

# A $\text{\LaTeX}$ 'ed solution to 2D Spatial Indexing.

Mitchell Scott (mtscot4)

October 7, 2024

# 2D Spatial Indexing Examples

Assignment 4-1: Quad Trees

Assignment 4-2: kD-Trees

Assignment 4-4: R Trees

# Table of Contents

Assignment 4-1: Quad Trees

Assignment 4-2: kD-Trees

Assignment 4-4: R Trees

# Quad-Tree Set-up

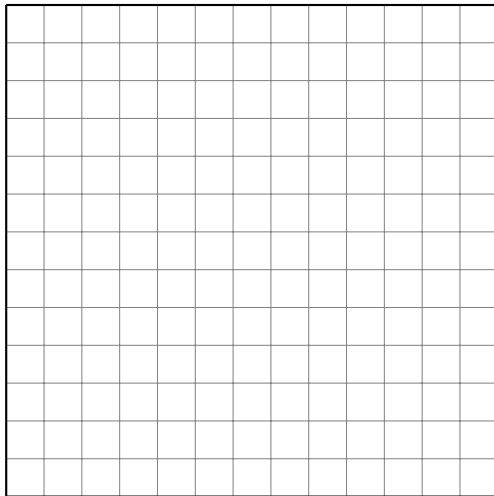
Insert the following points (in this order) into an initially empty Quad-Tree:

$$A = (0, 0), B = (10, 10), C = (8, 2), D = (9, 3), E = (2, 2), \\ F = (6, 2), G = (2, 10), H = (7, 3), I = (5, 5), J = (7, 4)$$

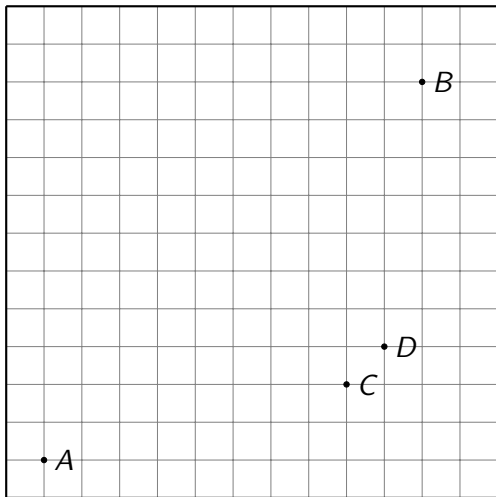
The maximum page capacity of this Quad-Tree is 4.

- ▶ Draw the current Quad-tree after each split.

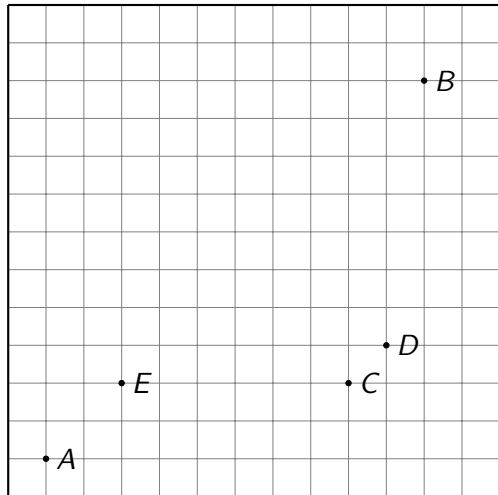
The initially empty Quad-tree has one entry that covers the entire data space, by definition. We decided to make it  $[-1, 12] \times [-1, 12]$ .



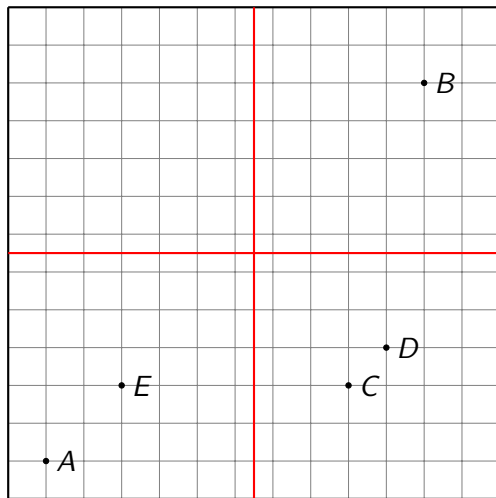
Since the page capacity is 4, we are able to add  $A$ ,  $B$ ,  $C$ , and  $D$  with no hiccups.



**Overflow!** Adding Point  $E$ , means that we will have 5 points in the data space, which is higher than 4. Therefore we have to do a quad-split.

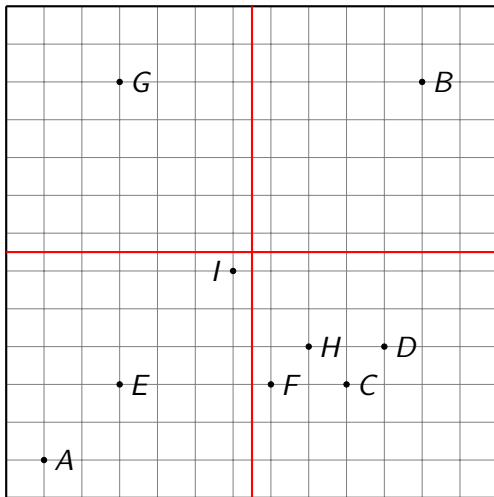


The Quad-tree is agnostic of point density, so we simply bisect the length and the width to create 4 children.

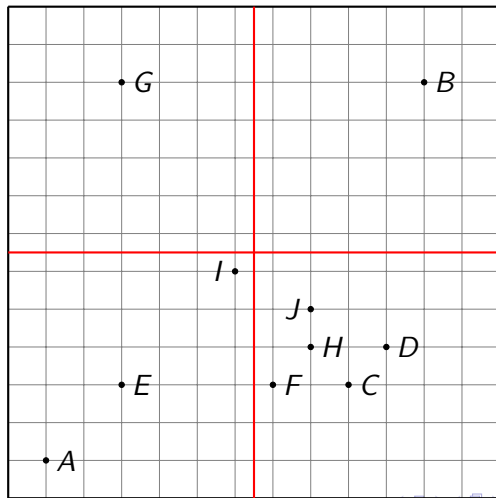




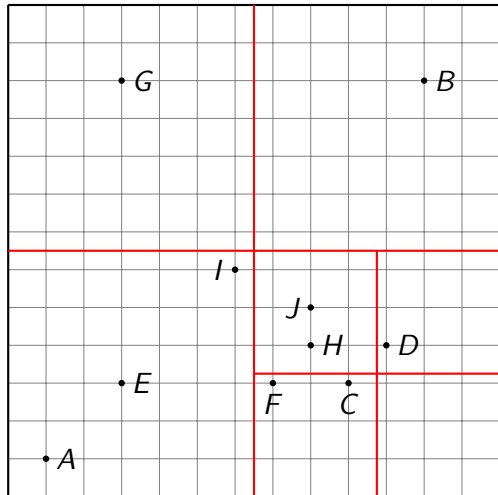
After the quad-split, we are able to add  $F$ ,  $G$ ,  $H$ , and  $I$  to the Quad tree without overflow.



**Overflow!** Adding Point  $J$ , means that we will have 5 points in the Southeast quadrant which is higher than 4. Therefore we have to do a quad-split on just the SE quadrant.



Again, we bisect length and width to separate the SE quadrant. Lastly, we have no more points to add and all boxes have less than 4 points. **DONE! :)**



# Table of Contents

Assignment 4-1: Quad Trees

Assignment 4-2: kD-Trees

Assignment 4-4: R Trees

# kd-Tree Set-up

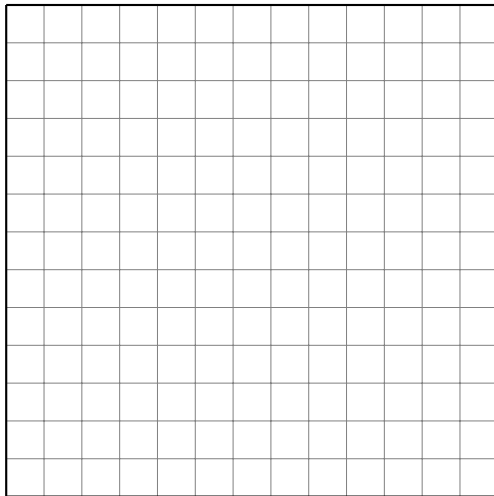
Insert the following points (in this order) into an initially empty Quad-Tree:

$$A = (0, 0), B = (10, 10), C = (8, 2), D = (9, 3), E = (2, 2), \\ F = (6, 2), G = (2, 10), H = (7, 3), I = (5, 5), J = (7, 4)$$

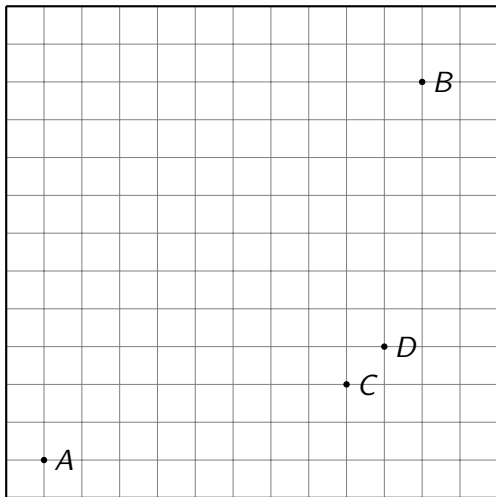
The maximum page capacity of this kd-Tree is 4.

- ▶ Draw the current kd-tree after each split.

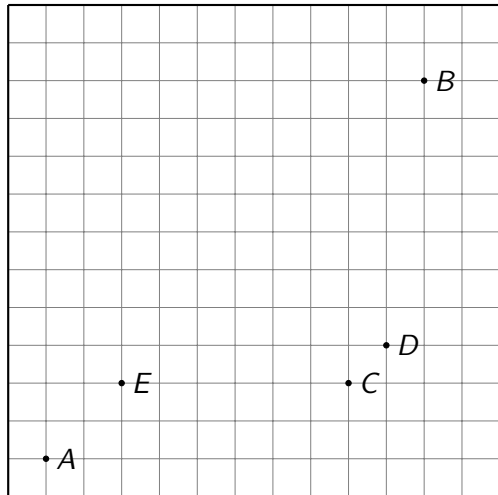
The initially empty kd-tree has one entry that covers the entire data space, by definition. We decided to make it  $[-1, 12] \times [-1, 12]$ .



Since the page capacity is 4, we are able to add  $A$ ,  $B$ ,  $C$ , and  $D$  with no hiccups.

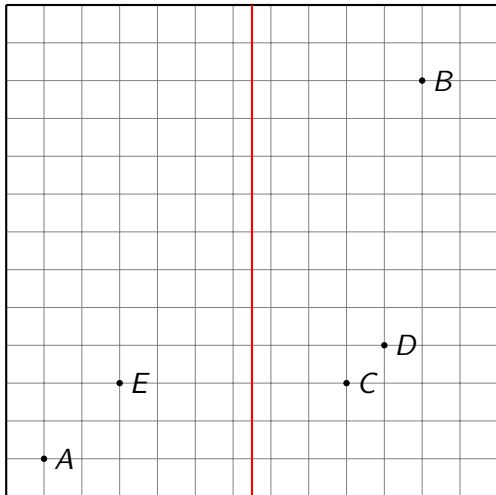


**Overflow!** Adding Point  $E$ , means that we will have 5 points in the data space, which is higher than 4. Since  $\text{depth}(\text{Point } E) \% 2 = 0$ , we will perform a horizontal split.

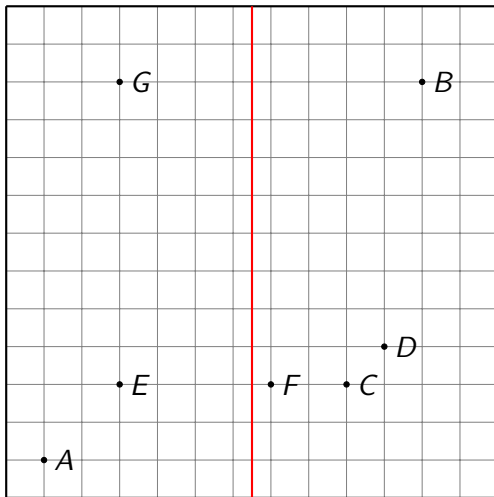




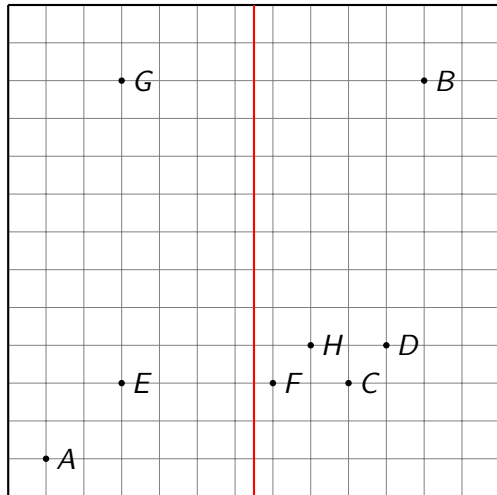
We need to make sure that the vertical line separates the points around the median, so there are 2 points on one side, 3 on the other.



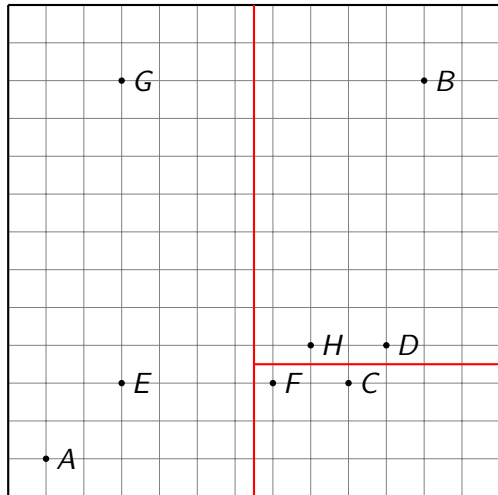
We can add Points  $F$  and  $G$  to the kd-tree, and both nodes still have  $\leq 4$  points.



**Overflow!** Adding Point  $H$ , means that we will have 5 points in the right node, which is higher than 4. Since  $\text{depth}(\text{Point } H) \% 2 = 1$ , we will perform a vertical split.

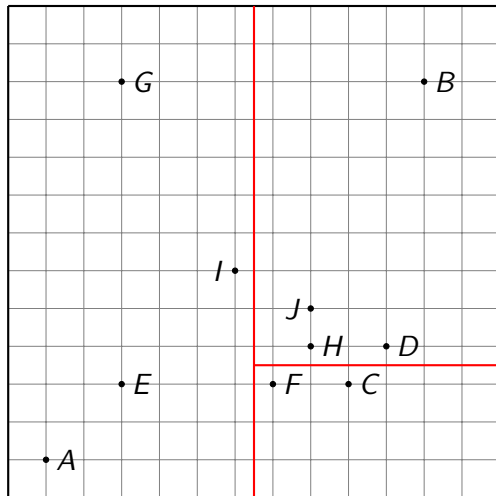


We need to make sure that the horizontal line separates the points around the median, so there are 2 points on one side, 3 on the other.



We add Points *I* and *J* to the kd-Tree. Lastly, we have no more points to add and all nodes have  $\leq 4$  points.

**DONE! :)**



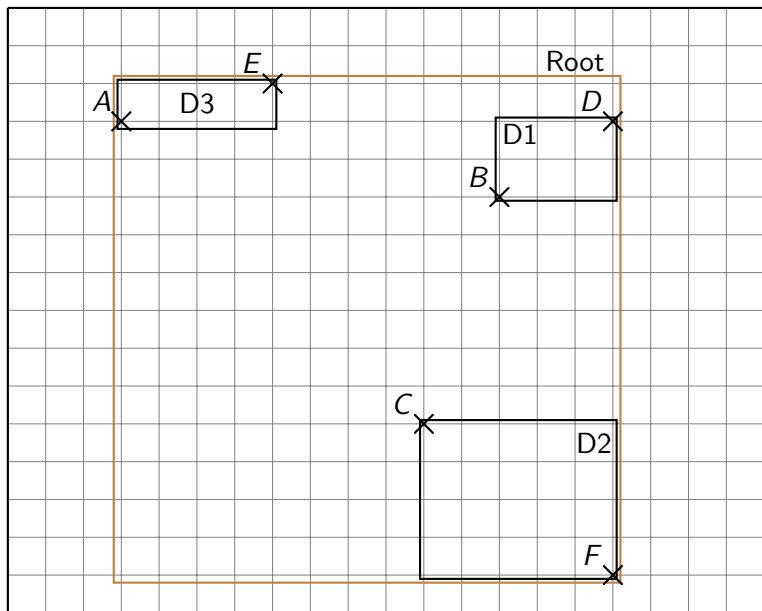
# Table of Contents

Assignment 4-1: Quad Trees

Assignment 4-2: kD-Trees

Assignment 4-4: R Trees

This is the R-tree into which we are inserting points.



We are able to add Points  $G$  and  $H$  into their nodes without overflow.

