Reachable Nearest Neighbor Query Processing of R-Trees for Moving Query Object

Anitha Yarava,
CSE Dept ,MITS,
Madanapalle, A.P, India

S. Murali Krishna
M.Tech,Ph.D.
Professor & HOD, CSE Dept.
MITS, Madanapalle, A.P, India

ABSTRACT
Efficient storage and retrieval of multidimensional data in large volumes has become one of the key issues in the design and implementation of commercial and application software. The kind of queries posted on such data is also multifarious. Nearest neighbor queries are one such category and have more significance in GIS type of application. R-tree and its sequel are data partitioned hierarchical multidimensional indexing structures that help in this purpose. While general approaches are available in literature that discussing finding of Nearest neighbor for moving query point, few have explored on visible NN queries, but retrieved NN object may not always be reachable from the query object, since some obstacle objects (i.e. Hills, Rivers, Vallies) might be in between query point and NN point. This paper proposes Reachable Nearest neighbor queries for moving query object. The results are compared graphically with existing models, the proposed model out performs the existing models in a significant way.

General Terms
This paper exposes a novel method to find out Nearest Neighbor object, which is reachable from query object, i.e no obstacle object could exist along the path from query point to NN object.

Keywords: Multidimensional, Reachable nearest neighbor, Directional query.

1. INTRODUCTION
Database software has been playing a major role in various segments of management, research and entertainment over a period of fifty years. But in the past decade, the fundamental conceptualization and representation of data itself has undergone a sea change. More abstractness, semantics and dimensionalities have been brought in by multimedia, mathematical and unconventional data. The demand for storage and retrieval of such data in huge volumes has resulted in the invention and discovery of powerful novel techniques. Moreover post operational storage and processing such as data warehousing and data mining require more powerful and reliable methods to organize data for rigorous treatment of them.

The most significant all of these developments is querying using multiple attributes. Applications such as search engines, image recognition systems, scene analysis systems, games, etc. heavily depend on fast storage and retrieval done based on multiple attributes, in other words multidimensional data.

For over a period of thirty five years may scholars have churned out methods to handle this issue, The outcome is structures like R-Trees, X-Trees, A-Trees, Polygon Trees, etc. Today's research has turned towards development of powerful analysis tools [7][9] for these structures for various types of queries such as range, nearest neighbor and directional and joins. This paper addresses nearest neighbor.

Nearest neighbor queries are significant in the context of GIS, machine design, traveling and space management. NN query returns a point object as a result, which is located at the nearest location to the query object, the existing Nearest Neighbor queries[4,8] focusing on retrieval of NN object but they are not sure the retrieved NN object is reachable from query point or not, since obstacle objects like Hills, Rivers, and Valise might be in between query point and NN object, to sort out this problem in this work we proposed "Reachable Nearest neighbor queries for moving query object" by collaborating NN query with Directional queries. A review of necessary concepts is given the following section.

2. RELATED WORK
2.1 R-tree
The R-tree [1] are some of the most popular multidimensional access methods that use data partitioning. Object minimum bounding rectangles (MBRs) are grouped together in leaf nodes according to their spatial proximity, which are then recursively grouped in higher levels until the root contains a single node. R-trees (like most spatial access methods) were motivated by the need to efficiently process window queries, nearest neighbor queries, directional and join queries on multidimensional data.

A sample 2-dimensional data set and the corresponding R-Tree is shown in Figure2.

2.2 Nearest Neighbor Query
Given a collection of N objects in a d-dimensional space, a classical nearest neighbor query[3] returns exactly one object as result, the object with the lowest distance to the query point among all objects stored in the database, query processing steps explained as follows.

```
NearestNeighborSearch (Node, Point, Nearest)

{
NODE Node // Current NODE
POINT Point // Search POINT
NEARESTN Nearest // Nearest Neighbor

//Local Variables
NODE                newNode
BRANCHARRAY     branchList
Integer             dist, last, i

// At leaf levels
If (Node. type = LEAF)
Then
    For i := 1 to Node.count
        dist := objectDIST(Point,Node)
    If (dist < Nearest .dist)
        Nearest .dist := dist
```

```
            Nearest .rect := Node. branch.rect
// Non-leaf level
Else
     // Generate Active Branch List
       genBranchList(Point,Node,branchList)

     // Sort ABL based on metric values
       sortBranchList( branchList)

     // Perform Downward Pruning
        last = pruneBranchList(Node, Point,
        Nearest,branchList)

     // Iterate through the Active Branch List
        For i := 1 to last
             newNode := Node. branchbranChList,
     // Recursively visit child nodes
        nearestNeighborSearch( newNode,
        Point,Nearest)

    //  Perform Upward Pruning
        last := pruneBranchList( Node, Point,
        Nearest,branchList)
}
```

Figure 1: Nearest Neighbor Search Pseudo-Code

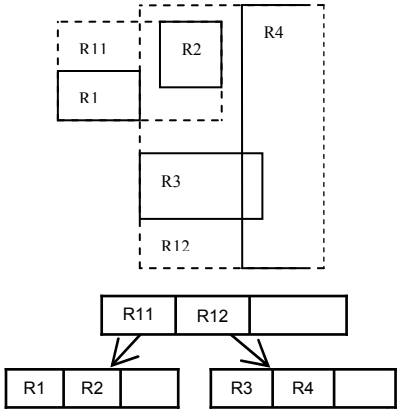


Figure 2: A R-Tree for sample data set

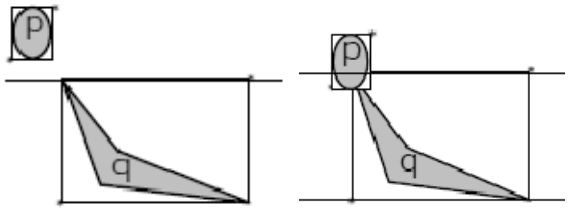
2.3 Directional Queries

Directional query [5] retrieves the set of point objects, which are existed at particular direction (East, West, North and South) from the Query point.

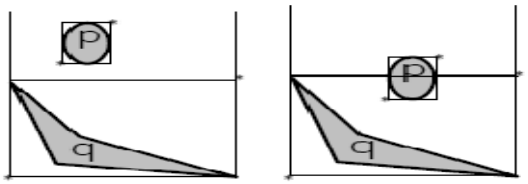
Directional queries are categorized as follows and shown in Figure 3.

a) Strong North
b) Weak North
c) Strong Bounded North
d) Weak Bounded North

In figure 3, 'q' is the query point and p is the object located in that particular direction.



a) Strong North b) Weak North



c) Strong Bounded North d) Strong Bounded North

Figure 3: Different Directional Queries

3. PROPOSED WORK

The existing NN queries [2][6], focusing on retrieval of NN object but they are not sure the retrieved NN object is reachable from query point or not, since obstacle objects like Hills, Rivers, and Valise might be in between query point and NN object, to sort out this problem in this work we proposed "Reachable Nearest neighbor queries for moving query object" by collaborating NN query with Directional queries, scenario is explained in Figure 4.

3.1 Reachable Nearest Neighbor Query for Moving Object

Given a data set P, an obstacle set O, and a query line segment q in a two-dimensional space, a RNN query returns a set of $\langle p, R \rangle$ tuples such that $p \in P$ is the nearest neighbor to every point r along the interval R ⊆ q as well as p is reachable to r. Note that p may be NULL, meaning that all points in P are not reachable to all points in R due to the obstruction of some obstacles in O. In contrast to existing continuous nearest neighbor query, RNN retrieval considers the impact of obstacles on visibility between objects; scenario is explained in the following figure 4.

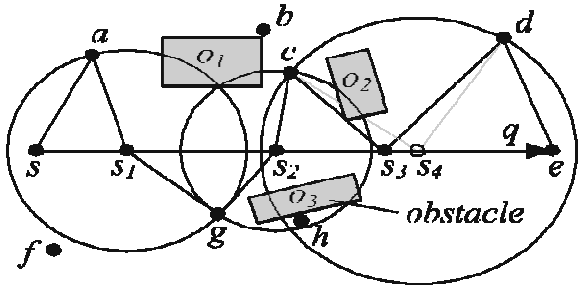


Figure 4. RNN search

Fig. 4, in which P = {a, b, c, d, f, g, h},
O = $\{o_1, o_2, o_3\}$ (i.e shaded rectangles) and q= [s,e]. The RNN query returns $\{\langle a,[s,s_1]\rangle, \langle g,[s_1,s_2]\rangle, \langle c,[s_2,s_3]\rangle, \langle d,[s_3,e]\rangle\}$ which indicates that point a is the reachable NN for any point along interval [s, s1], point g is the reachable NN for any point along interval [s1, s2], and so forth. Notice that point h is the NN for each point on interval [s2, s3] in the conventional NN retrieval, whereas it is not the reachable NN for any point on [s2, s3], because of obstacle O3.

3.2 Algorithm to find out RNN for moving Query point

ReachableNearestNeighborSearch(O,P,q)

//O is Obstacle set
//P is Objects set
//q is [s,e] , query line segment

For(each r ∈ R)
{
    //compute the NN point and Obstacle using conventional method (explained in fig8) //

Consider P_{lx} P_{ly} P_{ux} P_{uy} and O_{lx}, O_{ly}, O_{ux}, O_{uy} are retrieved MBR diagonal end points of object and obstacle respectively.

```
//Lets compute length of NN object MBR diagonal length //
Pdl = sqrt( (Oux - Olx)^2 + (Ouy - Oly)^2 )
//Lets compute length of NN Obstacle MBR diagonal length
Odl = sqrt( (Pux - Plx)^2 + (Puy - Ply)^2 ).
If( Pdl < Odl )
{
    //compute the midpoint of object MBR i.e (Pmx, Pmy)
    //compute the midpoint of Obstacle MBR (Omx, Omy)
    //compute distance between query point to NN object
    QPl = sqrt( (Qx - Pmx)^2 + (Qy - Pmy)^2 )
    //compute distance between query point to NN obstacle
    QOl = sqrt( (Qx - Omx)^2 + (Qy - Omy)^2 )
    if (QPl > QOl)
    {
        //object is not reachable, call
        reachableNearestNeighbor
        Search(O,P,q);
        where q=q-{Q}
    }
    Else
    retrieved object is reachable NN exit()
}
Else
    retrieved object is reachable NN
    exit()
}
```

Figure 5 : Algorithms for computing RNN

The above algorithm given in Figure 5, finds NN object and Obstacle, next using directional query computations it finds whether the obstacle is in between query and NN object along the same direction, if it is it will verifies the same procedure for next query point along the query segment

4. EXPERIMENTAL RESULTS

In the pursuit of establishing the claimed improvement due to the proposed method, YT model was taken as the base for comparisons. This is due to the fact of the superiority of YT model is already well established in the literature. The proposed model will henceforth referred as NNsk model

In the pursuit of establishing the claimed improvement due to the proposed method, YT model was taken as the base for comparisons. This is due to the fact of the superiority of YT model is already well established in the literature. The proposed model will henceforth referred as NNsk model

For each set, 500 query points were generated and the average relative Time Delay in answering a workload of 500 queries was measured..

5. CONCLUSION

Nearest neighbor queries are one of the most often posted queries on multidimensional data. To answer such queries within short time, databases employ multidimensional indexing structures such as R-trees and its ramifications. Even though quite a few models are available in the literature to find out NN point, but they couldn't give assurance on their reachability from query point. This work attempts to modify the conventional NN query with the aid of directional query to find out the RNN, results obtained are graphically presented and show a significant improvement over the previous models. The approach in discussed in this paper would facilitate better query optimization

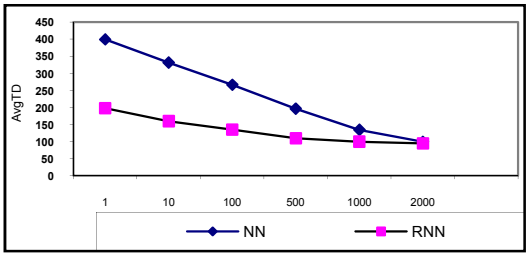


Figure.6 Comparison for NN and RNN

6. REFERENCES

1. Guttman, A,, "R-trees,: A Dynamic Index Structure for Spatial Searching," Proc. ACM SIGMOD, pp. 47-57, June 1984.

2. Yannis Manolopoulos Alexandros Nanopolos Apostolos N. Papadopolos, "R-trees Have Grown Everywhere" ACM Computing Surveys, Vol. V, No. N, Month 20YY.

3. Nick Roussopoulos Stephen Kelley Fr6d6ric Vincent "Nearest Neighbor Queries" SIGMOD'95,San Jose, CA USA Q 1995 ACM 0-89791-731 -6/95/0005..

4. Yunjun Gao · Baihua Zheng · Gencai Chen ·Qing Li · Xiaofa Guo" Continuous visible nearest neighbor query processing in spatial databases" The VLDB Journal (2011) 20:371–396 DOI 10.1007/s00778-010-0200-

5. .Dimitris Papadias, Yannis Theodoridis1, Timos Sellis1" The Retrieval of Direction Relations using R-trees", In the Proceedings of the 5th International Conference on Databases and Expert Systems Applications, DEXA, 1994, Springer

6. Wu, W., Guo, W., Tan, K.L.: Distributed processing of moving k-nearest-neighbor query on moving objects. In: ICDE, pp. 1116–1125 (2007).

7. Yufei Tao, Jun Zhang, Dimitris Papadias, "An Efficient Cost Model for Optimization of Nearest Neighbor Search in Low and Medium dimensional Spaces," IEEE Transactions on knowledge and data engineering, Vol.12, No.1, Oct2004.

8. Gao,Y., Zheng, B., Lee,W.C.,Chen, G.:Continuous visible nearest neighbor queries. In: EDBT, pp. 144–155 (2009)

9. Yannis Theodoridis, Emmanuel Stefanakis, and Timos Sellis, "Efficient Cost Models for Spatia Queries Using R-Trees" IEEE Transactions on Knowledge And Data Engineering, Vol 12,NO January 2000.