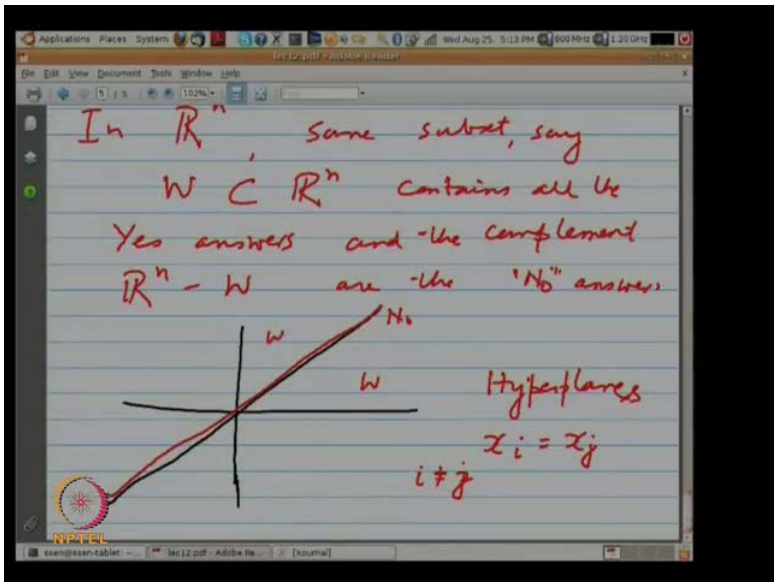
Computational Geometry
Prof. Sandeep Sen
Department of Computer Science and Engineering
Indian Institute Of Technology, Delhi

Module No. # 06
Lower Bounds on Algebraic Tree model
Lecture No. # 01
Lower Bounds

Today is the thirteenth lecture; hopefully it will turn out to be lucky. We will continue on what we started last time - namely, on deriving some kind of lower bounds on the linear decision tree model.

(Refer Slide Time: 01:00)



We started looking at this problem of element uniqueness and we are trying to parameterize…we parameterize a problem as…If you consider an input of the given N elements as a point in a high N dimensional space in R to the n and then we want to look at this space that is the Euclidean n-dimensional space where you know a point can be classified as yes or no depending on whether there is a duplicate element or more than one duplicate…we saw there were no duplicates.

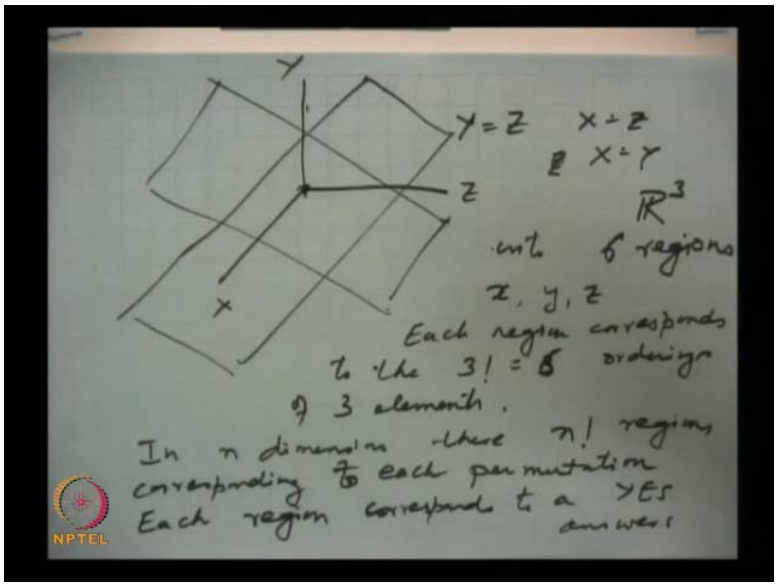
In two dimensions the space partitioning would look like…You have these four quadrants and you take at this line with slope 45 degrees and exactly the points on that line are those inputs - we are considering only two point inputs; the points in the line will

be those that are not unique, because x1 is equal to x2; any other point on either side of this line we will comprise the yes answers.

Analogously, if you go to high dimensions you can look at these pair-wise - sorry - the hyper planes defined by these x i is equal to x j where i is not equal to j and if the given input point lies on any one of these planes - the union of these planes - then the answer is no because there is at least one duplicate and if the point lies outside of the union of these hyper planes then, the answer is a yes answer.

What we are actually doing is, we are partitioning the input space into some kind of equivalence classes; in the very high dimensional space…May be even going from two to three may shed a little bit of light; when we go to three-dimensional space….You have these planes let us say you have right hand coordinate system - X Y Z right - you look at...This plane that I have just drawn is supposed to be 45 degrees, passing through the origin and it is supposed to be - what does it look like? - let us say Y equal to Z or something and you have two other planes where X equal to Z and X equal to Y.

(Refer Slide Time: 03:05)



There, you see these…for your some three such planes and if you think about it, these three planes are not completely independent in the sense that they are going to partition this r cube into actually six regions - can you say why?

Sir each of them when you.

So, yeah

That is the dividing first plane varying two particularly and one more

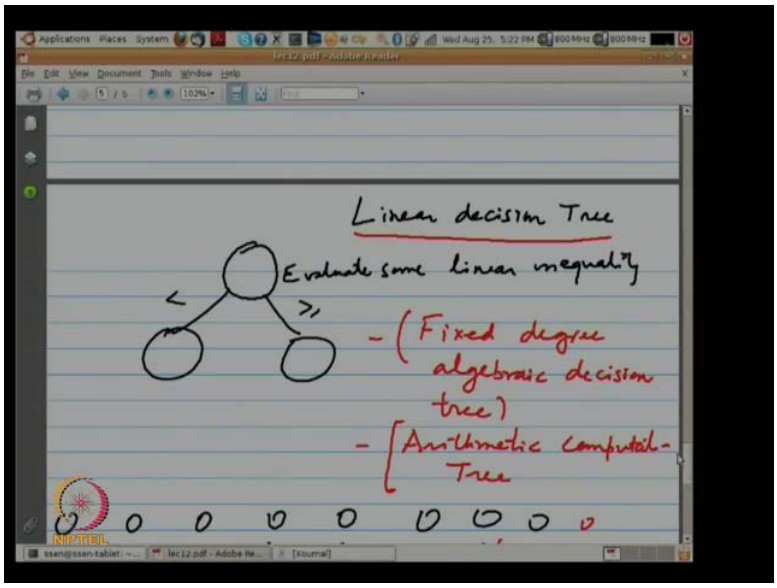
Then it divides into 4 parts, right? But then when I add the third one it is not eight parts; see the earth the earth ends - the planes, X Y plane,Y Z plane they also do the same thing; the three orthogonal axis they actually divide the space into 8 parts, but these 3 planes I claimed divide only into 6 parts; the reason is that Y equal to Z and X is equal to Z then by transitivity Y must be equal to Z; it is actually going to…two planes intersect in a line and the third plane has to pass through the line.

So, it becomes 6 planes rather than 8 planes and the other explanation is that in each of these 6 regions you have one of these 3 inequalities; sdepending on the permutation of….Let me use the - well, I am using X Y Z - X Y Z can be permuted in three ways; three elements can be permuted in three ways; each region corresponds to the three factorial that - sorry - thats equal to 6 orderings of 3 elements.

This is in 3 dimensions; when we go to n dimension - we are talking about n dimensions - the point is…the given input is an n dimensional point; input of n elements can be thought about a point in n dimensions then essentially - analogously - in n dimensions there are how many such regions? n factorial, right? The each…n factorial regions corresponding to each permutation; each region is a yes or no - the region is a yes answer; because, there are no duplicate elements…corresponds to a yes answer; so, we have essentially these n factorial regions and each region is kind of equivalent in the following way that within a region - within these - regions the answer is a yes.

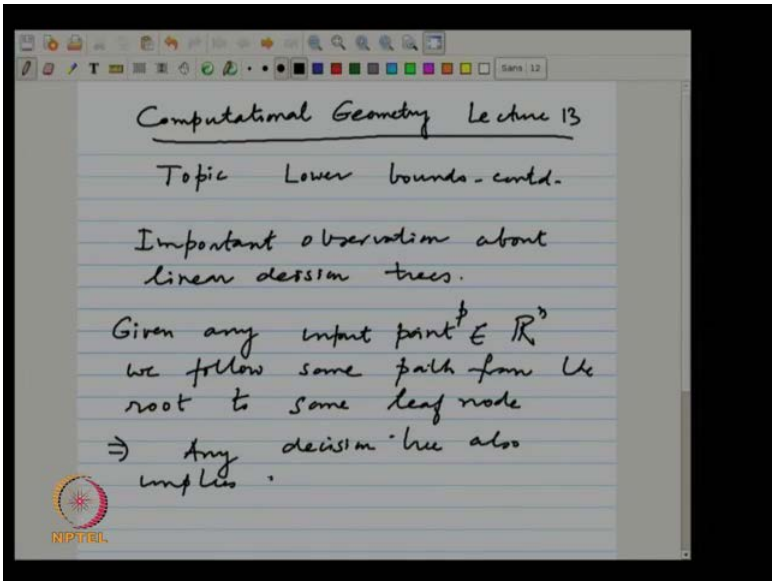
Now, what have these regions got to do with an algorithm? This is purely in terms of the input and the problem; the problem is element uniqueness problem and whatever is the input to that problem; at this point this has got nothing to do with the algorithm; somehow we will have to bring in some way that we can relate the way the algorithm behaves and how it is able to actually answer this correctly - yes or no.

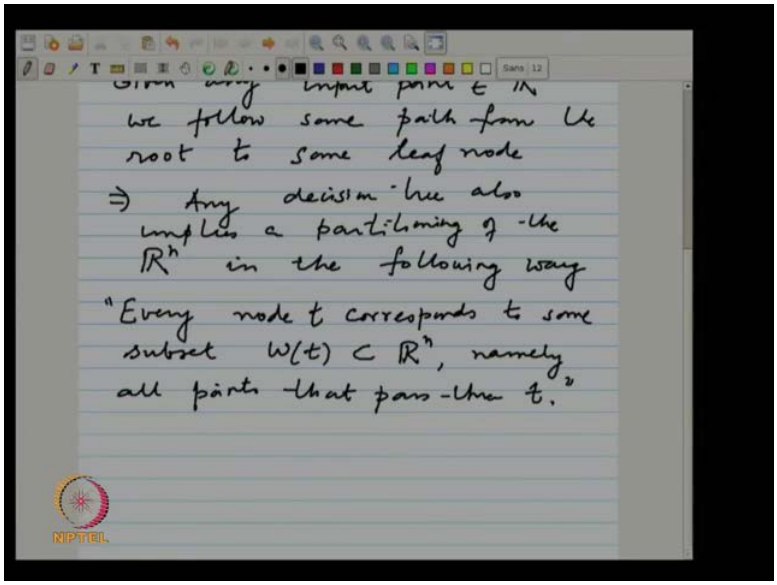
(Refer Slide Time: 08:29)



For that I need to use some…One… I will just exploit one basic property of the kind of algorithms we are looking at; the kind of algorithms we are looking at again, you recall, is in the linear decision tree model; we are not looking at fixed degree algebraic decision, tree we are not looking at arithmetic computational tree; as we discuss more you will see why I am drawing the distinction; right now my primitives are linear inequalities - linear equality is it greater than 0 sorry it does not equal to 0 strictly less than 0. This is basically…At any node of this decision tree we are making…We are evaluating this and moving left or right depending on whether the answer is yes or a no.

(Refer Slide Time: 09:16)



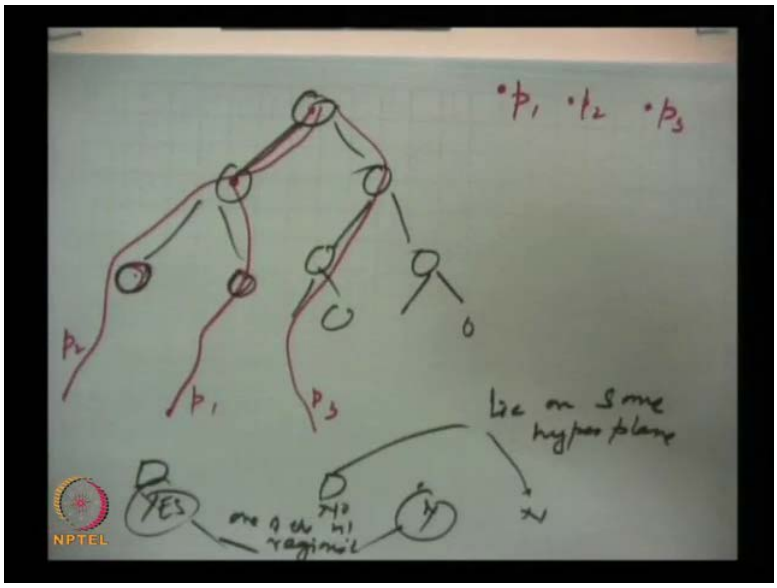


Now, these kinds of decision tree models have one important property and that is…Linear…An important observation about linear decision trees…And that is…Given any point….This point is in n dimension…The point, let us say, P in n dimension; we follow some path from the root to some leaf node which corresponds to either a yes or a no. In other words, this decision tree itself - whatever it is, some specific decision tree - any decision tree also implies a partitioning of the n dimensional space in the following way.

That is, every node - whether it is a leaf node or an internal node - corresponds to some subset, let us call it any node…what should we call it? Node…let us call it t; every node t corresponds to some subset - call it w t of R of n, namely all points that pass through t.

Whenever I have….Any decision tree based algorithm where the basic primitive is taking a linear inequality and just finding out whether it is greater than or equal to 0 at any node; whenever we reach a node, whichever points pass through that node we say that this node is associated with that subset of R(s) of n - w of t I am calling it; eventually, of course, leaf nodes correspond to all those points in the R to the power n that reaches that leaf node. Likewise, for any other internal node any point that reaches that node corresponds to that particular node., Which I am calling w(s) t corresponding to the node t.

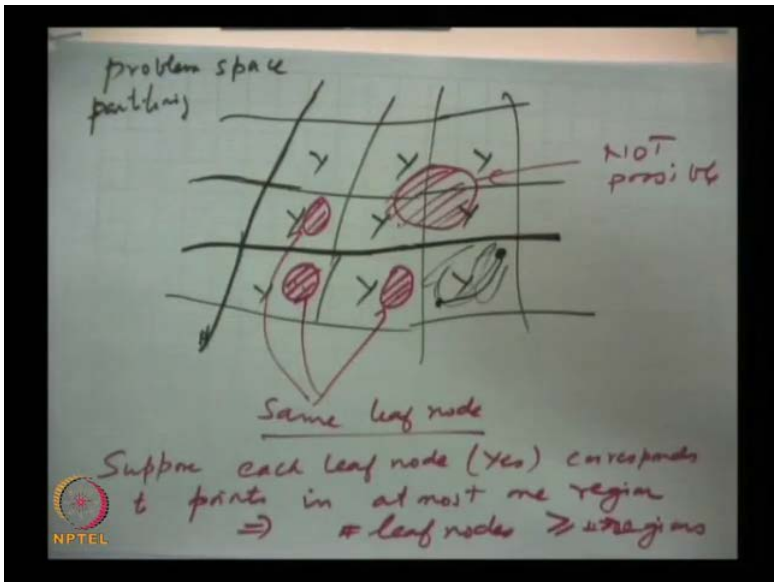
(Refer Slide Time: 13:28)



Let me draw a picture; here is my decision tree, some point, let us say, p1, there is some point p 2, there is some point p 3; when I start the algorithm with input p 1 it traces some kind of root through this tree - let us say this is p 1; p 2 may be traces another root to the tree - this is p 2; p 3 could be like this - this is p 3; if you look at the path that p 1 and p 2 took, they shared the same path till this node; but, then they went along different paths. The set of points that pass through this node and set of points that pass through this node they basically define some kind of partitioning of R to the power n - that is all I am saying.

Every decision tree also corresponds to some kind of partitioning of the R to the n; not just for this particular problem, but for any problem this decision tree will correspond to exactly those points that pass through those points; now, you have two partitioning - one is the partitioning defined by some by a specific decision tree; every decision tree will have its own way of partitioning - your algorithm and my algorithm would be different so the decision tree corresponding to the two algorithms will be different and they could imply different partitioning of the space; so, the portioning is not unique.

(Refer Slide Time: 16:44)



But, then we want to look at some properties of this partitioning so that we can relate it the problem; the problem that we are looking at is element uniqueness; element uniqueness has this n factorial equivalent regions that is defined - that is the property of the problem.

This tree also defines some kind of partitioning; now, there has to be some kind of…if you are solving the problem of element uniqueness then this decision tree that claims to solve the element uniqueness problem should also have some kind of correspondence with the portioning problem space; in particular, the leaf nodes that any decision tree has - suppose this is a leaf node yes, or may be this is a no, yes, no, whatever; all these yes points must fall in one of those n factorial regions; yes must be in one of the n factorial regions, because those regions correspond to the yes inputs.

And these no must fall into one of these unions of the hyper planes - this must lie on some hyper plane; the yes should lie in regions - one of the n factorial regions.

Let us blow up this figure it will be further…What we have now are some kind of partitioning of the problem space; problem space partitioning - what I am saying is that these are my yes regions - for this particular problem these are my yes regions and the one that lies on this line…Of course, this is not the correct representation of that particular problem, but it is an abstract diagram and anything that lies on these hyper planes they are the no regions.

There is a partitioning defined by the decision tree and a leaf node must have points the....The leaf nodes will have points either that fall on the lines or in these regions; what can we say about the regions corresponding to a leaf node; can it have something like this? For instance can the partitioning be like this? Can some leaf node basically be associated with all these points; this is the… The red partitioning is of the decision tree; so, can any single leaf node correspond to this shaded red region? What is the problem? See, it is seems to contain yes points and no points, which is completely ambiguous - so, clearly not permissible, not possible.

No single leaf node can span across two such regions; in other words, what we are saying is that a leaf node must necessarily be confined to…Well no, not necessarily, this single leaf node may correspond to this; all we are saying is that as long…The leaf node has to be consistent it cannot have both yes and no points; the leaf nodes, from what we have observed till now, is that it has to be completely contained within this region; may be, the same leaf node…May be all of them belong to same leaf node or same node, whatever; particularly, we are arguing about the leaf node perhaps they all belongs to same leaf because all them are yes answers.

This it is a plausible scenario; actually, we will prove or what we will observe is that even this is not possible; that is what we will prove. Why it is so that…. We would like to actually somehow argue that even this scenario is not possible, that is, a leaf node cannot be union of these regions coming from different… Since the word connected was used - I do not want to get in the definition of connected regions - but, you can see what is meant by connected regions here; this region is connected because I can go from any point in the region to another point in the region by staying completely within the region, that is all.

I am even saying straight-line segments, I am saying that as long as you can travel from this point to another point without going outside the region, it is a connected region; the question is that, can the leaf nodes comprise of union of subsets that lie in distinct connected regions - if not… Let us set that aside.

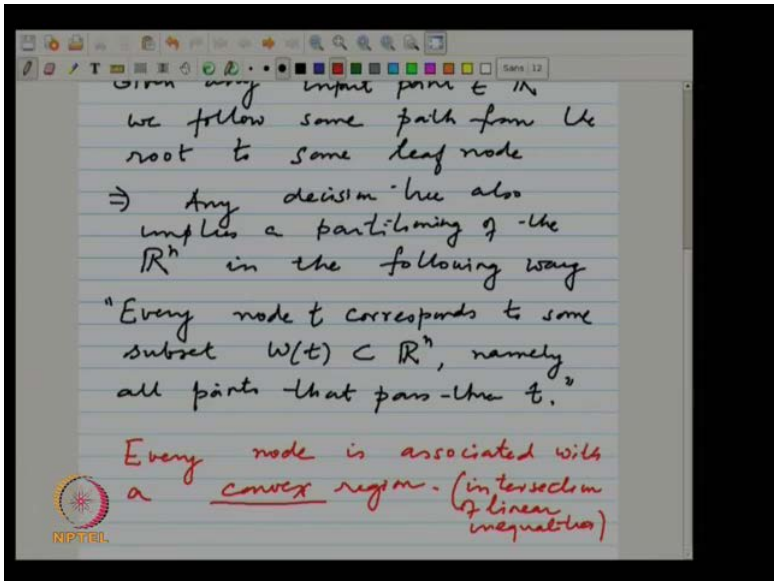
Suppose, we manage to prove that a leaf node can have…the region corresponding to a leaf node can be in exactly one connected region, what will that imply? Suppose, each leaf node - the yes leaf node, the yes one - corresponds to points in utmost one region;

this implies number of leaf nodes is greater than n factorial - number of regions…number of regions, which is equal to n factorial and then we are done, actually; you think about it - it is a binary tree and if the number of leaf nodes exceeds n factorial then you have the height of the tree to be N log N right log of this quantity, then you are done, basically.

So, why is this true? This is the connecting link. Point on the line when a no line can be expressed as a linear combination of two points in the symmetric components and when since both these answers are yes the linear combination

Why? What is the property of the linear decision tree that we are using here? The only thing that we are using in the decision tree model is that every decision is linear in quality.

(Refer Slide Time: 22:49)

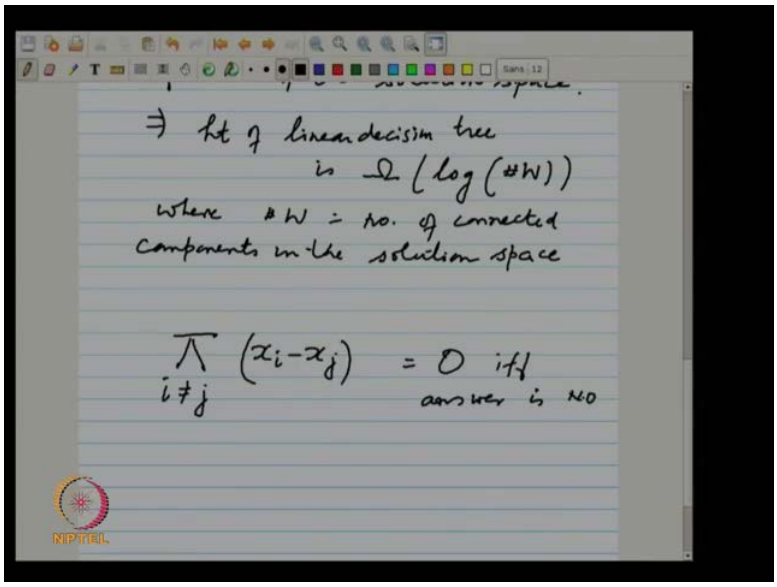


Either it is this side or that side…Every….The second property…Note…Let me highlight it - every node is associated with a convex region - this is operative word; because, it is an intersection of linear inequalities which are convex sets, that is it. The primary property is the fact that every node and therefore every leaf node is connected - is associated - with a convex region; therefore, if there are…A leaf node corresponds to two such regions it means that it must also contain any kind of linear combination; namely, the entire segment should lie within that leaf node, which means that this point

also should also lie on the leaf node; which is the contradiction because a leaf node cannot… It must be unambiguous - it must be only yes, so this cannot be.

There cannot be a no point associated with the yes leaf node; just from the convexity property follows that a leaf node can be associated with utmost one region and after that…over all if you write… We should try to capture this entire result; it basically amounts to the following that the number of leaf nodes must exceed the number of… We will actually say number of connected components solution space.

(Refer Slide Time: 24:39)



Solution space has n factorial connected components; the entire space…We are only arguing as… greater than or equal to; every region may have more than one leaf node, but certainly the number of leaf nodes must be more than number of regions, that is all we need.….What the algorithm is

Including leaf node, right?

But may be some intermediate node is there.

Any node corresponds to the intersection of inequality, which is a convex region - any node including leaf nodes.

The number of leaf nodes must exceed the number of connected components of the solution space, which means that the running time or the height of this decision tree -

sorry, linear decision tree - is omega of log of number of connected components of solution space there.

This is the summary of what we just observed; therefore, as a corollary your linear - any linear - decision tree based algorithm for element uniqueness has at least n log n n log n comparisons or linear inequalities or computes at least that so many linear inequalities.

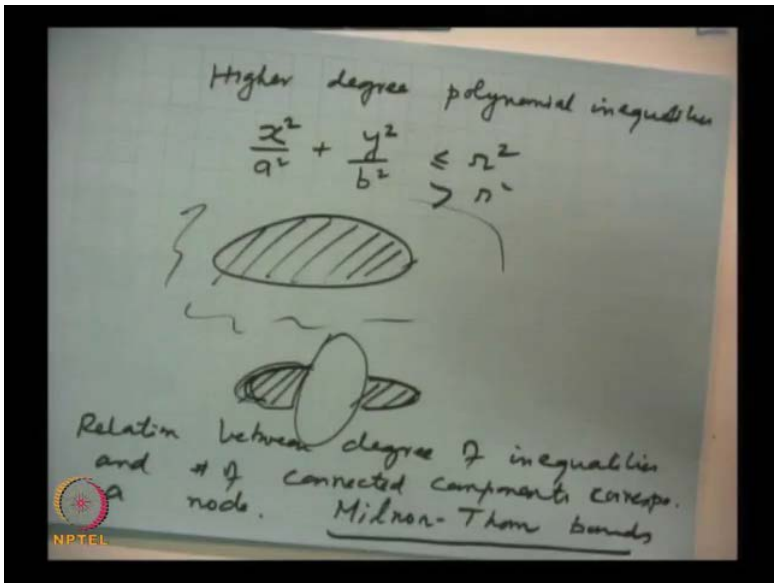
So far so fine; someone comes up and tell me, sir we do not need to compute any linear inequality I will just solve the problem in the following way - what is this? Equal to 0 if and only if answer is no; how many…forget about n square, does it compute any linear inequalities at all? Does it fit our model of linear decision tree model? No, so it completely falls apart, right? Basically, in that model it is has no lower bound because I am not using even a single linear in inequality; n square is number of…Moment I say this is a model of computation, I am only counting those operations I am not accounting for anything else.

When you talk about comparisons, when you sort you only count comparisons - n log n comparisons - someone comes and say no comparisons you know I just doing hashing, then that model completely falls apart; we will have to then somehow design a proper lower bound argument for that model; this is one possible algorithm, which does not use any linear decisions, so no inequalities; it completely beats that model - zero, essentially; there is a moment we decide to use something else, then this falls apart

Whatever, right exactly.

This is precisely the reason why proving lower bounds on the linear decision tree model may not be adequate; you would like to actually prove lower bounds using more realistic operations; that is why last time I mentioned these words - even today I mentioned them - what about higher degree polynomials? The moment you go to higher degree polynomials there is a complication; I will just mention - give you one example.

(Refer Slide Time: 30:26)



So, if I have...Suppose I have something like - I will just draw it geometrically - rather than writing I think algebraically, but essentially it comes from something like this; do you remember these kind of equations? What is it? Ellipse, right; you have some kind of an ellipse and this essentially says that this is the region we are talking about; if you say greater than something then let me talk about these regions; now, I… This is a degree two equation - it is just degree two equation.

If you take intersection of two degree two equations you could end up…I have taken an ellipse like this and taken an ellipse like this; I take one greater than and one less than; I have now…The intersection can be not one connected component; the moment I go beyond degree one I cannot say that the intersection of two such inequalities - higher degree polynomial inequalities - is one connected region.

This kind of argument does not hold any more - that whatever corresponds to a leaf node must lie within a single region; I could have actually, for the same leaf node things spread across; then, I cannot claim the number of leaf nodes must exceed the number of the connected regions of the solution space.

Precisely, that is what I am saying - the linear inequalities were a very special case; it went beyond comparisons, but it does not go far enough to deal with higher degrees - the non convex sets; you are…And it is true that you can actually go ahead and use higher

degree equations - inequalities in algorithm; no one is stopping you from doing that - it is completely realistic.

We cannot use that simplistic argument - not as simple as we did now, but people have been able to explain these bounds and I would not talk about it today; see the solution space partition remains the same - that is completely problem dependent; it is the kind of decision trees that correspond to a certain kind of primitive; the moment I increase the degree of this primitive, then I cannot use the simple counting thing; but, the what people have manage to do is that there is relation between - there is some kind of relation between - degree of inequalities and the number of connected components corresponding to a node.

This relation comes from some fairly deep theorems in algebraic topology and the one that we could have discussed - but, there is no point in just telling you about particular bound, but I will mention it; it follows from - all east historically it follows from what is called Milnon-Thom equations; Milnon-Thom bounds will give you some kind of handle on the number of connected components relates… Suppose I could say that if the degree is two the number of connected components is two then more or less what we said would hold; but, it is not that simple - it kind of grows exponentially with dimensions.

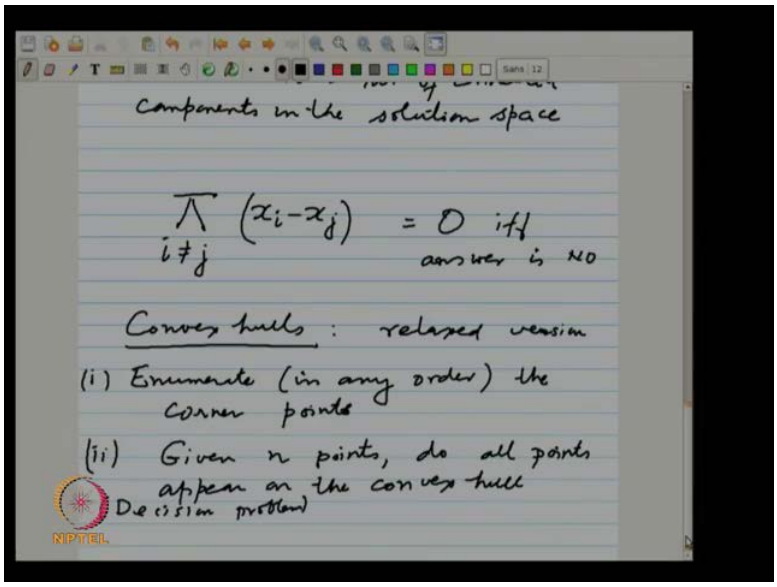
Then one had to sort of refine these arguments and finally, by the way, same kind of bounds hold - that for element uniqueness problem the height of the…even higher degree algebraic decision tree is roughly about log of the number of connected components of solution space; it requires some more careful counting, some more careful arguments and use of these things called Milnon-Thom equations.

If you are interested I can give you reference; in fact, it is there…if you have a copy of (( )) it even has that formula, but I do not want to just throw a formula that will not make sense; the formula itself is not very complicated; I will just limit my discussion on lower bounds to essentially linear decision tree models with some idea that…Essentially it is counting of the number of connected components with a leaf node - that is the central thing to get the bound and that can be done using some more fairly deep equations. That is one part of the lower bound and this the lower bound is for element uniqueness and therefore the same lower bound also holds for what other problem?

We could reduce element uniqueness to sorting; even if you were told that I just posed that problem, instead of comparison I am allowing use more powerful primitives like X 1 minus X 2 plus X 3 something like that - it is greater than equal to 0; even with that restriction, sorry, even with that flexibility you will still get a n log n bound and sorting; what really kills sorting is the moment you allow things let indirect addressing.

The next application of this bound will be to this convex hull thing that I have been promising; let us look at the convex - reopen convex hulls; the lower bound that I derived for convex hulls was connected with - was related to - sorting; that if I want the points to be in the convex hull - to be output in sorted order - then sorting can be reduced to convex hulls; therefore, lower bound for sorting applies to convex hulls, but it is a kind of a weak lower bounds; someone could say that I am not interested in looking at the ordered output of the convex hull I will be happy if you can tell me…

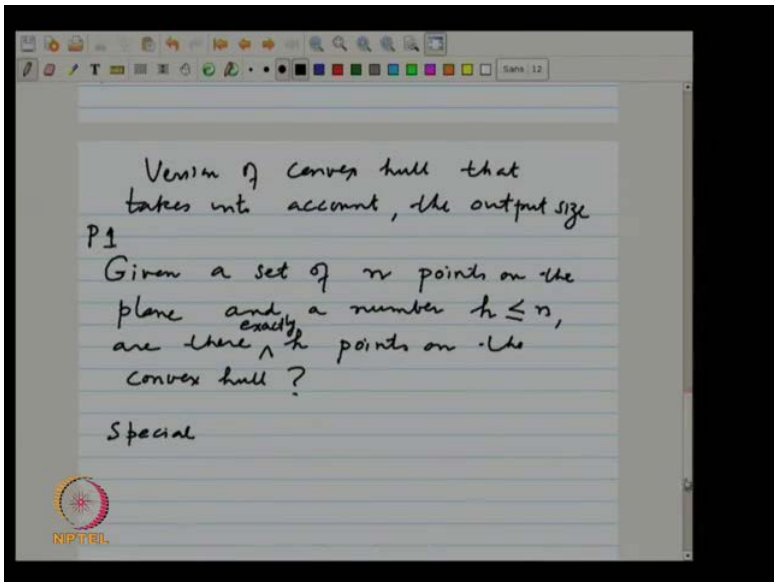
(Refer Slide Time: 37:22)



For convex hulls relaxed version can be the… Just enumerate in any order - I am not imposing any kind of sorted order - the corner points; this could be one version; another version could be…since we are more comfortable about lower bound for decision problems, let us make it a decision problem; given n points, do all points appear on the convex hull? Which is certainly a decision question, right? The one that we will actually pick up and which will also address this number two is…Where we will also somehow

incorporate the notion of output, because I have been telling you that this input and output together…We have an algorithm that combines over the inputs and outputs.

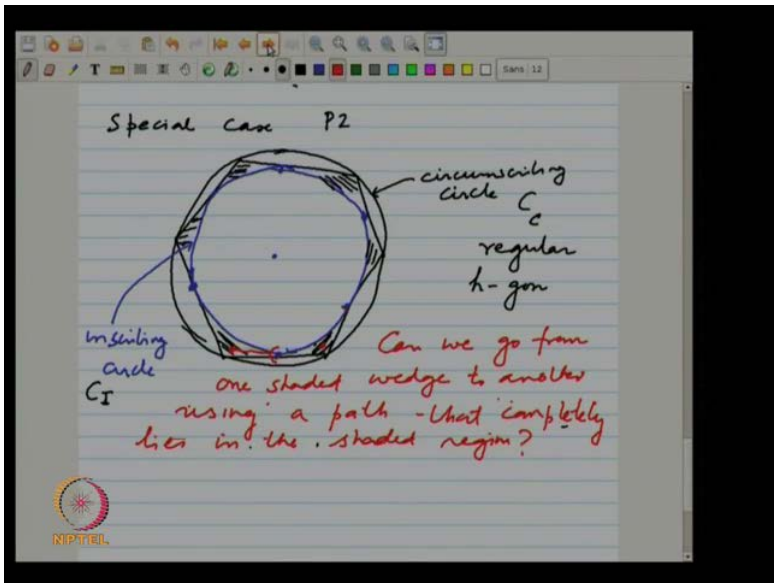
(Refer Slide Time: 39:16)



The problem that actually that will take up is the following - version of convex hull that takes into account the….That version will be as follows - given a set of n…We talk about two dimensional convex hulls - if you can get a lower bound on two dimensional convex hulls it automatically implies lower bounds on high dimensional convex hulls.

When we say h points it is exactly h points; I give this input set and give you some number - hundred - tell me whether there are exactly hundred points on the convex hull; some number and that number must be less than n because there can be more than n points in the convex hull; in fact, it should be between 3 and n because it has to be at least 3.

(Refer Slide Time: 39:19)



For this problem, whatever it divides is a special case of this problem; let us call this problem p1 and then we will talk about special case p 2; p 2 looks like this…I have…this is a regular h-gon and this is the circumscribing circle of a h-gon; I also have…similarly, this is my inscribing circle and this is the center of the circle.

What I have is a regular h-gon, which means that all these internal angles are the same and all these edges are of the same length; I am not going in to arguments where you can draw a regular h-gon for any h; then I construct the circumscribing circle and I consider the inscribing circle - I did not write very well - there is a touching point basically everywhere - exactly one touching point.

This structure essentially looks like…There are these wedges - this is the shaded region - and these shaded regions have no connections between them, that is, if I cannot…Well I mean…let me define what I mean by no connection; if you consider that I want to travel from, let us say, one shaded region to another shaded region - can we go from one shaded wedge to another without traveling out of the…let us call as circumscribing circle as C I and inscribing circle as - no, sorry C C and C I. Inscribing circle is called C I and the circumscribing circle is called C I; can we go from one shaded wedge and to another without traveling outside of C C and inside of C I - that is the question I am asking.

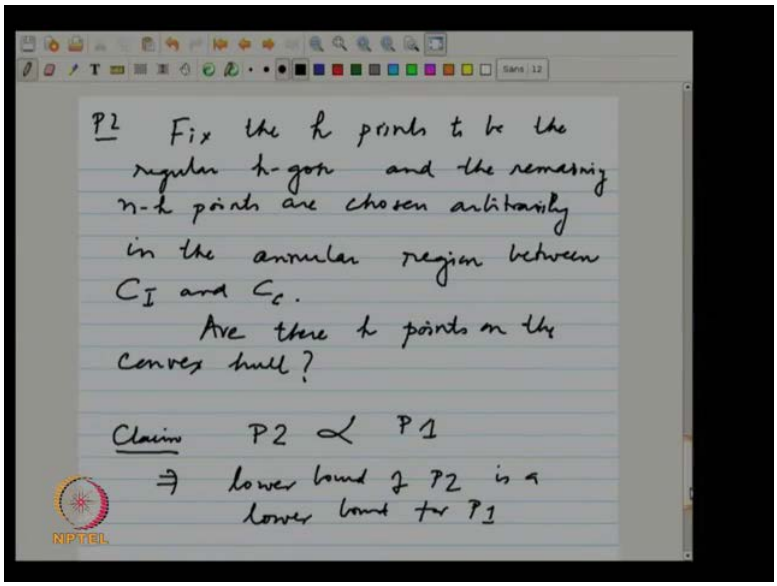
If I want to travel from this red point here and I want to, let us say, go the joining one - my path either I will be well well sorry no no no please please I retract this, just a

moment; one shaded wedge to another without traveling out of… no, not…just outside of C I should actually also bring in to the picture…no, not the boundary points…I should…let me not…I will make it even simpler - sorry; can we go from one shaded wedge to another wedge using a path that completely stays within the wedge in the shaded region.

The shaded region cannot include these points - the points of contact; essentially, what I am saying that is that these are disconnected, I cannot…but I cannot go from one wedge to another, if I try to go there I will have to sort of… whatever I do I have to take this path - either I am going to cross into the inscribing circle or I am going to come out here outside of this h-gon; or you know, essentially, I am going to go out of the h-gon have to go inside the inscribing circle.

What has all this got to do with lower bounds; the special case p 2 that I am trying to pose is the following - that we are actually given the points of the….The p1 was that, given a set of n points and some number h are there h points on the convex hull?

(Refer Slide Time: 49:08)



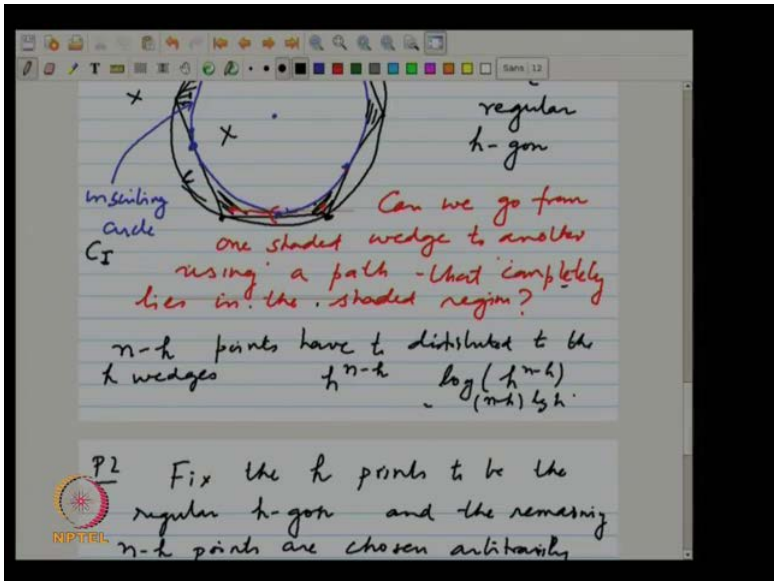
Now, I am actually fixing this edge points; fix the…p 2 is - fix the h points to be the regular h-gon and the remaining n minus h points are chosen arbitrarily; in the region - in the annular region - between C inscribing and c circumscribing; this h points are fixed - 1 2 3 4 5 h - the remaining points have to lie - it cannot lie within this inscribing circle, not allowed - it is not allowed - to be outside of circumscribing circle; it has to lie basically

in this annular region - the small ring between the inscribing and the circumscribing circle.

This is how I am defining my input and then, I ask the same question - are there h points on the convex hull? This is my problem - the new problem; claim is p 2 - the second version of the problem - is easily reducible to my original problem; why? p 2 only says these h points have to be on the h-gon - sorry, the vertices of the h-gon and the remaining points should be in that annular region, which means basically I take every point and see whether it is inside the - sorry, it is outside the inscribing circle and inside the circumscribing circle.

So, it is just two tests - two inequalities - that I have to test; in linear time I can reduce problem p 2 to p1 - if I can solve p1 I can solve p 2 because the only thing have to make sure is that every point must lie in the annular region, which is just a linear time test - every time, which implies that lower bound of p 2 is a lower bound for p1 because p 2 is reducible to p1.

(Refer Slide Time: 52:29)



How do I get to a lower bound of p 2? Why is it easier to for us to get a lower bound to p 2? See, the n minus h points…if there has to be exactly h points on the boundary…now I am going to count the number of connected components; in the solution space there is a certain number of connected components - we know that the height of the decision tree must be at least log of the number of connected components.

h points are fixed, the reaming n minus h points must lie in one of these wedges; because, if a point lies outside the wedge there has to be more than h points on the convex hull; so, between two yes configurations…essentially a distribution of these n minus h points - n minus h points have to be distributed to the h wedges; how many ways can you do that?

h to the power n minus h…Now, I claim that between…any of these two yes instances they must be in different connected components of the solution space; now, the solution space is in this high dimensional space - high dimensional space here is, we have n points and each point has two coordinates - we are talking about the R to the power two n - two n dimensional space; in that space I am trying to sort of visualize what are the connected components of the yes regions; I claim that the number of connected components is h to the power n minus h; why? Because, to go from any yes instance to another yes instance, you have to necessarily go through a no instance; there is no path that connects to wedge without going outside of the solution - I have to either cross inside or I have to go outside.

There is no way…I cannot draw a path in that region where I travel from this point to this point without crossing some kind of - whatever - separating boundary which is a no solution - there is no path; therefore, when I…That is basically what this figure… This is the two dimensional projection of that actually. I cannot go - because, if I try to go in this… if it is not connected in the two dimensional picture, so it cannot be connected in that high dimensional picture; log of this, which means that…And log of h to the power n minus h is n minus h log h - that is the lower bound.

Just think about it and you if you have any questions we can take it up later on; otherwise this is the end of the story for lower bounds; I mean, I am not going to go any further into lower bounds.