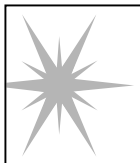
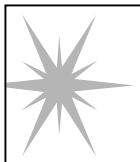


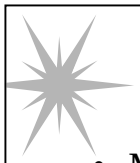
Multidimensional Access Methods
Trees Have Grown Everywhere

Timos Sellis
Nick Roussopoulos
Christos Faloutsos



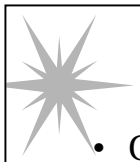
Outline

- Introduction
- The way we were 10 years ago
- What has been done since then?
- What was the influence?
- What is coming up?
- Summary



Introduction

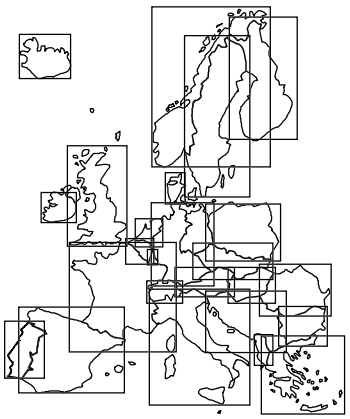
- Multidimensional Access Methods have been suggested for several applications
  - spatial databases
  - GIS
  - multimedia databases
  - ...even in traditional databases, active DBs and data warehousing
- Emphasis on spatial databases
- Can index large collections of multidimensional data .....but are they scalable, adaptable, fast?



The way we were 10 years ago

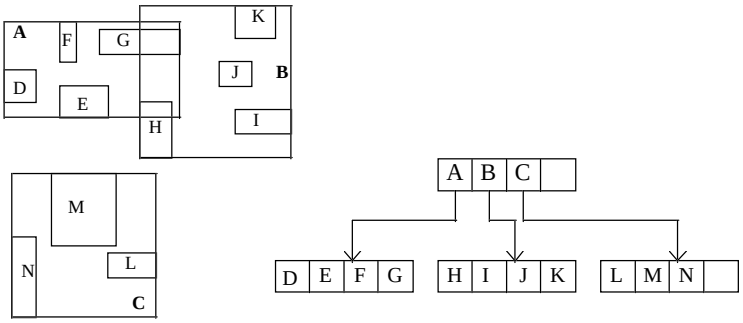
- Guttman first proposed the R-tree in Sigmod 84
  - Extends B-trees to N dimensions
  - Handles non-zero area objects as well as points
  - Approximate objects with Minimum Bounding Rectangles
- There were other proposals too
  - Mostly based on hashing
  - Also some tree-based: K-D tree, K-D-B-tree, etc.
  - Space filling curves: map 2-dimensional space to one dimensional and use B-trees
- The R-tree was the starting point of our work also (Packed R-trees Sigmod 85)

Minimum Bounding Rectangles

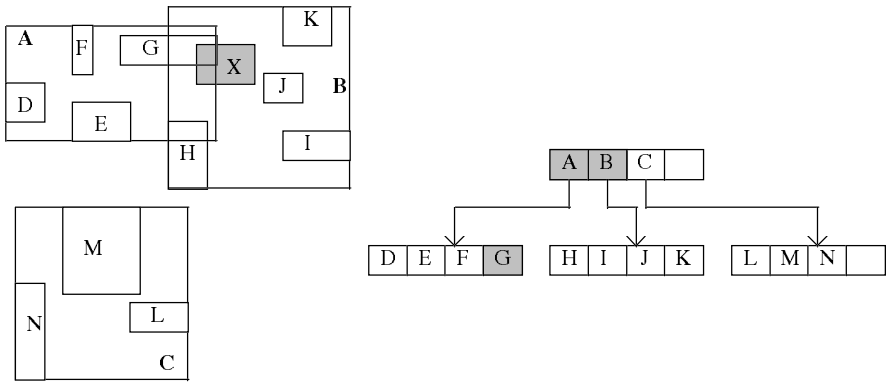


R-trees

R-tree [Guttman, 1984]



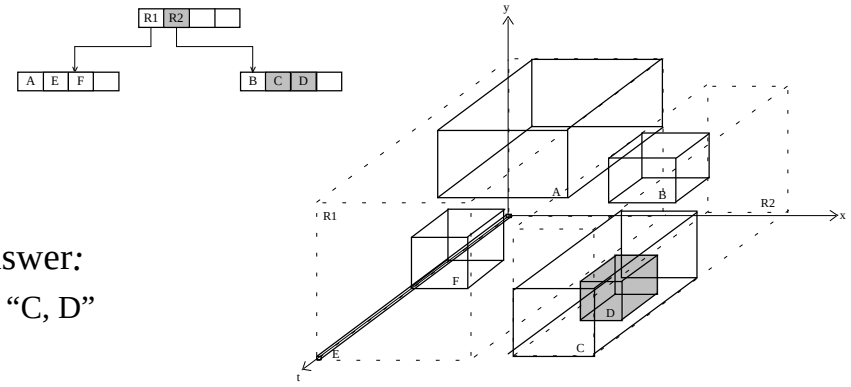
Range Query on R-trees



Range Query on R-trees

- Question*:*

  "Find all objects overlapping with D in space and time (spatio-temporal query)

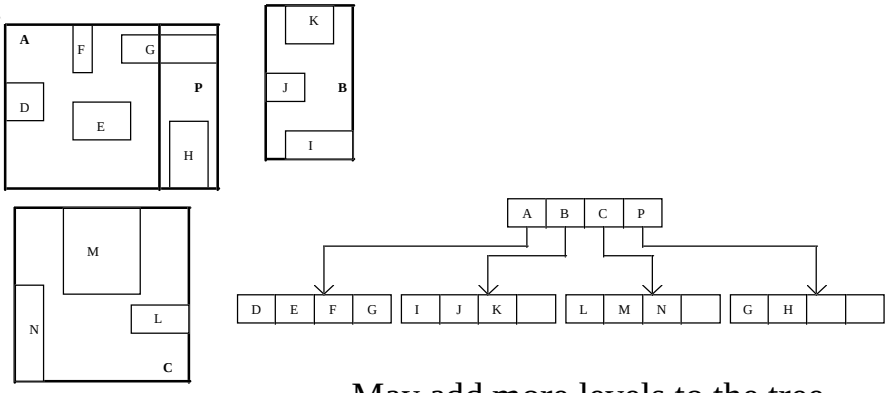


- Answer*:*
  - "C, D"

Packed R-trees

- Problems with random insertions
- Goal:
  - minimal coverage of leaf nodes
  - minimal overlap of intermediate nodes
- Sorting & packing of spatial objects improves search & space performance by 1-2 orders of magnitude
- Starting point for R+-trees, R*-trees, Hilbert R-trees, & Cubetrees

R$^+$-trees



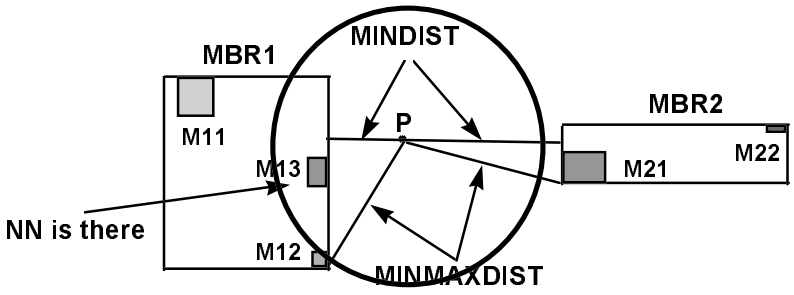
May add more levels to the tree
.....but it is faster

What has been done since 1987?

- Lots of other improvements and extensions to the basic structure (R*-tree, Hilbert R-tree, TV-Tree, P- and JP-Tree, and many more)
- Commercial systems are incorporating them
- Has given rise to lots of interesting other research
  - More packing algorithms
  - Spatial joins
  - Direction queries
  - Parallelization
  - Nearest-neighbor queries
  - Analysis of algorithms and structures

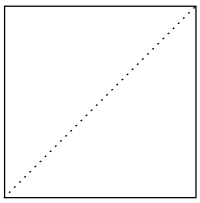
Nearest Neighbor Searching

downward pruning



Analysis of R-trees

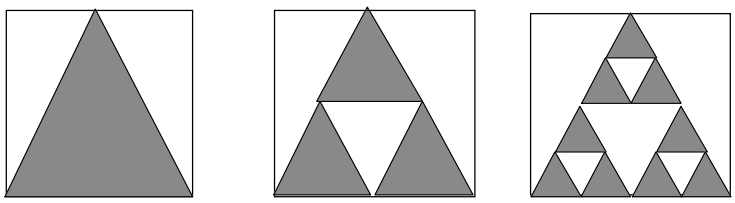
- Uniformity assumption
- BUT: pessimistic + unrealistic
- Solution: FRACTALS - What is the fractal dimension?
- ≈ “intrinsic” dimensionality
  - Nominal dimension = 2
  - “Intrinsic” dimension = 1



Analysis of R-trees

Non-integer fractal dimensions

- e.g. sierpinski triangle



- fractal dimension = log3/log2 = 1.59

Analysis of R-trees

Are real data sets fractal?

- Coastlines (fd = 1.1 - 1.58 - e.g. Norway !)
- Mamalian brain surface (2.7)
- Cardiovascular system (3!)
- Stock prices (1.5)
- “Montgomery County” (~1.7)

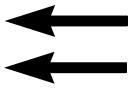
Analysis of R-trees

End result: Great accuracy in estimations !

- Range queries in R-trees (<10% error versus ~20% of uniformity)
- Spatial joins (<10% vs 100%)
- Nearest-neighbors (good bounds)

What was the influence?

- More than a decade of very active and intense research and development
- "Reaching out" to other areas, e.g.
  - active database systems
  - indexing multimedia databases by content
  - supporting OLAP and DataCube processing
  - indexing time sequences
  - data mining and clustering



Multimedia Search by Content

Main idea: "**GEMINI**" (GEneralized Multimedia INdexIng)

1. Extract a few (n) numerical features
2. Represent objects as n-dimensional points
3. Use a multidimensional access method

Multimedia by content

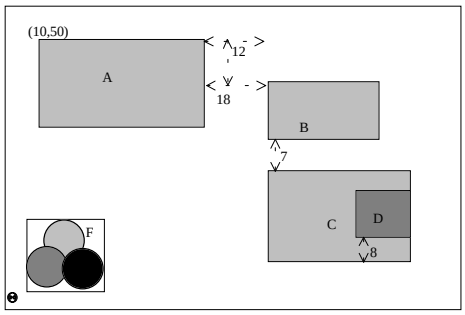
Used for

- Color images - e.g. in QBIC: *10 sec → 4 sec*
- Shapes (20 moments of inertia) :

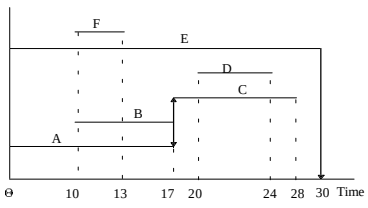
  order of magnitude speed-up
- Medical tumor shapes (pattern spectrum) :

  27 times faster
- Time sequences
- …..

Indexing Multimedia Presentations



spatial layout



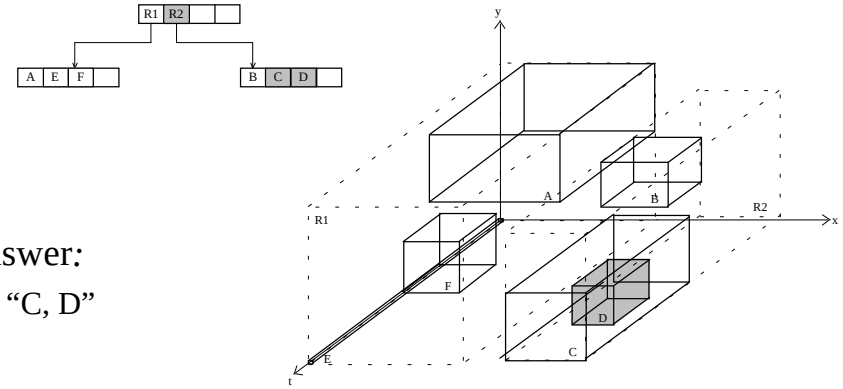
temporal layout

queries *spatial - temporal - spatio-temporal*

Range Query

- Question:

  "Find all objects overlapping with D in space and time (spatio-temporal query)



- Answer:
  - "C, D"

OLAP - Cubetrees

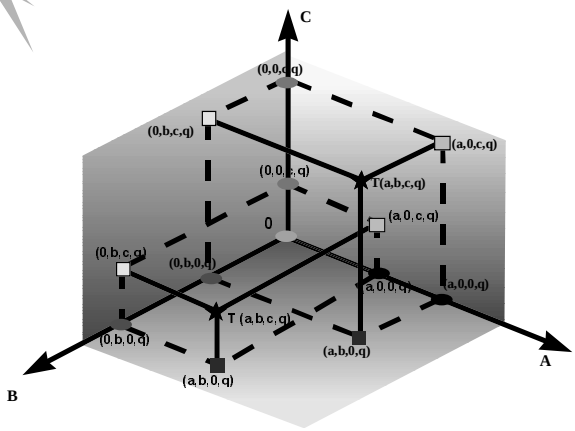


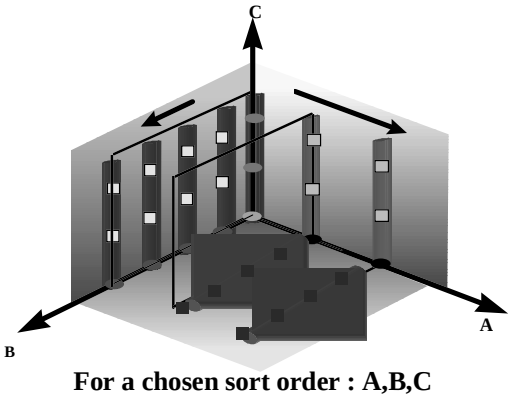
Table R(A,B,C,Q)

- **Relation tuple**
- **groupby(A,C)**
- **groupby(A,B)**
- **groupby(B,C)**
- **groupby(A)**
- **groupby(B)**
- **groupby(C)**
- **groupby(none)**

- **relation tuples: points in the N-d space**
- **groupby projections: also points**
- **point data is very efficient for multidimensional indexing**

Packing & Sort Order of Cubetrees

- **Cubetrees are packed and compressed from N-d to lower dimensions**
- **Sort order is used for merging during incremental bulk updates**



well clustered groupbys (on the same or consecutive pages)

these groupbys dispersed (on different pages)

For a chosen sort order : A,B,C

Cubetrees for OLAP Queries

- Very fast OLAP queries (2-3 orders of magnitude)
- Minimal space overhead (less than 15%)
- Efficient bulk updates (6 GB per hour on a single processor/single disk)
- Scalable solution (merge packing)

Need more work

- Extend standard DBMS architectures with
  - representations for spatial data types spatial operators
  - spatial indexes
  - access methods for spatial indexes
  - optimizer extensions
  - query language extensions to cover spatial queries
  - user interface extensions to handle graphical interaction, input-output of spatial data and relationships, etc.

What is coming next?

- Some of the issues we find intriguing

Benchmarking

- Define statistically well founded workloads for a variety of applications
- Provide an environment which includes an attractive user interface and tools for visualisation

What is coming next?

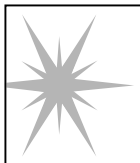
Performance Evaluation of Access Methods

- Thorough experimental examination of the approaches to test the behavior under real workloads
- Evaluation with realistic query types
- Extensibility on the range of queries
- Scalability behavior with growing volumes of data

What is coming next?

Query Optimization

- Relatively undeveloped in the area of spatial DBs
- In many cases, e.g. geographic DBs, the execution strategy chosen is not near-optimal; just barely reasonable execution order
- Systematic cost estimates of different execution strategies are needed



Summary

There is no question that trees have grown everywhere and will continue to grow in areas where high performance is needed, marking yet another strong contribution from the area of database systems.