Presentation on

Parallel Algorithms for Constructing Range and Nearest-Neighbor Searching Data Structures

By Sanket Neema

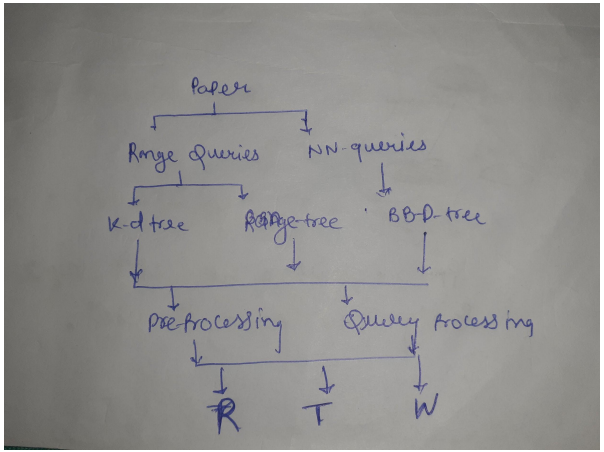
Authors:
Pankaj K. Agarwal
Kyle Fox
Kamesh Munagala
Abhinandan Nath
Duke University: {pankaj,kylefox,kamesh,abhinath}@cs.duke.edu

Flow of Presentation:

1. Introduction:
   A. MPC
   B. Parameters to check efficiency of algorithm
      a. No. of round
      b. Running Time
      c. Total Work
2. Some definitions:
   a. Range Spaces and VC-Dimension
   b. **ε-**approximation
3. Primitive Operations To Build Our Data structures
   a. PrefixSum(P, I, rank, value):
   b. Broadcast(S, I, β):
   c. Partition(P, I, Π, β):
   d. Sample(P, I, r, β):
4. KD-tree and Its MPC model.
5. Approx. Nearest Neighbour (ε-NN) using BBD trees and MPC Model
6. Range Tree and Its MPC Model.

Green: Popular Data Structure in Computational-Geometry

About paper 3 x 2 x 3 : Data structure Operation Parameters



- Type of Queries and corresponding Data Structure Used
  - Range Queries :
    - E.g: No. of emp. Whoes:
      Age b/w 30-35
      Salary 54k-70K
      .
      .
      .
      n. Constrains.
    - KD-trees
    - Range Trees
  - Approximate Nearest Neighbour (ε-NN) Queries :
    - E.g:
      1. Find 3 nearest restaurant from different movie theater.
      2. Astrophysics nearest star.
    - BBD trees
- Operations:
  1. Preprocessing.
     a. Why required
  2. Query Processing

MPC Model setup And Notation:

n input size
P Points in R^d arbitrarily distributed.
I be a set of m machines. {0, 1, 2. . . , m −1}
m, total no. of machine
s, Each machine has O(s) memory (or space). $s \geq n^{\alpha}$ $\alpha <= 1$
Input and output to/for machine is bounded by O(s).

- There are many other but I will explain then during context..

❖ Parameters to check efficiency of algorithm

- ➢ No. of round **R** :
- ➢ Running Time **T**:
  - ■ let $\beta \in I$,
  - ■ $t_{\beta r}$ denote the computation time spent by this machine in round r.
  - ■ running time ,

$$T = \sum_{r=1}^{R} \max_{\beta \in I} t_{\beta r}.$$

❖ These Does not count time it takes to Communicate b/w machine.

- ➢ Total Work **W :**

$$W = \sum_{r=1}^{R} \sum_{\beta \in I} t_{\beta r}.$$

★ Machine Activate Only when It recives input

Performance metric		*kd*-tree	BBD-tree	Range tree
Pre-processing	Rounds	$O(1)$	$O(1)$	$O(1)$
	Time	$O(s \log s)$	$O(s \log s)$	$O(s \log^d s)$
	Work	$O(n \log n)$	$O(n \log n)$	$O(n \log^d n)$
Query	Rounds	$O(1)$	$O(1)$	$O(1)$
	Time	$O(s^{1-1/d} + k')$	$O(\log n)$	$O(\log n + k)$
	Work	$O(n^{1-1/d} + k)$	$O(\log n)$	$O(\log^d n + k)$

Table 1. Our results.

General points:

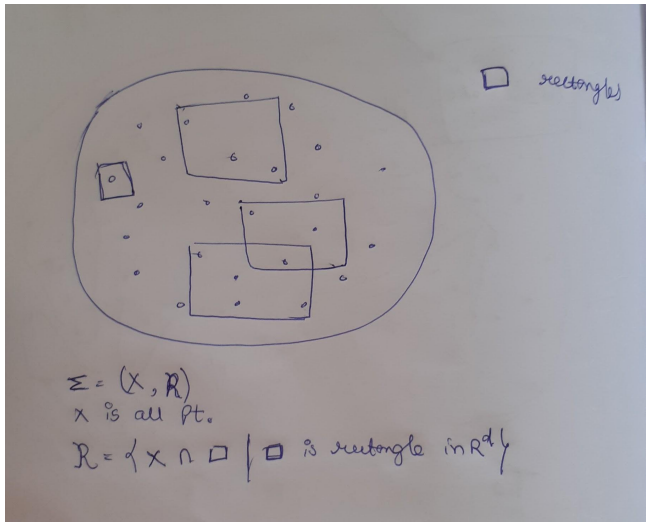
- The algorithms we discuss are randomization,
- We use ε-approximation small subset of points S, S ⊂ P.
- If we say query rectangle ☐ in d- dimension, we means we have given interval in each dimension. E.g- { (a1,a2),(b1,b2)..... (d1,d2) }

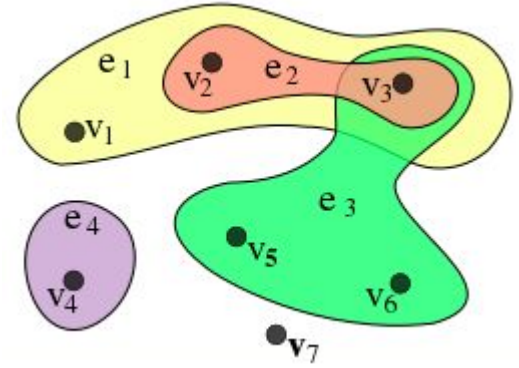
Range Spaces:

A range space Σ is a *pair* (X, R') where X is a ground set , and R' is family of subset of X.

Here: In these example set X={v1,v2,v3,v4,v5,v6,v7}
and set R' is {e1,e2,e3,e4}={{v1,v2,v3},{v2,v3},{v3,v5,v6},{v4}}.

E.g-2 :

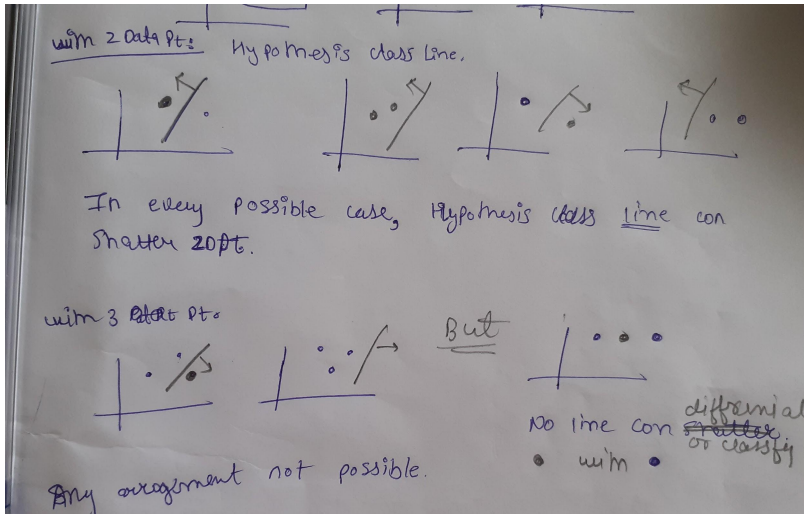


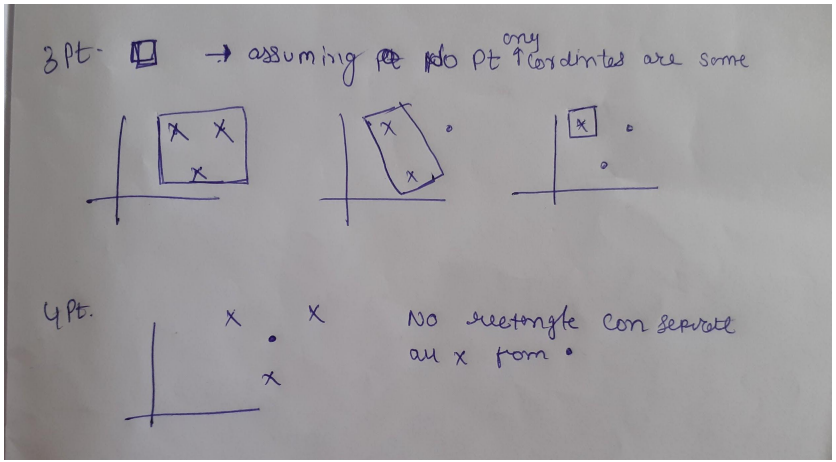


Wiki

Shattered by, and VC-Dimension :

A subset $X' \subseteq X$ is shattered by Σ if $\{X' \cap R \mid R \in R'\} = 2^{X'}$ i.e power set of X' (excluding null set).
Now, VC-Dimension of Σ is the size of the largest subset of X shattered by Σ.



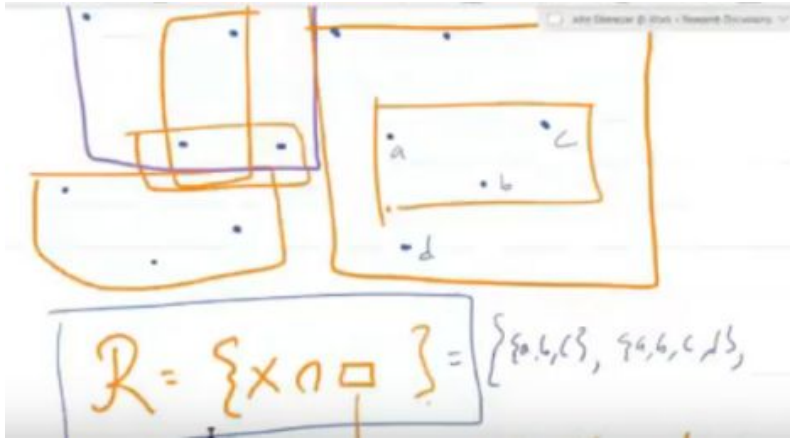


Here in these paper we assume, that give range spaces have **finite** VC-Dimension.

Epsilon-approximations:

As we have X, set with very high cardinality ,
we want X' that represent X.such that
X'⊆X , and cardinality of X' is small.

These R is one of rectangle in set of all rectangle



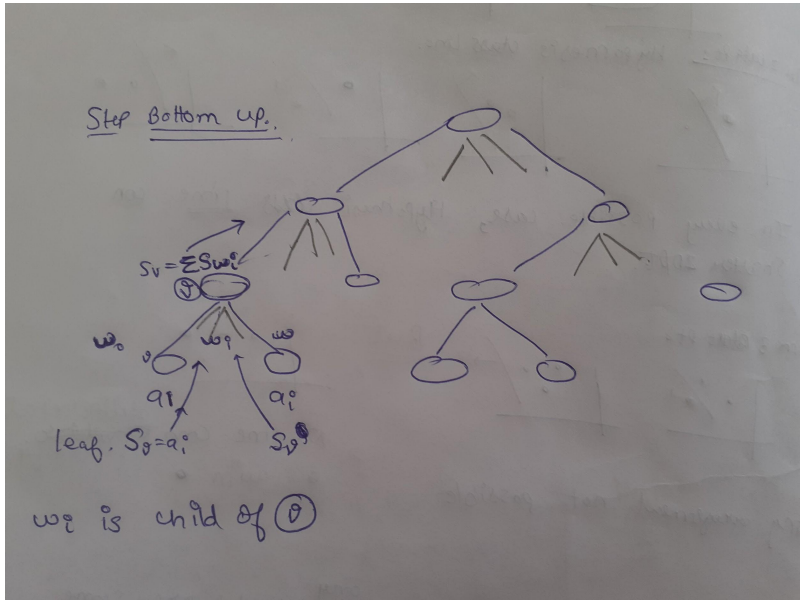
$$\left| \frac{|X' \cap R|}{|X'|} - \frac{|R|}{|X|} \right| \leq \varepsilon.$$

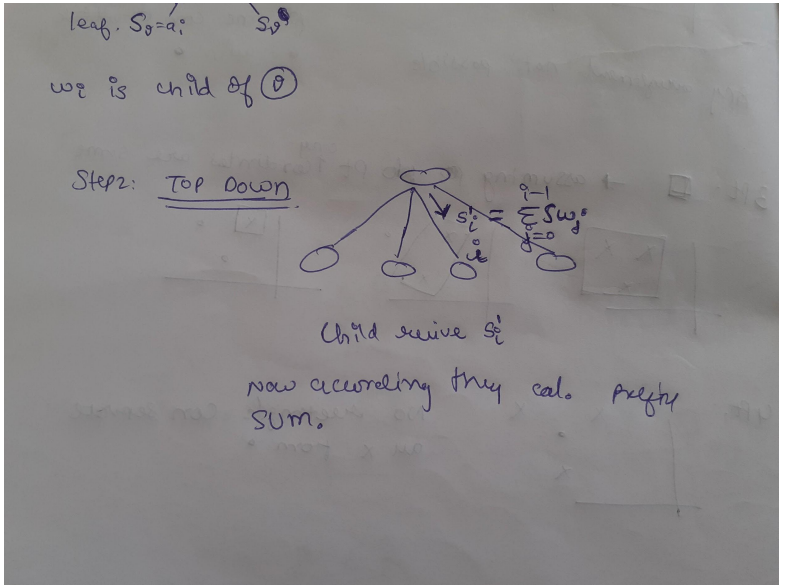
THEOREM 2.1 (35). *There is a positive constant c such that if $\Sigma = (X, \mathcal{R})$ is a range space with VC dimension δ, then a random subset of X of size $\frac{c}{\varepsilon^2}\left(\delta + \log \frac{1}{\psi}\right)$ is an ε-approximation of X with probability at least $1 - \psi$.*

Primitive Operations To Build Our Data structures:

1. **PrefixSum(P, I, rank, value):**
   a. Rank function : $P \rightarrow Z^+$
   b. Value function : $P \rightarrow R$

- ➢ No of rounds O(1)
- ➢ Time O(s)
- ➢ Work (n)



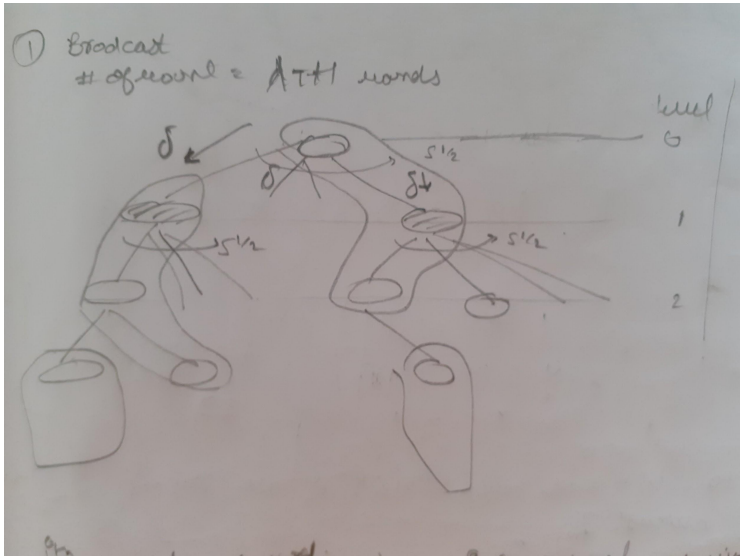


1.

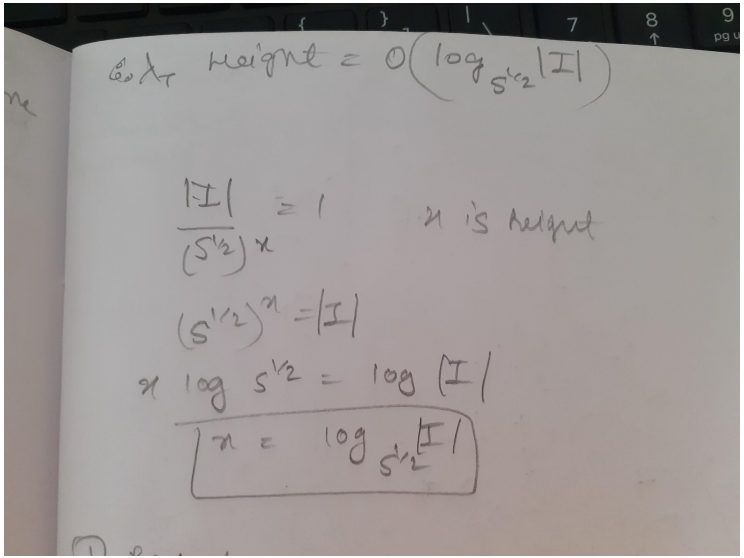
2. Broadcast(S, I, β):

- ➢ No of rounds O(1)
- ➢ Time O(s)
- ➢ Work (n)

Flow of Broadcast from machine Beta belongs to set I.

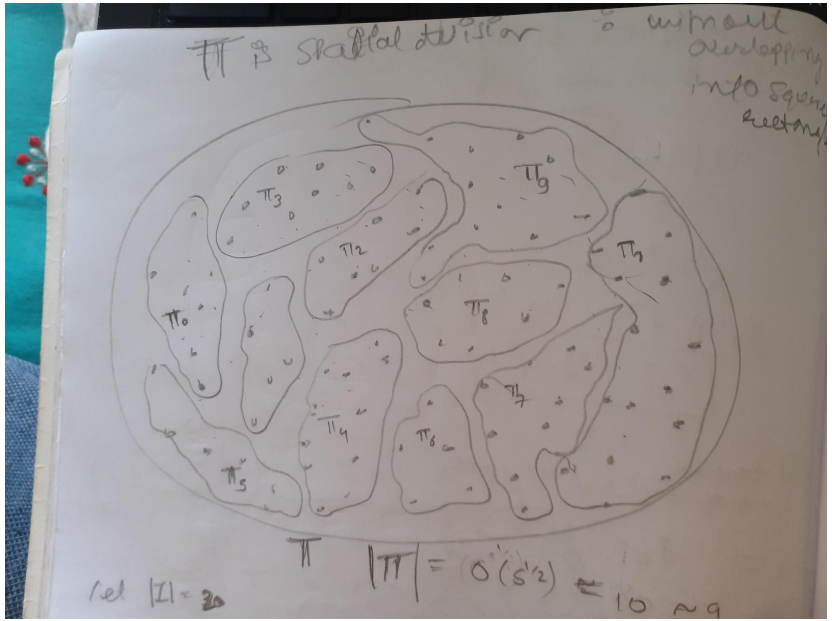


Height Calculation :



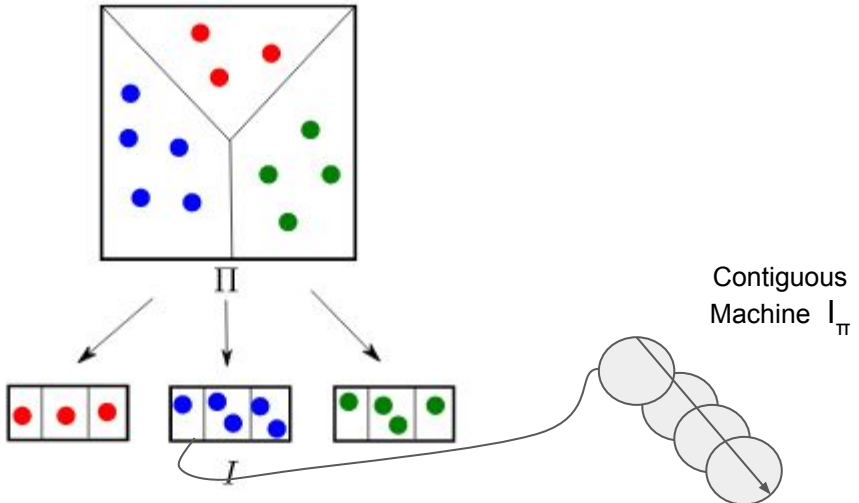
3. Partition(P, I, Π, β): Role is to take points in π of Π and store them in some contiguous machine.

- ➢ No of rounds O(1)
- ➢ Time O(slogs)
- ➢ Work (nlogn)

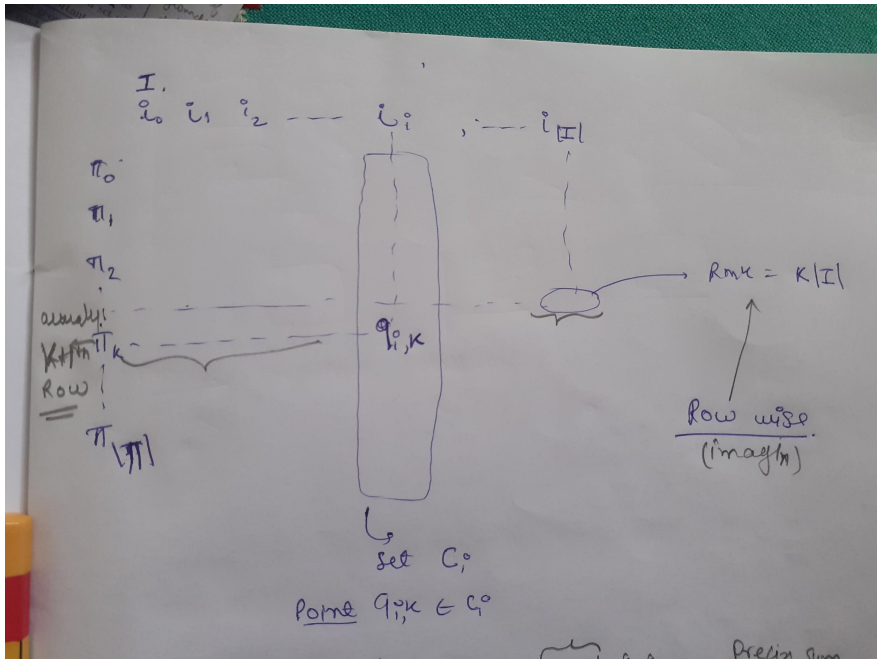


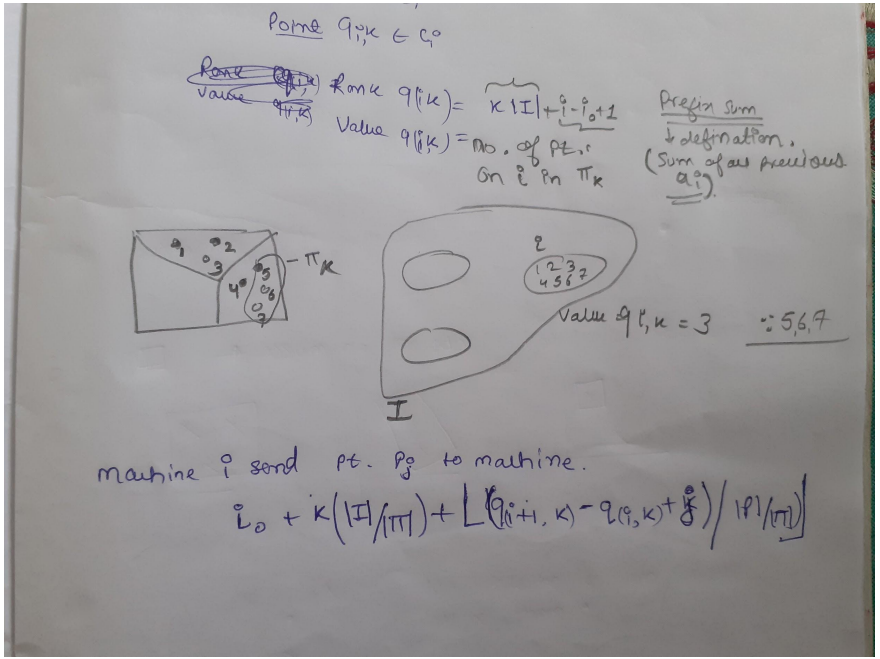
Let Initial Arrangement, shape are rectangle

- The points of P, lying in a cell π of Π lie on a distinct contiguous subset of machines. $I_\pi \subseteq I$.
- We required:
  - $|\Pi| = O(s^{1/2})$
  - $|\Pi| \leq |I|$
  - Points are evenly distributed lying in a cell π of Π. $O(|P|/|\Pi|)$



Brief View on Working of Partition operation:

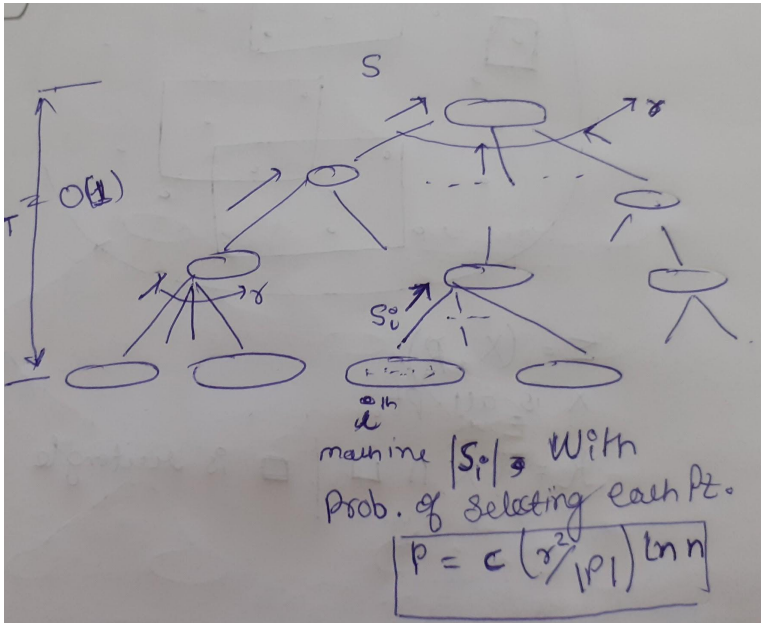


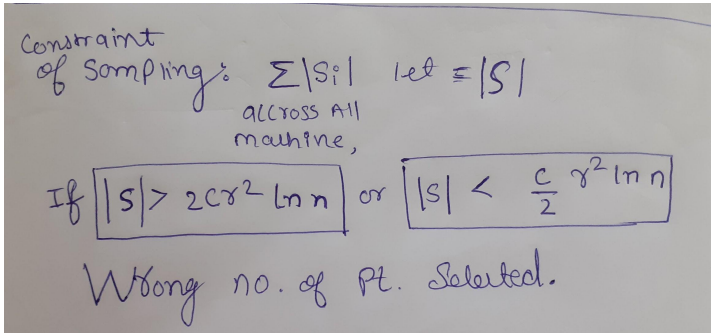


4. Sample(P, l, r, β):

- ➢ No of rounds O(1)
- ➢ Time O(s)
- ➢ Work O(|P|) or O(n)

- ➔ AIM: Las Vegas algorithm for computing the sample S ⊆ P .
- ➔ r is parameter for epsilon approximation that is r is nothing but **r=1/ε** .
- ➔ Let **β** be root machine.
- ➔ Const. c depends on the VC dimension of the underlying range space.





- ➔ The probability of restarting even once is at most $\exp(-(c/8)r^2 \ln n) \leq 1/n^2$ for large enough c.
- ➔ Each sample of size |S| is chosen with equal probability, so S is an (1/r)-approximation with probability at least $1 - 1/n^2$ as well by Theorem 2.1

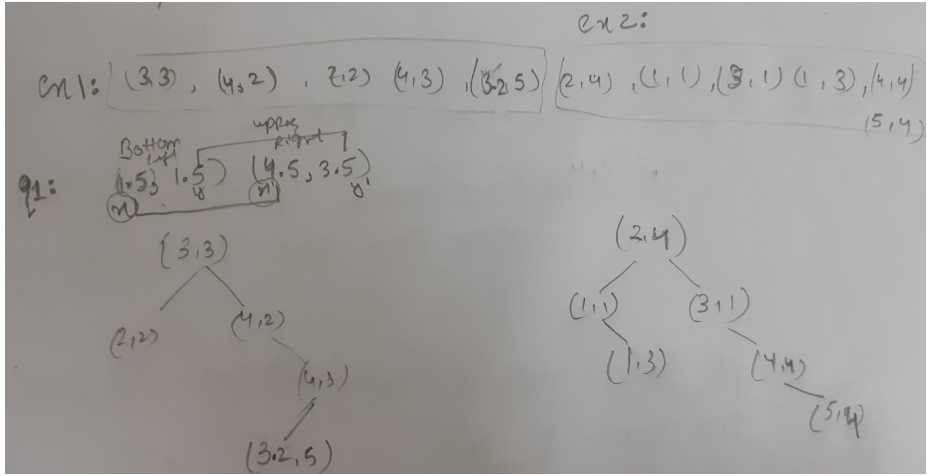
Result: The above procedure computes (1/r)-approximation S of P.

Data structure And MPC Model

1. KD- Tree:

Simple Understanding of kd-tree:

- The k-d tree is a binary tree where the underlying space is partitioned on the basis of the value of just one attribute at each level of the tree.
- E.g: In 2-D we first split on x-coordinate, next on y-coordinate, then again on x-coordinate, and so on.

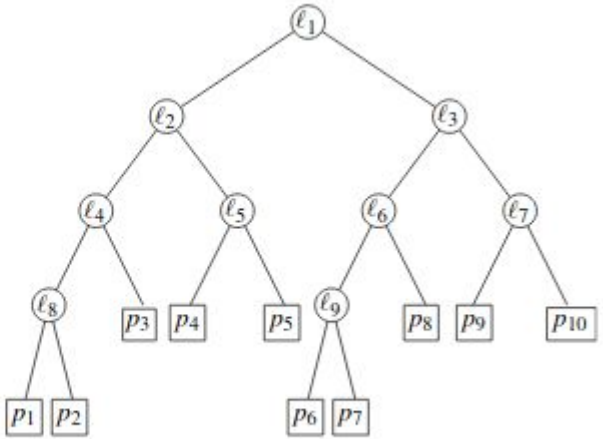


For d-dimension:

- Let Each node v of the tree is associated with a d-dimensional rectangle $\square_{V.}$,contains | P_v | points.
- v is at at i^{th} depth (root at depth 0) split into two $\square_{w.}$ and $\square_{z.}$ by an axis-parallel hyperplane h_v parallel to the **((i mod d) + 1)th axis (attribute).**

To make balance Partition:

- First sorting P along each axis
- Then follow Following algorithm:



Source: Computational Geometry - Algorithms and Applications, 3rd Ed Book

Build Kd tree (P, depth)

{

if (|Pv| == 1)

return leaf.

else,

we split Pv into Pw & Pz

|→ find median. of points in Pv acc. to ((depth % d) + 1)th axis.

Find split.

left ← Built Kd tree (Pw, depth+1)

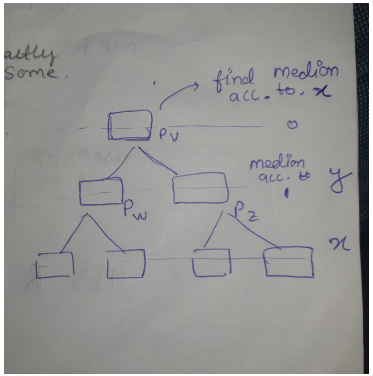
Right ← Built Kd tree (Pz, depth+1)

node v left child = left

——— Right ——— = Right

Return v.

}



Instead of passing P pass two sored, one on x-coordinate and one on y-coordinate.

$$T(n) = \begin{cases} O(1), & \text{if } n = 1, \\ O(n) + 2T(\lceil n/2 \rceil), & \text{if } n > 1, \end{cases}$$

Construction: O(n log n)
Storage: O(n)

Query On kd tree:

E.g:

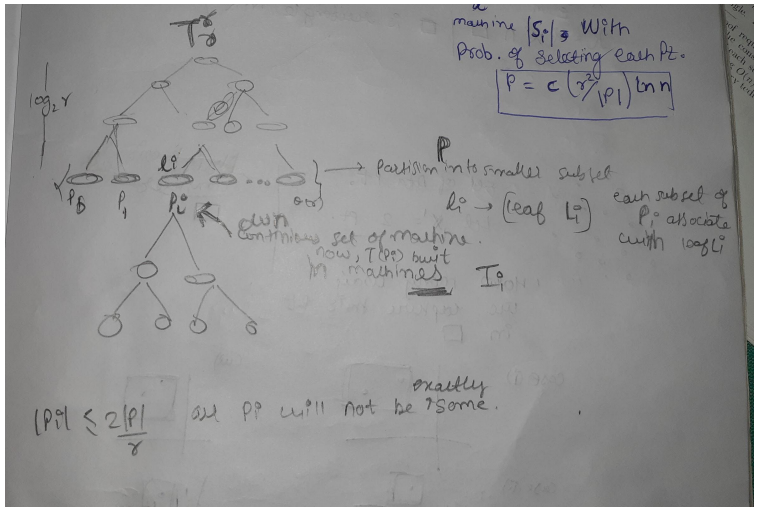
- Queries are performed recursively in a top-down manner starting with the root node.
- **Given a query rectangle ρ** and a node v of the tree, there are three cases :
  - $P_v \subseteq \rho$,then all points of P_v are reported
  - $P_v \cap \rho = \varnothing$ nothing to be done,
  - $P_v \cap \partial \rho \neq \varnothing$, we recurse on the children of node V .

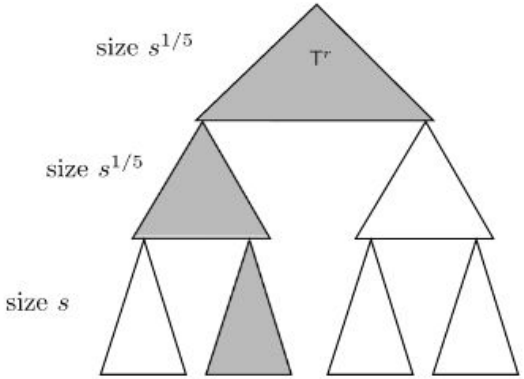
In d-dimension orthogonal range-reporting query:

- $O(n^{1-1/d} + k)$ time
  - where k is the number of points lying inside the query rectangle.
- O(n) space,

MPC Model of KD-Tree:

- Case 1: |P | = O(s), can be solve sequentially as T is stored on a single machine.
- Else:
  - we choose a parameter r
  - The top subtree T^r of T containing $\log_2 r$ levels and Θ(r) leaves is built and stored on one machine, say β.





- Shaded may be stored in single machine.
- Keeping O(s) memory constraint.
-

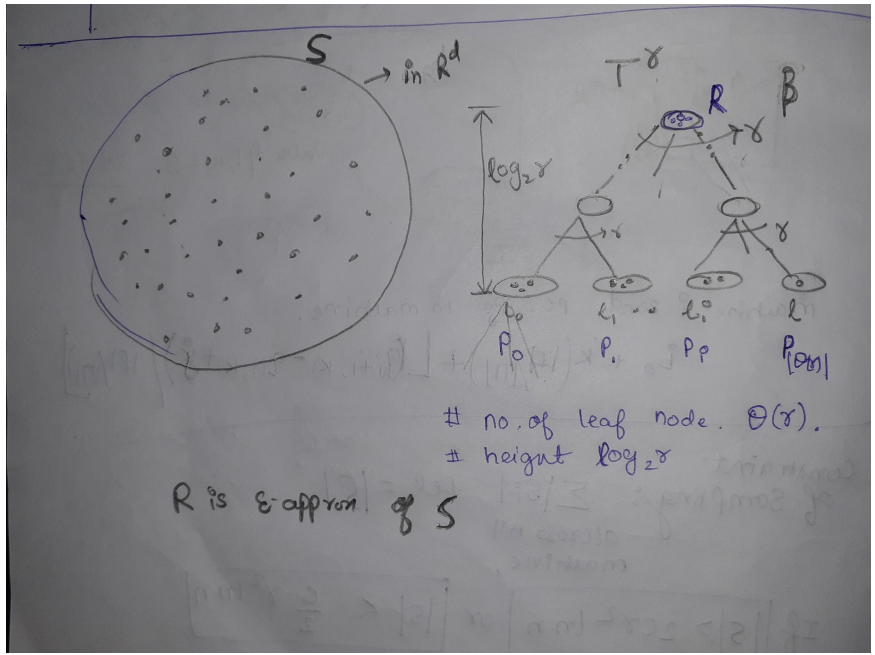
Algorithm Kd-tree MPC Model :

Algorithm 1 `Build-kd-tree`$(S, \square, I)$

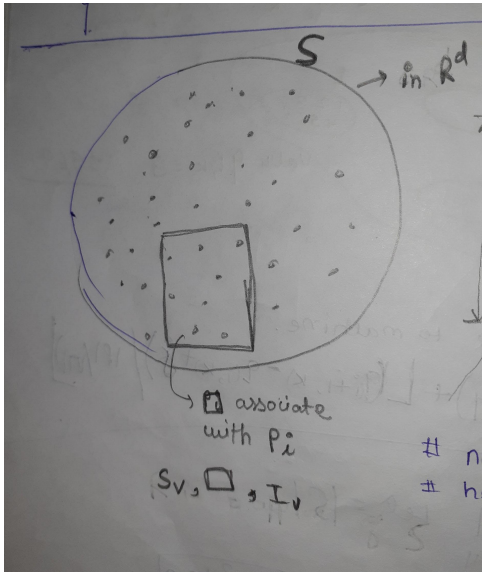
1: If $|I| = 1$, compute $\mathsf{T}(S)$ sequentially.
2: $r = s^{1/5}$.
3: $\mathsf{R} \leftarrow$ `Sample`(S, r, β).
4: $\mathsf{T}^r, \Pi' \leftarrow$ `Partial-kd-tree`(R, r).
5: `Partition`(S, I, Π', β).
6: **for all** $\square_v \in \Pi'$ in parallel **do**
7: `Build-kd-tree`$(S_v, \square_v, I_v)$
8: **end for**

- T^r stored in Machine Beta.
- R is (1/r) -approx. (since sample ( ))
- $|R|=cr^2 \ln n$
- The leaves of T^r partition P into subsets. i.e Π'

Step 3,4:



Step 5: Partition function, Use to store all Π', in contigious machines.
Step 6:



Break condition:
For Loop Iterates until, sub- Π' stored in one machine,.ie $|I_v|=1$.

Results: :

1. R is a (1/r)-approximation with probability at least 1- $1/n^{\Omega(1)}$.
2. Depth of recursion $O(\log_r n) = O(1)$
3. Running Time O(slogs)
4. The total work performed is O(n log n) with probabillity at least $1 - 1/n^{\Omega(1)}$.

Query procedure:

Given a query rectangle ρ and the set of machines I containing T.
Since We already build kd-tree, we have Topsub tree T^r.

1. This gives us a set of leaves of T^r whose squares intersect with ρ.
2. For each such leaf v, the query is performed recursively in parallel on the subtree rooted at v contained in the machines I_v.

Complexity:

- Number of Round = O(1)
- total work $O(n^{1-1/d} + k)$
  - k is the number of points in the query rectangle.
- time required $O(s^{1-1/d} + k')$
  - K' is the maximum number of points reported by a single machine.