Spatial Index Structures

Excerpt from Spatial Index Structures by Yi-shin Chen
Outline

- Introduction
- Spatial Indexing
- R-Tree
- R⁺-Tree
- Quadtrees
Introduction

- Spatial objects
  - Points, lines, rectangles, regions, ...

- Hierarchical data structures
  - Based on recursive decomposition, similar to divide and conquer method, like B-tree

- Why not B-Tree?
  - More than one dimension
  - Concept of closeness relies on all the dimensions of the spatial data
Spatial Indexing

- Mapping spatial object into point
  - In either same, lower, or higher dimensional spaces
  - Good for storage purposes
  - Problems with queries like finding the nearest objects

- Bucketing methods
  - Based on spatial occupancy
  - Decomposing the space from which the data is drawn
    - Minimum bounding rectangle (MBR) : e.g., R-Tree
    - Disjoint cells: e.g., $R^+$-Tree
    - Blocks of uniform size; e.g., Quadtrees

R-Tree [Guttman, 1984]

- Based on Minimum Bounding Rectangle
  \[(m,M) = (1,3)\]

- Bounding rectangles could overlap each other (e.g., R3 vs. R4)
- Each object is only associated with one bounding rectangle
R-Tree Semantics

- Height-balanced tree similar to B-tree for k-dimensions
- Every leaf node contains between m (m \leq M/2) and M index records, unless it is the root
- For each index record \((I, \text{tuple-identifier})\) in a leaf node, \(I\) is the MBR that contains the n-dimensional data object represented by the indicated tuple
- Every non-leaf node has between m and M children unless it is the root
- For each entry \((I, \text{child-pointer})\) in a non-leaf node, \(I\) is the MBR that spatially contains the rectangles in the child node.
- All leaves appear on the same level
- The root node has at least two children unless it is a leaf
Insertion Process

A new index entry

ChooseLeaf

Has room

Yes

Install X

No

SplitNode

Adjust all related entries

AdjustTree

Different variant:
- Exhaustive
- Quadratic
- Linear
- Packed
- Hilbert Packed
- …etc.
Process of Quadratic Split
(page 52 in Guttman’s paper)

Pick first entry for each group
Run PickSeeds

ABC

def
ghij
klm
Process of Quadratic Split

(page 52 in Guttman’s paper)

PickSeeds
PS1 [Calculate inefficiency of grouping entries together]
For each pair of E1 and E2, compose a rectangle R including E1 and E2
Calculate $d = \text{area}(R) - \text{area}(E1) - \text{area}(E2)$
PS2 [Choose the most wasteful pair]
Choose the pair with the largest $d$
Process of Quadratic Split

(page 52 in Guttman’s paper)

Pick first entry for each group (PickSeeds)

Check if done

No

Select entry to assign (PickNext)
Process of Quadratic Split
(page 52 in Guttman’s paper)

Pick first entry for each group
(PickSeeds)

Check if done

PickNext
PN1 [Determine cost of putting each entry in each group]
For each entry E
- calculate \( d_1 \) = the increased MBR area required for G1
- calculate \( d_2 \) = the increased MBR area required for G2
PN2 [Find entry with greatest preference for one group]
Choose the entry with the maximum difference between \( d_1 \) and \( d_2 \)
Process of Quadratic Split
(page 52 in Guttman’s paper)

Pick first entry for each group (PickSeeds)

Check if done

No

Select entry to assign (PickNext)
Process of Quadratic Split
(page 52 in Guttman’s paper)

Pick first entry for each group (PickSeeds)

Check if done

No

Select entry to assign (PickNext)
Process of Quadratic Split

(page 52 in Guttman’s paper)

Pick first entry for each group (PickSeeds)

Check if done

No

Select entry to assign (PickNext)
Process of Quadratic Split
(page 52 in Guttman’s paper)

Pick first entry for each group (PickSeeds)

Check if done

No

Select entry to assign (PickNext)
Your Exercise

- Build a R-Tree for these spatial data

- Assume \((m, M) = (2, 4)\)
- You can use the demo applets to verify your result
Searching for Objects in R-Tree (Querying)

1. Input \((T, j)\)
2. Check each entry \(E\) of \(T\)
   - Leaf? (Yes or No)
     - Yes: Check each entry \(E\) of \(T\) and Overlap (Yes or No)
       - Yes: Qualifying record
     - No: Search each subtree \(E\) of \(T\) and Overlap (Yes or No)
       - Yes: Overlap (Yes or No)

Diagram:
- \(G1\) and \(G2\) regions
- \(A, B, C, \ldots\) objects
- \(d, e, f, g, h, i, j, k, l, m, x\) entries
Main Drawbacks of R-Tree

- R-tree is not unique, rectangles depend on how objects are inserted and deleted from the tree.

- In order to search some object you might have to go through several rectangles or the whole database
  - Why?
  - Solution?
R⁻⁺-Tree

- Overcomes the problem with R-Tree
  - If node overlaps with several rectangles insert the node in all
  - Decompose the space into disjoint cells
R+-Tree Properties

- R+-tree and cell-trees used approach of decomposing space into cells
  - R+-trees deal with collection of objects bounded by rectangles
  - Cell tree deals with collection of objects bounded by convex polyhedron
- R+-trees is extension of k-d-B-tree
- Retrieval times are smaller
- When summing the objects, needs eliminate duplicates
- Again, it is data-dependent
Quadtrees (Example of Region Quadtree)
Properties of Quadtrees

- The blocks are required to be disjoint
- Have standard sizes
- At standard locations
- Based on successive subdivision of image array into four equal-size quadrants
- If the region does not cover the entire array, subdivide into quadrants, sub-quadrants, etc.
- It is data independent
PR Quadtree (for points)

- PR (Point-Region) Quadtree
- Regular decomposition (similar to region quadtree)
- Also independent of the order in which data points are inserted into it
- But if two points are very close, decomposition can be very deep
Example of PR Quadtree

Subdivide into quadrants until the two points are located in different regions.
PM Quadtree (for points + lines)

- PM (Polygonal-Map) quadtree family for points and Lines
  - PM1 quadtree, PM 2 quadtree, PM 3 quadtree, PM R quadtree, etc.

- PM1 Quadtree
  - Based on regular decomposition of space
  - Vertex-based implementation
  - Criteria
    - At most one vertex can lie in a region represented by a quadtree leaf
    - If a region contains a vertex, it can contain no partial-edge that does not include that vertex
    - If a region contains no vertices, it can contain at most one partial-edge
Example of PM1 Quadtree
PM Quadtree

PM1 quadtree

PM2 quadtree

PM3 quadtree
For Your Info: B-Tree Definition

A B-Tree of order $M$ is a height-balanced tree

1. All leaves are on the same level
2. All nodes have at most $M$ children
3. All internal nodes except the root have at least $M/2$ children
For Your Info: B-Tree Insertion

A new index entry

ChooseLeaf

Has room

Yes

Install X

No

SplitNode

AdjustTree