Database Systems WS 2023/24
Exercise 7: Distributed 04.12.2023, Due 11.12.2023 12:00 MEZ
Submitted by Erik Schwede, Suma Keerthi

1 Space filling curves for NN-queries (1 P.)

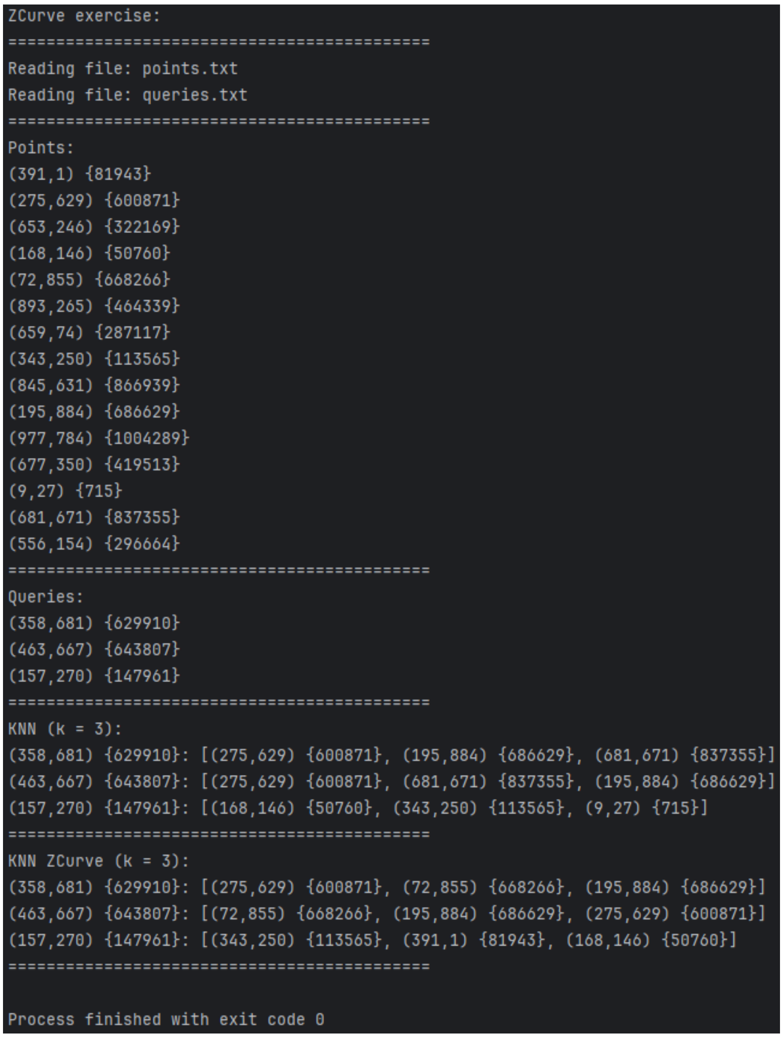
1. Implement space-filling z-curves in a language of your choice. Your program has to take two files which contain points as input. The first file contains all the points of the base data set, while the second file contains query points. The program has to calculate and display:

   - The k-NN, of each query point, in the base data set, based on the actual distance.
   - The k-NN, of each query point, in the base data set, based on the z-curve distance.

   You can use the template in OLAT, which already parses the files and provides utility classes. Submit the code and the output of your program when executed with $k = 3$ and the two data files provided in OLAT. If you do not use the template, also submit instructions on how to compile and execute your program.

   Note: If you use code from external sources, provide the source as a comment.

   Solution:



```
ZCurve exercise:
============================================
Reading file: points.txt
Reading file: queries.txt
============================================
Points:
(391,1) {81943}
(275,629) {600871}
(653,246) {322169}
(168,146) {50760}
(72,855) {668266}
(893,265) {464339}
(659,74) {287117}
(343,250) {113565}
(845,631) {866939}
(195,884) {686629}
(977,784) {1004289}
(677,350) {419513}
(9,27) {715}
(681,671) {837355}
(556,154) {296664}
============================================
Queries:
(358,681) {629910}
(463,667) {643807}
(157,270) {147961}
============================================
KNN (k = 3):
(358,681) {629910}: [(275,629) {600871}, (195,884) {686629}, (681,671) {837355}]
(463,667) {643807}: [(275,629) {600871}, (681,671) {837355}, (195,884) {686629}]
(157,270) {147961}: [(168,146) {50760}, (343,250) {113565}, (9,27) {715}]
============================================
KNN ZCurve (k = 3):
(358,681) {629910}: [(275,629) {600871}, (72,855) {668266}, (195,884) {686629}]
(463,667) {643807}: [(72,855) {668266}, (195,884) {686629}, (275,629) {600871}]
(157,270) {147961}: [(343,250) {113565}, (391,1) {81943}, (168,146) {50760}]
============================================

Process finished with exit code 0
```

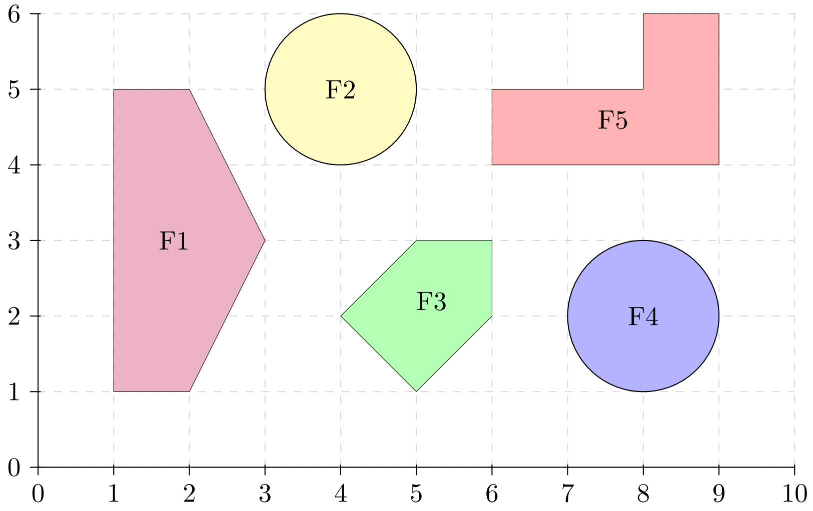
2. Which differences can you see between the two results of your implementation? Explain why are or why are not the results the same.

   Solution:
   Some of the actual k-NN are matching the k-NN according to the z-curve value, but not all of them. The z-curve mapping is only an approximation of the actual k-NN.

   Let's have a look at the points $A(0,0)$, $B(1,0)$, and $C(0,1)$. The real distance between A and B is 1. Also, the z-value is 1 in this case. The real distance between A and C is also 1, but in this case, the z-value is 2. Because of the shape of the z-curve, the z-value is only an estimation of the real distance.

2 R Tree (1 P.)

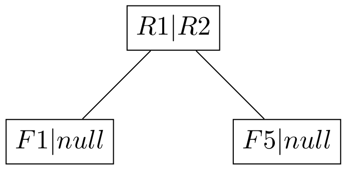
Given the following two-dimensional objects:



For the following R tree operations, explain exactly which steps are performed:

1. *Store the objects F1, F2, F3, F4, F5 in an initially empty R tree.* One node fits 1–2 entries.

   Solution:
   The objects get inserted one after the other; in theory, they can be inserted in any node. I decided for the following insertion order: F1, F5, F4, F2, F3. F1 and F5 get inserted into the tree.

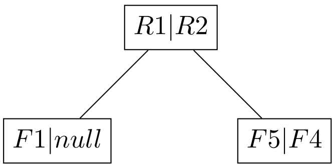


   R1 is the smallest rectangle that fits around F1, and R2 is the smallest rectangle that fits around F5. Insertion of F4:

Calculate the area differences of R1 if F4 gets inserted into R1 and calculate the area difference of R2 if F4 gets inserted.

$$A_{R1} = 8 \cdot 4 - 2 \cdot 4 = 24$$
$$A_{R2} = 5 \cdot 3 - 6 \cdot 4 = 9$$

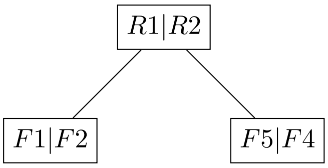
Insert F4 int R2 because the area increase is smaller then when inserted into R2



Insertion of F2:
Calculate the area differences of R1 if F2 gets inserted into R1 and calculate the area difference of R2 if F2 gets inserted.

$$A_{R1} = 4 \cdot 5 - 2 \cdot 4 = 12$$
$$A_{R2} = 5 \cdot 3 - 3 \cdot 5 = 15$$

Insert F2 int R1 because the area increase is smaller then when inserted into R1



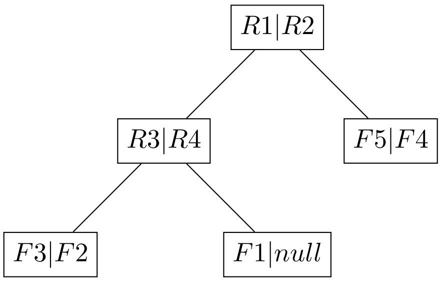
Insertion of F3:
Calculate the area differences of R1 if F2 gets inserted into R1 and calculate the area difference of R2 if F2 gets inserted.

$$A_{R1} = 5 \cdot 5 - 4 \cdot 5 = 5$$
$$A_{R2} = 5 \cdot 5 - 3 \cdot 5 = 10$$

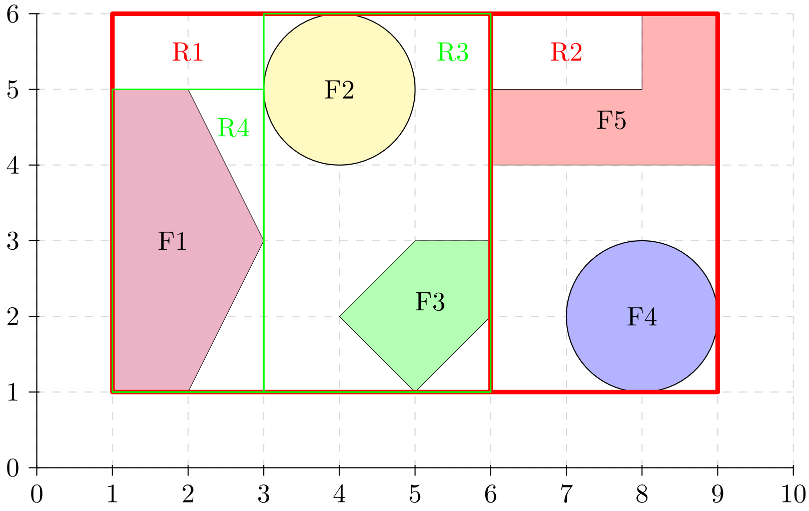
Insert F2 int R1 because the area increase is smaller then when inserted into R1. The maximum of objects in R1 is already reached. In order to insert F2 R1 needs to be split into two rectangles. The size of the two newly created rectangles R3 and R4 should be minimal in the optimal case. Calculation of the area of all possible split combinations:

$$\begin{aligned}
&R3[F1, F2], R4[F3] \; A = 4 \cdot 5 + 2 \cdot 2 = 24\\
&R3[F1, F3], R4[F2] \; A = 5 \cdot 4 + 2 \cdot 2 = 24\\
&R3[F3, F2], R4[F1] \; A = 3 \cdot 5 + 2 \cdot 4 = 23
\end{aligned}$$

According to the Calculation the last option is to optimal one because the area of R3 and R4 will be minimal. The R-Tree looks now like:



In the two dimensional plane the R-Tree looks like the following:



2. *Find all objects, containing the point (6,2).*

Solution:
Step by step execution:

Scan the root of the tree (R1,R2)

Check if (6,2) is in R1 ⇒ true ⇒ put R3 and R4 in the queue

Check if (6,2) is in R2 ⇒ true ⇒ put F5 and F4 in the queue

Check if (6,2) is in R3 ⇒ true ⇒ put F2 and F3 in the queue

Check if (6,2) is in R4 ⇒ false

Check if (6,2) is in F5 ⇒ false

Check if (6,2) is in F4 ⇒ false

Check if (6,2) is in F2 ⇒ false

Check if (6,2) is in F3 ⇒ true ⇒ leaf node ⇒ add F3 to the query result

(6,2) is only in object F3

To check if a point is inside a rectangle we have to check the borders of the rectangle. Lets say R is defined as $R((x_1, y_1), (x_2, y_2))$ and P is defined as $P(x, y)$. Then the following conditions must be true.

$$x_1 \leq x$$
$$y_1 \leq y$$
$$x_2 \geq x$$
$$y_2 \geq y$$

3. *Find all objects that are positioned completely in the rectangle Q, which is defied by the points* $(2, 1)(9, 3)$.

Solution:
Step by step execution:

Scan the root of the tree (R1,R2)

Check if Q is intersecting with R1 ⇒ true ⇒ put R3 and R4 in the queue

Check if Q is intersecting with R2 ⇒ true ⇒ put F5 and F4 in the queue

Check if Q is intersecting with R3 ⇒ true ⇒ put F2 and F3 in the queue

Check if rectangle of F5 is fitting fully into Q ⇒ false

Check if rectangle of F4 is fitting fully into Q ⇒ true ⇒ leaf node ⇒ add F4 to the result

Check if rectangle of F2 is fitting fully into Q ⇒ false

Check if rectangle of F3 is fitting fully into Q ⇒ true ⇒ leaf node ⇒ add F3 to the result

Check if rectangle of F1 is fitting fully into Q ⇒ false

F4 and F3 are the result of the query

To ensure that a rectangle is fully inside the other the following conditions have to be true.

$$Q((x_{q1}, y_{q1}), (x_{q2}, y_{q2})) \text{ The query rectangle}$$
$$R((x_{r1}, y_{r1}), (x_{r2}, y_{R2})) \text{ The rectangle the query should be inside}$$
$$x_{r1} \leq x_{q1}$$
$$y_{r1} \leq y_{q1}$$
$$x_{r2} \geq x_{q2}$$
$$y_{r2} \geq y_{q2}$$

To ensure that Q is intersecting with R the following conditions have to be fulfilled.

$$x_{r1} \leq x_{q2}$$
$$y_{r1} \leq y_{q2}$$
$$x_{r2} \geq x_{q1}$$
$$y_{r2} \geq y_{q1}$$

4. *Find all objects, intersecting with Q':* $(2, 2)(4, 4)$.

   We assume that sharing exactly one point, also counts as intersecting.

Solution:
Step by step execution:

Scan the root of the tree (R1,R2)
Check if Q is intersecting with R1 ⇒ true ⇒ put R3 and R4 in the queue
Check if Q is intersecting with R2 ⇒ false
Check if Q is intersecting with R3 ⇒ true ⇒ put F2 and F3 in the queue
Check if Q is intersecting with R4 ⇒ true ⇒ put F1 in the queue
Check if Q is intersecting with F2 ⇒ true ⇒ leaf ⇒ add F2 to the query result
Check if Q is intersecting with F3 ⇒ true ⇒ leaf ⇒ add F3 to the query result
Check if Q is intersecting with F1 ⇒ true ⇒ leaf ⇒ add F1 to the query result
F2, F3 and F1 are the result of the query

3 Index Structures in Metric Space (1 P.)

Show, for each of the following distance functions, that the properties for being a metric are fulfilled, or provide a counterexample.

1. For two vectors (or points) $a = (x_a, y_a)$ and $b = (x_b, y_b)$:
   $d(a, b) \mapsto |x_a - x_b| + |y_a - y_b|$.

2. For two vectors (or points) $a = (x_a, y_a)$ and $b = (x_b, y_b)$: $d(a, b) \mapsto (x_a - x_b)^2 + (y_a - y_b)^2$.

3. For two strings s_1 and s_2, where S is the set of characters of the string (e.g.: S(“Codd”) = {‘C’, ‘o’, ‘d’}): $d(s_1, s_2) \mapsto 2 * |S(s_1) \cap S(s_2)| \,/\, (|S(s_1)| + |S(s_2)|)$.

Solution: Following are the three properties for the Metric Space Indexing that the distance function d must fulfill:

- Symmetry :
$$d(a, b) = d(b, a)$$
- Non-Negativity :
$$d(a, b) \geq 0, d(a, b) = 0, iff a = b$$
- Triangle Inequality :
$$d(a, c) \leq d(a, b) + d(b, c)$$

1. For two vectors (or points) $a = (x_a, y_a)$ and $b = (x_b, y_b)$:
   $d(a, b) \mapsto |x_a - x_b| + |y_a - y_b|$.
   We are given two vectors a,b with values d(a,b). To show the values are symmetric, as per euclidean distance formula
$$d(a, b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$$

(i) For point $a(xa, ya) = x_a^2 - 2.xa.ya + ya^2 = ya^2 - 2.xa.ya + xa^2 = (y_a, x_a)^2 = |y_a, x_a|$

(ii) For point $b(x_b, y_b) = x_b^2 - 2x_by_b + y_b^2 = y_b^2 - 2x_by_b + x_b^2 = (y_b, x_b)^2 = |y_b, x_b|$

(iii) The distance $d(b, a)$ would be $d(b, a) \mapsto |y_b - y_a| + |x_a - x_b|$.

From (i), (ii) and (iii), we can say that $\mathbf{d(a, b) = d(b, a)}$ which satisfies that the distance between the points fulfills the **symmetric property**.

The second property the distance should fulfill is the non-negativity for which we assume different values of $d(a, b)$ to prove that $d(a, b) \geq 0$.

Let us assume $\mathbf{a(x_a, y_a) = (4, 2)}$ **and** $\mathbf{b(x_b, y_b) = (6, 8)}$, here the values of $x_a \neq x_b$, $y_a \neq y_b$.

(i) So,

$$d(a, b) \mapsto |x_a - x_b| + |y_a - y_b| = \sqrt{(4 - 6)^2 + (2 - 8)^2} = \sqrt{4 + 12} = 4 \geq 0$$

It should also prove that $d(a, b) = 0$ for which we assume a=b to prove $d(a, b) = 0$.

Let us assume $\mathbf{a(x_a, y_a)} = (\mathbf{2}, \mathbf{2})$ and $\mathbf{b(x_b, y_b)}$ **is** $(\mathbf{2}, \mathbf{2})$, $\mathbf{x_a = x_b}$ **and** $\mathbf{y_a = y_b}$

(ii) So,

$$d(a, b) \mapsto |x_a - x_b| + |y_a - y_b| = \sqrt{(2 - 2)^2 + (2 - 2)^2} = \sqrt{0 + 0} = 0$$

From the points (i) and (ii), we see that it describes the **non-negativity property**.

For the query point, we are considering c in this example.

The third property of distance to be fulfilled is the Triangle Inequality. Let us consider the points $d(a, b)$ and $d(b, c)$ to prove this

$$d(a, b) = |x_a - x_b| + |y_a - y_b|$$
$$d(b, c) = |x_b - x_c| + |y_b - y_c|$$

$$d(a, b) + d(b, c) = |x_a - x_b| + |y_a - y_b| + |x_b - x_c| + |y_b - y_c|$$

$$= |x_a - x_b| + |x_b - x_c| + |y_a - y_b| + |y_b - y_c|$$

$$= |x_a - x_b + x_b - x_c| + |y_a - y_b + y_b - y_c|$$

$$= |x_a - x_c| + |y_a - y_c|$$

$$= d(a, c)$$

Hence the above steps prove the **Triangle Inequality property.**

2. For two vectors (or points) $a = (x_a, y_a)$ and $b = (x_b, y_b)$: $d(a, b) \mapsto (x_a - x_b)^2 + (y_a - y_b)^2$.

   Le us consider the points a(2,1), b(1,1) and q(1,2), these points fulfill the **Symmetric**, **Non Negativity and Triangle Inequality** properties of the distance.

   Consider the values of a(2,1), b(1,1) and query point q(1,2)

$$d(a, b) \mapsto (x_a - x_b)^2 + (y_a - y_b)^2$$

$$d(a, q) = (2 - 1)^2 + (1 - 2)^2 = 2$$

$$d(b, q) = (1 - 1)^2 + (2 - 1)^2 = 1$$

$$d(a, b) = (2 - 1)^2 + (1 - 1)^2 = 1 + 0 = 1$$

Symmetric: d(a,b) = d(b,a) = 1 **True**

Non-Negativity: $d(a, b) \geq 0 = 1 \geq 0$ **True**

Triangle Inequality: $d(a, q) \leq d(a, b) + d(b, q) = 2 \leq 1 + 1$ **True**

Hence this example **fulfills all the properties** of the distance between the points.

3. For two strings s_1 and s_2, where S is the set of characters of the string (e.g.: S(“Codd”) = {‘C’, ‘o’, ‘d’}): $d(s_1, s_2) \mapsto 2 * |S(s_1) \cap S(s_2)| \,/\, (|S(s_1)| + |S(s_2)|)$.

   Let us consider the given string **S ={’Codd’}**, the sub-strings that can be derived from S are

   s1 = {‘C’, ‘o’, ‘d’}

   s2 = {‘o’, ‘d’, ‘d’}

   s2 = {‘C’, ‘d’, ‘d’}

   Calculating the d(s1,s2)

$$d(s_1, s_2) \mapsto 2 * |'C','o','d' \cap 'o','d','d'| \,/\, ('C','o','d' + 'o','d','d')$$

$$= 2 * |2|/(3 + 3) = 2/3$$

   Calculating the d(s2,s3)

$$d(s_2, s_3) \mapsto 2 * |'o','d','d' \cap 'C','d','d'| \,/\, ('o','o','d' + 'C','d','d')$$

$$= 2 * |2|/(3 + 3) = 2/3$$

   Calculating the d(s1,s3)

$$d(s_2, s_3) \mapsto 2 * |'C','o','d' \cap 'C','d','d'| \,/\, ('C','o','d' + 'C','d','d')$$

$$= 2 * |2|/(3 + 3) = 2/3$$

   Symmetric: $d(s_1, s_2) = d(s_2, s_1) = 1$ **True**

   Non-Negativity: $d(s_1, s_2) \geq 0 = 2/3 \geq 0$ **True**

   Triangle Inequality: $d(s_1, s_3) \leq d(s_1, s_2) + d(s_2, s_3) = 2/3 \leq 2/3 + 2/3$ **False**

   Hence the this distance in this example **doesn’t fulfill** the **Triangle Inequality property.**

4 Misc Metric Indexing (1 P.)

Below are the points used for the following questions:

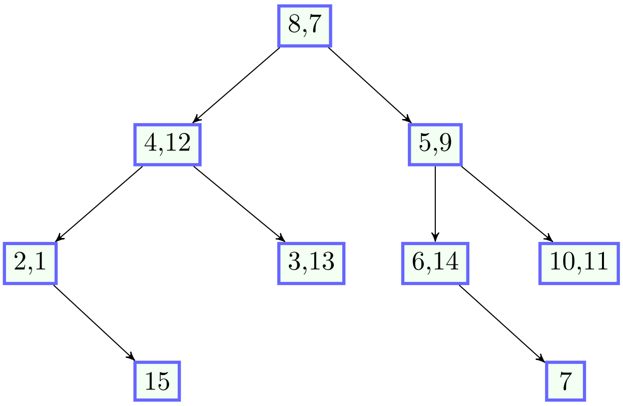
$$1 : (17, 29) \quad 2 : (4, 21) \quad 3 : (20, 29) \quad 4 : (4, 36) \quad 5 : (35, 28)$$

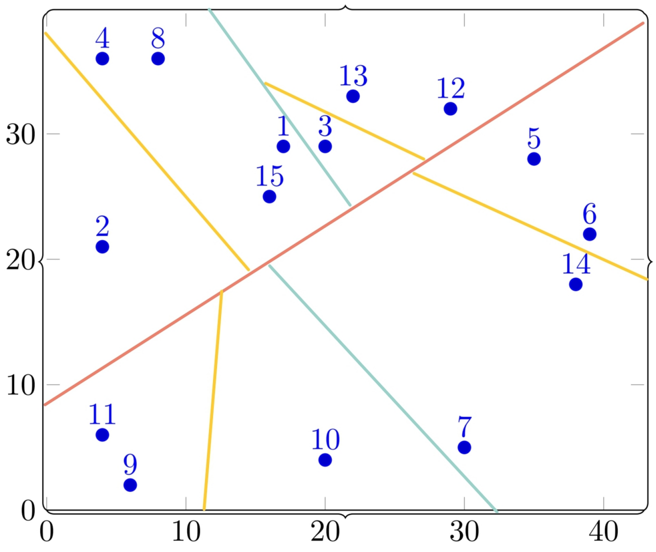
$$6 : (39, 22) \quad 7 : (30, 5) \quad 8 : (8, 36) \quad 9 : (6, 2) \quad 10 : (20, 4)$$

$$11 : (4, 6) \quad 12 : (29, 32) \quad 13 : (22, 33) \quad 14 : (38, 18) \quad 15 : (16, 25)$$

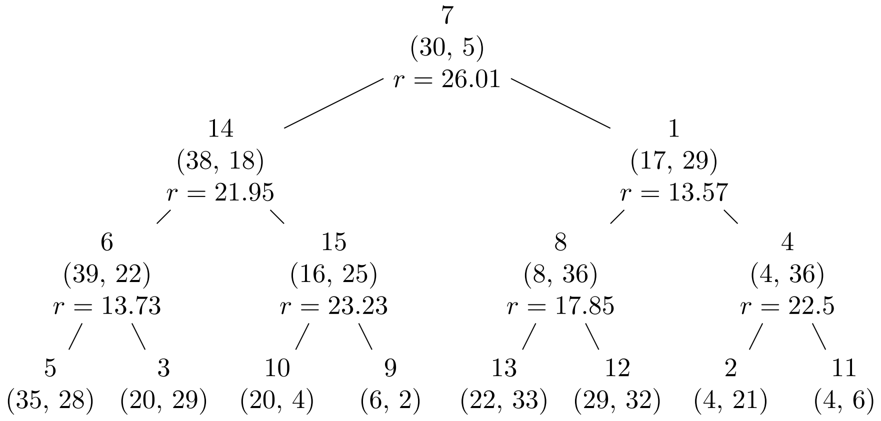
1. **GH Tree partitioning:** Create a GH partitioning, such that the leaf nodes of the tree have at most 2 elements. Draw the tree and draw the partitioning into the plot below.

   Solution:





2. **VP Tree:** Given the following VP tree, search for query point (25,4) with $\varepsilon = 11$. Describe which parts of the tree you pruned and why.



Solution:
The Euclidean distance formula for calculating distance between two co-ordinates (x_1, y_1) and (x_2, y_2) are

$$d(p,q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

We have to define if we want to go through the left sub-tree or right sub-tree. Formula for calculating the left sub-tree is

$$max\{d(p,q) - r, 0\} \leq \epsilon$$

Formula for calculating the right sub-tree is

$$max\{r - d(p,q), 0\} \leq \epsilon$$

- Distance between Root node **7:(30,5) and Query point (25,4)** with $r = 26.01$ and $\epsilon = 11$

  $$d(p,q) = \sqrt{(25-30)^2 + (4-5)^2} = 5.09 \leq 11 \quad \textbf{Add the node to tree}$$

  Verifying if we should go through left sub-tree

  $$d(7,q) - r = 5.09 - 26.01 = -20.91$$

  $$max\{d(p,q) - r, 0\} \leq \epsilon = max\{d(7,q) - r, 0\} = max\{-20.91, 0\} \leq 11 \quad \textbf{Go through left sub-tree}$$

  Verifying if we should go through right sub-tree

  $$r - d(7,q) = 26.01 - 5.09 = 20.92$$

  $$max\{r - d(p,q), 0\} \leq \epsilon = max\{, 0\} = max\{20.92, 0\} \nleq 11 \quad \textbf{Prune right sub-tree}$$

  Now we only consider the co-ordinates of the left sub-tree for the query point as we have pruned the right sub-tree

- Distance between **14: (38,18) and Query point (25,4)** with r = 21.95 and $\epsilon = 11$

  $$d(14,q) = 19.10 \nleq 11 \quad \textbf{don't add node}$$

  Verify left sub-tree

  $$max\{-2.84, 0\} = 0 \leq 11 \quad \textbf{traverse through the left sub-tree}$$

  Verify right sub-tree

  $$max\{2.85, 0\} = 2,85 \leq 11 \quad \textbf{traverse through the right sub-tree}$$

- Distance between **6: (39,22) and Query point (25,4)** with $r = 13.73$ and $\epsilon = 11$

  $$d(6, q) = 22.80 \nleq 11 \quad \textbf{don't add node}$$

  Verify left sub-tree

  $$max\{9.52, 0\} = 9.52 \leq 11 \quad \textbf{traverse through the left sub-tree}$$

  Verify right sub-tree

  $$max\{-9.07, 0\} = 0 \leq 11 \quad \textbf{traverse through the right sub-tree}$$

- Distance between **15: (16,25) and Query point (25,4)** with r = 23.23 and $\epsilon = 11$

  $$d(15, q) = 22.84 \nleq 11 \quad \textbf{don't add node}$$

  Verify left sub-tree

  $$max\{0.89, 0\} = 0 \leq 11 \quad \textbf{traverse through the left sub-tree}$$

  Verify right sub-tree

  $$max\{0.39, 0\} = 0 \leq 11 \quad \textbf{traverse through the right sub-tree}$$

- Distance between **5: (35,28) and Query point (25,4)** $\epsilon = 11$

  $$d(5, q) = 26 \nleq 11 \quad \textbf{don't add node}$$

- Distance between **3: (20,29) and Query point (25,4)** $\epsilon = 11$

  $$d(3, q) = 25.49 \nleq 11 \quad \textbf{don't add node}$$

- Distance between **10: (20,4) and Query point (25,4)** $\epsilon = 11$

  $$d(10, q) = 5 \leq 11 \quad \textbf{add node}$$

- Distance between **9: (6,2) and Query point (25,4)** $\epsilon = 11$

  $$d(6, q) = 19.10 \nleq 11 \quad \textbf{don't add node}$$

  Hence **7 and 10** nodes would be added to the result.