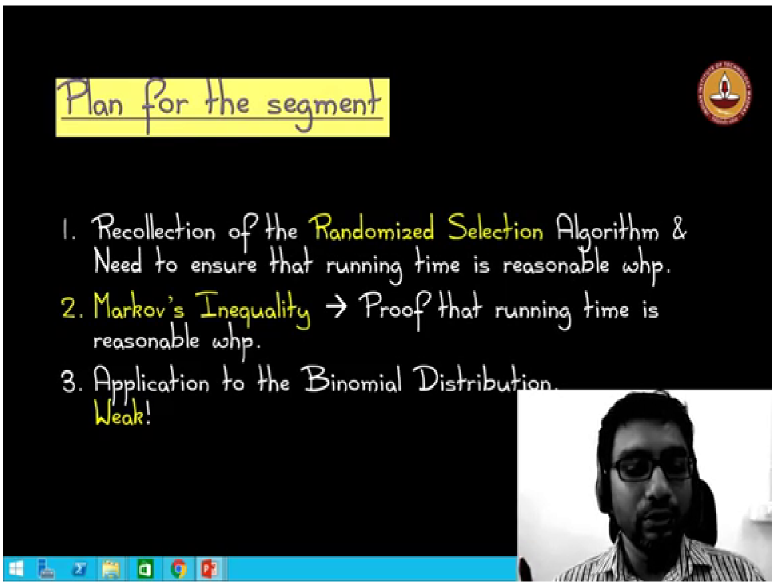
Probability & Computing
Prof. John Augustine
Department of Computer Science and Engineering
Indian Institute of Technology Madras

Module – 03
Tail Bounds I
Lecture - 15
Tail Bounds I - Markov's Inequality

So, we are now going to start a new module. In this module we are going to be concerned about tail bounds and what is that mean? Well we are interested in random variables like running time space complexity and so on and so forth in algorithm design. And we want to understand such random variables the first level of understanding we have from the expectation of that random variable, but that is not all we want to know we want to also understand how the random variable behaves with high probability. For example, the expectation could be small, but if an algorithm takes a lot of time every once in a while, then that might be a cause for concern. So, we want to be able to argue that the probability with which an algorithm takes a long time is very very small and tail bounds help us make such arguments ok.

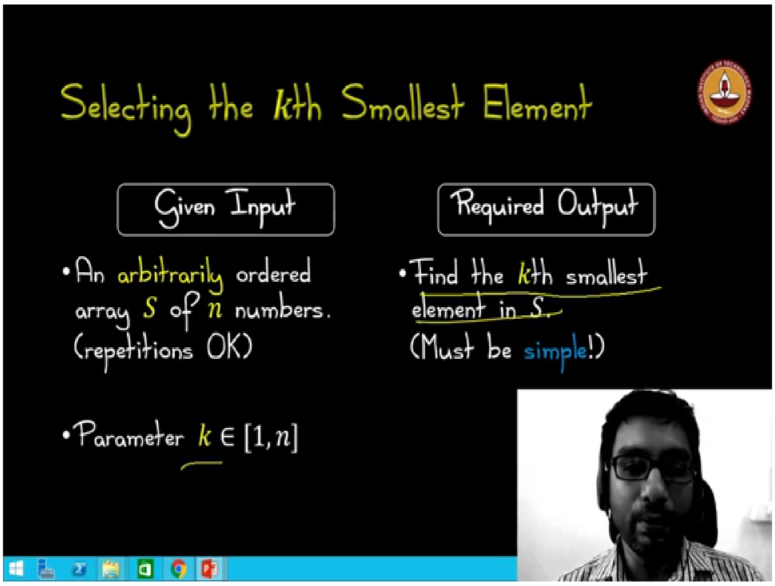
(Refer Slide Time: 01:22)



So, in today's segment we are going to look at Markov's inequality which is pretty much. The first starting point for these sort of tail inequalities we will be studying Markov's inequality, using randomized selection as the as the example problem and we will then prove Markov inequality, and then we will apply it to this randomized selection problem.

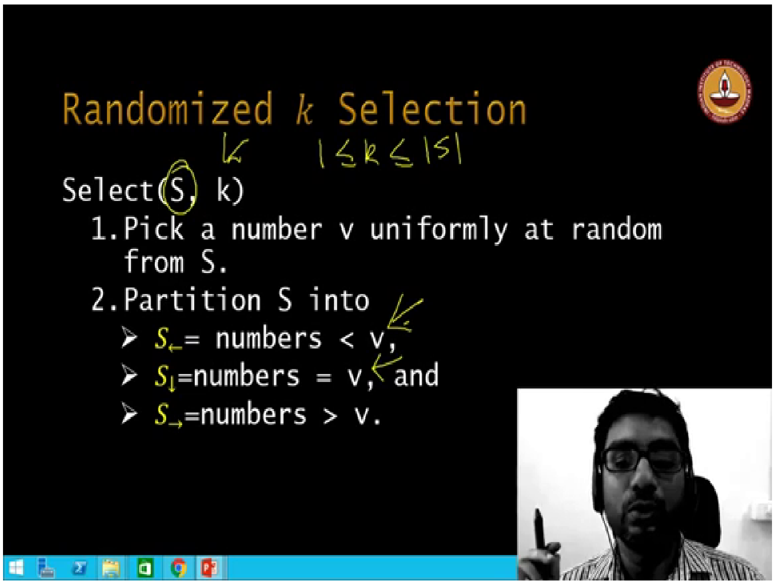
And we will also see how this Markov inequality plays out in binomial distribution, it turns out that its actually quite weak in general, but under certain conditions it will it will work and in fact, even though its weak in general, it is this is actually the starting point for all other tail inequalities that we will be studying ok.

(Refer Slide Time: 02:15)



So, without further ado let us look at the selection problem. So, we are given an arbitrarily ordered array of n numbers. This is an input array arbitrary order and we are also given a parameter k ok. And this parameter tells us specifies the index of the item that we want to output in the sorted order. So, in other words we want to find the k th smallest element in s, we would like this algorithm to be as simple as possible ok.

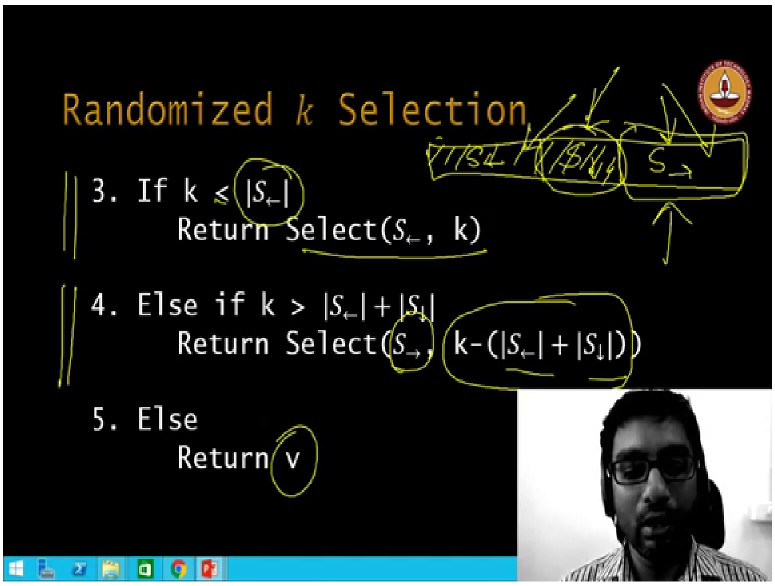
(Refer Slide Time: 02:54)



So, here is the randomized selection algorithm means it is kind of like quicksort it is a recursive algorithm, you take the entire array S and you have this parameter k. So, if case is case values 10, we would want to output the tenth smallest element. So, of course, k has to be in the range one to the size of the array.

So, now just like quicksort you pick a random number v and this v is picked uniformly at random from the set S and now we partition S into 3 parts S left arrow are all the items in S, that are less than v as down arrow is the numbers that are equal to v and S right arrow are all the numbers in S that are strictly greater than v and this can be done by a single sweep through the array S ok.

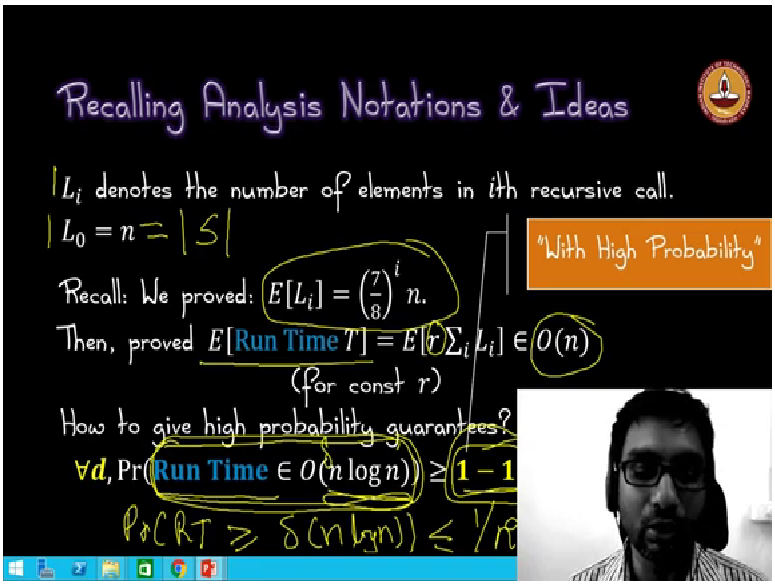
(Refer Slide Time: 04:14)



And now it is quite intuitive to see where the kth smallest element will lie ok, if S down arrow I mean S left arrow has a cardinality that is more than k, then in fact a greater than or equal if the cardinality of S left arrow is greater than or equal to k then what we can say is that the k th smallest element is in S left arrow. So, what we can do is simply recurs into that region ok. On the other hand if k is strictly greater than the first two parts the left arrow and the down arrow, then k is clearly in as right arrow ok. So, you recurse into S right arrow, but you have to provide a parameter that is slightly updated. So, no longer will k be the right parameter; you will have to remove S left arrow and S down arrow. So, maybe a picture will help us. So, if you look at the array S what we have done is we have split this array into S left arrow S down arrow and S right arrow.

So, the step 3 takes care of the case where k the kth smallest element is over here, step 4 basically takes care of the case where the kth smallest element lies in this region and because we are going to recurse into this region alone, we have to remove this many on this many numbers from k. So, that we only focus on item number whatever we get over here in this region and of course, if 3 steps 3 and 4 dot work; that means, k is in this region, in which case remember all of these items have value equal to k. So, you simply return the, and this is nothing, but the pivot element v that we chose. So, simply we can return v.

(Refer Slide Time: 06:57)

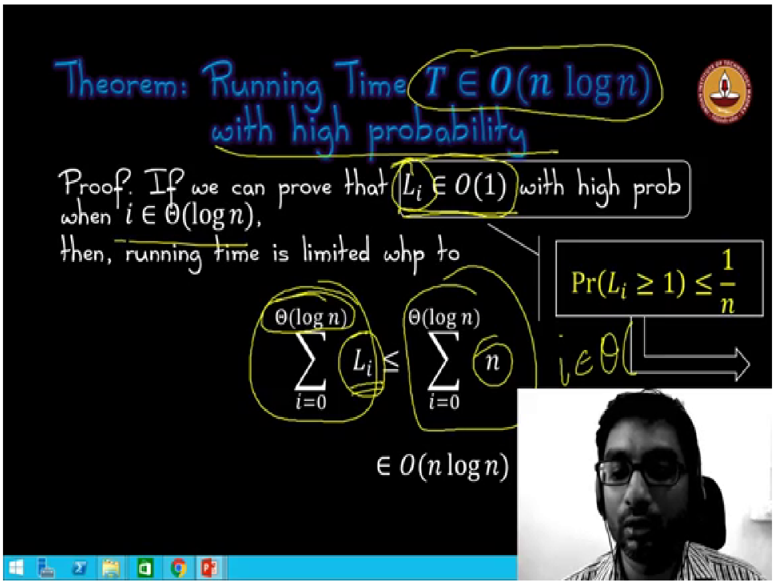


So, this is how the randomized case selection works at least this algorithm, and we would like to understand how good this algorithm is, and recall we have already looked at this algorithm and and we know the expected running time of this algorithm, which is quite good ok. And we understood this the we studied the expected running time of the algorithm by defining these random variables L i and L 0 and you recall that L i denotes the number of elements in the ith recursive call and L 0 is simply the number of items, that we started with its basically the cardinality of S ok.

And we showed that the expectation of L i is 7 over 8 raised to the power i times m and using that we were able to show that the expected running time T, is is this expectation of the sum of the Li's and scaled by r which captures the fact that a L i is just represent the number of elements whereas, the running time might be a little bit more, but still only a linear amount more. So, now we can you know solve I mean you can work out that this expectation applying any additive expectation and things like that will work out to often. So, we have seen this already, now what we really want to show in this, now is that the run time is not too large with high probability. I mean when we when we claim that some event holds with high probability, what we mean that what we mean is that the probabilities of the form 1 minus 1 over and to the sum constant d ok.

Which means in another way of saying the same thing is that the probability of the runtime being greater than or equal to some delta whatever the delta S times n log n should be at most some 1 over n to the d ok. So, these are two equivalent ways of stating this high probability claim you can if you are looking at the good event the runtime being o of n, log and being small you need to show that it works out with high probability, but if you focus on the bad event bad event is when the runtime becomes larger than some quantity. So, greater than some delta times n log n that is the bad event if you are focusing on the bad even then the probability of the bad even should be very small should be at most 1 over n to the d, there are two equivalent statements assuming the delta in this own; in the constant of the ol notation.

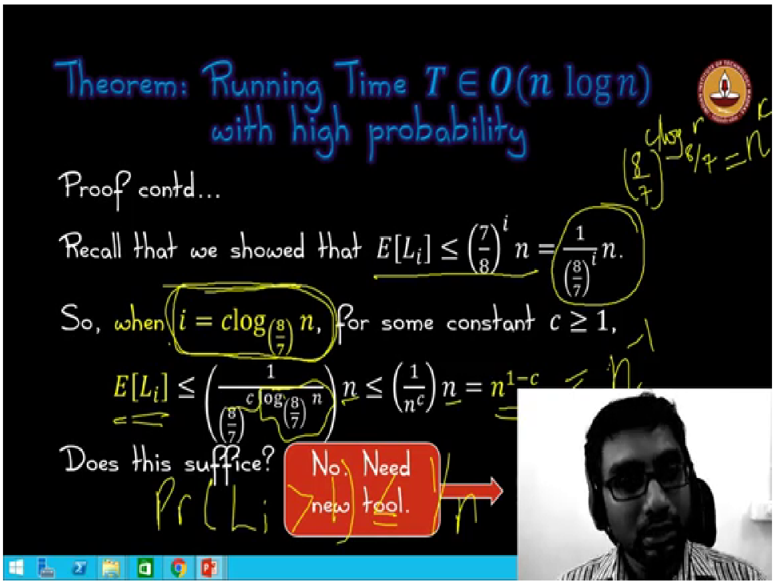
(Refer Slide Time: 10:25)



So, how are we going to achieve this? Prove this following theorem that the running time is O of n log n with high probability, and we will prove that L i becomes small with high probability when I becomes off log n ok. So, as the as you recurs further and further after the ith roughly the ith I mean i equal to O of log n at recursion, what happens is the size of the array becomes a constant ok. And once it becomes a constant then this algorithm is only going to the remaining running time is only going to be a constant. So, we do not care. So, if if we were able to show this, then the running time is essentially limited by this summation i equal to 0 to some theta of log in, and sum up all the individual Lis and of course, there might be a scaling factor r coming, but that is only a constant ok. and so, now, if if you if if if this is what we care about the easier thing to and notice that if after the log nth iteration if L i becomes small, we can simply stop with summing over the first log n L i values ok.

And so, then that this can be upper bounded by this quantity. So, here essentially what we are doing is we are replacing L i by a very gross upper bound n ok. So, so clearly the running time is upper bounded by the summation of i from 0 to log and theta of log in times of n and that is not hard to see that its o of n log n. So, now, what remains to be shown is this that the probability of L i greater than or equal to 1 is at most 1 over n when i is some data of log n.

(Refer Slide Time: 12:58)

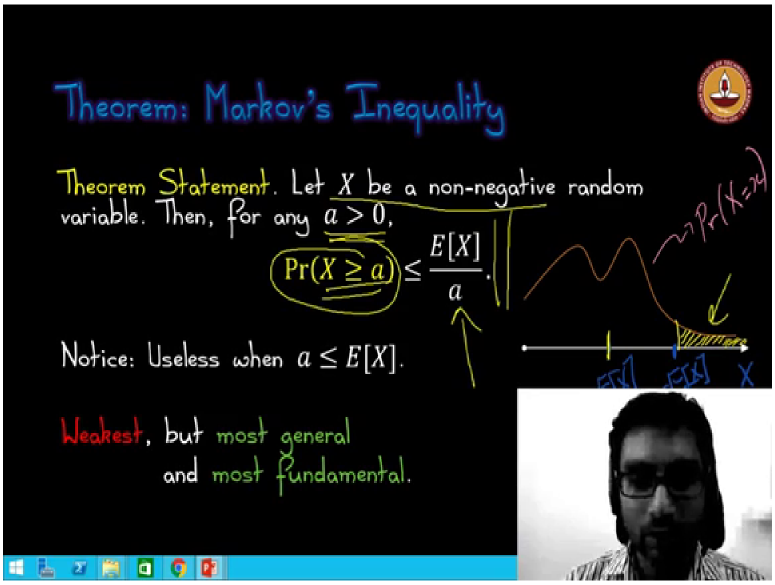


So recall that we have already shown something about L i, Lis we have already seen that L i is at most 7 over 8 raised to the power i times n, and we can write that as essentially take the 7 over 8 raised to the i to the denominator. Now it will have to take the reciprocal. So, it becomes 8 over 7 raised to the power i ok.

And now, let us remember we are interested in the case where i is some data of login. So, what we do is set i to be some c times log to the base 8 over 7 of and c is some constant. So, let us see how this plays out. So, now, when you plug this value of i into this term you get this expectation of lie now becomes at most 1 over 8 over 7 raised to the power c log to the base 8 over 7 and ok. But what is this quantity actually actually what is this quantity 8 over 7 raised to the power log base 8 to the 7 and that is nothing, but n and there is the c. So, this c will turn out to be n c. So, this whole inequality becomes at most 1 over n to the c times the n which is n to the one minus c. So, what have we shown? We have shown that the expected value of a L i is at most n to the 1 minus c when I takes on this value. So, that is pretty small.

So, for example, when c is 2, this will become n to the minus 1 and that is a very small fraction, but that is not exactly what we want, we want to show that the probability that L i is greater than 1 is at most some 1 over we want to show something like that ok.

(Refer Slide Time: 15:46)



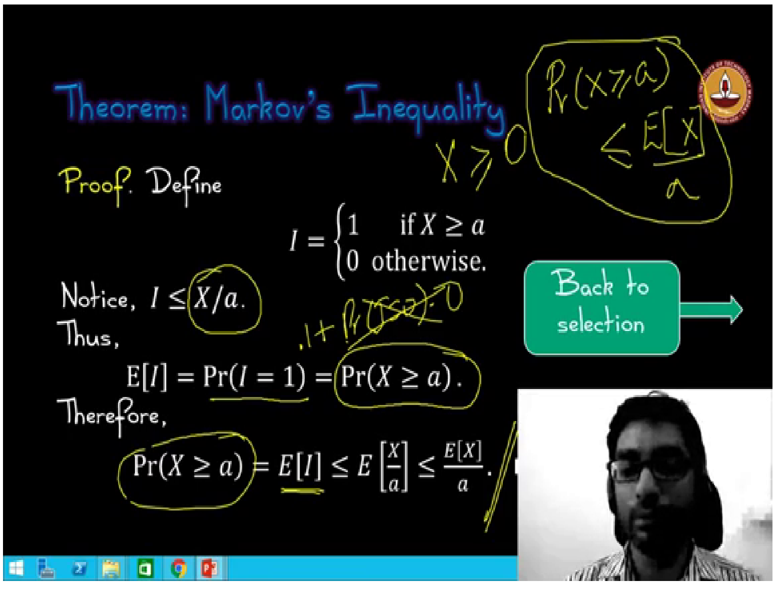
So, we need a new tool for this and this is where Markov's equality shows up. We have been able to bring the expectation of L i down to something very small we have to exploit them ok. So, Markov's inequality states the following let x be a non-negative random variable ok, then consider any value a and ask what is the probability that x is greater than or equal to a, and that is upper bounded by the expectation of x divided by a. So, a lot of times a picture would help understand what we are talking about. So, here is the distribution of x it can be a complicated distribution, but we want X to be non-negative that is an important requirement.

And so, let us assume that the expectation lies around here, were going to what we are asking is what is the probability that X is greater than or equal to a and that is the shaded portion over here the area under this in this shaded of the shaded portion and what does this inequality say that? That is that is at most the expectation of X divided by a. And you immediately notice that this inequality only makes sense as long as a at most expectation of X. If a is smaller than the expectation of X, then this inequality the right hand side will become greater than 1 and that is useless because we know that probabilities are always bounded by 1.

So, that is not very meaningful him and in general this Markov's equality tends to be very weak, but it is still fundamental because its power shows up when the expectation becomes small, then you will be able to get a very good upper bound its small upper bound on this probability. Remember this is typically going to be the probability of some bad event and so, you want this right hand side this is this is the bad region. So, you want do not you for example, you do not want your running time to be too large that is the bad situation and you would not argue that the probability of that bad situation is very very small ok.

So, you are most often you are interested in ensuring that this right hand side is as small as possible, and that particularly comes out I mean in the context of Markov's inequality, it is only useful as long as the expectation is sufficiently small ok.

(Refer Slide Time: 18:33)

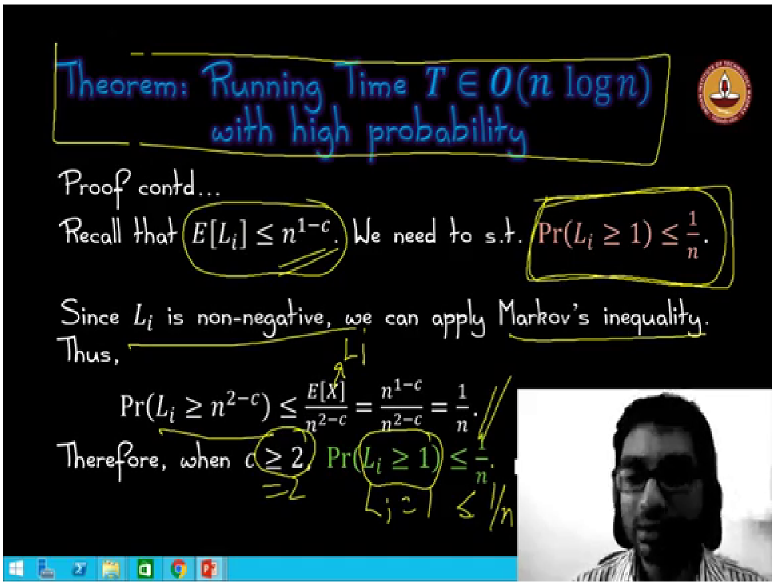


So, let us quickly prove this inequality, let us define a variable i and indicate a variable I that is 1 whenever X is greater than or equal to a and 0 otherwise ok. And this immediately tells us that I is at most x over a because if x is greater than a greater than or equal to a then I is 1 and the right hand side will be larger. If when X is when X is less than a, then I will jump to 0 and the right hand side will still be a fraction remember X always is greater than or equal to 0 that is a requirement for this Markov's inequality ok.

So, now let us look at the expectation of X I and that is nothing, but the probability because this is a 0 1 random variable this is going to work out to just being the probability that I equal to 1. Why because this is if you work it out its going to be the probability that I equal to 1 times one plus the probability that I equal to 0 times 0. So, this term just will vanish away and so, that is why we are simply writing it as probability that I equal to one and that probability is nothing, but the probability that x is greater than or equal to a ok.

So, which remember is exactly what we are interested in ok. So, this probability that X is greater than or equal to a is nothing, but expectation of I these are all equalities I am just using that which is at most now what is I? I is nothing, but X over a. So, at most expectation of X over a and by linearity of expectation its nothing, but expectation of X divided by a and that is exactly what we want ok. So, now, we we have a handle on Markov's inequalities and she says that the probability that X is greater than or equal to some a is at most the expectation of X divided by a ok. So, remember this now were going to go back to the selection problem ok.

(Refer Slide Time: 21:04)

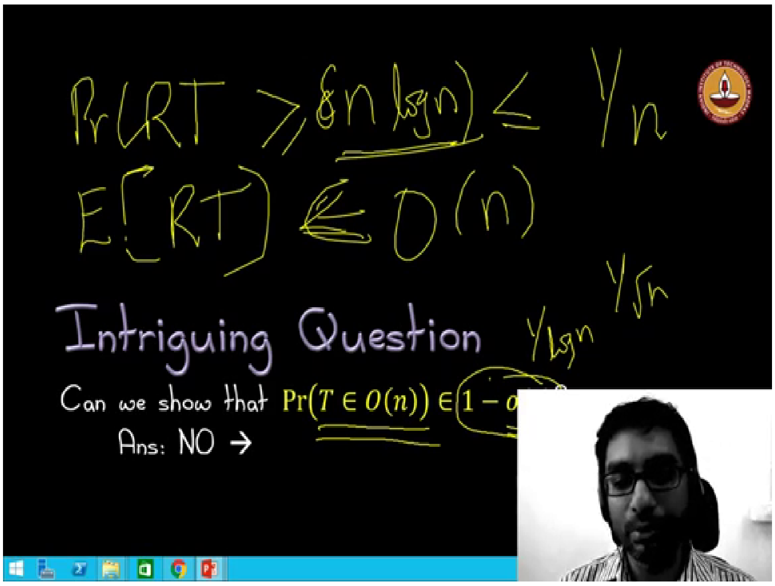


So, what do we know in the selection problem, when I is some c times log n we get this the expectation of L i is at most n to the 1 minus c ok. And remember now this expectation is sufficiently small when I is small. So, now, we need to what we are trying to show is something like this probability that L i is greater than or equal to 1 is at most 1 over n, this is the form that we want and. So, to get this form we have to apply Markov's inequality, the nice thing is L i is non-negative it is a running time or the size of the array. So, its non-negative. So, we can apply Markov's equality ok.

So, we ask what is a probability that L i is greater than n n to the 2 minus c that is nothing, but the expectation of X divided by n to the 2 minus c just by applying Markov's inequality ok. And we know that the expectation of x in the well is actually this axis should be a L i and what is the expectation of L i that is nothing, but n to the 1 minus c divided by n to the 2 minus c which turns out to be 1 over n ok. So, now, if if you set c equal to 2 or anything larger than 2, then essentially like it specifically if you set c equal to 2 you will get property that L i equal to 1 is at most 1 over n and of course, if you increase the value of c then you you only generalize and so, this is exactly what we want.

And that with that we complete the proof of this theorem that the running time of this randomized selection algorithm is at most o of n log n with high probability what we have shown here in this slide is this that the probability that L i is greater than or equal to 1 is at most 1 over n and you have to then consider. So, what we have just to recall how this proof completes, you have to remember that given that the probability that L i greater than or equal to 1 is at most 1 over n we can then plug in the fact that these Lis are at most I mean if it is not even going to be greater than 1 with you know then clearly you can upper bound it by n, and summation of these ends over the log n iterations is going to be O of n log n and this is going to be true with high probability ok.

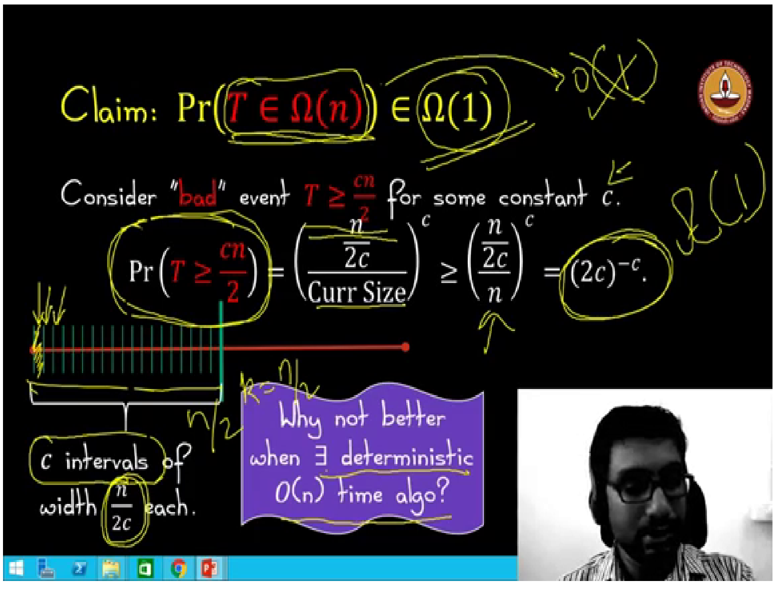
(Refer Slide Time: 24:21)



So, now were going to ask somewhat intriguing questions. So, what we have shown so far is that the probability of the running time being greater than or equal to some delta times n log n is at most 1 over n. Essentially saying that it does not I mean the probability that it goes too large and in that in that sense too large here is defined as n log n is that most 1 over n. But we also know that the expectation of the running time is at most some O of n ok. So, this is a gap expectation is O of n and, but we have only been able to show that the running time with high probability stays with an n log n ok.

So, the natural question is can we say something with or not even high probability, but some probability tending towards 1 for this event that T belongs to O of n what can we say about the probability that, T is some O of n say what is the probability that T is, at most 50 times n, where 50 is a constant right and can we argue somehow that that probability will be 1 minus some little O of n. Little O of n can be something like 1 over log n or 1 over square root of n or something like that, what it cannot be is a constant little O of n is something sub constant and were asking whether that is ah. So, basically this is a probability that tends to 1 as n increases is that possible as it turns out the answer is no. So, this you have to live with this divide between the expected running time and the bound that we have on the running time with high probability there will have to be a separation ok.

(Refer Slide Time: 26:43)



So, what we are going to show is that, the bad event that the running time belongs to some omega of n is going to be at least a constant for this to hold this is the bad event should be should have been little o of 1, but that is that is not going to happen because it is we are going to show that that is going to be omega of 1 ok. So, that is what we are going to show now, and we are going to show that in a in a carefully constructed manner ok. So, the bad event this we are going to specify what we mean by T belongs to big omega of n, this the bad event is going to we are going to consider the bad event specific bad event as T to be greater than some constant c times n divided by 2 and c is c can be any constant ok.

So, now we ask what is the probability that, T is greater than or equal to c n by 2 and we are going to argue that that probability is going to at least be some constant remember c is a constant. So, any function of a constant, this should be a constant this will be a constant right we would not argue that this probability is going to be more than a constant, which is which is exactly what we mean by omega of 1 how do we make this argument?

Let us assume let us look at the algorithm the execution of this randomized selection algorithm, but we are going to view it from the point of view that the items are sorted. So, in reality the items will not be sorted, but we are just going to for the sake of analysis observe what happens in this sorted order ok. So, this is the sorted order, but we are going to break the sorted order at least the first half of the sorted order into little pieces and these little pieces there are going to be c such intervals and each such interval is going to consist of n over 2 c items ok. Basically this is n over 2 and that n over 2 divided into c pieces is n over 2 c items each and in this case we are going to assume that k equal to n over 2. So, were basically trying to find the median element ok.

So, what could go wrong in this sort of view? Well remember and each recursive call we are going to pick a random element. So, what is the probability that the very first random element is going to be in this very first interval ok? Well the width of that interval is going to be n over 2 c and the overall and then this. So, you your random favourite element is going to be in that interval with probability n over 2 c divided by the total size of the array ok, that is the current size and what about the next iteration?

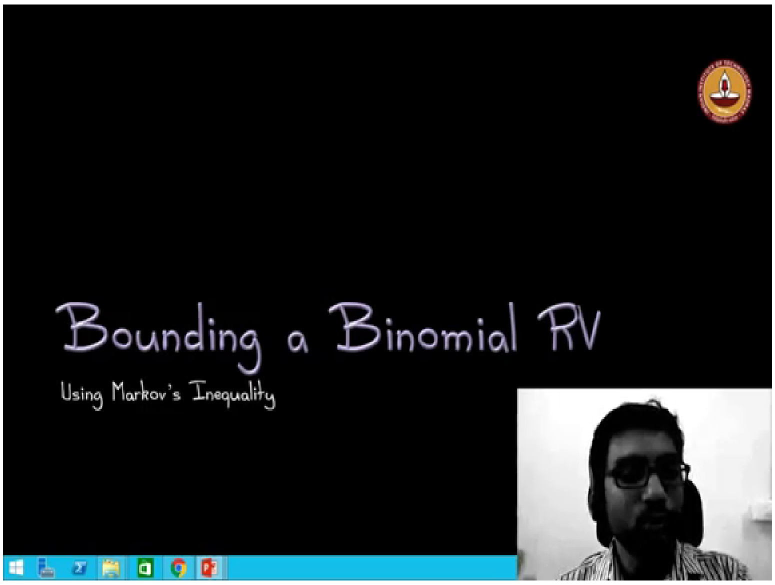
So, let us say that the first recursive call you pick a pivot in that region ok. In the second recursive call what is the probability that you would pick a pivot? In the second interval that is again going to be n over 2 c, but now the size might have been smaller because you your erase is reduced a little bit, but its only reduced by a little bit because you will probably eliminated the first interval. So, that is ah. So, I am just not going to worry about exact size, but I am just going to call it the current size and so on and so forth.

In each recursive call the bad thing that, could happen is that the pivot gets chosen from this tiny sliver of items at the far left and that those bad events can happen with these probabilities n over 2 c divided by the whatever the current size is and I am only interested in a lower bound for this probability. So, I can make I can I can replace this by a smaller quantity. So, what do I do? I look at this denominator it is the current size. And if I replace it by a larger quantity I certainly will know that this will be larger than whatever I get over here.

So, that being the case I am going to replace the denominator by n, because n is the full size of the original array and the current size could only have been smaller ok. So, I get and now I the n cancels out and I get 2 c raised to the power minus c which is a constant. So, what is the final outcome of this argument? We have been able to show that the probability of the bad event that the running time is greater than c n by 2 is at least a constant. So, this means that there you will not be able to prove any high probability result for a running time of O of n that is that is that clearly a separation between the expected time analysis, and the running time analysis with high probability.

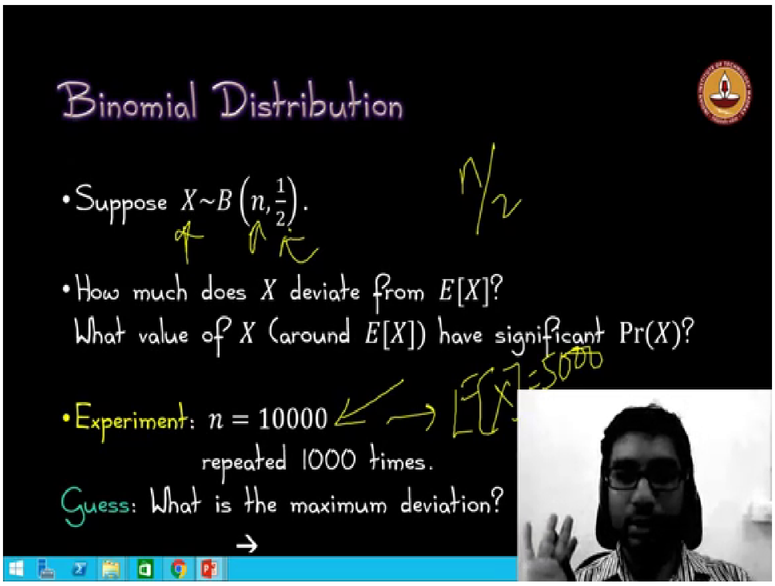
So, you may wonder at this point and especially those of you who have studied a selection problem in an undergraduate algorithms course, you may you may recall that there exists a deterministic of n time algorithm and why not something better as it turns out little while later, we are actually going to see an improved algorithm which is actually going to run in O of n time with high probability, that is going to be a different algorithm. But this randomized selection algorithm that kind of resembles quick sort is unfortunately going to have this sort of a chasm between expected time analysis and high probability analysis ok.

(Refer Slide Time: 34:27)



So, now let us see this whole randomized selection allowed us to explore this notion of Markov's inequality, we were able to exploit that to get a bound on the running time. Let us use the Markov inequality to bound a binomial random variable ok. So, it will help us get a sense of what it is useful for and how to use it ok.

(Refer Slide Time: 34:51)

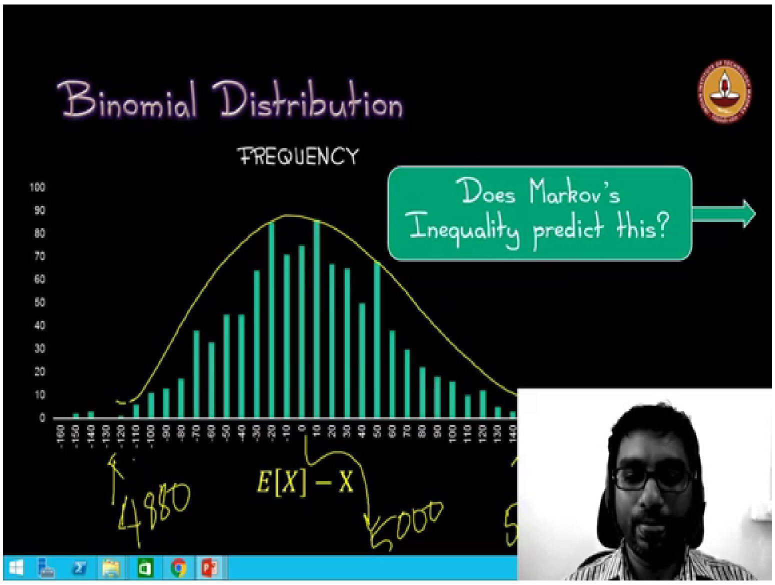


So, now, let us let us assume that X is drawn from the binomial random variable binomial distribution with parameters and meaning n coins are tossed and the p values are half, which means that the coins are all unbiased ok. So, the question we ask in this sort of tail bound analysis is how much does X deviate from its expectation ok. So, what range of values of X have significant probabilities?

So, here is an experiment that I quickly coded up, I am going to set n equal to 10,000 and I am going to draw X repeatedly 1000 times. And I am going to ask how much does it deviate from the expectation remember expectation here should be n over 2 right. So, in this context when n is 10,000 the expectation of X should be 5000 right. So, the question is; how much does it deviate from this 5000 and you may want to pause here to ask what what will that deviation be.

You know can we say most of the time it is going to be within say 2500 to 7500 or is it going your are you going to be able to see a wider range of numbers. So, for example, remember were repeating it then I mean 1000 times. So, how many times is going to be less you know less than 2000, this x how many times is going to be greater than 9000 these are all you know the type of questions that these tail bounds try to ask and so, you may want to pause a little bit to try and answer that question intuitively, just think about it and after you have passed and you have had some thought let us come back to this ok. Now having thought about it will show you a picture might surprise you ok.

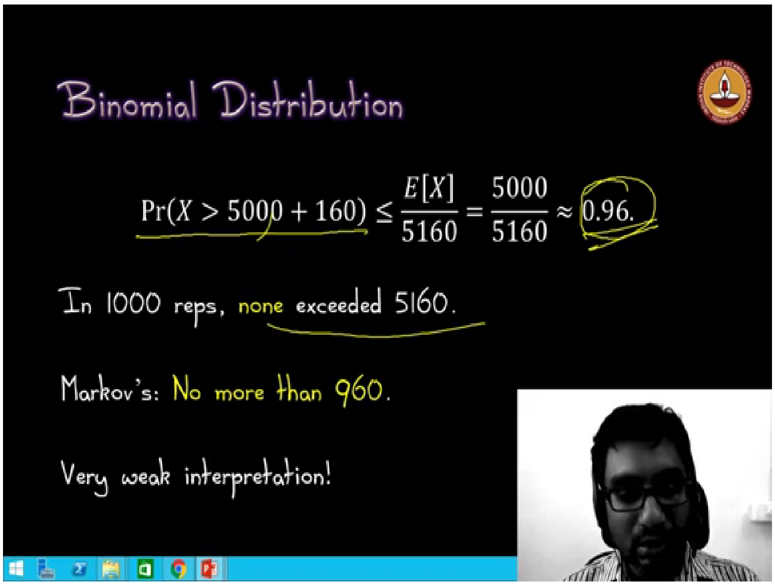
(Refer Slide Time: 37:08)



So, here I am plotting on the x axis its expectation of X minus X. So, this is the 0 here actually refers to the 5000 mark and so, this for example, 120 refers to the 5000 minus 120. So, that is like 4880th mark this refers to 5000 plus 140. So, that is going to be 5140 as a thumbs are out of these 10,000 repetitions most of the time you are within a plus or minus 120 130 ok. So, we never went below 4850 we never went above 5150. So, clearly we never would have touched 2500 or 7500 ok.

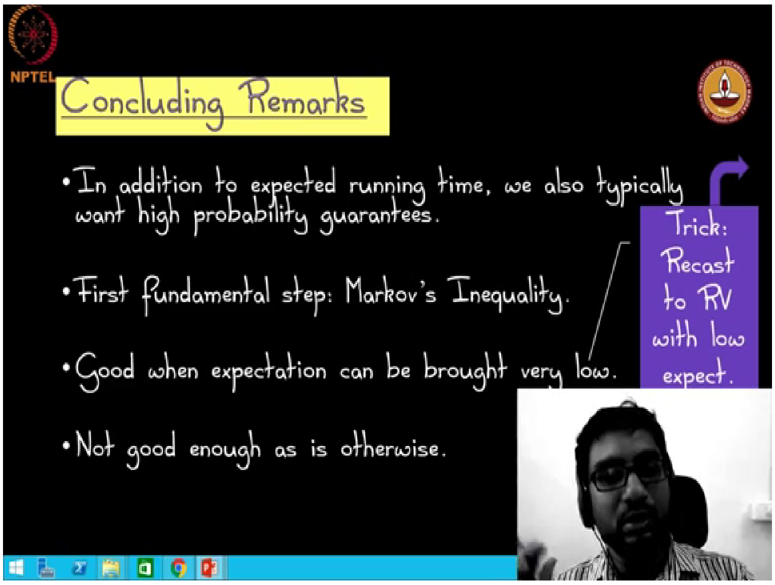
So, this binomial distribution when you think about, it is actually going to be very tightly bound around its expectation and this is this is an important intuition that you need to develop ok. So, even though its random it is highly predictable. Now let us see if Markov's inequality is really any good like how good is its prediction is it able to tell that this random variable X drawn from the binomial distribution is going to be close to the expectation or not let us let us see where it says.

(Refer Slide Time: 39:10)



Let us ask; what is the probability that X is greater than the expectation 5000 plus 160 that is going to be the expectation which is 5000 divided by 5160 that is 0.96 and what do you think about it out of the 1000 reps that we did for this experiment, none of them exceeded 5160. So, the probability should be much closer to 0, but were getting a probability of 0.96 ok. So, this is clearly telling you that Marcos is very very weak in this context at least it is not always weak its sometimes it is really the best starting point that you can have for these tail bound analysis, but in this context its weak ok. So, we need slightly more nuanced tail bounds to analyse the binomial distribution ok, and that is what we will be developing in the rest of this module.

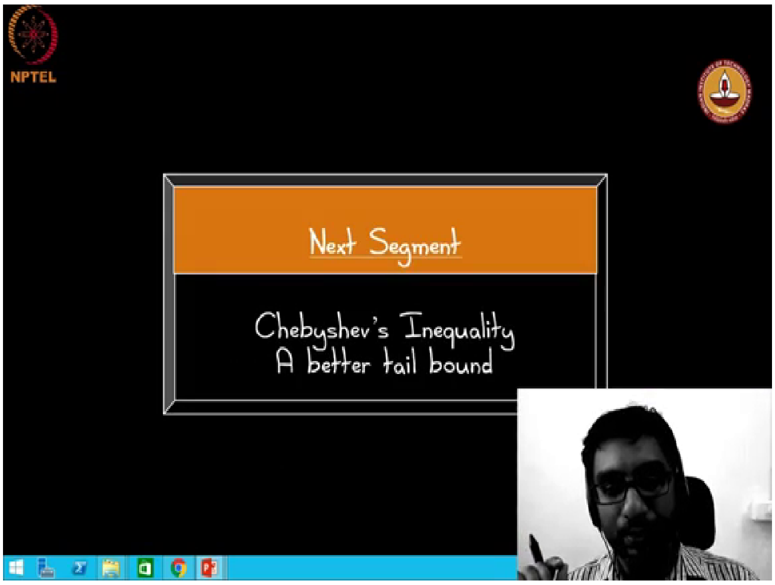
(Refer Slide Time: 40:22)



So, to conclude in addition to the expected running time, we also typically want high probability guarantees on the running time. So, we want to be able to say that the running time does not exceed some quantity with high probability or that it exceeds the this value with very small probability, both equivalent ways of saying things and we have looked at the first fundamental step towards these sort of tail bounds and Markov's inequality, and we have noticed that when the expectation can be brought low then Marcoss is quite good, but not so, good otherwise. So, it did not; was not very good for the binomial distribution, where the expectation was already high like 500 5000 or something.

So, the trick is to really exploit this feature of Markov's inequality. So, whatever random variable we want instead of bounding that random variable directly, if its expectation is high we want to be able to recast that. So, that were looking at a random variable with low probability and then apply Markov inequality and thereby good get tight bounce ok.

(Refer Slide Time: 41:44)



So, with that we conclude the first segment in the next segment we will look at a slightly tighter inequality call Chebyshev inequality.