AA Assignment

DATE:

Kreena Shah
60004210243
C'32

(1) R-trees are specialized data structures designed for spatial indexing, particularly for multidimensional data such as
(a) points
(b) rectangles
(c) polygons in space.
They are primarily used in spatial databases & geographic information system (GIS) to efficiently query & retrieve data objects

Need for R-trees :
(a) Efficient Spatial Queries
Traditional data structures like B-trees are not optimized for spatial data.
R-trees organize spatial objects in a hierarchical manner, enabling efficient spatial queries

(b) Space Partitioning
R trees partition the space into smaller regions, allowing for quicker retrieval of space objects

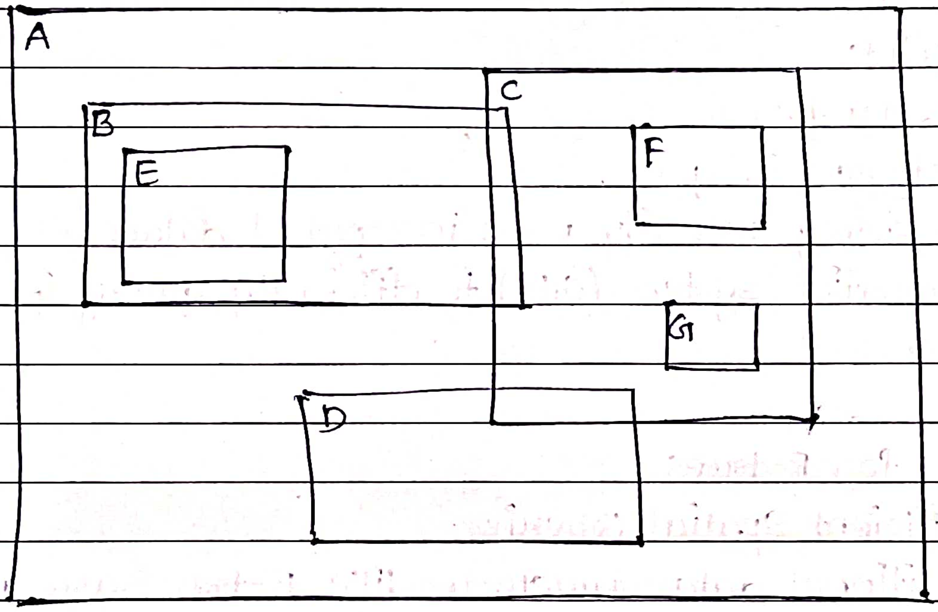
(c) Support for Dynamic Data
R-trees are dynamic data structures that can handle insertions deletions & update efficiently.

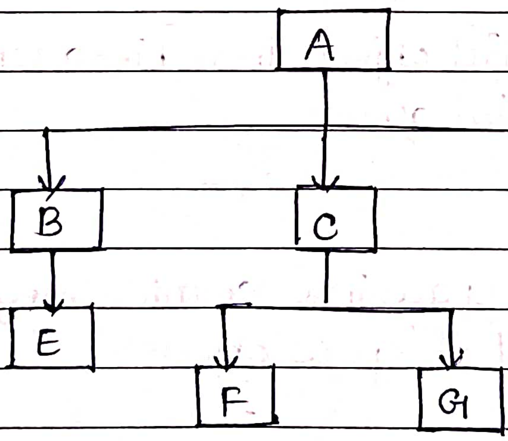
DATE:

(a) Indexing Spatial Data

By indexing spatial data using R-trees, database can significantly speed up spatial queries

Working





MBRs (Minimum Bounding Rectangles)

DATE:

It has hierarchical tree structure
It has 2 types of nodes
(1) leaf nodes
(2) actual spatial objects & their MBR

Branch node contains MBR of combined spatial area of their children.

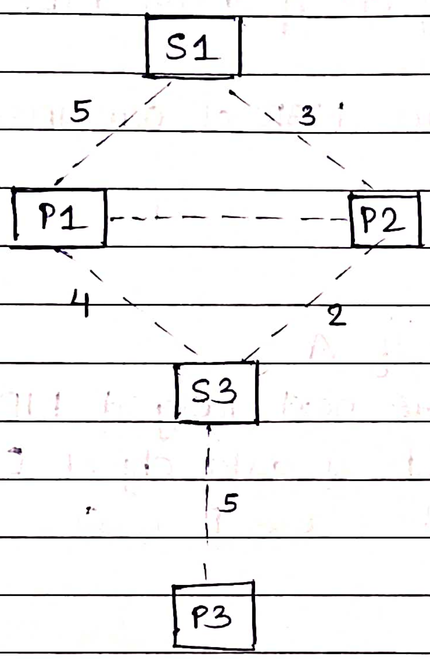
In the example,
the largest MBR is of A
Inside A, B,C,D are next largest MBRs
Inside B, there exists a data object E
Inside MBR of C, there are F & G
Next MBR is D.

(2) Weighted non-bipartite is a problem in graph theory & optimization where the goal is to find a subset of edges in a graph such that no two edges share a common vertex, & the sum of weights of the selected edges is maximized.

Example:
Consider a scenario where you have a group of students & a set of projects.
Each student has certain skills & each project requires specific skills.
The goal is to assign students to projects in a way that maximizes overall skill level of assigned students

DATE:

nodes S1, S2, S3 represent students
nodes P1, P2, P3 represent projects



One possible maximum weight matching
- S1 assigned to P1 (5)
- S2 assigned to P2 (5)
- S3 assigned to P3 (5)

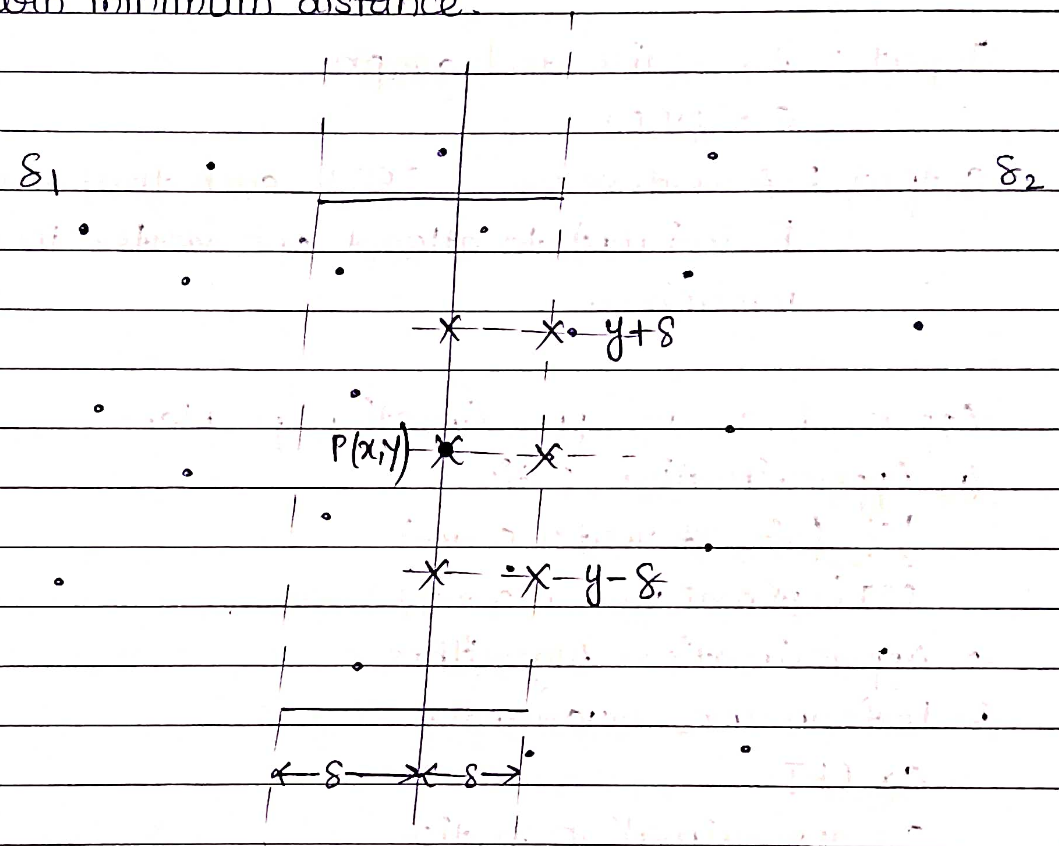
(3) Finding closest pair of points in a set of points is a classic problem in computational geometry.
One common & efficient technique is "Divide & conquer" algorithm

Time Complexity : O(nlogn)

Space Complexity : O(n)

DATE:

(1) Sort points
(2) Divide
(3) Recursively find closest pair
(4) Merge Step
(5) Return minimum distance.



δ_1 = closest min distance pair in LHS
δ_2 = closest min distance pair in RHS
δ = min (δ_1, δ_2)

Since the points in rectangle must be separated by δ_1 we have atmost 6 points to investigate.

DATE:

(4) The Vertex Problem is a classic problem in graph theory. Given an undirected graph, a vertex cover is a subset of vertices such that every edge in the graph is incident to atleast one vertex in the subset

Input : An undirected graph
$G = (V, E)$

Output : A vertex cover $S \subseteq V$ such that every edge in E is incident to atleast one vertex in S, & $|S|$ is minimized

Approach as an approximation problem

(1) Approximation Ratio

$$\frac{|S| \text{ (size of vertex cover)}}{\text{OPT (optimal vertex cover)}}$$

(2) Approximation Algorithm

(3) Performance Guarantee

$C \times \text{OPT}$

C : approximation ratio

(4) Trade off

(5) K-server problem is a classic problem in computer science & computational geometry, often used to model & analyze the performance of online algorithms in the context of server placement.

In this problem,

There are K-servers located at specific points in a metric space & request arrive at various locations in the space over time.

DATE:

The objective is to minimize the total distance traveled by the servers to serve all requests.

(6) Satisfiability (3-SAT)
3-SAT is a specific form of the Boolean satisfiability problem.
In SAT, we're given a Boolean formula in conjunctive normal form (CNF), where each clause consists of exactly 3 literals joined by OR & our task is to determine whether there exists an assignment of truth values that satisfies the formula

eg : $(x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x_3})$

Polynomial Time Reducibility is a concept to show that one problem is atleast as hard as another problem.
If problem A can be reduced to problem B in polynomial time, it means that any instance of problem A can be transformed into an instance of problem B in polynomial time, such that the solution to problem B yields solution to problem A

NP Completeness Proof
(1) Show the problem is in NP
(2) Reduce a Known NP complete problem to given problem