performance of the algorithm.

Computational details of local best keyword cover:

Assume that be the principle object. , consists of principle object and all the objects belonging to each non-principle query keyword that is close to and have highest keyword ratings. Also there is a provision to compute by incrementally retrieving the keyword nearest neighbors of

Definition of keyword Nearest Neighbor:

Assume that T is the set of query keywords and let k be the principle query keyword and K∈T and a non-principle query keyword is such that {T-K}. Suppose that be the set of principle objects and is the set of objects of keyword The keyword nearest neighbor of a principle object in keyword is if and only if {,{, score for all .

The first keyword nearest neighbor of a principle object corresponding to the keyword is denoted as and the second keyword nearest neighbor is denoted as , and also the third keyword nearest neighbor is denoted as , and so on . All these keyword nearest neighbors are obtained by traversing the KRR*-tree. Suppose consider a node in the KRR*-tree.

$$\{O_k, N_{k\bar{\epsilon}}\}.\text{score} = \text{score}(A, B)$$

Where A value is computed as A= dist(and B value is computed as B=min(and we take dist(is the minimum

distance between in the normal geographical space of two dimensions (x-dimension and y-dimension) and .maxrating is the maximum keyword rating of along the keyword rating dimension.
In order to compute local best keyword cover we retrieve the keyword nearest neighbor of incremental way. Also note that we traverse the
KRR*-tree in the best first search (BFS) strategy in order to retrieve the keyword neighbors of the principle object corresponding to the keyword First root node of the KRR*-tree is traversed and then all of its children are stored in the heap memory H. Then for each node, the score value {A,B}.score is evaluated. Finally among all the nodes in H1 the node with the highest score is replaced by its child nodes and the same process is repeated until an object that is not in the KRR*-tree is visited. Finally, is selected as the best object and the score value .score is selected as current-best.

Description of keyword Nearest Neighbor Algorithm:
Keyword nearest neighbor algorithm follows a special technique for performance improvement by processing the principle keyword objects in terms of blocks rather than processing the each block separately. Assume that k is the principle query keyword. All the principle objects associated with k are indexed by using KRR*-tree. Suppose that is the principle node in the KRR*-tree and the local best keyword cover of is denoted as and it consists of and the corresponding nodes of each non-principle keyword of the query. Note that any KRR*-tree is present at the hierarchical level1 and all the child nodes of root are present at hierarchical leve12, and all child of the nodes at level1 are presented at level2 and soon. Assume that the node is presented at level6 in the KRR*-tree, then the corresponding nodes of in keyword are all those nodes at the same hierarchical level6 in the KRR*-tree. Keyword nearest neighbors is obtained incrementally in order to compute, from all the corresponding nodes.

The execution procedure detail of keyword nearest neighbor algorithm is explained below. First the algorithm selects a principle query keyword K and then starts its execution. The algorithm visits the root node of KRR*-tree and then stores all the child nodes of the root in the heap H. The score value of is calculated for each node in the heap H. The node whose maximum score is H.head is processed first. If maximum node (H.head) is present in KRR*-tree, then it is replaced in H by its child nodes. First we will compute .score for each child node and then H.head is updated. We will check whether H.head is a principle object or not. If H.head is a principle object thenis calculated and then .score is compared with the current best solution. If .score is greater than the bkc, bkc is updated to. Also pruning is applied based on the result of some operations.

MCK returns(x,y,z) Bkc returns(a,b,c)
A new algorithm is proposed for improving the performance of the special query keyword search algorithm. This new algorithm is called extended version of baseline algorithm. In the extended baseline algorithm all the root nodes of all the indexing data structures for all the keywords are stored in the proposed tree. The proposed special tree may be implemented either by using general binary search tree (BST) or any other efficient, effective, scalable and robust multi-way indexing tree data structure. Also note that same procedure can be used to create, store and manage all the root nodes of all trees in multi-way spatial query keyword search tree representation also.
Original baseline algorithm scans all the root nodes of KRR*-trees sequentially using linear search technique but the time complexity of simple linear search is O(n). Linear search is more costly in terms of CPU Computations. When the number of query keywords increases the time complexity of linear search increases in a linear fashion. Here employed algorithm is binary search tree to manage all root nodes of all KRR*-trees. The time complexity of binary search tree technique is O(log(n)). In many situations this sub-linear or logarithmic time complexity predominantly reduces the time complexity spatial keyword search and increases the performance of baseline algorithm dramatically.

Logarithmic time complexity of proposed binary search tree procedure is suitable for most of the real life applications. Binary search method solves more than 70% of the real life problems. For many other real life problems balanced multi-way search tree method is the fitted tree structure to store and main all tree nodes of all KRR*-trees. Always time complexity of any tree data structure is O (log(n)). Binary search tree is not a balanced search tree. When the binary search tree is not balanced for a particular data set, then the multi-way balanced search tree is inevitable for time critical application problems.

The proposed new algorithm is modified version of the baseline algorithm. This new algorithm reduces the time complexity of the already existing system and the proposed new algorithm is given below:

Proposed Algotrithm: **Advanced_Baseline (Q, Root)**

Input:
step 1:A set of query keywords called Q
step 2: all the generated root nodes of all the KRR*-trees and are stored in the tree with root node as Root

Output: Efficient, effective, scalable and optimized key word cover

step 1:Store all the root nodes of all the KRR-trees Bold
step 2:call okw
step3:callHGenerateCandidatelist(Q,Root,okw)
step 4:While(H is not empty) do
step 5:Can the candidate in H with the highest score
step 6:Remove can from H
step 7:Depth-first-search(H,Q,cand,okw)
step 8:For each candidateH do
step 9:If(cand. score okw.score) then
step 10:Remove cand from H
step 13:Return okw

VI. EXPERIMENTAL DETAILS

INPUT DATADETAILS

Input data are submitted in the form object coordinates and object coordinates are specified as longitude and latitude values.

Objects of University Keyword

x1 (x1+x1-width) y1 (y1+y1-width) z1 (z1+z1width)

100.0 150.0 130.0 180.0 40.0 10.0

950.0 150.0 980.0 180.0 40.0 10.0
650.0 800.0 680.0 830.0 50.0 10.0
150.0 950.0 180.0 980.0 10.0 10.0
250.0 850.0 280.0 880.0 60.0 10.0
850.0 950.0 880.0 980.0 25.0 10.0
400.0 300.0 430.0 330.0 40.0 10.0
600.0 400.0 630.0 430.0 30.0 10.0
700.0 500.0 730.0 530.0 30.0 10.0
600.0 600.0 630.0 630.0 50.0 10.0
300.0 900.0 330.0 930.0 50.0 10.0
100.0 300.0 103.0 330.0 40.0 10.0
100.0 500.0 130.0 530.0 20.0 10.0
200.0 200.0 230.0 230.0 30.0 10.0

Objects of Temple Keyword

x1 (x1+x1-width) y1 (y1+y1-width) z1 (z1+z1width

200, 50, 240, 90, 50, 20
300, 100, 340, 140, 30, 20
800, 20, 840, 60, 10, 20,
850, 400, 890, 440, 50, 20,
250, 600, 290, 640, 10, 20,
350, 550, 390, 590, 90, 20,
650, 880, 690, 920, 20, 20,
750, 100, 790, 140, 50, 20
100, 900, 140, 940, 50, 20,
400, 800, 440, 840, 20, 20
600, 400, 640, 440, 30, 20
800, 700, 840, 740, 60, 20
600, 300, 640, 340, 40, 20
100, 300, 140, 340, 40, 20
250, 950, 290, 990, 20, 20

Object of Airports Keyword

x1 (x1+x1-width) y1 (y1+y1-width) z1 (z1+z1width)

520, 250, 540, 270, 50, 10,
950, 450, 970, 470, 20, 10,
850, 560, 870, 580, 30, 10,
200, 650, 220, 670, 30, 10,
700, 800, 720, 820, 10, 10,
150, 800, 170, 820, 80, 10,
600, 970, 620, 990, 50, 10
600, 150, 620, 170, 50, 10,
900, 100, 920, 120, 30, 10,
400, 900, 420, 920, 20, 10,
300, 400, 320, 420, 30, 10,
300, 200, 320, 220, 10, 10,
700, 250, 720, 270, 60, 10

---- **FINAL BEST KEYWORD COVER IS** ----
150.0,800.0,170.0,820.0,80.0,10.0
250.0,850.0,280.0,880.0,60.0,10.0
350.0,550.0,390.0,590.0,90.0,20.0
150.0,800.0,170.0,820.0,80.0,10.0
200.0,650.0,220.0,670.0,30.0,10.0
300.0,200.0,320.0,220.0,10.0,10.0
300.0,400.0,320.0,420.0,30.0,10.0
optimal string is
150.0,800.0,170.0,820.0,80.0,10.0
250.0,850.0, 280.0,880.0,60.0,10.0
350.0,550.0,390.0,590.0,90.0,20.0
BUILD SUCCESSFUL (total time: 38 seconds)

CONCLUSIONS

Spatial query keyword search problem is a very old problem. Already existing algorithms uses only inter-object distances for retrieving objects against the given spatial keyword query. Latest trend is towards the usage of using both inter-object distance as well as keyword ratings of objects. A new algorithm is proposed for increasing the efficiency of already existing keyword searching algorithm. New algorithm applies ordering idea of taking all the trees that are generated after using all keywords in the spatial query keyword search.

REFERENCES

[1]. I.D.Felipe,V.Hristidis,and N.Rishe,"Keyword Search on spatial databases," in *Proc.*IEEE 24th Int.Conf.Data Eng.,2008,pp.656-665.

[2]. D.Papadias,N.Mamoulis,and Y.Theodoridis,"Processing and optimization of multiway spatial joins using r-trees,"in Proc.18th ACM SIGMOD-SIGACT-SIGACT symp.Principles Database Syst.,1999,pp-44-55.

[3]. D.Papadias,N.Mamoulis,and B.Delis,"Algorithms for querying by spatial structure," in Proc.Int.Conf.Very Large Databases,1998,pp.546-557.

[4]. R.Agrawal and R.Srikant,"Fast algorithms for mining association rules in large databases,"in Proc.20th Int.Conf.Very Large Data Bases,1994,pp.487-499.

[5]. G. R. Hjaltason and H. Samet. Ranking in spatial databases. In Advances in Spatial Databases — Fourth International Symposium, pages 83–95, Portland, ME, August 1995.

[6]. N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In SIGMOD Conference, 1995.