Linear Programming and its Applications to Computer Science

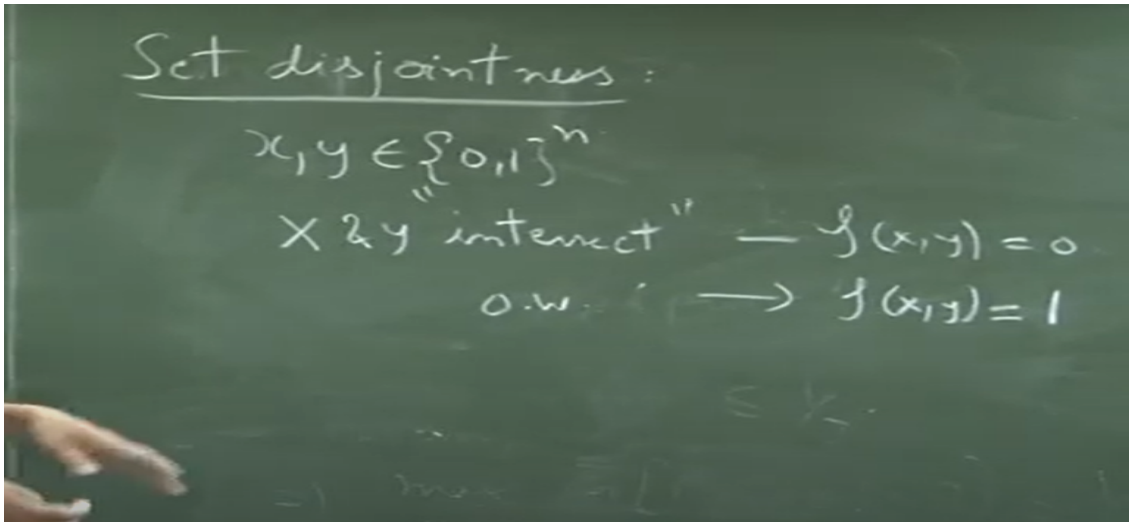
Prof. Rajat Mittal
Department of Computer Science and Engineering
Indian Institute Of Technology, Kanpur

Lecture – 38
Set Disjointness Problem

And I would not prove a big thing here, but at least I want to give you an idea of how these distributions are kind of found or what is the intention or how to create such distributions. So, remember the set disjointness problem, I can think of it as 2 strings in 0 1 to the power n. And if x and y intersect then f x, y is 0 otherwise right this string is giving me a subset of 1 to n right. And then if x and y intersect if the sets intersect they are not disjoint this is 0 if they intersect then f of x y is equal to 1.

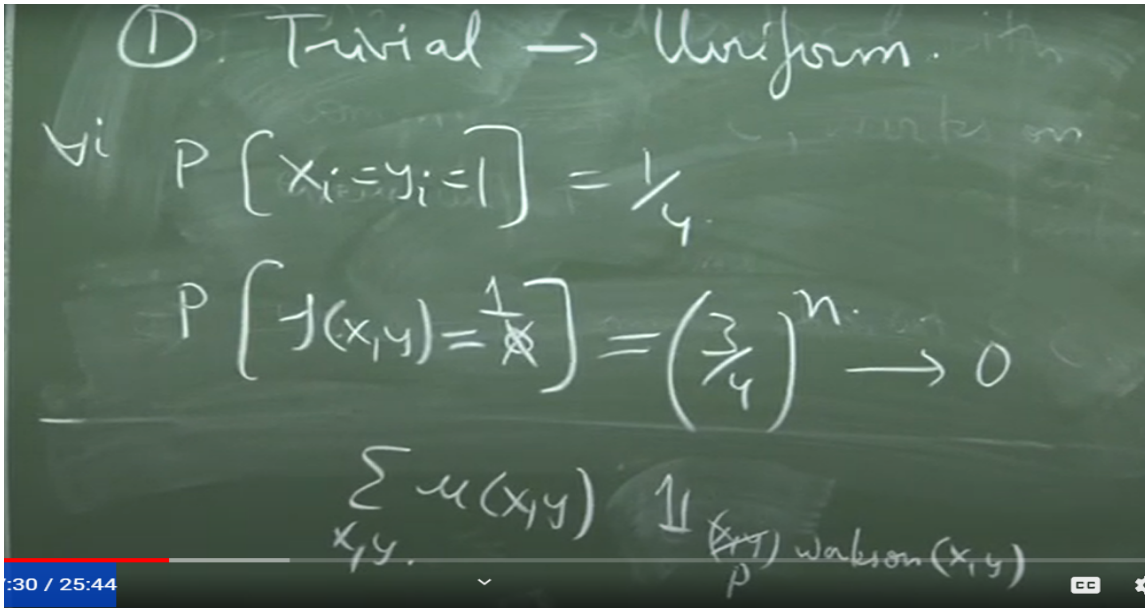
Now, for this kind of a problem this is a very important problem randomized communication complexity I told you things in data structures, data streaming multiple ways this communication problem comes out. Its lower bound is very important what do you think the lower bound should be n it turns out yes.

So, for deterministic protocol still you can think of n right, but proving it for randomized protocol seems very hard right. Even randomized I cannot do anything better than that it turns out to be n, but it took a long time. And I will just tell you what is the hard distribution for it the proof of it is difficult I have the reference in the notes good. So, I want to create a hard distribution for this.



Now, given any problem any function what is the trivial most distribution not f distribution uniform distribution right. So, let us start with trivial what if mu is uniform that is what I am trying to with this example I am going to trying to use some intuition.

So, hard distribution is going to be for which this quantity is bigger and r k is going to be n. So, I am going to say that for that distribution any deterministic protocol which has less than you know some c times n communication. I am going to fail with probability at least some constant 1 by 3 1 by 100 any constant is fine right because constants are for free right it is not.



No, no, no, but on a distribution there is a probability right now the probability that is the source of randomness is the distribution right. So, now, if I have a uniform distribution do you think this is big. So, this is you do not have to worry about linear programming anything this is like fun puzzle time right. So, let me ask this question suppose I pick mu x y uniformly at random what is the probability that f x y is equal to 0 or what is the probability that f x y equal to 1.

Sorry close by, but so what is the probability that at index 1 they intersect. 1 by 4. 1 by 4 they intersect means x i equal to 1 as well as y i equal to 1 right. So, for all i probability x i equal to y i was half, but I want 1 remember intersection means 1 this is 1 by 4 right. So, what is the probability that f x y equal to 0 that means.

. Sorry I should say 1 that is easy right 1 minus and we know what 1 minus 1 by 4 is right. So, right and in general in all these cases n is. So, what is the algorithm which will succeed here always answer 0 and you will succeed with probability close to 1 no communication with 0 communication just answering 0 will make this work more like. So, clearly this distribution is not good enough right. So, I want a distribution for which f x y equal to 0 with constant probability f x y equal to 1 with some constant probability otherwise I have had this 0 error sorry 0 communication cost protocol.

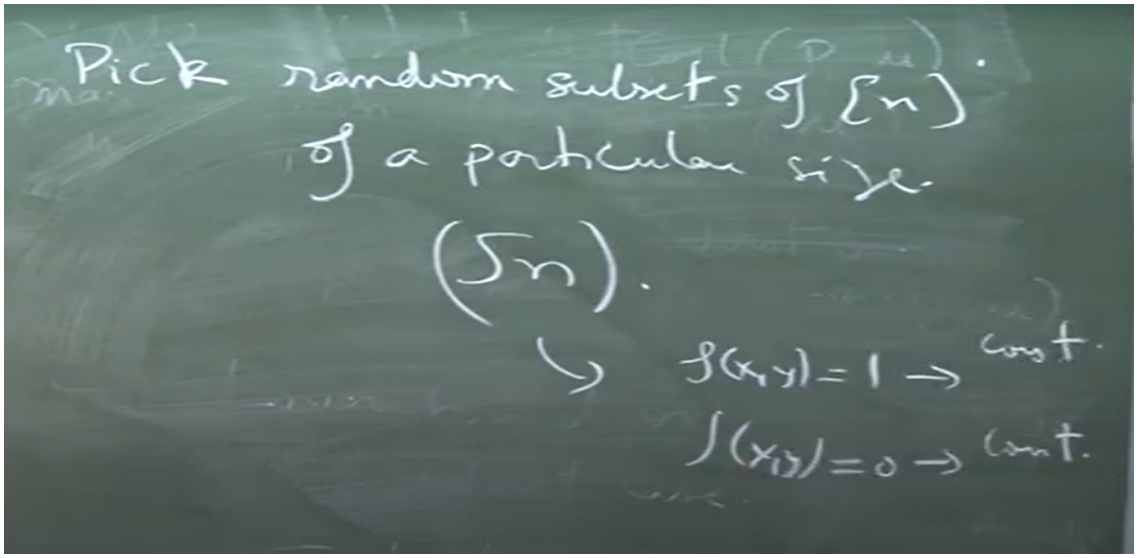
So, is it clear remember what is the probability what is the probability summation over

mu x y not probability sorry not x y works on p p works on x y right this is my success probability. Now, if mu x y is 0 sorry f x y is equal to 0 for almost all of the weight here I can just keep answering 0 then this will be very close to 1 think about this is this is like. So, notice if f x y is equal to 0 for most of the mass this is my algorithm just flatly answer 0 do not think about. So, actually it does not matter for any of these if there is a non constant like if it overwhelming proper probability over these inputs if I am going to have 1 as the answer why worry then just answer 1 it does not matter what is the input and I will succeed with high probability right. So, now that means if we selected things at random correct then there is going to be no intersection sorry there is going to be intersection with high probability right 1 idea could be let us take smaller subsets right.

If I had suppose I had restricted myself to taking random subsets of size 1 then what will happen with high probability they will be disjoint right if I take random subsets of size n then they are highly intersecting they are with high probability intersecting if I take it with 0 then sorry with size 1 then they are mostly disjoint let us look at something in between can you guess what is what at what set size. So, the idea is pick random subsets of n of a particular size no actually this is if you do this is kind of birthday paradox if you remember not exactly birthday paradox, but the calculations will be similar if you have n values you look at 2 10 values 2 of them will coincide, but again not exactly that, but this you can calculate yourself right if you take 2 random subsets of size root n what is the probability that they intersect this is just playing with binomial coefficients correct. And this will show you that f x y equal to 1 with constant probability with constant probability what do you think will this distribution work look again my face you might guess no right. So, I give you a hint no it would not work. So, give me a protocol which will work in this case which is of course, much less than n and again this is common sense mathematics yes very quick almost correct answer is it root n bits which bits are 1 how do I know, but that will take n bits though I do not have n bits right do I have n bits I do not right yes that is true I have a n bit communication complexity deterministic not even randomized exactly there is a protocol here of root n log n right how many what length do I need to specify an index in size 1 to n if I want to specify an index in between 1 to n what length do I need log n right I need log n bits to represent a number between 1 to n I need to represent how many numbers root n.

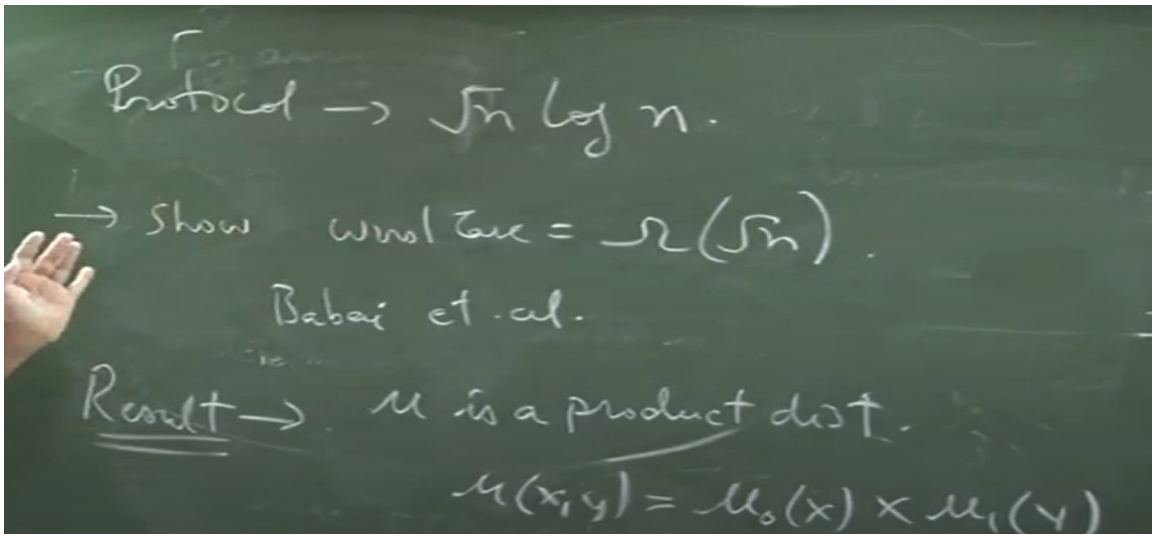
So, I will just send all the indices. So, L S can send all the indices which are there in my subset Bob can check easily is there a right. So, there is a root n log n protocol for this might seems stupid, but this is actually a nice hard distribution the first bound that set this renders what hard for a randomized protocol actually came from this distribution you can use this distribution the same distribution and show that worst case complexity. Again this is the distribution you will show that for any deterministic protocol you will require at least root n bits of communication or if you fix the communication to root n your error

probability will be more than some constant probably 1 by 3 sounds good. So, even though

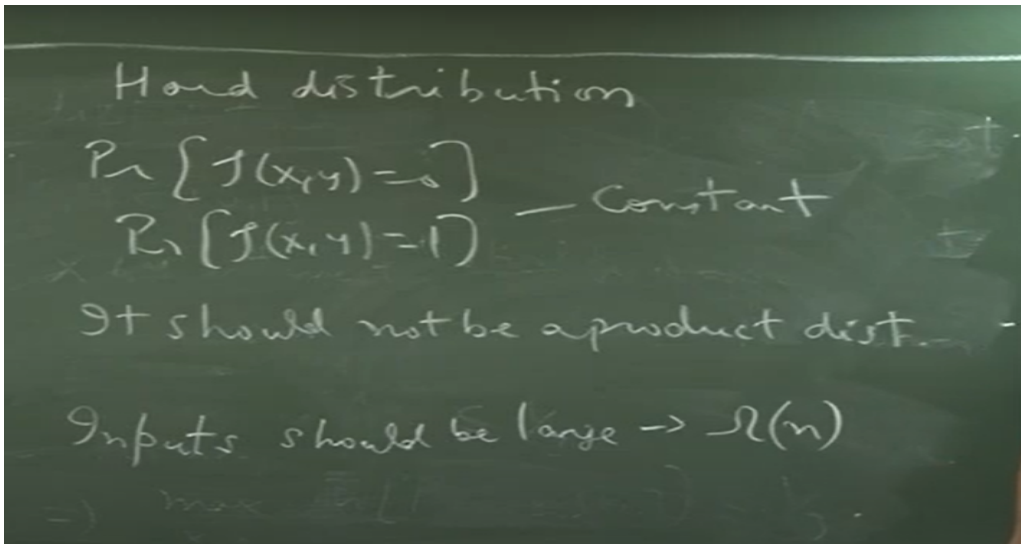


this might look like a simple distribution, but this works at least for proving some amount of some amount of amount and notice that this is not trivial right if I tell you to prove that you need at least root n amount of communication a randomized protocol even there is no intuition also right randomized protocol could probably be doing it in constant I do not know some exchange some bits and figure it out.

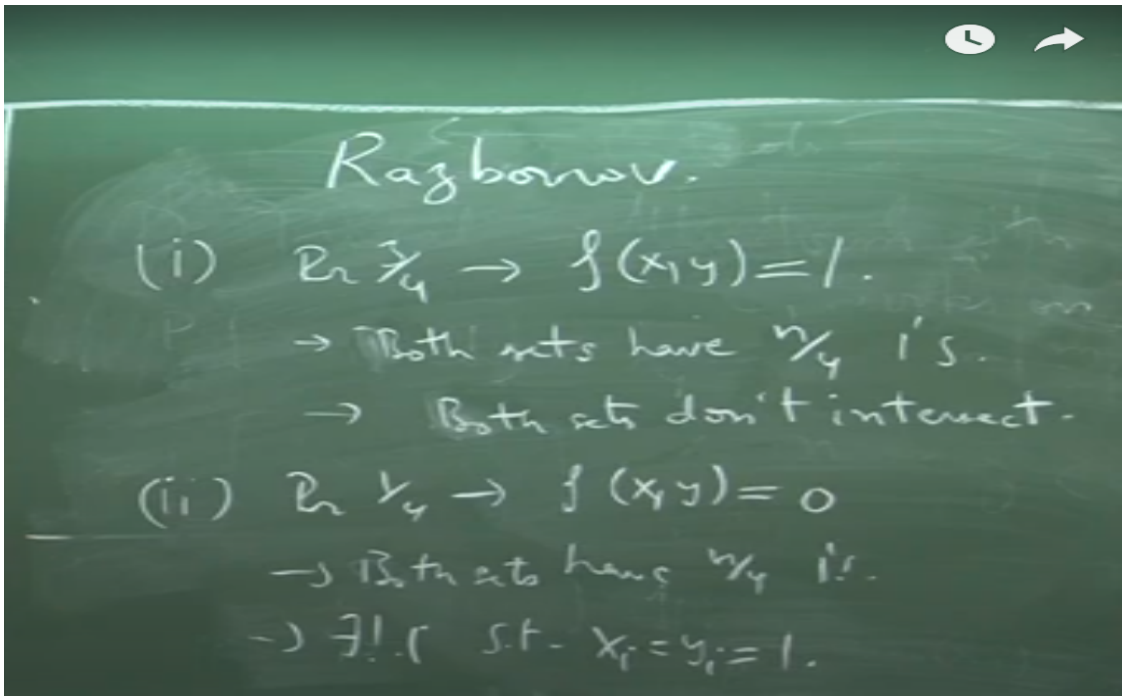
So, this was the first one this was I think by I do not know if you have heard Babai et al ok if I remember correctly, but in any case they could actually show something stronger. So, in this case what happens is the way I am picking x and the way I am picking y is independent I am picking a root n size subset from x and a root n size subset from sorry a root n size subset from n for x a root n size subset from n for y there is no relation between how x and y are picked. So, I am not going to specify this, but there is a result which says that if your mu is a product distribution what is a product distribution the probability of x y is equal to some probability of x probability of y. So, you take a probability distribution over just x is you take a probability distribution over just y is and then multiply that is the probability of x y. If you take any kind of product distribution they could still give root n size communication protocol and there is a result requires lot of math it is not obvious, but here they are.



So once you have this all these things this is how you know what to do for hard distribution right. One thing is for my hard distribution one thing is probability f x y equal to 0 this should be some constant if this is very large this is very large then I am already calculating. So, my hard distribution the way I choose it should be like this second thing it should not x and y should be dependent it should not be a product distribution right x and y should correlate anything else which you saw from all this discussion inputs should be large right. If most of the time my set has size root n or n to the power 0.9999 still I am done I can just set the complete set correct.

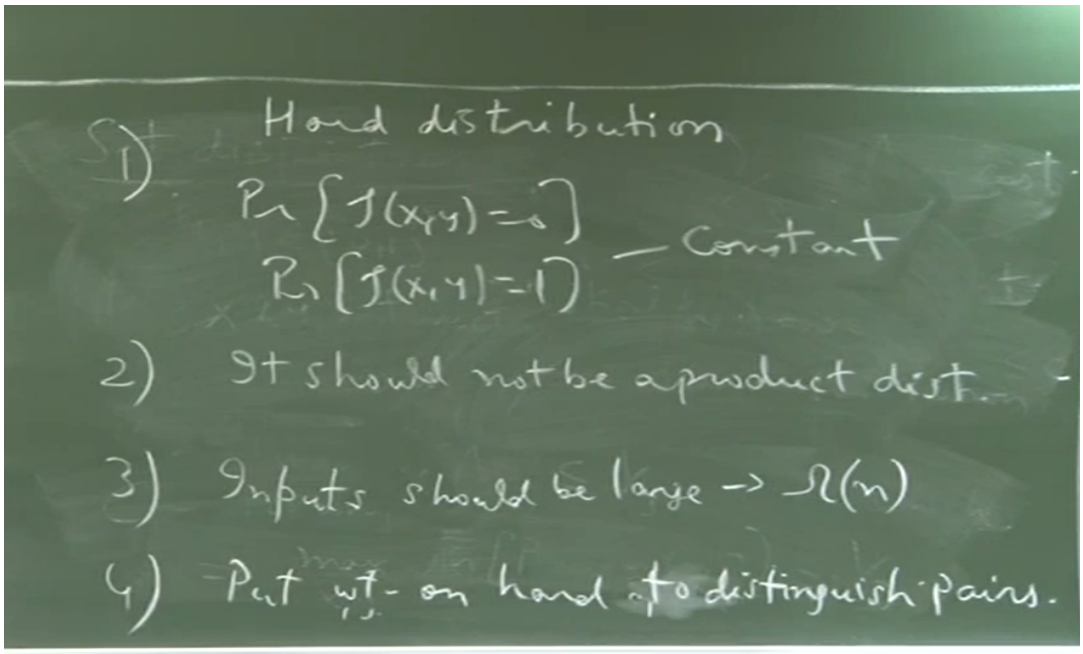


So, most of the time the set the size of my set should be big it should be around n right and then intuitively I should be putting weight on hard to distinguish inputs right like x and y should be very close in some sense. And the actual distribution actually there was initially there was a proof by Kalyan Sundaram and one more guy that was the first proof, but was very complicated probability distribution then Rasborov simplified in 92.



And this will tell you the distribution here the distribution here was they said with probability 3 by 4 I will say this is equal to 1. So, this ensures that and the probability 1 by 4 I will ensure that f x y equal to 0. So, both of them are with constant probability and now in this case both sets have n by 4 functions because I want side to be omega n right n by 4 is omega n such that right.

And by writing both sets do not intersect here I have related x and y it is not like I am picking x and y independently if x has 1 then y does not have 1 for sure right. So, I am correlated. So, notice how I am using all the 3 conditions here. Secondly with probability 1 by 4 I will enforce f x y equal to 0 both sets have n by 4 once there exist a unique I this is just saying there is a unique I such that again I have related this and I have created these things which are very close to each other that is the important point. So, in some sense I my distribution needed to have 4 properties a hard to distribution pair.



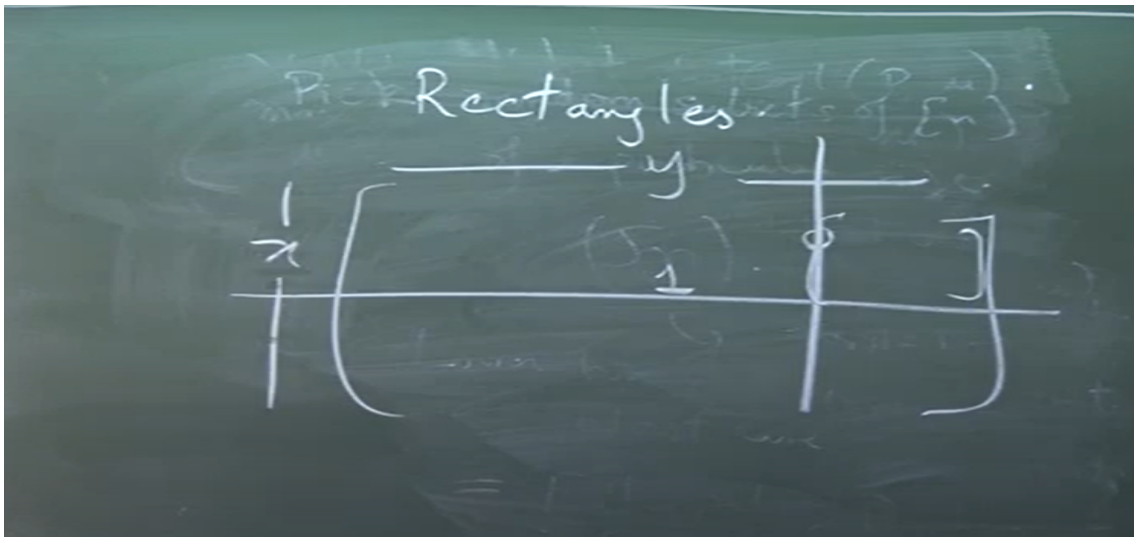
So, I wanted to put weight on pairs which are almost disjoint or intersecting at very few places right. So, that is why this kind of works, but proving that still requires lot of work. And this is just one example communication complexity query complexity is filled up with these cases where you take where you come construct a hard distribution to prove some. So, this is very communication complexity specific idea.

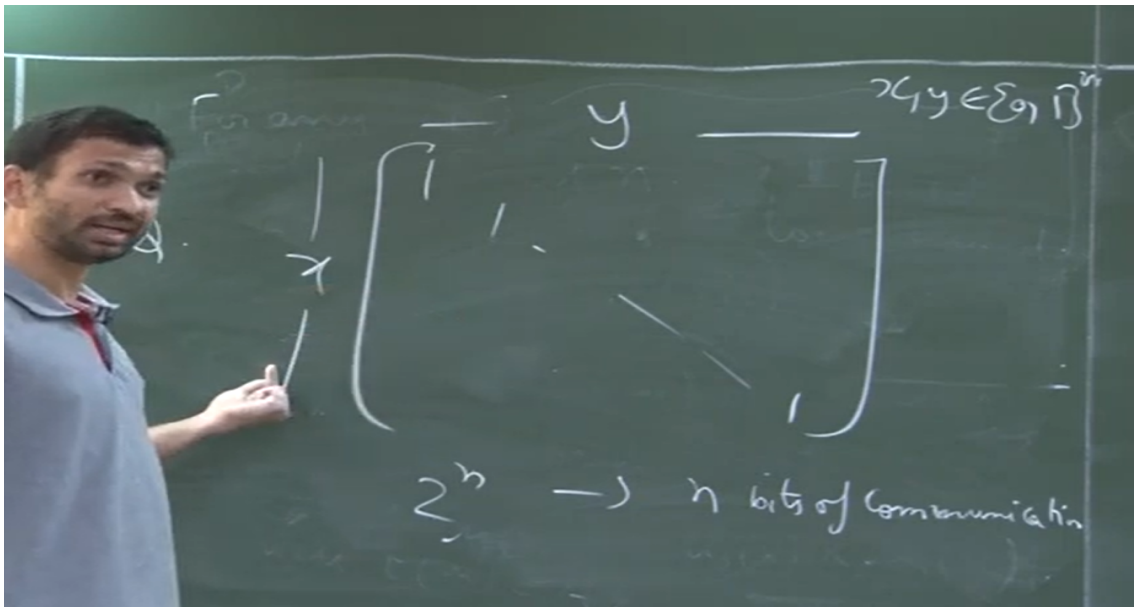
So, what happens is. So, the key thing is rectangles what do I mean by this look at this communication matrix all inputs of Alice are here all inputs of Bob are here right. And now I have 1 somewhere 0 somewhere depending upon the function output right this is called the communication matrix for the problem right. Now, what happens is in your communication when your communication happens let us say Alice transmits the first bit whoever is transmitting the first bit let us call it Alice she either transfers 1 or she either transfers 0. So, in these inputs are kind of divided into half I am talking about I have divided into half I am saying that these are you understand commutatorial rectangles these are not like all these are 1 all these are 0 some of them are 1 some of them are 0. And I can rearrange my rows because there is no ordering on the inputs and I can say that on these inputs Alice is going to answer 1 and on Bob's input on these inputs Alice is going to answer 0 correct then let us say it is Bob's term like this right.

So, a communication protocol gives rise to a collection of rectangles right. And what do we know about these rectangles the final rectangle once my protocol is finished my rectangle is either all 1's or all 0's because after the my communication is finished for all the inputs I have the single value. So, now I am also learning about communication complexity I can show off right. So, let us think of equality function equality function is you answer 1 if x and y are equal 0 otherwise what is the matrix for it right. How many

rectangles do I need to cover it 2 to the n notice the number of inputs are 2 to the n right x and y are coming from 0 1 to the n.

So, the size is 2 to the n by 2 to the n right and no 2 diagonals can be in a rectangle right because if I take this and this there is a 0. So, to cover it using what we call monochromatic rectangles either 1 or 0 monochromatic rectangles I need at least 2 to the power n every communication bit increases the number of rectangles by 2 after 1 bit of communication I got 2 rectangles after 2 bits I got 2 rectangles I got 4 rectangles if I need 2 to the power n rectangles I need n bits of communication. So, this is a exact proof that any deterministic protocol for equality will take at least n right. Now, when you do it for the deterministic protocol with this you want rectangles which are almost monochromatic and cover. So, what you want to show is that under this distribution now right now you have a distribution in mind and then you when I say almost monochromatic means if I am answering 1 on this the weight according to the hard distribution the ones is much more than the weight on 0.





Such rectangles cannot be you cannot have large such rectangles. So, I need lot of such rectangles to cover my communication matrix that means I should have communicated lot of bits that is the idea, but this requires. So, instead of exact monochromatic rectangles I want almost monochromatic rectangles and I would show that to cover the communication matrix. Now, good thing is now I have deterministic protocol. So, I know that in c bits of communication I will get 2 to the power c many rectangles.

So, I just show that almost monochromatic rectangles still you need lot of almost monochromatic rectangles to cover the communication matrix of set disjointness not equality that is the idea. That finishes today's lecture.