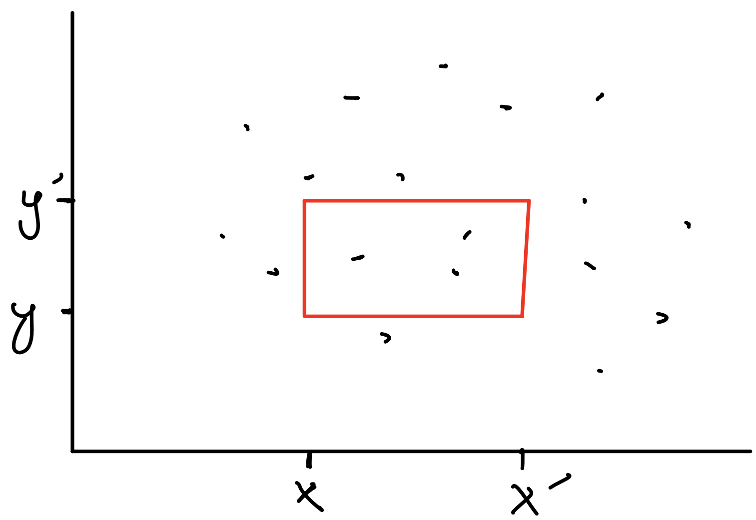
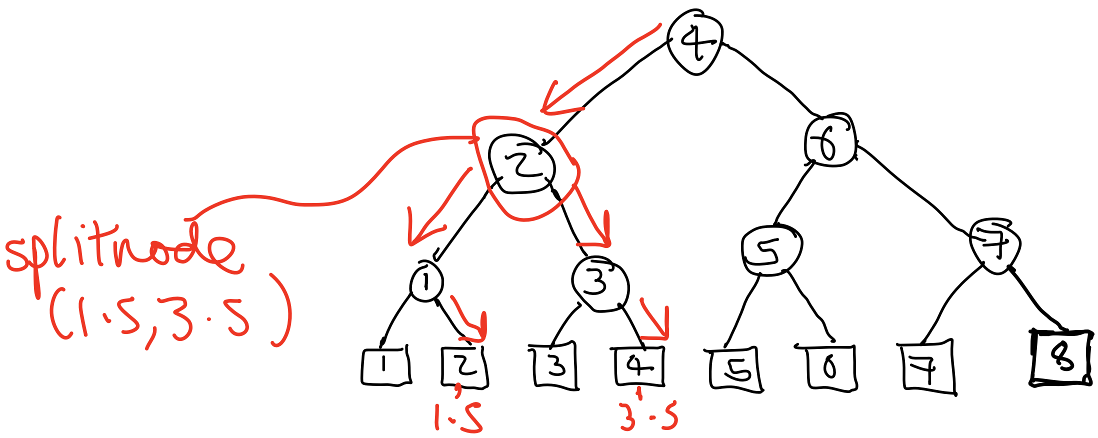
Lecture 8 - Orthogonal Range Searching

- Consider set $P \subseteq \mathbb{R}^d$ and a range $[x_1, x_1'] \times \dots \times [x_d, x_d'] \subseteq \mathbb{R}^d$.
- Find points of P belonging to the range.
- Relevant to querying databases.



1-d range searching

- $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}$ & $x \leq x'$
- Points of P stored as leaves in <u>binary balanced tree</u>.



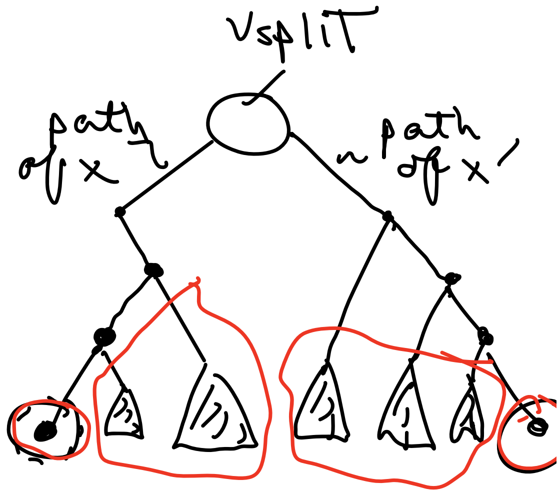
- Node v stores max. value in left subtree.
- Left subtree of node v contains elements $\leq x_v$, right subtree contains elts. $> x_v$.
- Point $x \in \mathbb{R}$ determines path from root to leaf: at node v, go left if $x \leq x_v$ & right if $x > x_v$.

Eg: $x = 1.5$, $x' = 3.5$,

- At $x \leq x'$, <u>splitnode (x, x')</u> is last node at which paths for x, x' agree.

Algorithm

- Input : $P = \{x_1, \dots, x_n\} \subseteq \mathbb{R}$ stored in a bal. bin. tree & $x \le x'$
- Find points of P in range $x \le x'$.

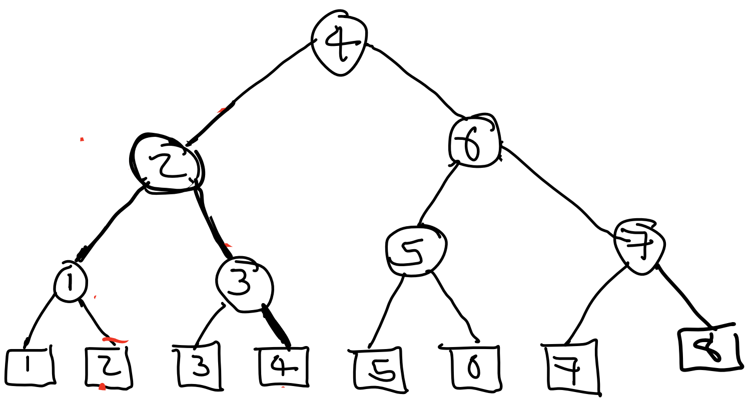


- Find splitnode v_{split}
- If v_{split} is a leaf, check if in $[x\ x']$ & report it, if so.
- Follow path of x from v_{split} to a leaf.
- If at node v, x moves left, we report all points in right subtree of v as solutions.
- At leaf, check if it belongs to $[x, x']$
- Similarly, follow path of x' to leaf, and when x' moves right, report left subtree. At leaf, check.

Why does this find all solutions?

- Suppose $x \le p \le v_{spl}$.
- Then p is reported when paths for x and p diverge, or at leaf p itself.
- Similarly if $v_{spl} \le p \le x'$.

[5.5, 9]



Do in class.

- See E-Learning for pseudocode.

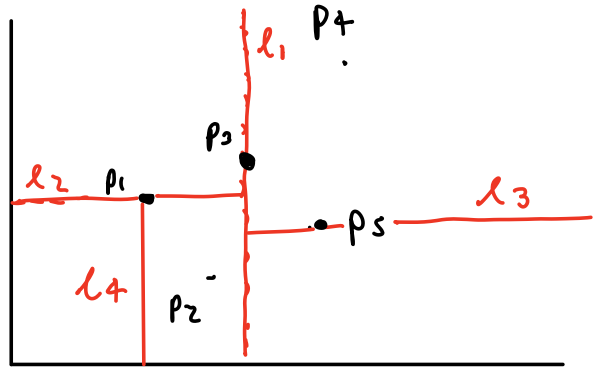
Complexity

- Time $O(\log n)$ to follow path of x.
- Likewise for x'.
- Time to report k solutions is $O(k)$.

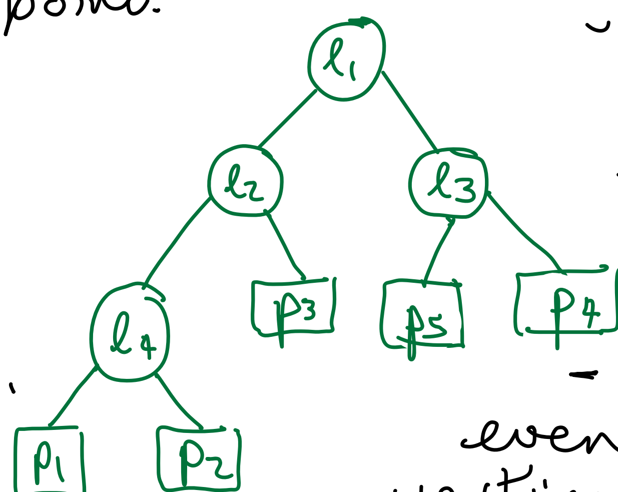
Total complexity $O(\underbrace{\log n}_{\text{no of leaves}} + \underbrace{k}_{\text{no of solutions}})$.

2-d range searching

- Set $P \subseteq \mathbb{R}^2$ (assume no 2 pts have same x or y coordinate)
- Using vertical line, split through median point ordered by x-coordinate. Count point on the line in left region (should be same number of points in either region or one more in the left)



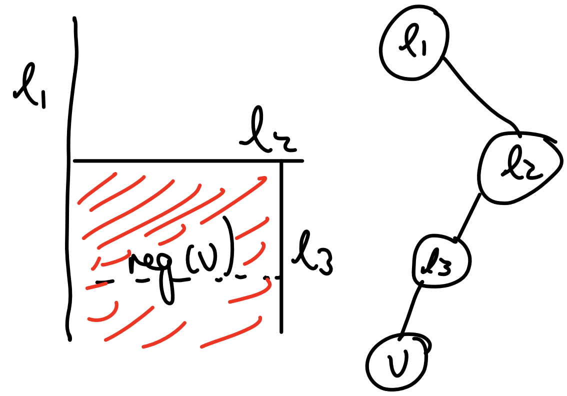
- Now split left & right regions using horizontal lines, through point with median y-coord, so lower (left) regions contains line & has same no. or 1 more point as upper (right) region.
- Repeat until each region contains 1 point.



- This data structure is called a KD-Tree.
- Takes O(n) storage, where n is no. of leaves.
- O(n log n) to const. KD-Tree on n points.

- Binary tree
- Leaves are points of P
- Nodes of even depth store vertical lines by x-coord.
- Nodes of odd depth store horizontal lines, by y-coord.

- Region of node v:
- rectangular region bordered by ancestors of v.
  (i.e. if v represents a line, region(v) is area which this line split into two.)



- Region (root) $= \mathbb{R}^2$
- Region $(lc(v)) =$ Region$(v) \cap$ left(v)

  left child

- Region $(rc(v)) =$ Region$(v) \cap$ right(v)

  right child

- A point p of P belongs to region(v) $\iff$ p belongs to subtree under v.
- Idea: search through subtree under node $v \iff$ region(v) intersects search rectangle (range).

Algorithm

- Given range R & KD-Tree of points P.
- Find points of P in R

- Move downwards through tree.
- At node v:
  - if v is a leaf, check if it belongs to P.
- otherwise, we look at $lc(v), rc(v)$.
  - If reg$(lc(v)) \subseteq R$, report subtree of $lc(v)$.
  - Else, if reg$(lc(v))$ intersects R, continue search of subtree of $lc(v)$.

  Sim, if reg$(rc(v)) \subseteq R$, ...

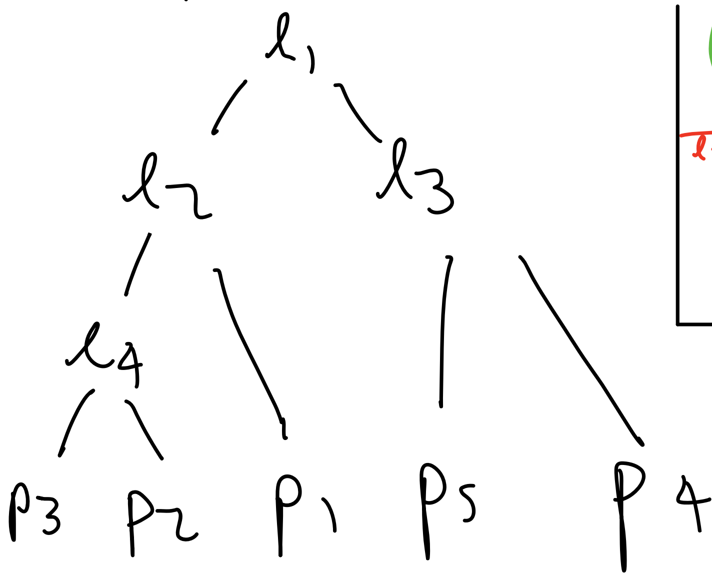
- See E-Learning for pseudocode.

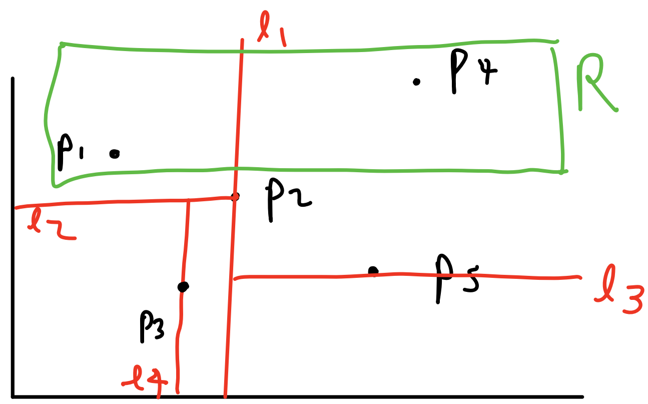
Complexity $O(\sqrt{n} + k)$

n no. of points in P

k no. of solutions

Example





- At l_1, look at l_2, l_3
- Both $reg(l_2), reg(l_3) \cap R$, but not cont. in

At l_2, look at l_4, p_1.

- $reg(l_4) \cap R = \phi$.
- $p_1 \in R \Rightarrow$ report p_1.

At l_3, look at p_5, p_4.

$p_4 \in R$.

- Removing assumption that no points in P have same x or y-coordinate

Observation: did not need points to be real numbers - only needed them to be elements of a totally ordered set: so we can compare elements & find medians.

- Pass from $\mathbb{R}$ to $\mathcal{C} = (\mathbb{R} \cup \{-\infty, \infty\})^2$
  elements of form $(a|b)$
- $\mathcal{C}$ has lexicographic order:
  $(a|b) < (c|d) \iff a < c$ or $(a = c$ & $b < d)$

$$\begin{array}{ccc} \mathbb{R}^2 & \longrightarrow & \mathcal{C}^2 \\ (p_x, p_y) = p & \longmapsto & \hat{p} = ((p_x|p_y), (p_y|p_x)) \end{array}$$

- Set $\hat{P} = \{\hat{p} : p \in P\}$.
- No two points in $\hat{P}$ have same first or second coord.

- Let $R = [x, x'] \times [y, y']$,
  $\hat{R} = [(x|-\infty), (x'|\infty)] \times [(y|-\infty), (y'|\infty)]$
- Then $p \in R \iff \hat{p} \in \hat{R}$ so only need to run our original alg (gen to a totally ord. set) on $(\hat{P}, \hat{R})$ instead

i.e. $(p_x|p_y) \in [(x|-\infty), (x'|\infty)]$
$\iff (x|-\infty) < (p_x|p_y) < (x'|\infty)$.
First ineq. $x < p_x$ or $x = p_x$ ~ $x \leq p_x$
Second ineq. $p_x \leq x'$.

Second approach - range trees

Idea: Given $P \subseteq \mathbb{R}^2$ & $R = [x,x'] \times [y,y']$

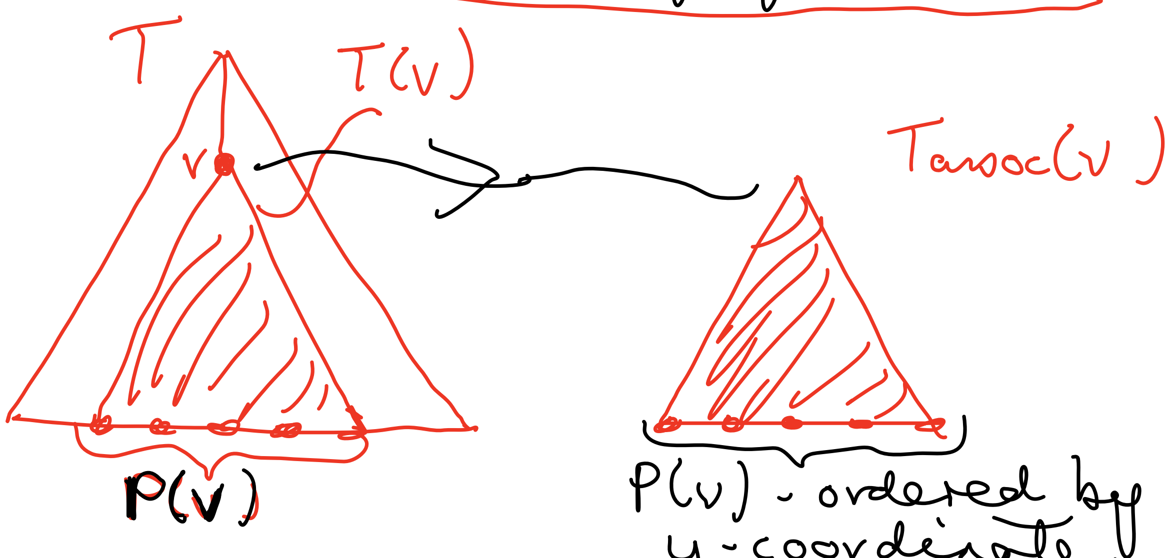
① Use a 1-d search to find points of P whose x-coord belongs to $[x,x']$.

② Search amongst those points to find those whose y-coord belongs to $[y,y']$.

Data structure : range tree.

- A binary tree whose leaves are elements of P, ordered by x-coord (assume no 2 pts have same x or y co-ord.)

- Each node v determines subtree $T(v)$ with set of leaves $P(v)$; For each such node we have another bin. tree $T_{assoc}(v)$ with leaves $P(v)$ ordered by y-coordinate.



• Storage $O(n \log n)$ - see E-learning

Searching a range tree $\breve{T}$

$R = [x, x'] \times [y, y']$

- Look at tree ordered by x-coord, Find split node of x & x'.

- If path for x moves left at v, each leaf in right subtree belongs to $[x, x']$
- Then we use a 1-d range search on $T_{assoc}(rc(v))$ to find those whose y-coord belongs to $[y, y']$.
- If v is a leaf, test whether it belongs to R
- Similarly search path of x' below split node.
- Furthermore, complexity $O(\log \underbrace{n}_{\text{no. of points}}{}^2 + \underbrace{k}_{\text{k no. of solutions}})$
- Finally, both Kd-trees & range trees can be gen. to higher dimensions. See E-Learning for this and comparison of

two approaches