Performance evaluation of computer system

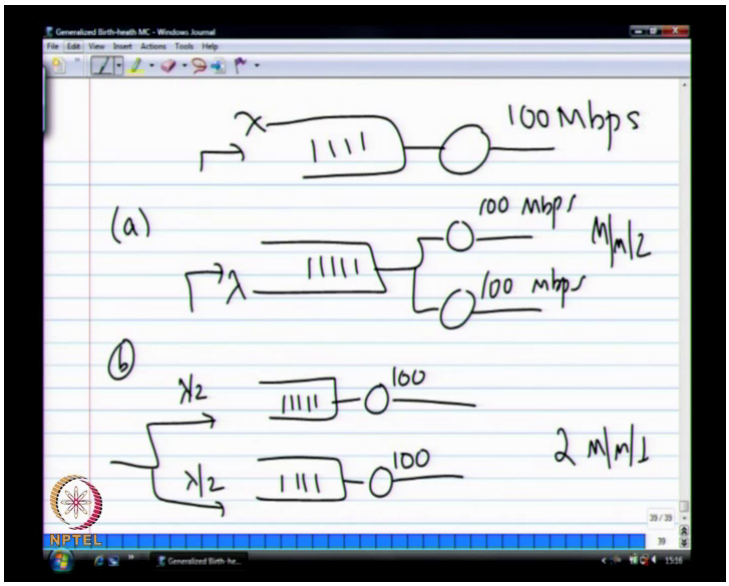
Prof. Krishna Moorthy Sivalingam

Department of Computer Science and Engineering

Indian Institute of Technology, Madras

Lecture No. # 16

(Refer Slide Time: 00:17)



Before you look at MG1, let us look at another design problem, right. So, you are currently operating a system let say, it is a packet queue, this is 100 mbps and then, you find that the queue length are large and you know that the system capacity definitely has to be improved because the arrival rate has increased. So, you have two choices.

You can of course, simply make this 200 mbps link that we saw last time, but can also add, it is impossible to go by a 200 mbps link. Only fix capacity is available. You simply add by one more link of 100 mbps. So, I am adding one more server keeping my queue constant or this is my lambda. Let us say that my lambda is splitting by whatever means into two streams. So, you have two separate queues. So, packets are split at the entrance into two separate queues. This is at banks also. In India we never know what kind of queuing system is used. You go to the bank and you will have one

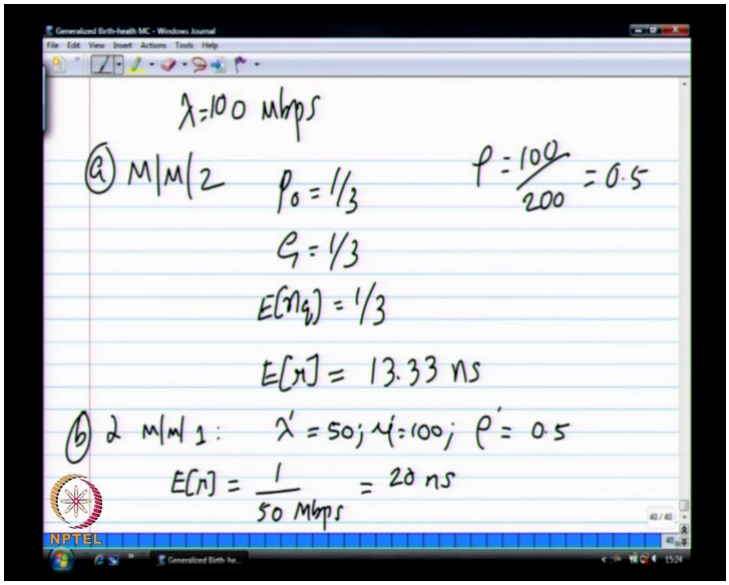
queue, multiple queues. The guy will be shouting, please follow one queue, but always a shortest queue we will just simply go and stand. So, we tried moral that one to multiple queue where people always find shortest queue to join.

So, what is the effective performance of that system? It is basically balanced at some point everything. Shortest queue will grow longer that you will see. This book does not have that example where customer choices are based on the queue length. So, this is called arrivals of discouragement, that is, we discourage arrivals based on the queue length. Which of these two would be better? This is another question. Separate input card input or separate at the service time.

(No audio available: 2:28-2:34)

The variance of wide variance will be more. Just mean will be the same or mean will be expected to be the same, but the variance will be different between the two. Any other it is not intuitive, but just (( )) have a single queue or two separate queues. Single queue. We just want a better performance in terms of response. That is the user only who cares about that. You are saying one counter is not enough. So, the banker says ok I will put two counters. Then, the question is today having a single, this is resume that is infinity. This is both m/m/1. It is either m/m/2 or 2 m/m/1s. Two separate queues are better? One queue is better, ok. Some of you have seen that, some of you have not seen that I am trying to guess. I will go through that and then, we will search for the intuition as to why one system is better than the other.

(Refer Slide Time: 03:56)



For now, we simply let us say lambda equals 50 mbps.

(No audio available: 4:00-4:08)

Yes, first I want to get numbers. Then, once we see that you know numbers tell us this is happening, then we can then should be able (( )) now undivided. Those who have not seen the problem or not sure which way to go, then you should give them the first look at the number. Then, see why should be better than the other.

So, the m/m/2 system this is row equals 0.5. So, we did this already yesterday on Wednesday. So, e f or p, everything was 1 by 3, right. P naught was 1 by 3, this thing was 1 by 3, E nq also was. Its rows was 0.5, sorry we should make this equals 100, right. Fine we will give the same example 100 by 200. So, row is 100 by 200. That equals 0.5 for that same area 0.5. So, you have n q is this and then, what was E f, what will be E of r? 13.33. What did we say? Milliseconds, nano seconds, will be micro seconds or I did 100 mbps. So, will be as just milliseconds at 100 mbps link and I am already looking at per bit. I am not talking about packet and all. So, 1 bit takes 1 nano seconds. So, yeah 1 bit takes sorry 10 nano seconds. So, this is looking per bit as a delay. So, it should be nano seconds. This lambda is 100 times 10 power 6 enforce. The nano second is correct.

Let us get conclusive answer on this. Some people still in how can when I am sending 100 mbps, how can 1 bit take anything more than 10 nano seconds? That is your one move take 10 nano seconds and then, my 2 m/m/1 once. Now, my lambda prime is 50, mu is 100 and row is, this is called dash dash. So, to differentiate the two rows is basically also 0.5 row dash. So, E of r is 1 over 50. Actually, we have micro seconds because 50-50 times 10 power 6, then 10 power 6 goes up and then, thousand, no sorry. It is the multiplying thousand above. So, these two numbers, agree with everyone? All right. Second number is it compute 1 over 50 into 10 power 6 (( )) 20 nano seconds. Therefore, this also (( )), but it is less. Those who voted for m/m/2 can now feel better, but your theory, that mean will be the same. Why? Because if you will look at m/m/1, look at Marco chain itself, the return rates are faster for at least some of the state return rates try 2 mu for all state. Not all, but the first everything is returning at 2 mu. Here, it is always returning mu. Only thing is my lambda is reduced to lambda by 2.

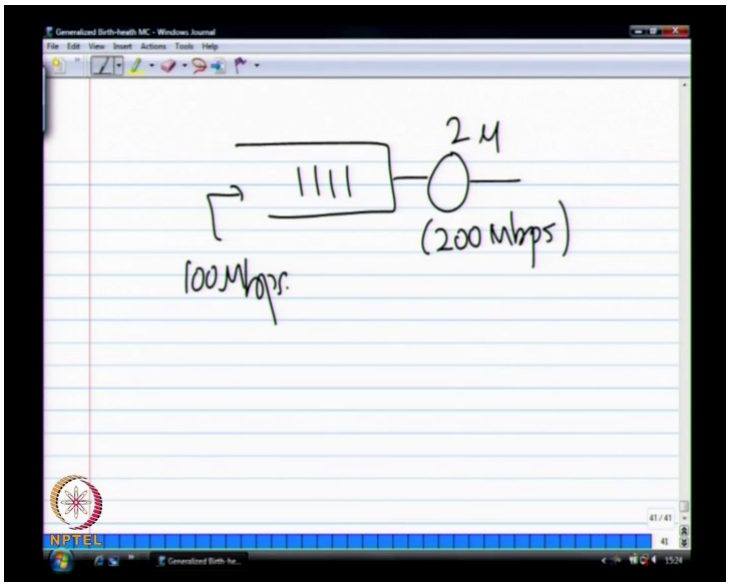
So, therefore, customers, the serve effective serve rate is better in the first system, m/m/2 system. It is 2 mu, that is one and the second intuition that we have tried to convince ourselves before is that it is possible that there will be a yeah that server is, that particular queues are empty and the server is free and there waiting customers in the other queue based on the randomness of arrivals. Therefore, one server might not be used for certain duration, that p naught, whereas in the other case, the server will never be empty as long as there is a customer waiting to be served. The single queue, the server will never be ideal as long as the queue is not ideal. That is another reason why you will find that the first system will especially perform well. Single queue is always better.

M/m/2 will be 2 mu and the second case, it is mu, but your rate is the, arrival rate is lambda by 2. So, it, but explanation is that the possibility of waiting. No, actually that single system m/m/1 with 100 mbps will be with 200 mbps will be the best. This will be better than yeah yeah definitely yeah yeah yeah . This will these two will be definitely better than that. Yeah, these two will be an improvement. So, here if I will give to an 100, what is going to happen? This is an unstable system. I cannot operate 100 mbps with the first, with the basic system. This I cannot. Lambda equals 100 means system is unstable. This is one of the reasons why we want to increase the

capacity because lambda is approaching so close to mu that the lines are getting very long. That is why you want to read the sentence. Therefore, these two will be an improvement.

Then, the other problem we saw the day was is in m/m/1 queue with 2 mu better. Finally, that we show that ultimately what is the best way do this to double the server capacity.

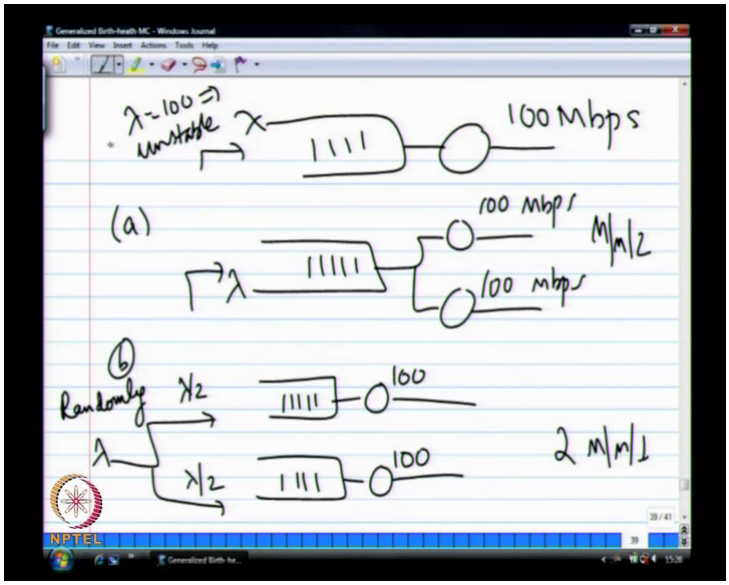
(Refer Slide Time: 11:13)



If you remember the companion problem was. I said this is 2 mu. If you are there is an option to actually double line capacity or double the server capacity, then stick to the single queue. This is what to (( )) you in various system we look at the network, then the system application. If you look at CPU, where you go two CPU, servicing processes are build of single CPU or double clock cycle or whatever the queue try to do so, but it is always impossible to simply double the clock speed. You have to look at two courts and then, you look at whether you want to think of as a processor. This processor threads being executed by the CPU. This is the ready queue and (( )) have one common ready queue. Then, have two separate ready queues one for each core.

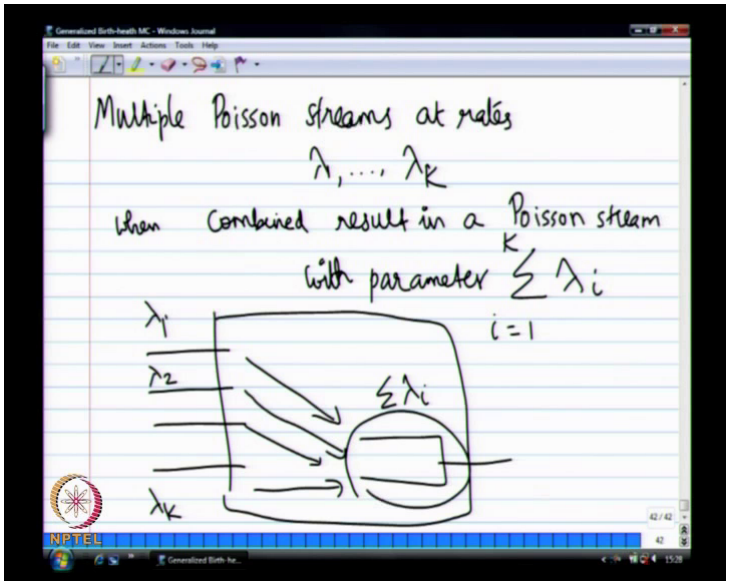
No no no no this is a random this is purely probabilistic.

(Refer Slide Time: 12:34)



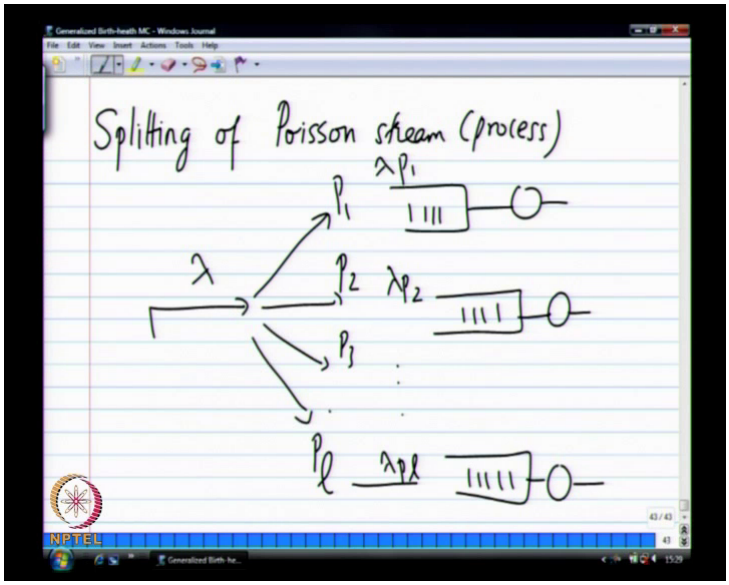
Yeah, this does not depend upon the queues. Yeah this is not depended on queue length. Let go and see if you consider the queue length, yeah this is random. That is why this is randomly selected or uniformly random, but now one important thing that we have not talked about is that this is the Poisson process and I split that probabilistically into one or two more streams. Why should that also be a Poisson process? I am making an assumption here which, yeah the probability, some probability p i whatever be the distribution, that is the property which I have not taken, but that is an important property which we are now stumbling upon. I think in one of your (( )) has proved that. So, merging of Poisson stream and splitting of poison stream, it especially results in still a Poisson stream.

(Refer Slide Time: 13:28)



So, multiple Poisson streams at rates respectively lambda 1 through, it is a lambda k when combined. It results in a Poisson stream with parameter. This is again convenient assumption which is (( )). It is nice because we look at for example, q. When there are, when we look at one output queue of router, so this is the picture of router. So, router will get several packets from various input line cards and there are (( )) to this particular output port. So, packet from this queue will this port will come to the queue from this port to this queue and so on. So, when I have several each of these Poisson stream, independent Poisson stream, each of the midrate lambda 1 and lambda 2 and so on, then the effective arrival that this q, see this sigma lambda i and that is so Poisson. That is why I can use my m/m/1 assumption, I merge different price on stream.

(Refer Slide Time: 15:22)



These are fundamental properties of Poisson process that we can look at some in standard textbooks like a splitting. I am saying stream whatever Poisson process. It is stochastic process. So, if the input is coming with (( )) lambda and then, based on some probability which is fixed p 1, p 2, p l, so probability p 1 go to this will go to one queue, p 2 goes to another queue and so on. Then, their respective queues will see the voice on process with rate lambda p 1, lambda p 2 and so on. So, that is why splitting example we saw before makes use of this fact, even though I am splitting, but with equal probability. So, it is simply coin toss. It does not depend upon the state of the queue.

So, its state dependent questions on this which you mean same dependent. Queue length dependent? Queue length dependent will that be theory better? So, you always join the shortest queue, but there is still non 0 probability that service time for two pockets might be so short that they finish off and then become empty. Then, there is no packet in that corresponding queue because you guys all join the other queue. Just queue jumping is not allowed. If you do not allow queue jumping, then you are stuck in the queue that you join. Yeah, so for that still the probability that the second queue also. You may be balancing the two queues that if there is a close formed, I will try to find that. If not, then we can simply simulate, right. When we go to the simulation part, we can simply simulate that straight queue jumping. So, you simply look at

current state two queue and take the shorter one and see that because shorter does not mean that the service time, an average will be less, but instantaneous service time could be fairly large. It is that always happen, right. You go to the bank, you pick the shortest line. Definitely something is wrong with the line, that guy is going to take forever. So, that is nobody goes over there. It is a reverse (( )). Nobody goes to the shortest line. That guy want to get DD and that is it.

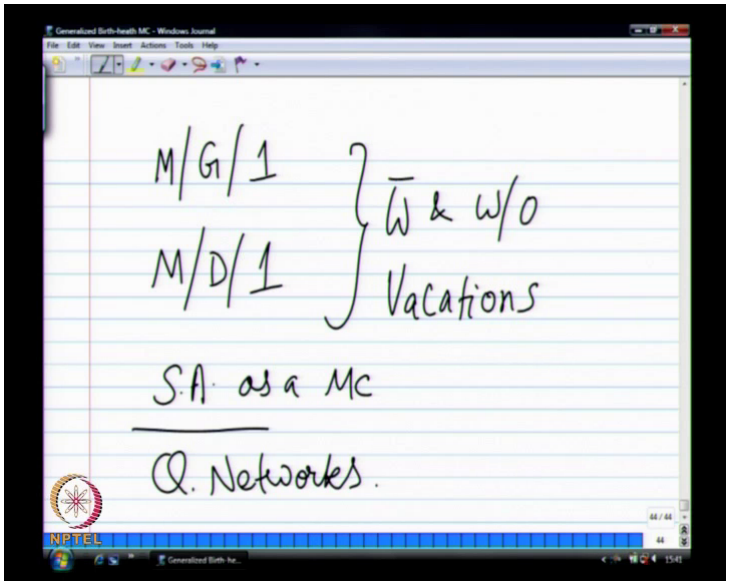
(No audio available: 18:54-19:08)

This is after joining the queue yeah. That I have not seen, but this is our third variation we are talking about. One is the joining time and other is keep going back and forth which is some finite probability of staying in this queue or going to the other queue.

(No audio available: 19:25-19:33)

Yeah or you can have like a loud balance that trance in which is distributed system try to you find that the queue length is longer, you simply migrate jobs to be on the other processes. That also know one thing I have to find. I am not, I am not. Yeah all these things are easy stimulated than solve. So, we will stop and I will just give you so we will look at this M/G/1. So, we have done m/m/1 so far, will not today next week and this M/D/1 is particularly interesting in least networks because we are talking of deterministic packet length. So, fixed packet length, especially when ATM systems were being discussed, they will have fixed packet lengths. M/m/1 is more even for you know for telephone systems, it is not exactly suited for routers because packet (( )) do not follow the exponentially distribution and terms of length.

(Refer Slide Time: 20:44)



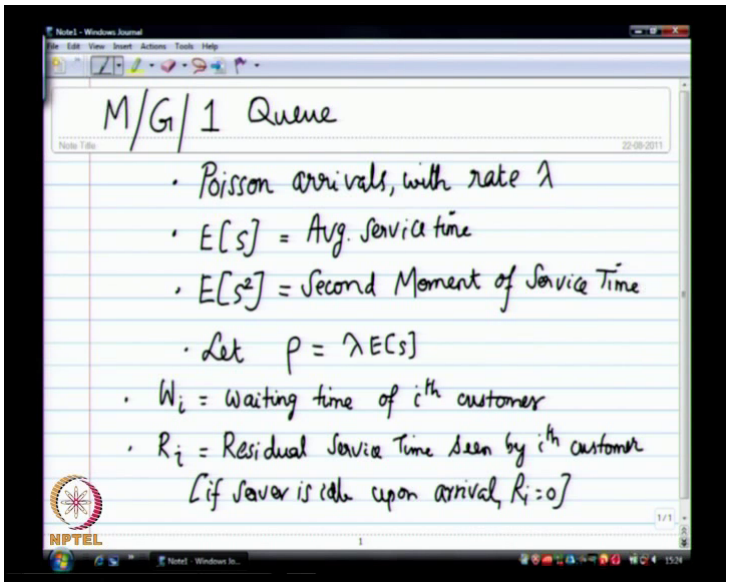
So, we will do that. Then, we will do this M/D/1 with vacations and the reason will be the M/D/1 of vacation because that is the special case of TDM. TDM can be moral as an M/D/1 server with vacations and you can get a very nice close form solution for that. That is why we would like to do that and then, I am sorry what a vacation means. What vacation means is server in our example so far, we have what called as a (()) conserving server as long as there is a customer, we are going to serve the customer, but we human, we never do that. We always tend to take vacation or breaks as we call in the service sector.

So, in this case again, it is not only server taking break, but you look at a machine. So, machine will be constantly servicing jobs and then, you bring down the machine for maintenance and that maintenance duration can be fixed and can be variable because of repair also. Suddenly, it has to be repaired. So, the time to repair is again not a fixed time. That is also exponential. It can follow general distribution. So, in that context what happens during the busy there maintenance period, the system is idle, customers are waiting. How do you characterize that? Within the presence of vacation what is the expected waiting time and delay and things like that. This is that is why this vacation comes. It is not because this is true in case of machine scheduling and things like that.

So, then we will go back to the (( )) allow have that example that you looked at. It is also there for your project implementation. We looked at different approach to (( )) allow have that because I worked it out twenty years ago. You have to suffer through that. We will come back and (( )) have a Marco chain, but not in terms of the number of users, but in terms of single of users just to give in other way of try to modular system. You can also get to the same set of results by modeling (( )) of single user in the context of other user. It is little bit more involved, but (( )) solve set up equation to get. I think that is probably where we will stop.

Then, the GMN all that stuff, the results are there in the text book. We will stop at that point with respect to the open queues. Then, after that we will move into Q in networks. So far we will be all single queues. Then, we will look at how you interconnect using together. The queuing network is actually you look at a real computer system. The examples that we saw before in the tutorial also that all of you skipped where there is CPU feeding to device queues and then, it comes back. So, those are the kind of the systems where you need to get some idea of what is the (( )) system. Let us switch gears to M/G/1. M/D/1 is after M/G/1. M/D/1 is automatic.

(Refer Slide Time: 23:54)



So, see all discussion on the test, let us come back to. So, for M/G/1, the arrivals on Poisson like before and the rate of the arrival is lambda arrivals per seconds. As per the service process goes, it can be any service distribution. There are besides the mean

in the case of exponential, we knew 1 over mu is mean, 1 over mu squared was the variance and all those things. So, here we need besides the mean service times. Let us say just sitting at bank watching how long it takes to serve each customer. So, you can compute the mean very easily, ok just way the sample (( )). You can also compute the second moment.

We simply square off all the service time divided by the number of customers requests that have been serviced. So, this is the second moment of service time. It will happen that with these two, we can quickly derive the M/G/1, same thing waiting time and all the other metric you want to look at we can easily derive. So, we will let, so the row is the same as before. It is lambda by mu, where 1 over mu was the average service time. So, here it is simply lambda into E of s.

The definition is same as before ratio of a product of arrival rate into the average service time. Then, we have some other definitions. So, W i is the waiting time of the ith customer. Customer comes in and how long customer wait in the system before getting service. This is the waiting time. Now, when the customer, ith customer comes in, two things are possible. One, the server is busy or the server is idle, two possibilities. So, if the server is idle, then this notion of residual service time. That is why R i represents. This is residual service time seen by ith customer.

See the customer, if a customer is being serviced, served by the server, what is the remaining service time for that customer? Say j is the customer being serviced, so that is this, but I see as I enter, so R i is the customer j whose been serviced the remaining service time. That is what R of i (( )). In the case of exponential service, that is always 1 by mu. Whatever point you look at the server, the customer service is always 1 over mu because of memory less property, but in the case of other distribution that need not be the case.

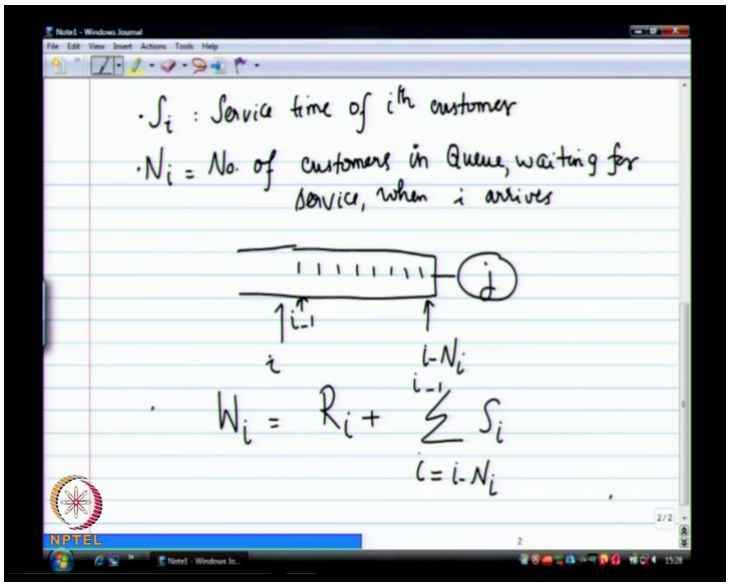
Therefore, there is some remaining service time for the current customer being service that what R of i represents, that is the server is busy. If it is not busy, it simply goes to 0. There is no, this guy will get served. The server is idle. The ith customer will automatically get serviced by the server itself and therefore, there is no waiting time itself goes to 0. So, if server is idle upon arrival, then R i goes to 0. R i goes to 0, W i

also goes to 0. There is no waiting time for this particular customer. R i is just the customer that is getting serviced. That is all.

W is the other customer. W is the total waiting time. Say I come to the queue and there are five customers already in the queue. There is this other guy getting serviced by the servant. The guy is getting serviced by the servant. That remainder of the service time is the R i. W i is the time that is going to take every one to get serviced. That is all.

Yeah, the time taken in for the remaining customers in the queue, so when we derive the W i, you will see that why this R i is special. This thing is; this server? No, there is no R i. The server is idle which means the queue itself is idle. There is no, the queue is empty rather. If a new customer comes in and finds the server is ideal, what does that mean? That at point there is no customer to be serviced. Otherwise, the server would have been servicing the customer in the queue assuming it is a work consuming queue. In the case i as the part of one by the fractional there, in the case of exponential, R i is always 1 by mu because it is so because whatever time you look at the system, it is mu.

(Refer Slide Time: 28:56)



Now, let us look at this. So, then we will say S i is the service time of the ith customer and then, N i. So, N i is the number of customers in the queue waiting for service at

the time when i arrives. So, this is looking at the system from ith i perspective. So, if you represent this as my queue, here are some people waiting. There are some customer j who is getting serviced and then, i is, this is the ith customer. Assume that customer numbers sequentially. So, i is the id of the customer just now entering the system. So, what will be the id of the customer at the head of the queue? There are N i customers. Therefore, it is i minus N i.

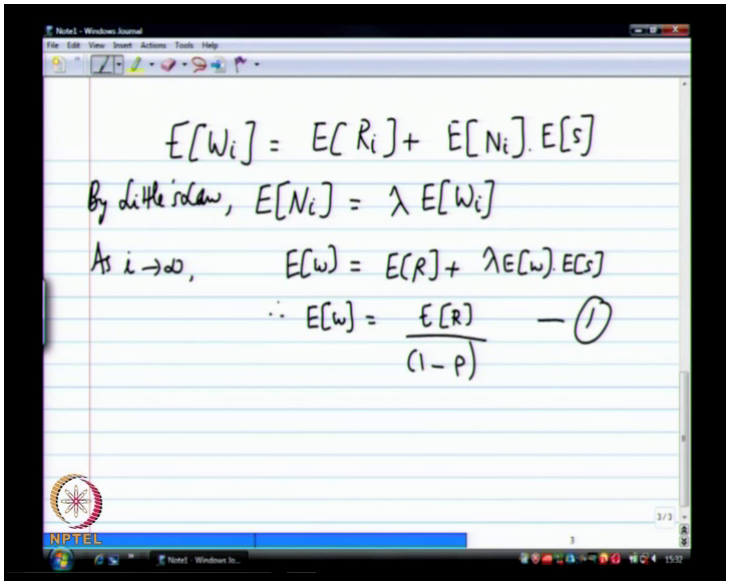
Now, we have this waiting time for the customer i is given by the residual time for this customer j, this fellow which we called R i because its R i is not the time that i see when there is somebody j is getting serviced on the system. What is the remainder of j service time system is what we denote as R i and then, i equals (No audio available: 31:08-31:25).

See simply adding of the service time of all the people in front of me plus the residual service time for the customer under service. That is all. This is not very hard, right.

(No audio available: 31:35-31:43)

So, these S i are random variables. So, they follow whatever distribution that we are saying and N i is independent of this S i. If I take the expectations on both side, again I am skipping some of the, if I take the expectation for some of random variables, then it will be simply product of the corresponding expectations

(Refer Slide Time: 32:08)



So, E of W i is E of R i plus this is the expected number of customers in the system when I enter the system into simply E of s. We said that taking expectation on both sides, s is nothing, but service time and what do the service time average in the long is simply E of s. So, this N i is the number of customers were in the queue when I enter the system. So, we now know there from little's law, E of N i is the queued customer. So, what is that? Lambda into E f. We call as n queue before, but using it slightly rate differentiation because I am taking from another book, my old book Bostacars and Galagar, MIT. So, that is 1982-87 book. Some of the terms except this x, I am changing that to f s is everything is following that notation.

So, by little's law and then, we can assume that has approached infinity, we assume. So, this is basically the average weight in times E of w which we have seen before is essentially we call it E of R. Simply drop all the I's as I (()). Therefore, (No audio available: 34:04-34:16).

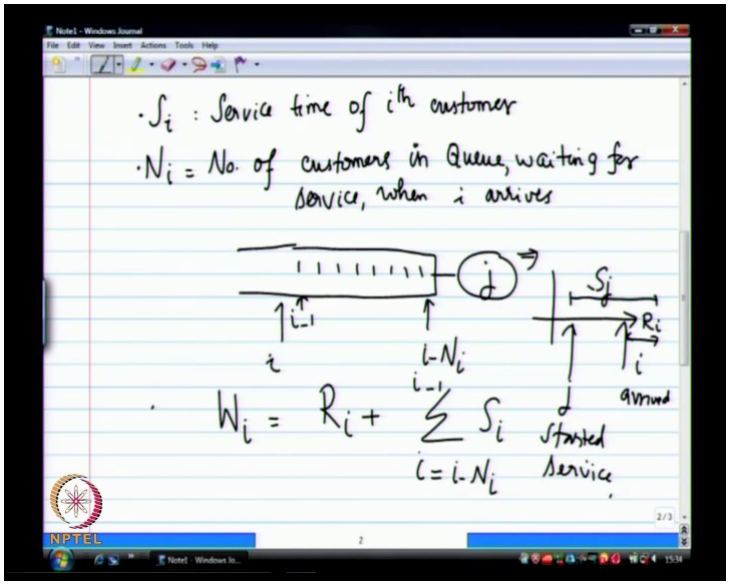
So, N i is this one and E f is also there which often that lambda E s is 1 minus is row. So, I simply this is basically row into E of w. Yes several take vacation and that will be simply that plus average vacation time. That should be 0 yes. Basically, E of R is now related to E of w. So, the average waiting time seen by a packet is simply the, is proportional to the residual time of the customer, average residual time of the

customer that is into divided by 1 minus row. That is first part of the derivation. This is one. R is basically w equals R by 1 minus row.

(No audio available: 35:25-35:39)

So, this guy arrived at some point in time. Let see that what we represents.

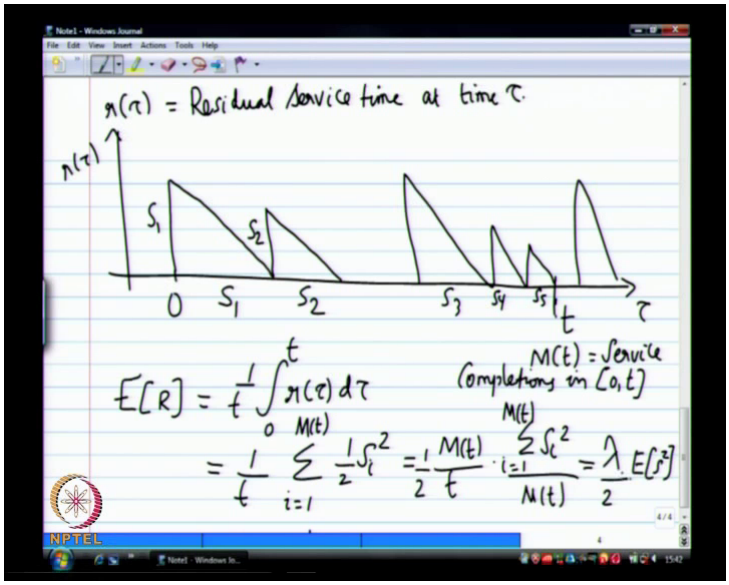
(Refer Slide Time: 35:53)



So, this customer j is currently in service. If you were to draw this symptom of a time axis, this is when j stared service, this is when I arrived and j's finishing time is this. This is the time that is going to be taken to service j, the total service time and so, the time that is remaining is what I call as (( )). That is all the remainder, right.

So, I have to wait until this guy gets finished, all the people in front of me gets finished. So, the average of that simply the average number of customers I see in the queue when I enter into the average service time. Average I see 5 people when I enter and each customer is going to take 10 seconds on average. Therefore, the average time to get people in the queue would not have started service in simply 50 seconds. Now, what we need do is some of figure out what this E of R is. To get to the E of R, we use a slightly different way to.

(Refer Slide Time: 37:05)



So, let us say that I defined r f tau. So, there is a time varying function call r f tau, that is, the residual service time at time tau. I fix some random time tau and see either the customer in service is not in service. If there is no customer in service, then the r of tau is 0. If there is a customer in service, then simply the remainder of that customer service time. So, I can plot this with respect to r of tau.

Let us see that upon to this point, this is my say you know 0. Up to this point, there was no customer to be serviced. Then, suddenly there is a customer to be serviced. So, what will when customer start serviced, let that customer service time be S1, the customer number 1. So, r f tau will become simply S1 because instant what is the service time Si. Residual service time is S1. Then, as t in or tau increases, this keeps on decreasing linearly. So, it will come down to 0 and it will take S1 time. If there is yet another process or job waiting to be serviced, let us say that takes time S2. So, it is like a sort of graph like this.

Assume there is for little while, there is no customer to be served, then after that it goes like this, it goes like this and so on. So, then I am interested in the sometime in time instant t at which point time we want to do some (( )). So, up till this point t, let m of t be the number of service completion. So, how many customer were served in the interval 0 t? This is S 3. This continues, but I am interested of this is some point t.

Now, if I take the area under the curve and divide that by t, is that my average residual time E of R? The average residual time is simply r of tau from 0 to t. Divide this by t. So, that is r of tau is instant of time and then, dividing it by total time taken. That is my average residual time. Each r of t is instant residual time and adding them off dividing that by total time interval t, length of the interval t, that is the average residual time. Not a very hard thing to accept, right.

So, what is that? Just look at this. Just write of a set square. That is all. So, this is 1 over t i going from 1 to M of t half set square. Simply the area falls the triangles which is half a square set square simply adding the whole of. So, this now we can make a little bit interesting by saying this is M of t by t and then, I have this S i square by M of t.

So, what is M of t by t? M of t by t is the number of completion, average number of completions and in an m/m/1 system and M/G/1 system with infinite queue; the number of completion is equal to the number of arrivals. Trooped is basically equal to lambda as long as row is less than 1. That is the balance because only if have 10 customers are coming for unit time, then they will leave the system again. If you look at the number of customers leaving, that is also going to be 10. So, lambda is also equal to mu.

Therefore, this is basically lambda into so M t by t is the trooped of the system number of customers leaving the system which in a balance system which row less than 1 is always going to equal the number of customers arriving. I cannot have more customers leave the system than arrival. That will be very suspicious. It has to be equal and I cannot have an average number of customer should be can be one customer less, may be depending on the observation interval, but in the long time, we simply say that it is equal. Ok, I forgot the half. So, we will do lambda by 2 into, so what is the sigma s squared by m of t? E of s squared is the second moment by definition. E of s is simply s 1 s 2 by n, E of s squared is simply s 1 squared s 2 squared divided by m. So, this is, so that is why the results not depend upon row which depends upon E of s as well as E of s squared is the second moment.

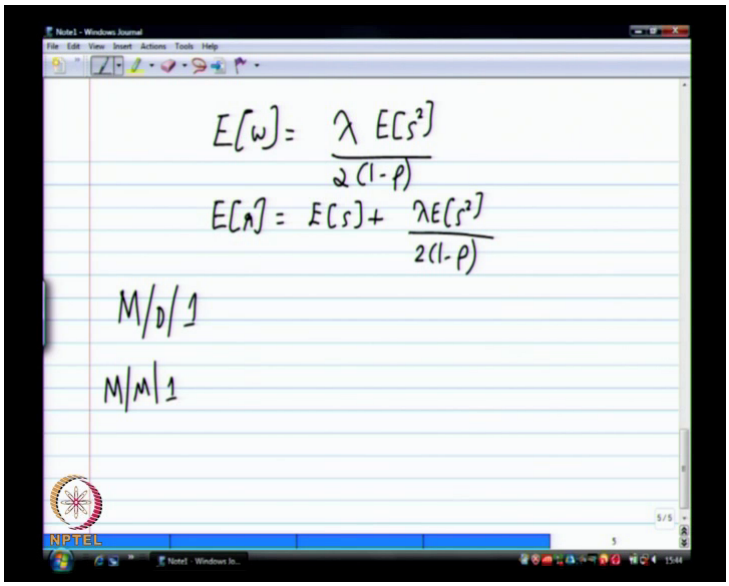
(No audio available: 43:00-43:43:19)

Yeah by i (( )) yes so in the, we are taking that is one derivation. This is other way of also looking at the residual time. What will be any packet on arrival, what is the residual time, we are going to see. This is the long term mean residual time is what we are talking about.

(No audio available: 43:30-43:35)

Yeah, in the second part I am deriving it, direct define it that way, but it also can be, it also can be defined in this way, derived in this way. What is the mean residual time that any customer arriving at any point of time we will see. If customer comes here at this point, it will be (()) will be 0, but what the long term average is over some interval that you are measuring of the value. That is what you are computing. So, remove all of these are mean. Therefore, it is not instantaneous.

(Refer Slide Time: 44:09)



So, therefore, E of w equals lambda. Then, there other things are straight forward. E of r is simply E of s. So, with this basic result, we can also figure out what M/D/1 is. We can go back and see. All we need is second moment. What is the second moment of finite distribution? Second moment of M/D/1 is not 0. Second moment is not 0, variance is 0. Second moment is simply E of s squared because all values are E of s, E of s and service time is constant. Therefore, E of s is simply one value. There is only one value whatever its sum, we just have E of s.

Therefore, second moment is simply E of s squared. So, it will be lambda E of s to the whole squared. So, we will have lambda, E s will give you row into service time there were it by 2 into 1 minus row and that will, so what we will see at for the least waiting time will be seen for M/D/1. We cannot do any better than that because that is (( )) that c of v is also 0. Variance is going to be 0. We will derive that in terms of the book has the slightly different formula. Rajan's book has slightly one plus the question of variation whole square. That we will come back and derive it next week.