The R*-tree: An Efficient and Robust Access Method for Points and Rectangles

N. Beckmann  H. P. Kriegel  R. Schneider  B. Seeger

Praktische Informatik, Universitaet Bremen

February 23, 2013

Presented by Zhitao Gong
Outline

1 Introduction

2 Reasonale for R*-tree
   • Principles of R-tree
   • Possible Optimizations Criteria
   • Problem and solution

3 Detailed Optimization

4 Performance Comparison
   • Intuitive Example of Optimization
   • Dataset
   • Experiment Results

5 conclusion
Principles of R-tree I

General constraints

- Non-leaf node \((cp, Rectangle)\)
- Leaf node \((O_{id}, Rectangle)\)
- Capacity number of rectangles contained in one node
  - \(2 \leq N \leq M\) for root and
  - \(m \leq N \leq M\) for non-root

where \(2 \leq m \leq M/2\)

- Height all leaves on the same level
Principles of R-tree II

Dynamic Features

- Insertions and deletions mixed with queries
- No periodic global re-organization
- Different sequences of insertion build up different trees
R-tree Example

Figure: R-tree example
Optimizations Criteria I

1. Minimizing dead space

Figure: Dead Space for R1
Optimizations Criteria II

2. Minimizing overlap

Figure: Overlap between R1 and R2
Optimizations Criteria III

3. Minimizing margin

4. Optimize storage utilization
Problem and solution I

Scenario

- **Goal** Efficient retrieval
- **Criteria** 4 possible optimizations
Problem
The known parameters of good retrieval performance affect each other in a very complex way.
What we want:
\[ \text{deadspace} \downarrow, \text{overlap} \downarrow, \text{margin} \downarrow, \text{storage utilization} \uparrow \Rightarrow \text{performance} \uparrow \]

- deadspace\downarrow \text{ and overlap} \downarrow \Rightarrow \text{storage utilization} \downarrow
- deadspace\downarrow \text{ or overlap} \downarrow \Rightarrow \text{margin} \uparrow
Solution

Use an engineering approach to find the best possible combination
Detailed Optimization

InsertData Algorithm

Figure: InsertData Algorithm
Optimization of *ChooseSubtree*

**R-tree**
- least area enlargement
- smallest area
Optimization of \textit{ChooseSubtree}

\textbf{R-tree}
- least area enlargement
- smallest area

\textbf{R*-tree}
- N points to leaves, determine the minimum overlap cost
  - least overlap enlargement
  - least area enlargement
  - smallest area
- Otherwise, determine the minimum area cost
  - area enlargement
  - smallest area
Intuitive Example of Optimization

Figure: Overfilled
Intuitive Example of Optimization

Figure: Overfilled

Figure: Quad-R-tree
Intuitive Example of Optimization

Figure: Overfilled

Figure: Quad-R-tree

Figure: R*-tree
Data file and Query file

Data file
- Uniform
- Cluster
- Parcel
- Real-data
- Gaussian
- Mixed-Uniform

Query file
- rectangle intersection query
- point query
- rectangle encosure query
Results for Uniform I

Disk Accesses

Figure: Normalized disk accesses data
Results for Uniform II

Storage Utilization

Figure: Storage Utilization comparison
Results for Uniform III

Disk Accesses Per insertion

Figure : Disk accesses per insertion
Conclusion

- Efficient SAM and PAM
- Outstanding performance
- Generalizing to polygons