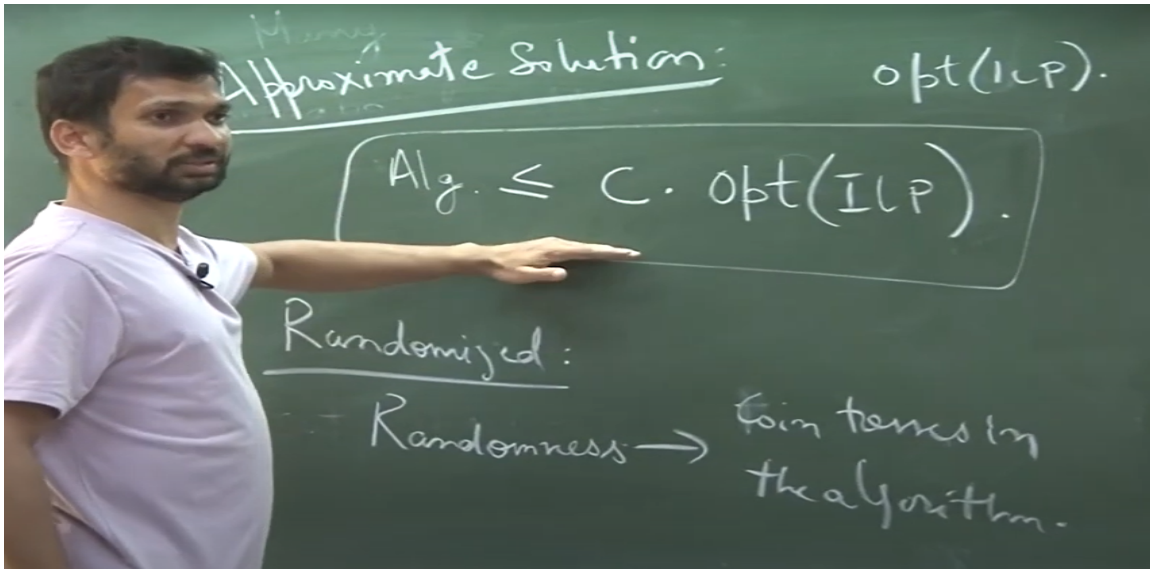
Linear Programming and its Applications to Computer Science

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

Lecture – 45
Rounding for Set Cover

Is giving an approximate solution right, what does this mean? Right. So, suppose I have algorithm value correct and I have the optimal of ILP, this is the actual value which we wanted, but we cannot hope for this, but you want to make sure that our algorithm value is not too far, too far in which direction not too large, it is a minimization problem that is what I want to emphasize here right. Since, it is a see you cannot get a smaller value, I am not talking about the LP value, LP value will be large, LP value will be smaller, but the final solution, final discrete solution which your algorithm will give cannot be smaller than the optimal. So, what you want is that your algorithm is not bigger than C times the optimal right. Now, C could be a constant, it could be a function of N or let us say M or something, but we want to optimize this. A better approximation algorithm is where the C is smaller, make sense and even the now for a complexity theory scientist even this is a game when you can show that for any constant C this problem is still NP hard.

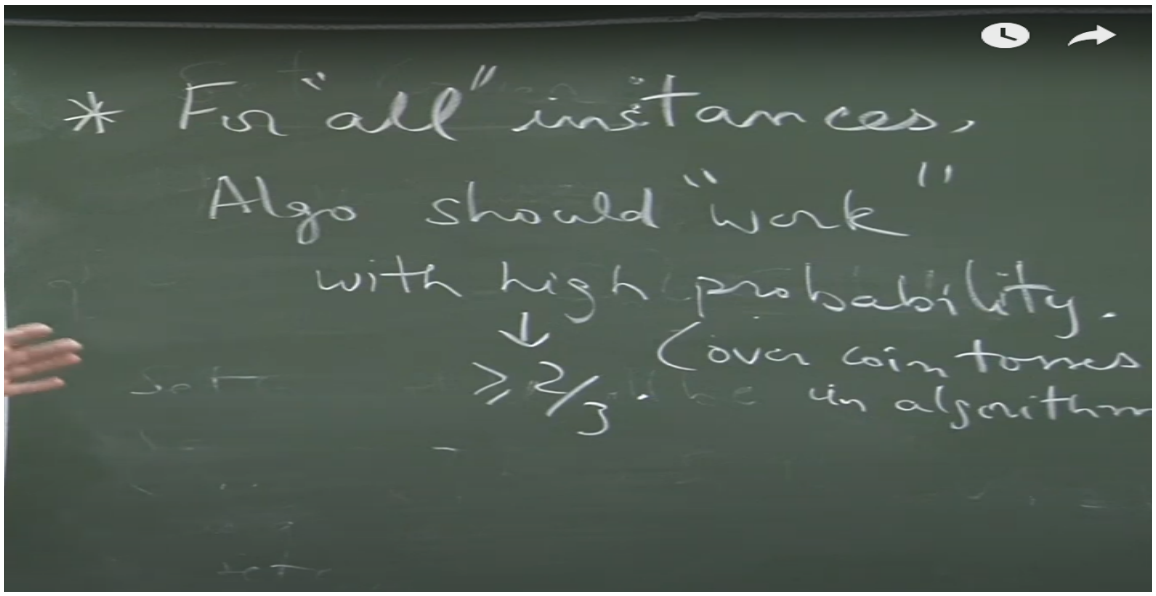
Even that is this is called inapproximability, study of inapproximability people have got a Godel price and many things on this. So, not just that your exact problem is hard, but even showing that approximating up to some factor is still very hard right. That is still a big, big open if you work in complexity theory you might get into that is called inapproximability. So, approximately approximation algorithms inapproximability.



So, now this is our target for some nice that is what we want to, but now there is a small nugget here which is that our algorithm is going to be randomized algorithm. And thankfully we have seen randomized model in communication complexity. So, if you remember randomization random randomness can come from in your randomness randomized algorithm randomness come from two sources. One is you toss coin in the algorithm and second the randomness is because of the input. Remember we talked about this Yao's minmax lemma we said worst case complexity average case complexity in case of average case average was over inputs right.

And I talked about the fact that mostly we care only about the worst case. That means the randomness which pertains to us which we will care about is because of the coin tosses in the algorithm. What it means is for every input whatever instance of set cover you pick my algorithm should work with high probability. So, this is the goal. This is important for all instances there is no randomness on the inputs for all instances this high probability is over coin tosses in algorithm.

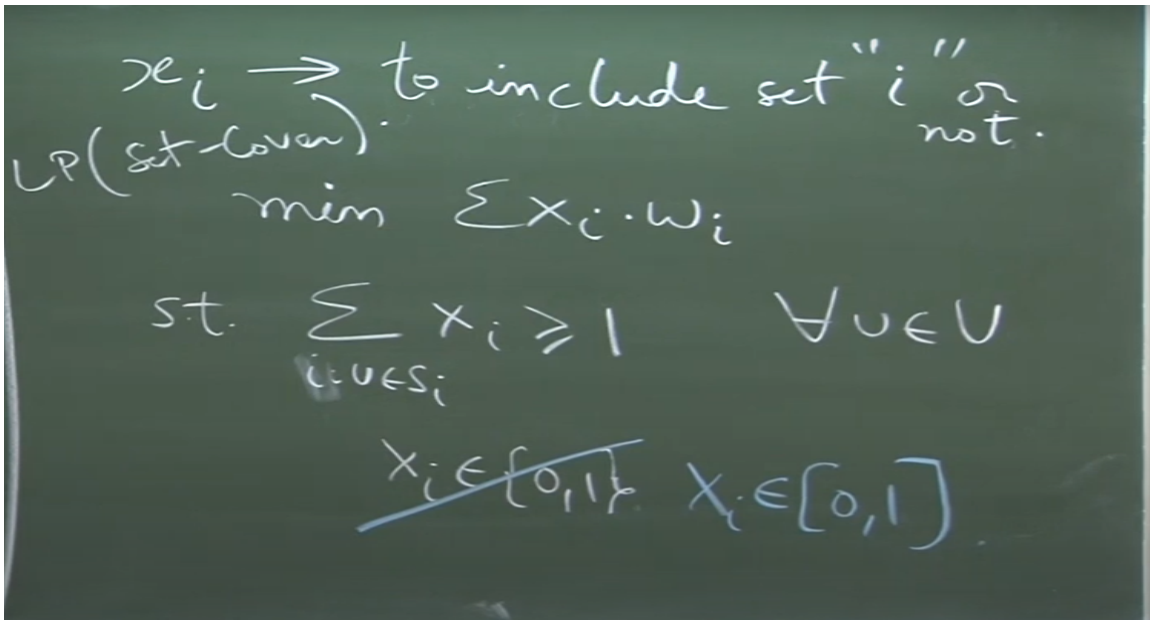
What does it mean work? What is the definition of work for us? Right, but now in our case this is the definition for work right. For a general randomized algorithm it might be saying give the exact answer. But now our definition of work is this right. So, we want our algorithm to give the nice approximate solution most of the time for most of the coin tosses this is our goal. So, now what we want to do is very clear we have a set cover problem nice problem can capture many of the instances.



Can we make sure can we give an algorithm which solves this problem with the which has an approximation factor small approximation factor and works with high probability. And this we do not play around with high probability we say probability is let us say 2 by 3 or something because going from constant to constant is just constant factor. So, we

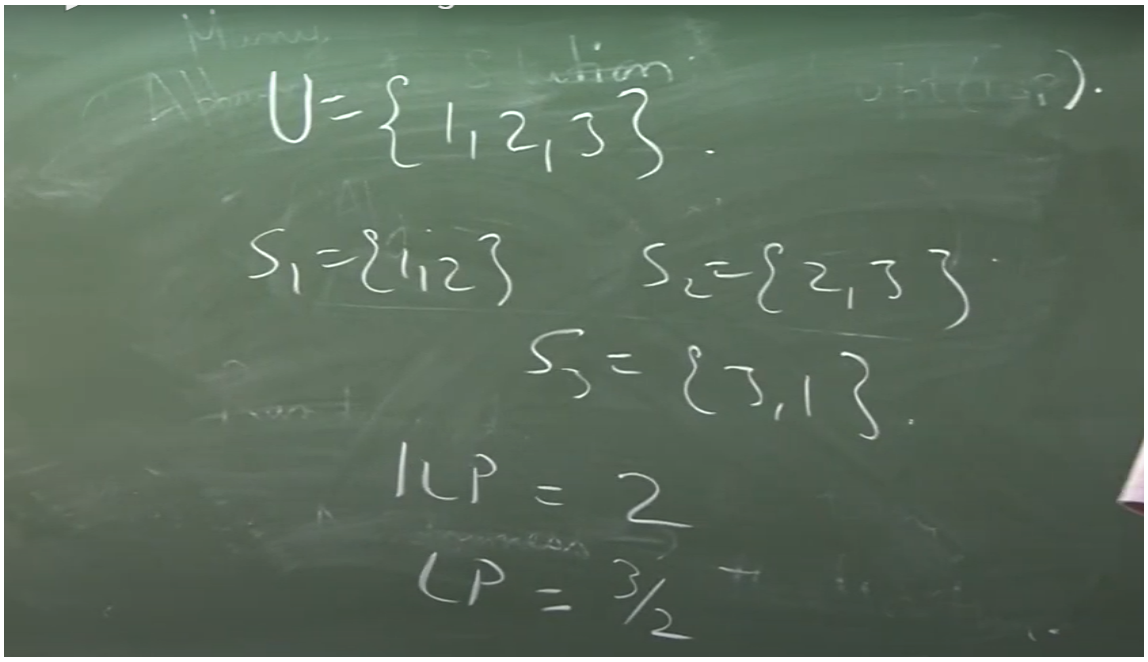
do not care about it. So, this high probability we can say greater than 2 by 3 or you can pick your favorite constant here which is bigger than half. So, you would not worry about that if this point is not clear I can convince you about this after the class we did the same thing in the communication protocols also right.

Great so the task is clear. Now forget about all this let us focus on this. This is what we have this is the LP for set cover and we know that we have algorithms to solve this efficiently ellipsoid interior point we would not worry about how we solve it we have those solutions and finally, we will get a solution in polynomial time. So, for every U there should be at least one set which is covering it this is exactly what this is saying look at all the sets which cover U they should this their sum should be more than 1. That means at least whatever cover you have choose whatever set you sets you have chosen one of the set should be over U otherwise this thing will be 0 right.

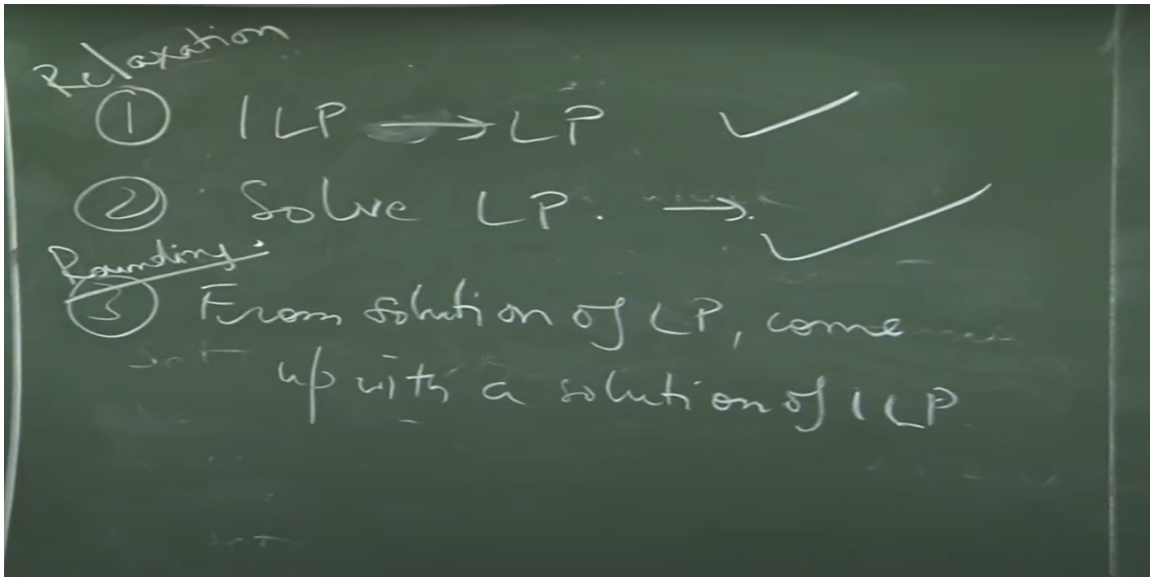


So, now for us the question is this LP close to the integer linear program on set cover. So, can you think of an example when these two values are not equal the concept of you should read about what uni modular matrices are I am not saying if the constraint matrices are integers then the vertices are integers I did not say that. No uni modular is not just saying that the entries are integer no no no that would have been great you would have gotten a million dollars you would not for at least for that idea. So, it is you can if you want you can read about it, but this is definitely this is not going to happen. So, my question is now can you create a simple example of set cover where your fractional solution right the solution here is going to be a fractional solution.

What problems should you choose so that the fractional solution is better than the actual integer solution.

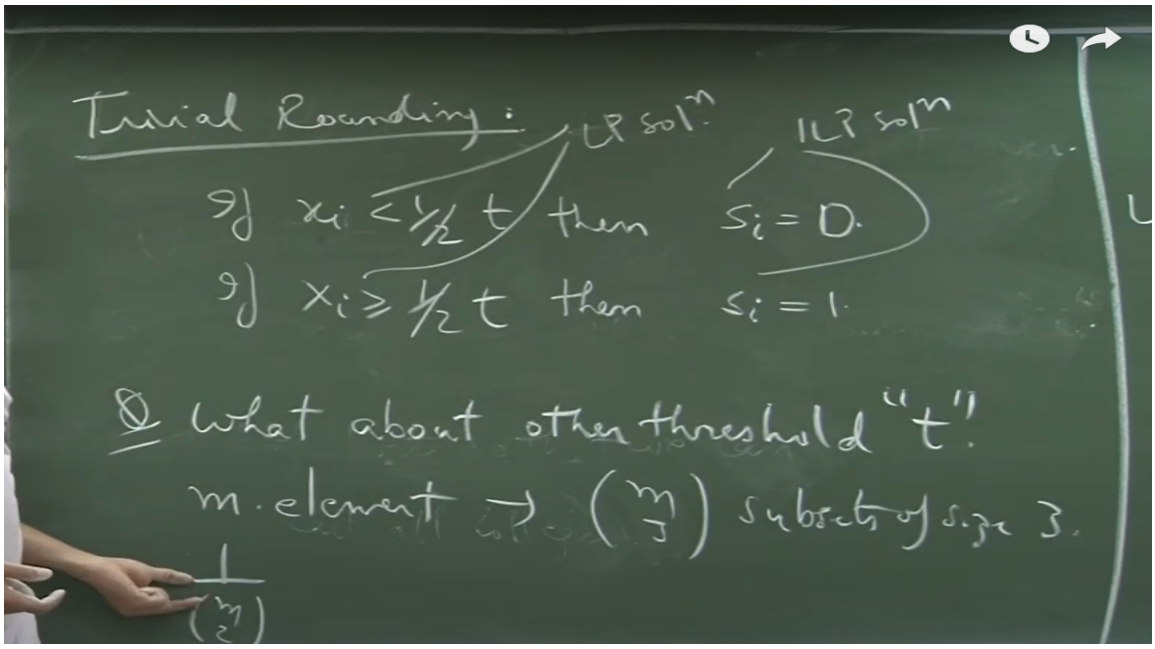


It is not hard, sorry I am not even asking you described in general right just give me a simple example small example what is 1 2 2 3 3 1 good. This is set cover instance what is the value of ILP I can assign half to each of these and then suddenly my value is smaller right. So, this is not like min cut this could be away from this solution right good. So, then what this means is when I solve this linear program it will happen many a times that my values will be fractions it would not be integers and now I want to convert them into integers right. So, what is my strategy I do not know whether it works or not, but my strategy is convert ILP into LP but the good thing is even though these are three steps we have already done this we have talked about.

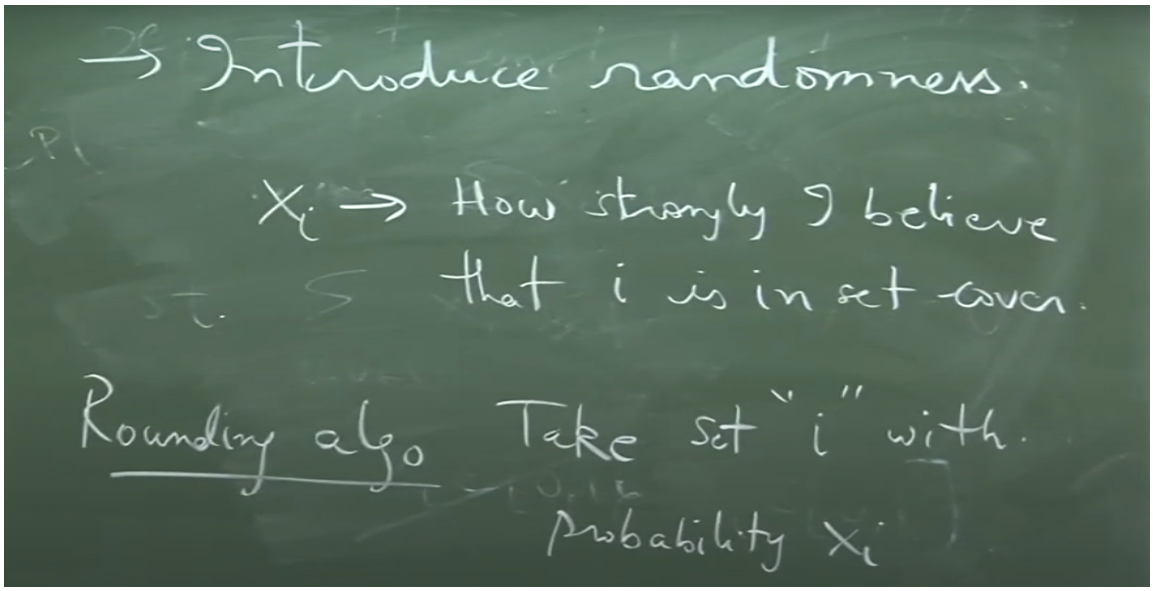


So, done so this was relaxation what we want to do now is rounding correct. So, now, suppose someone gives you this 3 by 2, 3 by 2 what to do sorry 1 by 2, 1 by 2 value on

these three sets how do we decide which sets to keep which sets not to keep in my rounded solution how do we round generally fractions right. So, that the trivial rounding in this case would be. So, this is the value of my LP right and my integer solution call it small s i is equal to 1 this is the ILP solution this is how I can convert LP solution to ILP solution there is one good thing about this solution. So, remember what do we want to do we want to make sure that what we get is a set cover and second thing is that the value of my solution is not far away from my LP solution.



So, now, if you convert it like this the good thing is one of these things is satisfied which is it is not far it cannot most be twice what about the other quantity might simply be empty right what about some other threshold t probably half is not good why right. So, suppose I have M elements take all possible M choose 3 subsets of size 3 correct I will need only M by M by 3 or something right, but what I can do is I can assign every set 1 by M choose 2 and then as I increase M this becomes smaller than any threshold and when you round you will not get any solution at all make sense this example again the set cover instance is clear take M size M size universe take all possible 3 size subsets there are M choose 3 many subsets I am giving you a solution which has value higher than M which is just assign everything 1 by M choose 2 sorry the value is lower than M. So, it is better than optimal, but if you are given this solution your threshold your this rounding scheme will be very bad on it will not pick any set given a threshold I can pick an M. So, that this instance becomes really bad for this rounding when I describe this problem I also described approximation algorithm and. So, there is a reason for that means we should yes we should introduce randomness.



So, this is another kind of a technique to another thing to learn I told you when you look at a problem try to see if you can convert into linear program if you are creating an algorithm you cannot try to introduce randomness. Now, there are courses offered in this building whether this randomness is actually necessary or not if you want to learn that you have to sit on those courses not in this, but at least what we have seen is that this allows us to make simple nice algorithms for sure. So, this is this introducing randomness is kind of powerful technique to create nice algorithms and this is the thing which will help us here too. So, notice xi in some sense if it was 0 or 1 it was an indicator if it is not 0 or 1 it is telling me somehow how strongly I believe that i is in set cover sorry i should be included or not right this is the intuitive feeling from the linear program. So, what could be the natural rounding take it with probability exactly right.

So, my rounding algorithm is take set i with probability xi no independently every set i keep it in with probability xi, I do not keep it in with probability 1 minus xi. So, in a randomized algorithm in a randomized trial I might not get any set at all that is also a possibility, but I do not worry about it because chances of that happening is very small right. So, this is a very good point there could have been two ways if I wanted to pick just one set out of all of them then as you said I would have normalized xi summation xi equal to 1 and then pick the set with the xi probability, but I am not going to do this here what I want to do is pick multiple sets.