Sachin Srivastava  
R Tree Implementation using Java  
CS 539

This is an implementation of an R Tree. The page size is 4096. The index pages have nodes each with one pointer and 4 extreme values minx, miny, maxx and maxy. We have 5 integer values per node. This equals to 5\*4 equal to 20 bytes per node. Hence, there are 204 such nodes in one index page. So, the Fanout of the tree is 204. The tree has a root page, 1st level has index pages and the 2nd level has data pages.  
The leaf node has actual data pages containing tuples. Each tuple has two integers and 500 characters. This equals to 1008 bytes per node. So each data page contains 4 tuples.

First the read the file and store the value of x and y. The character value is a random character generated using UUID class. At first the data is sorted using the Hilbert Value. The sorted data is then entered in the RTree using bulk loading method. The class which implements bulk loading is R3. The complete R Tree is stored in this class. This class also contains functions for point search, range search and maximize function.

indexPage and dataPage are classes which define the index page and data page of the tree. Each of this class has a corresponding bounding box. I have used arraylist to implement dynamic allocation of pointers and tuples. The class bound contains the bounding box and the pointer directing to the corresponding nodes, whereas the class tuple stores each tuple.

When doing the pointsearch I am traversing the tree based on the bounding box at each level. At each level I am checking the bounding box based on the minx, miny, maxx, maxy values of bounding boxes. Once the correct box is located I am getting the pointer value and going to the next level. This is repeated till we find the correct tuple. I am also checking for duplicate values, hence will keep traversing the tree until all the bounding boxes are checked.

The range query is implemented in a similar way. For the range query all the boxes which fall under the given range are traversed. All the tuples which fall under this range are reported.

To calculate the maximum function, we are traversing the tree in the following way. We are first calculating the f value using minx and miny and storing it in cmin. As we are traversing the tree we are updating the cmin values using minx and miny of different bounding boxes. Any box which has a an f value using the maxx and maxy less than the current cmin is pruned and not traversed. That is as we are calculating the cmin values, we are only calculating traversing the tree for boxes which lie above this point.