International Journal of Database Theory and Application
Vol.7, No.6 (2014), pp.55-62
http://dx.doi.org/10.14257/ijdta.2014.7.6.06

The New Strategy of Object-Based Directional Query

Run-tao liu[1,*], Yan-ming Wang[2], Zhen-guo Zhao[2] and Guang -Yue Tian[2]

[1]*Institute of Information and Scientific Computing Technology, Harbin University of Science and Technology, Harbin 150080, China*
[2]*Department of Applied Mathematics, College of Applied Sciences, Harbin University of Science and Technology, Harbin 150080, China*

liurthar@163.com

Abstract

To increase directional query efficiency, based on the study of existing algorithms of directional query, new pruning rules for directional query were given, combined with the new index structure MB-tree. The rules exclude the MBRs outside the query area and output all leaf nodes in the MBRs inside the query area. Based on the orders defined in MB-tree, a new algorithm of directional query is given combining with MB-tree by using recursive method, and the new algorithm can reduce I/O cost effectively. Experiment showed that the new directional query algorithm reduces the number of visited nodes, decreases I/O cost, and improves the efficiency of directional query.

Keywords: *directional query; MB-tree; pruning rules*

1. Introduction

Object-based directional query [1] is the problem of whether or not to cross area and object. For example, government wants to build a new road. The buildings on the road need to be removed. These buildings that needs to be removed are directional query' s result.

Now, the study of directional query is little. The performance of query algorithm is not satisfactory. Xuan Liu, Shashi Shekhar, and Sanjay Chawla proposed directional query based on R-tree, this algorithm need to visit the whole R-tree [2], wastes time and has low query efficiency. Therefore, we use MB-tree [3] as the index structure, establish a new algorithm. Experiments show the performance of directional query based on MB-tree is better than that based on R-tree [4-5].

We introduce the definition of directional query, definition of MB-tree and structural feature of MB-tree in Section 2. Section 3 describes the algorithm of directional query based on MB-tree pruning rules. In Section 4 we present algorithm of directional query based on MB-tree. The experiments are given in section 5.

2. Related Work

2.1. Directional Query

Object-based directional query is also known as region query. This is an example of directional query in two-dimension space in Figure 1.

ISSN: 2005-4270 IJDTA

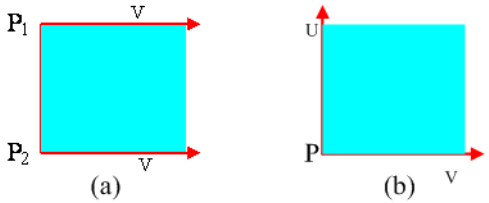

Figure 1. Directional Query

Figure 1(a) shows P_1, P_2 as endpoint, vector V as direction of query. Figure 1 (b) shows P as endpoint, vector V and U as direction of query.

2.2.MB-Tree

n objects, $\{ I_i | (a_x^i, a_y^i)$ is I_i 's left bottom point, (b_x^i, b_y^i) is I_i 's right top point$\}_{i=1}^{n}$. I_i is the minimum bounding rectangle of object *i*. Minimum bounding rectangle (MBR) is denoted by its left bottom point and right top point.

Definition 1 I_i, I_j ($i \neq j$) are two different object MBRs, when $b_y^i > b_y^j$; or when $b_y^i = b_y^j$, $a_x^i < a_x^j$; or when $b_y^i = b_y^j$ and $a_x^i = a_x^j$, $a_y^i > a_y^j$; or when $b_y^i = b_y^j$ and $a_x^i = a_x^j, a_y^i = a_y^j$, $b_x^i < b_x^j$, then I_i is called greater than I_j according to *y*-coordinate. It is denoted by $I_i \overset{y}{>} I_j$.

Definition 2 I_i, I_j ($i \neq j$) are two different object MBRs, when $a_x^j > a_x^i$; or when $a_x^i = a_x^j$, $b_y^i >$; or when $a_x^i = a_x^j$ and $b_y^i = b_y^j$, $b_x^j > b_x^i$; or when $a_x^i = a_x^j$ and $b_y^i = b_y^j$, $b_x^i = b_x^j$, $a_y^i > a_y^j$, then b_y^j I_i is called greater than I_j according to *x*-coordinate. It is denoted by $I_i \overset{x}{>} I_j$.

There are two types of nodes in the index structure of MB-tree, leaf node and intermediate node. Intermediate node is expressed by I_i (T,[(a_x^i, a_y^i), (b_x^i, b_y^i)], chd_1, chd_2, ⋯, chd_i, ⋯, chd_M). The MBR of all leaf nodes inside an intermediate node is denoted by [(a_x^i, a_y^i), (b_x^i, b_y^i)]. chd_i is pointing to pointer of the i-th child node. M is the maximum numbers of child node. T is the number of the child nodes that intermediate node actually has. Leaf node denoted by （0, [(a_x^i, a_y^i), (b_x^i, b_y^i)], ∅, ∅, ⋯, ∅）.

Definition 3 The MB-tree needs to satisfy the following conditions:

(1) if root node is not null, it has two subtrees at least, and the subtrees are also MB-trees.

(2) Intermediate node has T subtrees, root node of MB-tree as first level. When a node is located on an odd level of MB-tree, its child nodes are ordered by Definition 2. $MBR_i \overset{x}{>} MBR_j$, i=1, 2, ⋯, T -1, i<j≤ T; otherwise, its child

nodes is ordered by Definition 1. $MBR_i \overset{y}{>} MBR_j$, i=1, 2, …, T -1, i<j≤ T.

(3) Intermediate node is also an MB-tree.

3. Prune Rules

Suppose θ is the angle between Vector V and *x*-coordinate.

Theorem 1 $\theta \in (0, \pi/2)$, if the left top point of an MBR is on the right side of vector V (right bottom point on the left side of vector V), the MBR is outside the query area; If right top point of an MBR is on the right side of vector P_1P_2, the MBR is outside the query area. It is shown in Figure 2.

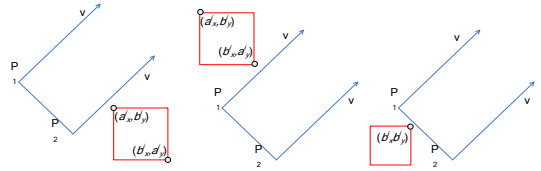

Figure 2. Example of Theorem 1

Theorem 2 $\theta \in (\pi/2, \pi)$, if the left bottom point of an MBR is on the right side of vector V (right top point on the left side of vector V), the MBR is outside the query area; If the left top point of an MBR is on the right side of vector P_1P_2, the MBR is outside the query area. It is shown in Figure 3.

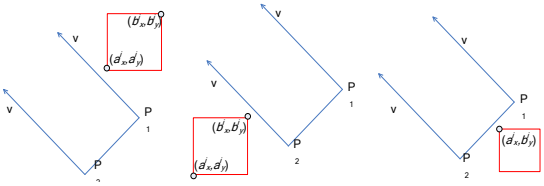

Figure 3. Example of Theorem 2

Theorem 3 $\theta \in (\pi, 3\pi/2)$, if the left top point of an MBR is on the left side of vector V (right bottom point on the right side of vector V), the MBR is outside the query area; if the left bottom point of an MBR is on the right side of vector P_1P_2, the MBR is outside the query area. It is shown in Figure 4.

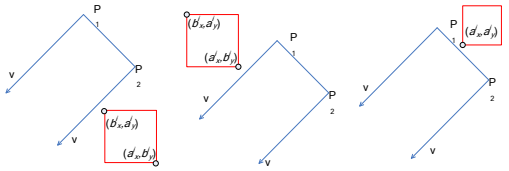

Figure 4. Example of Theorem 3

Theorem 4 $\theta \in (3\pi/2, 2\pi)$, if the left bottom point of an MBR is on the left side of vector V (right top point on the right side of vector V), the MBR is outside the query area; if the right top point is on the right side of vector P_1P_2, the MBR outside the query area. It is shown in Figure 5.

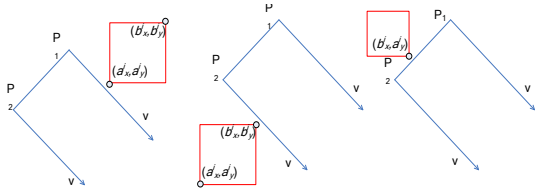

Figure 5. Example of Theorem 4

Theorem 5 The left top point and the right top point of an MBR are located between two vectors. If n>0, m>0, and the left bottom point is on the left side of vector P_1P_2, then the MBR is inside of the query area, as shown in Figure 6(a); If n<0, m<0, the left bottom point of an MBR is on the right side of vector P_1P_2, the MBR is inside the query area, as shown in figure 6(d). The left bottom point and the right top point of an MBR are between two vectors, if n<0, m>0, then the left top point of an MBR is on the left side of vector P_1P_2, the MBR is inside the query area, as shown in Figure 6(c); if n>0, m<0, the right bottom point of an MBR is on the right side of vector P_1P_2, the MBR is inside the query area, as shown in Figure 6(b).

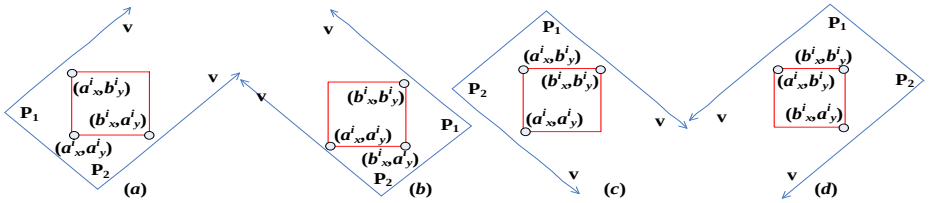

Fig.6. Example of Theorem 5

Theorem 6 Given vector V= (m, n), the endpoint of vector P(x_p, y_p), any point H(x, y), if A_{PH} <0, H(x, y) is on the right side of vector V; if A_{PH} >0, H(x, y) is on the left side of vector V. where

$$A_{PH} = \begin{bmatrix} x & y & 1 \\ m & n & 0 \\ x_p & y_p & 1 \end{bmatrix}$$

Query area [P_1,P_2,V, （m, n）], need to visit MBR I_i (T,[(a_x^i, a_y^i), (b_x^i, b_y^i)], chd_1 , chd_2, …, chd_i, …, chd_M)。

Prune Rule 1 When m*n>0, m>0. $A_{P_1P_2l}$ <0（l is the right top point of I_i） or A_{P_1d} >0 , A_{P_2d} >0 （d is the right bottom point of I_i） or A_{P_1c} <0, A_{P_2c} <0 （c is the left top point of I_i） , if $(a_x^j > a_x^i, b_y^j < b_y^i)$ or $(b_x^j < b_x^i, b_y^j < b_y^i)$ or $(a_x^j > a_x^i, b_y^j < b_y^i)$. I_j is pruned off according to the MB-tree order relation.

Prune Rule 2 When m*n<0, m>0. $A_{P_1P_2l}$ <0（l is the left top point of I_i） or A_{P_1d} <0 , A_{P_2d} <0 （d is the right top point of I_i） or A_{P_1c} >0, A_{P_2c} >0 （c is the left bottom point of I_i） , if $(a_x^j > a_x^i, a_y^j > a_y^i)$ or $(b_x^j < b_x^i, b_y^j < b_y^i)$ or $(a_x^j > a_x^i, b_y^j < b_y^i)$, I_j is pruned off according to the MB-tree order relation.

Prune Rule 3 When m*n>0, m<0. $A_{P_1P_2k}$<0（*k* is the left bottom point of I_i）or A_{P_1d} <0, A_{P_2d}<0 （d is the right bottom point of I_i）or A_{P_1c}>0, A_{P_2c}>0 （c is the left top point of I_i）, if ($a_x^j > a_x^i, b_y^j < b_y^i$) or ($b_x^j < b_x^i, a_y^j > a_y^i$) or ($a_x^j > a_x^i, a_y^j > a_y^i$), I_j is pruned off according to the MB-tree order relation.

Prune Rule 4 When m*n<0, m<0. $A_{P_1P_2k}$<0（*k* is the right bottom point of I_i）or A_{P_1d}>0, A_{P_2d}>0 （d is the right top point of I_i）or A_{P_1c}<0, A_{P_2c}<0 （c is the left bottom point of I_i）, if ($a_x^j > a_x^i, a_y^j > a_y^i$) or ($b_x^j < b_x^i, b_y^j < b_y^i$) or ($a_x^j < a_x^i, b_y^j > b_y^i$), I_j is pruned off according to the MB-tree order relation.

Prune Rule 5 When m*n>0, A_{P_1c} A_{P_2c}<0, A_{P_1d} A_{P_2d}<0 and m>0, $A_{P_1P_2k}$>0 or m<0, $A_{P_1P_2l}$<0; or When m*n<0, A_{P_1k} A_{P_2k}<0, A_{P_1d} A_{P_2d}<0 and m>0, $A_{P_1P_2c}$>0 or m<0, $A_{P_1P_2d}$ <0.（*k* is the left bottom point of I_i ,d is the right top point of I_i , *l* is the right top point of I_i ,c is the left top point of I_i）, all the data nodes will be output as the query result in I_i.

4. Query Algorithm

This paper used MB-tree as index structure, proposed a new algorithm to deal with data rectangle. The algorithm improved the efficiency of the query by excluded method.

[P_1,P_2,vector V=(m, n)] is the directional area. Judge-MBR-Dir-Relation is used to judge the relation between an MBR and the direction.

Algorithm 1 Judge-MBR-Dir-Relation(I_i)

Input：[P_1,P_2,vector V (m, n)], I_i (T,[(a_x^i, a_y^i), (b_x^i, b_y^i)], chd_1,chd_2…chd_i …, chd_M)

```
Output: r
Begin
  if（m*n>0）
     c=( a_x^i, b_y^i ),  d= ( b_x^i, a_y^i )；  k=( a_x^i, a_y^i ),  l=( b_x^i, b_y^i )；
   when(m>0)
     if( A_P1d >0 , A_P2d >0 or  A_P1c <0 , A_P2c <0)
       return r =0; /* outside the query area*/
     else
       if( A_P1c  A_P2c <0,  A_P1d  A_P2d <0,  A_P1P2k >0)
         return r =1; /* inside the query area*/
       else
         return r =2;/ * intersect with the query area */
   when(m<0)
     if （ A_P1d <0, A_P2d <0 or  A_P1c >0, A_P2c >0）
       return r =0；/* outside the query area*/
     else
       if （ A_P1c  A_P2c <0,  A_P1d  A_P2d <0 , A_P1P2l <0）
         return r =1; /* inside the query area*/
       else
```

return r =2 ; /* intersect with the query area */
else
c=(a_x^i, a_y^i), d=(b_x^i, b_y^i) ; k=(a_x^i, b_y^i), l=(b_x^i, a_y^i)
when(m>0)
if(A_{P_1d} <0 , A_{P_2d} <0 or A_{P_1c} >0 , A_{P_2c} >0)
return r =0; /* outside the query area*/
else
if(A_{P_1c} A_{P_2c} <0, A_{P_1d} A_{P_2d} <0, $A_{P_1P_2k}$ >0)
return r =1; /* inside the query area*/
else
return r =2;/ * intersect with the query area */
when(m<0)
if （ A_{P_1d} >0, A_{P_2d} >0 or A_{P_1c} <0, A_{P_2c} <0）
return r =0; /* outside the query area*/
else
if （ A_{P_1c} A_{P_2c} <0, A_{P_1d} A_{P_2d} <0 , $A_{P_1P_2l}$ <0）
return r =1; /* inside the query area*/
else
return r =2;/ * intersect with the query area */
End

Dir-Query is proposed according to structural feature of MB-tree and return value of algorithm 1.

Algorithm 2 Dir-Query（I_i）
Input: the head node of the MB-tree.
Output：result set Q.
Begin
Q← ∅:
Invoke **Judge-MBR-Dir-Relation**（I_i）
While（r=0）
if（m*n>0）
/* prune rule 1 */
when（m>0）
prune off the MBRs satisfying b_x^i <X_{P1} or b_y^i <Y_{P2};
when(A_{P_1c} <0,m>0, $a_x^i \le a_x^j, b_y^j \le b_y^i$)
prune off I_j ;
when(A_{P_1c} >0,m>0, $b_x^j \le b_x^i, a_x^i \le a_y^j$)
prune off I_j ;
/* prune rule 3*/
when（m<0）
prune off the MBRs satisfying $a_x^i > X_{P2}$ or $a_y^i > Y_{P1}$;
when(A_{P_1c} <0,m<0, $b_x^j \le b_x^i, a_x^i \le a_y^j$)
prune off I_j ;

when(A_{P_1c} >0,m<0, $a_x^i \le a_x^j, b_y^j \le b_y^i$)

prune off I_j ;

else

/* prune rule 2*/

when (m>0)

prune off the MBRs satisfying $a_x^i > X_{P1}$ or $b_y^i < Y_{P2}$;

when(A_{P_1c} <0,m>0, $a_x^i \le a_x^j, a_x^i \le a_y^j$)

prune off I_j ;

when(A_{P_1c} >0,m>0, $b_x^j \le b_x^i, b_y^j \le b_y^i$)

pass I_j ;

/* prune rule 4*/

when (m<0)

prune off the MBRs satisfying $b_x^i < X_{P2}$ or $a_y^i > Y_{P1}$;

when(A_{P_1c} <0,m<0, $b_x^j \le b_x^i, b_y^j \le b_y^i$)

prune off I_j ;

when(A_{P_1c} >0,m<0, $a_x^i \le a_x^j, a_x^i \le a_y^j$)

prune off I_j ;

While （r=1）

Q← all leaf nodes inside the MBR；

/* prune rule 5*/

While （r =2）

if （I_i is not a leaf node）

Dir- Query （I_{chdi}） ；

else

return Q；

End

Theorem 7 Algorithm 2 can stop within finite steps and output the correct result, whose time complexity is O（$M\log_{M^n}$）.

The proof of this theorem is omitted.

5. Experiment

Experiment is done in Inter Core i3, 2GHz CPU, 2GB RAM, Windows XP platform and used FLASH program language. Experiment data is randomly generated on computer. Direction area: vector V(1, 1),P1(20,20),P2(30,100), M is 50. The number of visited nodes in the query is in Figure 7.

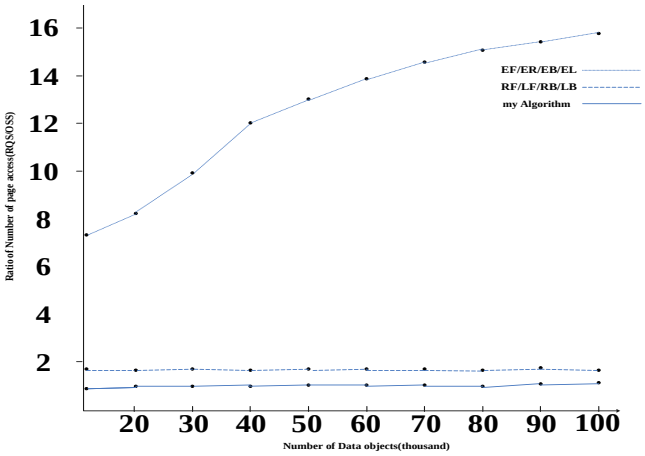

Figure 7. The Number of Visited Nodes

6. Conclusions and Future Work

In this paper, we propose the directional query based on MB-tree in spatial databases. Our experimental evaluation demonstrates that the ratio of number of operations is lower than the query based on R-tree[5]. The number of visited nodes is reduced evidently.

Next step, we will focus on research to reduce the number of visited nodes and the cost time, which also can improve the efficiency of directional queries.

Acknowledgements

This paper is supported by the 2011 scientific and technological research project of the Education Department of Heilongjiang Province of China under Grant No.12511103.

References

[1] X. Liu, S. Shekhar and S. Chawla, "Object-Based Directional Query Processing in Spatial Databases". IEEE Transactions on knowledge and data engineering, **(2003)**, pp. 95-304.
[2] A. Guttman, "R-trees: A Dynamic Index Structure for Spatial Searching", Proc of ACM SIGMOD In: Conf on Management of Data, **(1984)**; New York, USA.
[3] R.T. Liu and Z.X. Hao. "An order-based index structure for spatial data in an MB-tree", Journal of Harbin Engineering University, **(2010)**, pp. 481-487.
[4] Y.Q. Xiao, J. Zhang, N. Jing and J. Li, "Direction relation query processing using R-tree", Journal of Software, **(2004)**, pp. 103-111.
[5] Z.B. Zhang, J. Zhang and Y. Li, "A fine directional query filtering method with an R-tree", Journal of Harbin Engineering University, **(2010)**, pp. 1490-1495.

