Module 5: Disk-based Index Structures

Lecture 21: R-tree Variants

Prev topic | Next topic | Prev page | Next page

## The Lecture Contains:

- Greene's R-tree
- R*-tree
  - Insertion
  - Splitting
  - Forced reinsertion
- R + - tree
  - Example
  - Searching

### Greene's R-tree

- Different splitting strategy than R-tree
- Choose axis to split
  - Find two most distant rectangles (seeds)
  - For each axis, record separation of seeds
  - Normalize separation by range of axis
  - Choose axis with largest separation
- Distribute regions
  - Sort entries by min value along chosen axis
  - Assign first $(M+1)/2$ entries to one group, last $(M+1)/2$ entries to another group
  - Assign remaining entry (if $M+1$ is odd) to group with largest increase in volume

## R*-tree

- Three optimizations over R-trees
    - Insertion
    - Splitting
    - Forced reinsertion when over flow
- Instead of just volume, four parameters:
    - Volume wasted in a node ("dead space")
    - Volume of overlaps
    - Margin (i.e., sum of edges)
    - Space utilization

## Insertion

- If child pointers are not leaf nodes
    - Choose subtree with least volume increase (as before)
- If child pointers are leaf nodes (level leaf-1)
    - Choose subtree with least overlap enlargement
    - Overlap ( $E_k$ ) = sum of volume( $E_k \setminus E_i$ ) for all other entries $E_i$
    - $O(M^2)$ because $O(M)$ entries need to be checked and each entry, overlap to be calculated for $O(M)$ siblings

## Splitting

- Two parameters:
  - Margin: Margin(MBR(Group 1)) + Margin(MBR(Group 2))
  - Overlap: Volume(MBR(Group 1) $\cap$ MBR(Group 2))
- Choose split axis
  - For each axis, sort entries by low values
  - Enumerate $M - 2m + 2$ possible groupings or distributions

## Splitting

- Two parameters:
  - Margin: Margin(MBR(Group 1)) + Margin(MBR(Group 2))
  - Overlap: Volume(MBR(Group 1) ∩ MBR(Group 2))
- Choose split axis
  - For each axis, sort entries by low values
  - Enumerate $M - 2m + 2$ possible groupings or distributions
    - k$^{th}$ distribution: First $m - 1 + k$ entries and rest
  - Compute sum of margins along that axis of all distributions
  - Do the same by sorting high values of each axis
  - Choose axis with least sum of margin values
  - $O(2d.M \lg M)$
- For chosen axis and low/high value, choose distribution with least overlap

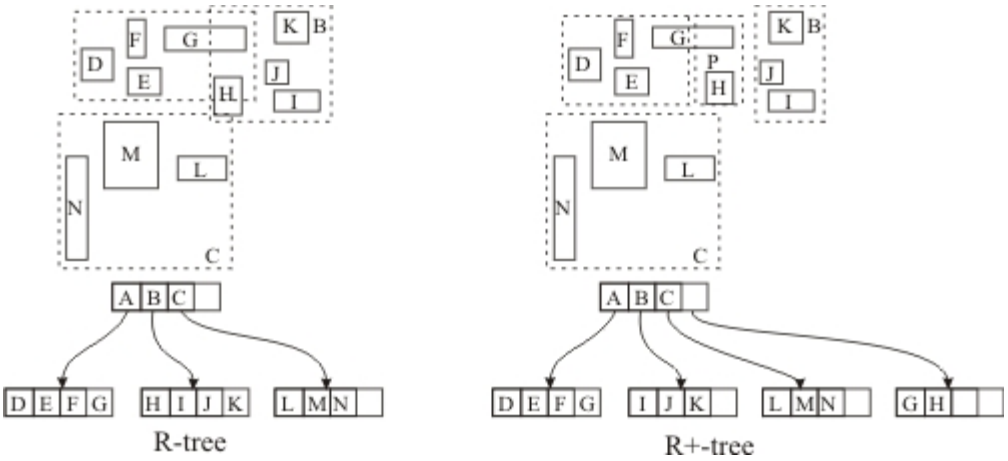
## Forced reinsertion

- When over flow occurs, do not split immediately
- Choose p entries to be re-inserted
    - Empirically, $p$ = 30%
- Which $p$ entries?
- For each entry, compute distance of its centroid to centroid of node
    - Remove from closest to farthest (close reinsert)
        - Reinsertion will not increase volume of node
    - Remove from farthest to closest (far reinsert)
        - Likely to be reinserted at some other node
    - Not clear which one better
        - Experimentally, close reinsert better
- $O(N \lg N)$
- Done only once at each level to avoid infinite loop

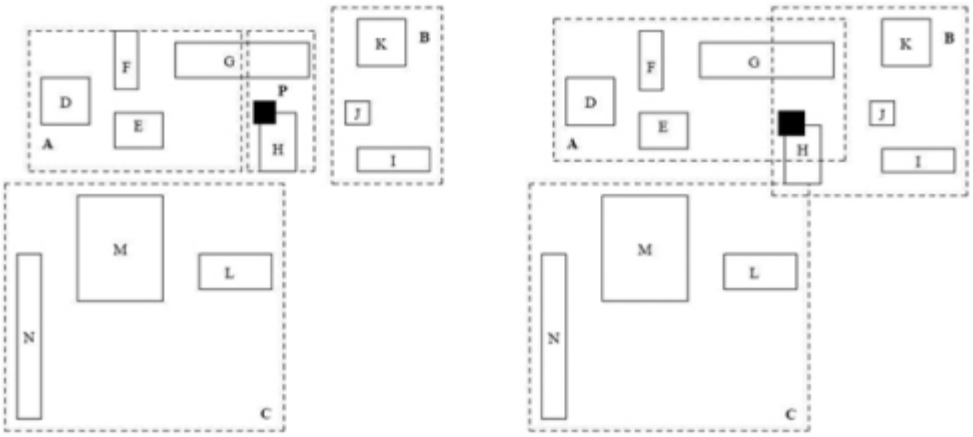
R + - tree

- Mix of R-tree's data partitioning with K-d-B-tree's space-partitioning
- Reduces dead space problem of K-d-B trees by using index rectangles
- Reduces node overlap by making siblings disjoint
- Data objects split into rectangles
    - Disjoint
    - Union gives original data object
- Data object stored in mutliple leaves
- Downward splitting
- No space utilization guarantee

Example


F
G
K
B
D
E
H
J
I
M
L
N
C
A
B
C
D
E
F
G
H
I
J
K
L
M
N
R-tree
F
G
K
B
D
E
P
H
J
I
M
L
N
C
A
B
C
D
E
F
G
I
J
K
L
M
N
G
H
R+-tree

## Searching




- In R-tree, nodes A and B are searched
- In R+-tree, only node P is searched
- Trade-o between overlapped searches and increased height
- Not much better than R*-trees