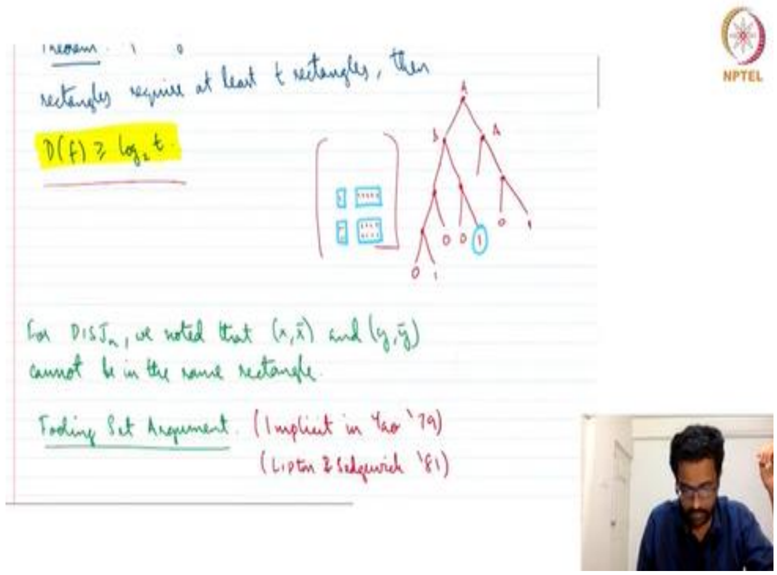
Computational Complexity
Prof. Subrahamanyam Kalyanasundaram
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Lecture - 56
Lower Bound Techniques

(Refer Slide Time: 00:15)



Hello and welcome to lecture 56 of the course computational complexity. In the previous lectures we discussed communication complexity. This was a model where, there are two parties, Alice and Bob and they together want to compute a certain function. And what we measure is the amount of communication that happens between Alice and Bob, and both of them are considered to be computationally powerful.

But what we focus on is the extent of communication that happens between Alice and Bob during the computation of the function. So, we described the model and we saw what is a protocol, we saw what is a definition of communication complexity. And we saw this example of a protocol tree, so where Alice speaks and then it is a binary tree, where they exchange like based on the Alice say something, maybe then Bob says something, then Alice say something back.

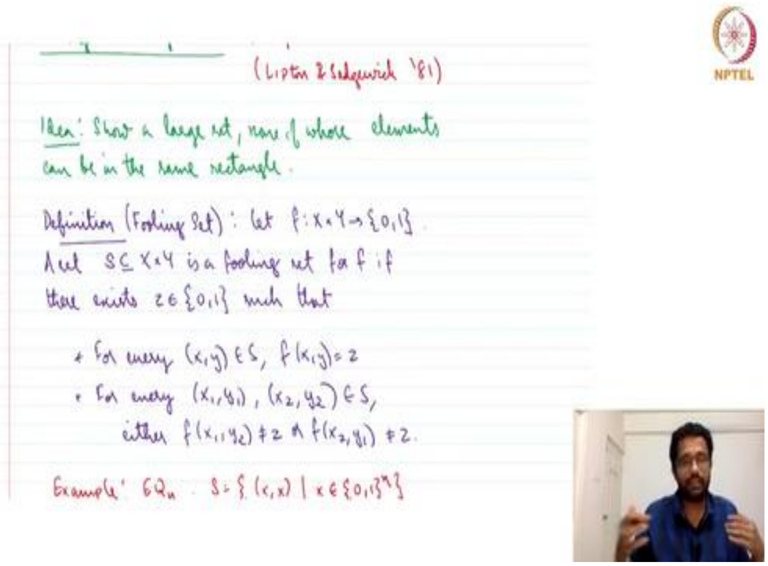
So, there is a tree and at the end the leaves indicate that the function value is 0 or 1 or 0, something like that. So, at each leaf the function value is a certain number, and we also saw the definition of combinatorial rectangles, so combinatorial rectangles are certain rectangles they need not be contiguous rectangles, but they are rectangles in the matrix that indicates, that is described with the function, that they are computing.

And we further said that each node of this protocol tree corresponds to a combinatorial rectangle, and that each leaf corresponds to an f monochromatic combinatorial rectangle. So, let us consider this leaf that I am on circling with blue. So, this leaf will correspond to a one monochromatic rectangle, so this will correspond to a set of inputs in the matrix that will be a one monochromatic rectangle.

So, it need not be a contiguous rectangle it could be something like this, it could be four pieces or something like that. So, all of the entries here will be once, so this is what we said. And because of that if any partition of this matrix, let us say we try to divide this matrix into monochromatic rectangles. If we consider a partition and whatever way we try to divide this matrix into monochromatic rectangles requires at least t rectangles.

The smallest number of rectangles that you can split into, smallest number of monochromatic rectangles that you can split the matrix into is t. Then, we need to have at least t leaves in this tree. So, the deterministic communication complexity is at least log t, so which is what this highlighted result here, this is what we saw in the last lecture.

(Refer Slide Time: 03:47)



And, we will also saw the one lower bound for the function disjointness, also for equality. So, now we will in this lecture, we will see two or three techniques that build on these combinatorial rectangles. We will see two or three techniques to get lower bounds for communication complexity. So, these are again, these are since we are not there is not a course on communication complexity just these are just introductions to some of the techniques.

And some very basic simple steps. If you are interested there is a book on communication complexity by, there is a book by Kush Levitz and Nisan, you can check that out. Anyway, for disjointness, we saw that consider x, x complement and y, y complement, we saw that these are disjoint. But, at least one of the pairs x, y complement or x complement, y must not be disjoint. So, they cannot be in the same rectangle is what we said, same monochromatic rectangle.

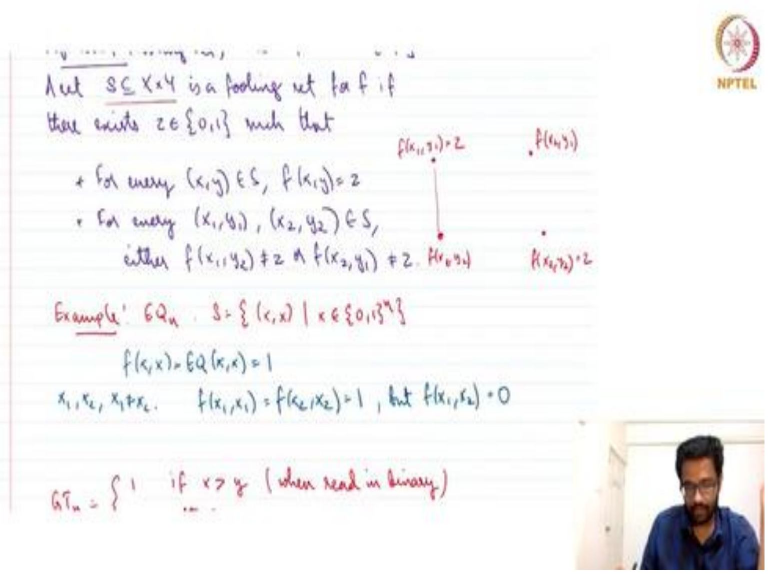
So, now let us try to generalize this idea. So, this type of argument is called fooling set argument and this was implicit in the Yao's paper that introduced communication complexity but made explicit later by Lipton and Sedgwick. So, the idea is to show a large set of inputs pairs, such that none of these pairs can be in the same rectangle. So, if you should hundred input pairs none of which can be the same rectangle then all of these have to be in separate rectangles.

So, there has to be 100 rectangles and that gives you a bound on the number of leaves, and therefore a bound on the height of the tree, which is what we are looking for. So, what is a fooling set? A fooling set for a function f is a set of input pairs. So, S is subset of x cross y. If there is some z for which for all the members of the fooling set, for all the members of x, y of the fooling set, so, S is a fooling set f of x y is z, and so think of z as let us say 0 for simplicity.

So, one may say that for all x, y in the set f of x, y is 0. And, if you take 2 members of the set x 1, y 1 and x 2, y 2, now consider the cross points. So, we know f of x 1, y 1 is z, we know f of x 2, y 2 is z, what the claim is that at least one of these two, either f of x 1, y 2 or f of x 2, y 1. One of these should not be z, one of these opposite diagonal entries should not be z. So, because of which they cannot be in the same rectangle.

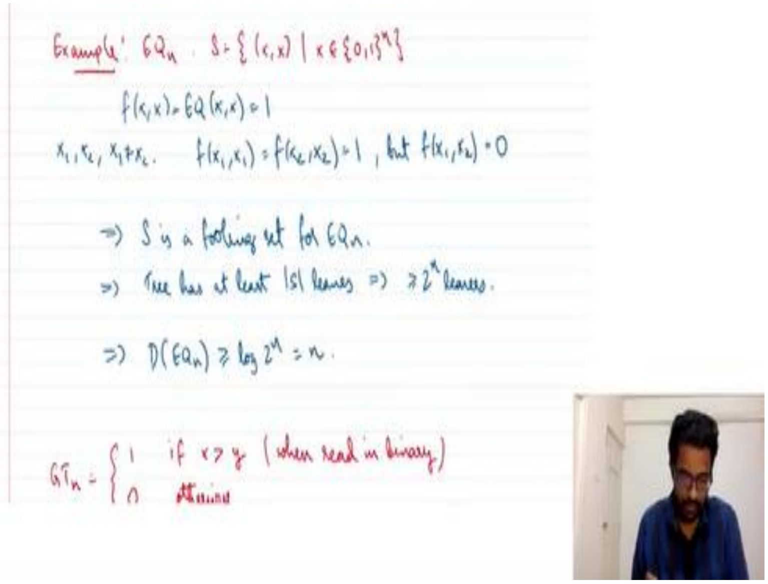
So, z could be 0 or 1, so this is what we mean by fooling set. All the members of the set should have the same function value, but if you take any two members of the set x 1, y 1 and x 2, y 2. Then either f of x 1, y 1, x 1, y 2 should not be z or f of x 2, y 1 should not be z. So, maybe some simple examples are in order equality.

(Refer Slide Time: 07:37)



So, consider the set of all numbers f of x, x and these are so we know that f of x, x or where f is equality here, this is 1 because these are equal. But, consider when let us say x 1 and x 2, where x 1 is not equal to x 2. We know that f of x 1, x 1 and f of x 2, x 2 = 1, where f is equality. But f of x 1, x 2 is not is equal to 0, it is not equal to 1. So, any x 1 and x 2 that you take f of x 1, x 2 is not 1. So, this means, this is a fooling set.

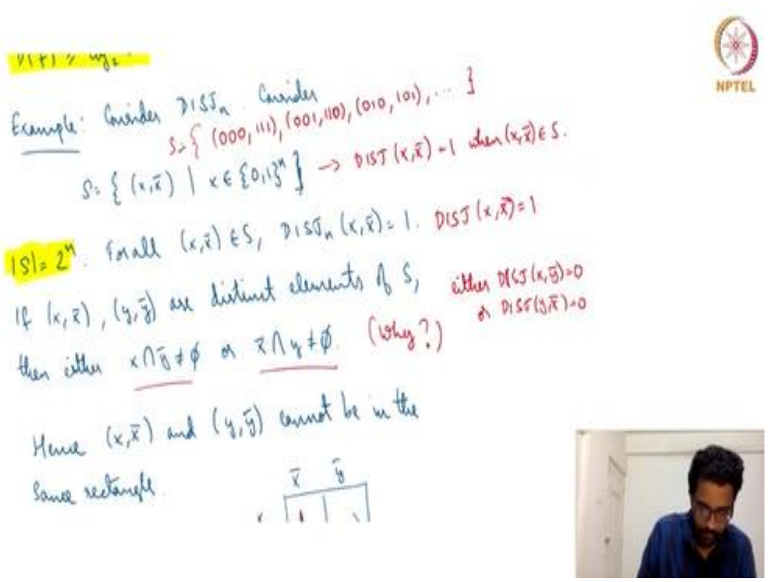
(Refer Slide Time: 08:37)



So, this means s is a fooling set for equality. Now, since s is a fooling set that means, deterministic tree has at least any tree that computes a function, any protocol that computes the function. That protocol must have at least size of as many leaves, which is at least 2 power n leaves, because there are 2 power n values of x, so 2 power n elements are there in the set S. And this means that since it has 2 power n many leaves D of equality.

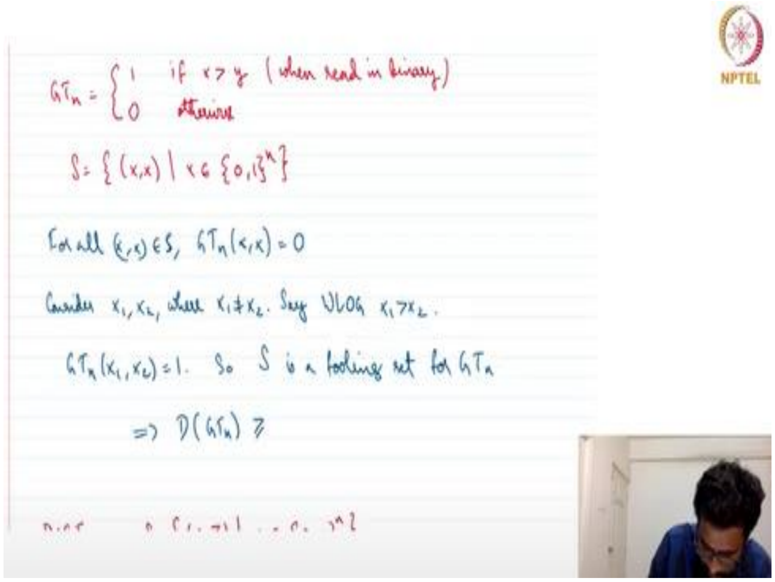
We have already seen a proof for this is at least log of 2 power n, which is equal to n. So, this is a deterministic complexity of equality.

(Refer Slide Time: 09:46)



And if you recall in the last lecture, we made the similar argument for disjointness. We said that you consider S or that is a set x, x complement, then the cross elements may not be disjoint. So, there should be at least S many leaves of the tree, so log S the lower bound on the communication complexity is at least log S which was again n. So, same argument we are making here just that the fooling set is different.

(Refer Slide Time: 10:17)

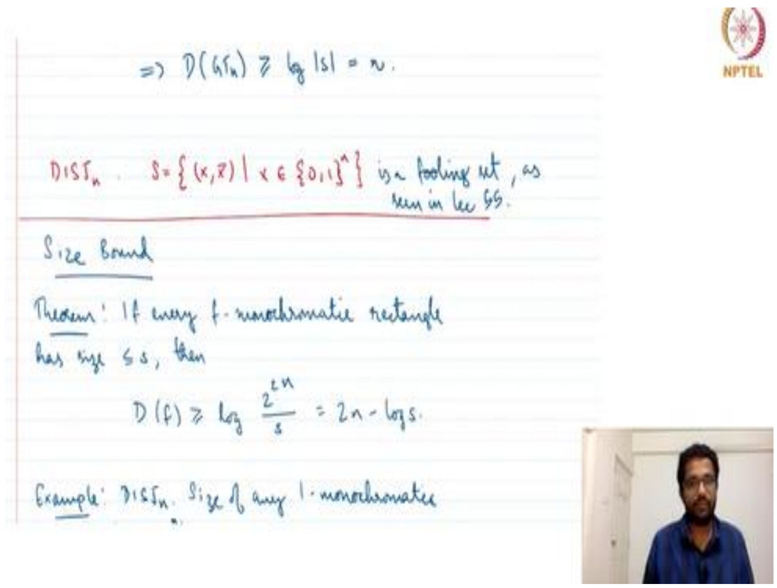


Now, we will see another example, this is a similar example greater than, so in this case the set is the fooling set is the same set actually. So, consider so GT n is 1 if x is greater than y, when red is a binary number. So, for all x, x in S, GT n of x, x is 0. Because, the same number x = x, so it is not greater than x. But, consider x 1 and x 2, where x 1 is not equal to x 2. Let us say without loss of generality without loss of generality x 1 is greater than x 2.

So, without loss of generality means one of them has to be bigger than the other, because they are not equal. So, let us say x 1 is greater. So, now GT n of x 1, x 2 = 1, because x 1 is greater than x 2. So, again this is a fooling set, because pick any x 1 x 2, one of them is bigger than the other, so the greater than function will be equal to 1 for that entry. So, S is a fooling set, for greater than.

This implies deterministic communication complexity of greater than is at least log of the size of S, which is at which is equal to n. Again, the size of S is 2 power n, you can see.

(Refer Slide Time: 12:20)



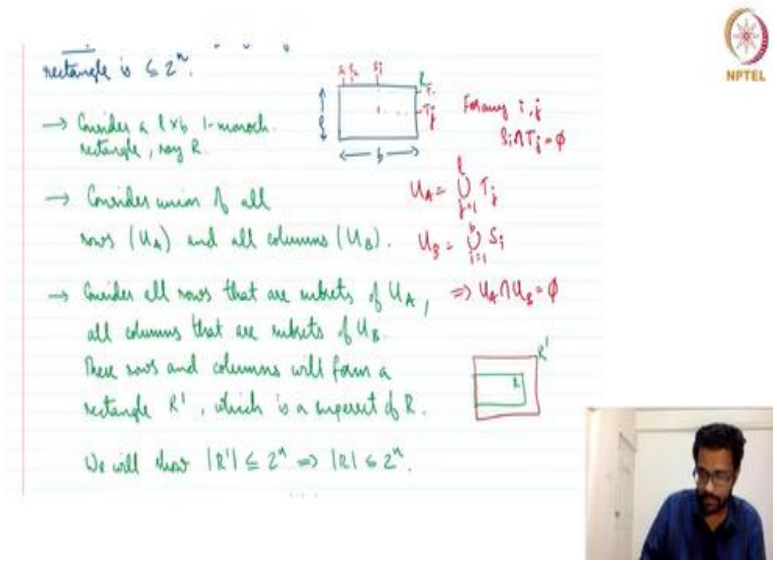
And disjointness we already saw this last lecture, this is a fooling set as we have seen in last lecture, I think 55, so this is 56. So, this is one approach to showing lower bounds. So, again the template is we want to show that there are many leaves so the height is big, when the height is big the protocol the number of bits exchanges large. So, it usually boils down to showing lower bounds, although sometimes we also have to show upper bounds.

Because, upper bounds to show upper bounds it is enough to show some protocol and this is sometimes most of the times this this happens to be a bit more easier, so lower bounds are more interesting. So, we in fooling set, we try to show a big set such that none of these elements can be in the same rectangle, so the number of leaves have to be large, so the height has to be large. So, another approach is to show a bound on the size of monochromatic rectangles.

So, if we say that every monochromatic rectangle is small, then we need many monochromatic rectangles to cover the entire to partition the entire matrix. So, one way to show a lower bound in the number of leaves is to show a upper bound on the size of a monochromatic rectangle. So, the statement is very simple, if every monochromatic rectangle has size at most S, then we need so there are 2 power 2n elements in the matrix. So, the matrix is 2 power n cross 2 power n.

So, 2 power 2n elements in the matrix. So, each monochromatic rectangle has size at most S, so any partition into monochromatic rectangles requires at least 2 power 2n divided by S mini rectangle. So, this is the t that we stated here. Any partition of x cross y into monochromatic rectangles require at least t rectangle, this t. So, D f is the log of this and the log of this is log of 2 power 2n - log of x, so log of 2 power 2n is simply 2n, and log of S. So, this is one way to get a lower bound on the communication complexity.

(Refer Slide Time: 15:11)



So, we will see one example for this, in fact it will be slightly different, so what we will show here is? We will show that the function is disjointness again we have seen a lower bound for this, but this is another approach. So, we will show that the size of any one monochromatic rectangle is at most 2 power n. So, this by itself is not enough because, we have to show that to get a lower bound, we have to show that the number of rectangles is large.

But here we are only upper bound in the size of a one monochromatic rectangle. So, this is not going to be enough I will explain the end of this proof. But it is easy to fix. So, y is the size of any one monochromatic rectangle in the disjoints matrix at most 2 power n. So, consider the rectangle let us call it R. So, this is one monochromatic rectangle, so maybe there is some set, S 1 here some set T 1 here.

So, maybe I write R to the corner, so this is R and all the entries here are 1. So, this means S 1 and T 1 are disjoint. So, now we have many sets, let us call S i and T j may be, so s 1, s 2 and so on and similarly T 1, T 2 and so on. So, suppose R is, so we want to show that R has size at most 2 power n. So, let there be l columns, so I say l cross b, so but I have marked b cross l, so maybe I will fix this, so l cross b matrix R. So, this is a one monochromatic rectangle.

Now consider the union of all the rows, which is T 1, T 2 etcetera. So, there are l rows. So, the union of all the rows, so u A, we call it u A and we will use another colour u A is just nothing but union of i or j equal to 1 to l T j. And, u B is the union of all the columns i = 1 to b S i. So, u A is the union of all the sets, so when I say union I mean union of the rows, I mean this union of the sets denoted by the rows and the union of the sets denoted by the columns.

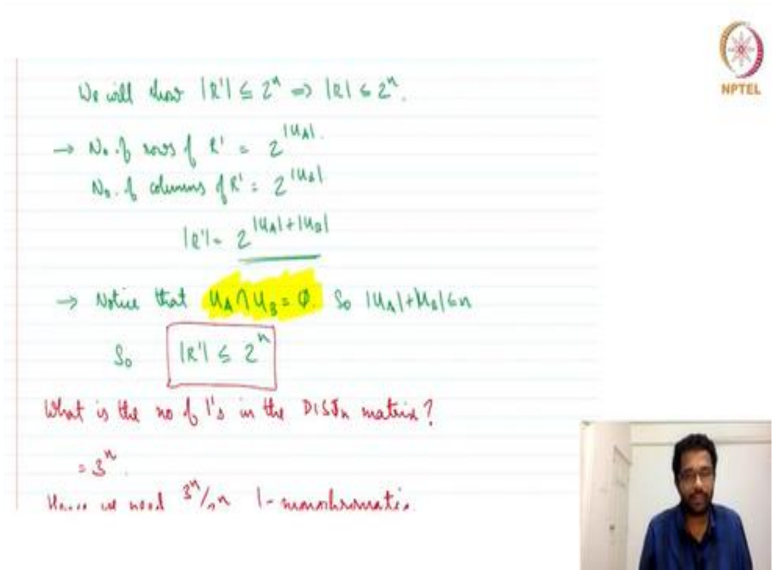
So, now each one of S 1 any S i is disjoined with any T j, so for any this is one monochromatic rectangle. So, any S i, for any i, j S i and T j are disjoint. So, it should be the case that even for the union, union of A, the union of the rows should be disjoint from the union of the columns. Because, any one of these is disjoint, so the union also will be disjoint. So, union of A and the union of the rows and the union of columns will also be disjoint.

So, which is not very difficult to see. Now, consider a bigger matrix, consider a bigger set now consider all the rows that are subset of u A, so u A is the union of all the rows here. Now, there could be other rows that are subset of u A, but not in this matrix R, not in this rectangle R. And consider all the columns that are subset of u B, even those that are not in R. So, this could possibly be, so let us say this is R, what I am saying is?

Consider all the rows that are subsets of u A. So, there could be other things that are not in R. So maybe I will use red colour, so maybe it is something like this. So, but then R will certainly be contained in this, because all the rows corresponding to the rows of R, are indeed subsets of u A, because u A was constructed by union taking the union. And all the columns of R are subsets of u B. So, that will certain, so R will certainly be a subset of this matrix.

So, now consider this matrix R prime, and what we have the claim is that? R prime is a superset of R, which is also not very difficult to see.

(Refer Slide Time: 20:27)



What we will show? So, we wanted to show that any one monochromatic rectangle is of size at most 2 power n. So, we want to show R is of size at most 2 power n. In fact, what we will show is R prime is of size at most 2 power n. So, which means R prime is a bigger it is a super set of R, so which implies R is also of size at most 2 power n. How will we show that? So, what is the number of rows of R prime? R prime has all the subsets of u A as rows.

So, if u A has size of u A many elements, the number of rows of R prime is 2 power the size of u A. Similarly, the number of columns of R prime is 2 power the size of u B. Because, the columns of R prime are all the subsets of u B. So, if u B had 20 elements, then R prime will have 2 power 20 columns. So, the size of R prime is nothing but the product of the number of rows and number of columns.

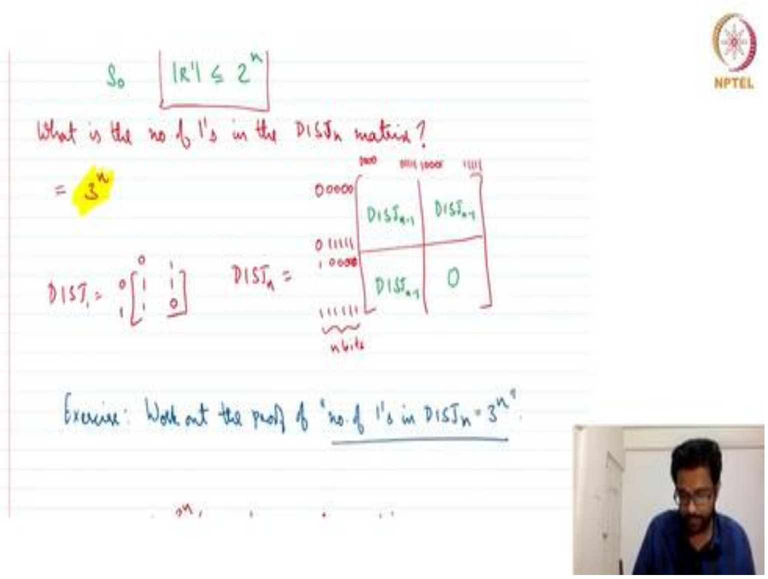
So, 2 power u A, size of u A multiplied by 2 power size of u B and the product of that is 2 power size of u A + u B, which is what we have here. So, R prime is of this and I already mentioned that u A intersection u B is empty set, I mentioned it here. So, we have that u A intersection u B is empty set. So, what is the size of u A + size of u B? Size of u A + size of u B has to be at most n, because there are only n elements in the universe.

So, the u A and u B are disjoint, so each element has to go into only one of them. If size of u A + size of u B is greater than n, that means there is some common element, because there are only total n elements. If there are common elements, then they cannot be disjoint. So, size of u A + size of u B is at most n. So, size of R prime is 2 power the size of u A + size of u B which is at most 2 power n and this is what we wanted to show.

Size of R prime is at most 2 power n and hence the size of R is also at most 2 power n. So, we took an arbitrary rectangle and showed that it is of size at most 2 power n. Now, to get a bound on communication complexity, we need to show that the number of rectangles is at least something. So, we showed that the size of one monochromatic rectangle is at most something. But that is not quite enough because what if the number of ones itself is very small.

If the number of ones itself is like 2 power n or something, then maybe if a constant number of R, R is rectangles are enough to cover them, so that does not give us a bond.

(Refer Slide Time: 23:30)



So, we have to also show further that the disjointness matrix has a lot of ones. So, this number of ones in the disjointness matrix actually 3 power n. How do we see this? So, one way to see this is that so look at the matrix, so maybe we will write a small matrix, it is a 2 by 2 matrix or maybe I just write something in general. So, 000000 right up to 011111, 100000 up to 111111 similarly column 00000, 01111, 10000 and 11111.

So, I can divide it into quadrants this matrix, now let us see what, maybe it is simpler to, let us say this is the matrix for n bit inputs, so this is the n bits. So, let us look at the quadrants, so let us see the entries here. So, what are the entries over here the bottom quadrant? So, the claim is that this is all 0s, because all of this have the leading bit equal to 1, so all of this the leading bit is equal to 1 for the row input as well as the column input.

So, this cannot be disjoint, so the bottom right quadrant is all 0s. Now, let us look at the top left quadrant. So, the leading bit is 0s for both the column entries as well as the row entries. So, this is actually, so what are we doing here, let me call it the disjointness matrix of n. So, what we will have here in the top left is actually the first entry is 0 for both the column and row. So, what we have here is actually nothing but the disjointness matrix for n - 1 bits.

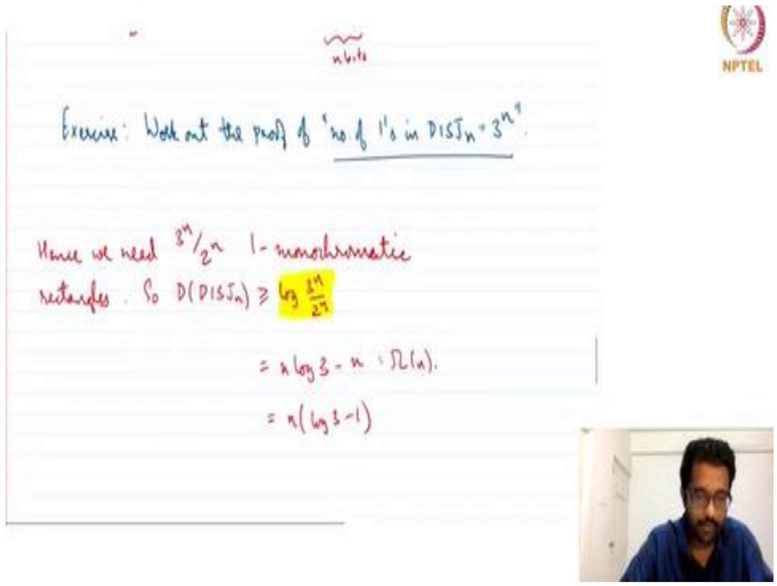
And the top right also is the disjointness matrix for n - 1, because the leading entry is 1 for the column entries but 0 for the row entry. So, the there is no issue with the leading entry, the leading entry does not prevent disjointness and the rest of the matrix is just the disjoint matrix for n - 1. And by the same analogous argument the bottom left is also the disjointness matrix for n – 1, entries. Because, the leading entry for the rows is 1 but the column is 0.

So, that there is no issue there, but the remaining entries correspond to disjoints of n - 1. And so, the number of ones in the disjointness matrix for n is three times the number of ones in the disjointness matrix for n - 1, simply because there are three copies of the disjointness matrix. And if you if you look at the disjointness matrix for one, so this is just this disjointness matrix for 1, this is simply 0101, this is 1, this is 1, this is 1, this is 0.

The only way they could be not disjoint is if both the rows and columns have one. So, disjointness of one has three entries, three ones and so disjointness of two will have three times, three 9 ones and so on and so the disjointness of n will have 3 power n ones. So, the number of ones in the disjointness matrix is actually 3 power n. So, I am not writing the proof in full detail, but so as an exercise you can work out the proof of 3 power n ones, proof of number of ones in disjointness matrix, disjoiness n = 3 power n with full details.

I have almost told you the details of this you can write down the steps and convince yourself. So, now we have seen that the number of 1s is at is at least is equal to 3 power n, and we know that each one monochromatic rectangle has at most 2 power n size.

(Refer Slide Time: 28:41)

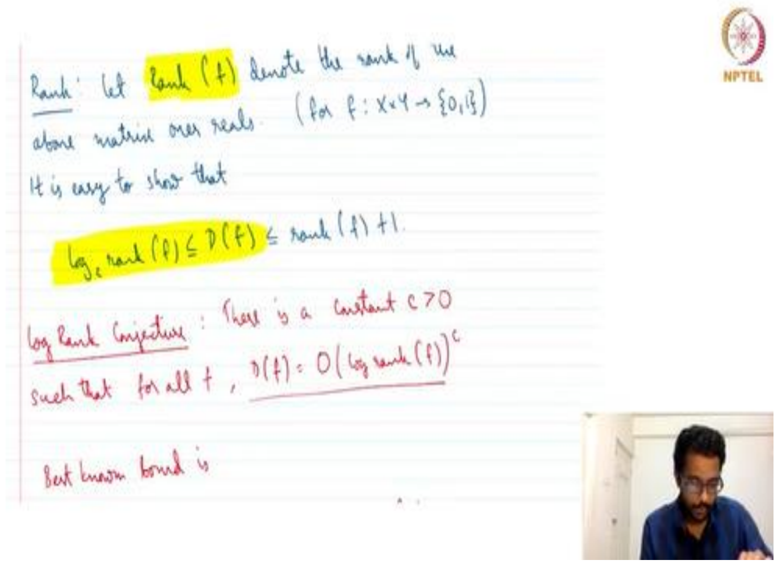


So, the number of one monochromatic rectangles must be at least 3 power n divided by 2 power n. Because each one monochromatic rectangle can only include 2 power n entries, at most 2 power n entries and the total number of ones is 3 power n. So, we need at least 3 power n divided by 2 power n rectangles. So, the number of leaves must be at least 3 power n divided by two power n, so the communication complexity is at least log of that.

Which is; what we have written here it is log of 3 power n divided by 2 power n. So, which if you just work out this n log 3 - n, because log of 3 power n - log of 2 power n. But this is omega n because it is like n multiplied by a constant, so this is simply n into log 3 - 1, so it is omega n. So, this is not exact bound like it is not saying it is at least n, but it is at least some constant times n log 3 - 1 times n.

So, this is another approach to a size finding a lower bound for the deterministic communication complexity.

(Refer Slide Time: 29:56)

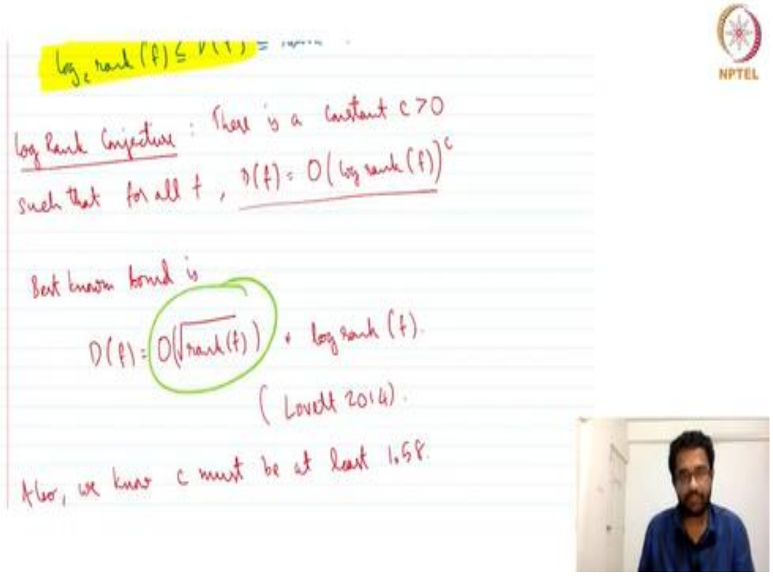


And there is one more thing that I will just mention, so now we have seen what is the matrix that corresponds to a certain function, we call it m of a certain function. And consider 0, 1 function, now for that function let the rank of the matrix be denoted by the, so rank of f, that the word rank of f, let us use this to denote the rank of the matrix of the above function over reals, so you can compute rank over different fields, so to consider the rank of the above matrix.

So, the matrix is defined by the function over real numbers over the real field. And you can see that the deterministic communication complexity, so the important part I will highlight is at least log rank of f, log to the base two as always. So, you can see the deterministic communication complexity is at least log rank of f, again it is not very difficult to see this, it is a rather simple proof but since we have limited time, I do not want to elaborate that.

If you are interested, I can point to references. So, deterministic communication complexity of the function is at least the log rank of f, and you can also show that it is at most the rank of f + 1, and this is also another way to get lower bounds.

(Refer Slide Time: 31:40)



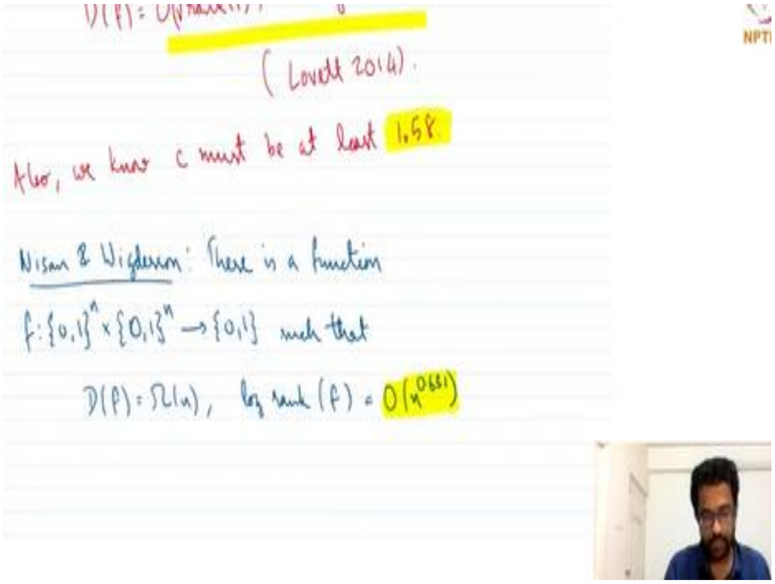
And this led to something called the log rank conjecture. So, which it is a very popular or in the area of communication complexity this is a very famous question, open question. So, the conjecture is that for any function f, there is a constant C such that the deterministic communication complexity is at most log rank of f to the power of this constant. So, this above the thing that we; wrote about the one that I have highlighted.

It says it lies between log rank and rank. So, rank is much, much more than log rank, it is exponential in log rank. What we are saying is that? The deterministic communication complexity is at most polynomial or a constant power of the log rank, not exponential. It is polynomial in the log rank is what this is I saying. And, this question has been open for at least maybe two three decades now.

And now it is still unknown and the best known bound is something like this and that is by (()) (32:50) Lovett. The deterministic function communication complex series order square root f, a square root of the rank of f multiplied by the log rank of f. So, this quantity is there, this the square root rank of f. This this makes it still exponential in the log rank of f. So, it is still not known.

And one thing that is known is that we know of a function, where the deterministic communication complexity is n and log rank of f is at most n power 0.631.

(Refer Slide Time: 33:27)



So, log rank is at most n power 0.631 and deterministic communication complexity is at least n. So, this means that the C, if deterministic community the log rank conjecture is indeed true. The C should be at least like enough to power this, so that it becomes at least one, so the value turns out to be 1.58. So, we know that if log rank conjecture is true, then C must be at least 1.58. There have been some series of works.

But the best known bound is what I stated about by Lovett, which is this and we know that C must be at least 1.58 and that is what is known, and this this function is again it is not very difficult to work out. If you are interested, I will give you the details of this, that this function it is log rank is at least this much. This was this function was discovered by Nisan and Wigderson. So, with that I think I will conclude this particular lecture.

So, we saw the fooling set argument and we saw the size bound and I mentioned the rank and the rank mount and also stated the log rank conjecture. And again, there are other techniques, other variants of communication complexity such as randomized communication complexity, non-deterministic communication complexity etcetera. But which, we will not have time to get into. In the next lecture we will see an application of communication complexity into the rest of complexity theory. Then I will stop this lecture, thank you.