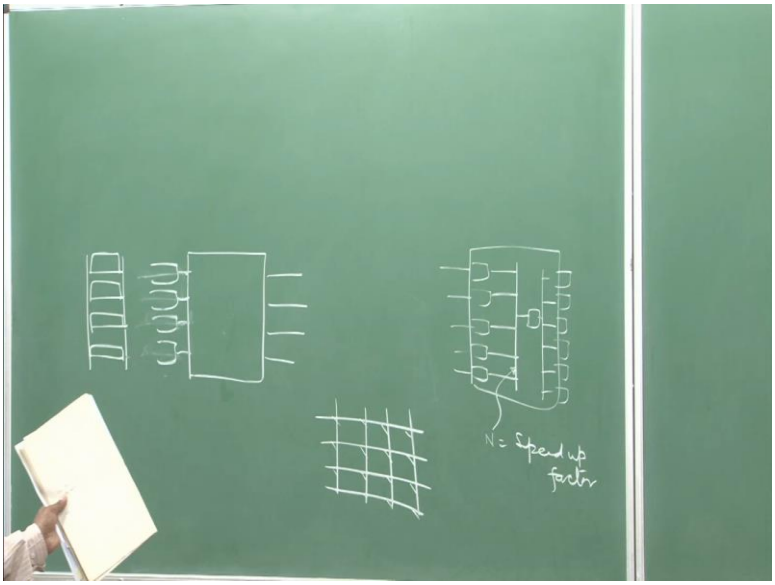
Digital Switching
Prof. Y. N. Singh
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Lecture – 18

(Refer Slide Time: 00:25)



What we were doing last time was the input and output queued switches, and I have actually given the example. Why we were doing input output queued switches, because we were looking at a cross bar, and how to implement packet switching system, using a cross bar; that was basically the idea. What I had actually, mentioned that there will be inputs, and there will be corresponding with the outputs; I am showing it on left and right. Cross bar; we always show from left end bottom; that does not matter, actually. Just for the sake of showing, I am showing it, and what we had was that all the packets will be coming in synchronized slots. So, at the beginning of the slot, you will set the cross bar, depending on whatever, cross connection you require; the packet will move through. Then, the next slot will come. Again, you will do the same thing, and so on.

We will keep on doing it everytime; the packets will be switched out to the appropriate out going ports. Sometimes, it will be possible that two packets will be there, which want to go to the same output; that is a contention, output contention. One of the packets has to be dropped, if it is only a cross bar, because technically the speed at which, the packet or bits are going on this line; packets going inside the cross bar, and packets, which are and bits, which are going on the outgoing lines; they all three same.

So, speed of factor is 1 in that case. One of the packets has to be dropped, or it has to be buffered, if you can keep memory here. If you do not keep the memory here, the packet will be simply dropped. Only one of the many contending packets will be going to the outgoing port, and of course, then I told we can probably, improve this situation; try to minimize on this loss. I can provide some kind of buffers or the queues at the input port. So, the first idea is that I keep the buffers here. These are known as technically, input queued switches. Only important thing here is still, my switch is operating at the same rate at which, the lines, input lines and output lines work. That is, I think I have mentioned earlier.

If there is a contention, I actually, allow one of the contending packets to go to the outgoing port; remaining packets are kept in the buffer. So, it is possible that some packets, which were written in the buffer, cannot go even in next time. So, what will happen? The packet, which is coming in that particular slot, will be queued up in the back of this particular queue, which is held up. This could have been transmitted, but cannot be, because somebody is there, ahead of it, which cannot pass through. Then, of course, I gave an idea that instead of one queue, I can maintain multiple of them for separate outgoing ports. I will try to resolve, as for as possible, or backlog the stuff, but if cannot happen; I will use whatever, are the other free outgoing ports for which, whatever, packets are there; I will schedule them. I said that instead of one queue, you will be maintaining multiple of them like this, for each outgoing port, and will be picking up from there. So, this can take care of ahead of the line, blocking, actually. But FIFO (( )) inside this queue together, is not going to be maintained; that is only thing, but this will work, very effectively. Another option was that I can, if somehow I can actually, have a speed of factor inside the switch.

The speed of factor cannot be there, if it is a plain cross bar. If I use only a cross bar like this, and there are cross points ,which are connecting here, which are triggered for every slot, depending on whatever combination is required, sorry, these lines were going to work at the same rate at which, you are receiving and the bit is going out. There is no speed of factor; it is 1. If I actually modify this switch; I have not mentioned it explicitly, but that was an assumption; that packets do come at the incoming port in the switch. They are buffered; there is some limited buffer. So, may be 1 packet is buffered for every input. Input queues can still be there; I am not bothered about them, but what happens, I

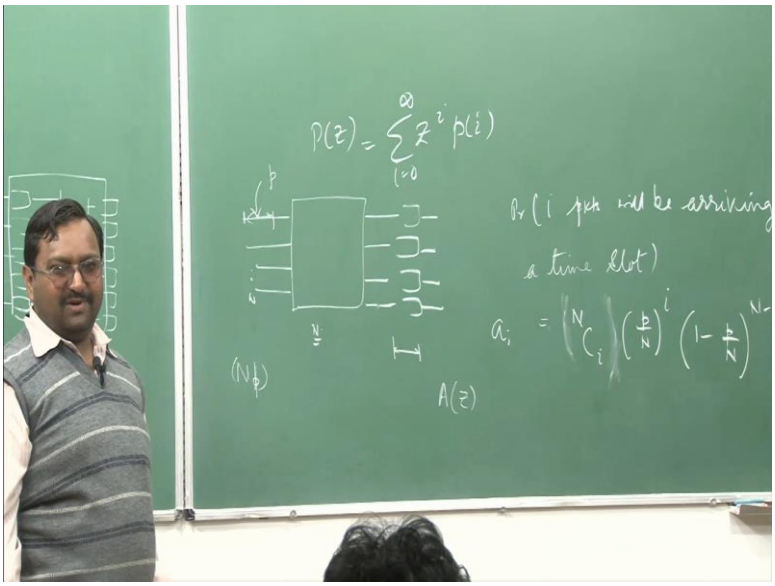
actually, have some mechanism now, which is not a cross bar; it is like; with cross bar also, I can do space switching, but I can also do time switching; I can read into memory and write kind of thing also, can be done.

I am not looking at a cross point. Cross point is a circuit switching scenario. I am looking at transferring a byte or a slot. Time switching was possible, because I was looking for a octet being transferred from one point to another point. In time or in a space; both ways, it is can be done. Here, if I can actually, run a processor, I can actually, connect a bus, for example, and this bus, I can read from this bus, using a processor. If this bus's speed is such that I can read all these packets in one single time slot; that is possible. So, this processor can read this packet in one; if there are n packets coming in per slot, one by n th of the slot; this packet can be read by the processor. It can be analyzed and then it can be pushed on to an outgoing bus, if it is the dual bus system, and can be pushed on to the outgoing line. So, there can be output buffers also, but this is not a cross bar, but technically, it is still a switch; n by n switch, packet switch.

Alternatively, it can also be a cross bar, but then cross bar has to operate, within one single slot period, technically, there will be n such small minor slots. So, you will be putting the packets here, at some line rate, but when you will read out the packet; that will be done at n times faster rate. So, each packet will be scheduled in different sub slots, correspondingly. Then also, I can use a cross bar, and I will get all packets here, at higher in speed and time, higher speed. I can put them in the buffer. Outgoing line rate will be same as incoming line rate; that also, can be done. So, there are various ways of doing it. This bus now, can address and can actually, push the packets into various outgoing things. If this is working at n times faster rate, it is n times speed up factor; its speed up factor will be n times. They are always kept as same.

I think that is what we have done, more or less in the last time. Now, we have to go to the analysis part of it. One of the important things is that what will be the utilization, and what will be the average queue length, which will be there, for input queued system and output queued system.

(Refer Slide Time: 08:17)



Let us look at first of all, the output queued system. There is a speedup factor of n, which has been assumed. Even, if all packets are there, they want to go to the same port. It is possible that I can read out all these packets at n time's faster speed, and write into the outgoing buffer; that is technically, possible. Now, look at the outgoing buffer; what happens there? We have a higher clock rate. It is like, what is this buffer? Buffer is technically, a RAM. So, what you do is you actually, have dual system here, in the input side, or here. You actually, first of all, read into this particular RAM. Then, the next slot you are writing into this RAM, if the packet is coming. Only you do not require any actually, memory, if you have output buffering. On the input side only, you will do this, and this packet will be read at very fast rate.

So, this has to be now, addressed by the processor, and read out in faster rate; n times faster rate. So, I can read out the whole packet in one by nth of the fraction; one by n of the packet duration, slot duration; I should be able to be read it. So, I can read all the packets; I can address one by one, each one of them; read it out, analyze and then transfer to the outgoing buffers. If all of them want to go to here, it is fine. The first, I will read here; next, I will read here; I will do addressing. As I am doing it at n times faster rate, I should be able to read all the memories, all the RAMs. While, in the input side, the packet, which is coming has been written to second memory, and what will happen is I will simply do; once this has been read out, next slot; you will be writing

here, and the read out will be happing from this. All addresses, everything remains the same.

We just do a very fast flip flop. Actual implementation will be something similar. You cannot read and write, simultaneously; that is very important. You have to first of all, write the whole packet. Then only, you can do the reading. Reading is done at a faster rate. So, packet is compressed in time, technically. There are multiple memories, which will be there in each of these inputs. So, processor can do that. So, I am taking this particular scenario.

Student: That if cannot happening in writing now verticals on the language.

No, that depends on the line rate. You do it at faster rate. Then, you have to increase this speed here also, to ensure that all packets are read out and written. Even, if there is a contention, all n packets can be written into one slot, because you are doing it n times faster rate; you do not require a cross bar. You can read all packets and then write down at appropriate locations. Cross bar is required when the speedup factor is 1. Even, you can build up a hybrid. You can do a cross bar and a speedup factor of k. So, cross bar size required will be actually, smaller. Some that kind of a scheme can also be built, but I am not actually, looking into that as of now. I am worried about the output queued system, first. How this will be modeled? In this queue, how the packets are actually, coming; that is first question. So, in every time slot, it is not that one packet can come in this queue. I may get no packet; I may get one; I may get two; I may also get n packets in the queue. So, there is a whole batch, which arrives at every slot, and this best size itself, is variable now. So, if you want to estimate a probability that I packets will be arriving in a time slot, this probability will be what? p is the probability that a packet will be there in a slot.

So, on an average, n into p; these many packets are coming per slot into the switch. Now, each packet is independent, which is arriving; so with equal probability, I can assume. It will be going to any one of the outputs. So, with 1 by n probability, it will be directed to any one of the output. From here, this guy will get a packet with probability p, divided by n. From here, again p divided by n, and so on. So, from any output port will get from any input with probability p by n. Sum, total of all possible scenarios has to be equal to 1; that probability, that actually, will be ascertained here. These are uniform (( )).

Student: Previously, you talked about n minus 1 port sir. p by n minus 1 individual probability outside, previously packets.

Yes, right. Now, I am not bothered. This is a transit switch kind of thing, but you are right. I should not be sending a packet back to myself; that is usually, is true. That way, I think some other case where, we consider this. I am keeping a time also, does not matter, because I know, I will ultimately, be taking n tending to infinity. So, for that kind of hypostatical switch, we will try to compare the throughput performance.

Student: that is piton circle is saying.

This speedup factor has to be n, if the output queuing has to be implemented; otherwise, output queuing cannot be implemented.

Student: why?

It will become a hybrid system. Its speedup factor is k. There is a probability, finite probability that all packets, which are coming, are directed to one port.

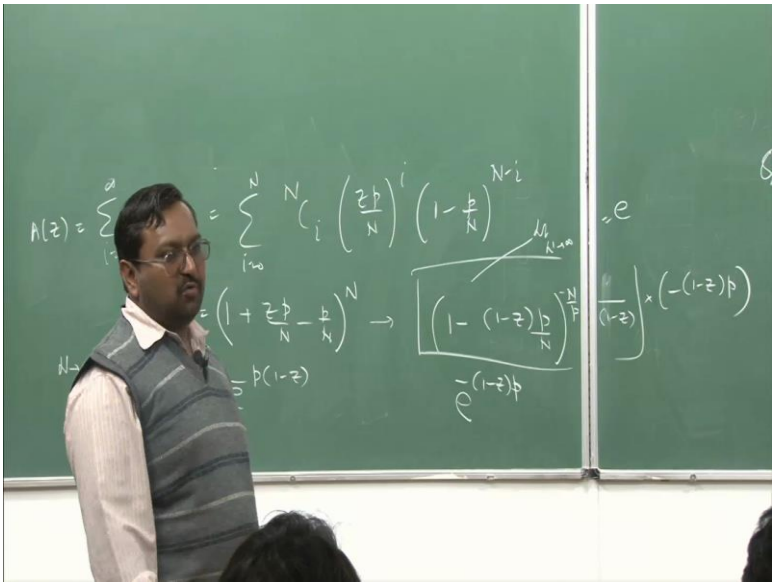
And these all packets cannot to be pushed to one port, unless I am reading at n times faster speed. So, in one single slot, I am going to read n packets. That is why n times more speed is required inside the switch. Then only, output queuing can be implemented. If you are ensuring, there is a mechanism or discipline or whatever, is by which you say there will never be more than k number of packets being directed to one single outgoing port, where, k is less than n. If this can be ensured somehow, or your traffic tells that thing, then only k times speedup factor is good enough to implement the output queues, but then of course, you require a cross bar, because you are doing parallel transmission. Here, only one packet is read and one is being transmitted out in one sub slot; in their n sub slots. So, all packers are read out and transmitted to all outgoing ports.

But there, you will be requiring cross bar, because there k. There may be actually, n packets coming; only k can be read in that sub slots; k sub slots are there. What happens to the remaining packets? There has to be some kind of a cross bar mechanism, which is required; parallel paths. So, this probability will be; this is n c i; clamatorial n factorial, divided by i factorial, divided by n minus I factorial. This actually, comes from basic elementary probability theory; p by n is a packet arrival. So, there are I packets, which

are arriving, I am saying. This is I packets. So, this has to be i. The packets are not directed to me; that probability is this; n minus i, and I call it a i. From here, I can find out a probability generating function; a z. I will be directly, using certain results without proving them, because those, you can then you have to refer to the probability theory for the proofs.

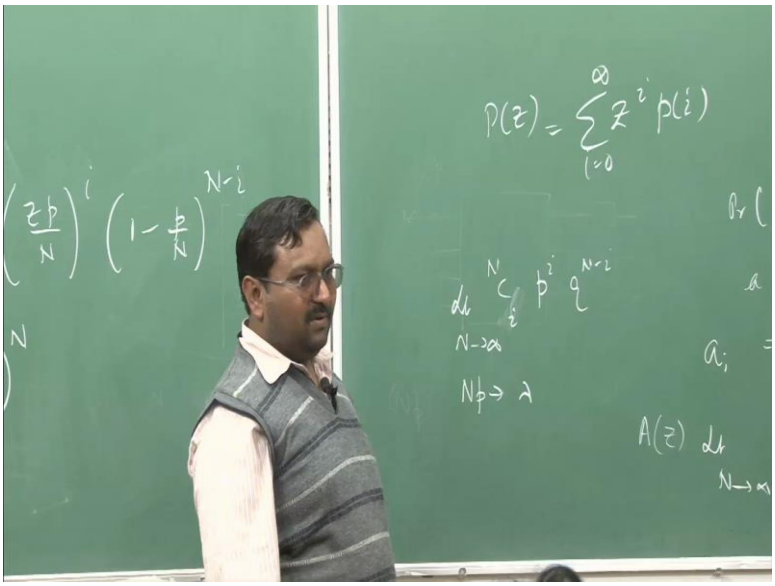
I think, each one of you understand what is the probability generating function for a discrete probability. These are discrete probability; I can only take this speed value 0, 1, 2, 3 to n. The probability generating function is usually, defined as z raise power i probability of i; i goes from whatever, is a valid values. That will be your p of z; that is your p z f. It is like g transform, honestly speaking, is like g transform, but this has a property, because once I know g transform and when two variables add, I can actually, do to find out the g transform of the combined stuff.

(Refer Slide Time: 17:52)



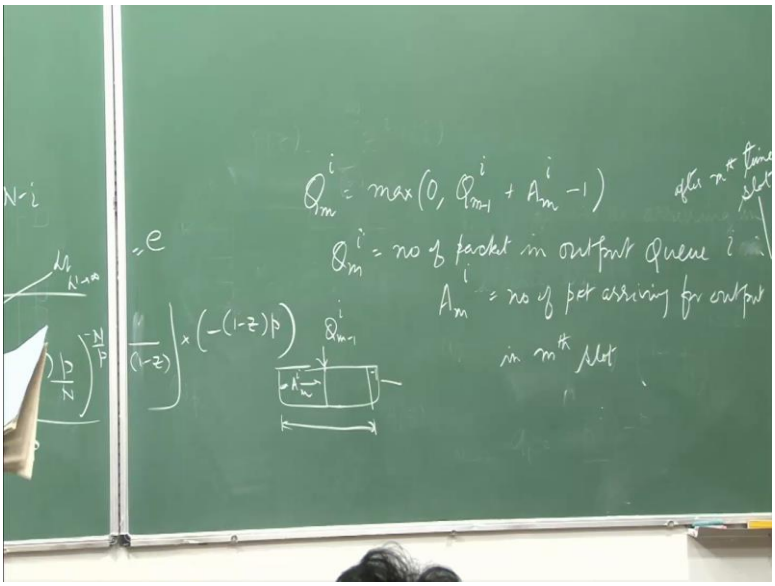
So, I will use those things, directly. So, a z, in this case, will be given by; you can actually, plot. Put whatever is this a i. This is what will be a z. You can actually, solve for this. So, this will be z p by I had just included z inside; and this is a complete binomial, going from i is equal to 0 to n, because there are no other possibilities. So, I can write this. This is what will be the g transform.

(Refer Slide Time: 18:52)



Of course, if n goes to infinity, in that case, a i will be what; a poison statistics. Thus, I think, it is also a standard result. This is what it will converge to. Remember, this probability n c i p raise power; you must have done in terms of p and q, I think. So, when you do this, your n into p convergent goes to lambda kind of thing; you must have done that way, and this then converge on to lambda raise power i, e raise power lambda, divided by i factorial; so personal statistics. I am actually, technically, use same thing. Since, it was p by n; so capital n to p by n, in limit, will always become equal to p. So, for the poison, when n goes to infinity, your a of z will be e raise power minus p, 1 minus z; this also, you can get from here. Take this a z; this itself can be converted to 1 minus. So, this whole thing will become nothing, but e, in limit; that is what I have written, here. This whole thing will be e. So, this will be the second expression. Now, here actually, one important thing, which is there; since, it is a batch arrival, I am not actually, proving; I am just stating all the results straight forward. So, q m will be the number of packets, which are queued up for the queue, corresponding to output i.

(Refer Slide Time: 22:14)



This should be equal to maximum of 0 or whatever, was the earlier batch, plus whatever, were the arrivals in the current time slot for output i, which comes from the probability distribution, which I had written earlier, minus 1, which will be successfully transmitted. Sometimes, it is possible that you have only this sum is 1; 1 minus 1 becomes 0. So, this value can never be smaller than 0. That is why that maximum operator has been put; maximum of the two. So, this is the batch arrival thing.

Student: Sir, explain the argument inside the max; what is the second argument mean?

A m I; a m I is number of packets, arriving for output i in m th slot. So, that probability distribution is given by whatever is there. This is going to have this particular g transform; a z is nothing, but probability generating function for this variable, a m i. Q m i means the number of packets in the queue, and this is a transfer.

Student: i, we are taking as the number of packets arrived, sir.

No, this i is summation, after once you do, it is all gone; is a dummy variable.

Student: This i is gone, sir, this i is taken as different.

This i for the output.

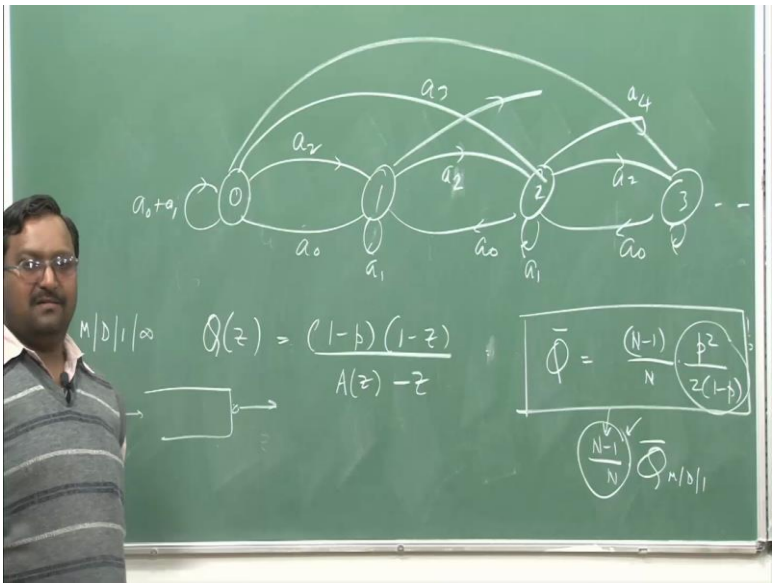
i is very commonly used; i and j are use for dummy variables for summation. So, I think you should be able to, by context, should be able to clear out. The q m is number of

packets in output q, i in m th time slot; after m th time slot, actually. After m th time, because m th time slot, you will be getting the packets; this is at the end of m minus first.

Student: Sir, taking q n minus 1 in a summation term, there I am assuming that whatever, packets are buffered in up to n minus 1 th slot; that can also come in the n th slot.

What is happing is you have a q at n minus first slot; there was some size. This size was q of m minus one i. In the m th slot, what will happen? New batch will be coming. So, that batch will now fill it up. So, this size will be q i m. In this time slot, one packet will also go out in m th time slot. So, I just remove this is minus 1. So, this is the size, which will be there, left at the end of the m th time slot; that is what it means, and of course, you can build up a markov chain, and if you try, you can solve, build up a balanced equation, but I am leaving it; I am not doing that actually.

(Refer Slide Time: 26:21)



The Markov chain for this particular q will look like something, for example, you are in 0 state. If there is no packet coming, you come back to the same state; probability, that packet arrived is 0. Probability, that one packet comes, it will also go out in that same times slot. So, at the end of it again, you will be in 0; a 0 plus a 1. The same probability is a i, which I have done form there. This is no rule for us, but anyway, I am still describing. I will come here, if I have a 2. One of them will go out, other one will be queued up. So, that will be the queue state. I can have 2; I can have 3, and so on. From 0, I can come here, if I have a 3, a 4, and so on. If I have a n, I can go to n minus first state

from here. From 1, I can come back. If no packet is there, one packet will go out. If there is one packet, I need two packets to come in, because one will be going out. One packet is going out; two packets comes in; a 2. Then only, I will come to step 2, sorry, one packet, sorry, this should be a 1 here; not a 2. Just a minute; it has to be a 2 only. This packet will be going out, 2 will be coming in; so you will be in the state 2. You will remain here with a 1.

Student: a 1 will go back to state 0; a 0 will stay in 1 only. One packet is.

This state means, how many packets are queued up?

Student: That is one.

If there is no packet arriving, which is a 0, what will happen? This queued up packet will go out? What is in the next state?

Student: Yes.

There is no packet, now.

Student: Yes, 0.

0; so 1 to 0, you are coming with this here. If there is only one packet in the buffer, no packet arrives; what will happen? This packet will go out, will be transmitted out. Next state of your buffer will be 0. So, it will be coming back here. So, this is unending like this; a 2, a 3. This is the standard markov chain given in all queuing query books. So, for this kind of system with this batch arrival process, I can actually, build up a; this kind of queue, I can now build up what we call; I basically, require what is the probability that you are in this state; what is the probability that you are in this state; what is the probability you are in this state, and (( )) state conditions. So, you can start with the building of balanced equations, and some of all probabilities have to be equal to 1 with that kind of thing; you can even solve this. So, people have done that. In order to represent all probabilities, I am representing them by a probability generating function. I can generate all probabilities with this; all state probabilities.

So, q z for this; I am just taking the result; I am not proving it. This will be 1 minus p, 1 minus z. Whatever was the p g f probability generating function for a i, the arrival

process; you have to take that minus z, and once you have a probability generating function, I can find out the mean value. So, from here, you can find out what is the mean q length or average q length of the output buffer system. From here, you can find thing that out. So, how you will do it? No, not little's theorem, here.

This is the p g f, and what is your; this is your i bar, or a bar; you can call it, actually. This is the mean. This is nothing, but mean q length, and for this, I know already, this expression. For here, if I can take the derivative of p of z, this will be i z i minus 1, p of i; i goes from 0 to n, zero to infinity, and if you put z is equal to 1; what you will get; this. So, for finding the mean, you simply take the derivative of probability generating function and put z is equal to 1.

Student: This expectation.

Expectation of i. Summation of all probabilities will be always, yes, 1. So, that is why if you put p z, p 1 is always, will be equal to 1; probability generating function. At z is equal to 1, will be always be 1, because then that is nothing, but some of all probabilities. Take the derivative; you can find out first movement, second movement, third movement and so on, and hence for that you can find out any movement around any kind of value; that is possible. So, variance also, can be estimated with this. So, all complete statistics is available here, for a discrete system. Now, similar thing does exists even for continuous time variable; continuous variable probability density functions. So, we also have something equivalent there also, but I am not bothered; I am bothered about only discrete. This is a very commonly used technique. Once I have this, I can, from here, estimate what is q bar. I am not solving it; I am directly writing the results; you can try it out.

Student: It functions standard a the result of queuing theory

Yes. So, you have to look into either, Clendrop or any other book. Clendrop is one of the famous ones; three volumes, I think.

Student: Clendrop.

(( )) Clendrop. There are many books available in the library, you can search, but this is the standard result for batch arrival process. Since, I am going to use only the result to

analyze; I am not bothered about the results derivation. So, q bar will be coming out to be from here, n minus 1 by n; 2 into 1 minus p. Again, this is a standard result. This result, interestingly, is nothing, but n minus 1 by n. This particular part of the result is what you get for a special kind of queues known as m d 1 queues; m d one infinity queues. This actually, means there is a queue. Here, arrival is markov; the departure is deterministic; m d one queue. There is only one server, and buffer length possible is infinite here. For that the average value turns out to be always p square divided by 2, divided by 1 minus p. So, this value; it is nothing, but m minus 1 into what I call the quell thing for m d 1. So, this is the only variation; multiplying factor which comes, because of batch arrival.

Student: put it b m d.

D is deterministic established time; it is basically, constant service time. Deterministic means, constant service time; once what it service time for every packet.

Student: What is the q?

1 is one queue.

Student: What is the infinity?

Infinity is infinite buffer length. If there is the finite buffer length, it cannot (( )) you to drop the packets. So, packet loss probability comes into picture, here.

Student: What is q bar?

q bar is the average q length.

Student: This d bar can determine by how many packets going to the output of the queue?

What?

Student: This d bar factor?

d bar is what?

Student: The service time; you can see service time.

That is a constant slot period; one slot period is deterministic. For every packet is serviced in one slot period; it takes only one slot for packet to be transmitted down. For that this is the thing where, p is the probability that is in the slotted packet, will be there. That is what you will get. So, n only comes here; the batch effect is here, remember. When n goes to infinity, what will happen; n is so large so this 1 can be neglected; n by n will be canceled is as of as, as good as in m d 1 queue. So, batch effect does not matter. So, that is I think, one important thing.

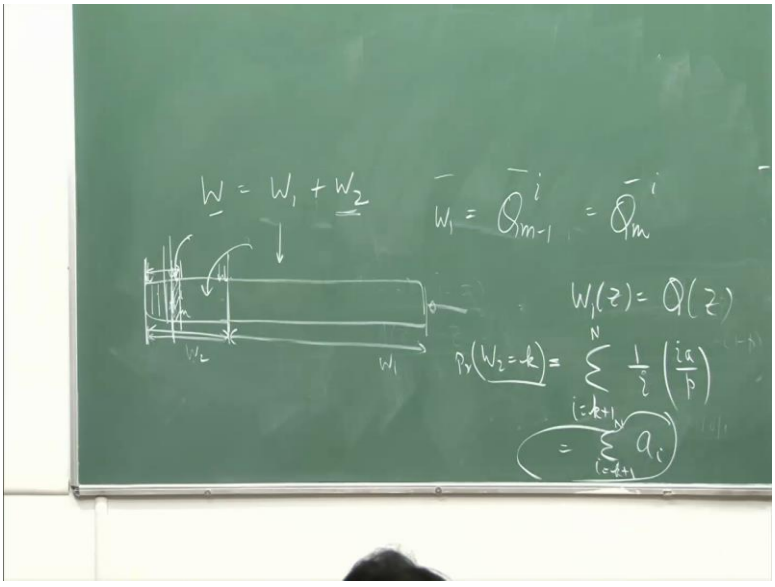
Student: arrival?

It is a poison statistics, is basically, exponentially distributed enter arrival times.

Student: we need a poison distribution?

If you start counting the number of packets coming in a duration; that is poison, but between two packets, what is the gap; that is the exponential distribution thing. The exponential distributed enter arrival time is what is known as markov end process. Now, the question is how much time, a packet will be waiting, before it goes out; what is the waiting time in this switch? Your switch is working at a speedup factor, already. So, only delay in the output q; that is what will be the delay, faced by every packet and that also, can be then estimated.

(Refer Slide Time: 37:26)

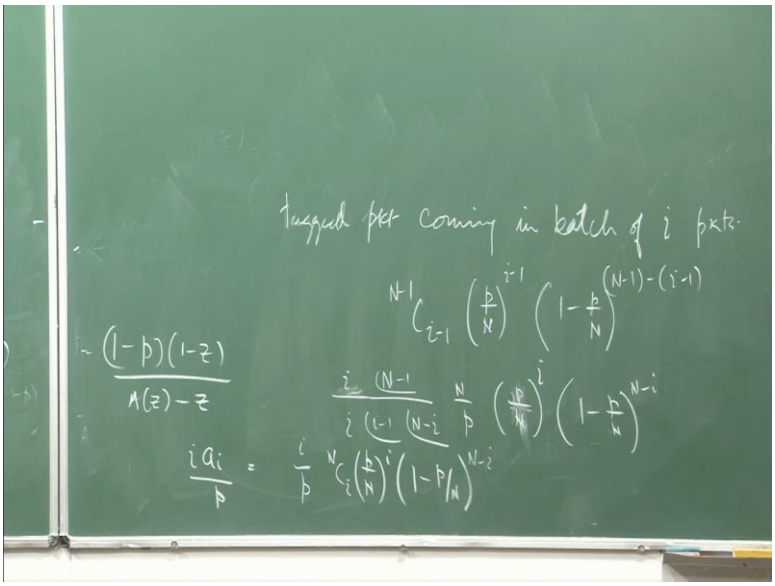


Technically, there will be two delays, which will be there, participating. You do certain packet; you just sit on that packet; you tag it, and then see what suffered by this packet, statistically? Whatever it is suffering; it was suffered by other packets. You are picking up any packet randomly, and tagging it, actually, and then seeing what is happening to this. So, you have to look at all possibilities and from there, you can estimate the average waiting time. So, average waiting time w, will be two parts; w 1 and w 2.

So, w 1 corresponds to; this will corresponds to something, even if you arrive, there is already, buffer has lot of packets. You arriving in m th slot, but whatever, are there in the buffer already, there occupied till this point. You will always coming only in this place; in the m th slot, these many packets have come. So, w 1 is nothing, but this delay. This much we have to suffer; you have no other option. Now, the packet which you have picked in this a m i; this packet can be the first packet. It can be second one; it can be this; it can be this. So, I have to find out statistically, what is the value. Remember, this delay w 1, and this delay, which is going to be suffered here, which is w 2; are independent of each other. So, I can simply add them to find out the average. So, w 1 will be nothing, but q m minus 1 i. Those many slots you have to wait. That will be your w 1, and if I want to find out an average value, I have to just take the bar of it. Once I take the bar of it m minus 1, m, m minus 2; does not matter.

Statistically, this will be nothing, but q m bar, which will come from there. So, what is the w 1 z; p g f of this will be nothing, but q of z; whatever we have solved earlier; same p g f will be used here. The w 2 is slightly, a tricky part. So, w 1 z is nothing, but 1 minus p, 1 minus z, a z minus z. Then, coming to the next one; this is slightly tricky, because we have to now, find out w 2. So, w 2 will have; first thing, whatever your tagged packet is coming in a batch of i, again, i is a separate index here. This is not that number; output port number. Now, the i is again, I am using as a new variable.

(Refer Slide Time: 40:44)



The tagged packet is coming in batch of i; what is this probability? The tagged packet must be coming from one of the ports, and this port cannot send another packet. So, remaining n minus 1 packets, must have sent what we call i minus, remaining i minus 1 packets to this particular device. This actually means, you will have n minus 1, c i minus 1; p by n is the possibility; i minus 1; this the probability with which, your packet will be in the batch of i packets. Here, i is the batch size. So, tagged packet is going to come in that size of i; that probability is this. So, you can solve for this; this is pretty simple; n minus 1 factorial, i minus 1 factorial, n minus i factorial; this I can write as n by p; I have just minus 1, I have taken here. I can now multiply by i and divide by i.

So, I can write i divided by p, n c i, p by n, and this is nothing, but i into a i by p. So, what I want is I want to find out that w 2 is equal to k; what is the probability. For this, what I will do is if w 2 has to be k, I need to have at least, batch size which is greater than k; it as to be k plus 1 minimum. For any batch size, if it is greater than k plus 1, it should be placed in any one of these, always, in this slot. If batch size is k plus 1, for example, the probability it will be at this location is 1 by whatever, is the first, the number in the batch; 1 over i. If it was k by 2, the batch size is this much, but I have to place it here. This packet could have been anywhere, but to be here, it is again, 1 by i. If I take this much large batch size, again, this packet is going to come here; this probability is 1 by i. Your tagged packet can be placed with equal probability anywhere, but at k plus first place, it is going to be placed; that probability is 1 by i, and i has to be from k

plus 1 onward, to till whatever, can be the maximum batch size, which is n. So, this actually, means this probability has to be nothing, but i is equal to k plus 1 to n, 1 over i and probability that the batch size of i, will be there; this is what will be the probability.

This is interestingly, nothing, but this. I think we will close here, now. Tomorrow morning, what we will do is I will start from here. I will build up a probability generating function of this. I already have a probability generating function of w 1. I use w 1 and w 2 as probability generating function to find out probability generating function of w. When w 1 plus w 2 is equal to w, probability generating function of w 1; w 1 z multiplied by w 2 z, has to be equal to w z. So, once I know w z, I can find out what is the mean waiting time in the queue, and I know how this whole thing is going to perform. This is how I think, we model this particular switch.

Then, we look into the input queuing part to figure out what is the best strategy for operating input queue. So, what is the maximum throughput which, you can achieve in the input queued switch. There is a limit; you cannot get to 1; that is not possible, because of the contentions, which happen.

Student: (( )).

See, 1 by i is the probability that you will be placed at a plus first position, and this can only happen, when the batch size is k plus 1. Even, if you can take m batch size, what is the probability, you will be placed there, again, in 1 by i. Your packet can be equally, likely anywhere. So, I have put 1 by i, is that probability, but you are at a position k plus 1 in the coming batch, and there is a probability that you will get a batch of i a; batch of i. Some of all possible things create the average value; that is the probability that you will be having w 2 is equal to k.