A Survey of R-Trees

Aaron Gorenstein
University of Wisconsin - Madison
agorenst@cs.wisc.edu

Rebecca Lam
University of Wisconsin - Madison
rjlam@cs.wisc.edu

March 9, 2013

1 Introduction

What was the motivation for this survey? What was the point of the original paper?

An overview of the rest of the paper here.

2 Overview of R-Trees

To solve the problem of performing efficient searchs on spatial data, Guttman proposed the R-tree, which inspired a variety of different variations analagous to the family of B-trees. In Section 2.1 we outline the original R-tree paper, and in Section 2.2 we examine the variants and draw appropriate comparisons.

2.1 R-Trees

In 1984, Guttman first proposed the idea of modifying the B-tree structure to use minimum bounding rectangles (MBR) as a way to restrict the search space during a lookup for spatial data. This data structure is called the R-tree. R-trees are structured similarly to B-trees except, instead of having separation values in each internal node that divide its subtrees, R-tree internal node entries correspond to MBRs that bound its descendents. For instance, the MBR of a particular node completely overlaps the MBRs of the nodes of its child and its child's children. ike in the B-tree case, nodes correspond to disk pages and leaves point to database objects.

R-trees are bound by two parameters m and M, the minimum and maximum number of entries for each node except the root, respectively. An internal node entry is of the form (mbr, p), where mbr is the MBR containing the MBRs of its descendents and p is the pointer to its child subtree. The mbr entry is of the form (I_0, I_1, ..., I_{n-1}), where n is the number of dimensions and I_i is of form $[a,b]$, a closed bounded interval along the i-th dimension. Similarly, a leaf node entry is of the form (mbr, oid), where mbr is the MBR containing the object, and oid is the identifier for the object in the database. Finally, the root node must have at least three entries except if it is a leaf.

2.1.1 Search

In order to find all entries contained by a bounding rectangle in the R-tree, the pseudocode of Figure 1 is used.

2.1.2 Insert

Insert.

2.1.3 Delete

Delete

2.2 R-Tree Variants

Much like its cousin, the B-tree, the R-tree has a few main variants such as the R^+-tree and the R^*-tree, which we discuss in the following sections.

```
function SEARCH(T, S) ▷ Return all entries contained by S given an R-tree rooted at T
    if T is not a leaf then
        for all E in T do
            if E.mbr overlaps S then
                SEARCH(E.p, S)
            end if
        end for
    else
        for all E in T do
            if E.mbr overlaps S then return E.oid
            end if
        end for
    end if
end function
```

Figure 1: Pseudocode for searching a R-tree given a search rectangle

2.2.1 R+-Trees

2.2.2 R*-Trees

3 Implementation Challenges

4 Database Related Challenges

5 Modern Applications

cool topics like parallelism, concurrency, emerging applications

6 Conclusion