Computational Geometry

Prof. Sandeep Sen

Department of Computer Science and Engineering

Indian Institute of Technology, Delhi

Module No. # 04

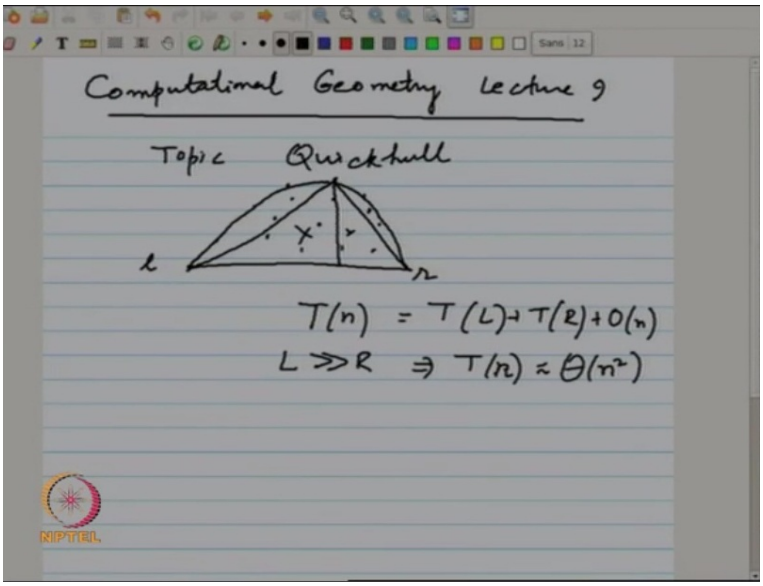
Convex Hull Different Paradigms and Quickhull

Lecture No. # 03

Quick Hull

So, welcome to lecture 9 of Computational Geometry, and we will continue with the topic that we started last time, namely quick hull. So, I gave you an example of this algorithm, which is traditionally known as quick hull, has behavior similar to quick sort; in some sense that you are dividing into two sub problems based on the points that we can discard from within the triangle, so just quick recap.

(Refer Slide Time: 01:03)



So, we are only looking at let us say the upper hull. So, you have l and r, the left most and the right most point, and the remaining points can be arbitrarily distributed, and you find the point that is further strong, this line may be in this point, look at this triangle and clearly all the points in the triangle can be eliminated; so, it is only the points that are

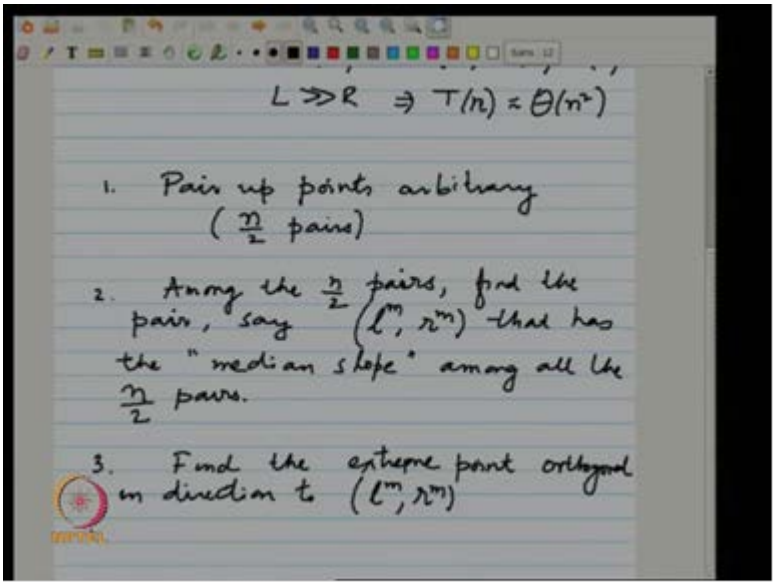
outside the triangle you need to recurs, and recursively you built the left part of the upper hull and the right part of the upper hull, and then of course, pasting them is together.

The problem with this, is that the two sub problems can be very skewed, which means that you may have a recursion or something like T's of n is equal to T's of the left; so the left sub problem per T's of right sub problem plus order n, and if l and r are not balanced, if l is much larger than r, then this could imply T's of n is about n square kind of thing. So, this is what is the problem with the fork load fix quick hull, and there is not much hope of avoiding this behavior in the way that we or at least not in the straight forward way that we do for quick sort, where you can pick the pivot at random. So, we will again do something at random, not I mean the process is not random, but we use randomization, but to give you the basic the crux of the idea, how randomization is useful, I will actually describe a deterministic algorithm.

So, the algorithm is of will have the following steps, so it may it may not be very transparent the description is a kind of some mystery, so but we will unreliable that. So, the first thing that you do is you pair up points arbitrarily, so I am not saying at random, I am saying arbitrarily, any way that you want. So, if you start with n points, which means that you will have about n over 2 pairs, I am avoiding floor and ceiling, so n over 2 let us say n is a power of 2 or something that, so n over 2 pairs.

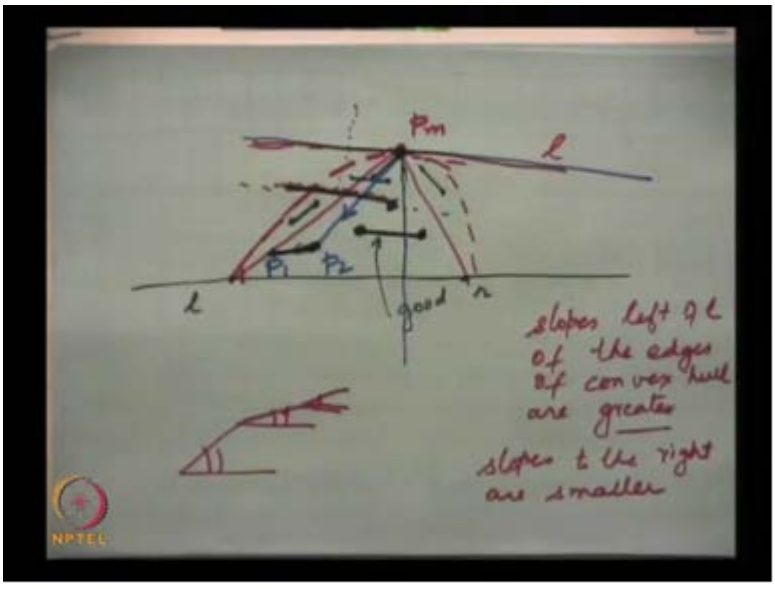
Now what you do is among these n over 2 pairs, find the pair, say call it, let us call it l m, r m that has the median slope, so of course, we are not going to compute the slops explicitly somehow figure out which pair has the median slope. So, of course this pairs can be thought about lines. So, thus look at the slopes of this n over 2 lines and l m or r m is the 1 let us say that has the median slope.

(Refer Slide Time: 05:21)



Find the extreme point orthogonal in the in direction to l m r m, I will draw a figure to explain what I mean by this. So, maybe I should stop here and then we will kind of deduce the algorithm from here, let me shift to the drawing a figure for this.

(Refer Slide Time: 06:34)



So, we are given this l r and the points now the points are paired up arbitrarily, so maybe I have 1 pair like this and another pair like this.

So, not drawing too many of them, so this is the way I have paired up the points arbitrarily, now what, suppose this, and look at the slopes of all these lines. So, if we are extended these lines, this pair defined lines, and among these lines find out the line or the pair that has the median slope, suppose it is this one, this is on the head of median slope, and then, what I mean by find extreme point orthogonal to this, is you move in the direction in this direction or you can think about essentially moving, sort of translating this, in this direction that you move in this direction, and look at the point that the last point that this line is going to intersect.

So, it happens in this case it is probably this point, so that is the extreme point orthogonal to the direction of the slope. So, this is the point that, let us call this some p's of m, so this is the extreme point in the direction orthogonal to this line. So, with this we will define the following triangle. So, first of all you draw a line perpendicular to this, so this point will define the apex of the triangle, and look at this triangle.

Now, the difference between the fork load quick hull, and this one is that I have chosen p's of p's of m in a certain manner, it is not simply the point that is further from this line, it depends on what the median slope is, and I have chosen the extreme point orthogonal to the direction, and of course, the same thing would apply that all the points within the triangle can never be an extreme point or a corner point. So, we can of course, discard all that, that is the way the fork load quick hull works.

Here, also we can discard those points, but my arguments will not use at all, because we really do not know how many points will be discarded, may be none will be discarded, so I am not going to use that argument at all while analyzing this algorithm. We will use completely separate arguments, but over and beyond what we are going to discard, we can also discard this points in the triangle. So, if you actually implement the algorithm, you would actually discard the points in the triangle also.

Why would the point p m be on the hull? So, any extreme point in any direction will be a corner point, it is like just rotating the axis. So, what I have done, I am just saying that in this direction, if I rotate the axis on this direction, p m becomes an extreme point. So, by the same logic that the left most to the right most point of extreme is also extreme, whatever direction you go, it is a basically a tangent, it defines a tangent. So, we can pass a line through it. So, that all the other points will be on one direction.

Sir how we figure out the (voice not audible10:29 to 10:34)

Maybe we will come back to that later, I mean it should be simple enough I believe, so you just draw a line from the point, you draw the line and from intersection with the triangle if able to figure out whether it is inside or outside, how many times we are going to intersect, that is the general kind of a way of figuring out, if the point is inside the polygon. So, triangle is a very special case of a convex polygon, so that is should be easiest.

I guess what you are asking me here is that, can we figure out whether it is a convex linear combination. I do not even need to do all that, I do not need to solve any linear equations, I can simply given a triangle, I can traverse start just draw a line along any direction, and if you intersects once, then it is inside and if it is intersects twice then it is outside, that is easiest way to do.

So, I have this triangle, but what is this triangle buying us, not much actually, I would actually focus on something else. So, of course, I do not know what the convex hull is, but what I know is, the convex the part of the convex hull to the left of this point p m will look something like this, and the part of the convex hull to the right of p m will look something like this.

So, what I am actually trying to draw your attention to, this is not a such a nice figure, is this is the slope of the tangent passing through p m, now if you look at the edges of the convex hull to the left of p m, they are actually ordered by slopes. So, this one has a certain angle, so let me just enlarge this. So, there is one edge and then it turns, and there it is a monotonically sort of decreasing angle for the edges, and eventually you hit this tangent. So, you are actually kind of rotating like this, so the angles are, if you look at it counter clock wise the angles are decreasing. So, that is also useful observations of property about convex hulls are particular about upper hulls, where the angles from the left to right are kind of decreasing.

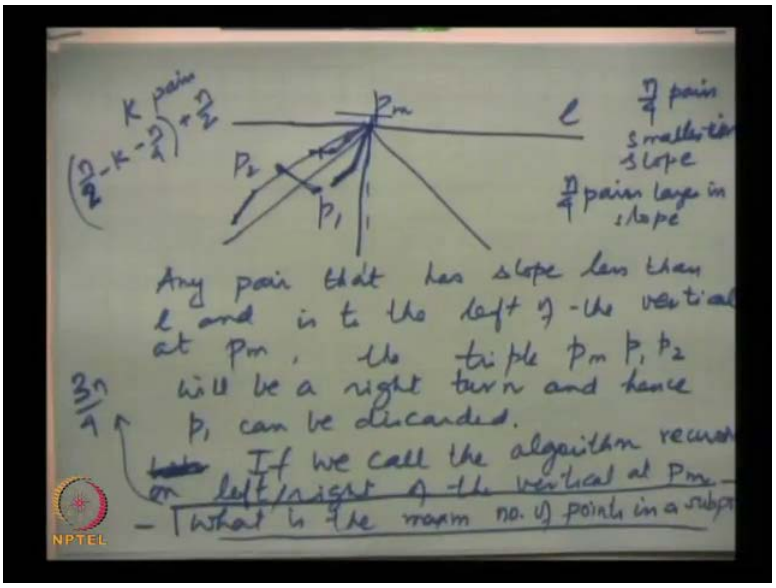
So, if you look at the left of p m, at this point p m, and this line let us say, let us call it l, so the line l has a certain slope. So, all the slopes to the left of l, of the edges of convex hull are greater, and slopes to the right are smaller. Now, you look at any pair, so look at the pair like this, I will distinguish between a few kind of pair, so this pair has the two end points on the two sides of the this vertical line that I have drawn from p m.

So, this is one kind of a pair, where I am going to somehow recurs again on the left and right hand sides, now for a pair like this, one point is to the left of this vertical line, the other point is to right of this vertical line. So, this pairs we will call as good pairs. So, this is a good pair, which essentially means the 2 points are split between the 2 sub problems that will define, haven't even a quite formally define the sub problem, the sub problems will be the part of the convex hull to the left of p m, and the part of the convex hull to the right of p m. So, these 2 points they are splitting up one in each, we will actually try to control the size of the sub problems, to the left and right of this vertical line.

So, now if you look at, let us say a pair like this, suppose I am looking at this pair, then what can you claim about this pair, these are the 2 end points, what can you claim about this pair. So, this well it does not look like that, but if I draw, let us say I pick this 3 let me label them, let us call it p 1, p 2 and p m. So, if I look at this triple p m, p 2 p 1, p 2 is the point that is closer to the vertical line, p m is a point that is closer to the vertical line; it is not p m, p 1, p 2 it will be p m, p 2, p 1, so I am looking in this order p m, p 2, p 1, and what kind of turn is it making at p 2, no I mean it is along this.

So, it is making a right turn, and can you deduce anything on the basis of this, p 2 cannot be, because p 2 you should be able to express as a convex linear combination of something. So, p 2 cannot be a point on the convex hull or it cannot be essentially a corner point or an extreme point. So, what is special about this p 1, p 2, I mean if I have right turn I will discard 1 of these points, but I may not have any right turns that I may have all left turns. Let us look at this situation, so this is the slope of, so this is the l, so this is the slope l, now I claim that in the right to the left of this vertical line, any pair that has a slope less then p m, even if you thing about a slope equal to l, sorry not slope of p m, the slope of l. So, if there is any pair in the left half, which has even a slope even equal to l, then this p m this and this point that will make a right turn. So, maybe I should grow it up, let me try it to grow it up again, let me take another page.

(Refer Slide Time: 17:48)



So, all I am saying is, I have a some kind of a triangle I have defined, this is the vertical line, here is my l which came from the median, and any pair which has slope less than this and to the left half will necessarily be a right turn.

How did you (voice not audible18:19 to 18:24)

The median slope.

So, l was obtained from the median slope of among all the n over 2 pairs. So, this is the slope, and I am claiming that anything that has slope less then this on the left hand side. If I had slope more than this, if it is like this I cannot say much, because this may or may not be a left turn, but anything that has slope that is less then l on the right hand side any pair so, both the points of the pair lie to the left hand side, if the pairs are split up I do not even argue anything about that, if the pairs lies to left of this vertical line, and has a slope less than l.

So, any pair that has slope less than l and is to the left of the vertical at p m, you can actually prove if you want, but you can see right here, that the triple p m, p 1, p 2 will be a right turn, and hence p 1 can be discarded, and you can make an analogous observation about the right half, slopes greater and to the right, again you can discard one point.

So, what is this actually giving us, so we are really aiming for the following, we are aiming for a kind of an even split, or almost even split. If the 2 points around 2 sides we

do not care, we will call it recursively on those points the left and the right sub problems, because each point has will contribute one to the left and one to the right. We are bothered about those pairs that lie to one side of the vertical line.

So, from this observation that we just made, the question is what, so if we call the algorithm recursively on left right of the vertical at p m, what is the maximum number of points in a sub problem, so that is the question what we are asking. So, what do you think it is, based on this observation, not n over 2, n over 2 will be great; that means exactly even split, what have we argued.

See those pairs whose points get evenly split we have no issues with that, so let us only think about those pairs which lie completely to one side of the vertical line, and here let us say let us argue about the left half, the same argument applies to the right side. So, what is the maximum number of pairs that can be to the left half, there are n over 2 pairs, can all the n over 2 pairs be on the left half, and if they all the n over 2 pairs on the left half what does it mean.

(( ))

So, there is something about the median here, suppose all the n over 2 pairs are to the left what does it mean. So, n over 4 of them, so we have chosen the median of n over 2 pairs. So, which means that there are n over 4 pairs smaller in slope and n over 4 pairs larger in slope, now someone says that all the n over 2 pairs are on the left half, what does it mean. So, at least half of them have slopes smaller than the line l, and therefore, 1 point among those pairs can be discarded. So, what is the maximum number of points in left half.

3 quarters, it can be n over 2, of the once whose slopes are larger, of which we cannot argue, may be it they also may define a right turn, it may be, the slope maybe larger and still may be a right turn. So, the slope may be larger than the p m, but it still may be a right turn, see this still a right turn, but the slope is rather than p m it depends on where that p m is, but certainly n over 4 of them among n over 4 of them, we should be able to discard 1 point, because those slopes are smaller, at least one point among those can be discarded so, that means the maximum sub problem size is no more than 3 n by 4. So, this is the main observation, so answer to this question is 3 n by 4, so remember this.

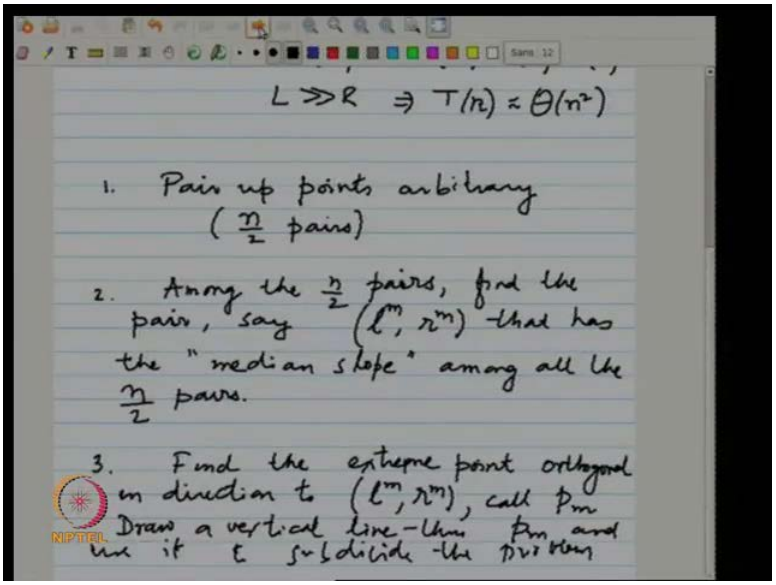
(( ))

So, there are some pairs that are evenly split up, maybe there are k of them, k are evenly split up. So, out of the k we have only. So, if there are k pairs evenly split up, then we have n over 2 minus k pairs that can be on one side completely. So, among the n over 2 minus k, you should be able to discard at least 1 point out of the n over 4 pairs. So, this plus n over 2 is the number of points on one side, and the maximum value of this is 3 n by 4.

So, that is what the median slope is buying us, it's making sure that it cannot be too much more than the n over 4 pairs, but any anything more than n over 4 pairs, half of them will be gone, and half of those points will be gone. So, the important thing is that is a constant fraction, 3 over 4 is still a constant fraction of n, so it is not that n minus 1 points can be a here, and 1 point can be there, the real bad behavior of quick sort, if you recall was that, when n over 1 points are there, almost n points are there, and the remaining are here. So, this does a good job of splitting the problem.

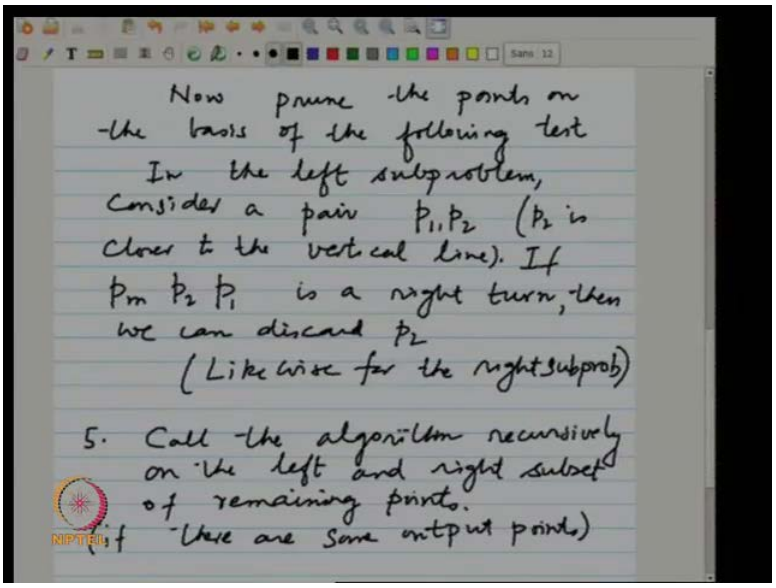
So, let us continue with the discussion with the algorithm.

(Refer Slide Time: 26:48)



So, we said that, find the extreme point orthogonal in the direction to l m, r m, call it p's of m, draw a vertical line through p's of m, and use it to sub divide the problem. So, we will use it to sub divide the problem, and then we are going to do some pruning.

(Refer Slide Time: 27:33)

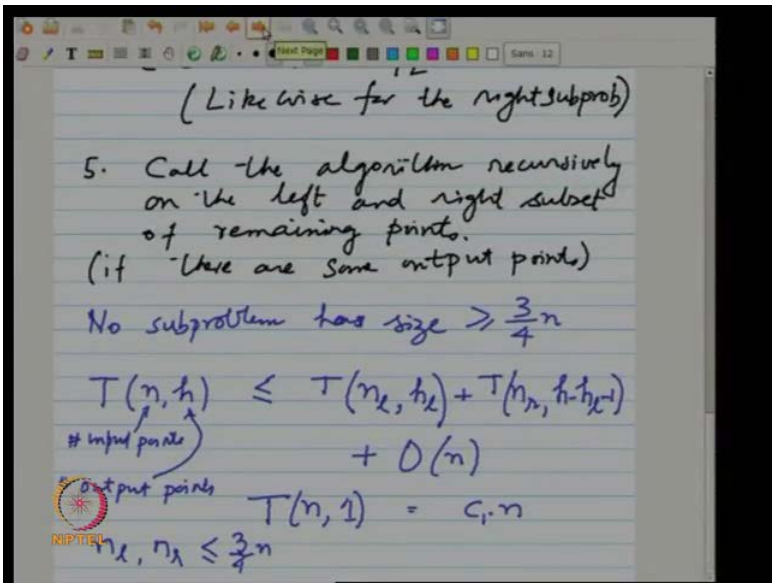


Now, prune the points on the basis of the following test. In the left sub problem, consider a pair p 1, p 2; p 2 is closer to the vertical line. If p sub m, p sub 2, p sub 1 is a right turn, then we can discard p2, and likewise, for the right sub problem, this is a Macroning strategy, and that is it. Call the algorithm recursively on the left and right subset of remaining points. When do we stop, when basically the sub problem is empty.

So, you find 1 point of course, you can always find 1 point, if you find 1 point that will be the extreme point, and then no other points will survive there, that will be the limiting case. So, if we had, there is no points outside the triangle we are done. So, we will certainly discover at least 1 output point, provided it is a nontrivial problem. If, there are some output points. So, we call it recursively, only when there are output points.

The observation that we made till now is, the important thing is no problem.

(Refer Slide Time: 30:31)



No sub problem has size greater than 3 over 4 n. So, I could possibly try to write a recurrence, and this could give me a reasonable result, because I have kind of dividing the problem into 2 hulls, where no fraction is more than 3 over 4 n, even if it is not n over 2, you can still show that it will be about n log n, but I have more ambitions, I am not giving up on the output sensitive part of it, because we are kind of looking at the output point, and doing this dividing concur. So, let us try to analyze, not only with respect to input size, but also with respect to the output size.

But this 3 n by 4 is this size of 1 borrow only, if the other half contains 0 points, means if all the points are one side only then we can say (( ))

In any other cases it is only less than 3 over 4 n right.

But, the bound if we get n log n bound so, that would be loose bound, because this does not occur in bound

No, it is a loose bound for sure, but then it cannot be better than n log n anyway. If you are not considering the output size, not taking into consideration, then in the worst case convex hull, according to the lower boundary proved is n log n anyway. So, we do not need to look for any final analysis of this, but the analysis that we will try to attempt will parameterize not only the input side, but also the output sides, and if you recall, we had looked at something similar, when we were discussing the maximal points. Now, let me

write this in terms of input and output, so T of n, h; the number of input points and output points.

What are we done, we are saying that this running time is less then T of the left half, the points in the left half, the output points in the left half, plus t of the points in the right half, plus the output points in the right half. So, we can write it as h minus h l minus 1, because 1 point we have deducted already, plus how much work have we done in doing all these things, we have done what, let us go back to the algorithm, pair of the points, trivial, order n, among the n over 2, find the pair that has a median slope, now it is a median slope, but then you can use somehow some kind of, again left hand right hand test to do this, how quickly can you find the median of a set of points, n points.

Order n time. So, I am claiming that with just slight modification of that, where the individual tests are not less than greater than, we are not going to compute the slope, but again in terms of some left hand right hand tests, we can again find this median in order n time. So, this is order n. In fact, this is not even order n; I mean this n over 2 pairs can be just looking at the first and second point, third and fourth points. So, it is basically cause zero actually, the step 1, but let me just write down order n, finding median is order n, find the extreme point orthogonal to the direction, how much will this be.

(Voice not audible 34:51 to 34:57)

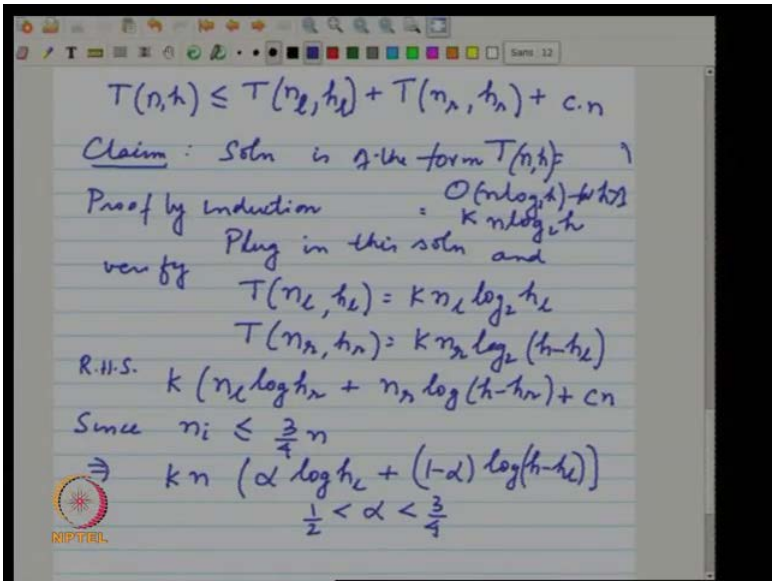
If you write a program, how would you implement.

(Voice not audible 35:01 to 35:05)

Order n, because you can think about the rotating the axis and finding the extreme points, again it is some extreme complication. So, all that, again everything, this is also order n. So, we have only spent order n time in the first 3 steps, now you want to do this pruning, what does pruning imply again, I am looking at every pair only once, there only n over 2 terms. So, again order n, and we call the algorithm recursively that is all, the recursive cost is covered here. So, overall then I can claim that this is the kind of work there I need to do. For the terminating case, let us say t n, 1 there will be at least 1 output point, otherwise it cannot be, that means convex hull must be at least 1 output even the upper hull. So, let me put down a constant it is order n, but let me put down a concrete constant, let us say c 1 times n. So, this is my reference

And last time when we are dealing with maximal points, I did not solve the recurrence and I just claimed that, so we have some nice thing about this n l and n r, what we know is n l, n r ;what can we say about them, both are less than 3 over 4 n. So, this is the one crucial things that we will require to get a bound on this recurrence, let me just quickly work to the recurrence, I could have left it as an exercise, but I thought let me just go ahead and do it.

(Refer Slide Time: 37:21)



So, let me write this as T of n , h, let me write some concrete constant c times n, we know there is a base case, I am not going to write that. So, I am claiming that the solution.

Claim: solution is in the form n log of h. So, why is this claim true. Proof by induction: so, someone may have objection to n log h, saying that if h is 1 this is 0 and though everything become 0. So, I will be a little sloppy, and I claim that, we can say this is order n log h for h, strictly greater than 1, for h equals to 1 its order n anyway. So, proof by induction basically says, that I am going to plug in this solution to the right hand side and verify.

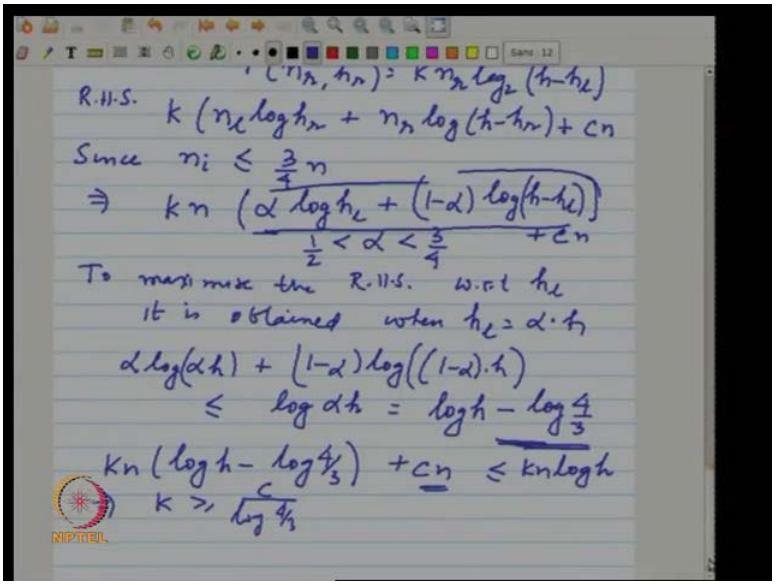
So, now this solution means I cannot use the big O notation, I must have again some kind of concrete constant, so let us say h is equal to k times n log 2 of h, for some constant k, equation and verify, because both the sub problems are strictly smaller than the original sub problem. So, this is a valid proof by induction. So, then what do we do so, we get T

of n l, h l equal to k n l log 2 of h l, T of n r, h l. So, I will be again a little sloppy instead of writing h l, I should have written h minus 1 minus h l, but to keep notations simple I just write h minus h l, I am only making it worse actually, in the sense that I could have taken out 1 from here, So, right hand side is k times

So, the total right hand side cost is basically then, k times n l log h r plus n r plus c times n. Now, if you look at this thing carefully, the real battle will be that we have this additive terms c times n, and that is somehow should get subsumed so, we will see that. So, since n's of i is less than 3 over 4 n.

So, we can write this the whole thing becomes k times n. So, we want to actually maximize the right hand side, because that is the worst case.

(Refer Slide Time: 42:58)



So, to maximize of course, this is a plus c n term, to maximize the R H S you can differentiate basically with respect to, essentially h l is what we want to differentiate with. So, I have done the job for you, I will only just mention what it is, you just do it, and it turns out that h l, it is obtained when h l equal to alpha times h. In other words, whatever is the maximum value of alphas, so, alpha is less than 3 over 4. So, only when you have basically all the output points loaded to 1 side that will give you the maximum, that actually is quite consistent with the behavior of these functions.

And, then what happens is, if you just look at this part. So, the maximum value of that is, we just substitute that, so alpha log alpha h plus 1 minus alpha times h. So, this is, which is alpha times something plus 1 minus alpha times something, and assuming that alpha take some maximum value 3 over 4, and 1 minus alpha is smaller than that. So, the whole thing should be less than log of alpha h, which is equal to log of h minus log of 4 over 3 as alpha is less than 4.

So, this is basically the advantage that you are going to get, because this is the minus term. So, if you look at this expression, and this is only for this expression, we have additive c n term, we have k n multiplied by that. So, we are going to multiply by k n by this whole thing, so you are going to get k n times log of h minus log of 4 over 3, and plus there is a c n term. So, the minus is very important, because otherwise it will be k n log h, which is the left hand side, and you have to somehow utilize this quantity, and that is utilize essentially by this minus log 4 over 3, this is a fraction, log of 4 over 3 to the base 2 is the fraction. So, we have to essentially, that means we have to choose, so this is the true.

So, this whole thing should be less than k n log h, that is what our inductive assumption was, and this is true, only if you choose a large enough k that is all, and it implies that k is larger than or equal to c over log of 4 over 3, so that is a fraction. So, as long as k is larger than this quantity, this whole thing goes through. So, if some constant larger than that, as long as it larger than that, then it means that the solution is, that the induction proof is goes through, which means that the salutation is order n log h, k is a constant, as long as we chose k constant larger than c over this the induction proof goes through. So, the reason I went through this, without asking to do it at home is, this kind of proof technique we use very generously when we are dealing with complicated recurrences, computer science also do not done, only deals with recurrence like 2 n over 2 plus order n, or T's of n equal to T's n minus 1 plus order n, these are the kind of standard recurrences that algorithm books talk about, and then the exercise is, we will find this more complicated looking recurrences, and wondering why the hell are we studying that.

So, you do come across many complicated recurrences when you are designing, at least we have a new idea who designing a algorithm, divide and concur is a very useful strategy, and it is not necessary that in the divide and concur you will always get T's of n equal to 2 T's of n over 2 plus order n, there is a fairly complicated recurrence, it is a 2

dimensional recurrence, means it is a recurrence in 2 variables, and if we are not seen this before, you would not have any idea about this thing. So, the first person who solved this recurrence so, came up this recurrence, wouldn't have had much idea and we basically have to try all kinds of things, but eventually, you will have to guess and verify, most complicated recurrence is, there are no real, what do I say, there is a no technique for solving them other than essentially guessing and verifying.

So, the final result is that, quick hull, the way we described runs in. So, let us call it the modified quick hull, just to distinguish it from the classical, what that triangles and all the points in the triangle, runs in order n log h time. So, not only have we readdressed the problem of the order n square, actually ended up getting a bound which I promised last time that, yes there are algorithms that can get you n log h

Now, I will just make 1 change in the algorithm, this is not really like quick sort, what would be quick sort. This, kind of quick sort basically means, I choose a median, and then do the partitioning, and of course then I have a nice recurrence for quick sort T's of n equal to 2 T's of n over 2 plus order n, when you are actually doing quick sort, but we do not find an median in quick sort, can you tell me why, we do not try to find the median when we are running quick sort, we find a random element.

(conversation not audible 49:39 to 49:41)

So, finding the median, although there is a linear time median finding an algorithm, it is a fairly complicated algorithm, you do not want to go and implement that, it requires some kind of space management, they are some high constants. So, you do not want to do that, we actually want to just take the simplest version of it, that is find a random element, use it as a splitter and go on, but then what happens is that your analysis is not that kind of deterministic analysis, your analysis is then we have to, probably you do not even encounter this analysis in the first portion algorithms. So, it is kind of I told you the expected running time.

So, what is known to us is, if we choose a splitter at random the running time of quick sort is expected order n log n. So, this is the kind of result you have told, and there is a proof of this is not very simple, it is not very hard, but it requires some careful reasoning, it is not an average case bound, it is not average over the set of inputs, it is for any inputs

this is the running time, because it is a only randomize to be is a choice of the splitter, and that the algorithm is controlling, it is not anyone else.

So, to make this a true quick hull, we should not be finding medians then what is the obvious thing to do, you pair them up that is fine. So, going back to our description of the algorithm among the. So, this median slope will just to be substituted with pick a random pair, among the pairs that you have created, just pick a random pair, now once you do that, it follows that. So, similar result follows for Quick hull i e expected order n log h it remains n log h n log h, but the analysis again is much more complicated than this, if you are interested I have this papers. So, I will just give you a. So, somehow this algorithm is not very well known in the community, not yet, although this algorithm the version that I represents.

So, the deterministic version is due to this paper by Chan Snoeyink Yap, the deterministic version and it goes back to something nineteen 1995 or 1996, let us say 1995. So, it is not that new, its 15 years old, but somehow most people do not seem to be aware of it, they still use that, many places in the literature, they referring to Quick hull as the triangle, which does not have an analysis, proper analysis, it is an order n square algorithm in worst case, and the randomized version, where I am saying that the analyses is lot more nontrivial. So, that is available and this paper. So, I do not know does it come up here.

So, this is a paper so, that we discovered the algorithm almost simultaneously. So, this is also around 1997 or 1996 whatever. So, you can. So, these papers are available on the web site, if you are interested in analysis of the random algorithm then look it up. So, I will stop here and next week also I will continue with convex hulls. So, I have done the algorithm part of the convex hulls, there are many more other algorithms. So, I will pick up may be 1or 2 more, to illustrate certain other points, not about the algorithm per say, we have the algorithm now. So, there is nothing more to it, and this is for the best and fastest algorithm if you implement it, but there are other aspects of convex hulls that I want as an illustration of some ideas that I want to still continue with convex hulls next week including lower bounds.