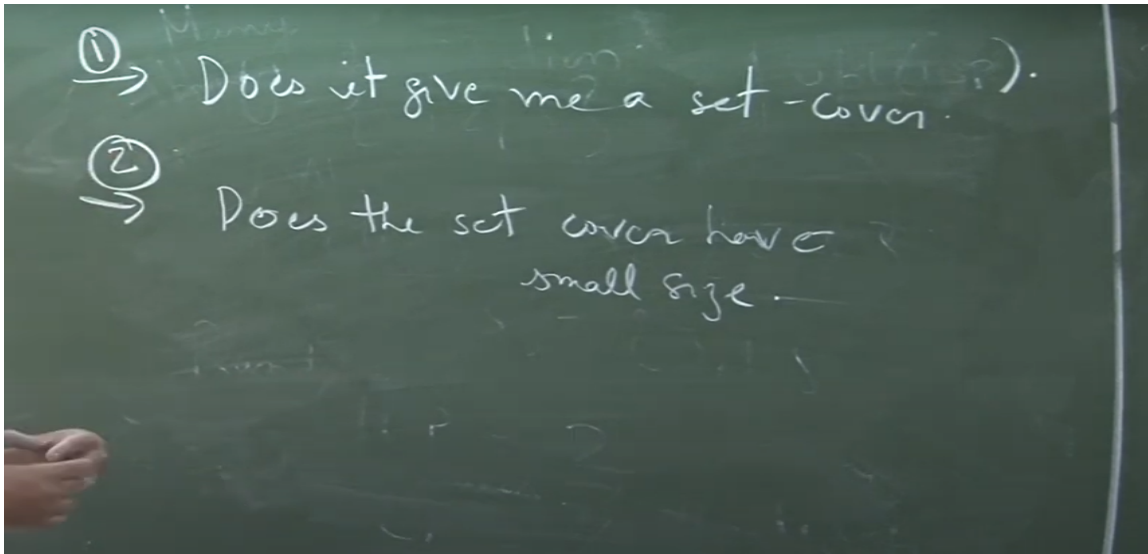


Linear Programming and its Applications to Computer Science
Prof. Rajat Mittal
Department of Computer Science and Engineering
Indian Institute Of Technology, Kanpur

Lecture – 46
Analysis of Rounding

And now once again there are two questions. What is the thing which I want to ensure? So, I should get a set cover, my set should my solution should be feasible and the set cover should have small size, right. Simply put for my ILP, do I have feasibility, do I have nice objective value. This is what I want to figure out. So, which one do you think is going to be easy? How about for 2? Ok. So, this was a trick question both of them are going to be hard.



So, I win all the money, both of you, all of you lose, ok. What I mean to say is again both of them are not very difficult, both of them are not very easy. There is going to be error or error in probability. There is going to be probability mass when this is going to be a problem, there is going to be a probability mass when this is going to be a problem.

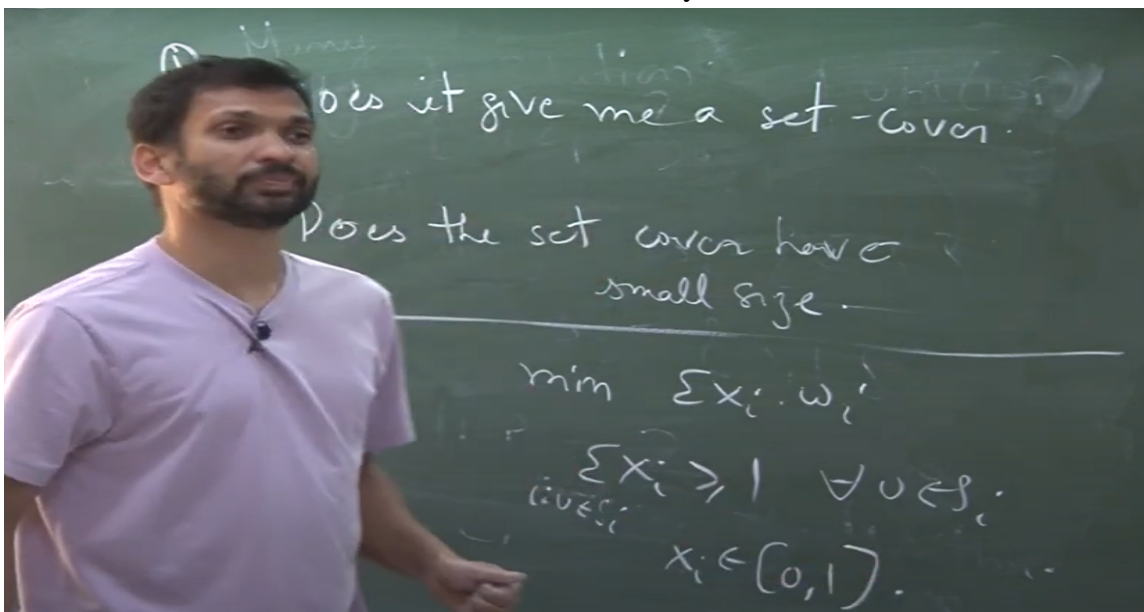
But those error probabilities are small in both the cases, right. Then by what bound can I say that I am done. If this also has a small probability of happening, this also has a small probability of happening. I can say none of this occurs with very good union bound and people have taken probability course with me should know this, right. So with union bound I can say that none of this will happen.

If this happens with probability 1 by 100, this happens with probability 1 by 100, then none of this happens with probability 1 minus 2 by 100. Correct? Is this hard to believe? This is fine right. So, in this case I am not using anything. Are they independent? Are

they whatever? The worst case is this can happen plus this can happen. I can remove all those bad cases and still I am good.

What I am going to do is show the two and then I assume you are convinced that it is fine with the entire case, ok. So let us start with 1, ok. What is the probability that U is not covered? Some element U in U is not covered, right. So, U is not covered if none of the sets which cover U are picked. What is the probability that x_i is not picked? $1 - x_i$.

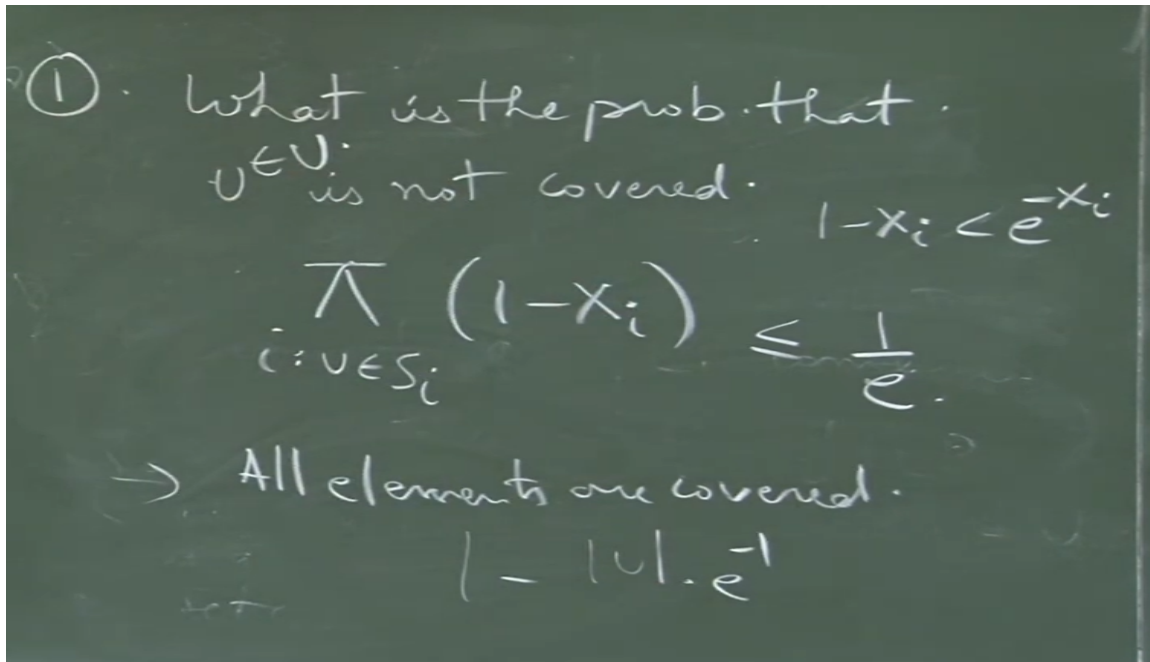
What is the probability that none of the set is covered? Multiplication of that again because we are doing this experiment independently. And this almost looks like what I have a rose. Remember the condition was summation x_i was more than 1. I should keep a marker here. Let me, you remember the LP.



Correct? Agreed? So, now this and this look almost the same except this is a product and this is a multiplication. Any ideas have you seen these bounds where I can convert this into something so that AMGM will not help much. You know that $1 - x_i$ is log or exponential. So when you do this you realize that this probability is going to be less than 1 by e . Did I get it? e to the minus 1 I assume.

This is because you will get e to the power minus summation x_i . And I know that summation x_i is 1, ok, right. How do you prove this? Taylor expansion. Or as Soham says do not say it once you go to a university or a company to work by graph. All your theory teachers will be crying here.

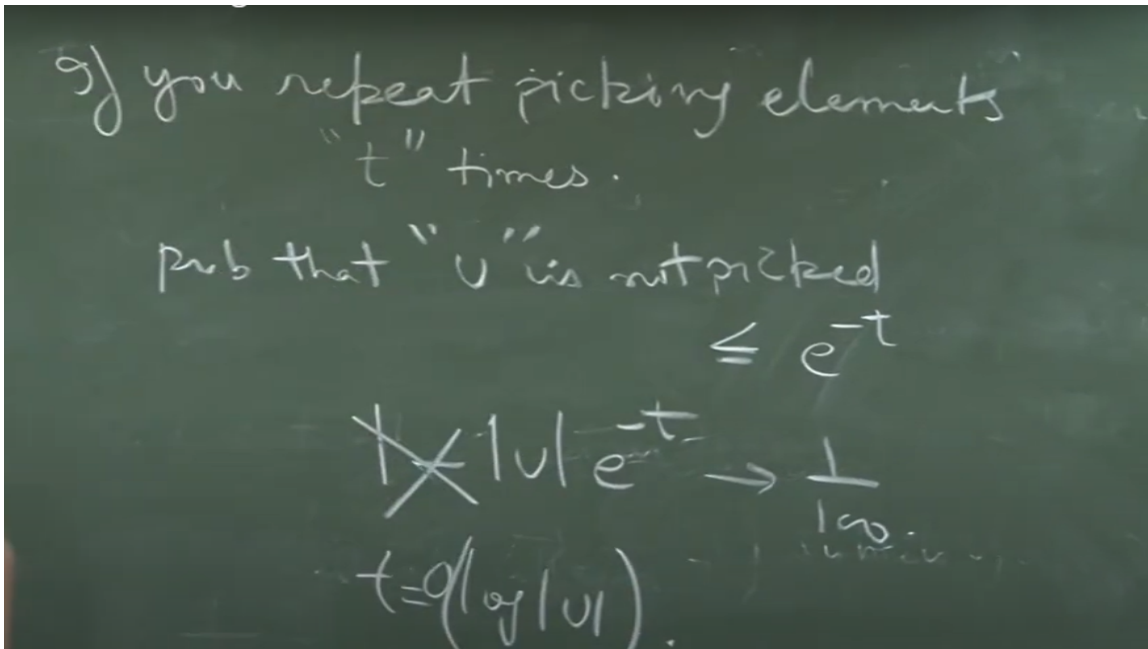
You say such things. That is why I said theory teachers will be crying. Systems people will be happy. But ok, so you got $1/e$. But then now suppose one element is not covered, other element is not covered, right.



What is the probability? Now if I want to put a bound that all elements are covered. The best I can get is $1 - |U| \cdot e^{-1}$. Correct? Again this is right. So the probability that one element is not covered is $1 - x_i < e^{-x_i}$. What is the probability that there is at least one element which is not covered? The bound I can give is only $|U| \cdot e^{-1}$.

That means my success probability is going to be $1 - |U| \cdot e^{-1}$. This is bad. I cannot hope if I have to solve set cover instances which have only two elements, things would not be that interesting. So what should we do? Very nice, very nice. What I can do is I can repeat.

Since there is high probability that I do not get a set cover, I keep repeating my experiment. I pick with probability x_i , once more do this, once more do this. What you can show is, t times probability that u , a particular element is not picked is right. It is less than e^{-t} . Every iteration is independent.



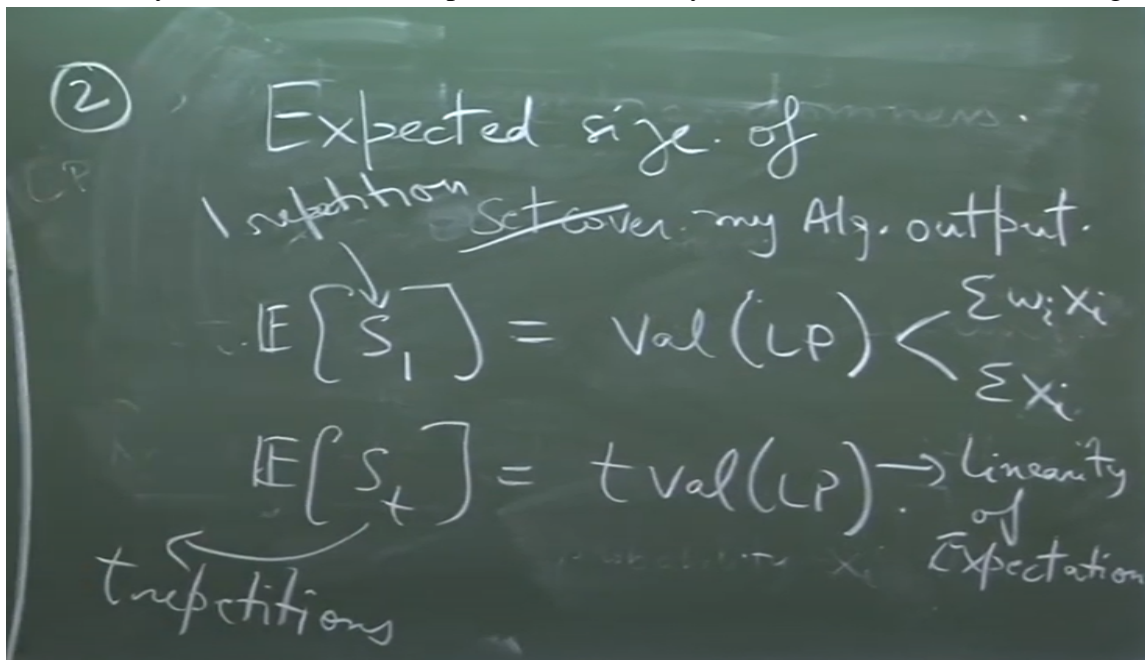
You fail with e to the minus 1, e to the minus 1, e to the minus 1, you will fail with e to the minus t , agreed? Now, what you want to make sure is that this is around 1 by 100. Sorry. This error probability is size of u multiplied by e to the power minus t and you want it to be really small. This is the first case there. What t should I pick? Very good.

So I will pick the nicely probably 100 times $\log u$, I do not know whatever or $\log 100$ times $\log u$ or whatever and then this goes like 1 by 100. This probability is very small that I do not get a set cover. And now comes the second part. I want to show, so what is the error when u is not picked in all the t iterations. In the first experiment you fail with probability 1 by e , second 1 by, last 1 by, what is the probability you fail in? Yes, so what is the probability that I fail in all the t ? It is e to the minus t because they are done independently.

Oh, you can assume that you put multi sets in the objective value where I will do, you can even assume that your set cover. If now what can his question is which is an obvious question, what happens if I get two successes right for a set? Do I keep it once, do I keep it twice? Like in your actual algorithm you will keep it once but the way I will analyze the cost I will assume you keep it twice. Still my factor would be close. Now, I want to analyze what size I am getting for this set cover. What I want to show is with high probability my set cover has small size right.

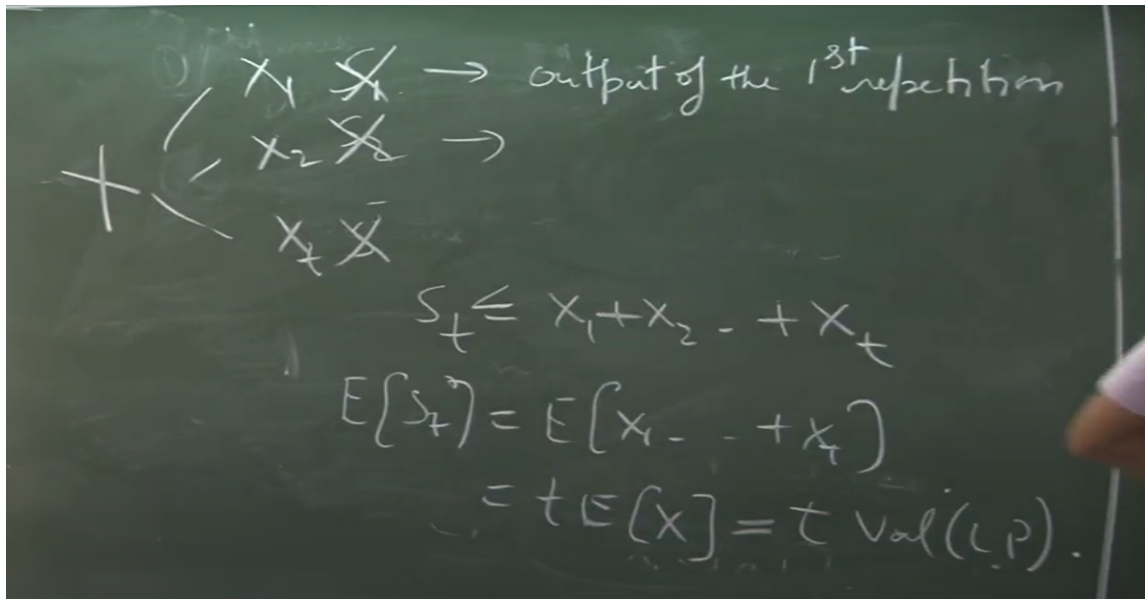
But my size is not a value it is a random variable agreed. So when you have a random variable you study its expected value. So the question is what is the expected size of my I am calling it a set cover it need not be a set cover or I should say what is the expected size of my algorithm output. Sounds good. So, let us not worry about t times repetition.

If I just repeat it once what would be my expected size or summation $\sum w_i x_i$ if you are taking the weighted version. So I will just call it the value of the LP right. So, expected size with just one repetition is equal to value of LP is this clear. This is summation $\sum x_i$, ok. This is by definition the expected size everyone is clear about this right.



The expected size this is the expected size this is the probability with which you are picking the set 1. So probability times the value this is the expected size. And now if I make t repetitions of the algorithm this will become t times value of LP. I see that many people have taken course of probability with me, ok. So if you say linearity of expectation and you smile that means you have taken probability course with me right.

Because I think this is one of the most fundamental results in probability theory. Just to explain this what has happened here. Let us say S_1 is your variable which tells you what is your output of the first repetition for all the n values right, ok. I need new variables. I have called S_t with t repetitions I will just call it x_1, x_2, x_3 right.



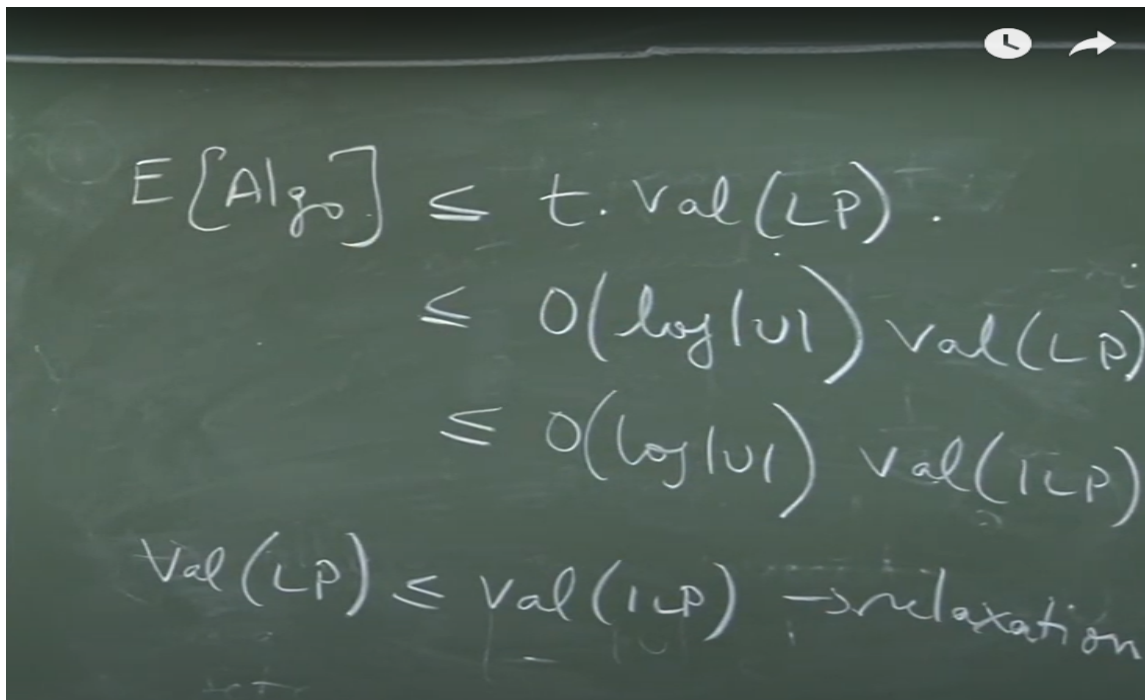
$x_1 \bar{x}_1 \rightarrow$ output of the 1st repetition
 $x_2 \bar{x}_2 \rightarrow$
 $x_t \bar{x}_t$

$$S_t \leq x_1 + x_2 + \dots + x_t$$

$$E[S_t] = E[x_1 + \dots + x_t]$$

$$= t E[x] = t \text{val}(\text{LP}).$$

So good since you mentioned this what is going to happen is going to come out like this right. So what do I know? What is S_t ? But not just that actually what will happen is less than equal to because this is the point which Dev pointed out. It might happen that I have already picked the set. So second time when I when I toss a coin it comes out to be head I will say oh I already have the set. But this is definitely an upper bound.



$$E[\text{Algo}] \leq t \cdot \text{val}(\text{LP}).$$

$$\leq O(\log|U|) \text{val}(\text{LP})$$

$$\leq O(\log|U|) \text{val}(\text{ILP})$$

$$\text{val}(\text{LP}) \leq \text{val}(\text{ILP}) \rightarrow \text{relaxation}$$

And value of these are all copies of random variable x , good. So now what I know is that my expected value is less than t times value of LP. What was my t ? What did I want to show? What was the definition of approximation? Whatever I the true value the value of ILP or the true set. Why is this true? It is a minimization problem. We have

increased the feasibility set, ok.

Good we have got an approximate algorithm. No exactly right. So what we have shown is that the in expectation we will get a high enough value. But that is not sufficient. Remember our notion our end goal was different.

Actually to be fair there are two kinds of end goals when you talk about randomized algorithm,ok. One is called the Las Vegas notion and one is known as the Monte Carlo, ok. I am not saying which one is better which one is worse. But in one case we just say that in expectation our solution will be nice, ok. If I want the value to be 20 my value will be less than 20 in expectation.

And people are happy this is one notion of happiness. Another notion of happiness slightly more complicated. It says at least with 2 by 3 probability I should get at least more than 20. And we started with that end goal.

Remember that is what we talked about. So if my end goal is the Monte Carlo version right. This is succeed in expectation right. Says with high probability we succeed right. We want to convert it. So do you know what tool do we use to do this? What how will you apply Chernoff here? What is Chernoff bound? You need some independent.

You do not have to go into the complicated notion here. You do not have to worry about Chernoff. Even the Markov will work here. How many people know what Markov inequality is? ok. So I will remind you if you do not you should know it. Especially if you have taken a probability course.

And this is made tailor made for such kind of a thing. Our point what is Markov inequality used for? So suppose my expectation is close to what I want. Can I say that with high probability I will get that value. This is what Markov inequality is for. It says if I have a positive random variable this is important.

Sorry I should not say expectation. Probability that X is greater than, ok. So now you must seems like something and the proof of this is ultra easy. You just write down the expectation. Remember expectation is kind of collecting mass. And this will say that the mass over aE_x is enough to give already give me the expectation.

Las-Vegas. — succeed in expectation

Monte Carlo — with high prob.
we succeed.

Markov inequality. $X \geq 0$

$$P[X \geq aE(X)] \leq \frac{1}{a}$$

There is no negative mass because my X is greater than equal to 0. So it will be a contradiction. If I assume that my probability is higher than $1/a$ then just this part will give me X more than $E(X)$. What about the rest? And as very nicely put many a times this is not enough this is weak. But then if we have nicer conditions or random variables then we can prove stronger bounds like Chernoff, ok. We have Chebyshev inequality, Chernoff bound all that. If our random variables are very well. Good. So now what can we say now? We have this inequality. We can say the probability is at most $1/a$. So my expectation is order of a times value of LP. C. Sorry. So ok a this is the correct.

$$Pr \left[\text{Algo}(\text{val}) \geq \frac{1}{a} \cdot o(\log|U|) \cdot \text{val}(LP) \right] \leq \frac{1}{a}$$

$$Pr \left[\text{Algo} \geq o(\log|U|) \cdot \text{val}(LP) \right] \leq \frac{1}{a} \leq \frac{1}{100}$$

take $a=100$.

Am I confusing this is fine right? This is less than, ok. And now comes the power of Ω notation, right, sorry. Confusion right? So I can just absorb this constant in this order of notation. So this was all the analysis. Yes? So your expectation is bigger than is smaller than this right? What do we know? So your expectation is smaller than this right? So what you know is that probability Algo greater than. This is just good this is the thing which I always get confused in what direction, but this is in the right direction.

What do I know? I know that this is let us say 100. Now if I put a bigger number this probability has to decrease right? So this definitely implies right. So this is saying probability greater than 100. This is saying probability greater than 200. Now probability greater than 200 is clearly greater than probability.

The image shows a chalkboard with the following handwritten text:

$$\Pr \left[\text{Algo}(\text{val}) \geq a \cdot E[x] \right] \leq \frac{1}{a}$$

$$\Rightarrow \Pr \left[\text{Algo}(\text{val}) \geq a \log(U) \cdot \frac{\text{val}}{(LP)} \right]$$

$$\leq \Pr \left[\text{Algo}(\text{val}) \geq a E[x] \right]$$

$$\leq \frac{1}{a}$$

So let me just write it if this is confusing. Thank you. That is just because this quantity is smaller than this right? So I will give a graphical proof of this right? This is $A \cdot x$ this is order $\log U$ whatever A times order $\log U$. So this sum is going to be this. This was a major result in our premier conference. So I do not think we have a constant sized thing. So you will start with you know square root of U or something or any approximation algorithm.

So $\log U$ is much better than anything. So you give me an algorithm in $\log U$. So this was like a major result after a long time people could give an approximation factor of $\log U$ right? But U is a parameter in the algorithm right? Is it de-randomized? I can take a look you can ask on hello and I can take a look and tell you what is the current node. So what I am trying to do. So this is good you have raised this point. So the technique seems very simple right? The analysis seems kind of loose and that is the power of relaxation in rounding.

That very simple clear idea gives you something which nobody knew before. So a nice clean technique which gives you a great result. And same thing you will see with SDP. With semi definite programs you will relax and round and will get this approximation factor of 0.85 which people did not dream of. And now with other evidence we kind of believe that that is the tightest possible. So the technique is once I have told you the technique it is very crisp. But to come up with this technique it took people 50s of years.

So that is the point. So the Math is not hard. The idea is not complicated. But still you get lot of reward for it. And that is why I was so excited to say relaxation in rounding. This has been like there are so many results if you look at my PhD career and now like last 20 years.

So many results on this relaxation rounding. And you get like there are actually there was a result which said for a host of problems that the best thing is to take the natural SDP, natural relaxed SDP and round it in a natural way. For combinatorial particular max cut kind of problems for any of those the best solution is take the SDP, take the relaxation rounding. This will give you the best algorithm under some assumption which is close to P not equal to NP .

Not exactly P not equal to NP . It is called unique games conjecture. I am going off topic. But yeah so this technique again looks simple but is very very strong. Lot of lot of work has been done on it. So all this analysis seems like array yeah yeah yeah follows follows follows.

In one class I could cover this. But 50 years of work did not produce this result. So that is the beauty of these ideas, ok. Good. So where was I? Yeah we got we get that with A equal to 100 probability that my algo gives this value is less than 1 by 100 right.