A Density-Based K-Nearest Neighbors Search Method

I. S. Jang, K.W. Min, W.S Choi
Telematics Research Division, ETRI
161 Gajeong-dong, Yuseong-gu, Daejon 305-350, Korea
{e4dol2, kwmin92, choiws}@etri.re.kr

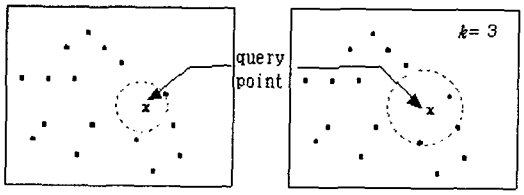
Abstract: Spatial database system provides many query types and most of them are required frequent disk I/O and much CPU time. k-NN search is to find k-th closest object from the query point and up to now, several k-NN search methods have been proposed. Among these, MINMAX distance method has an aim not to visit unnecessary node by applying pruning technique. But this method access more disk than necessary while pruning unnecessary node. In this paper, we propose new k-NN search algorithm based on density of object. With this method, we predict the radius to be expected to contain k-NN object using density of data set and search those objects within this radius and then adjust radius if failed. Experimental results show that this method outperforms the previous MINMAX distance method. This algorithm visit fewer disks than MINMAX method by the factor of maximum 22% and average 6%.

Keywords: GIS, KNN, Query.

1. Introduction

Spatial database system provides many query types - Point Query, Range Query, Join Query and Nearest Neighbor Query - and most of them are required frequent disk I/O and much CPU time.

The Nearest Neighbor queries in spatial databases refer to finding the spatial objects nearest to some given query points like Fig. 1-a. NN queries are used in a wide range of applications, such as Geographic Information Systems (GIS), Computer Aided Design (CAD), computational biology, decision support, and pattern recognition. NN queries in spatial databases can be classified into five major categories: simple k-NN queries, approximate k-NN queries, reverse NN queries, constrained k-NN queries, and k-NN join queries. In this paper, we focus on simple k-NN queries. k-Nearest Neighbor (k-NN) queries are to find the k spatial objects closest to some given query points like Fig. 1-b.



(a)Nearest Neighbor (b) 3- Nearest Neighbor

Fig. 1. Example of k-NN Query

It is simple for k-NN to search all spatial objects. But it is not desirable for huge large spatial objects. So, several k-NN search methods have been proposed. They do not visit unnecessary spatial object for k-NN query efficiently. For example, there are Voronoi cell(Cell-based index), R*-tree(Tree-based index), SS-tree, SR-tree(Clustering Method) and etc.

In this pager, we propose new method with using density. The remaining sections of this paper are organized as follows.

In Section 2, the related work is surveyed. In particular is introduced R*-tree with MinMax. In Section 3, we present a new k-NN query processing algorithm that can accommodate range estimation methods using density. Section 4 describes our performance experiments and presents the results.

Finally, we conclude the paper and highlight our future research directions in Section 5

2. Related Work

In this section, we show pruning method using R*-tree [2]. The algorithms retrieve the k nearest neighbors from the spatial indices by pruning away nodes that cannot lead to the k nearest neighbors. For example, Roussopoulos et al. [2] proposed an algorithm using the R*-tree for simple 1-NN queries and the algorithm can be generalized to handle k nearest neighbors.

The 1-NN algorithm performs a depth first traversal on a tree index. At each node, its child nodes are sorted according to the distance between their bounding rectangles (covering the spatial objects under the child nodes) from the query point, and they are visited in that order.

The algorithm maintains the most recently found nearest neighbor as it traverses the index tree. An index node is pruned if its bounding rectangle is farther away from the query point than the current nearest neighbor obtained so far. The main drawback of the algorithm is that it traverses the index tree in a depth-first manner. Once an index node is chosen to be visited, all the nodes in its sub-tree have to be either visited or pruned before its other sibling nodes can be visited. Such local search strategy therefore incurs some unnecessary disk accesses In order to reduce disk accesses further, Hjaltason and Samet proposed another algorithm that uses a priority queue to store all the nodes ordered by the distance between their bounding rectangles and the query point. The index nodes are then visited according to their order in the priority queue like fig 2.

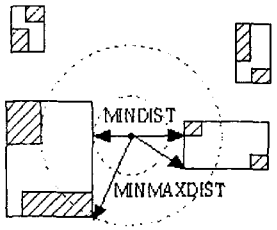


Fig 2. Pruning Method

But, the pruning method has following two problems. First, it does minimize search area. For example, it is can not optimal radius like Fig 3. According to shape of MBR, query point. Therefore, it is decrease performance.

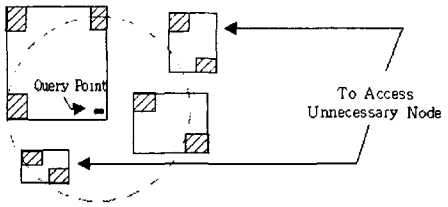


Fig. 3. Example of visiting unnecessary Node.

Second, in Pruning method, search area is changed dynamic. So, it is difficult to parallelize on k-NN query for load-balancing. For this, we propose new method for k-NN.

3. Density-Based on k-NN Query

In this Section, we will outline our proposed algorithm for k-NN queries using density.

1) Density vs Range for K-NN

Range of searching have related with distribution of Spatial Objects. If spatial object is dense, then search space comes to be narrows, otherwise search space comes to be wide. In this paper, considering this feature, we suppose to estimate search range with density.

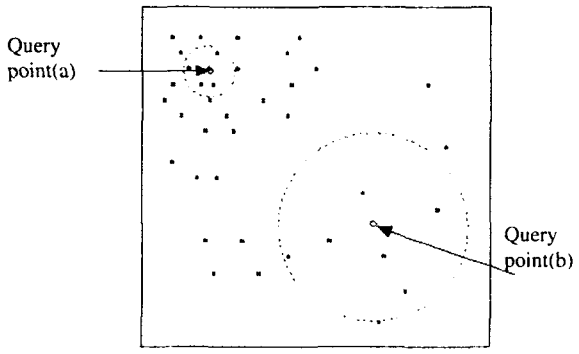


Fig.4. Example of visiting unnecessary Node.

2) Algorithm

For this algorithm, we have 4 Steps.

[STEP 1] Estimate density at area of given query point.

[STEP 2] Estimate search area with the density

[STEP 3] Process K-NN with the search area

[STEP 4] If result is fewer than k, enlarge search area

Following Table 1 shows symbol and description.

Table 1. Symbols and Description

Symbol	Description
k	The required number of nearest spatial objects
r	Radius or Range
q	Query Point $(q_1, q_2, \ldots, q_d)$
R_d	d -dimension
$A(q)$	Area of query point
$A_R(q,r)$	Cube Area. Within distance r from q
$A_S(q,r)$	Sphere Area. Within distance r from q
$\hat{N}(q)$	The estimated number of spatial objects in q area
$\hat{N}(A)$	The estimated number of spatial objects within distance r from q
$N(A)$	The number of spatial objects within distance r from q
$V(A)$	Volume of area A
$S_d(r)$	Volume of Cube. The radius of cube is r
$P_{den}(q)$	Density of given query point
$R_{den}(q,r)$	Density of $A_R(q,r)$
$S_{den}(q,r)$	Density of $A_S(q,r)$

[STEP 1] First, it is estimated density of given query point. To estimate density is related to selectivity. There is much method about Selectivity. In this paper, we omitted description of selectivity detail. Only we use the summary information about partitions or buckets of spatial objects to estimate range. Buckets are created by dividing the entire space into different groups.

Density of query area is calculated by following.

$$P_{den}(q)=\frac{\hat{N}(q)}{V(A(q))} \tag{1}$$

[STEP 2] if density is estimated in (1), then we can estimate search range (radius).

$$r=\frac{1}{2}\sqrt[d]{\frac{k}{D(q)\times R_d}} \tag{2}$$

Therefore,

$$r=\frac{1}{2}\sqrt[d]{\frac{k\times A(q)}{\hat{N}(\ldots)\times R_d}} \tag{3}$$

In here, based on uniform distribution assumption, r is estimated. Therefore, r should be updated. So following step is that flow.

STEP 2-1. Estimating range $r \leftarrow \frac{1}{2}\sqrt[d]{\frac{k \times A(q)}{\hat{N}(q) \times R_d}}$

STEP 2-2. Computing $\hat{N}(q,r)$ in area $[q-r, q+r]^t$

STEP 2-3. If then, $\left| \hat{N}(q,r) \times R_d - k \right| > \alpha$

Repeat STEP 2-2 with $r \leftarrow r \times \sqrt[d]{\frac{k}{N(q,r) \times R_d}}$

[STEP 3] With r Refined, we execute query.

[STEP 4] if the number of result is more than k, sorting result by distance is query result. But, if it is less than k, then do revise r with following equation

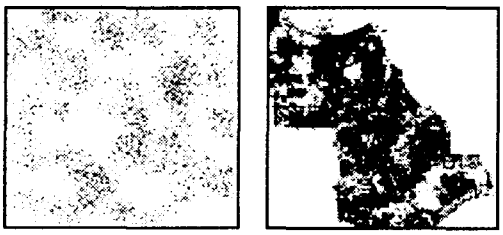
$$r' = r + \delta \quad , \quad \delta = 2r(\sqrt[d]{\frac{k}{N(q,r)}} - 1)$$

4. Experiments

In this section, we conduct experiments to compare existing pruning method with purposed method based on read dataset and Synthesis dataset. Pruning method is based on R*-tree that is outperformed. Also, we use multi-histogram for selectivity.

Dataset is two types, both is 2-d spatial point objects.

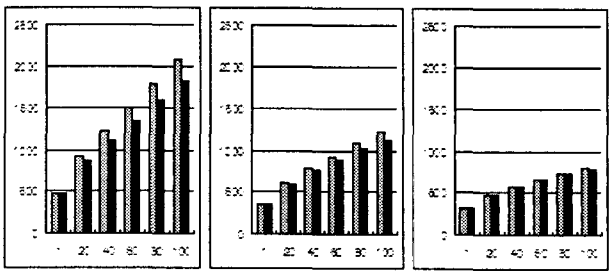
One is distribute Synthesis Data with 10,000 points. The other is real data with 36,548 points from long-beach. On implementing R*tree, page size is 1k, 2k and 4k. k is 1, 20, 40, 80 and 100. Query points use 100 points generated randomly. Dataset is like Fig 5.



(a) Synthesis Data (b) Real Data of Long-Beach Area

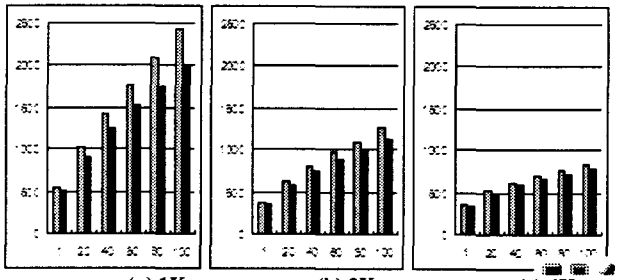
Fig. 5. 2-Dimsion Dataset

Fig 6 and Fig 7 show performance result. Y-axis means the number of total disk access. X-axis means k. In Fig, (a), (b) and (c) distinguish by page size. In graph, left bar mean pruning method, right bar mean purpose method



(a) Page Size :1K (b) Page Size : 2K (c) Page Size : 4K

Fig. 6. Performance of Synthetic Ununiform Dataset



(a) 1K (b) 2K (c) 4K

Fig. 7. Performance of Long-Beach Real Dataset

Experimental results show that this method outperforms the previous pruning method. This algorithm visit fewer disks than existing method by the factor of maximum 22% and average 6%.

5. Conclusions

K-Nearest Neighbor (k-NN) queries are to find the k spatial objects closest to some given query points. Nearest neighbor queries are important in many applications. In this paper, we therefore propose algorithm using density to handle k-NN queries. Experimental results show that this method outperforms the previous pruning method., This algorithm visit fewer disks than previos method by the factor of maximum 22% and average 6%..

References

[1] [1] N. Beckmann, H. Kriegel, R. Schneider and B.Seeger, 1990. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles, Proc SIGMOD Conference, pp.322-331

[2] [2] N. Roussopoulos, S. Kelley and F. Vincent, 1995. Nearest Neighbor Queries, Proc SIGMOD Conference, pp.71-79.

[3] [3] K. Cheung and A. Fu, 1998. Enhanced Nearest Neighbor Search on the R-tree, SIGMOD Record, 27(3):16-21.