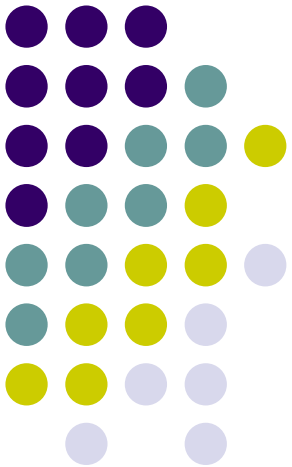
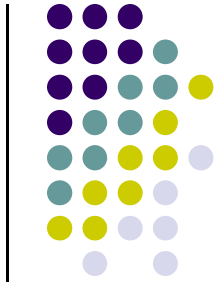
R-TREE AND SPATIAL DATABASE QUERYING

Nguyễn Quốc Thịnh - 7140027
Hồ Quang Chi Bảo - 7140219

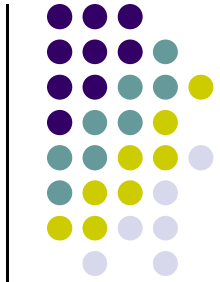


Contents



- Spatial problems in real word
- Indexing single-dimension data
- Indexing spatial data
- Summary
- References
- Q&A

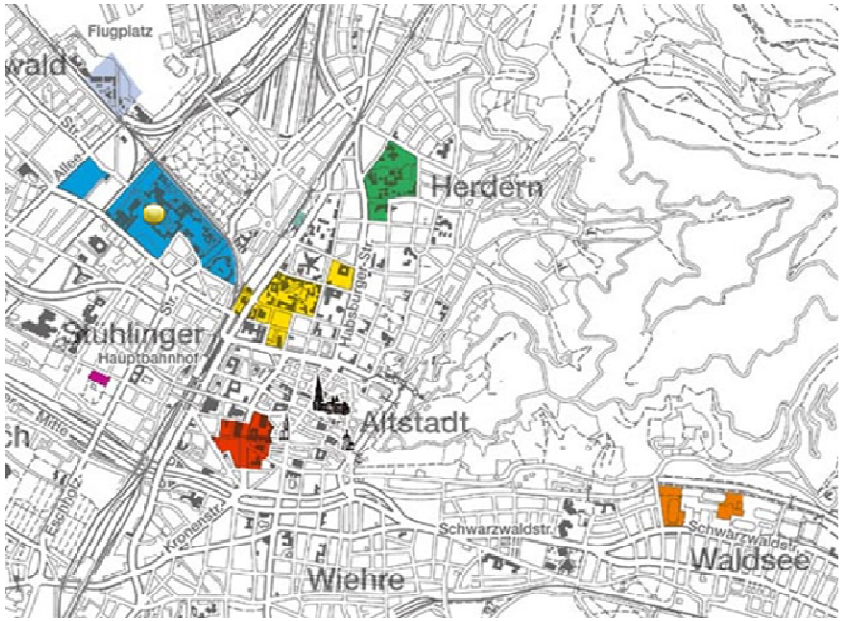
Outline



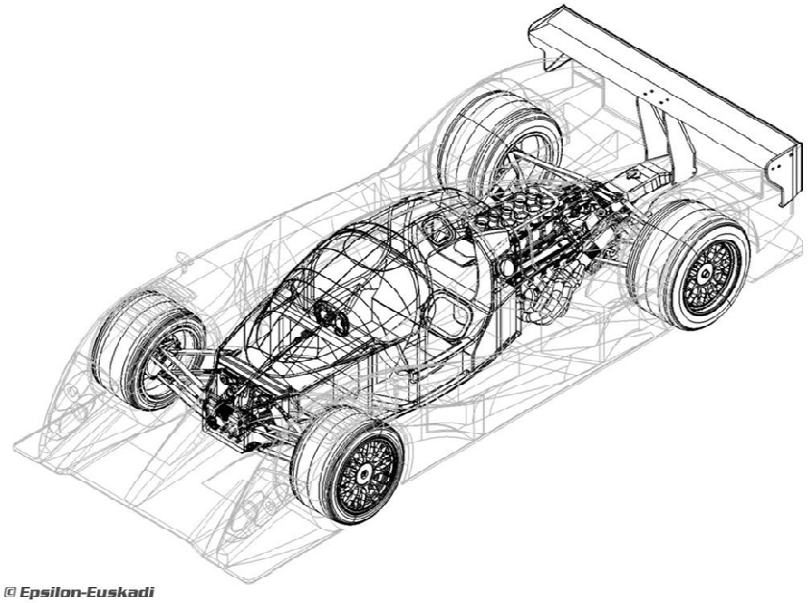
- Spatial problems in real word
- Indexing single-dimension data
- Indexing spatial data
- Summary
- References
- Q&A

Spatial data in real word

GIS (Geographic Information System)

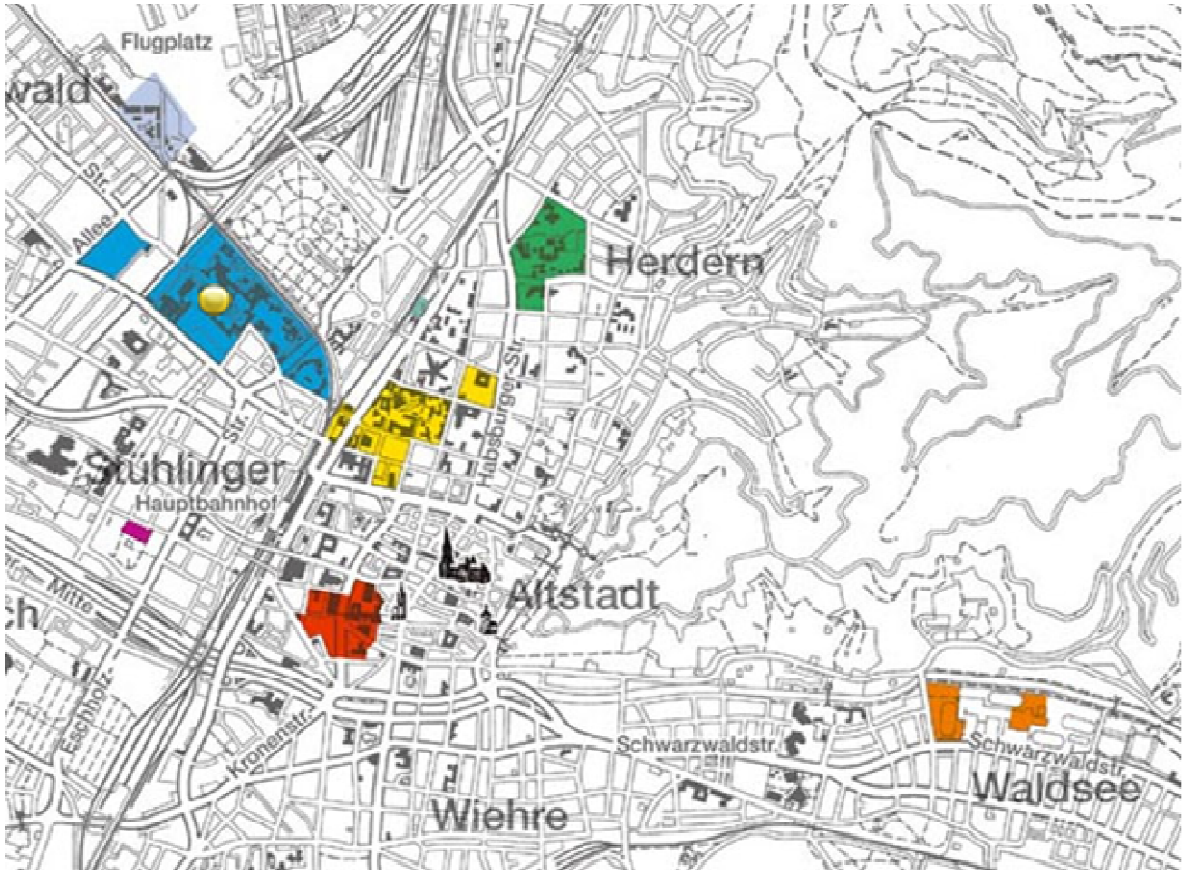


CAD (Computer Aided Design)

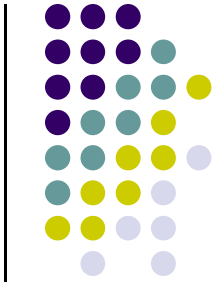


Simple problem

Find your university in your city

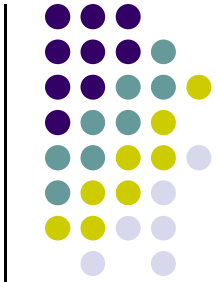


How to…



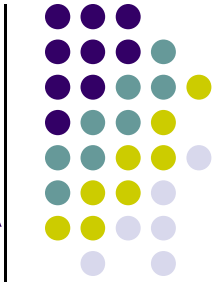
- Store spatial object into database?
  - Points, lines, surface,…
- Retrieve effectively a spatial object to answer a query quickly?
  - require a data structure to index new data objects.
- Review single-dimensional data indexing

Outline



- Spatial problems in real word
- Indexing single-dimension data
- Indexing spatial data
- Summary
- References
- Q&A

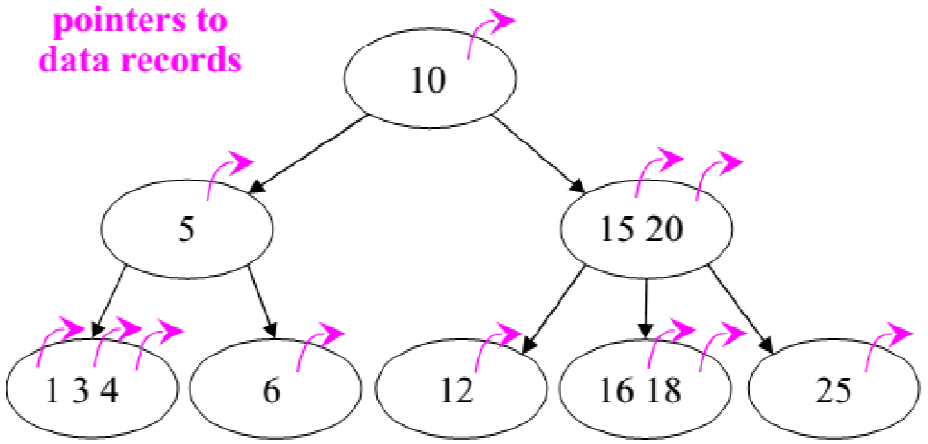
Indexing single-dimension data



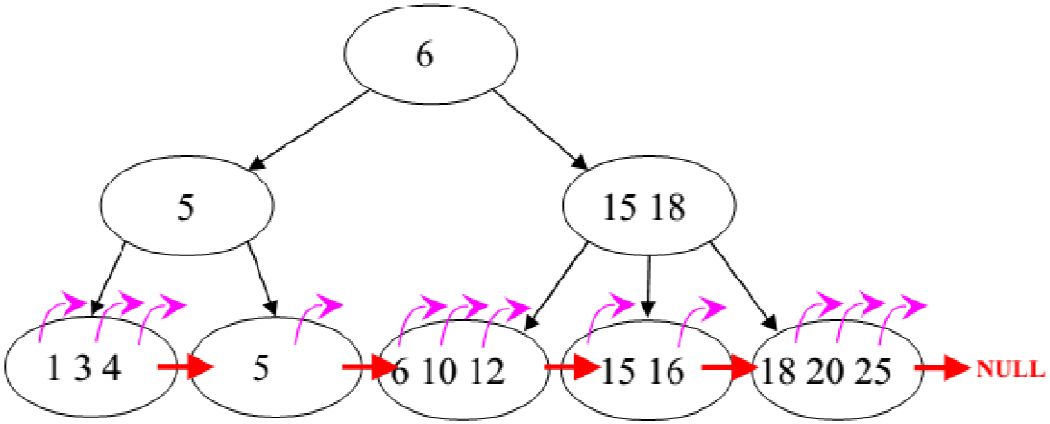
- Single dimension data:
  - integers, characters, strings, …
- Indexing data structure:
  - B-Tree
  - B^+-Tree

B-Tree vs. B+-Tree

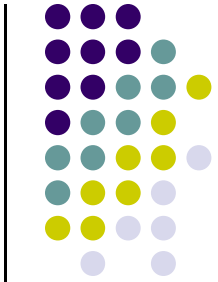
B-Tree



B+-Tree

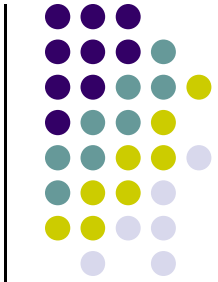


Outline



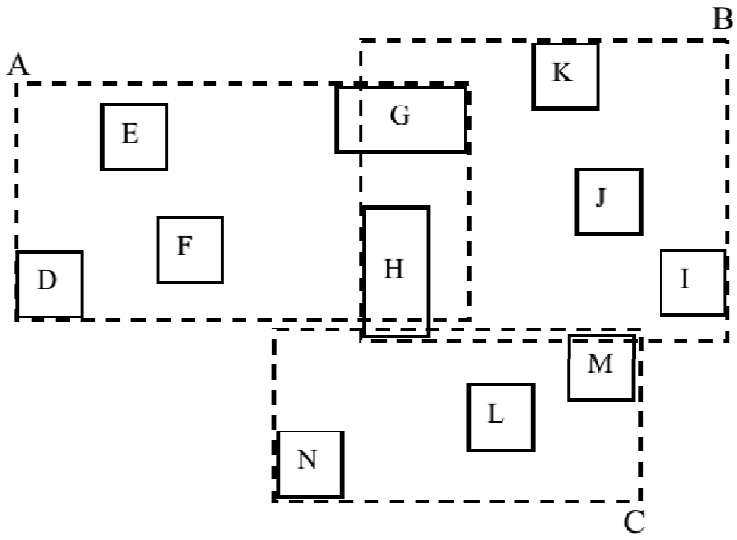
- Spatial problems in real word
- Indexing single-dimension data
- Indexing spatial data
- Summary
- References
- Q&A

Indexing spatial data

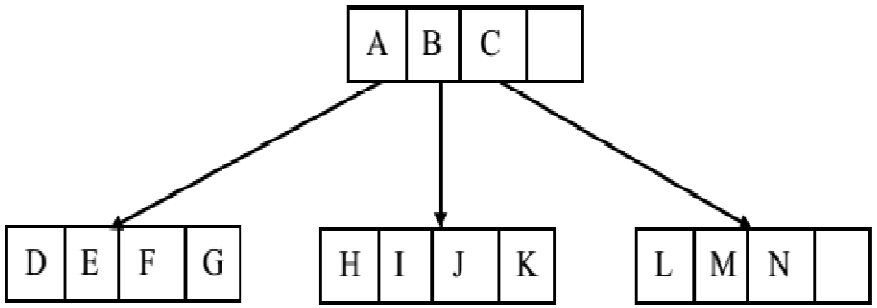


- Multi-dimession data:
  - Spatial data
- Indexing data structure: R-Tree
- Proposed by Antonin Guttman in 1984

R-Tree Example

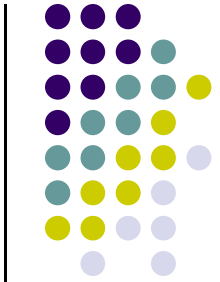


MBRs



R-Tree

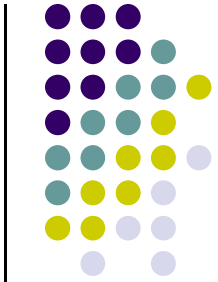
R-Tree



- Hierarchical data structures based on B^+-Tree
- **M**inimum **B**ounding n-dimensional **R**ectangle (MBR)
- Leaf node entry form (*mbr, object-id*)

- Internal node entry form (*mbr, child-pointer*)

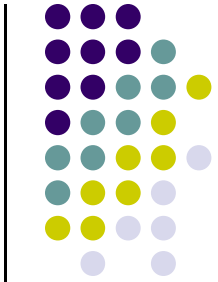
R-Tree characteristics



- Each node (unless it is root node): store from **m** to **M** entries
  - M: maximum number of entries
  - m: minimum number of entries
  - $m \leq M/2$
- Minimum allowed number of entries in root node is 2 (unless it is a leaf)

- All leaf nodes are at the **same level**

R-Tree Height

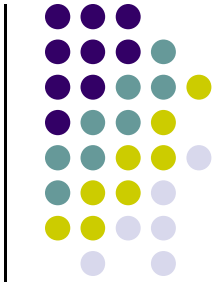


- Maximum height [1]

$$h_{\max} = \lceil \log_m N \rceil - 1$$

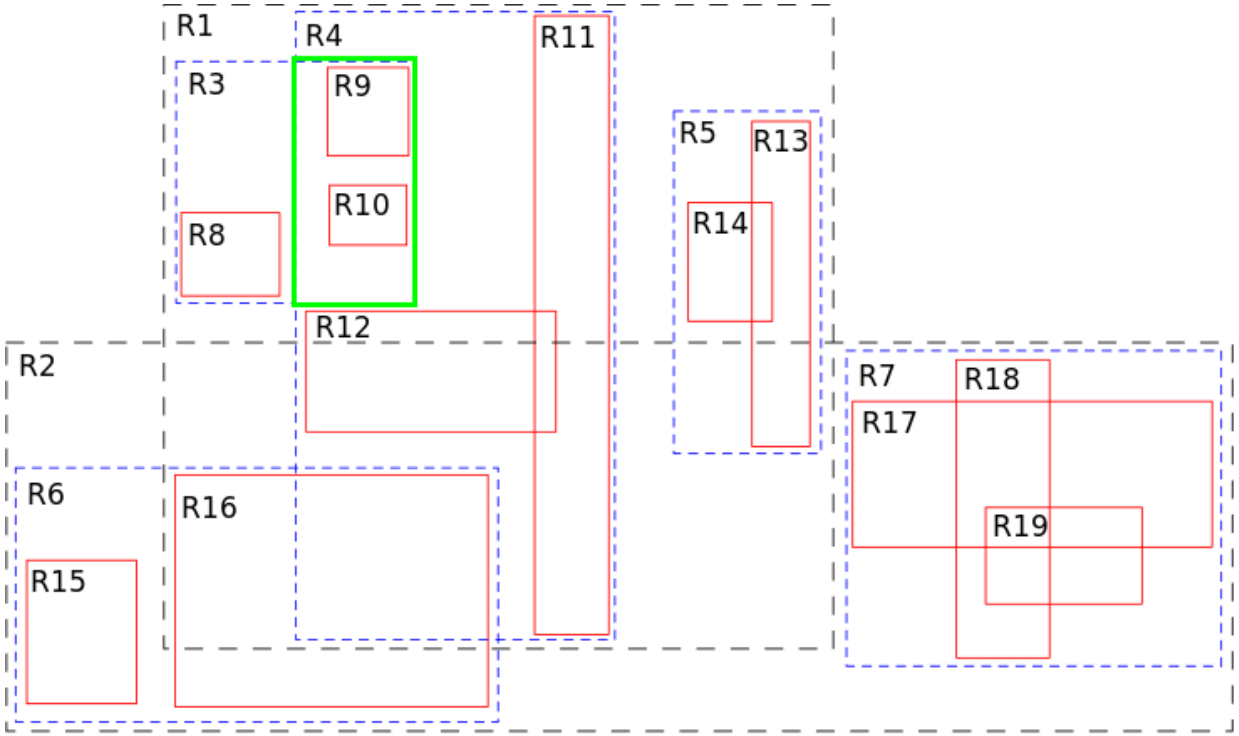
  - *m* : minimum allowed of entries
  - *N*: number of data rectangles

R-Tree Search

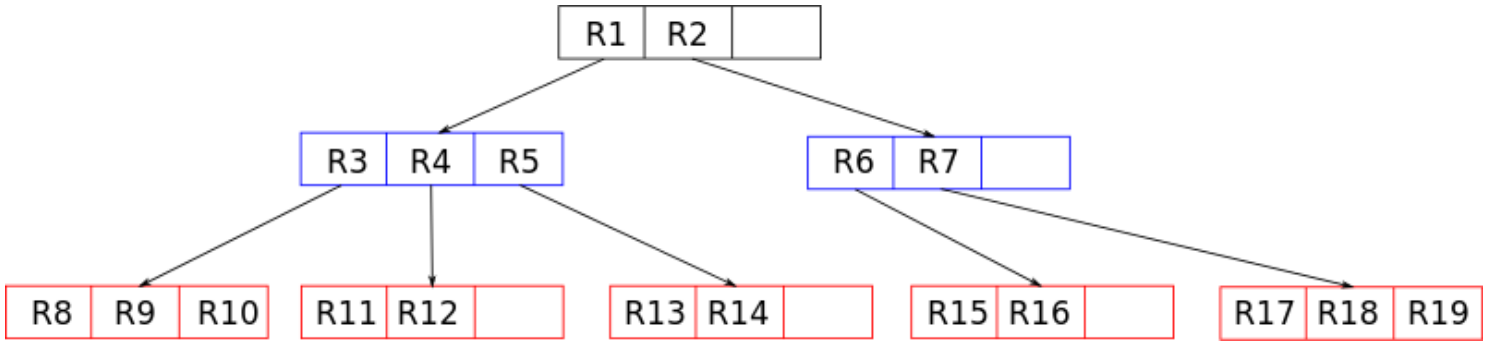


- Range query [1]
  - Find all data rectangles that are intersected by input rectangle Q
  - Result of this query just for filter step

R-Tree Search

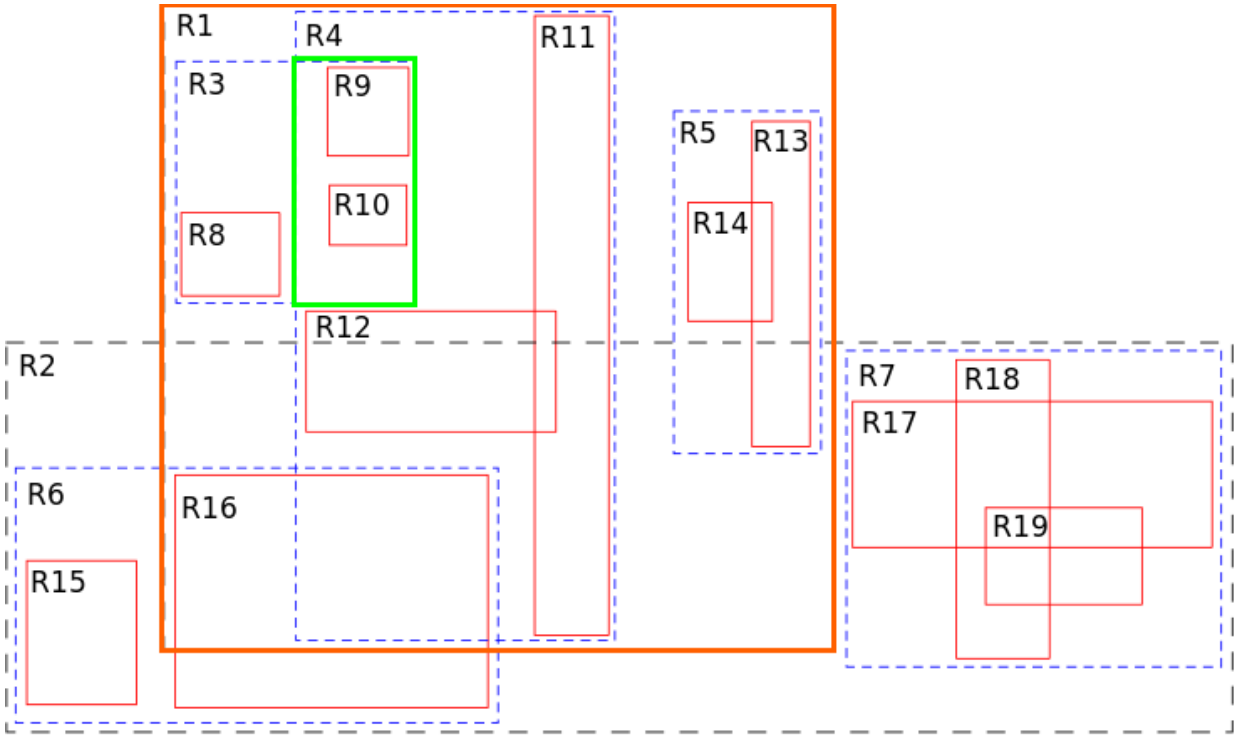


MBRs

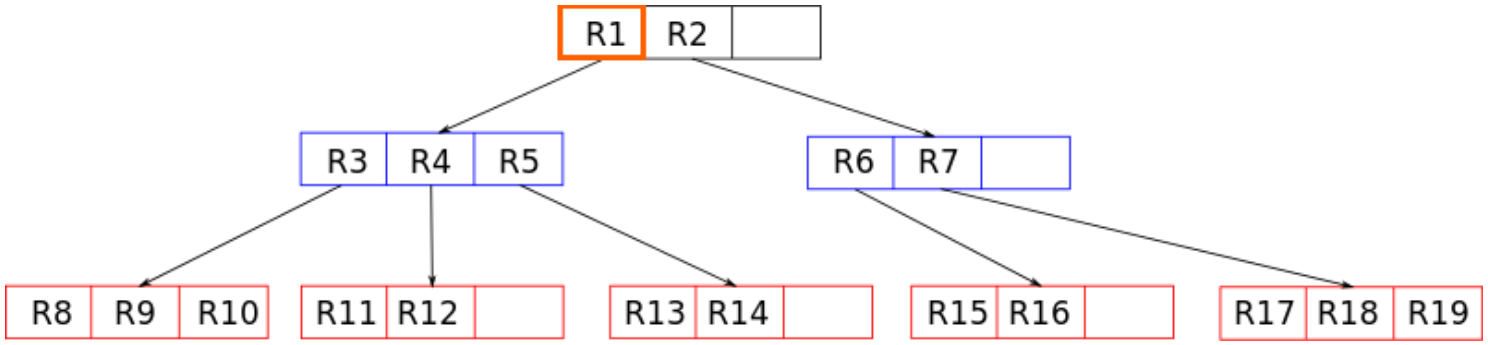


R-Tree

R-Tree Search

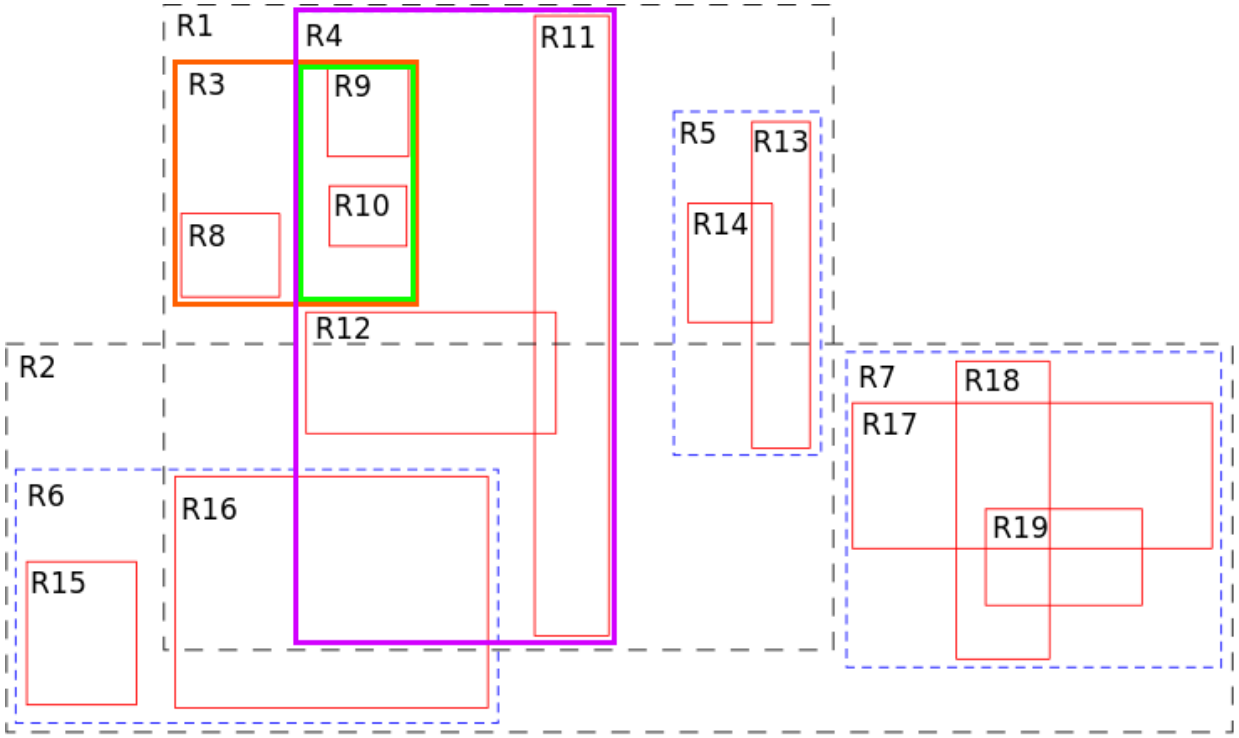


MBRs

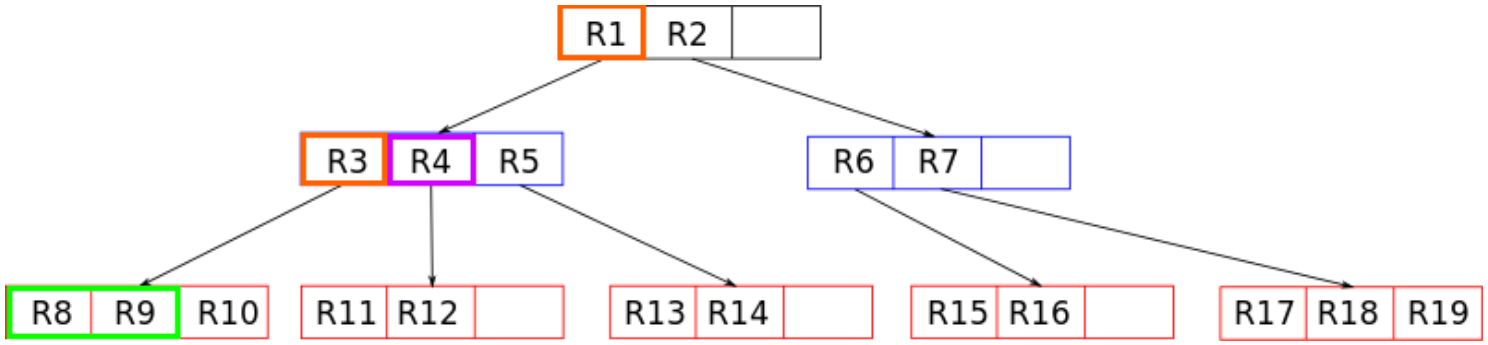


R-Tree

R-Tree Search

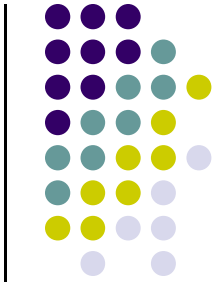


MBRs



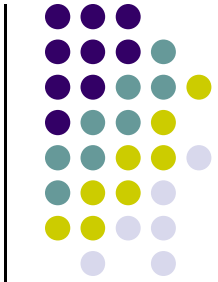
R-Tree

R-Tree Insertion

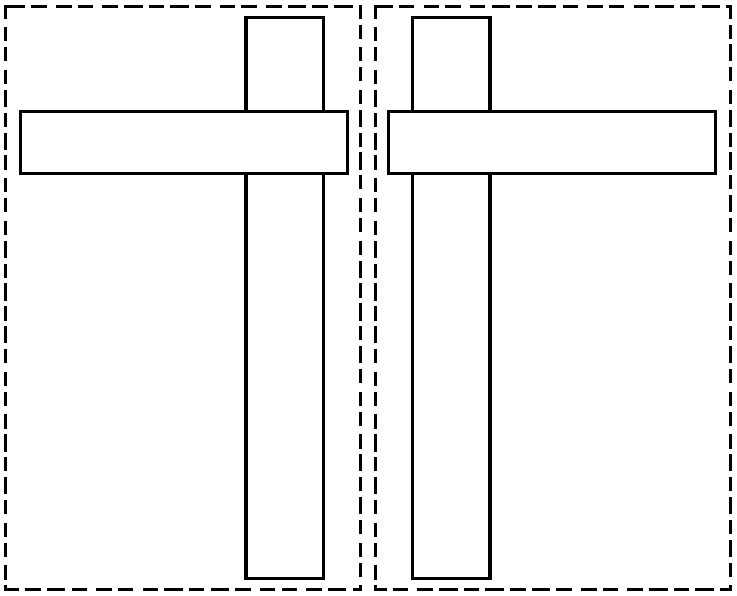


- Similar to insertions in B^+-Tree
  - Traverse R-Tree to locate a leaf can hold new entry
  - Found leaf available:
    - Insert new entry into found leaf
    - Update all node within the path from root to that leaf
  - Found leaf unavailable:
    - Split node
    - How to split?

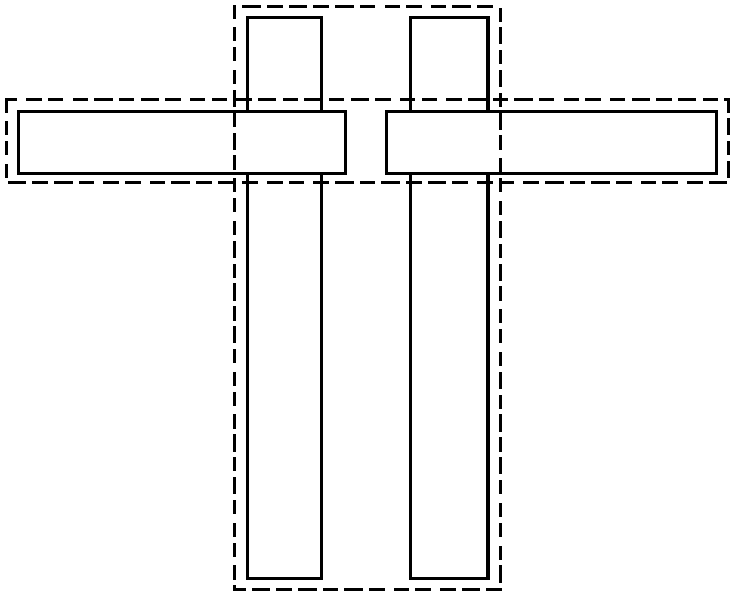
R-Tree Splitting



- The total area of the two covering rectangles after a split should be minimized.

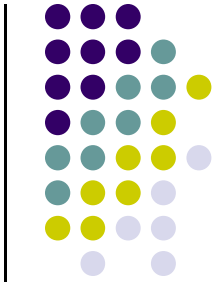


Bad split



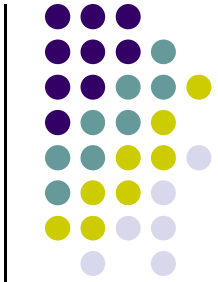
Good split

R-Tree Splitting

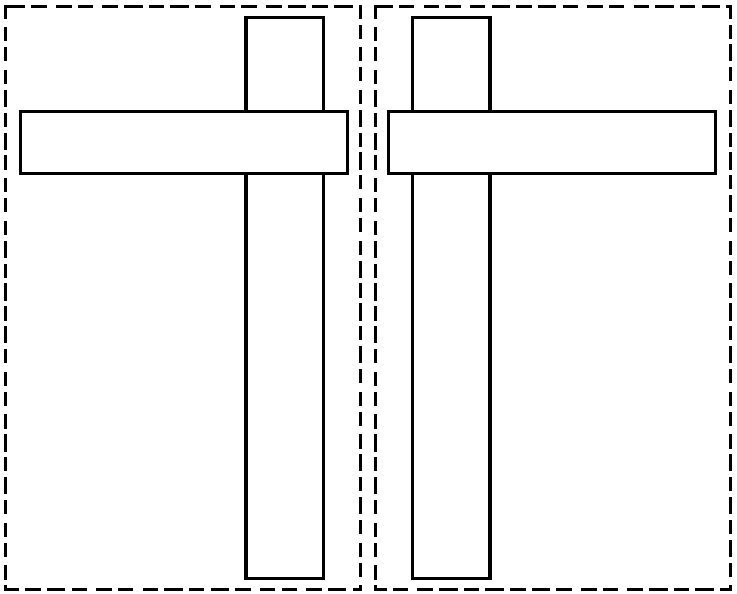


- 3 Splitting Algorithms (Guttman)
  - Linear split
  - Quadratic split
  - Exponential split

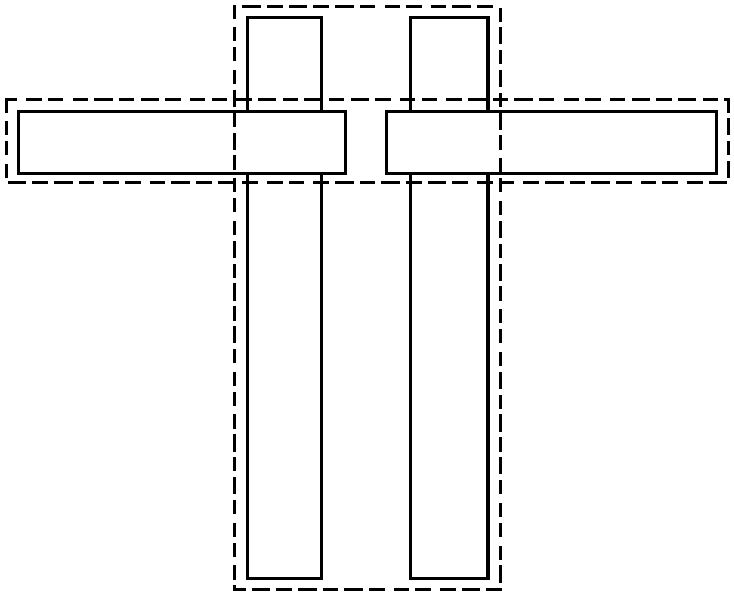
R-Tree Splitting



- Linear Split
  - Select objects are farthest apart
  - Require the smallest enlargement respective MBR

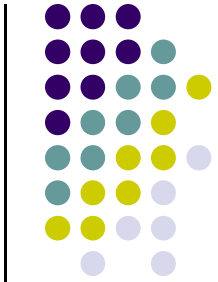


Bad split

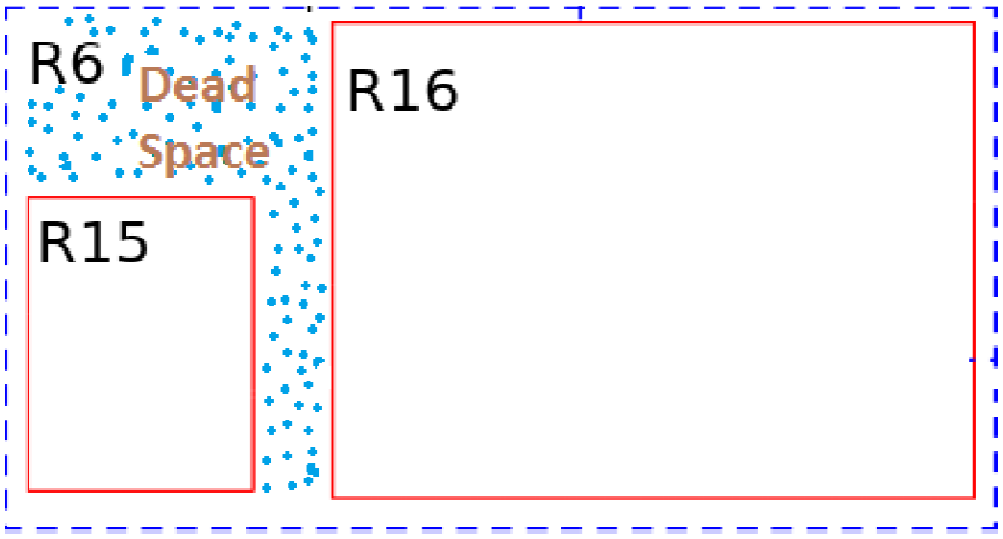


Good split

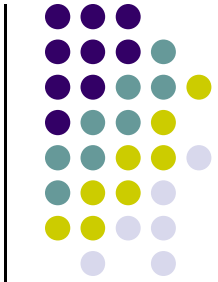
R-Tree Splitting



- Quadratic Split
  - Select 2 objects as seeds for the two nodes
  - Selected objects if put together create as much dead space [1] as possible

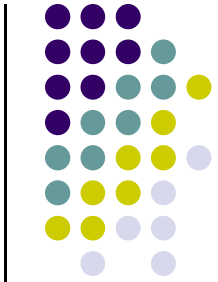


R-Tree Splitting



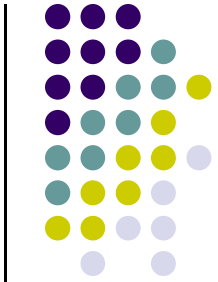
- Exponential Split
  - Test all possible groups
  - Choose minimization of the MBR enlargement

R-Tree Splitting



- Which splitting algorithms should be used?
  - Guttman suggested using the Quadratic splitting
- How about deleting?

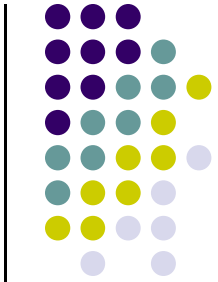
R-Tree Deletion



- Similarly to deletion in B^+-Tree

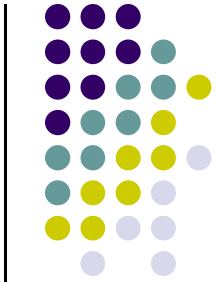
- Handling underflowing node
  - B^+-Tree: merging two sibling nodes
  - R-Tree: reinsertion

R-Tree Disadvantage



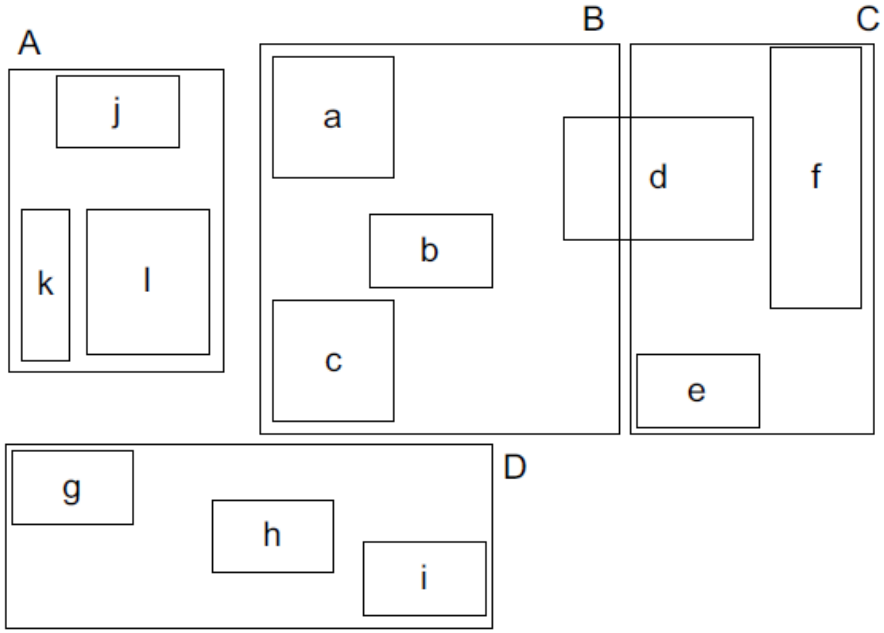
- Multiple paths from the root to the leaf level
  - Overlap MBRs
- Focus only minimized enlargement MBR
- How to improve them?

R^+-Tree

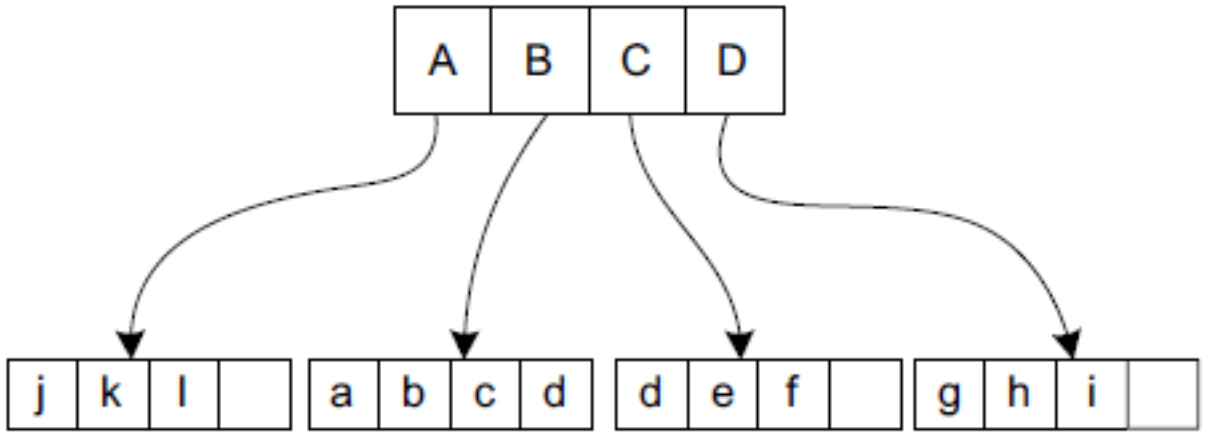


- Proposed in 1987 by Sellis, Roussopoloulos, Faloutsos
- Fix the overlapping in R-Tree

R+-Tree

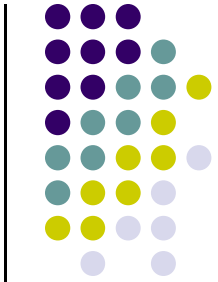


MBRs



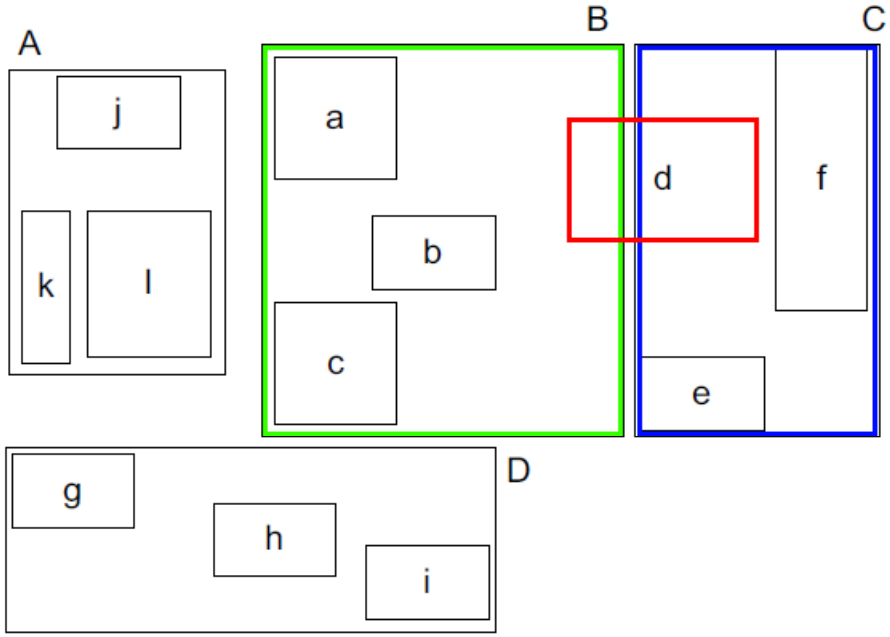
R+-Tree

R^+-Tree

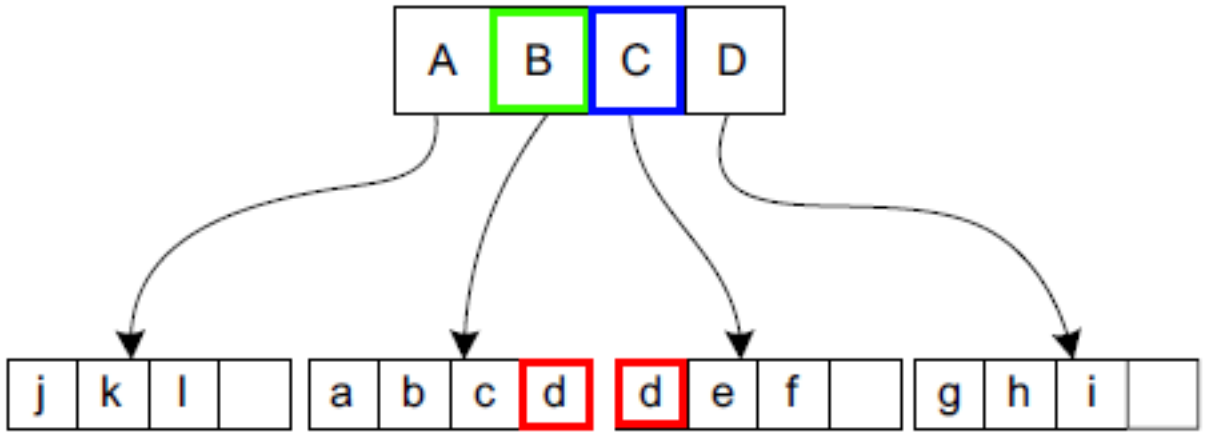


- Do not allow overlapping of MBRs at the same tree level
- May duplicate object's entries

R+-Tree

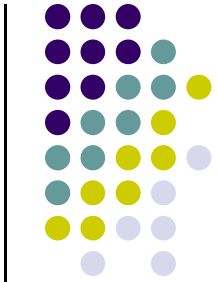


MBRs



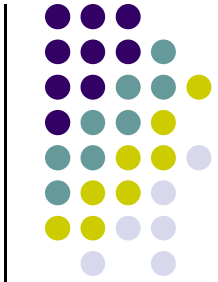
R+-Tree

R^+-Tree Disadvantage



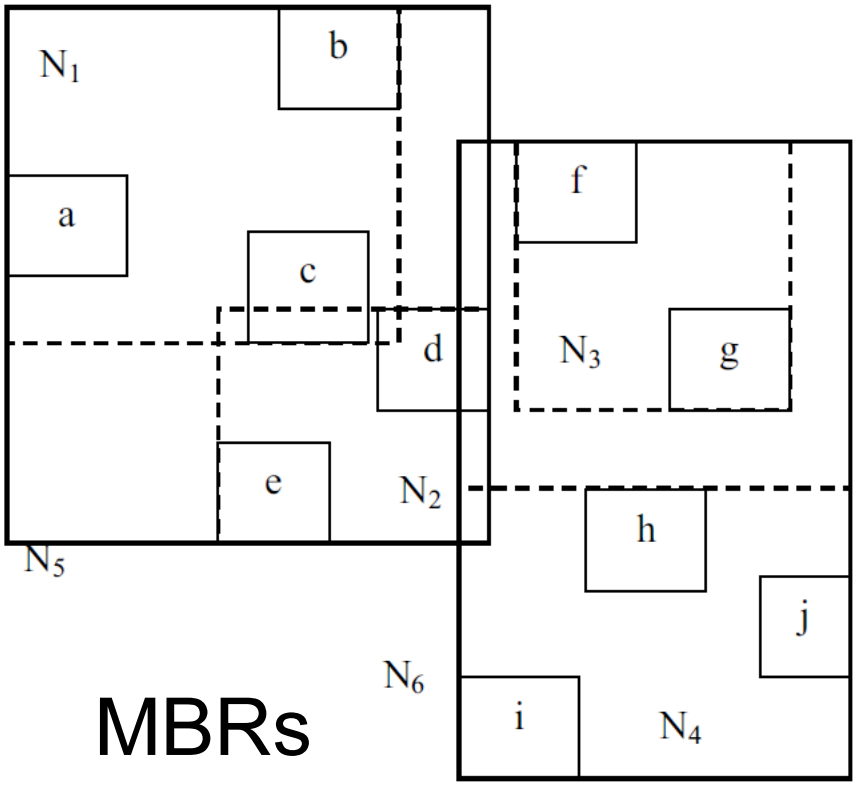
- Redundantly stored in several nodes
- Performance with range queries

R*-Tree

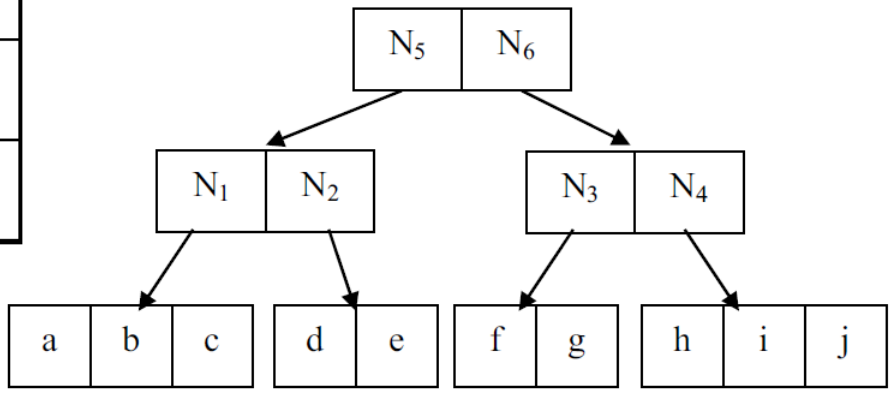


- Proposed in 1990 Beckmann, Kriegel, Schneider, Seeger
- To improve the performance during query processing

R*-Tree

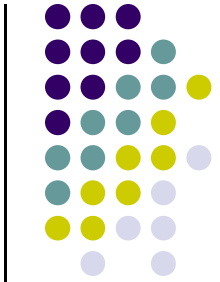


MBRs



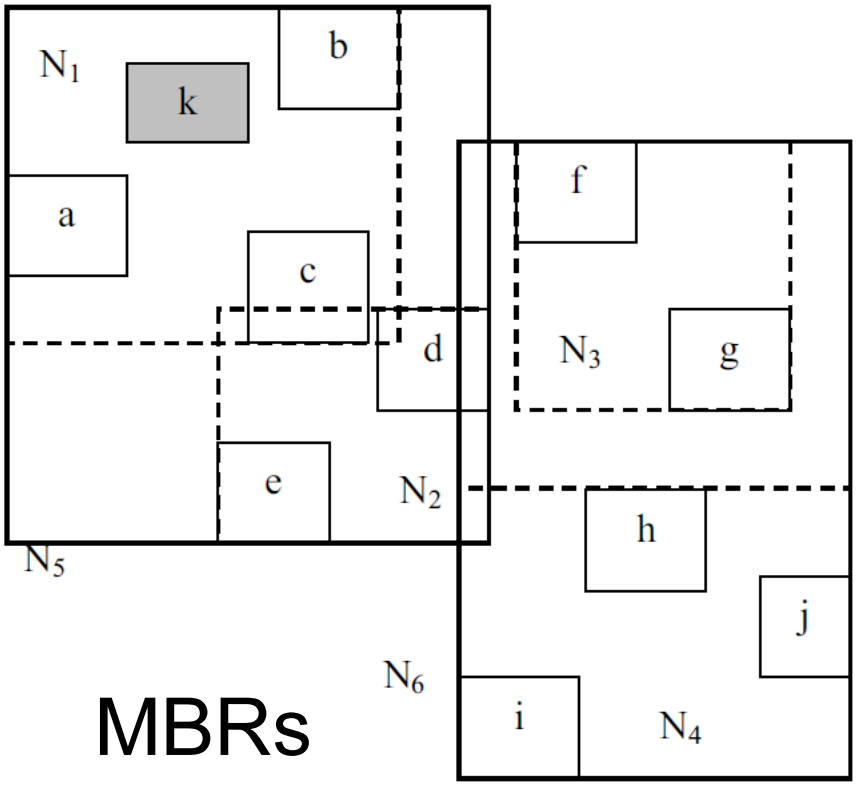
R*-Tree

R*-Tree

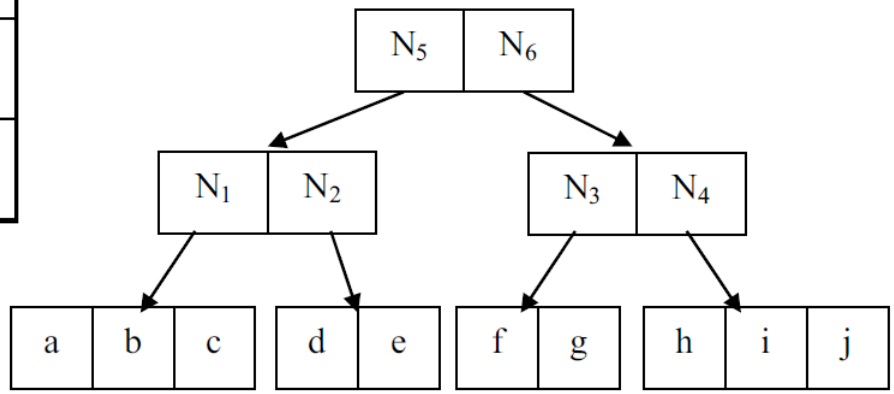


- Minimization of the area covered by each MBR (1)
- Minimization of overlap between MBRs (2)
- Minimization of MBR margins (3)
- Maximization of storage utilization (4)

R*-Tree Insertion

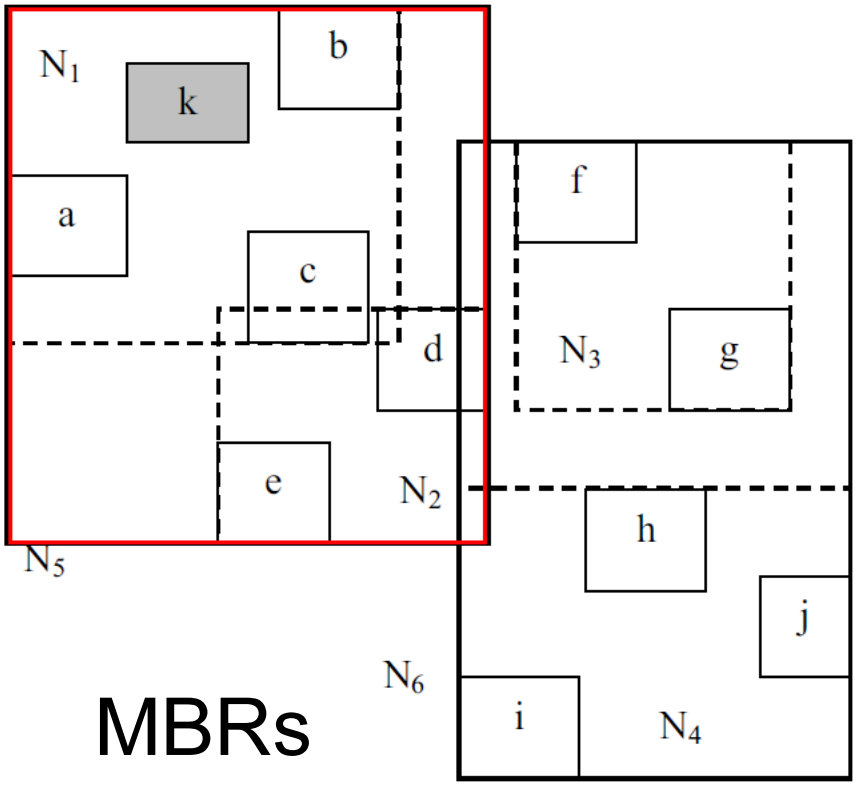


MBRs



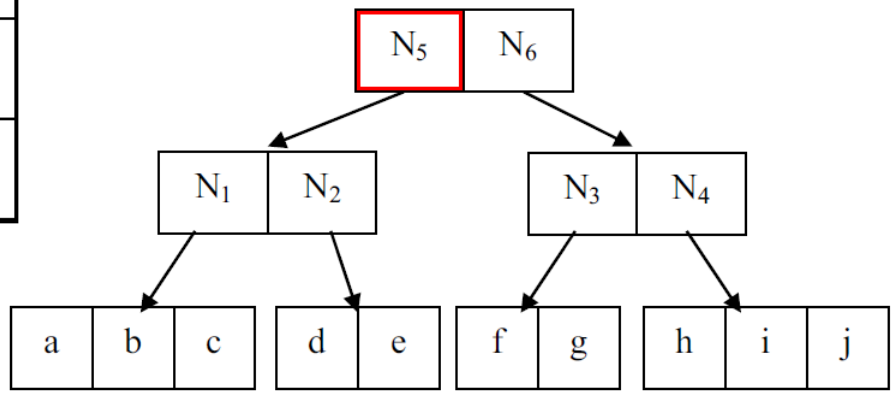
R*-Tree

R*-Tree Insertion



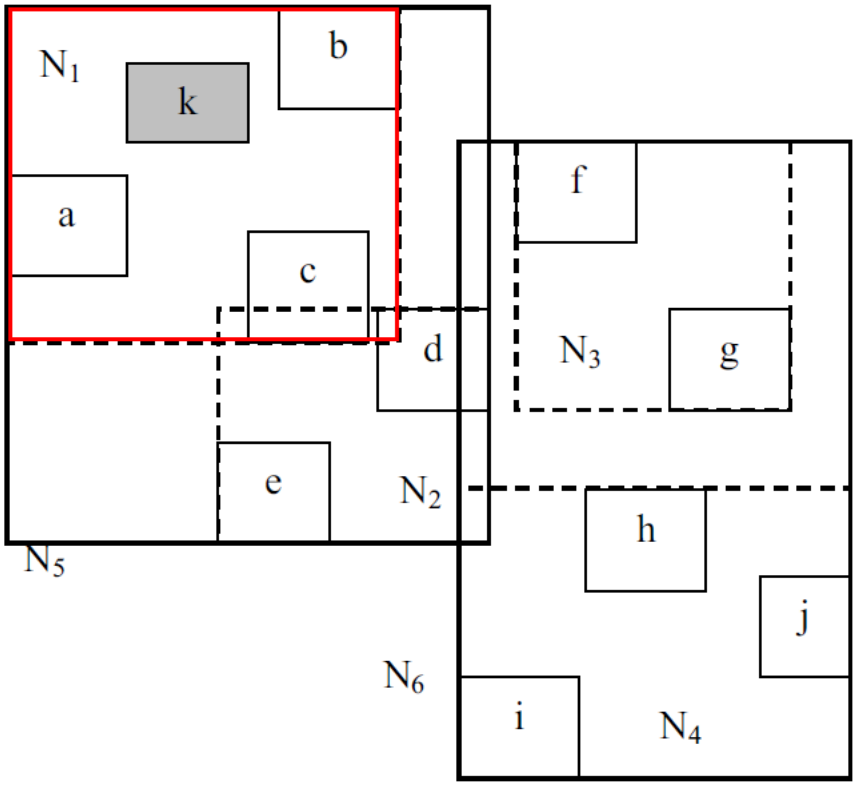
MBRs

(1)

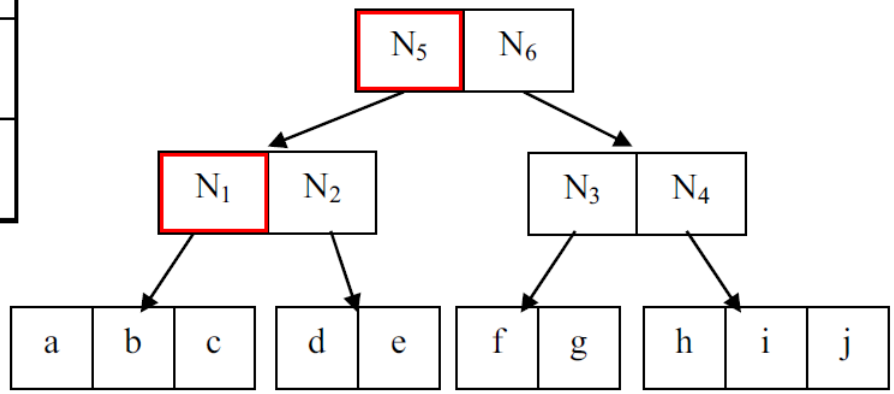


R*-Tree

R*-Tree Insertion

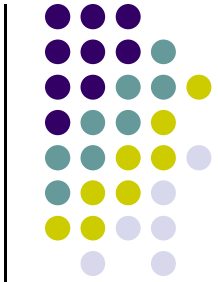


(1)



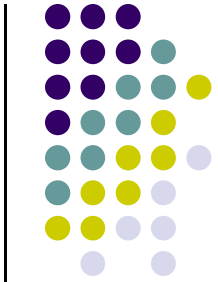
R*-Tree

R*-Tree Insertion

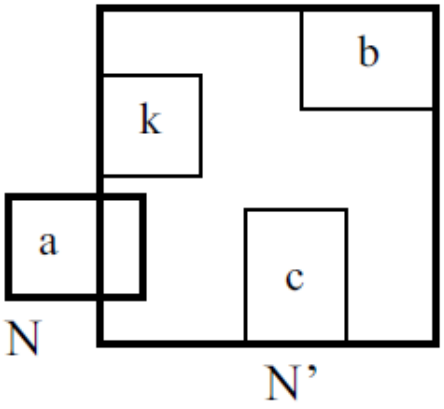


- N1 overflow
  - Do not immediately resort to node splitting
  - Reinsertion
    - Get 30% entries whose centroid distances from node centroid are among the largest
    - Entry b is selected
- Reinsertion Failed → Split

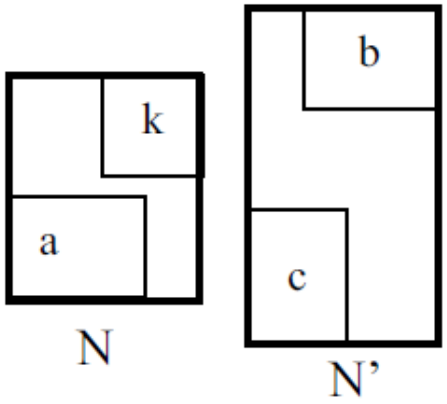
R*-Tree Splitting



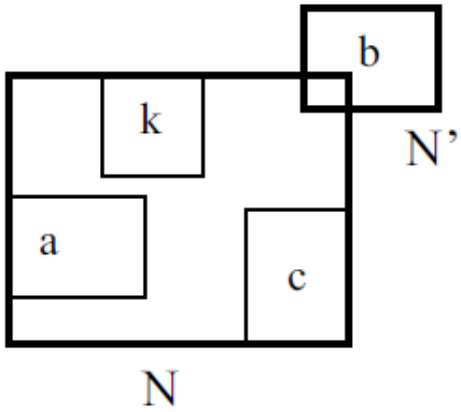
(3)



1-3 division



2-2 division

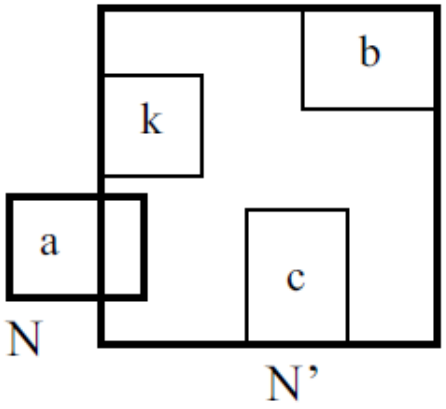


3-1 division

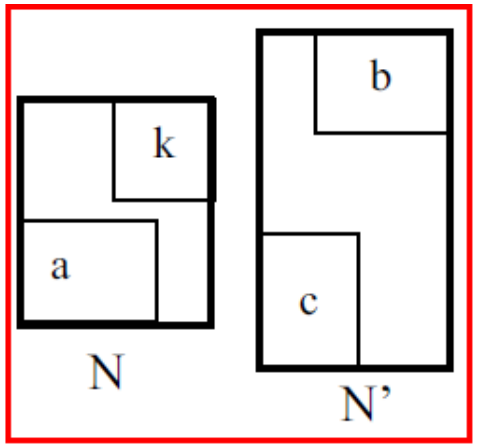
x-axis

R*-Tree Splitting

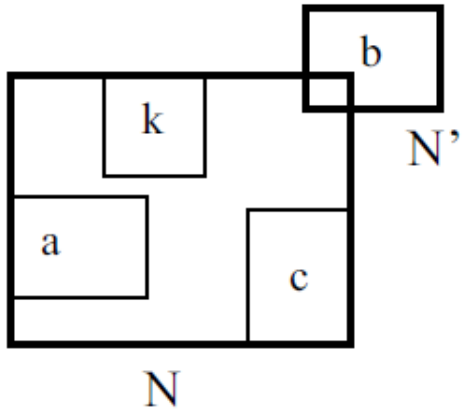
(2)



1-3 division



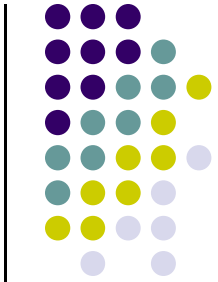
2-2 division



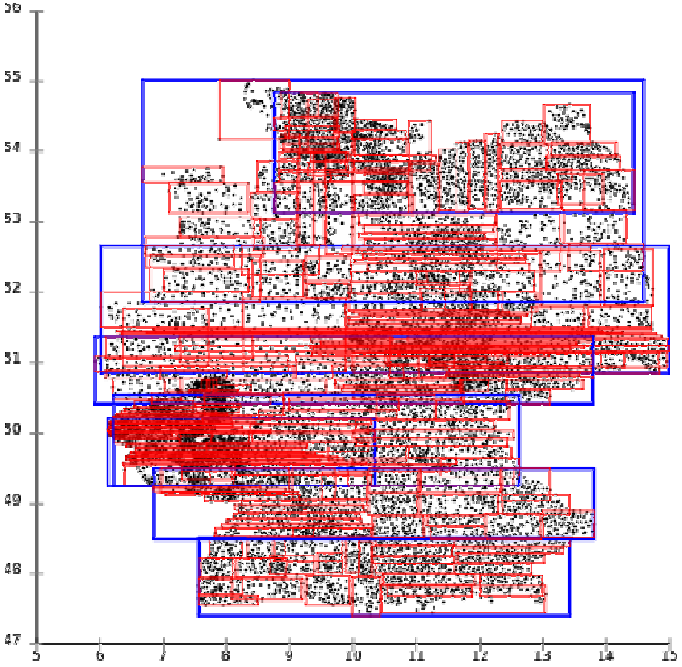
3-1 division

x-axis

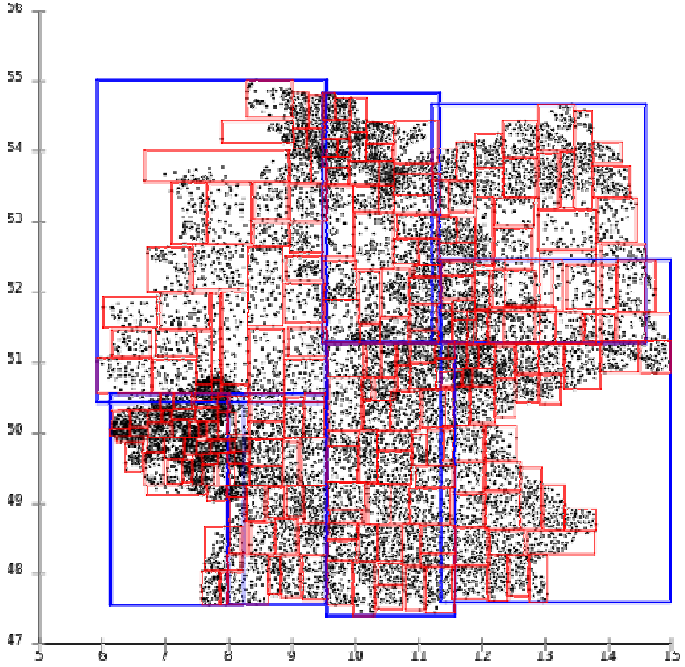
Splitting in R-Tree vs R*-Tree



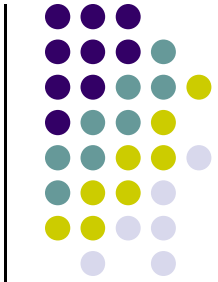
R-Tree [2]



R*-Tree [2]

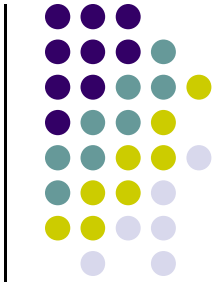


R*-Tree Deletion



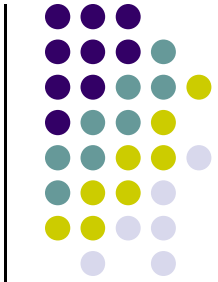
- Deletion in the R*-*Tree:*
  - *Same with the deletion algorithm* of the original R-tree

Outline



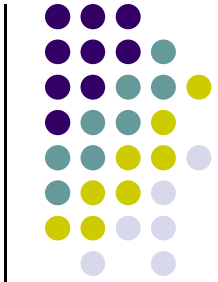
- Spatial problems in real word
- Indexing single-dimension data
- Indexing spatial data
- Summary
- References
- Q&A

Summary

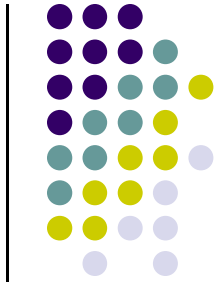


- Introducing spatial data
- Reviewing B-Tree, B^+- Tree for single-dimension data indexing
- R-Tree, R^+-Tree, R*-Tree for spatial data indexing
- R-Tree is one of fundamental data structures to index multi-dimension data.

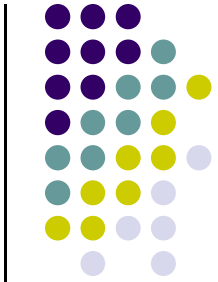
References



- [1] Yannis, M., Alexandros N., *R-Tree:Theory and Applications*, Springer, 2006
- [2] http://en.wikipedia.org/wiki/R*_tree
- [3] http://en.wikipedia.org/wiki/R-tree
- [4] http://www.itgsnews.com/2012/04/grade-12-revision-cad-and-cam.html
- [5] http://20bits.com/article/interview-questions-database-indexes



Q&A



Thank you for listening!