MBR을 이용한 실용적 공간 데이터 관리

이재호
서울시립대학교 전자전기공학부
jaeho@ee uos.ac.kr

A Practical Approach to Spatial Object Indexing Using Minimum Bounding Rectangles

Jaeho Lee
Dept. of Electrical Engineering
University of Seoul

Abstract

We present a simple and efficient spatial object indexing scheme based on the minimum bounding rectangles (MBR) of the objects for use in applications in geographic information systems (GIS). We also provide the rationale behind the simple indexing scheme instead of other complex hierarchical indexing approaches such as the R-tree and its variants.

1 Introduction

In geographic information systems (GIS), efficient handling of spatial data such as locations of rivers, roads, gas pipelines, and cities is the key factor governing the overall performance of the systems. Especially the efficient implementation of the spatial relational operators defined in the spatial data model of the GIS is of crucial importance. Databases for GIS thus use various index mechanisms to facilitate efficient retrieval of the relevant spatial data with the spatial relational operations.

Index in this context is a means of accelerating database queries. By creating an index on an item in a database table, not only will this ensure referential integrity, but it will also enable queries to be completed quicker, providing that the field that the index has been created on, is used in the query statement [1].

A number of different representations of data have been proposed for handling spatial data. One popular approach indexes based on the concept of a minimum bounding rectangles (MBR) As shown in Figure 1, only four numbers ($x\_min, y\_min, x\_max, y\_max$) are sufficient to represent an MBR

In this paper, we present a simple and efficient spatial object indexing scheme based on the minimum bounding rectangles (MBR) of the objects for use in applications in geographic information systems (GIS). We then provide the rationale behind our choice of the simple indexing scheme over other complex hierarchical indexing approaches such as the R-tree [4] and its variants.

2 Principles of the R-tree

In this section, before we proceed to introduce our simple indexing scheme based on the minimum bounding rectangles, we briefly describe the principles of the R-tree and discuss the drawbacks of applying these mechanisms for our GIS applications.

The minimum bounding rectangles (MBR) of spatial objects preserve the most essential geometric properties of the object, that is, the location of the object and the extension of the multi-dimensional object in each axis [2]. In this paper, we only consider two dimensional objects for GIS, but the idea can be easily extended to higher dimensional objects.

The MBR of the objects may be grouped into hierarchies, and then stored in another structure such as a B-tree. The R-tree [4] and its variants such as the R^+-tree [6] and the R^*-tree [2] are examples of this approach. The R-tree and its variants are probably one of the most popular access methods for rectangles.

An R-tree is a B^+-tree where a non-leaf node contain entries of the form *(pointer to a child node, MBR of all rectangles of the child node)* A leaf node contains entries

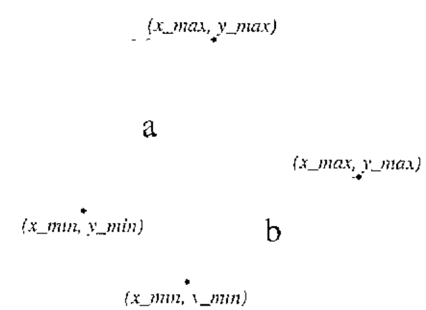

Figure 1: Minimum Bounding Rectangles (MBR)

x_min lower left x	*y_min* lower left y	*x_max* upper right x	*y_max* upper right y
.	.		
.	.	.	.
.	.		

Figure 2: MBR Schema

of the form *(reference to the spatial object in the database, MBR of that spatial object)*. In addition, an R-tree satisfies the following properties:

- The root has at least two children unless it is a leaf.
- All leaves appear on the same level.
- Every non-leaf node has between m and M children unless it is the root.
- Every leaf node contains between m and M entries unless it is the root.

where M is the maximum number of entries that will fit in one node and m is a parameter specifying the minimum number of entries in a node such that $2 \leq m \leq M/2$.

The R-tree uses a heuristic optimization criterion to minimize the area of each enclosing rectangles in the inner nodes. Other variants of R-tree incorporate additional refined heuristic optimization criteria, but the principles of R-tree remain same [3].

3 MBR-based Simple Indexing

Our simple spatial object indexing scheme is also based on the *minimum* bounding rectangles (MBR) of the objects, but the rectangles are not grouped into hierarchies as in the R-tree. Instead the MBR is associated with each object stored in a relational database. The MBR part for each object has the form shown in Figure 2

Spatial query is processed in two steps, called filter and refinement step [3].

1. *filter step:* Find all objects whose MBR is relevant with the spatial operation.
2. *refinement step:* For those objects, check whether they really fulfill the query condition using the exact representation as necessary.

Our indexing scheme is focusing on the filter step in that it attempts to filter out irrelevant objects as early as possible before the costly refinement step is performed.

Our MBR-based indexing do not need any additional procedure to implement on a database. We can simply add one MBR condition to the query as shown in Figure 3 and 4. Figure 3 shows the query to retrieve the OIDs of the counties which have intersecting gas-lines longer than 1000 unit length. We can transform this query to use our MBR-based simple indexing as shown in 4. The query has one additional following as follow·

```
SELECT
        a.foid FROM county a, gasline b
WHERE
        (b.length > 1000)
  AND
        INTERSECT(a.location, b.location) = 1;
```

Figure 3 Query without MBR

```
SELECT
        a.foid FROM county a, gasline b
WHERE
        (b.length > 1000)
  AND
        NOT((a.x_max < b.x_min) or
            (a.y_max < b.y_min) or
            (b.x_max < a.x_min) or
            (b.y_max < a.y_min))
  AND
        INTERSECT(a.location, b.location) = 1;
```

Figure 4. Query with MBR

```
NOT((a.x_max < b.x_min) or
    (a.y_max < b.y_min) or
    (b.x_max < a.x_min) or
    (b.y_max < a.y_min))
```

The four sub-conditions within the `NOT` expression form the condition for two MBRs to be *disjoint* (disjoint index condition). Thus the overall expression is the condition for two MBRs to be *non-disjoint* which has the possibility to `INTERSECT`.

For the queries which test the `DISJOINT` spatial relationship, the condition can be simply replaced with the *disjoint index condition* as mentioned above In our GIS system, this transformation of queries to use the MBR-based indexing is performed automatically by analyzing the spatial operation in the query.

4 Discussion

Our early experiments with the MBR-based simple indexing demonstrated promising results and showed the effectiveness of the scheme on real GIS data For the example query shown in Figure 3, the modified query in Figure 4 using the MBR-based simple indexing produced results about three times faster than the original query. In this section, we discuss the benefits and effectiveness of our indexing scheme

- It is simple and easy to implement on relational databases As presented in Section 2, the R-tree and its variants use B^+-tree structure to hierarchically organize the MBRs. Thus it is hard to implement and maintain especially on existing databases.

- It is generic scheme and thus system independent. Other complex index schemes such as the R-tree need special embedded procedures, which are system dependent, to implement the complex index algorithms.
- Since the MBR of each object is generated when the object is stored into the database, there is only static overhead but no runtime overhead to maintain the index.
- It can take advantage of the efficient builtin database access methods to use the indexing scheme instead of the complex external procedures Databases are optimized to effectively handle queries and thus we do not have to develop additional optimization at the level of indexing algorithm.

5 Conclusion

In this paper, we presented a simple MBR-based spatial object indexing scheme for relational databases and discussed the rationale behind the simple indexing scheme over other complex hierarchical indexing methods. Initial results are promising for our GIS applications where relational databases are used to store the spatial objects. Currently, we are investigating the effectiveness of the simple indexing with various operating parameters. It seems that the effectiveness of our MBR-based simple indexing grows as the size of the data increases to a certain degree, but yet we need extensive experiments to identify the relationship.

We are also planning to perform comparative experiments with other schemes such the R-tree and Quadtrees [5].

References

[1] Michael Abbey and Michael J. Corey. *Oracle: A Beginner's Guide*. Oracle Series. Oracle Press, June 1995.

[2] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. The R*-tree: An efficient and robust access method for points and rectangles. In *Proceedings of the SIGMOD Conference*, pages 322–331, Atlantic City, NJ, June 1990

[3] Thomas Brinkhoff, Hans-Peter Kriegel, and Bernhard Seeger. Efficient processing of spatial joins using R-trees. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 22(2):237–246, June 1993.

[4] Antonin Guttman R-trees: A dynamic index structure for spatial searching. In *Proceedings of the SIGMOD Conference*, pages 47–57, Boston, MA, June 1984

[5] H. Samet and R. E Webber. Storing a collection of polygons using quadtrees *ACM Transactions on Graphics*, 4(3):182–222, July 1985

[6] Timos Sellis, Nick Roussopoulos, and Christos Faloutsos. The R^+-tree. A dynamic index for multidimensional objects. In P M. Stocker and W. Kent, editors, *Proceedings of the 13th international Conference on Very Large Databases (VLDB)*, pages 507–518, Brighton England, September 1987. (Also University of Maryland Computer Science TR-1795).