3.

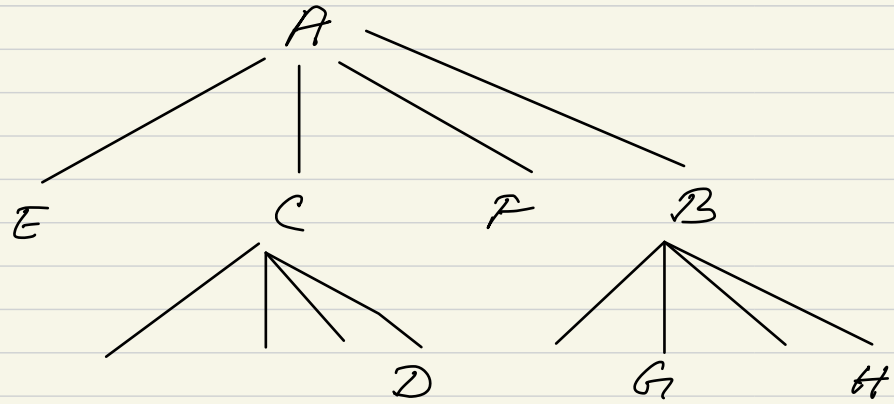
(a) Following are assumed coordinates:

name	x	y
A	35	40
B	32	13
C	45	80
D	72	64
E	7	42
F	26	31
G	83	14
H	79	5

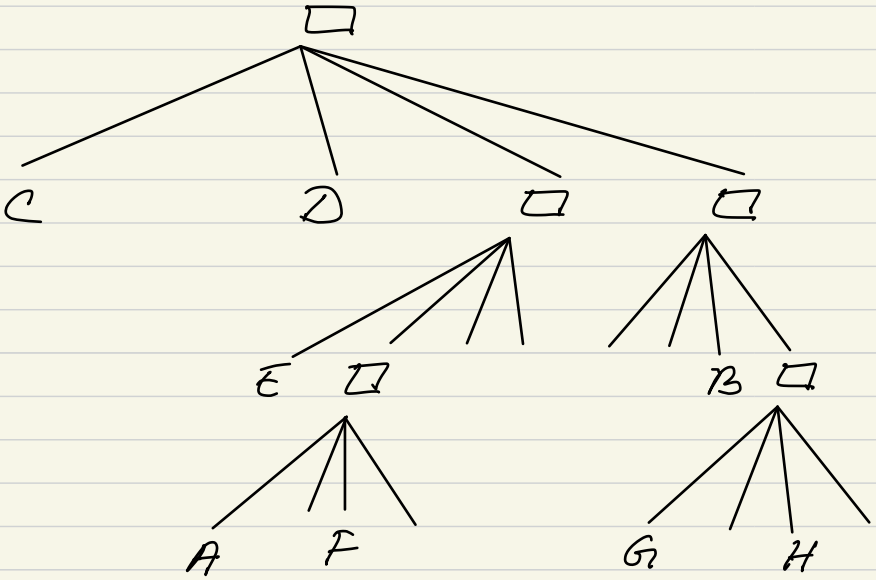
Assuming the following order to generate a Quadtree that has less empty leaf nodes than PR Quadtree based on above coordinates is

A, B, C, D, E, F, G, H

Quadtree : No. of empty leaf = 5



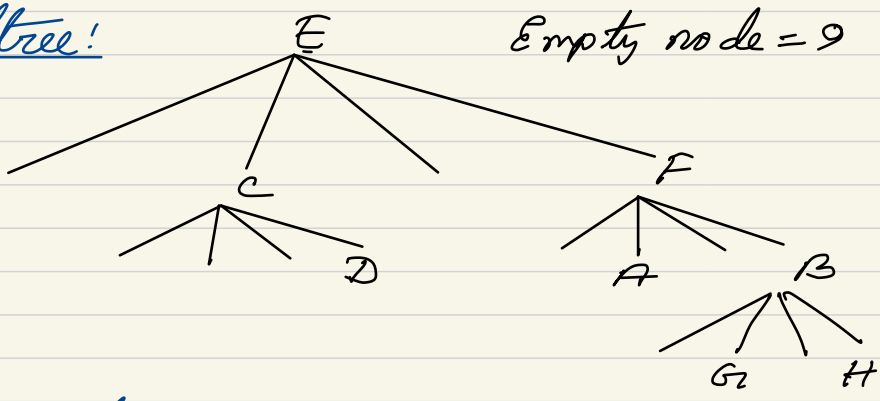
PR Quadtree No. of empty leaf = 8



Assuming the following order which will generate Quadtree with more empty leaf nodes than PR Quadtree:

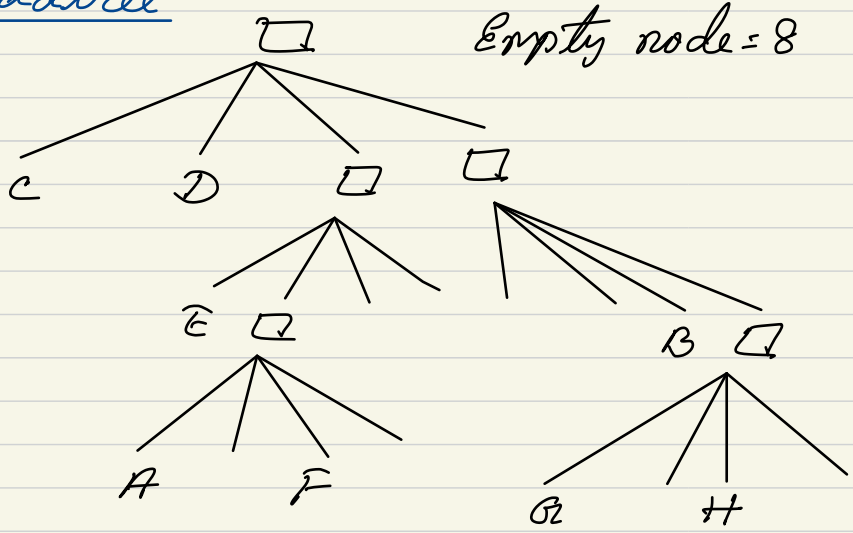
E, F, A, B, C, D, H, G

Quadtree:



Empty node = 9

PR Quadtree



Empty node = 8

(b) Pseudocode for optimized quadtree creation with minimized height is as follows:

OptimizeQuadtree (coordinate, Plane)

→ if (coordinate = nil, Return

→ else calculate (x_i, y_i) coordinate which's distance is minimum from center of plane, i.e. Median

→ if $(x_i, y_i) = 0$, Return

→ else plot (x_i, y_i)

→ $S_W = (0, x_i), (y_i, y)$

→ $S_E = (x_i, x), (y_i, y)$

→ $N_W = (0, x_i), (0, y_i)$

→ $N_E = (x_i, x), (0, y_i)$

→ Remove (x_i, y_i)

→ Recursively call

- optimizeQuadtree (coordinates, SW)
- optimizeQuadtree (coordinates, SE)
- optimizeQuadtree (coordinates, NW)
- optimizeQuadtree (coordinates, NE)

Explanation:

In the above pseudocode, we first derived the median from available dataset, then divided other data points into 4 regions, on each side of median, providing same amount of datapoints. From each of those 4 regions, we calculate the median again and go deeper. This way, the height of the tree is optimized.

(c) As we know, PR Quadtree always has 4 uniform regions, with depth k, probability of finding a particular point at depth k will be

$$\frac{1}{4^k}$$

For collection of V points, the probability that none of the points lies in a given cell at depth k would be

$$\left(1 - \frac{1}{4^k}\right)^V$$