Query Processing and Advanced Queries

Advanced Queries (2): R-Tree
Given a point set and a rectangular query, find the points enclosed in the query

We allow insertions/deletions online
Each leaf node can hold up to 2 points.
Query Answering using \textit{kd-Tree}

Each leaf node can hold up to 2 points.
Point Access Methods can index only points. What about regions?

- Use the transformation technique and a PAM
- New methods: Spatial Access Methods SAMs
  - R-tree and variations
Approximate each region with a simple shape: usually Minimum Bounding Rectangle (MBR) = \([(x_1, x_2), (y_1, y_2)]\)
Transformation Technique

- Map an d-dim MBR into a point: ex.
  \[ ((x_{\text{min}}, x_{\text{max}}), (y_{\text{min}}, y_{\text{max}})) => (x_{\text{min}}, x_{\text{max}}, y_{\text{min}}, y_{\text{max}}) \]
- Use a PAM to index the 2d points
- Given a range query, map the query into the 2d space and use the PAM to answer it
SAM: The Problem

- Given a collection of geometric objects (points, lines, polygons, ...)
- Organize them on disk, to answer spatial queries (e.g., range query, NN query, etc)
Indexing using SAMs

Two steps:

- **Filtering step:** Find all the MBRs (using the SAM) that satisfy the query
- **Refinement step:** For each qualified MBR, check the original object against the query
[Guttman 84] Main idea: allow parents to overlap!

- => guaranteed 50% utilization
- => easier insertion/split algorithms.
- (only deal with Minimum Bounding Rectangles - MBRs)
R-trees

- A multi-way external memory tree
- Index nodes and data (leaf) nodes
- All leaf nodes appear on the same level
- Every node contains between m and M entries
- The root node has at least 2 entries (children)
Example

- Fan-out = 4: group nearby rectangles to parent MBRs; each group -> disk page
Example

- F=4

Diagram with regions labeled P1, P2, P3, and P4, and elements A, B, C, D, E, F, G, H, I, J.
Example

F=4
R-Trees - Format of Nodes

- \{ (MBR; obj_ptr) \} for leaf nodes

\[
\begin{array}{c|c}
\text{x-low; x-high} & \text{obj ptr} \\
\text{y-low; y-high} & \ldots \\
\end{array}
\]

Diagram:
- Node structure:
  - MBR: Minimum Bounding Rectangle
  - Obj_ptr: Object Pointer
- Example:
  - P1, P2, P3, P4
  - A, B, C
R-Trees - Format of Nodes

- \{(MBR; node\_ptr)\} for non-leaf nodes

```
x-low; x-high
y-low; y-high

node ptr

... ...
```

```
P1  P2  P3  P4

A    B    C
```
R-Trees: Search
R-Trees: Search

P1

P2

P3

P4

A

B

C

D

E

F

G

H

I

J

P1

P2

P3

P4

A

B

C

D

E

F

G

H

I

J

CMPT 454: Database Systems II – Advanced Queries (2)
R-Trees: Search

Main points:

- Every parent node completely covers its ‘children’
- A child MBR may be covered by more than one parent - it is stored under ONLY ONE of them.
- A point query may follow multiple branches.
- Everything works for any(?) dimensionality
Spatial Queries

- Given a collection of geometric objects (points, lines, polygons, ...)
- organize them on disk, to answer efficiently
  - range queries
  - k-nn queries
Spatial Queries

- Given a collection of geometric objects (points, lines, polygons, ...)
- organize them on disk, to answer efficiently
  - range queries
  - k-nn queries
Spatial Queries

- Given a collection of geometric objects (points, lines, polygons, ...)
- Organize them on disk, to answer efficiently
  - Range queries
  - K-nn queries
R-Trees - Range Search

pseudocode:

check the root
    for each branch,
        if its MBR intersects the query rectangle
            apply range-search (or print out, if this is a leaf)
R-Trees - NN Search

- P1
- P2
- P3
- P4

q

A
B
C
D
E
F
G
H
I
J
Two Metrics to Ordering the NN Search

- \( \text{MINDIST}(P, R) \) is the minimum distance between a point \( P \) and a rectangle \( R \).
- If the point is inside the rectangle, \( \text{MINDIST} = 0 \);
- If the point is outside the rectangle, \( \text{MINDIST} \) is the minimal possible distance from the point to any object in or on the perimeter of the rectangle.

\[
\forall o \in R, \text{MINDIST}(P, R) \leq \| (P, o) \|
\]
**MINMAXDIST**

- **MINMAXDIST(P,R):** for each dimension, find the closest face, compute the distance to the furthest point on this face and take the minimum of all these (d) distances.

- **MINMAXDIST(P,R)** is the smallest possible upper bound of distances from P to R.

- **MINMAXDIST** guarantees that there is at least one object in R with a distance to P smaller or equal to it.

$$\exists o \in R, \|(P, o)\| \leq MINMAXDIST(P, R)$$
**MINDIST and MINMAXDIST**

- $MINDIST(P, R) \leq NN(P) \leq MINMAXDIST(P, R)$
R-Trees - NN Search

Q: How? (find near neighbor; refine...)

![Diagram of R-Trees and NN Search](image)
R-Trees - NN Search

Q: How? (find near neighbor; refine...)