Point Quadtrees

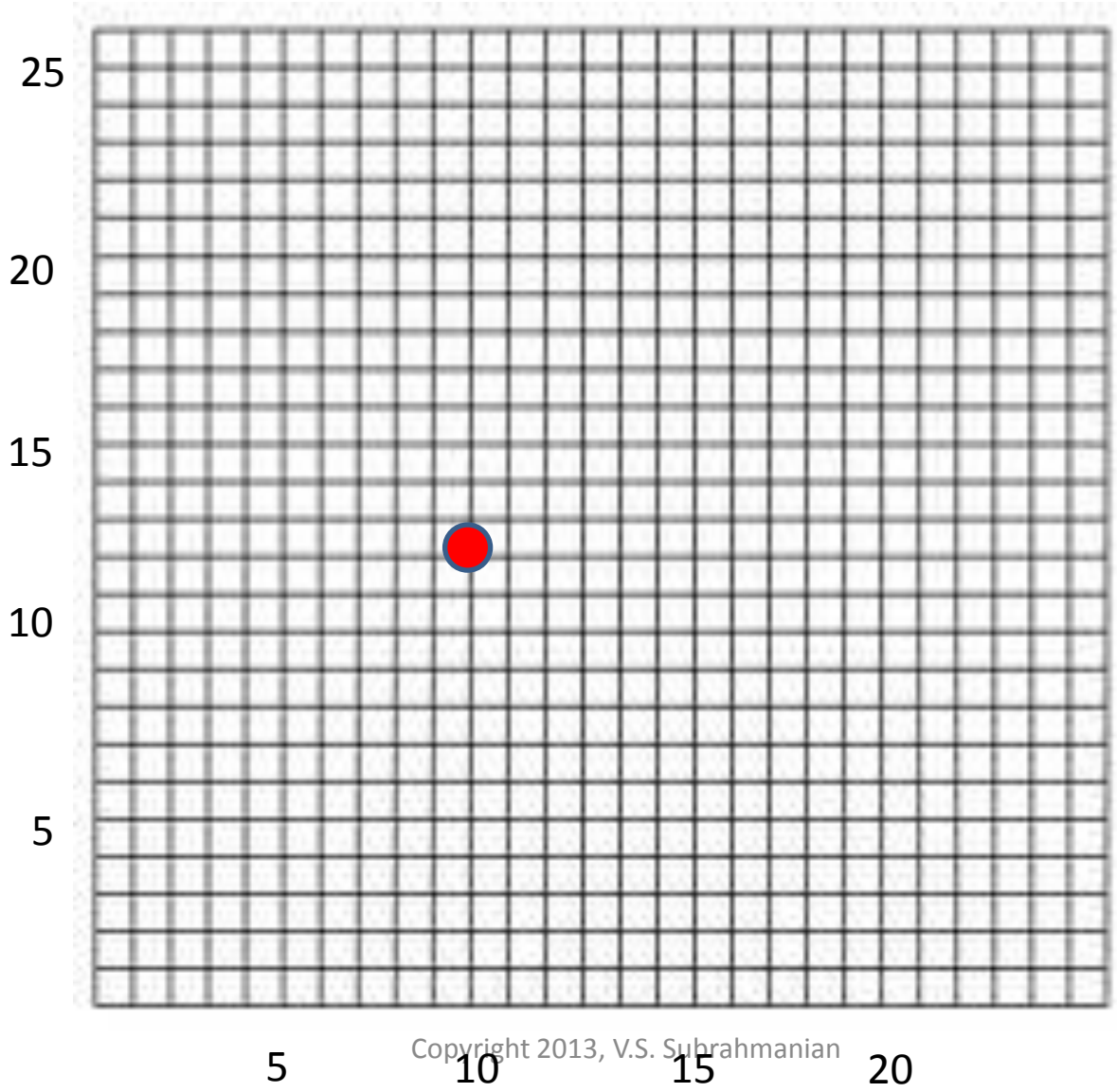
V.S. Subrahmanian

Spring 2013

Definition of a Point Quadtree

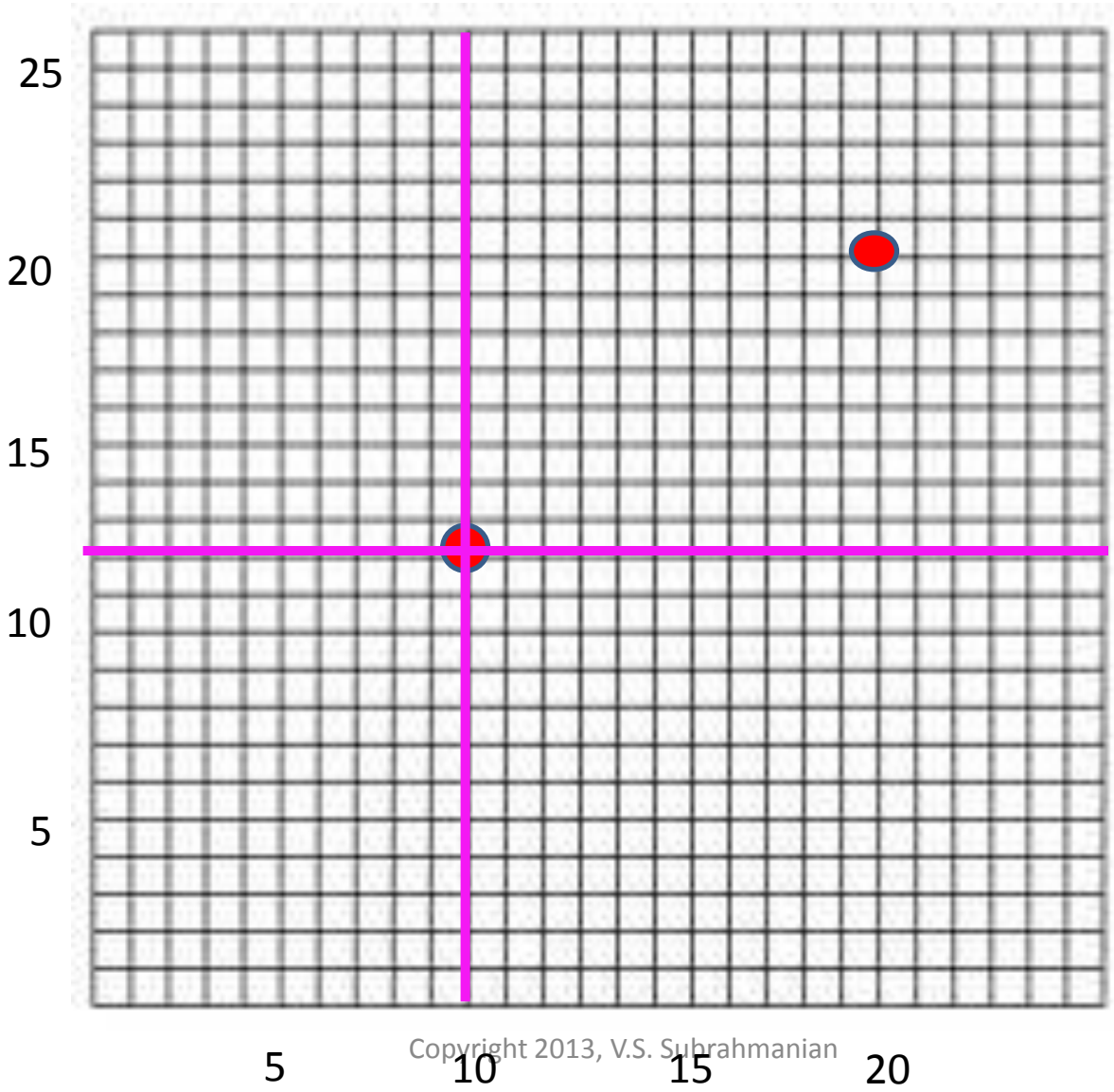
- Point quadtrees are used to represent 2-dimensional data from **R x R.**
- A point quadtree is a tree of arity 4 where each node has (at least) the following fields
  - POINT
  - NW,NE,SW,SE pointer fields to point quadtrees
  - For all non-leaf nodes N,
    - All points in N.NW are to the NW of N.POINT
    - All points in N.SW are to the SW of N.POINT
    - All points in N.SE are to the SE of N.POINT
    - All points in N.NE are to the NE of N.POINT

Point Quadtree Example: Insert (10,12)

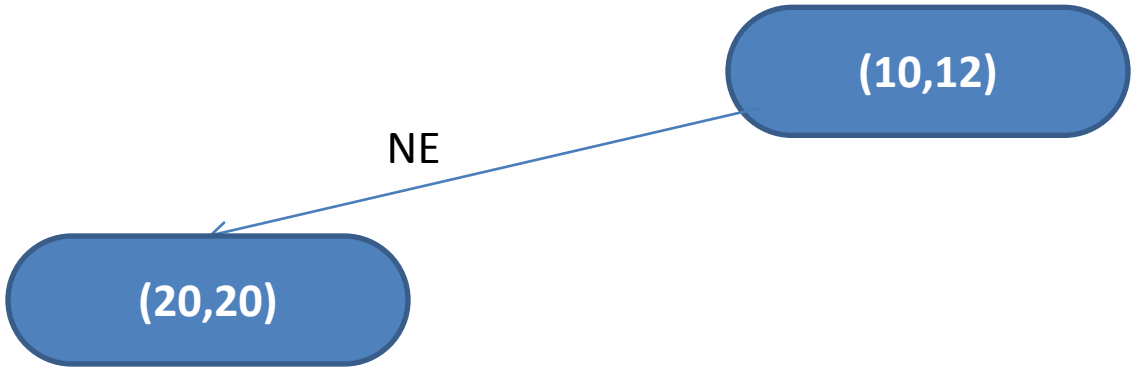


Insert (10,12)

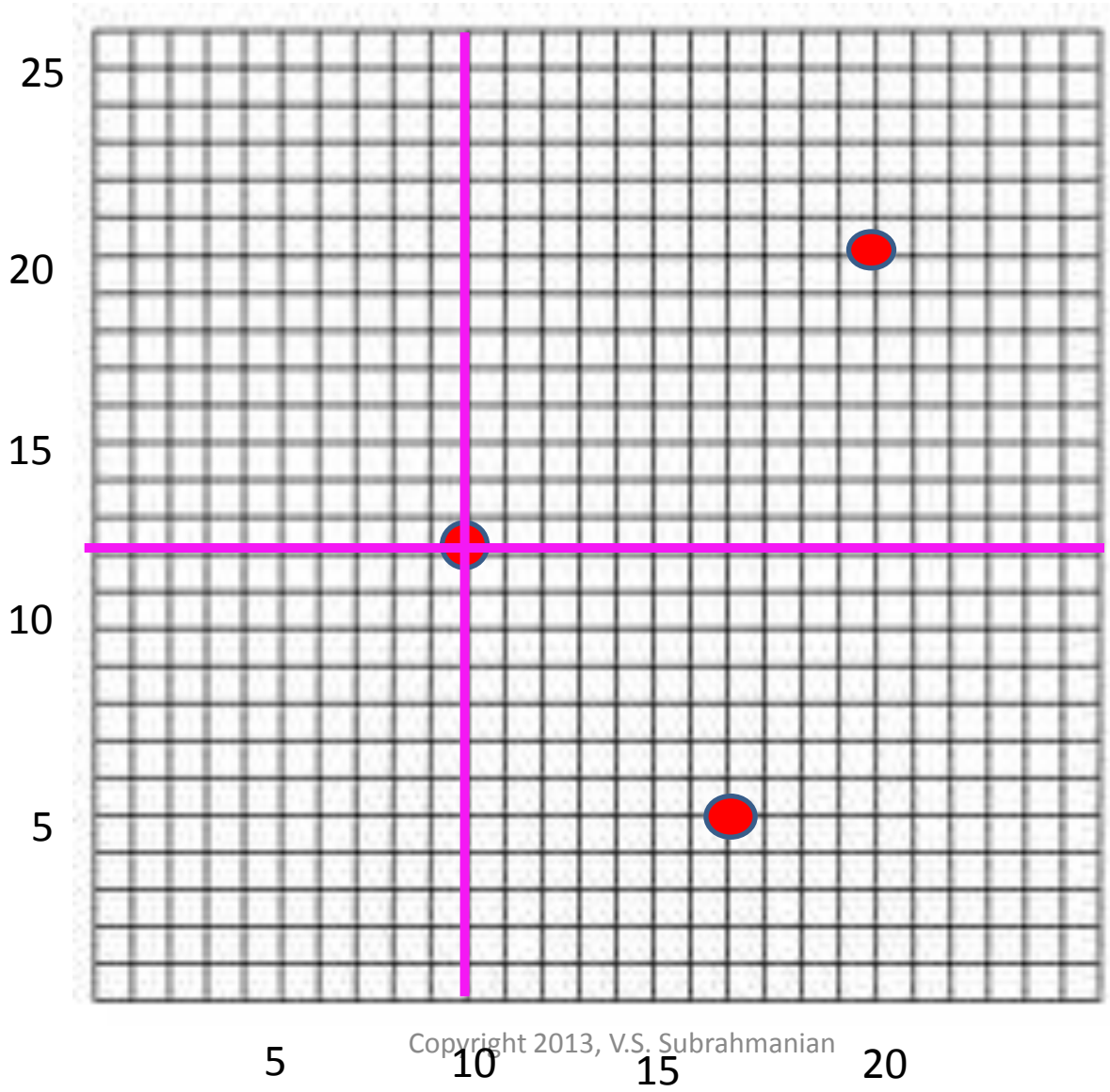
Point Quadtree Example: Insert (20,20)



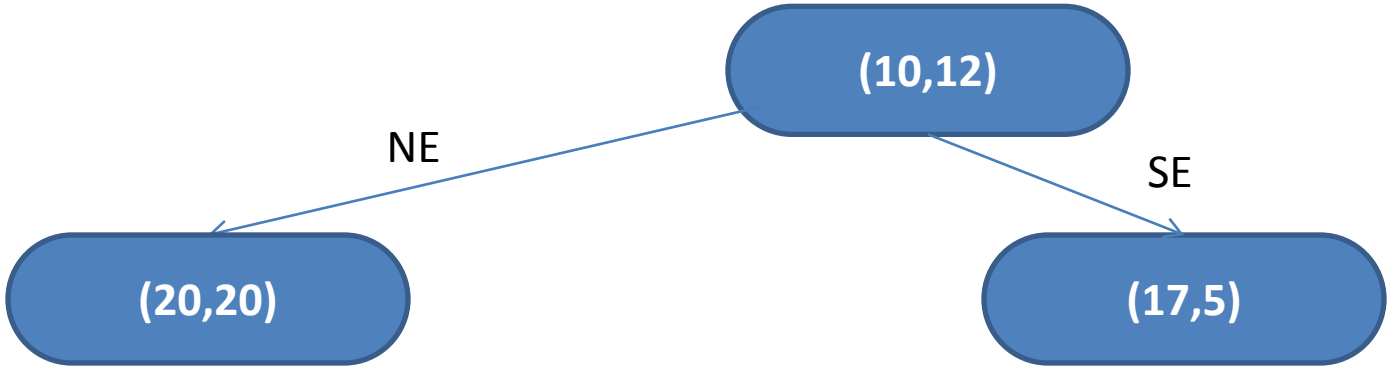
Insert (20,20)



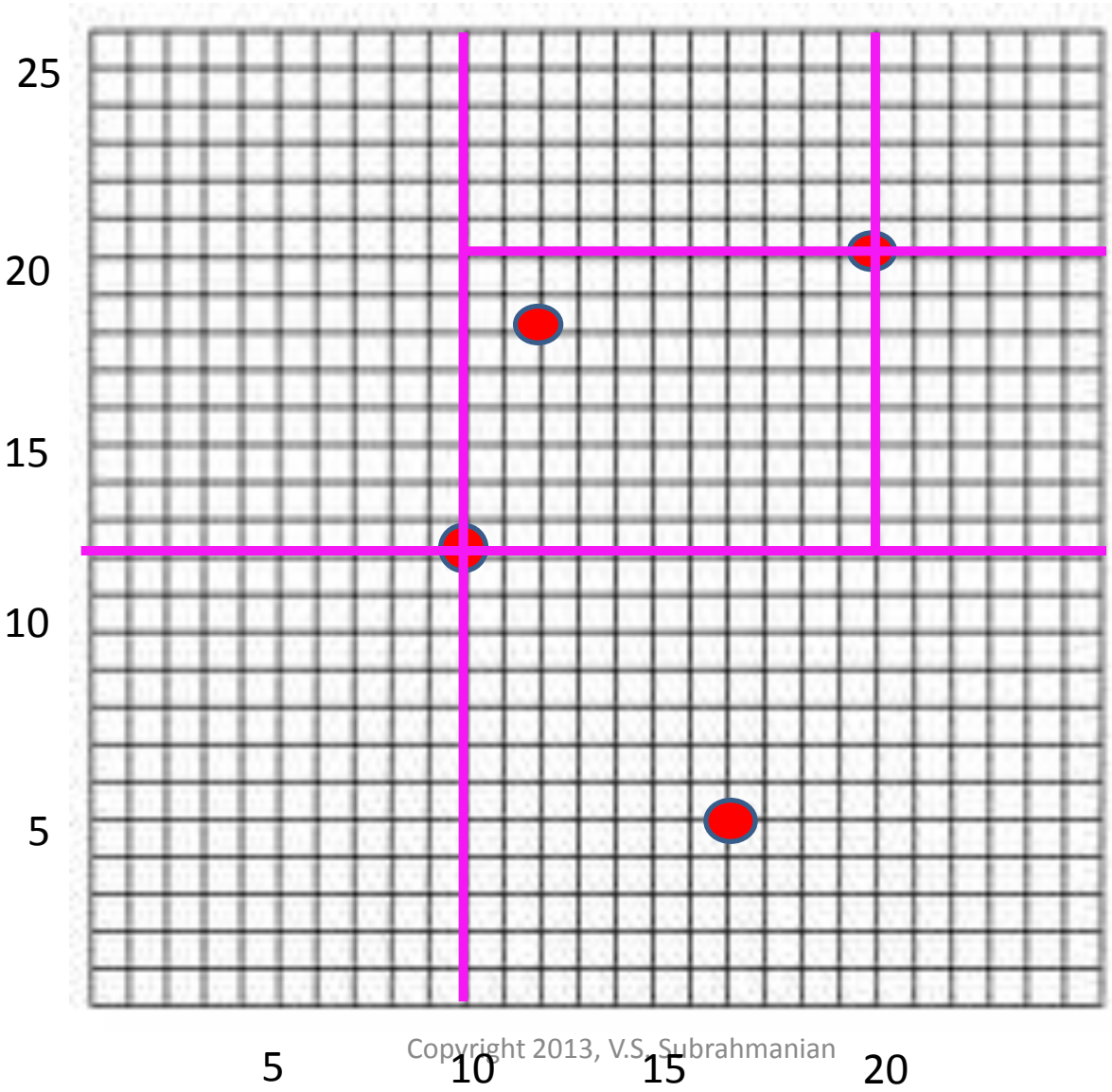
Point Quadtree Example: Insert (17,5)



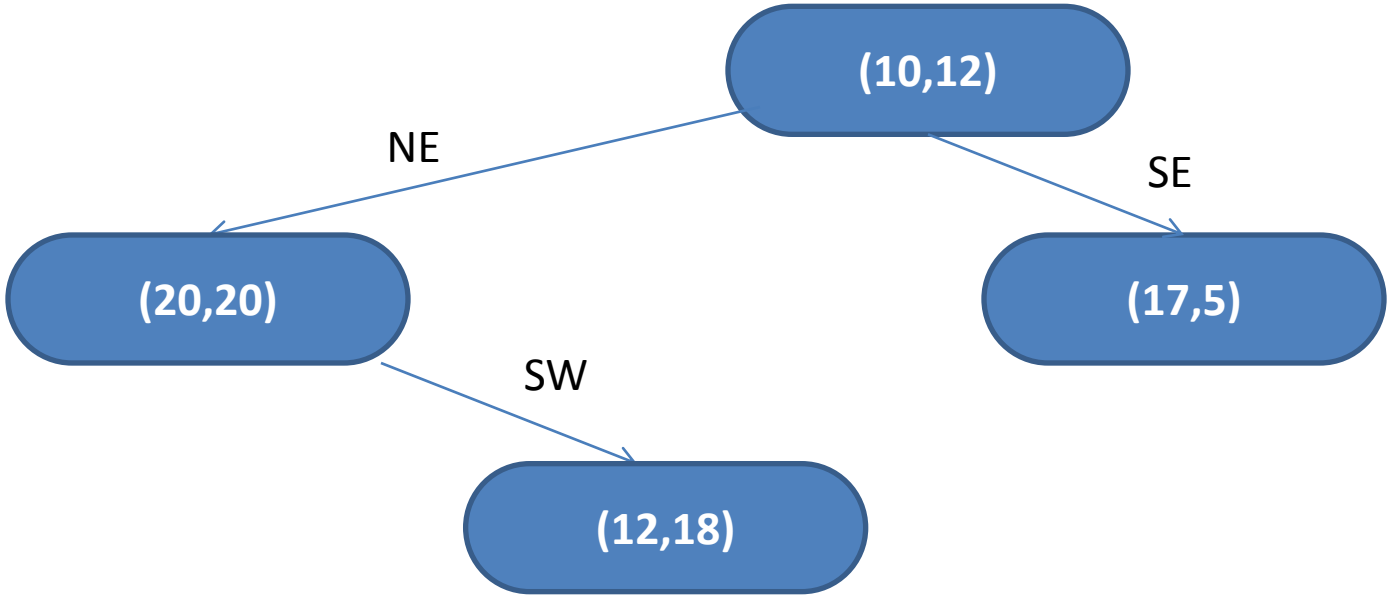
Insert (17,5)



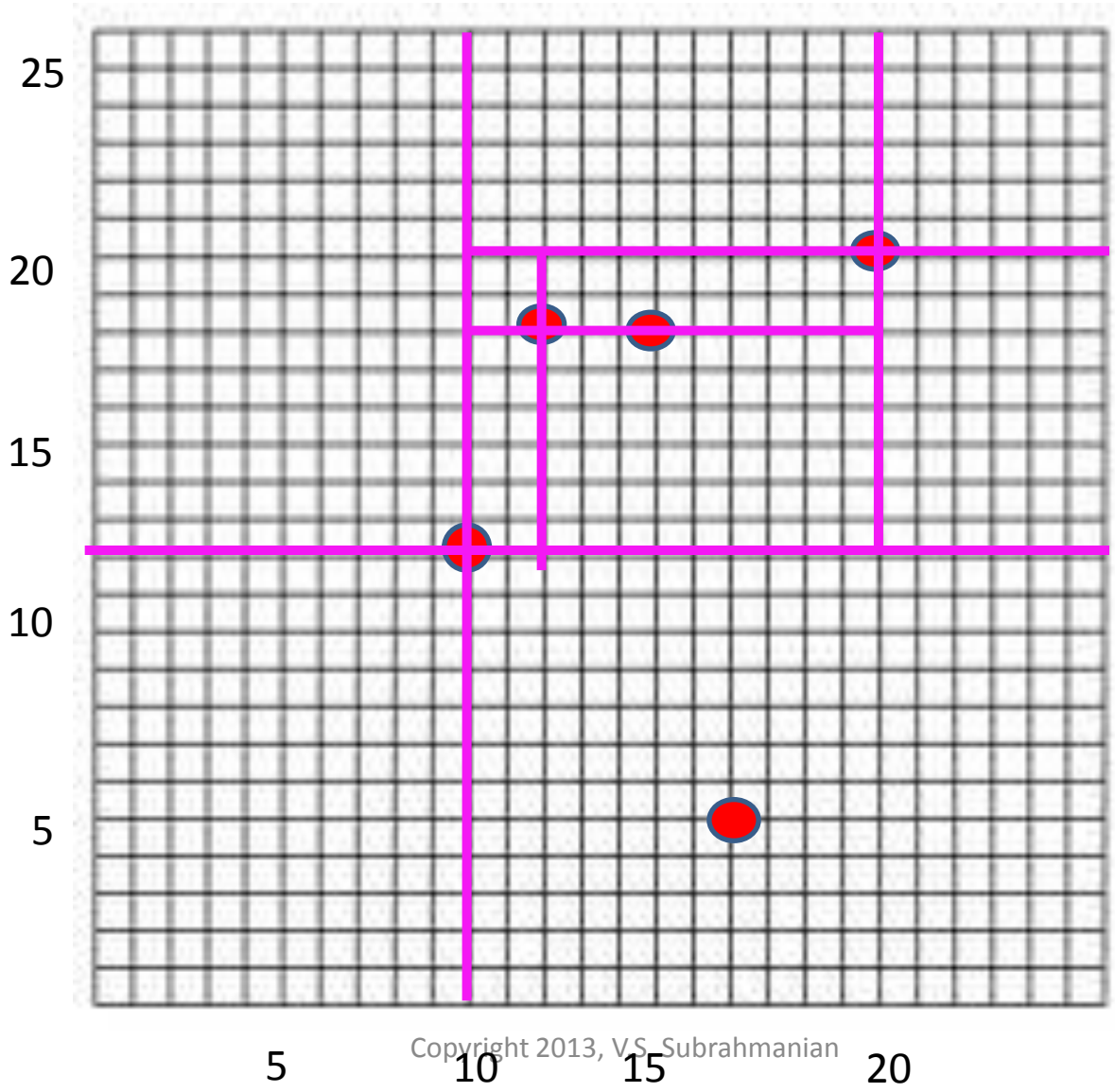
Point Quadtree Example: Insert (12,18)



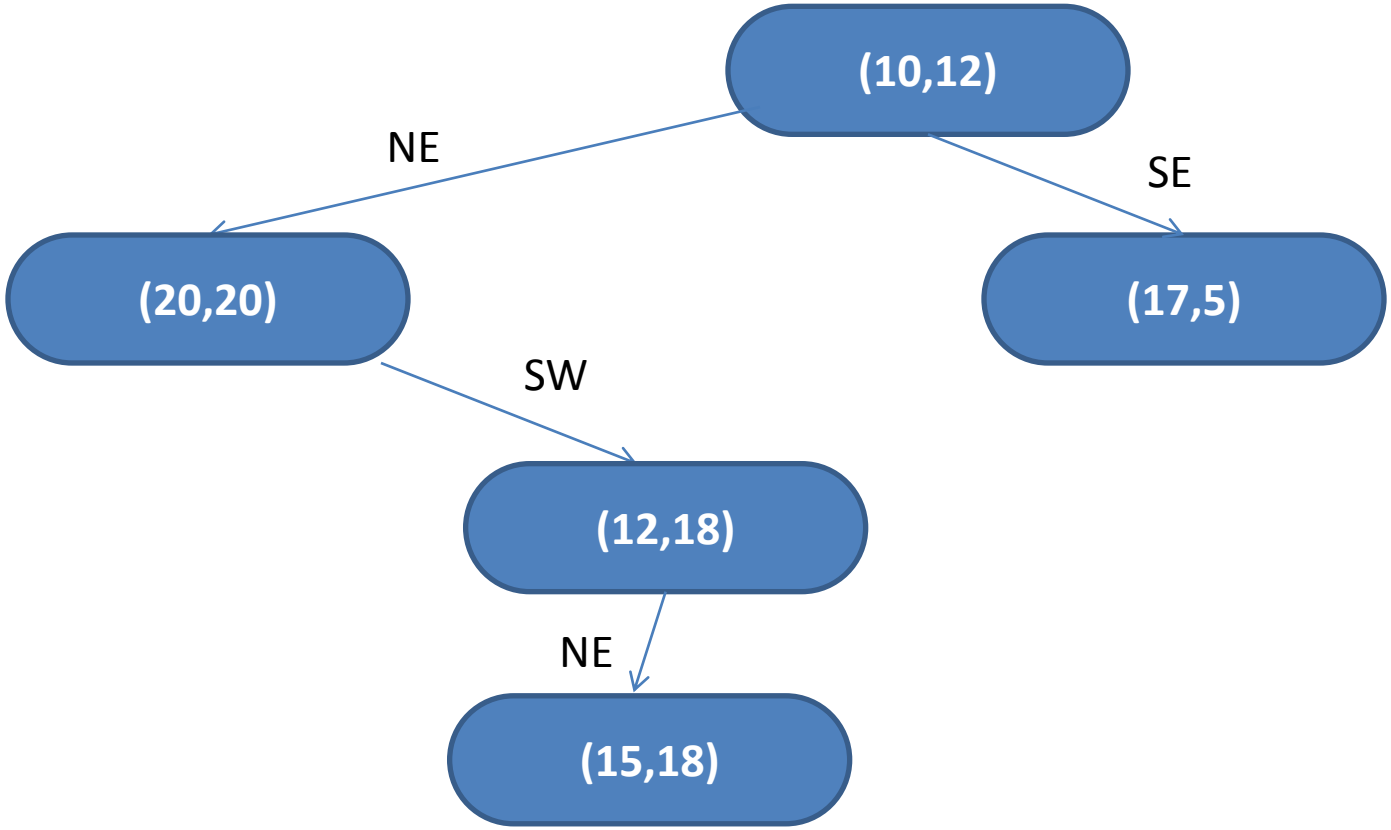
Insert (12,18)



Point Quadtree Example: Insert (15,18)

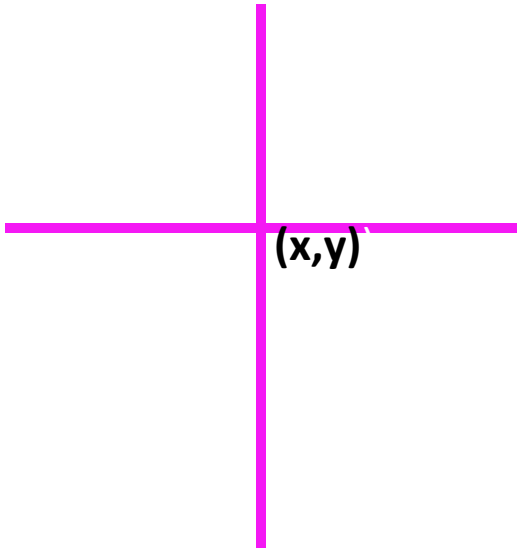


Insert (15,18)



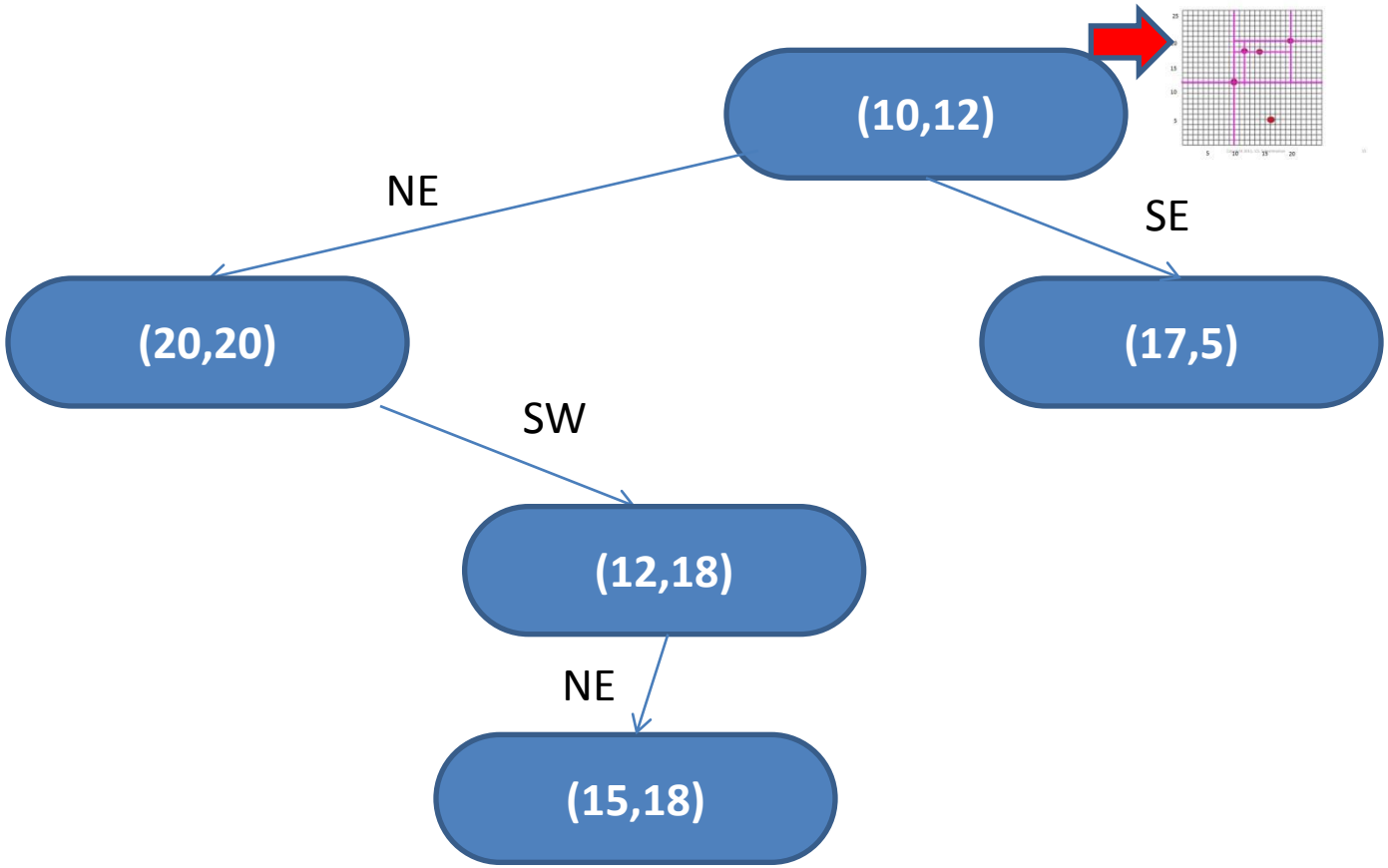
Branching Convention

- If you are inserting point (x',y') and you are at a node labeled (x,y), branch to
  - NE if x' ≥ x & y' ≥ y.
  - NW if x' < x & y' ≥ y.
  - SE if x' ≥ x & y' < y
  - SW if x' < x & y' < y.



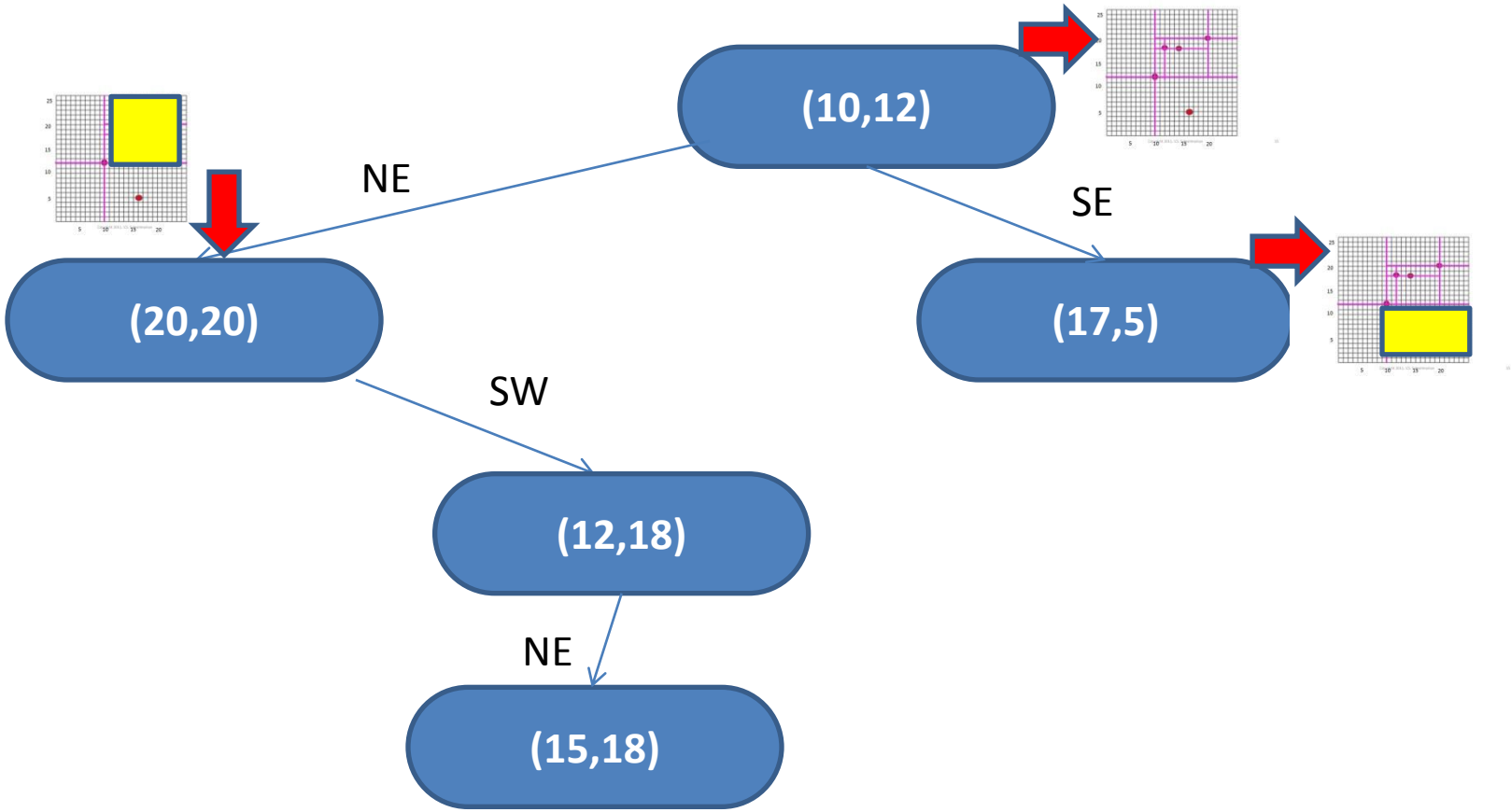
Intuitively, quadrants are closed on the left and bottom, open on the right and top.

Regions are associated with each node



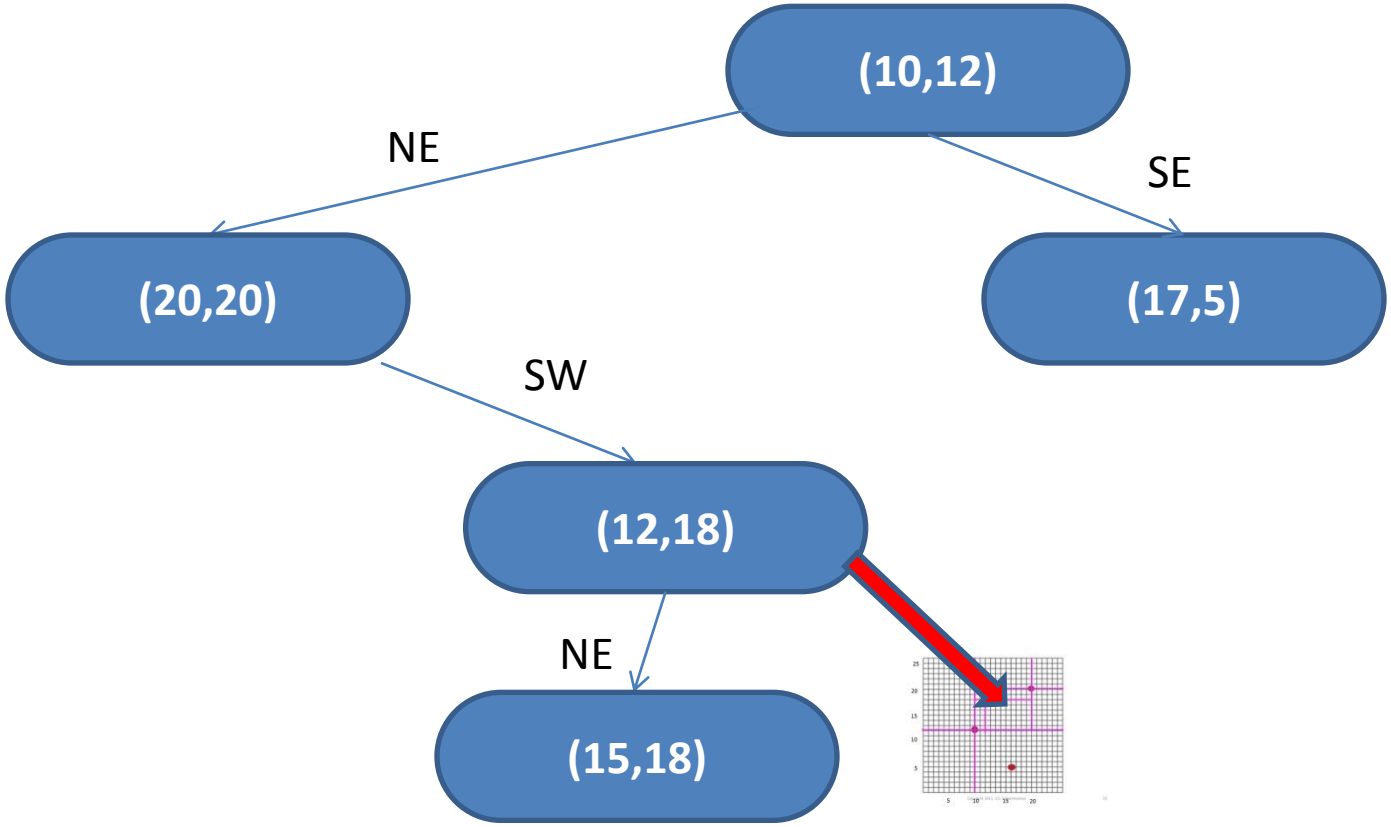
As in the case of BSTs and B-trees, each node in a point quadtree implicitly associates a region REG(N) with each node N. This region is always a rectangle. The root represents the entire region.

Regions are associated with each node



Region(Root.NE) is the northeast quadrant after splitting the region associated with the root by drawing a vertical and horizontal line through Root.Point. Similarly for SE child of root.

Regions are associated with each node



In-class exercise

- Insert (5,5).
- Insert (2,17)
- Insert (14,16).
- Insert (14,19).

Range Queries

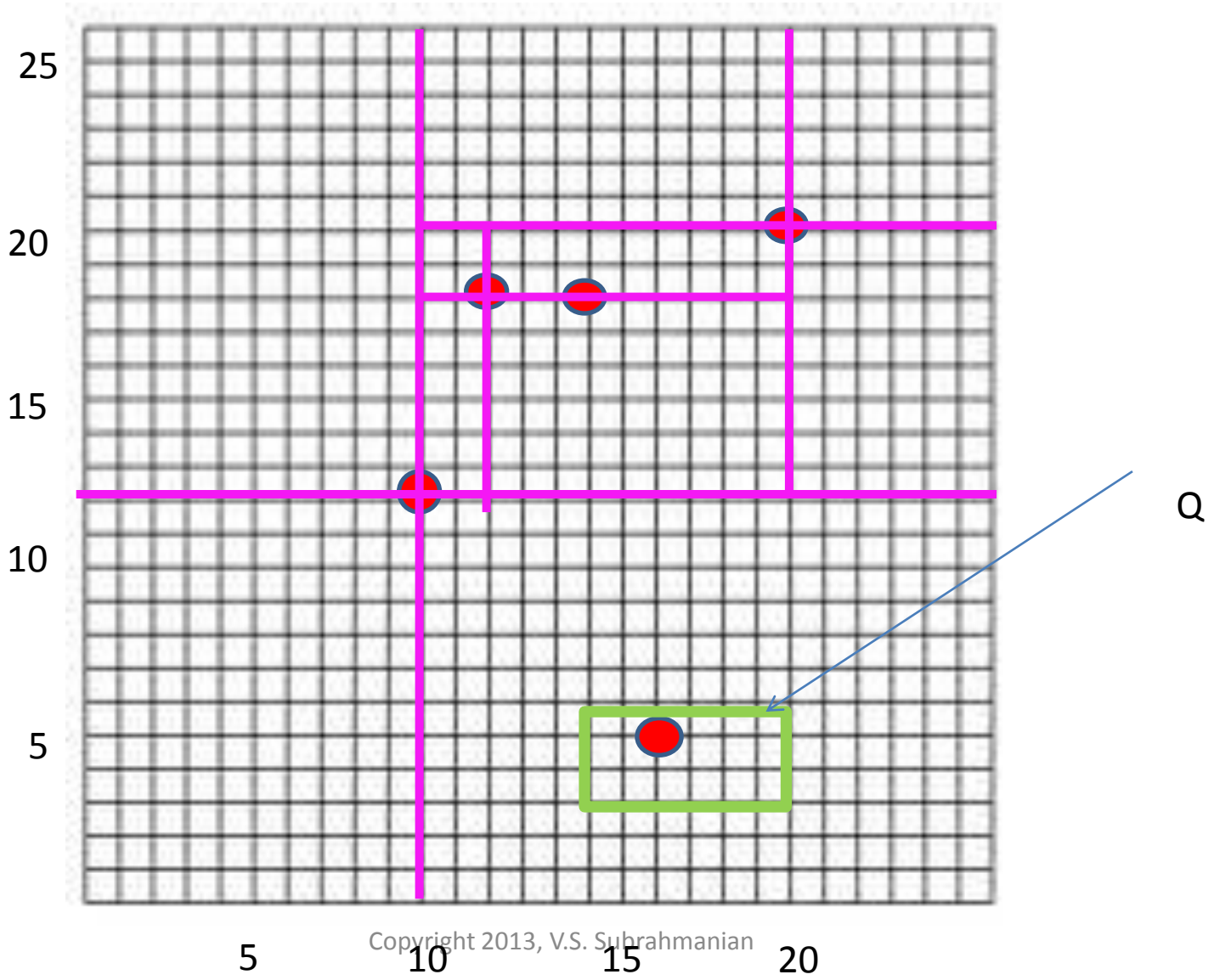
- In a *range* query, you are given a query region *Q* and a pointer to the root of a point quadtree.
- You must find all points in *Q* that are within the specified query region *Q.*

Range Query Algorithm Sketch

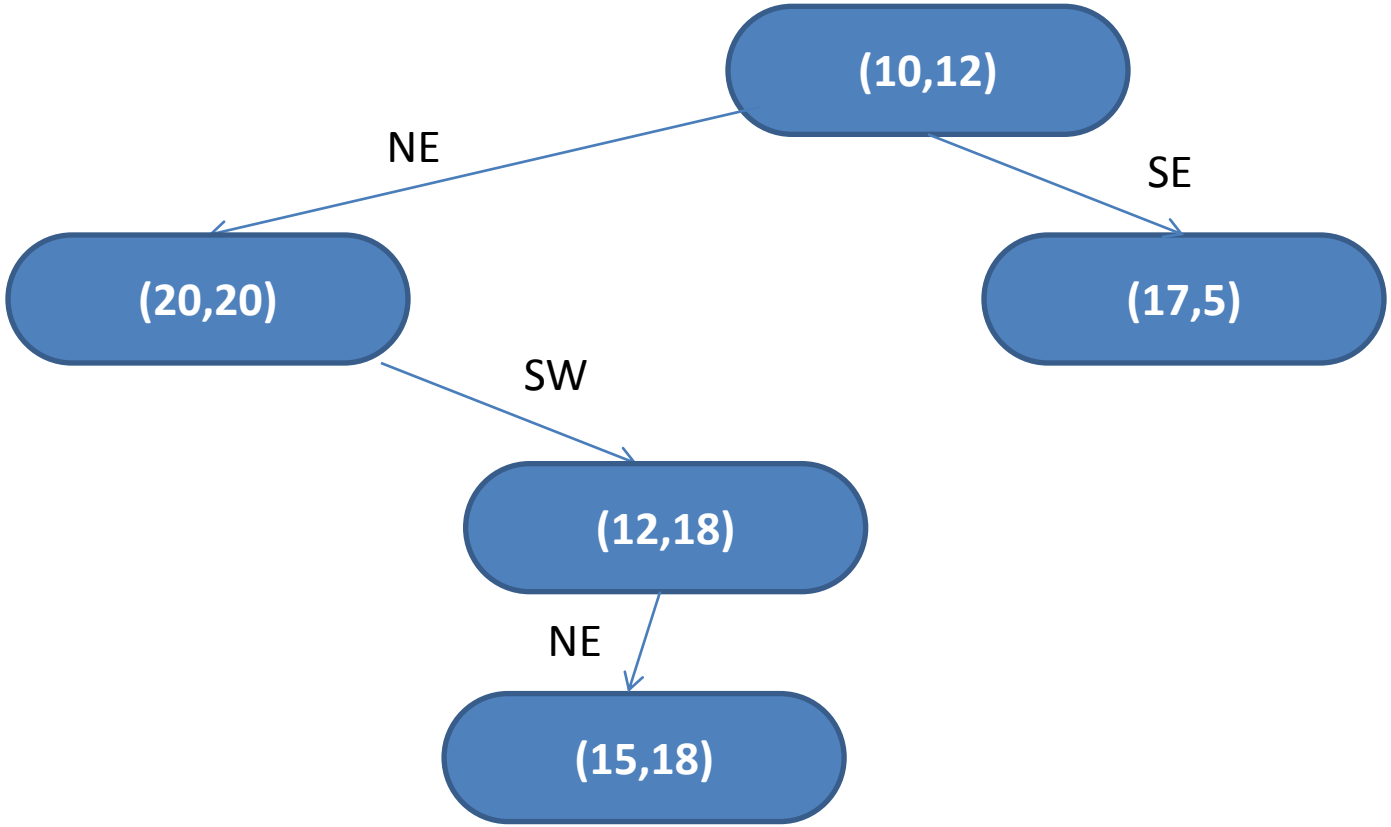
Visit-Node(N)

- If Region(N) intersects Q then
  - check if N.POINT is in Q
  - If yes, add to solution
  - If not, visit all 4 children and repeat recursively.
- Algorithm can change a bit depending on what “shape” Q has. More on this later.

Point Quadtree Example Region Query

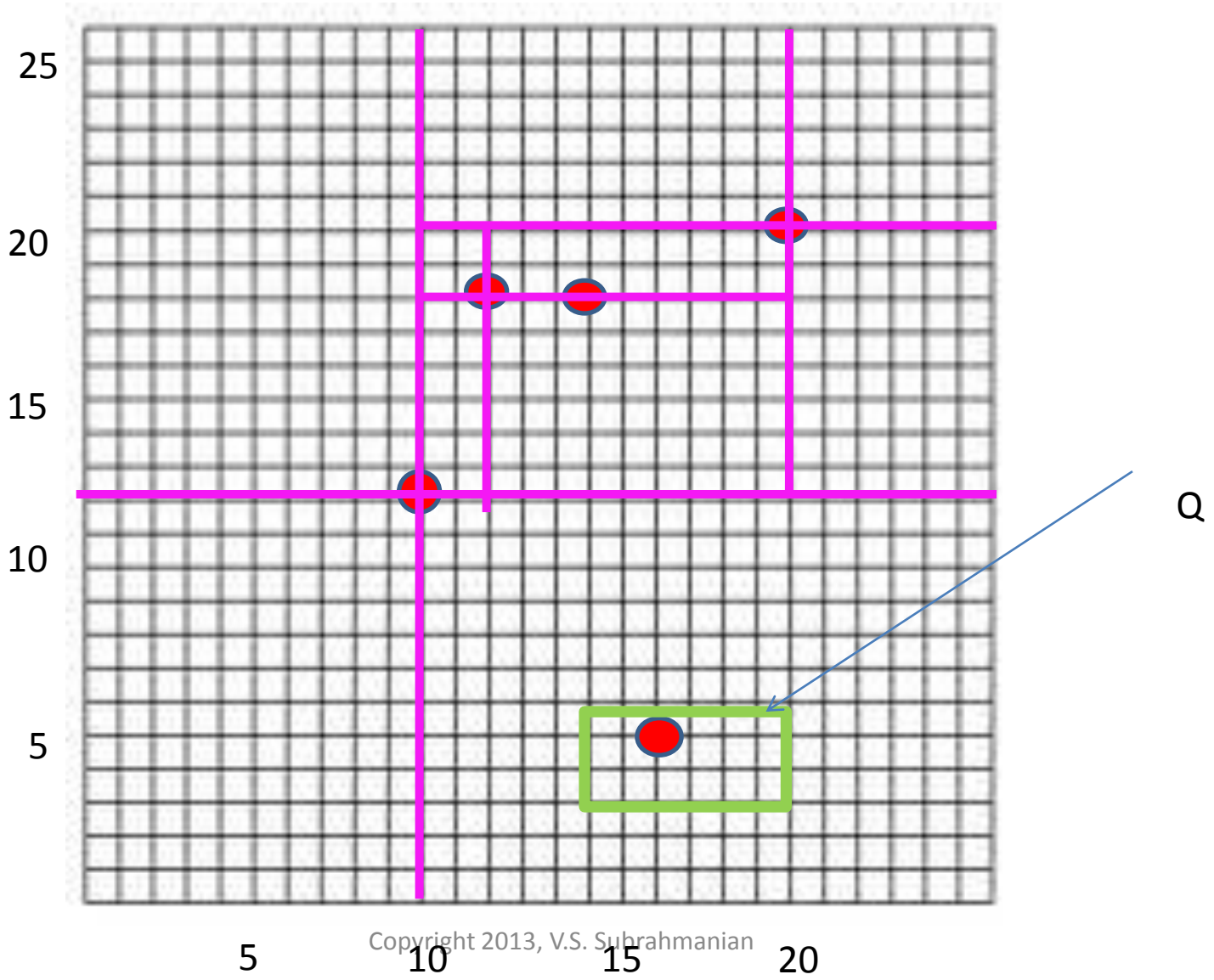


Processing query Q

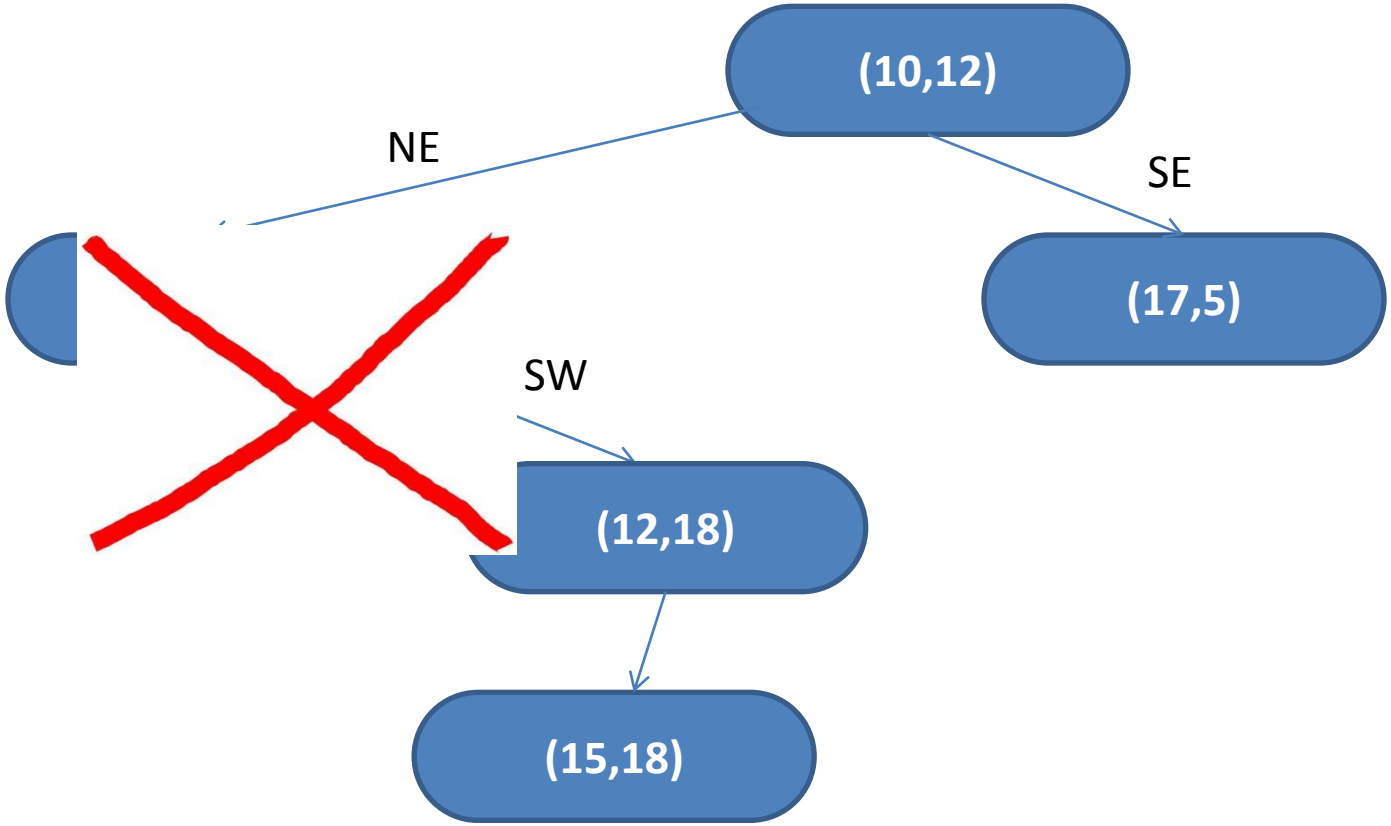


Does Region(Root) intersect Q? Yes.
Is (10,12) in Q? No.
Explore children.

Point Quadtree Example Range Query

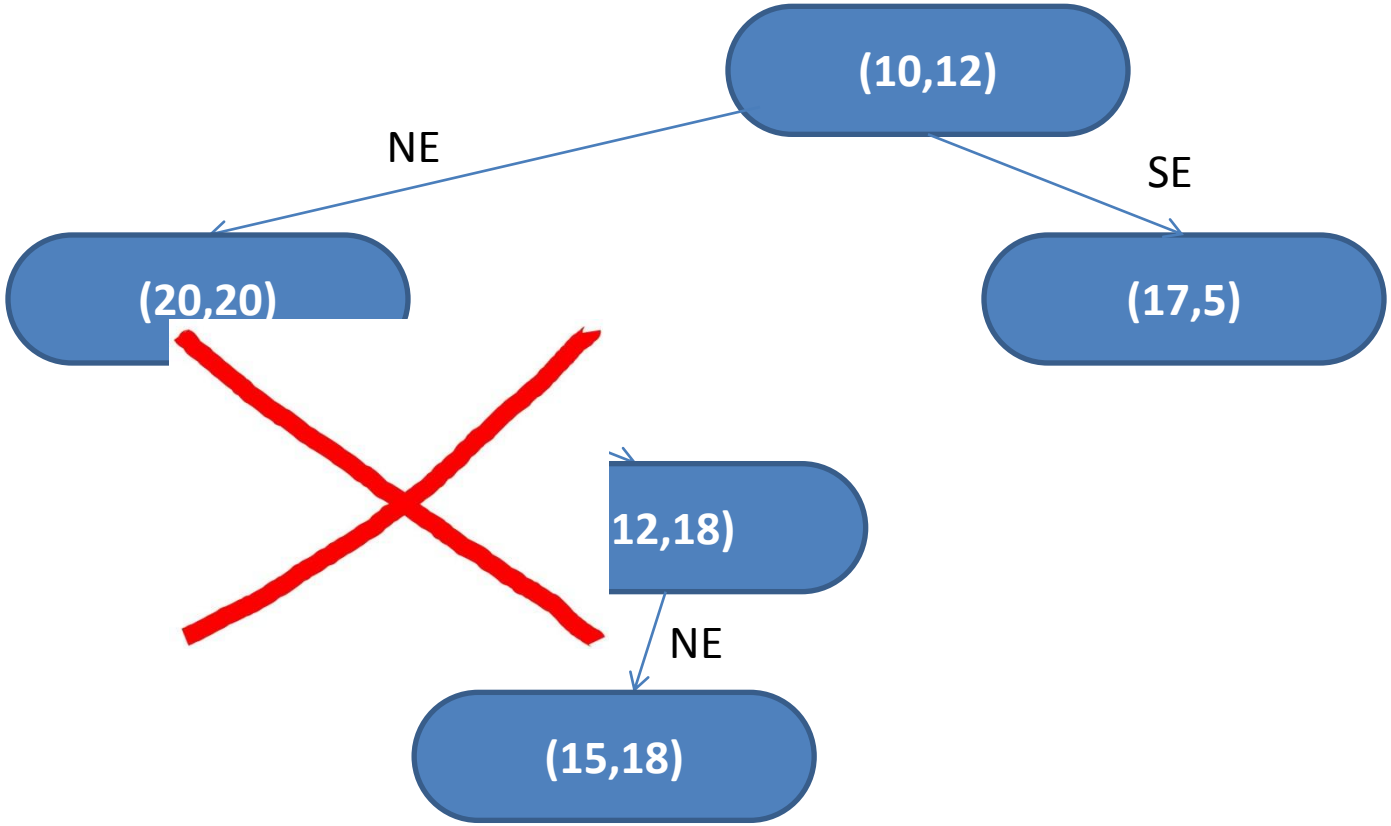


Processing query Q



Does Region(node labeled 20,20) intersect Q? NO.
Prune!

Processing query Q



Does Region(node labeled 5,15) intersect Q? Yes
Check if (5,15) is in Q. Yes.
Nothing left to search. Return Answer.

In class exercise

- Build a point quadtree from scratch.
- Execute some range queries on it.

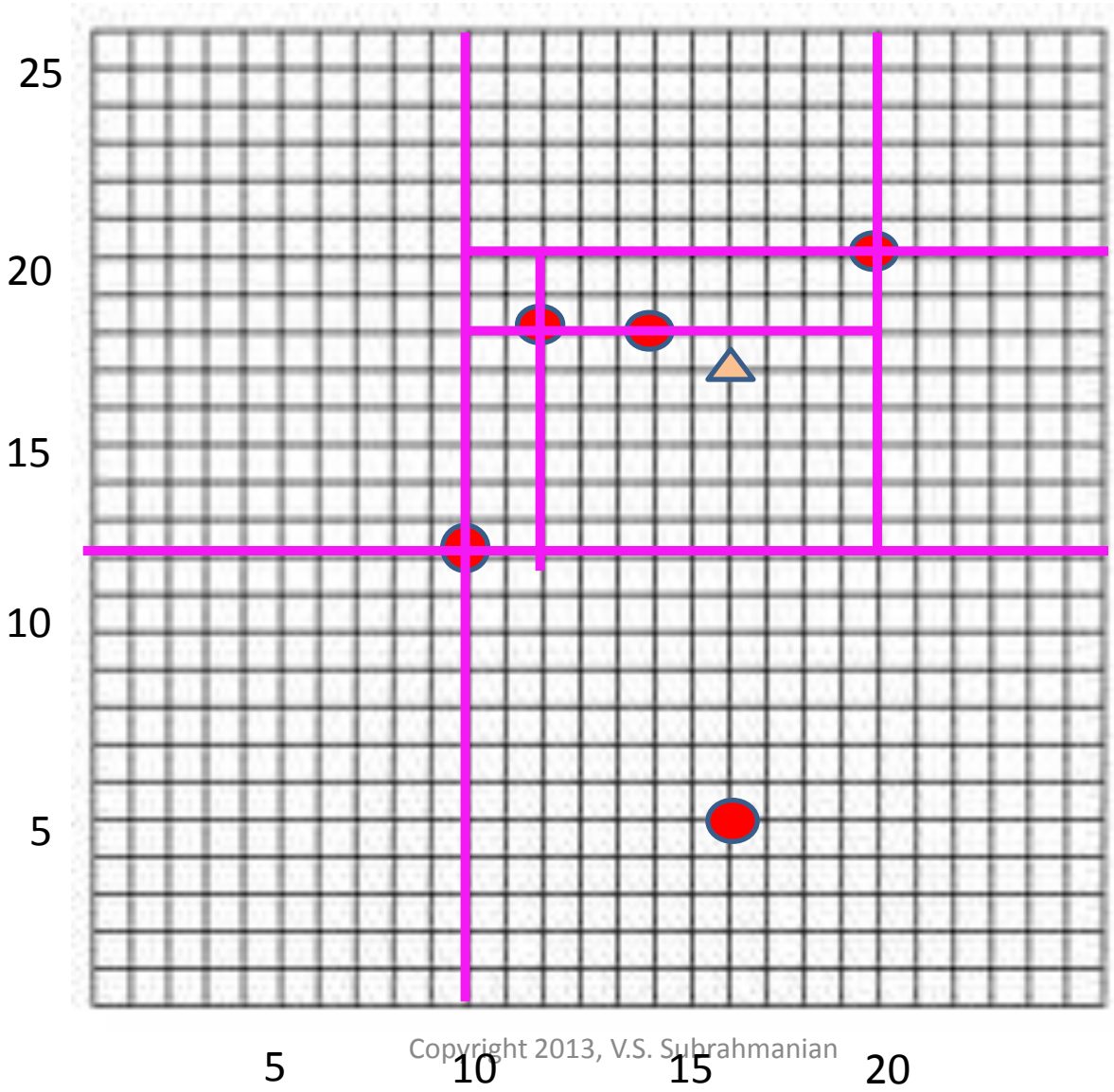
Nearest Neighbor Queries

- You are given a query point P and a pointer T to the root of a point quadtree.
- You must find a nearest neighbor of P in T, i.e. a point P' in T that is as close as possible to P.
- Note that P may not be in the tree.
- Suppose P=(x,y) and P'=(x',y'). Then the distance between P,P' is given by $\text{sqrt}((x-x')^2 + (y-y')^2)$.

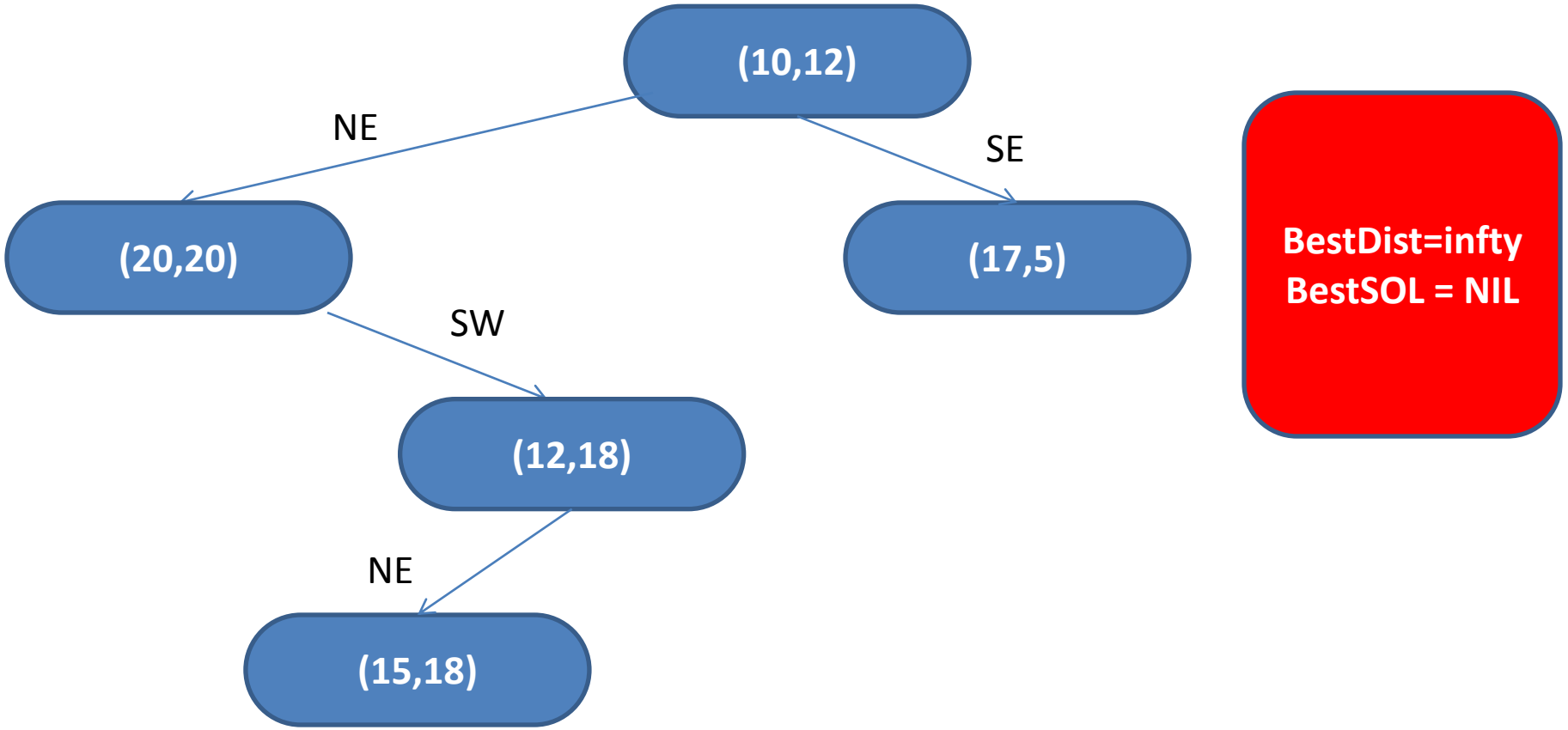
Nearest Neighbor Query Sketch

- The approach is similar to that for BSTs and B-Trees.
- Initialize BestSOL=NIL, BestDist = infty.
- When visiting a node N, check if d(Reg(N),P) ≤ BestDist.
  - If not, prune. Here the distance between a region and point is the distance between P and the closest point in the region.
  - If yes,
    - check if N.Point is closer to P then BestSOL.
      - If yes, update BestSOL and BestDist
    - visit all 4 children of N.

NN Example: P = (16,17)

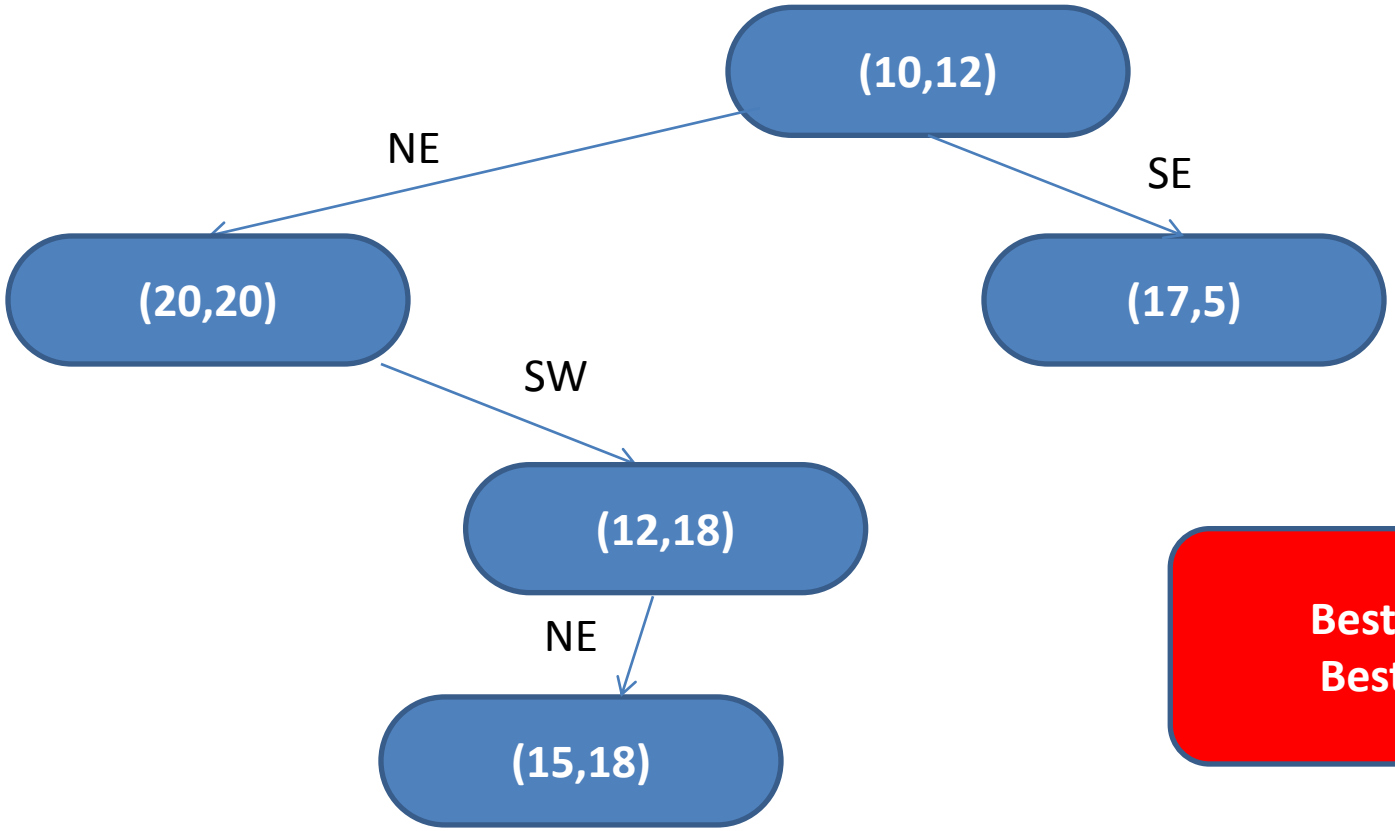


Example NN Query: P= (16,17)



Is d(Reg(Root),P) < infty? Yes.
Check if (10,12) is a better neighbor? Yes.

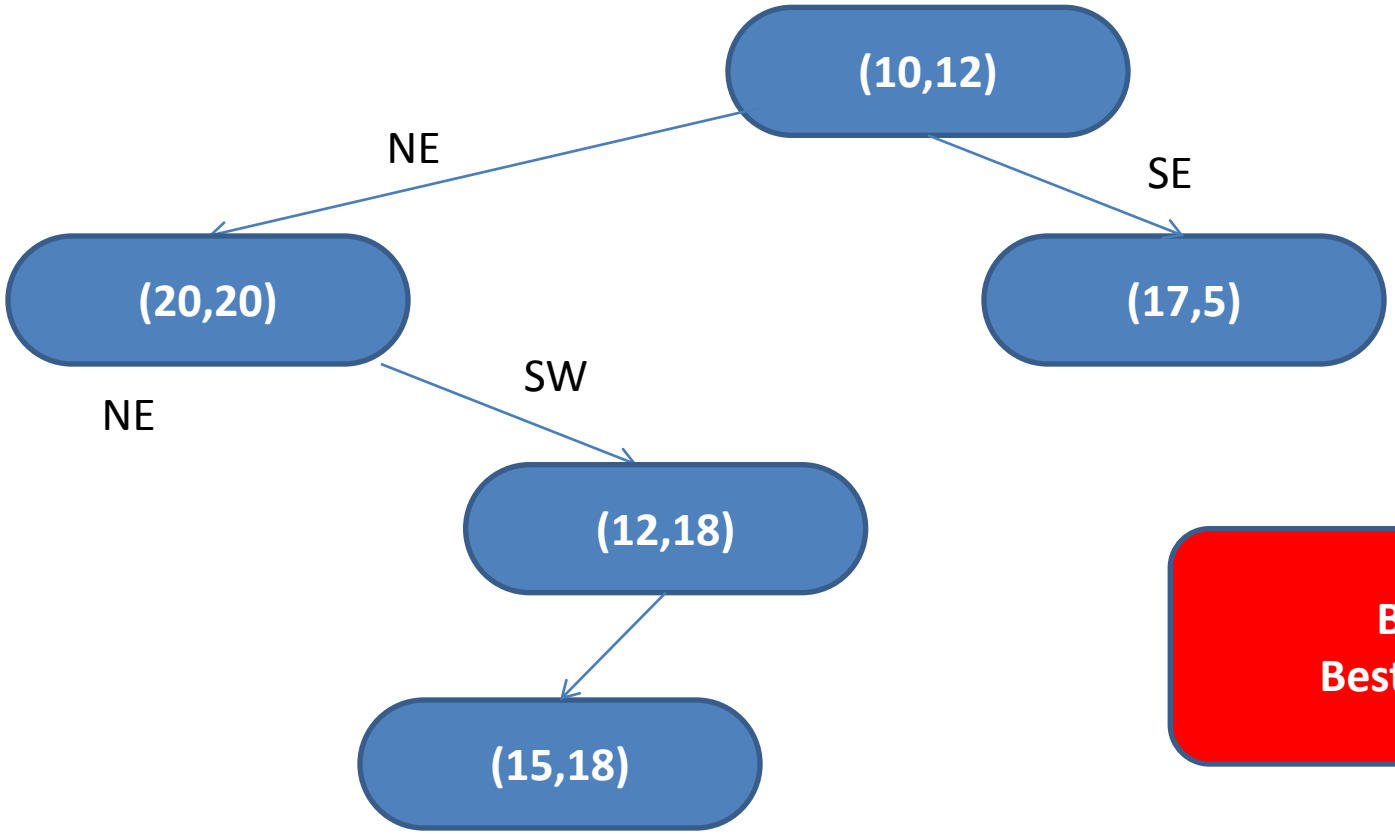
Example NN Query: P= (16,17)



BestDist = sqrt(61)
BestSOL = (10,12)

Update BestDist and BestSOL. No pruning possible, so must consider both children.
Is d(Reg(node marked 20,20),P) < sqrt(61)? Yes, its 0.
So check if (20,20) is a better solution.

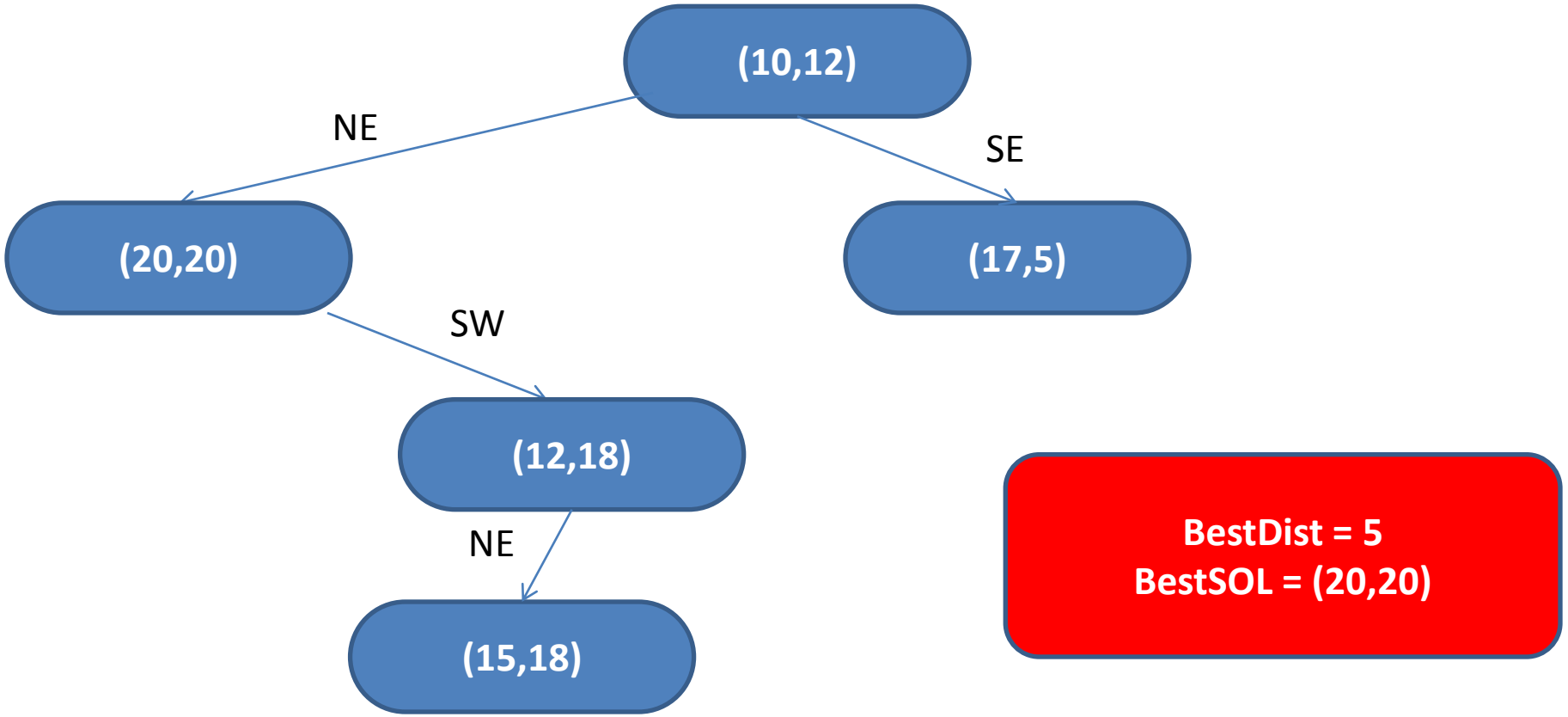
Example NN Query: P= (16,17)



BestDist = 5
BestSOL = (20,20)

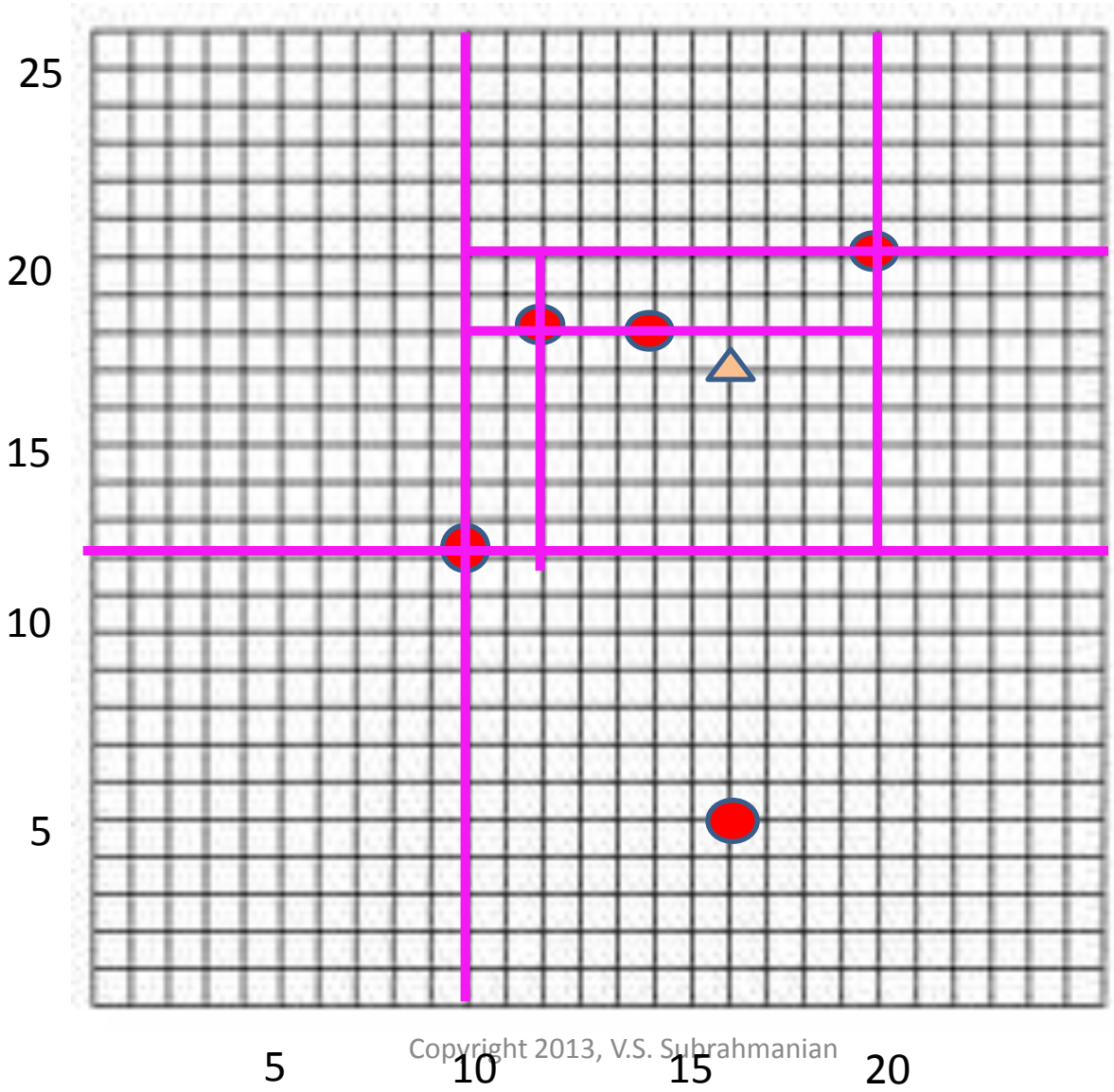
Yes, (20,20) is a better solution. So update BestSOL and BestDist. No pruning possible.

Example NN Query: P= (16,17)

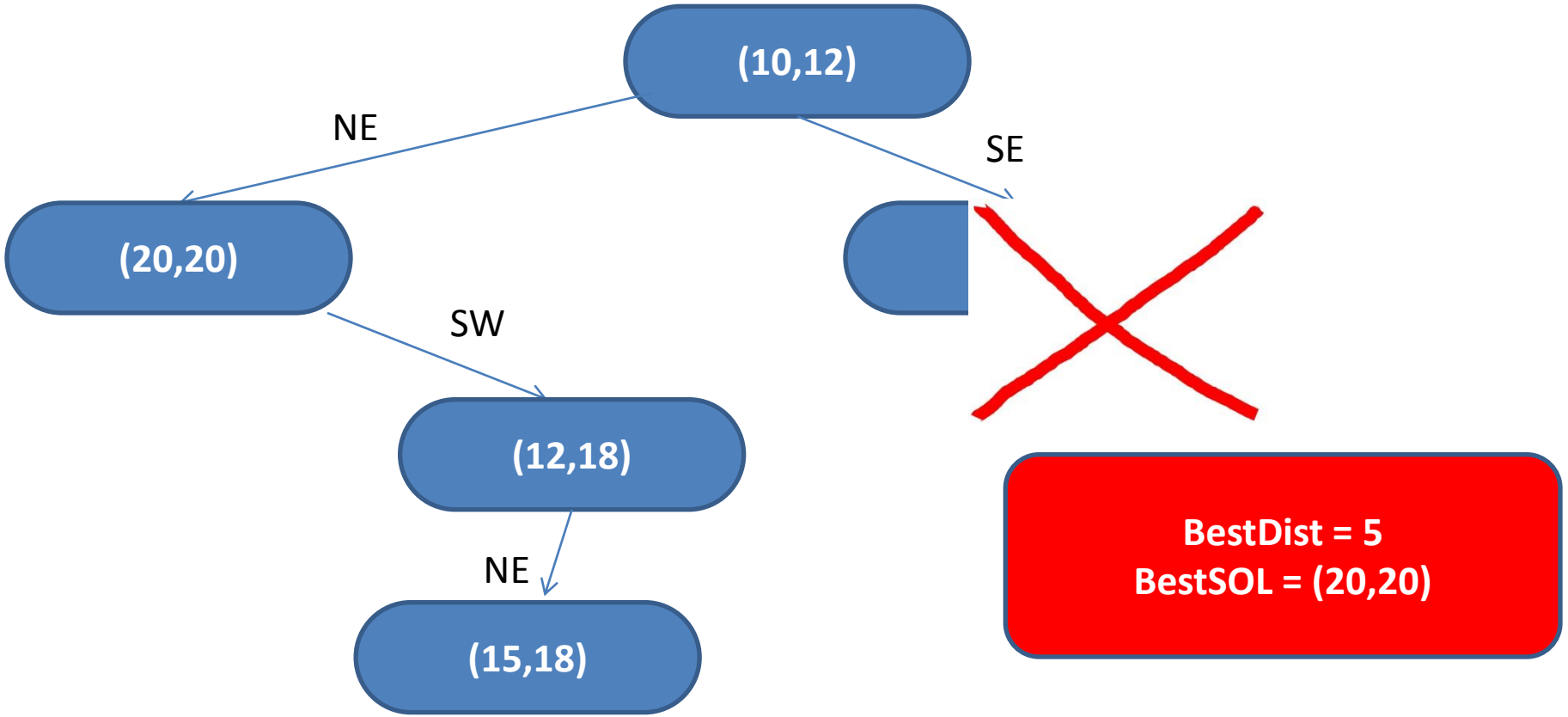


Let us next consider (17,5) – see next slide.
The distance between the region associated with (17,5) and P is more than 5 because regions are not “up-closed” by our convention. Can prune.

NN Example: P = (16,17)

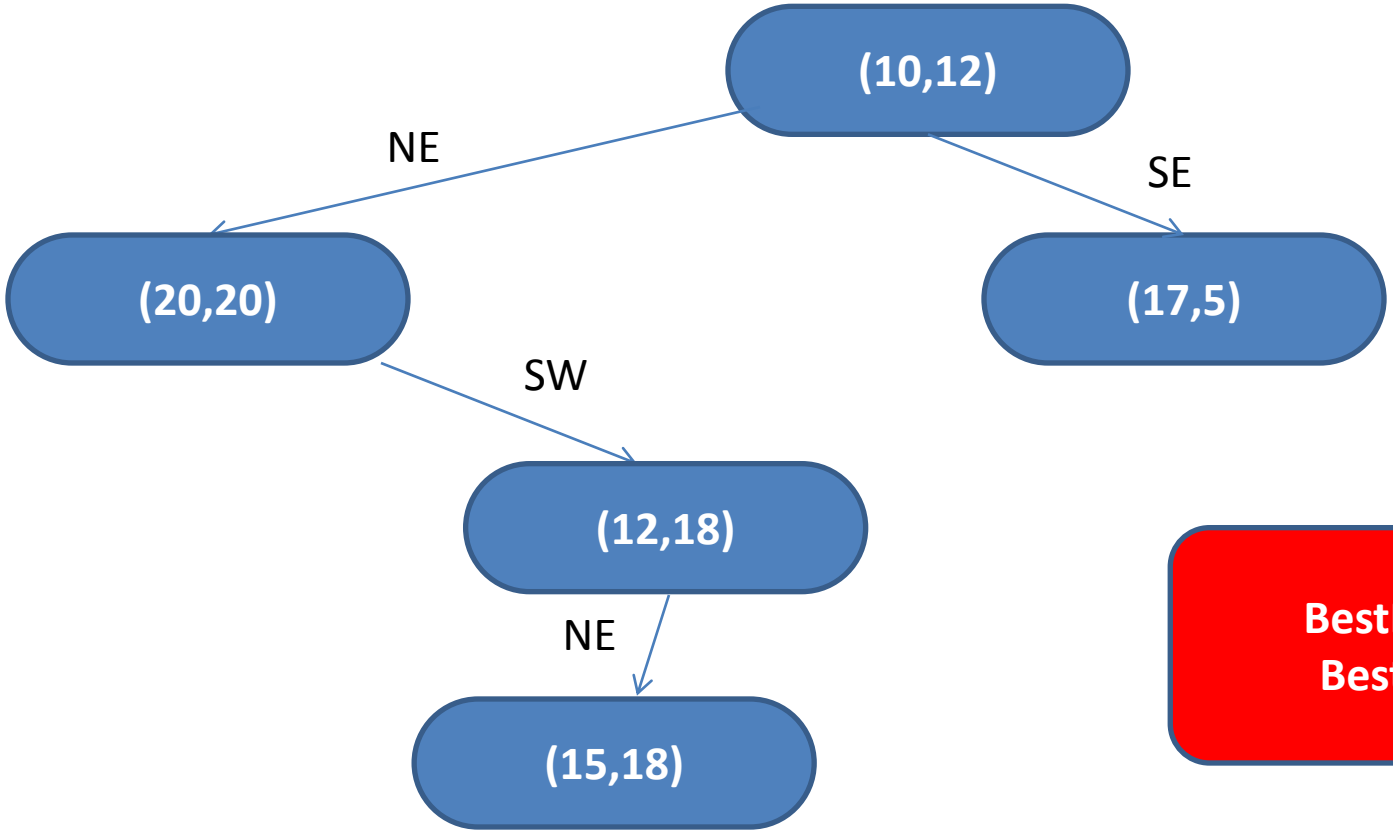


Example NN Query: P= (16,17)



Let us next consider (12,18).
The distance between the region associated with (12,18) and P is 0.
Check if (12,18) is a better neighbor.

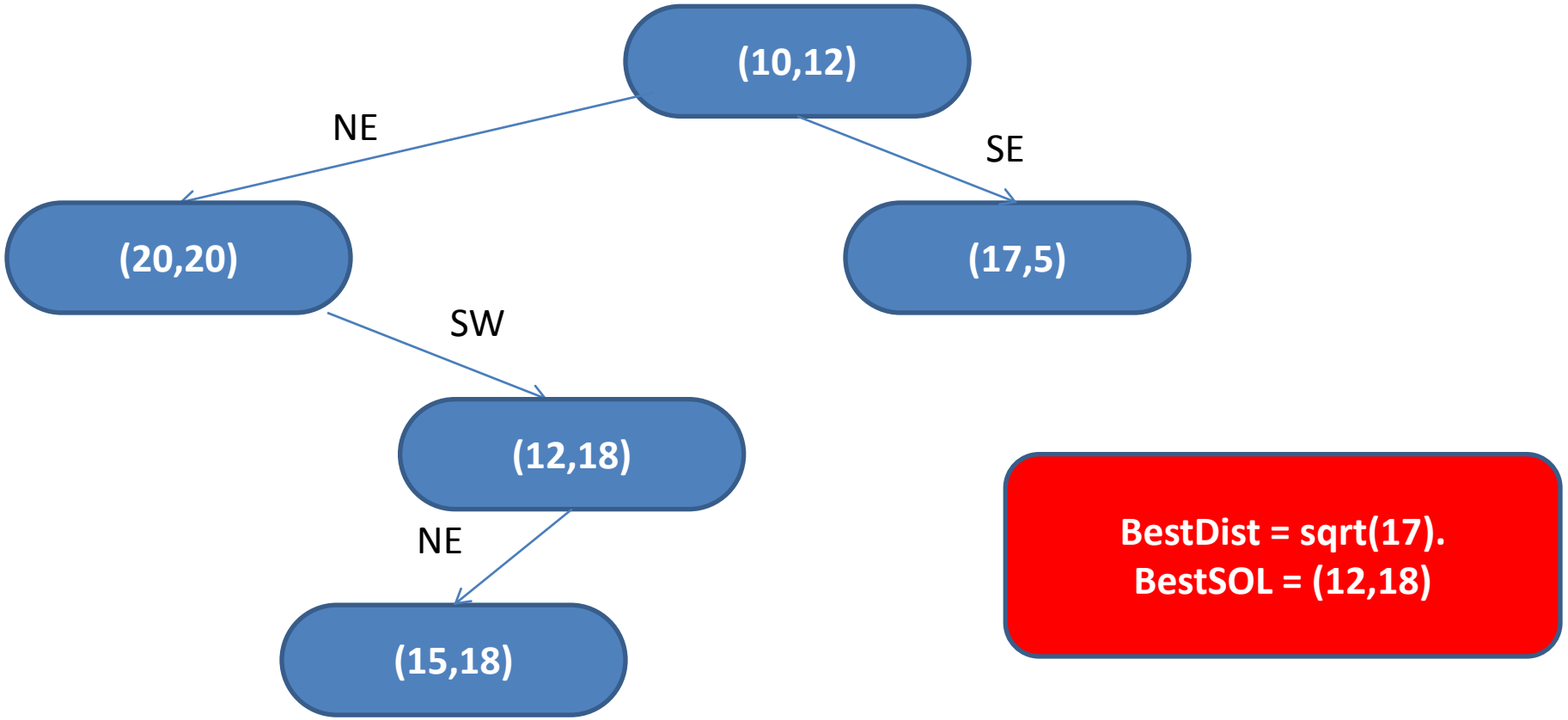
Example NN Query: P= (16,17)



BestDist = sqrt(17).
BestSOL = (12,18)

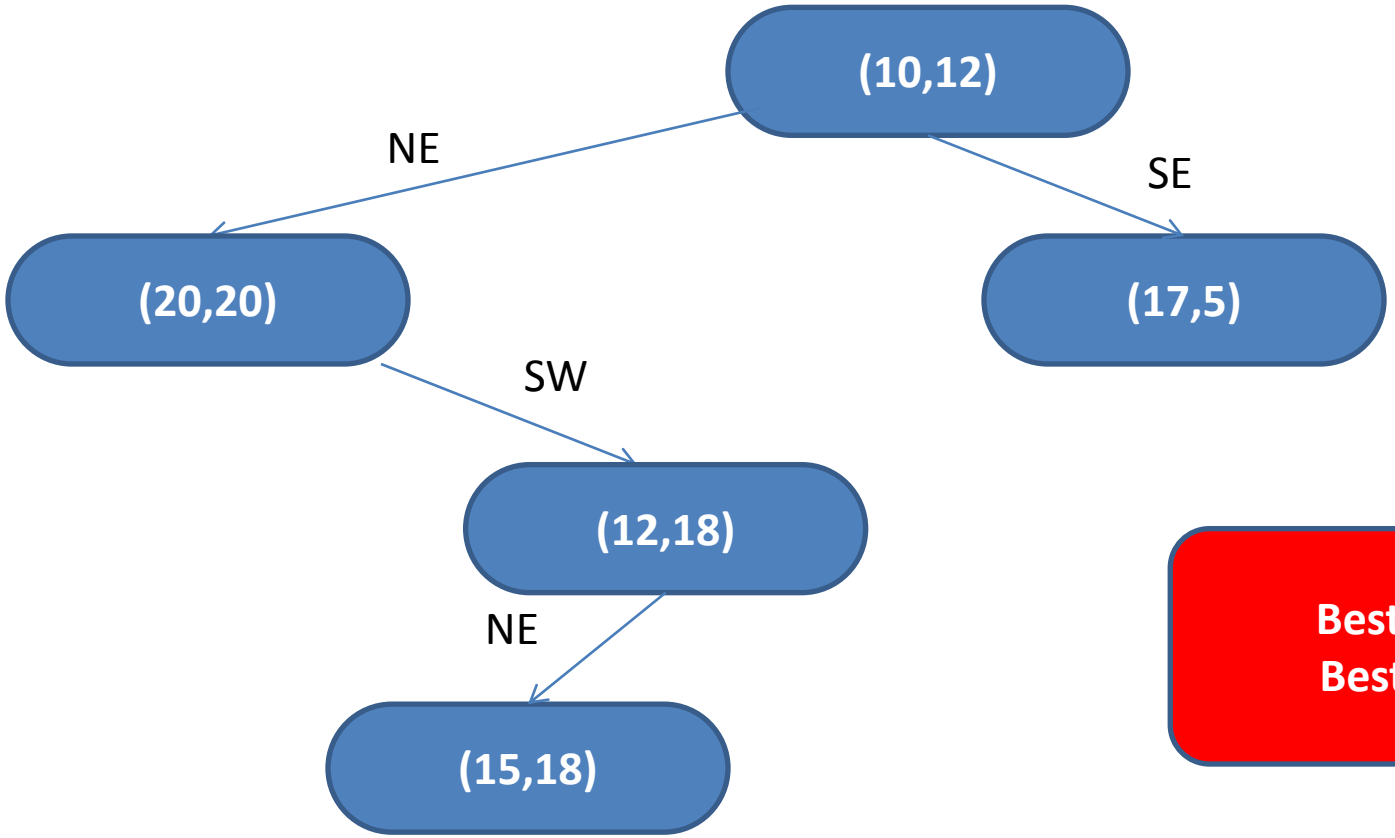
Yes it is (distance of sqrt(17)).

Example NN Query: P= (16,17)



Consider node marked (15,18). Cannot prune as distance to its associated region is less than sqrt(17). Check if (15,18) is a better neighbor. Yes.

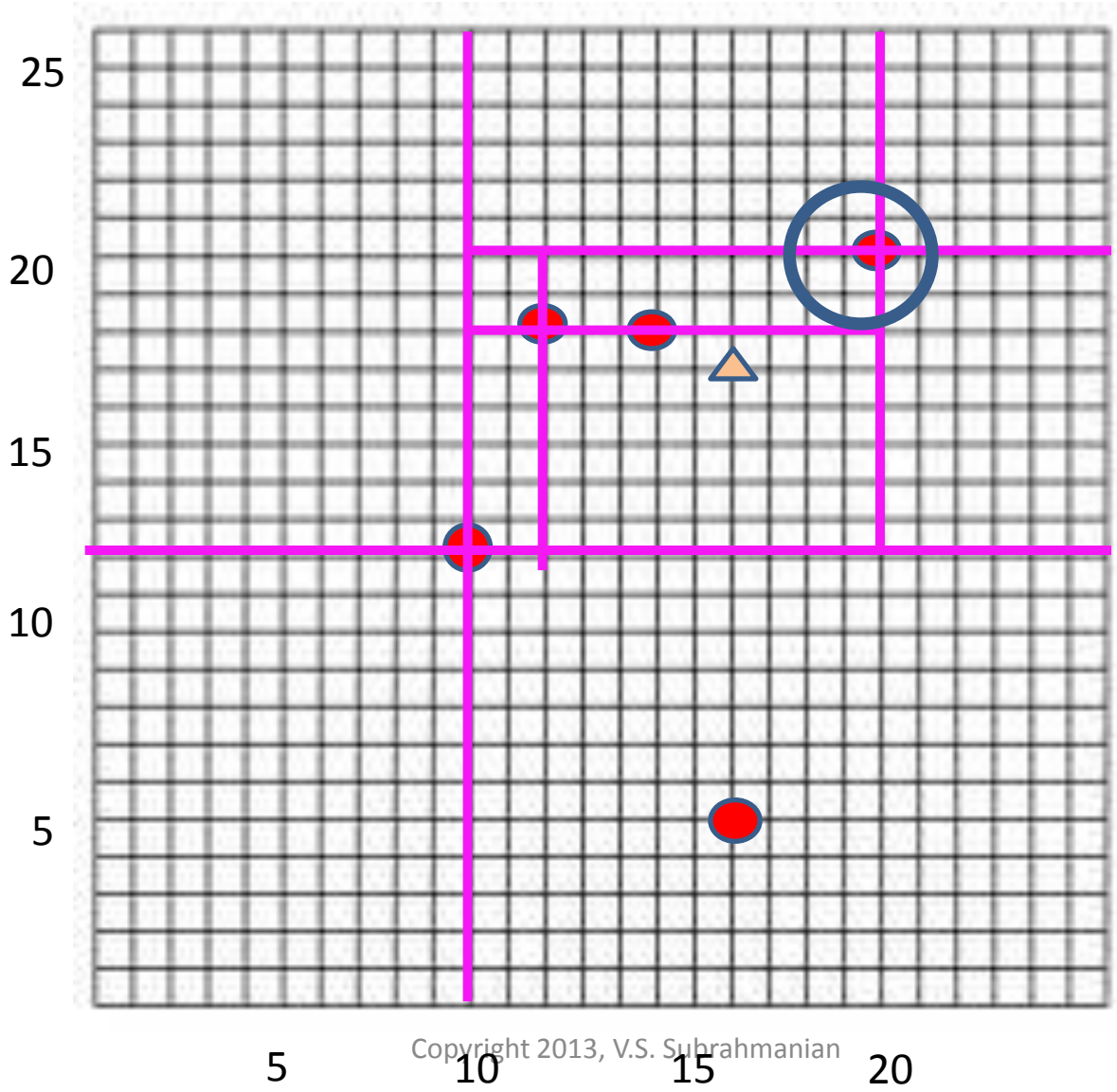
Example NN Query: P= (16,17)



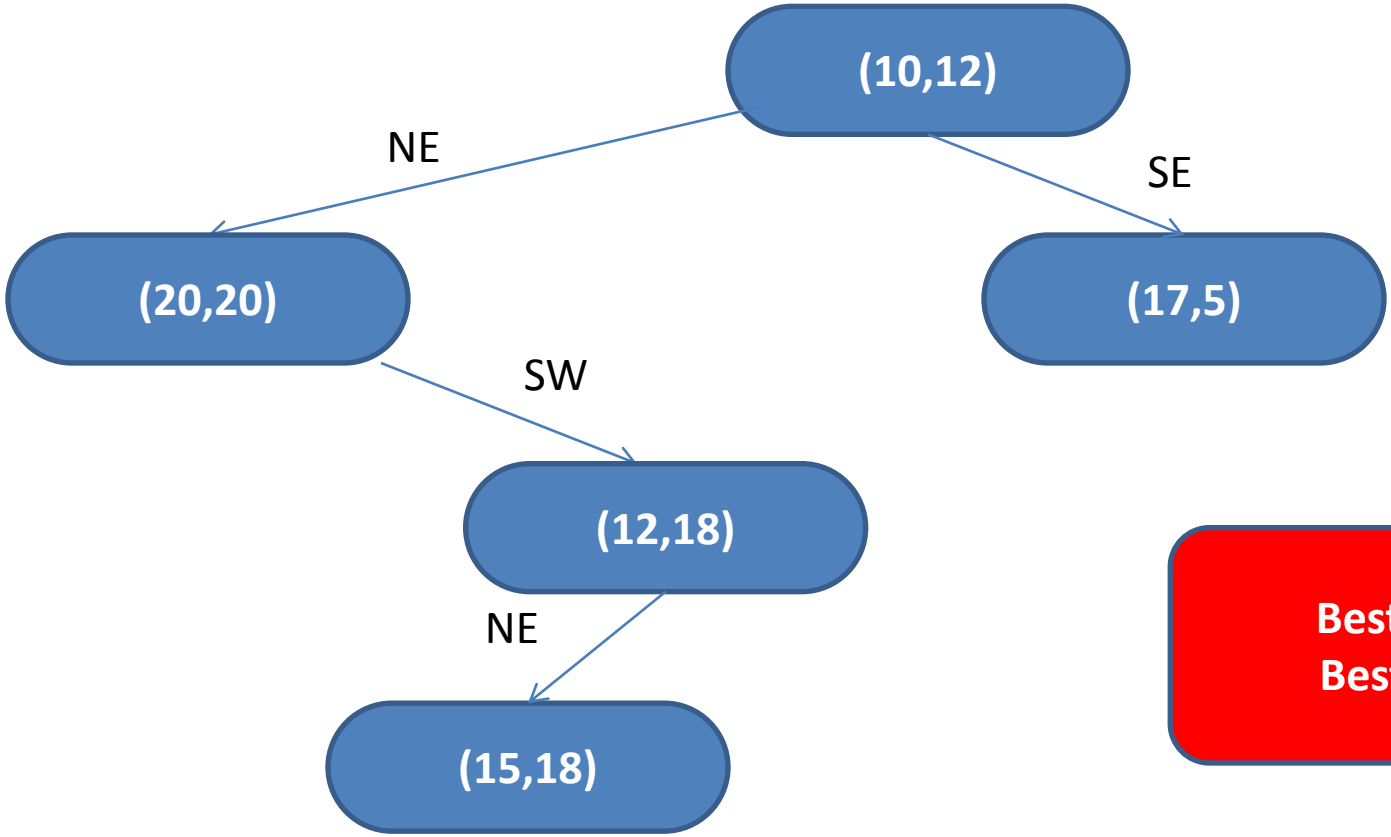
BestDist = sqrt(2).
BestSOL = (15,18)

Distance between P=(16,17) and (15,18) is sqrt(2). So update BestDist and BestSol. Done!

Discussion: What to do about Range Queries with Circles



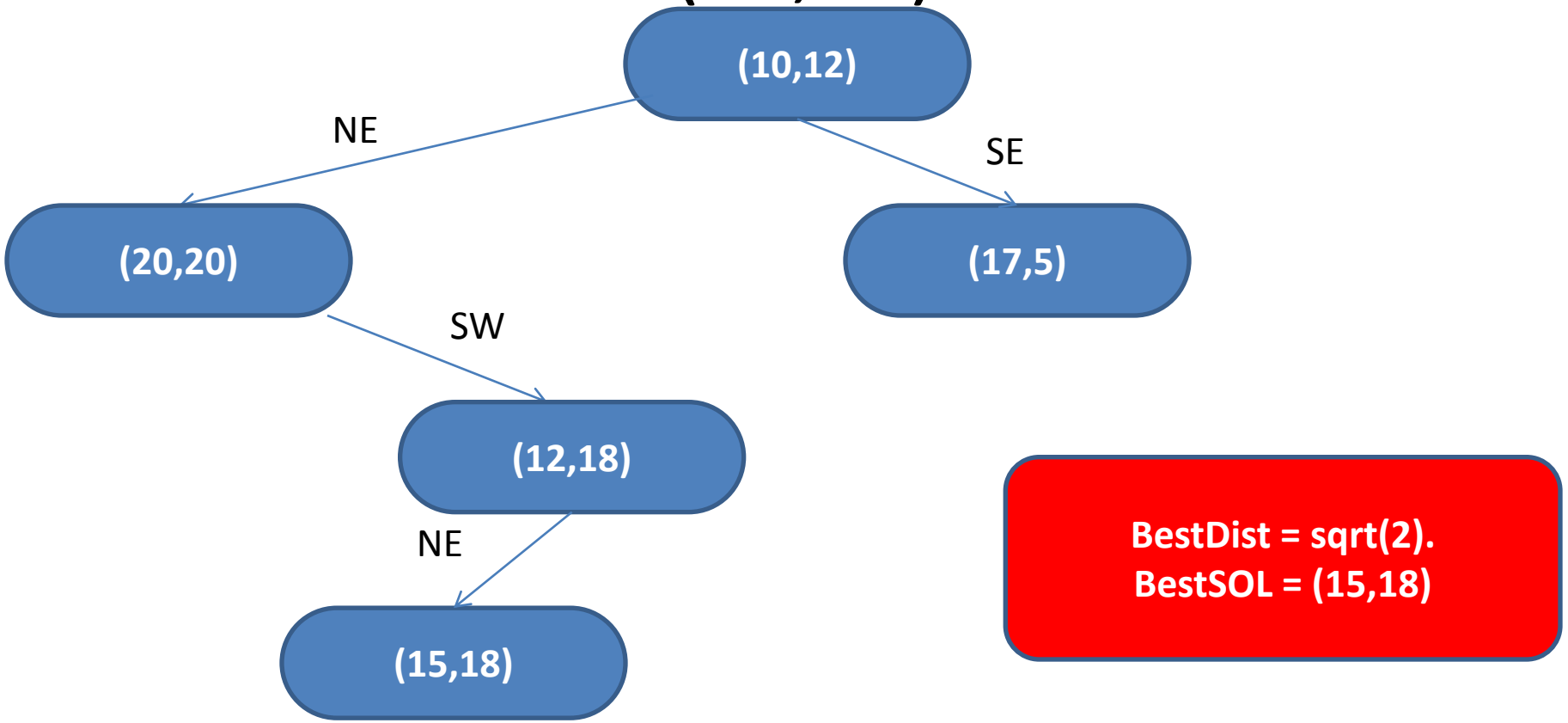
Range Queries with Circles



BestDist = sqrt(2).
BestSOL = (15,18)

Distance between P=(16,17) and (15,18) is sqrt(2). So update BestDist and BestSol. Done!

Deletion in Point Quadtrees. Delete (20,20)

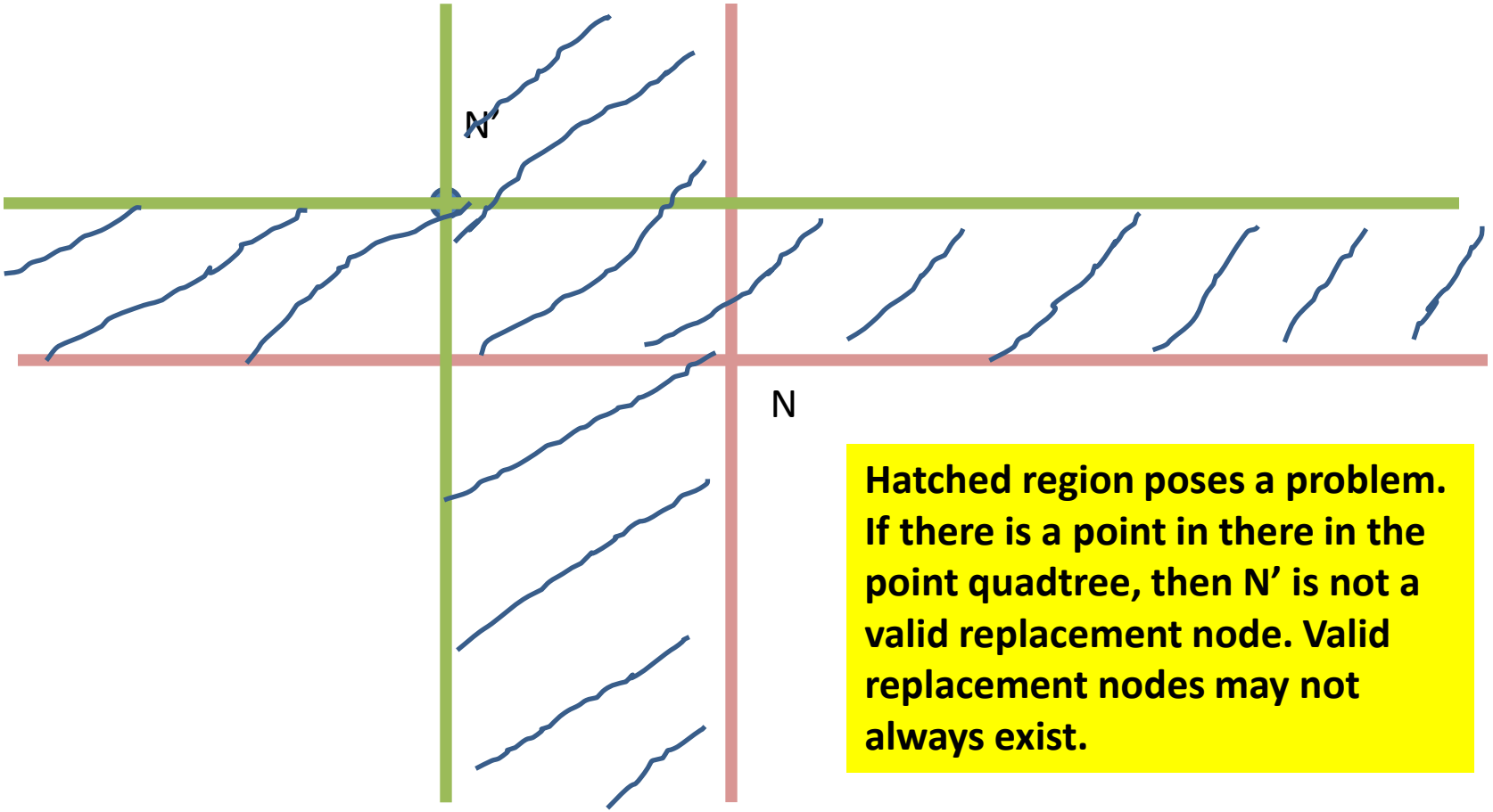


Distance between P=(16,17) and (15,18) is sqrt(2). So update BestDist and BestSol. Done!

Deletion in Point Quadtrees

- When deleting a node N in a point quadtree, we try to
  - Replace N with a descendant N' such that
    - all nodes in N.NW are to the NW of N' (except N')
    - All nodes in N.SW are to the SW of N' (except N')
    - And similarly for N.NE and N.SE.
  - Recursively delete N'.
- Will this always work.

Let's See – Suppose N' is in the NW quadrant w.r.t. N.



Hatched region poses a problem. If there is a point in there in the point quadtree, then N' is not a valid replacement node. Valid replacement nodes may not always exist.

Deletion in Point Quadtrees

- The main problem with a point quadtree is that we might have to re-insert a bunch of keys in the worst case.
- Thus, point quadtrees are not appropriate for applications where a lot of deletion is involved.