Dimensionality Reduction 2: Rectangle-Point Containment

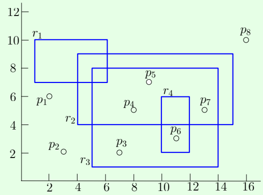
Yufei Tao

CSE Dept
Chinese University of Hong Kong

Problem

Let R be a set of axis-parallel rectangles and P be a set of points, all in $\mathbb{R}^d$, where d is a fixed constant. We want to report all pairs of $(r, p) \in R \times P$ such that r contains p.

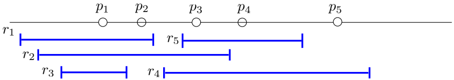
A 2D example



We will show how to solve the problem in $O(n \operatorname{polylog} n + k)$ where $n = |R| + |P|$ and k is the number of pairs reported.

1D

When $d = 1$, R is a set of intervals and P a set of points, both in $\mathbb{R}$.



It is easy to settle the problem in $O(n \log n + k)$ time.

2D

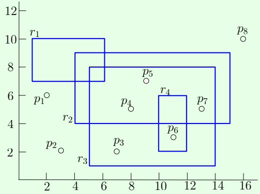
Assumption: R does not contain any rectangle of the form $(-\infty, \infty) \times [y_1, y_2]$ (i.e., a horizontal stripe).

Removing the assumption will be left to you (it is easy).

Every rectangle in R defines at most two **finite** x-coordinates, and each point in P defines one x-coordinate. Call those coordinates the **input x-coordinates**.

A left-open or right-open rectangle defines only one input x-coordinate.

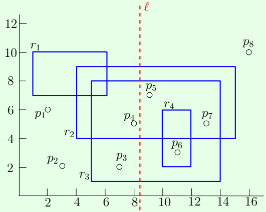
2D



Input x-coordinates: $1, 2, ..., 16$.

2D

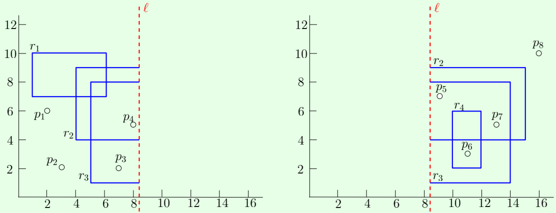
Divide the input x-coordinates in half with a vertical line ℓ.



We will assume that such a line ℓ exists. Handling the opposite scenario is left to you.

2D

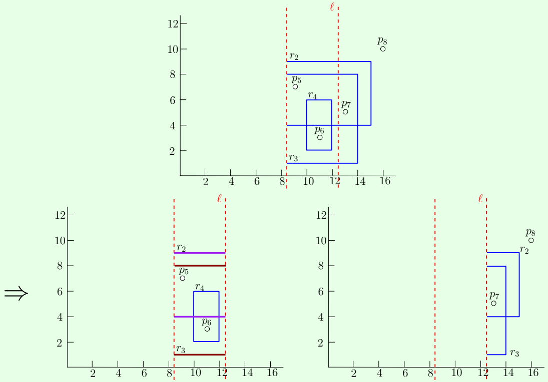
The line ℓ creates two sub-problems.



Note that each sub-problem can contain left-open or right-open rectangles. No new input x-coordinates are created.

2D

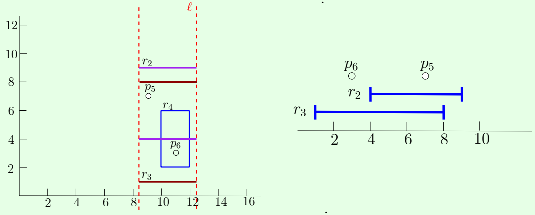
Divide the right sub-problem into two "sub-sub-problems":



Issue: In the first sub-sub-problem, r_2 and r_3 define no input x-coordinates. Thus, we **cannot** solve the sub-sub-problem recursively (think: why).

2D

Dealing with the issue: solve a 1D instance of the problem on the y-dimension and get rid of such rectangles.



The 2D Algorithm

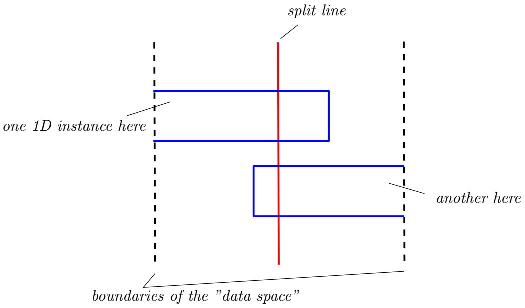
1. Let R_{span} be the set of rectangles that do not define input x-coordinates (they span the current data space in x-dimension).
2. Solve a 1D instance on R' and P' where R' and P' are obtained by projecting R_{span} and P onto the y-axis, respectively.
3. Remove R_{span} from R.
4. Divide the input x-coordinates equally with a vertical line ℓ.
5. Let R_1 (or R_2) be the set of rectangles in R that intersect with the left (or right, resp.) side of ℓ. Let P_1 (or P_2) be the set of points in P that fall on the left (or right, resp.) side of ℓ.
6. Solve the left sub-problem with inputs R_1, P_1 and the right sub-problem with inputs R_2, P_2.

2D Analysis

Let $f(m)$ be the running time of our algorithm when there are m input x-coordinates.

$$f(m) \quad \leq \quad 2 \cdot f(m/2) + 2 \cdot g(m)$$

where $g(m)$ is the cost of solving a 1D instance of size m.



2D Analysis

$$f(m) \leq 2 \cdot f(m/2) + 2 \cdot g(m)$$

We know that $g(m) = O(m \log m + k')$ (where k' is the number of pairs reported by the 1D instance). Solving the recurrence gives $f(m) = O(m \log^2 m + k)$.

As $m \leq 2n$, we now have an algorithm of $O(n \log^2 n + k)$ time.

Remark: In this week's exercises, you will be guided to improve the running time to $O(n \log n + k)$.

d-Dimensional

In general, we can use a $(d-1)$-dimensional algorithm to solve the d-dimensional problem. It will be left as an exercise to design a d-dimensional algorithm in $O(n \operatorname{polylog} n + k)$ time.