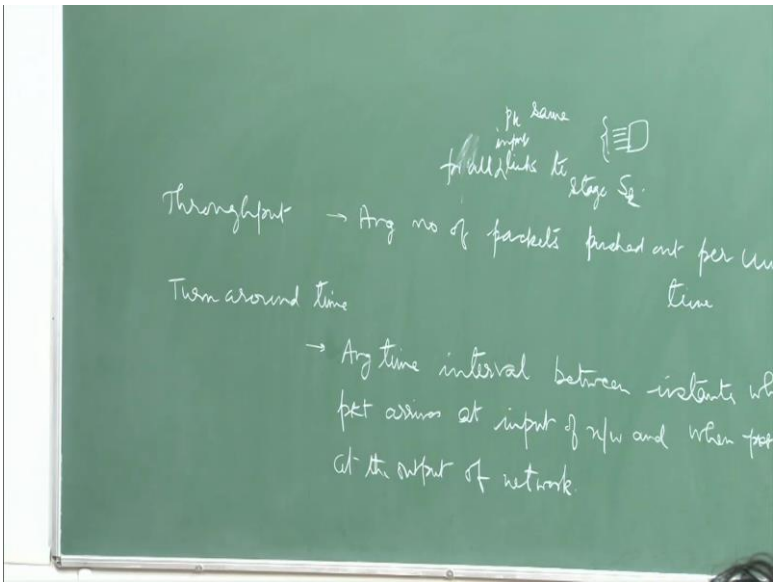
Digital Switching
Prof. Y. N. Singh
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Lecture – 25

So we will actually go ahead with the calculations actually we were trying to do last time. So, I thought I actually have told last was the two matrix which have to be used for performance analysis.

(Refer Slide Time: 00:34)

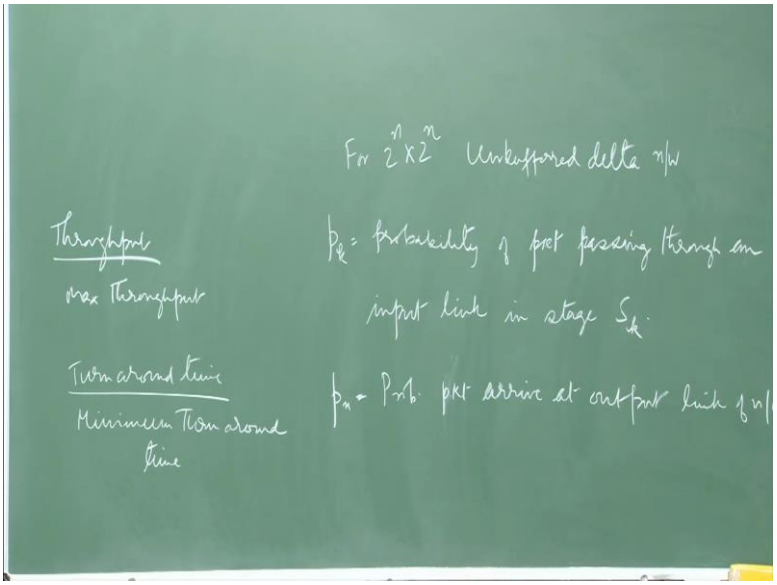


One of them was throughput, and other was one turnaround time. So, the way they are defined that throughput is nothing but average number of packets which are being pushed out per unit time; that is how throughput is taken. It has nothing to do with the input; of course, if you keep on changing your input rate, throughput will change. And throughput ultimately should saturate, okay; throughput ultimately should saturate. Turnaround time will be your average time interval between instants when packet arrives at input of network and when packet leaves at the output of network, okay.

And of course, the maximum throughput will be existence when they are no conflicts. So, for example, at each input you have packets in such a way there are no conflicts, and all of them can be pushed without any contention to the outgoing port. So, that will be the case for maximum throughput and minimum turnaround time also will happen when

there are no conflicts, because you will not be storing anything in the buffer; it will just pass through, okay. So, usually then we use the normalized version of this.

(Refer Slide Time: 03:07)



So, whatever will be the maximum throughput possible, you can normalize with that. This will ensure your normalize throughput is always less than one; this is for comparison sake, it is independent of size as now. So, that is a normalized value. Similarly, for turnaround time also. So, what is the basic unit of time counting, you are essentially looking at that. So, you will divide it by minimum turnaround time, okay. So, these are the two normalized version which actually have to be used.

So, first of all we will actually estimate the same thing for unbuffered delta not the buffered one but unbuffered one which we had done earlier, okay. So, this one is pretty simple. So, for 2 raise to the power n by 2 raise to the power n unbuffered delta, you can assume p k will be the probability that. So, remember the case of state for different ages, there will be different probabilities.

So, I can actually attractively compute if I know the input probability, I can compute whatever the output probability and that will give me a throughput; that is basically the idea here, okay. And from here you can define p n in the similar session. This is probability that the packet arrive at output line; all n stages have been finished. The total n minus one zero I am counting zero one two three n minus one are stages; the moment I talk about p n that is output link, okay, because there is nothing like a s n stages not

there. S zero to s n minus one is there. So, these will be your technically your throughput actually.

Now there are certain assumptions which have to be there. So, at every intermediate switch, in general there can be many inputs; n s we had done earlier, each one of them is going to connect to a set of inputs. So, each one of these inputs will be connecting to set of inputs of the network. So, in earlier case it was only two set I x and I y I was talking about. They were always disjoint; here also all the sets which you are going to form will be disjoint. Once they are disjoint, since, all arrivals at the network are independent and they are kind of independent identical processes is actually arrival process.

So, that independence also holds true here. In fact, it holds through at every intermediate switch, okay. So, this actually means I can assume that p k is going to be same for all links all input links to stage k. So, not only these switch; in that stage whatever number of switches is there all inputs in that stage, they will be all independent actually. So, there is no correlation between them. And this actually you can keep on extending. It means this p n will be same for all output links also; that independence will still be maintained not for output side but yeah.

Student: Sir, p k is same all switches at stage k?

Hmm.

Student: Is Stage the same for all k.

P k is going to be same for all switches in stage k, right.

Each one is an independent set actually, and each one of them will be now connecting 2 raise to the power k inputs. So, since the number of inputs is actually being same to which it can reach, and each set is different. Okay, within a switch its each set will be different. So, this will be showing independence; this is actually true for everybody. So, they are going to be same; they are independent within a switch.

If you take two different switches, then they are not independent, because they might be coming from the same input, but since every packet is also coming independently, it is okay, because independence comes because of the fact when a packet comes with the equal probability to choosing an output destination. There is no bias; with 1 by n

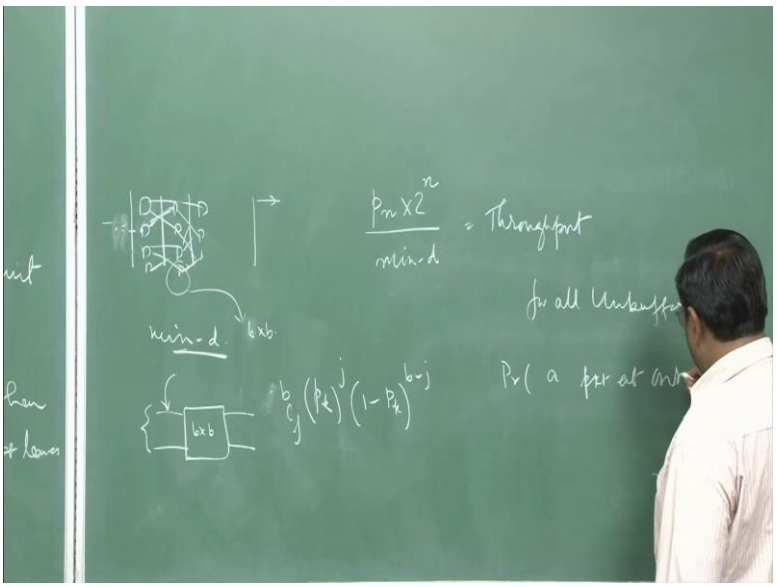
probability 1 by 2 raise to the power n in this case; it is going to choose anyone of the output outgoing lengths.

And each packet in all inputs is going to do the same thing. So, statistically yeah, every input port is technically same. You just find out probability at place, use it at everywhere unless you have a symmetric situation; it is a symmetric or uniform traffic conditions actually here. And important thing is that it is independent of what kind of delta network you are taking; whether, it is a shuffle net based or it is an inverse based or whatever you can design or it is an omega network or whatever it is. So, far it is a delta network, the situation is going to be true.

In fact, it is true for any banyan network, not only this; only in the case of banyan network is that tag, the address tag will not be uniform for a given output. Address tag has to be different at different inputs for same output actually; that is the only variation. But so far that condition there is exactly one path from any input to any output is maintained, and no input links and no output links in the intermediate stages are left open; this is going to be true, okay.

Technically, it is a demultiplexer tree which you are building all the time. And of course, this is now throughput. I can write a simple statement from here. Now paper actually gives slightly different variation here. I also could not figure out why it is being given that way.

(Refer Slide Time: 11:18)



But what it is saying is in the whole switching network of k stages, when the packet enters and packet goes out that is a period which is min-d. So, one lot of packet comes in; it takes min-d to go out, and once the min-d is over, then the next packet will come in. So, there is one packet is being injected at each input or if p is the arrival rate or the probability of having a packet on this input side; this is p num fraction of packet on an average are being injected per min-d slot.

So, min-d is kind of a slot but actually it is not required. In one slot, if I am moving from one stage to another stage. So, I can use pipeline; when the packet moves here, the another packet from input can come in here. So, technically my number of packets has to be this particular slot; that assumption has not been taken while computing the throughput performance.

So, throughput actually can be k times technically than what I am writing, okay, if this pipelining has also assumed. So, pipelining is not assumed in the paper; that is one thing which is done there. So, throughput will be nothing but p raise to the power n, which is probability that you are going to have a packet at the output link. And total number of output links will be 2 raise to the power n, and these many packets will go per min-d slot. But need not be per min-d slot; it can be per min-d divided by k.

So, k times higher actually it can be achieved. So, that will be the throughput; I am just going to state whatever they have told. So, this will be the throughput for all unbuffered delta networks and that for that matter all banyan networks so far that condition is satisfied; that no input and output links in the intermittent stage are left open and you have exactly one path from each input to each output. And these as we have to just estimate what is p n; what p n has to be done through a recursion. You cannot make a direct estimate once the p at the input link is given.

So, in this case now every stage you are going to have a switch remember and you are connecting them like this; whatever way I am not bothered currently, something must have been done. Now each of this switch can be of a by b size general cross bar, or in this case, it has been taken as b symmetric case; it is not asymmetric, you can solve for asymmetric case also. That is also fine, where a and b both are not equal.

Now number of packet which are going to come are binomially distributed. They are b inputs as a single unit cross bar remember which is a simple switch. So, for a cross bar of

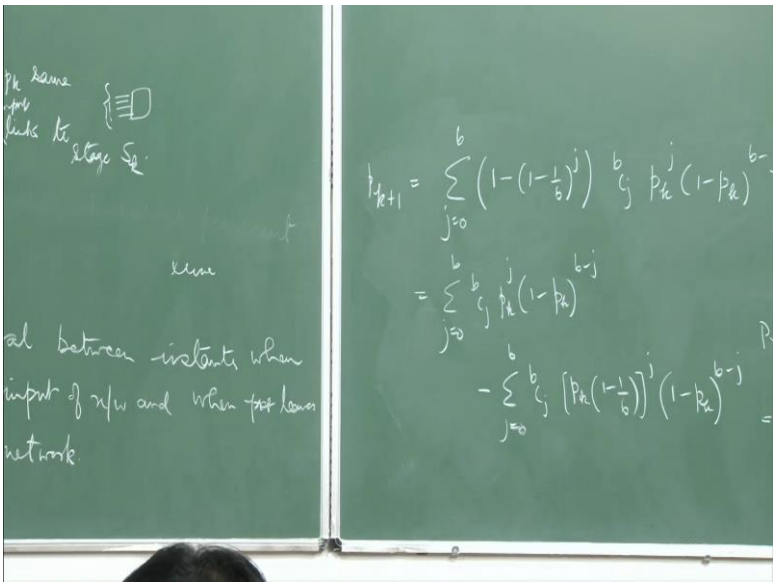
actually you can do it for in general for a by b also; this paper does for b by b. So, the packet switch will arrive will be binomial distributors. There can be j packets which can be arriving, and this arrival can happen with the probability; here the probability is p k remember. Probability that a packet will come will be p k at a input port.

So, this actually means b c j; that is a probability that j packets will come and probability that a packet you take an output link. So, this probability is nothing but will be p k plus 1, okay. So, probability you will have packet on an outgoing link; that probability is there has to be at least one packet out of these which is selected here. This can happen for all outgoing links with equal probability.

So, this can be estimated as probability a packet, and this is under the condition that given that j packets arrive at inputs. So, this value will be nothing but you will simply say that each packet. Now this is a conditional probability; j packets have already arrived. I have to multiply by this; this is a conditional probability remember, okay. So, this probability is that out of these j packets which have arrived under that condition, none of them is being directed to an outgoing port I can find out that probability. And each outgoing port is going to be selected with equal probability actually the 1 by b.

So, you will have 1 minus 1 over b; that is the selection probability see, and 1 minus 1 over b that you are not going to select an outgoing port. None of the j packets will be going to select an outgoing port is this and 1 minus of this at least 1 is going to select an outgoing port 1 minus of that. So, that is the conditional probability that packet will be there given j packets, okay. And then, of course, you can find out what is p k plus 1 from here.

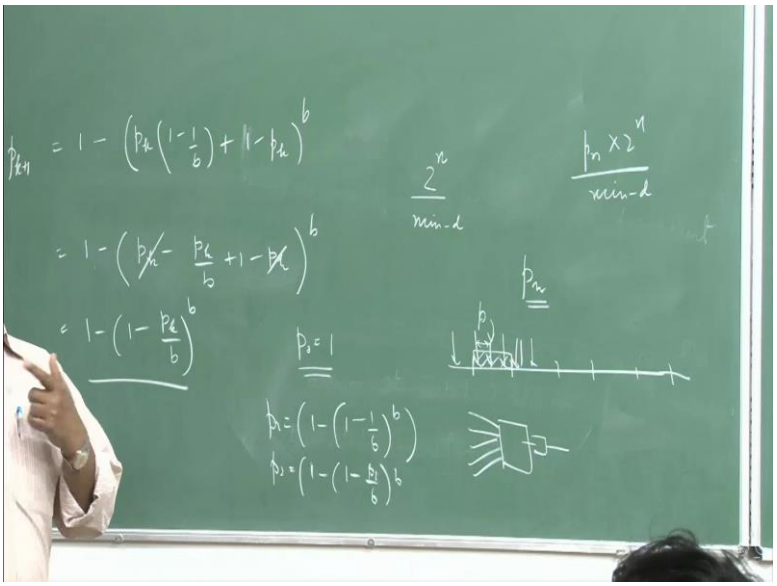
(Refer Slide Time: 17:57)



So, from here p k plus 1; so, j can go from 0 to b. And of course, there is a conditional probability 1 minus 1 minus 1 over b j and I just multiplied by whatever is the condition which was b c j. So, this is absolute probability that there will be at least one packet at the outgoing port. I have just summed up with all possible j's; condition thing I have removed technically. So, this actually can be solved.

And this will turn out to be. So, this whole thing now can be written as this one actually I am taking summation of 1 minus 1 over b j and p k I can combine actually. So, this term is of complete binomial. So, it will become 1. This one is also I can always write a raise to the power j and b raise to the power p plus q raise to the power b; in that form I can use, again it is a compile binomial, okay. And based on that, I can get the solution. So, it is an iterative thing which you will get actually.

(Refer Slide Time: 20:06)



So, it will be 1 minus. So, that will be the probability of p k plus 1 in terms of p k; at this I think you can also figure out one is for input q's we have done similar thing when the packets were dropped, okay, except this p k was converted to something else there. And of course, you can always take for the maximum loading condition p 0 can be taken as 1. And from there I can find out what is the maximum possible throughput which you can have. So, even in buffered delta will always throughput which is lower than this; that is the maximum which you can get, this is not maximum, sorry, this is not maximum.

So, this throughput is p 0 is of at one you can compute, and maximum throughput is all outgoing packets will have a without conflict this is a maximum throughput, you divide by this actually. So, normalize throughput you will get is nothing but p of n which you have to get from here, but this has to be solved through recursion. So, p 1 will be 1 minus 1 minus 1 over b because p 0 is 1. So, this is going to be smaller value actually; every time the value will keep on reducing decline actually.

So, every step you will do it after k stages, find out what is the value of p n, and that is your throughput; that is the normalize throughput the whole p n. You are dividing by this value. Actual throughput which I define was p n into 2 raise to the power n by but maximum which you can achieve is when all outgoing ports will actually have one packet every min-d slot. And if you allow pipelining, you multiply it by k.

Student: That k is the number of slot in the minimum d.

Min-d yes contents.

Student: K slot.

K sub slots. Each sub slots you go from one input stage from the input of a stage to output of the stage for every stage.

Student: Sir, you have to sum up main recursively means p you have assume p k plus 1 and then further you have to go or this p k 1, p k plus 1 will be.

You know how to get p k plus 1 from p k.

Student: Yes sir.

So, if you know p 0 is being 1, another maximum loading condition, what is p 1 you can find out.

Student: Yes sir.

So, p 1 will be?

Student: Here you need to sum up all these after this.

Why you need to sum up? You are getting absolute values.

Sum up is only p 0 p 1 p 2 p 3; all values are there. You do not know their values but you know their relations. Then use last one when you will say sum of all probabilities has to be equal to one is the last equation, which make sure number of equation becomes equal to number of variables to solve each one of them. Here you are getting the values actually directly; there is no Markov chain actually, it is not a Markov chain.

You will put 1 over 1 minus b. This is the p 1 with value be p 2 you can find out now putting p 1, find out p 3 p 4, so on iteratively is the recursive relation which you can get at best.

Student: Sir why you are using this expression?

That is what I get.

Student: There are n inputs you can use exponential or even percent ratio.

Why, I am not worried; I am not worried about the input statistics. Input statistics is the packet is always there.

So, in the input stage it cannot be exponential or poison, remember; there is a reason for that. I am talking about a discrete event discrete time system in slots. So, packet can only start here and it has to close by one slot. So, only statistics which you can use is either a packet can be there or cannot be there. So, this is nothing but p; we use that probability. So, on an average how packets per unit slot you will have? It is p; p is the probability there, and that is the arrival rate per line.

So, most of the discrete time system that is what you are using going to use for packets ways packet switching system; packets are not coming at any arbitrary time interval. If that is permitted, then that is a different situation. In our case it is not; it is a fix length packets coming one packet can come in one time slot, you cannot have two coming in actually.

[FL] Those systems actually can be done, but usually the way we do it in that case is if you have a switch for example, and there is one outgoing link with buffer, there are many inputs which are coming in; they are time synchronized. All these packets which are coming will be put in one single queue. Now since they are not time synchronized, I cannot get in this case technically what I am assuming, because all slots are synchronized; they are fixed length.

In one time slot, I can get either zero packet, I can get one, I can get two, I can get three; that is the binomial thing, but if I do not I am not permitting synchronization and all packet are of fixed length. Then packet can arrive at any point of time on time scale; all of them need not come simultaneously. It is not zero, one, two, three in this slot. So, packets will come sometimes here, sometimes here, sometimes here, then I talk about interarrival time which is exponential distributed or alternatively a poison distribution that how many number of packets will come in per unit time.

Because these are all independent, because of that independence, I can take that assumption not at a source; source usually actually have a correlated what we call arrival process. But as you go into the core when the packets will get dispersed and packets are coming from various different sources, then only poison statistics can be assumed.

Student: Assumption like number of packet tends to infinity and probability of each finite arrival means very close to zero, then I put a Laplace is equal to Poisson distribution. In that case you could have turned out to a limiting case of binomial distribution, sir.

Fine, but then what is happening is you are assuming this time slot to be.

Student: Very small.

Very small. That is what we are technically doing.

So, contrast time variable is nothing but a limiting case of this, when packet can arrive not at discrete intervals but at any time interval at any time instant. A real life always works this way; it is not this but if there are many incoming lines, you cannot control what is happening in the behind, you can always use it as a poison thing. So, when you look at this kind of queue system, okay, then you will say that the arrival at statistic here is poison, but that is not matter; we will always be doing the poison this thing binomial only.

Student: Poison will not give the length of the packet it will give the only unit time.

Poison will give only the number of packets which are going to come per unit time.

Student: Per unit time.

The length of the packet will tell you what will be the departure process.

Student: Departure process.

If all packets are of fixed length, it is a deterministic departure process.

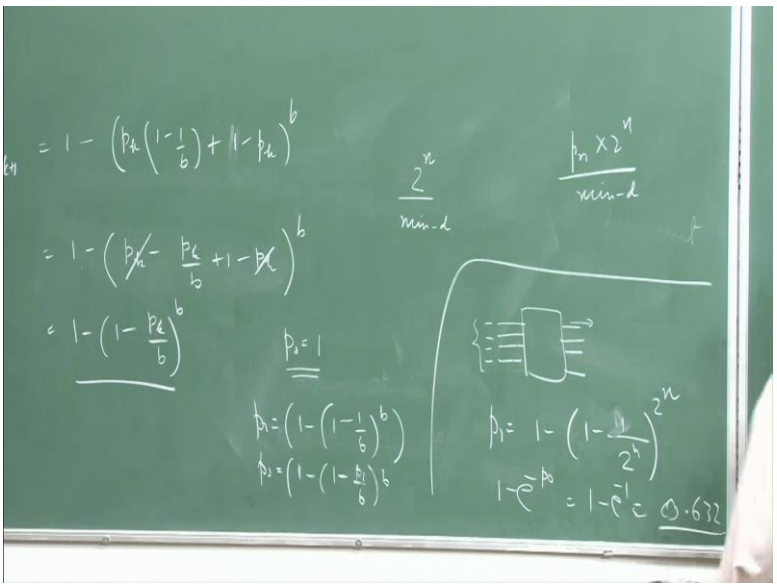
So, you know that what is going to be, it is a fixed length. If you know packets are only of length l and l 1 and l 2, they are only too deterministic departure time; either it will take time t 1 or t 2 with some statistical distribution. And if it is exponentially distributed, that is what we assume most of time. Then you will define this thing as 1 over mu will be the average processing time or number of packets which can be processed per unit time will be mu is the departure rate.

But we are not looking at these cases here. Important thing is that we are going iterative thing to build up the outgoing probability, because you will be able to get p n from here and henceforth the throughput performance.

Student: Sir, what is the explanation for in unbuffered case you will be having throughput higher than the buffer phase. It is because in unbuffered case.

Buffered case you can see every time the probability is going down.

(Refer Slide Time: 29:50)



So, I am actually assuming if there are j packets for example; I am taking this four ports. I get all four packets, all four wants to go to this outgoing port; only one of them will be going. So, remember when I am looking at the probability, I am actually telling even if one or more than one if they are trying to go to an outgoing port, one will go; that is how I am estimating p k plus 1.

So, it means remaining is being dropped while in case of a buffered delta case, this is not going to happen. And of course, we are using buffered delta model we are using back propagation of the signal. So, that the input itself at least you will get what is the maximum throughput which can be achieved under maximum loading condition. In this case, you put maximal loading, the throughput is limited to p n when p 0 is going to be 1.

We will find this as a maximum throughput; cross bar I need not do this iterative calculation. Cross bar in this are only different in what? There is calculation for a cross

bar technically, okay, b by b cross bar. But I am using recursive nature of that. So, every time probability is falling down because of recursion. In case of single cross bar, same thing will now become your p 1 is 1 minus 1 minus whatever was p 0 2 raise to the power n 2 raise to the power n; that will be your throughput performance.

So, p 0 you can make it 1 if it is a full loading condition. So, this is what will be your value. And of course, when n becomes very large 2 raise to the power n becomes very large this can be approximated as. So, p 0 you can put to1 1 minus e raise to the power minus 1 which is 0.632. So, that is a maximum throughput which you can achieve with the cross bar single cross bar. The whole switch is implemented using a single cross bar, or I am using it implementing by multiple cross bars by creating delta network

Student: Sir, p is equal to 2 raise to the power 5.

Yeah, there is one single cross bar being used for the whole switch.

So, unbuffered delta we know that how we will compute; for this one we know how we are going to compute, but this has to be done computationally. There is no closed form expression which can be plotted. But these are exit computational results which will be there.

Student: Sir, p is calculated; first is the unbuffered, delta the throughput is p n.

P n, normalized throughput will be p n.

Student: 1 divided by this maximum.

So, similarly you can also normalize this also if you wish; you have technically normalized this when you say p 1. So, maximum is whatever is 2 raise to the power n divided by min-d. So, that you have normalized; p 1 is your throughput where that is a p n. So, p n certainly will be less than p 1 or more.

Student: Sir, what we want for all to build a network, the performance under those scenarios, performance is almost same, throughput.

Throughput performance will be same for all delta networks.

Student: All delta networks under those condition what was mentioned.

Under all blocking condition because there is no blocking technically. The switches are all symmetric in one single stage, and all inputs are independent. If the independence is not there, then their performance will be different. Then you have to assign the input ports such a way that if there are two correlated inputs which are always going to pump the packets; make sure there is two correlated packets are always going at different switches in the different stages. They should never come to the same stage, because they should not try to pump the packets simultaneously.

Student: Second case we have calculated for?

It is a cross bar implementation.

Student: No in p k plus 1 we have calculated.

This one.

Student: Using the cross bar.

This is a single cross bar implementation of 2 raise to the power n by 2 raise to the power n switch.

Student: First pervious one p k plus 1 it is?

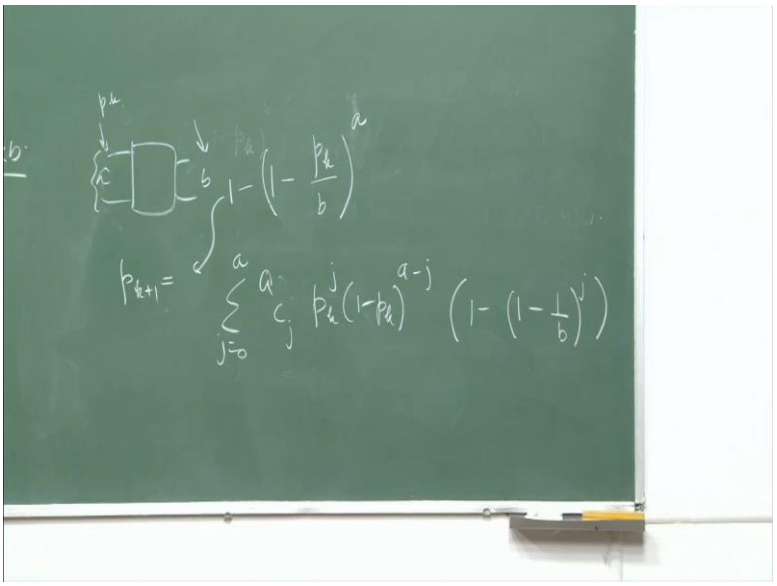
That one I am using b by b switches to create 2 raise to the power n by 2 raise to the power n switch but delta.

Student: Sir, there is one doubt in this; across b if you a have that is r, throughput will be same or different?

Formula will become different; that is the only thing. Formula will be different in that case.

Oh, you can compute, not an issue; you can compute for a by b. I should have done that, because that is done in the earlier paper. Actually I have not done the analysis at that time, because that was anyway required to be done here, but this paper only talks about the symmetric ways.

(Refer Slide Time: 35:05)



So, for a by b it will be. So, what is the probability if you have a probability p k known here, what is a probability that you will get at least one packet on this side? So, each packet is being uniformly distributed over this thing, okay. So, number of packets which will be j, you can actually compute that. So, p k by d that is a packet will come to certain outgoing port, okay; none other packets are going to come to this outgoing port 1 minus of this and all possible j scenarios actually. No, all this is will be a, sorry; this will be a all inputs.

This one only tells that at this particular input I will have a packet with probability p k; with p k by b this packet will be directed to this outgoing port 1 minus of that this packet which is going to come here will not be directed here; this should be true for all a. So, I have to multiply it by a; it is not b a, b b here; that is the only difference which will come. And 1 minus of this, there will be at least one packet which is going to come out here. So, your p k 1, this will be equal to p k plus 1. So, this recursive relation will change in this case.

Student: Sir this recursive relation is for buffered.

No unbuffered; there is no buffering here, buffered delta is tricky to held up.

Student: When we normalize that relation by p it is second 2 power n minus 1.

[FL] Remember in the switch, total outgoing ports are 2 raise to the power n; best case can be when every min-d slot one packet is going out, if one packet is going every min-d slot.

So, throughput performance is 2 raise to the power n packet per min-d slot; that is a maximum throughput. So, whatever throughput actually compute divide by this to normalize it. So, that is what is being done. So, you divide this by 2 raise to the power n by min-d, you will get this; sorry, you will get p k. P n will be the throughput; I have written somewhere here. So, this will be p n will be the normalized throughput.

So, I just find out what is the value probability at the outgoing port or port throughput I am estimating technically, okay; per port maximum throughput is 1. So, I am just estimating that p n.

Student: So, in the previous case we took 1 minus 1 to the power of j. So, that is j is the count of number of packets which are coming at the input.

I could have directly calculated also by radiate explicitly for j packets come with condition on j.

Student: So, since a is the number input lines, should not that be a p k, sir, at the exposure. It should be raise to a p k, because a p k will give the number of packets.

Number of I am not clear.

Student: Arrival rate is p k, sir.

As you are telling, this has to be p k by b raise to the power a p k.

Student: At least it is raise to that exponential where it is number of packets which are there, sir.

Okay, let me compute with j packets; we will show on the other way around. This is a direct calculation.

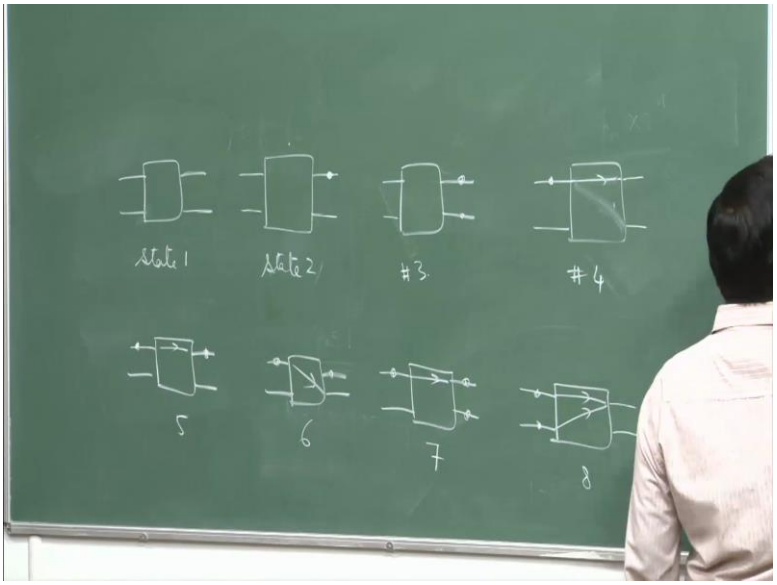
Suppose j packets are going to come here, what is the probability for that? A c j; when j packets have arrived, what is the probability that at least one packet will be coming to an outgoing port. So, these are conditions. So, that probability is 1 minus 1 over b j total j

packets, sum it over j; j can go from 0 to a, remember, solve it; you will get exactly same thing that p k will not come in the exponent .It is just a way to understand actually. So, both are fine, no questions so far.

So, next is now we have to go to the actual analysis of buffered delta system. Now buffered system I have to now look into what we call a state transition of switch. So, before I move onto that particular thing, I have to define the switch state. Now remember they are buffer switch are involved here. So, there all possible combinations are there. So, we will define fourteen states of the switches here, and that I will use in building up the recursive relation.

Again this is done through a recursive relation which is computationally computed, okay. There is no simulation; it is an actual calculation which has to be done. So, it is a computational procedure. So, there is no closed form expression as such, but computational you will get the exact values.

(Refer Slide Time: 40:39)



So, the model goes like this; most of the states I am looking into two by two thing. So, one state is when the buffer on the outgoing side which is nothing but the input buffer of the next stage, okay; that is free and this is free; that is one particular possible state. So, this known as state one, yeah. So, I am keeping all the inputs free and then trying to put packets from the output side buffer.

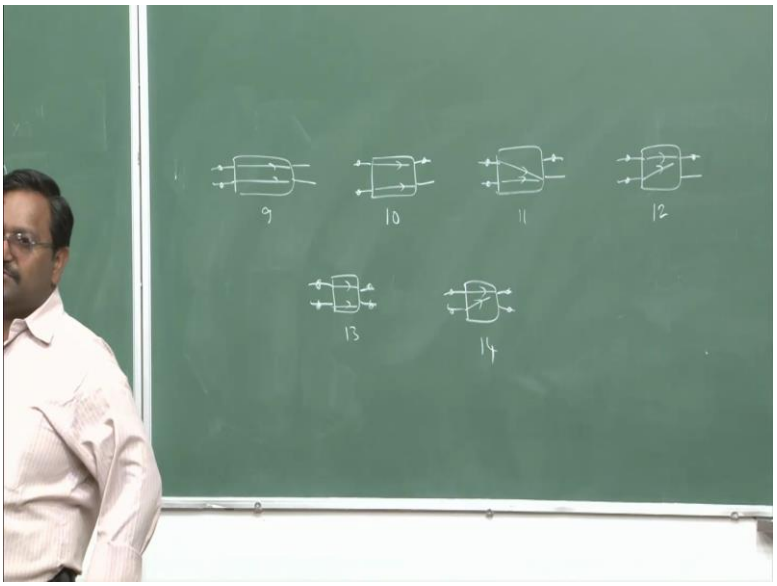
So, one possible states, I am just numerating the states now. When you will have only one packet, now you can say this packet can be here or this packet can be here, but both are equal interstates. So, both are going to be all equivalent states are merged together the way we had done earlier, okay. So, equivalence is I can rotate. So, I can do whatever it is, but input and output cannot be swapped the way which we did in the earlier case, okay, in case of that white sense down blocking system. So, this is the state two, okay.

Third one will be now, when I will have two packets which will be there. This is state three. Whenever the packets are there at the input buffer, they will be directed to some outgoing port. There was no direction of the packet was shown, because there was no packets at the input buffer. So, in this case, this is what will be the direction. Now remember even if I had this direction is not this, I put it like this way. This is equivalent to the earlier state because I can swap these two ports. So, it does not matter, okay.

So, I will represent this thing by this because these numbers will used remember these straight numbers. I will be mentioning in the subscript later on in that computational model. Then you will have a state. So, I am looking at one input and then one output two output kind of combinations, all possibilities. So, one is this; this is state five. Now this input can be directed to this one; that is one possibility. This can be directed to a free port; that is a sixth state.

Now whether p u direct to top or bottom, it is all the same; there is only one input. Basically, this cannot go, because this is already filled up. So, this is the state seven. Then you have all cases when one input was there in the switch are taken care off not two inputs which are there. So, if there are free ports, one possibility is both of them will conflict; that is the state eight.

(Refer Slide Time: 44:06)



They do not conflict; that is the state nine. So, whether it is a bar state or cross state does not matter; there is no conflict. When two inputs are there, both will be state nine

Student: State seven will be conflicting

Conflict can only happen if you have more than one input, but it cannot go onto the outgoing buffer, because the next buffer is occupied for some reason.

So, no outputs I have taken care of. One, this could be one case; this could be one case, and then you will have a state number thirteen and state number fourteen. Cross and bar states in this case are exactly same; they will all merge into state thirteen, okay. So, whether the conflict happens because both want to go up or both want to go down; both are same states here. They are total fourteen states which we have to define for a switch. So, all switches will always be existing in one of these fourteen states in a buffers delta system, okay.

So, now here I actually close the today's lecture and we will start on looking into probabilities and steps which are required for doing computational stuff for buffered delta. So, that we will do on Thursday; you have to actually refer to the paper, because I am trying to do it as slowly as possible but still there will be confusions.