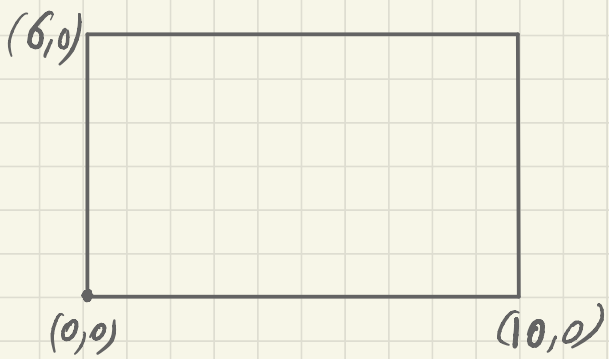
Assignment 2

Hussein Houchaye (ID: 101 232 199)

Question 1.

Let B be the bucket size. (B = 3).

Suppose we have the following Domain

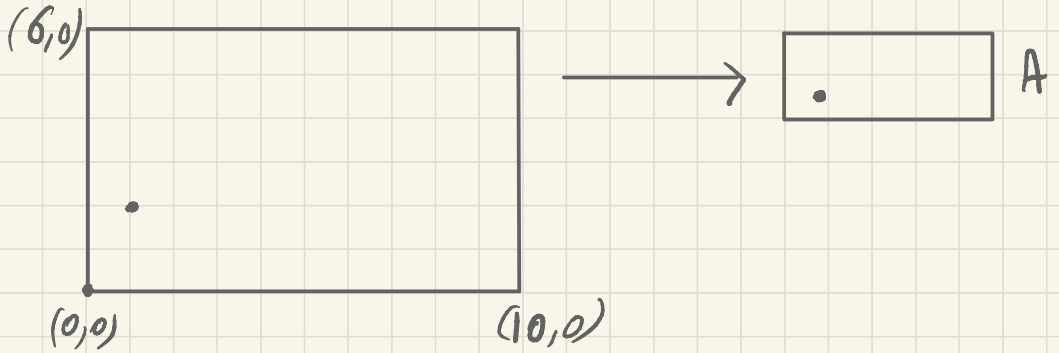


we want to insert the following point set

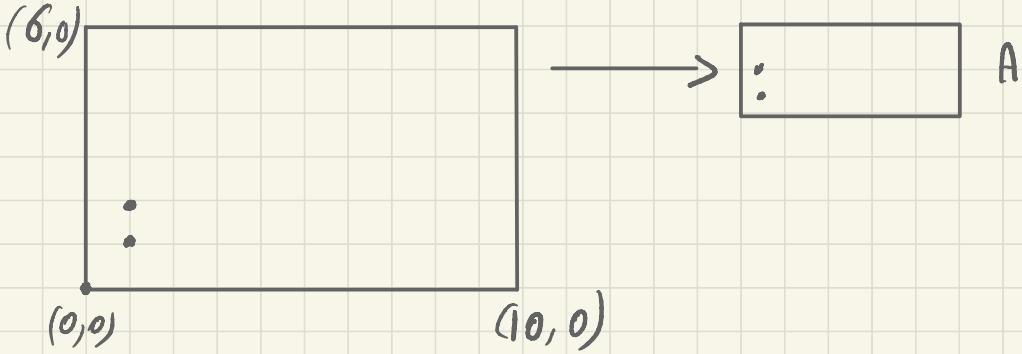
$$P = \{(1,2), (1,1), (3,2), (1,4), (5,2), (8,3), (9,5)\}.$$

Inserting $p_0 = (1, 2)$

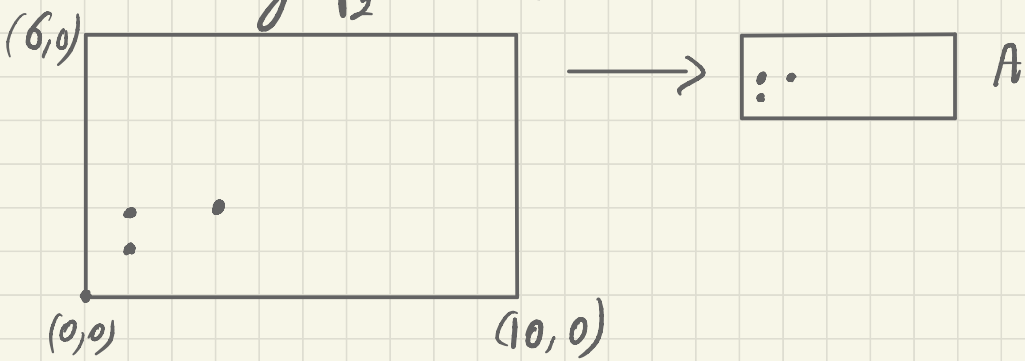
each bucket corresponds to a block



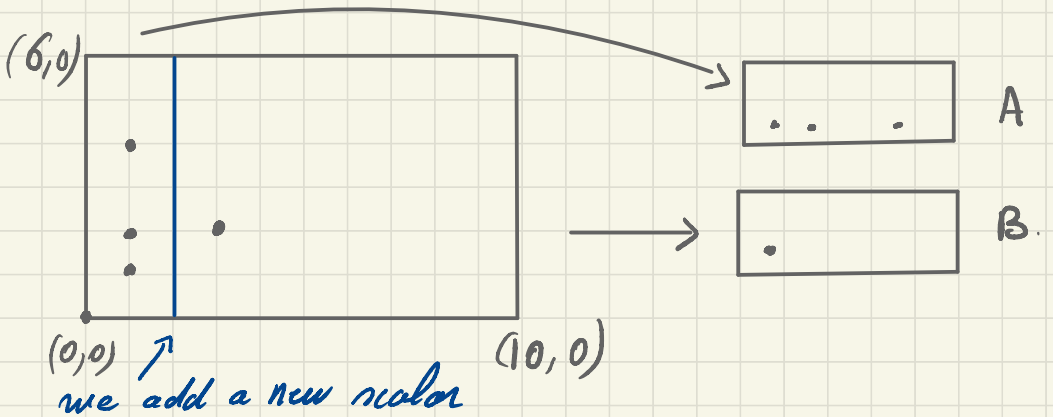
Inserting $p_1 = (1, 1)$



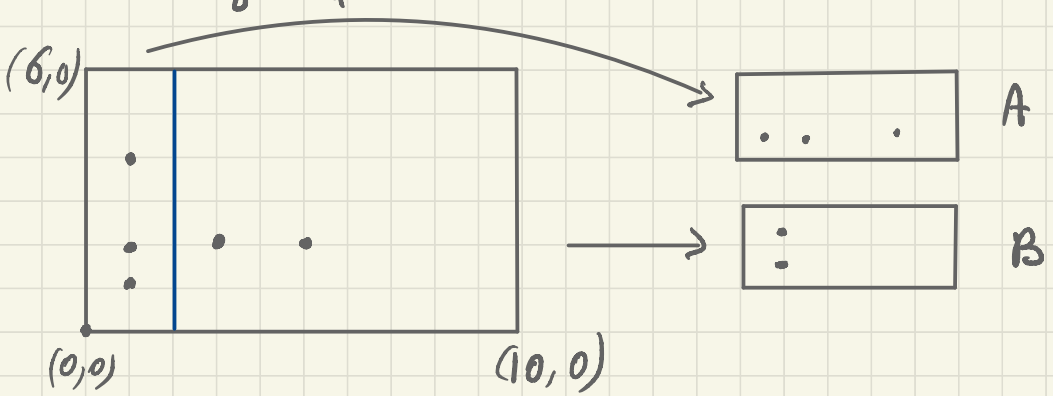
Inserting $p_2 = (3, 2)$



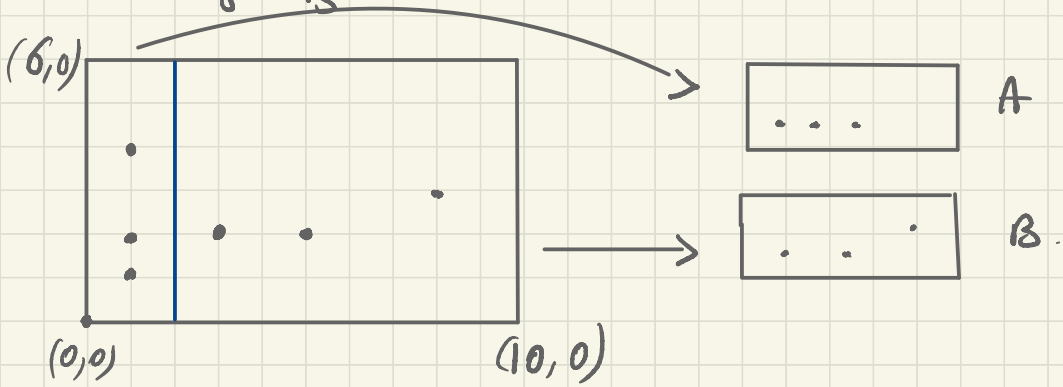
Now, we insert $P_3 = (1, 4)$. Since the bucket size is 3 and we have 4 points by now. We must split the domain and the buckets.



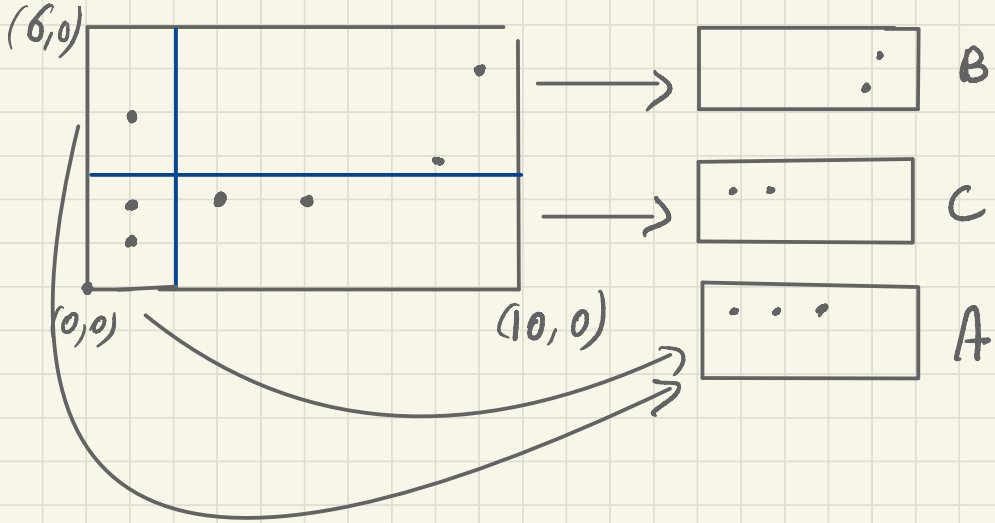
- Inserting $P_4 = (5, 2)$



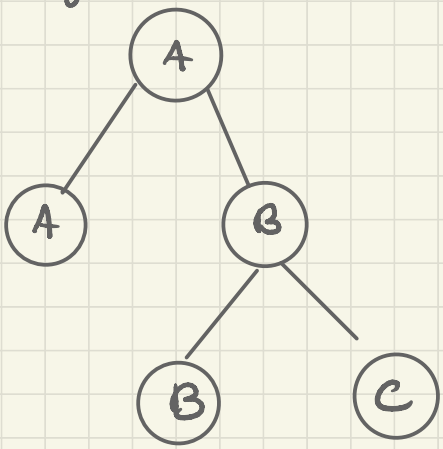
- Inserting $p_5 = (8, 3)$.



- Now, we insert the point $p_6 = (9, 5)$ that will cause an overflow in bucket B. Therefore, we perform a split operation.



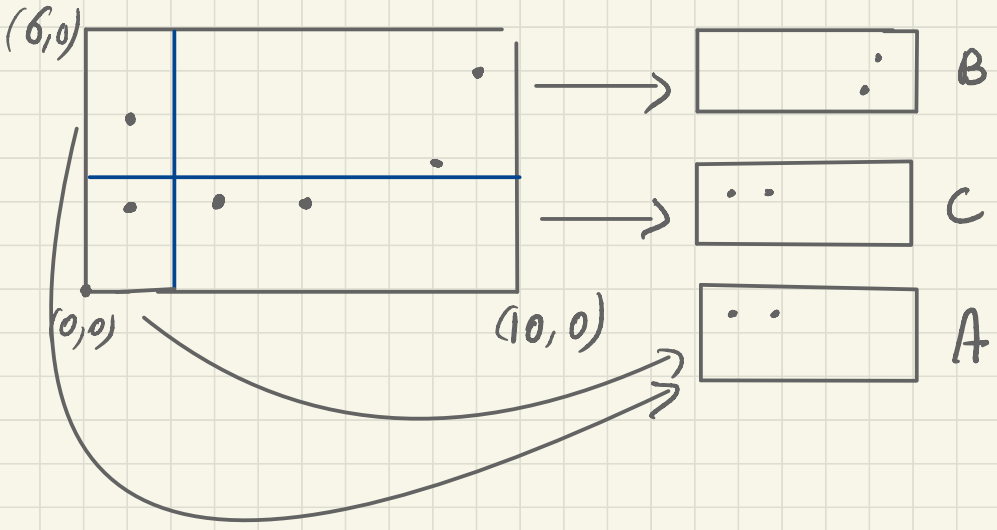
The split history tree.



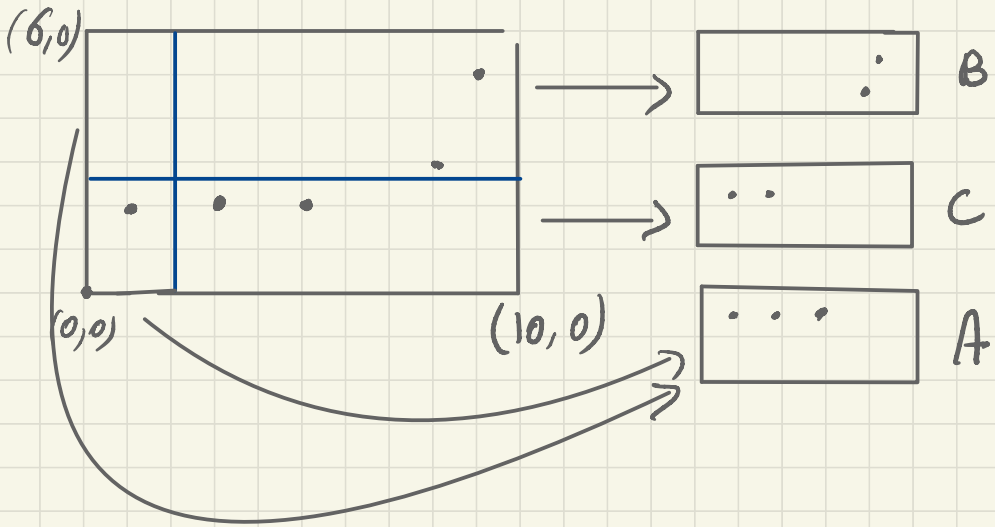
- Now, consider the following set of points that we wants to delete.

$$D = \{(1,1), (1,4), (9,5)\} \subset P$$

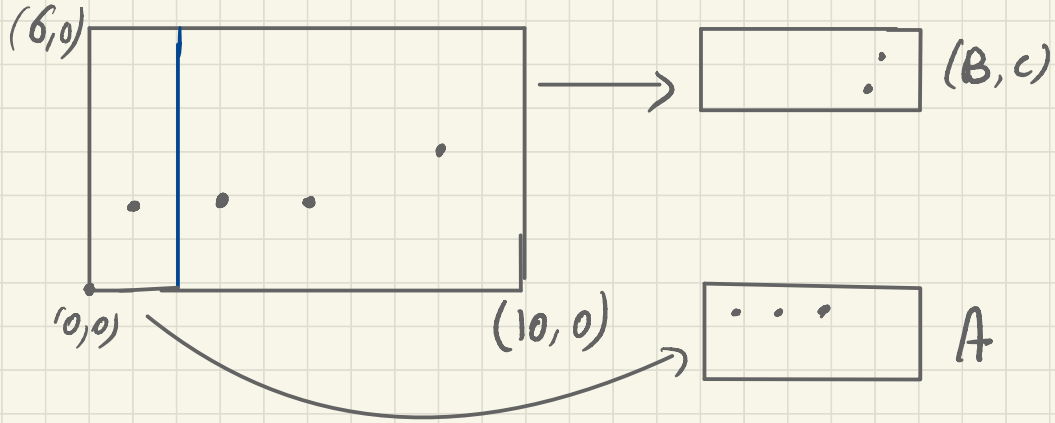
- Deleting $p_1 = (1,1)$



- Deleting $P_3 = (1, 4)$



- Deleting $P_6 = (9, 5)$, Now we can merge (B, C)



We merged B and C because they are Twin. we could also delete another point and merge the last 2 cells.

Question 2.

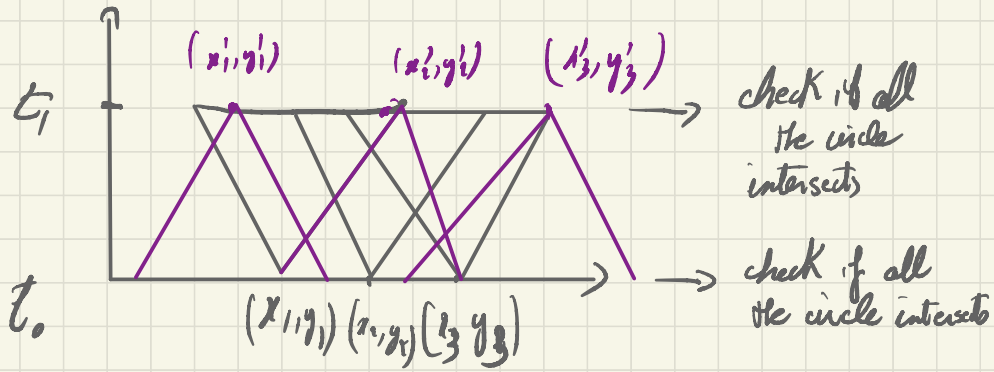
- Determine if a meeting among all n participants is possible at all.

  - For such a meeting to happen the following must be satisfied:

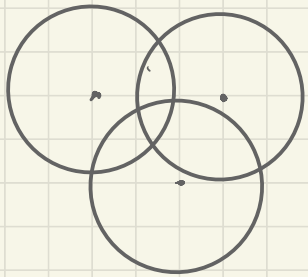
  (1) . all the n upper cones must intersect.

  (2) . all the n lower cones must intersects.

  (3) . the intersection of the intersection of the n upper cones with the intersections of the n lower cones must not be empty



- So, our problem now, given n circles in $\mathbb{R}^2$
how can we check if all intersects or not

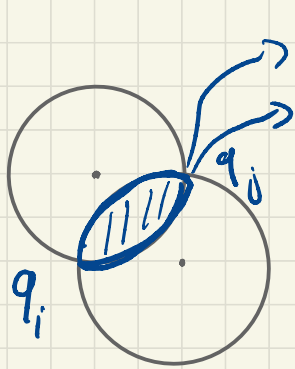


- pick two circles $C_i = (p_i, r_i)$ & $C_j = (p_j, r_j)$
where p_k is the center & r_k is the radius, $k \in \{i, j\}$.

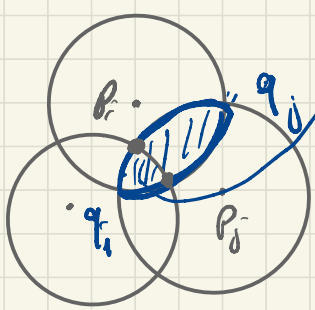
C_i intersects C_j

$$\iff d(p_i, p_j) \leq r_i + r_j.$$

- So, the algorithm will pick any two circles
& check if they intersect, if they don't
it can return "no possible meeting"
- otherwise, it will compute the intersection points
+ the curves that connect them.
(there are 2 intersection points).



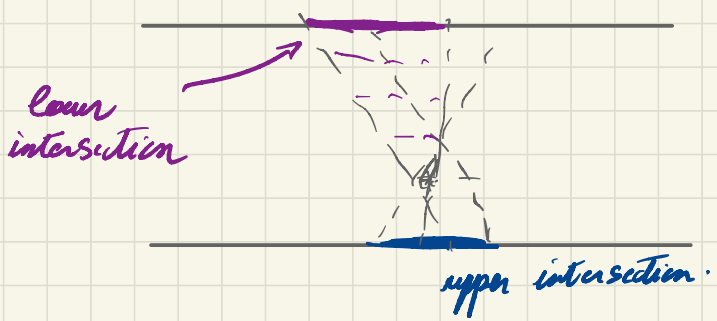
- Now the algorithm will pick another circle, say $C_e = (p_e, r_e)$. & test if it intersects with the region R.



- Then the algorithm continue in the same way.
- the time complexity is the number of circles n times the time to check if a circle intersects a convex region.
- Thus, it is $n \times 1$ number of the arcs that make the convex region.

$= O(n^2)$ in the worst case.

- Suppose now, we have the intersection of all lower & upper cones.
- Since the intersection of n convex sets is convex & a convex set is a connected set. all the upper (or lower) intersects in one set.



- now we trace back the edges of the region in time to check if the intersection of the intersection of the cones is not empty.
- the time complexity should be quadratic if we want test one side the region with all the other sides!

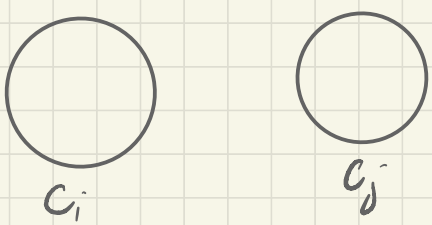
- Determining if a meeting is possible with $n-1$ participants.

- It is basically similar algorithm as before. But we might need to keep track for more than one intersection region.

  - In the first step, pick 2 circles.

    $C_i(p_i, r_i)$ & $C_j(p_j, r_j)$.

    - if they do not intersects.

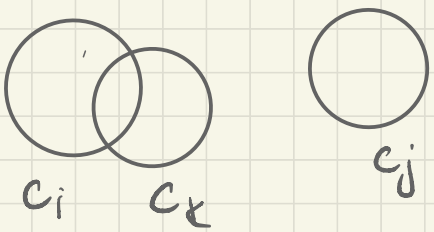


pick another circle C_k.

if C_k is disjoint from the previous two circles we can immediately say no meeting is possible

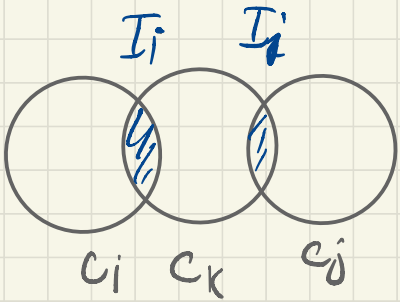
- otherwise C_x either intersects C_i or C_j or $C_i \cup C_j$

Case 1. C_x intersects C_i or C_j



we disregard C_j and use the previous algorithm on the remaining circles. (i.e all the circles except C_j).

Case 2:

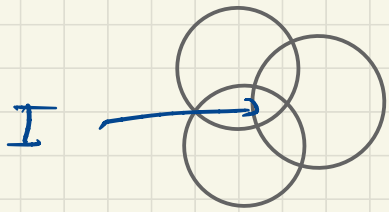


we keep adding circle C_e & if C_e does not intersect I_i or I_j then the answer is negative

otherwise if it intersects I_i (WLOG), we disregard C_j & continue using

the previous algorithm.

- Lastly, if the tree circles intersect,



we continue as befor with having only one opritunity of picking a circle that does not intersect I.

in other words, we have one chance to pick one non intersecting circle & continue. However, if we pick another one that is not intersecting. Then obviously no n-1 circles intersects.

- the time complexity is similar to the previous algorithm. $O(n^2)$. where n is the number of circles.

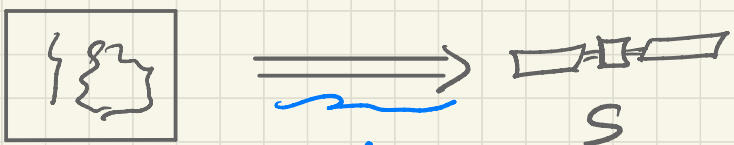
- Determing if a meeting is possible that is held as follows: For one hour everyone meets, then, they all travel together to a second location and continue the meeting for another hour.

- Increase by 1 hour the departure time of all the participants from their previous location, & Decrease the arrival time to their next location by one hour.
- So, now we can think that the duration of their meeting is zero hour.
- Therefore, the problem now is to check if they can all travel from one location to the other (all at the same time)
- This problem is equivalent to check that the two locations are in the same intersection area.
- Thus, the time complexity is linear in the size of the boundary of the intersection volume i.e $O(n)$.

– i.e the complexity of checking if a point lies in a convex 3 dimensional polyhedron.

Question 3:

Suppose we have a map M stored a a Mesh. (Triangular Mesh) and we want to transfom it to a satalitte S.



we have constraint on the size of the object that we can translate.

- However, we can do some computation over M. and send it to S where S can also do some computation

- We take the Map M and we can remove the area with little interest.

for example: we do not need a lot of Triangles to represent seas & deserts as they look homogeneous and they have very simple structure.

- The second step is to use the progressive Meshes Techniques with the selective refinement scheme (the one discussed in class).
- So, we will end up with a coarser Mesh for M, say M_0, with sequence of vertex splits that can be viewed as a forest of binary search tree.
- Finally, we send M_0, with vertex split information. then S, can execute selective refinement for each vertex in the region of interest.