CSL 630 Lecture 24, Nov 3

Range Trees for orthogonal range searching

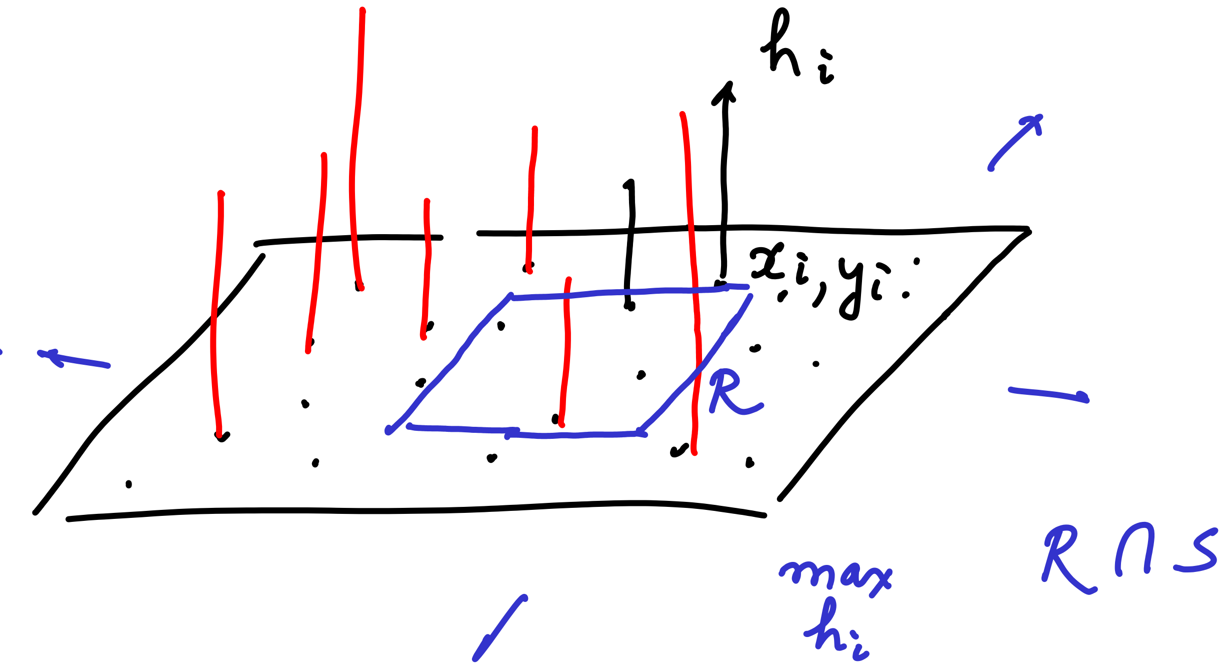
Given a set S of n points in R^d, design a data structure that support queries of the kind

For any rectangle in d dimension

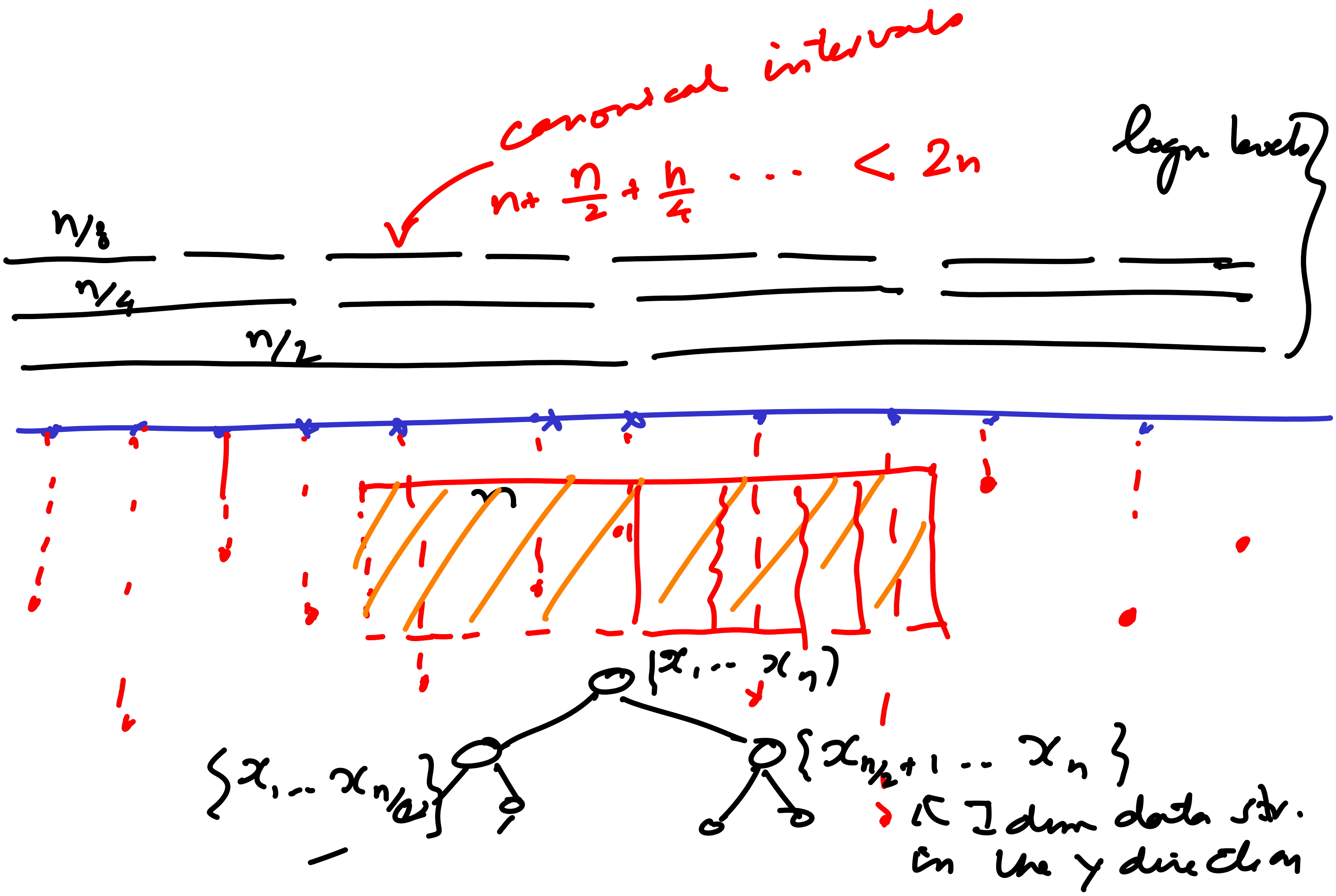
$$R = [x_l^1, x_u^2] \times [x_l^2, x_u^2] \cdots \times [x_l^d, x_u^d]$$

- Report points in $R \cap S$
- Count $|R \cap S|$

In general compute some function f on $R \cap S$



h_i
x_i, y_i
R
max h_i
$R \cap S$



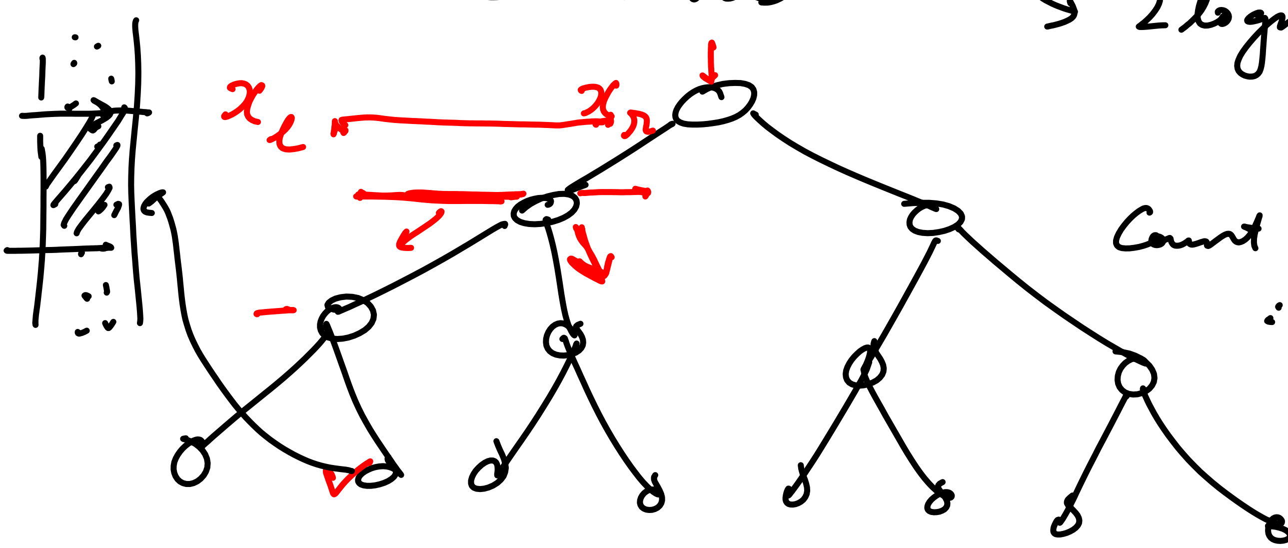
canonical intervals
$n + \frac{n}{2} + \frac{n}{4} \dots < 2n$
log n levels
$n/8$
$n/4$
$n/2$
n
$\{x_1 \dots x_n\}$
$\{x_1 \dots x_{n/2}\}$
$\{x_{n/2+1} \dots x_n\}$
[ ] 1 dim data str. in the y direction

To answer a 2dim rectangular query, we first find out the (atmost $2\log n$) subintervals in the x-direction.

For each subinterval, we have built a 1-dimensional range query data structure in the y-direction.

Ans = $\bigcup_{\text{subinterval}}$ queries in disjoint subintervals

$\to 2\log n \times$ Query of each interval



Count query : $O(\log^2 n)$

Storage = Sum of storage in each level

$= \log n \times n = O(n\log n)$

Preprocessing :

Initially sort along x: $O(n \log n)$

We allocate the points to the appropriate canonical subintervals (x-direction) and then build data structure for each cannonical subinterval.

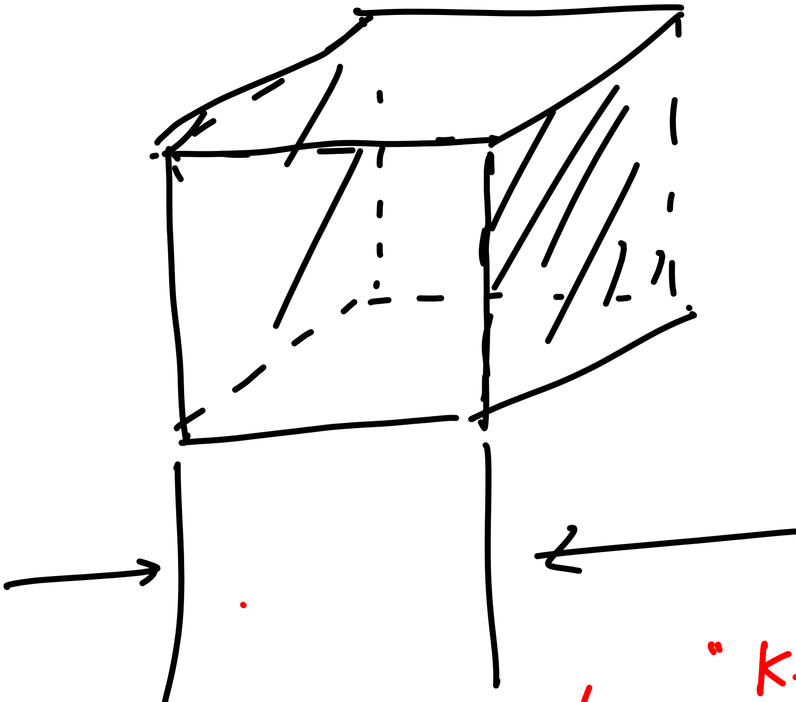
Preprocessing time: $\underbrace{\sum}_{\text{levels}} \underbrace{\sum}_{\text{nodes within a level}}$ preprocessing-time within a node

$$\sum_{l=1}^{\log n} \sum_{i=1}^{2^l} T\left(\frac{n}{2^{l-1}}\right)$$

$\frac{n}{2^{l-1}}$ ← # points in a node in level l

$$T\left(\frac{n}{2^{l-1}}\right) \rightarrow \underbrace{c \cdot \frac{n}{2^{l-1}} \log\left(\frac{n}{2^{l-1}}\right)}_{\text{Sorting in y-direction}}$$

$$= \sum_{l=1}^{\log n} \cancel{2^l} \cdot c \cdot \frac{n}{\cancel{2^{l-1}}} \cdot \log n \leq c\, n \log^2 n$$



reducing $\log^d n$

for d dimensions

"K-d tree"