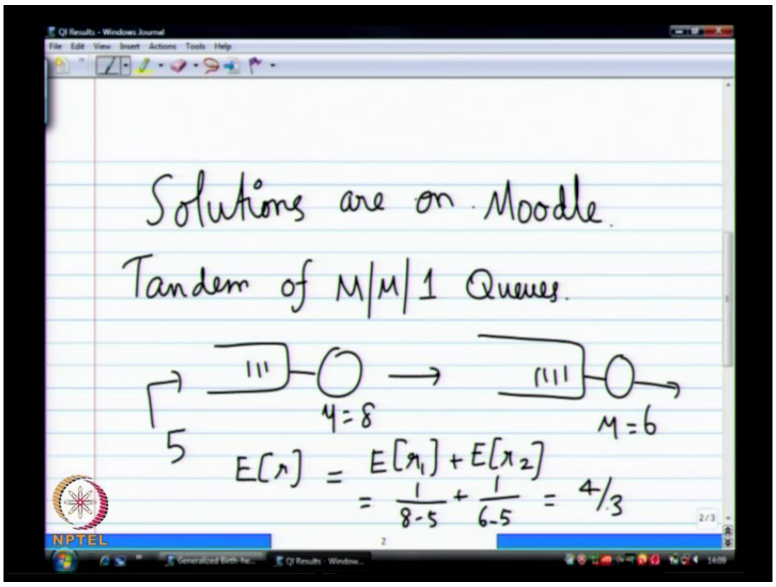
Performance Evaluation of Computer Systems

Prof. Krishna Moorthy Sivalingam
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture No. # 19

Queuing Networks – II

(Refer Slide Time: 00:23)


Solutions are on Moodle.

Tandem of M/M/1 Queues.

$$E[r] = E[r_1] + E[r_2] = \frac{1}{8-5} + \frac{1}{6-5} = 4/3$$

So, now let us say visit our single tandem right, tandem of queues. So, tandem of queues is a special case right. So, tandem of M M 1 queues used to do the evaluation, you can calculate the number of customer in each of the queues, simply add them up right. So therefore, you can evaluate all of them independently, that is why so called product form network right. So, for this for example, if I have a queue right; so lambda equals10, and say mu equals 8, that piece into second queue. I briefly gave you the derivation for E of n right, total number of customers in the system. So, for this system what is E of r?

(No audio from 01:02 to 01:17)

So, how do you compute E of r? E of r is simply the resident, the response time in queue 1 plus the response time in queue 2 right. Therefore, this is simply 1 over or (( )) or should had this is10 about that, make this 5 right. So, your lambda should be less than both mu's for the system to be stable right. So, this is simply 1 over and whatever that what so. Say, 1 by 3 plus

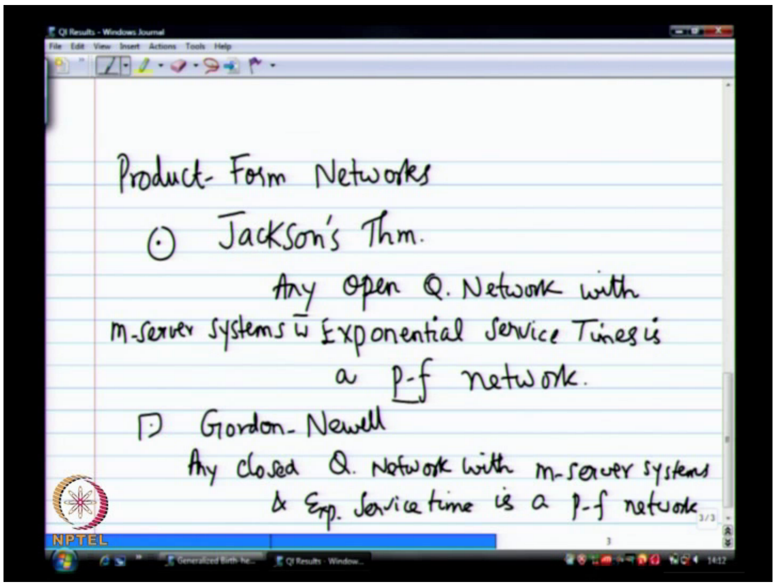
1 by 2, 5 by 6. So, that is (( )), and this applies to any series, we have several others, this is the special case right. We are going through multiple servers in time.

(No audio from 02:10 to 02:19)

Now, everybody is awake; I should be awake. So, 4 by 3 that better. So, this is one particular form of product form network. So, then there has been other words that people have been slowly proving one class after the other, that such in such class of networks is also close product form right. So, some of the classic result right, so products form networks.

(No audio from 02:57 to 03:09)

(Refer Slide Time: 03:00)


Product- Form Networks

⊙ Jackson's Thm.

Any open Q. Network with m-server systems $\bar{w}$ Exponential Service Times is a p-f network.

⊡ Gordon- Newell

Any closed Q. Network with m-server systems & Exp. Service time is a p-f network.

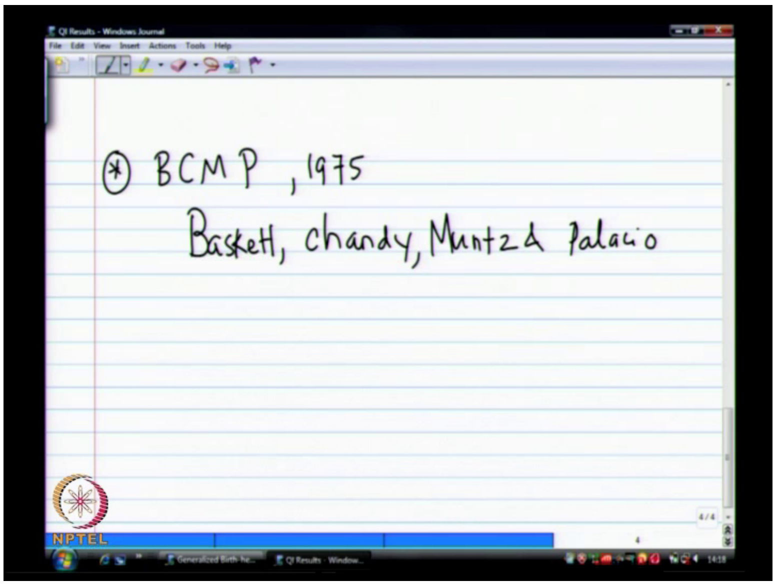
So, based on based on (( )) theorem, we could show that tandem queues, we can evaluate; that is a one special case. What do we have as set of network of queues right. We saw before open and closed and so on right. What do we have set of M M 1 queues, where there is some probability of moving from one queue to the other and so on, and it is an open queue right. So, what Jacksons proved? This is Jackson's theorems how that, any open queuing network with exponential service times is also is the product form. So, that is one of the classic results in the 60s and 70s, and then there is one more well few more results.

This is1963, if you are interested, and then there is another theorem which is by Gord on Newelland the thing is this is, it can also be M server queue right open network. It is not just M M 1(( )) M M M writes this, even burs applies to M M M, which we did not talk about, but burs also if you look at the system that is would be the output. So, Gordon Newell then proved that, it is not just for open network; it is also for closed networks. But the same condition, exponential service time is what they all initially worked upon. So, here they showed that, any closed queuing network with a M server system and exponential service time is a product form network.

So, this is the second result. Here, keen you can go back and check out some of those hold paper system how they showed that; but both these cases, the restriction is there; the service time is exponential. So, what is this enable? This enables us to essentially treat all the queues independently, compute the service time and those that is what we will see. Next section will be allowed to analyze queuing networks in try to find out waiting time and response time, number of customers in the system, throughput all of that with the help of these results is what we can right.

Sometimes we intuitively just say yeah you know just add up the numbers of a in a customer reach of the queues and that is the total number of number is (( )).But delay is for example, we can intuitively makes some guesses, but it is buried to have some proven results (( )).And then there are another force another class I am sorry there is, this does not say anything about the enable time; only the service time are exponential. But have to go back and look at the original papers really also required, Poisson right in that particular case.

(Refer Slide Time: 06:59)



(*) BCMP, 1975

Baskett, Chandy, Muntz & Palacio

So then there are the one class called the BCMP class, which I am going to skip; because this talks about the other types of queues also and other type of service disciplines right. For example, processor sharing, I called it first first come first serve processor sharing and as well as not rank last in last come first serve. So, about 3 or 4 particular types of service disciplines besides the FCFS and some other restrictions on exponential is still there. But the type of discipline that we can use besides FCFS, we can have another discipline too. So that was the BCMP has and another set of conditions; any queuing network that satisfies this, so called BCMP condition.

BCMP is named after the authors and if you really want, this Baskett, this will not be on the (( )) and this was in 1975. They showed for a larger class of networks that, this set of product form solutions possible. Will it actually affects arrival time distribution? No, their response time will change which LCFS versus FCFS, like processor sharing. If you look at processor sharing, results are there in nothing (( )) they are the mean time does not change. Because the order of scheduling does not affect essentially are mean time, but variance in delay can be different right.

So, first order moments might be the same, but the other moments will be (( )) processor this is M M 1 with processor sharing is the scheduling discipline. So, it is not whereas what we talked about is FCFS right this is different FCFS, once you take a customer, you service the customer completely. With processor sharing, you service each customer for a very tiny duration like round robin right and then you keep going to the next customer. It is not even round robin usually largely quantum; this is like a bit by bit rate. So, an extremely smalltime is allocated to each customer you service and then go to the next customer, next customer, and so on.

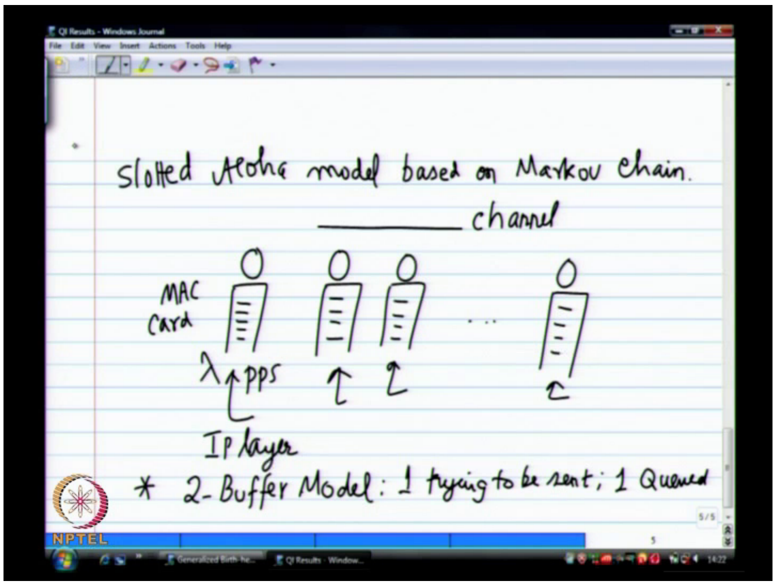
So, with that essentially, we will end up with same average time; mean time will not change, but the other parameters will change. That is what, this (( )) and those thing we will verify with simulation. When (( )) M M1 Q, I will give you for FCFS the same reaction code also supports LCFS for example. What if we have LCFS as a queuing discipline? What will happen? Then you can see LCFS does it lead to will always lead to, it can lead to salvation right if it is last come first serve. (( )) for ever sit there, until the queue get empty is going to be sitting there.

But those things, we can play are all easily with with simulation and see, what is the stable operation and condition. There will be second project, second assignment that will come out. And then third assignment, the plan how is to have queuing networks right. First will be simply set of queues with may be tandem queue also we can try out and then after that the last assignment will be set of networks of queues by which time, I would have covered all the open network and closed queuing networks; then we have all that will be done. So I can actually, we will do the theory is there; also simulation to show that, whether the theory is correct or not.

And the next thing with the simulation is that, I can say exponential only this formula formula will work, but what if it is not exponential. So, you can simulate non exponential service times and then compare those results with the base line exponential right. So, in future you can always say exponential gives me one sort of trend settings or some sort of lower bounds on the system asserts. So, this is the set of set of things that it will come. So, what we look at next is. So, before I go to queuing networks, because I going to be couple of weeks in which, there will be some classes I want to start queuing networks (( )) if possible.

(( )) So, now I promised some time back, we will look at this slotted aloha model as the markov chain with the slightly different perspective; that is 1; that is pending. We also yesterday also talk about semi markov model. So, we talk about markov, semi markov model is just another extension and sometime this useful to just passed it you just cannot make everything as an M M as the markov chain. So, you have this so called semi markov model.

(Refer Slide Time: 11:22)


Slotted Aloha model based on Markov chain.
channel
MAC Card
λ pps
IP layer
* 2-Buffer Model: 1 trying to be sent; 1 Queued

So, now let us look at slotted aloha (No audio from 11:21 to 11:46) and the reason we do that is, aloha is the very easy protocol to simulate in simulative (( )). So, we will have some simulation numbers and we always would like to know, right what the theoretical element is? This, the theoretical element that we saw one is the g e power minus g, right which is the very cross result. We saw the other model also; where you have the set of right states, then you looking at the number of customer, number of users in the system. So, that is one way of looking at modeling the system right.

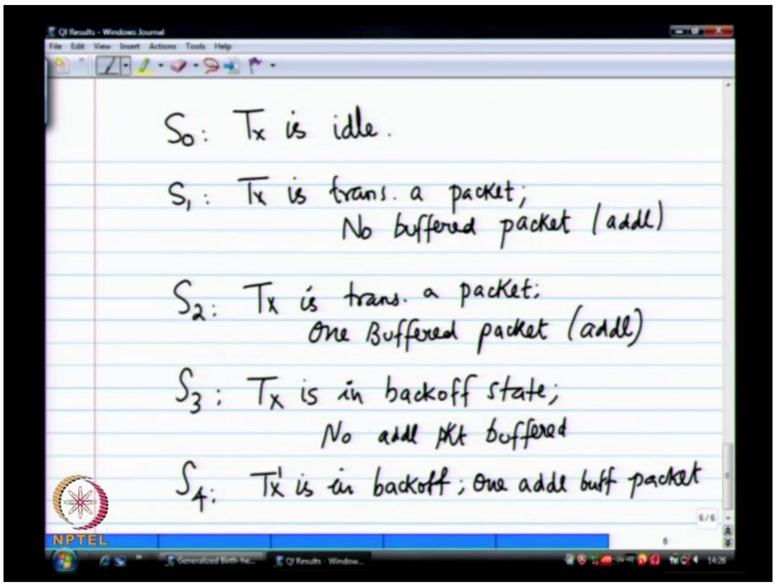
In that system, the state was captured as the number of backlogged customer and the others are un backlogged and you trying to see how the system transitions right. So, when if actually attempted the problem on the sample assignment, try to build it and right you could have seen. Now, that it is not ok. We will look at different model and then see whether those; this is actually implementable also easily. When when as part of your one of the assignments, will include aloha; aloha code is there; already I have written that. It made the bugs in it. So, I will expect you to fix it; the code is twenty years old. So, this model that I am going to talk about is something that was my first markov model.

So, you guys are struck with it; so we have to see what this markov model. But any way, I think it is still useful and we can verify whether my model is correct or not. So now, this model for aloha is going to be based on a single users percept. So, there is a notion of a channel right. So, there is the channel common channel that is being shared there is a channel and then like before we have a set of users. So, this assumes that, this represent the NIC card of every user right. So, NIC card; this is your right MAC card that is implementing aloha. So, I will model that as a server and then the MAC card is getting packets from which layer? It can be from both layers.

So, let us assume that, the packet is coming from the I p layer that, whatever comes from the physical layer simply goes to the I p layer. So therefore, there is not much of sharing is there right. Sharing is needed only, when you want to send the packet. So therefore, these are packets coming from the I p layer. So, packets are coming from the I p layer and lets what would I use here lambda. So, a lambda packet per second is the arrival rate. So, there has every is… So, now in the simulator you can imagine, I can simply build right. One process or one event, group of event are handling particular user and then simply replicate that as many terms you want thousand users, you can replicate by thousand right.

I can essentially create copies of it. So, now the modeling is based on the user perspective not in the system perspective. So, what are the states that each user is going to spend? It goes through different states. So, to actually model that we had one restriction right, so this is so called 2 Buffer model which means, there can be one packet trying to sent to be sent and then 1 that is queued. The packets are arrived to the queue at most be this case only 2 right. We can generalize this to be buffer model; if you want same model can be expanded, but let us we get 2-buffer model because it is easy to deal with. So, let us define few parameters that we will need here. (No audio from 15:36 to 15:46) There are 5 states.

(Refer Slide Time: 15:51)



S_0: Tx is idle.

S_1: Tx is trans. a packet; No buffered packet (addl)

S_2: Tx is trans. a packet; One Buffered packet (addl)

S_3: Tx is in backoff state; No addl pkt buffered

S_4: Tx is in backoff; One addl buff packet

So, S naught represent the state that, this is modeling the transmitter of every user; this is one particular result that, M modeling right. So, this is modeling that the user transmitter is idle. Basically, there are no packets to be sent in the queue right. This is one particular queue is the queue that I am looking at no packets in the queue. Therefore, the transmitter is idle. The transmitter in aloha will be in 2 possible states essentially right. So, one is when a packet arrives newly to the queue always try to send; therefore, during transmitting state and at end of this… this is slotted aloha right.

So therefore, this is a DTMC one slot is one packet. So, I have a packet the beginning of the slot, I will the send the packet. At the end of the slot, either there is a collision or there is success right. If there is a collision, then I go on to back off state; that is the definition of aloha and in back off what do we do? We solve last time back off probability p b; what do we had a and b last time. So, if you are in back off, there are different ways you can come out of back off right. One is to simply schedule treat that as the uniform interval; uniform between 0 and 20right.Any time between the next 1 and20 slots, transmitter will wake up and try to retransmit the packet that is one model of the back off handling back off.

Otherwise you simply have this probability p b wherein, every slot you will keep flipping the coin like geometric process right; keep the ping and then decide to send or not or (( )) geometric. So, that we are assumed the second one wherein, every slot there is the probability p b of trying to resend; with 1 minus p b you will not resend, you simply stay (( )). So, S 1 transmitter is sending a packet and there is no buffered packet. So, there is occupancy of this queue is simply 1 and then the parallel state is possible that right.

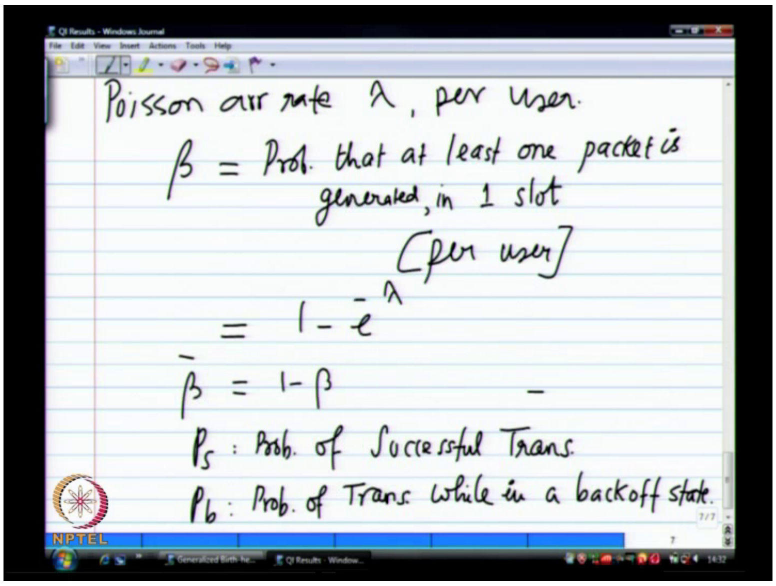
I tried I failed and then between another packet, k minute got queued and only 2 packets can be in the queue right; third packet comes simply get shocked. And here, this is the situation where I have a packet queued and also I am trying to transmit a packet. (No audio from 17:53 t0 18:05)So, this no buffer I mean no additional packet is buffered right. So, there is all there is one packet in the buffer, which is the one that I am trying to send. So, either transmit in every slot, the transmitter can be none of these state. Either sending or simply sitting in back off, even though there is a packet to be send or an idle because there is no packet to be sent.

So, this is S 3 is a transmitter is in back off state with no additional packet buffered and S 4 is ditto as S 3, except there is one packet buffered (No audio from 18:53 to 19:16)yeah total queue. We are assuming queue length to be 2. Yes, the total buffer capacity in MAC layer card is only 2; it can be, it can store 2 packets. So, one under transmission; second one is simply waiting for service for the first 20 (( )) and it simply first come first serve. So, until the packet and the head of the queue finishes successfully, the second packet cannot be send, (( )) one additional packet yeah; you already back off statement, so the total of 2 slots; so I am in back off.

Let us say only one packet is come; one packet is sitting in the queue and I am try to send it; it failed and no other packet has come for a long time. Therefore, that is one S 3 is; but in the time that I am waiting, because the back back off I try to send in a very starting probability p b. So, some slot I will never try to send at all, that time one more packet comes then the occupancy of the queue will be 2.So, we are assuming that before I am talk about in S.

See S 4 state will happen; this is the sequence, you are in S naught right I will draw that picture then let us let us look at the transition diagram, the state vision; then you will able to figure this. So any questions on… This is the state I am sending, no no no other packet is waiting; I am sending, but one packet is waiting and I am not sending in back off, either there is a packet or there is no packet. Yes for buffer is full; yeah even S 2 also buffer is full; S 2 and S 4 all slots are used. S 1, S 3 1 slot is free for single packet and we assume that we will make some assumptions right. So, we have makes some definition.

(Refer Slide Time: 21:07)



Poisson arr rate λ, per user.

β = Prob. that at least one packet is generated, in 1 slot [per user]

$= 1 - e^{-\lambda}$

$\bar{\beta} = 1 - \beta$

P_s : Prob. of Successful Trans.

P_b : Prob. of Trans while in a backoff state.

So, beta is the probability that at least one packet is generated; this is for at a given user in one slot. So, if the queue is empty, we will have to make a small assumption. We can relax that is really want to which we will gives slightly more improved performance. We will assume that in one slot only, one packet can be generated; no more than one. If you if you really want, we change this model to simply do that also. We can do that will make the exercise right change the (( )) whether it make that behalf an improvement. But sometimes these things will make difference; sometimes it might not. If I do not assume, depends on the lambda value I am looking at as well as your time slot right.

If time slot is very large, then saying that only one packet will come at time slot is little bit of an approximation is possible that, more than one will come. But we will just assume that at most one packet is going to come; this is all per user. This entire model is on user perspective. In one slot yeah this is per user right. So, what is this going to be? Given that the arrivals are right. So, we have Poisson I did not say that, but I am saying this now. Poisson arrival rate, the lambda which I mentioned earlier; this is per user; not the system byte.

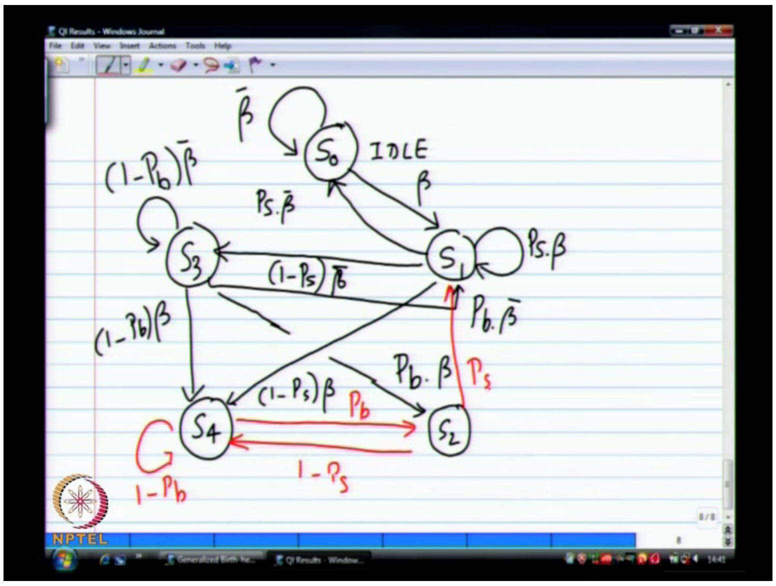
So, what is this probability of no packet generated in Poisson is e power minus lambda, because every lambda in t equals 1 here. So, this is simply 1 minus and I will call beta to be 1 minus beta bar. I have to simply call the e minus e power minus lambda anyway I have to. So, what other parameters do we need to know? So, beta has the probability of one packet at least one packet coming any to another probability of success right. Probability of when I send the packet, what is the probability of success? It can be 1,if there is no other user in the system, but it hold not be 1. It will be some smaller number, if there are several users in the system that we will have to figure outright.

Right If there are n user trying to compete and what is the probability right? That I will succeed; that will be actually a binomial right. We will come to that later. So, P s is the probability of successful transmission that will depend upon how many users are there in the system. But simply assume that there is some variable called P s. Let we can compute that will be a single value, which is steady state analysis and P b is the probability of the back off probability or probability of transmitting while in a back off state. So, within if I am in back off I will flip a coin with probability P b. I will try to resend and1 minus P b I will stay in the same state.

P b can be 1, if you want that becomes very aggressive. If P b equals 1 what will happen? 4 users collide; everybody then keeps on trying in the next slot nothing is going to go through. So, P b has to be relatively small. But P b is very small, then your 1 over P b is that number of slots you have to wait before actually will try to send. The expected number of slots before actually be retransmitting will be very large if P b is small. Because only small and only is the probability P b say, point 0 0 1 will take1000 slots before even try retransmitting. There is extremely passive, they are the extremely aggressive. So, this is the design ratio; what value of P b is going to be…

So, that we will have to assume is given to us. In a real system P b is actually not static. It is dynamic. You look at that the system, how many collisions you have and try to adjust back off probability based on that. In this analysis, I can only assume fixed P b. When I go if I go to dynamic P b, my model is simply collapse. So therefore, I have to deal deal with it, but in my simulation I can do whatever I want to, and then see whether varying the back off probability makes big difference or not dynamically. So, this is our setup so far. Now, we will come to the picture.

(Refer Slide Time: 26:15)



So, we are in state S naught; S naught is the idle state. So, the system will start in state S naught, and then with there are 2 events that can happen when I am in idle state. Again this is the slotted aloha system. So, all event changes take place only in the in a beginning of a slot, so we do not change state in between; that is again convenient assumption. So, if I am in idle of the beginning of the slot; in the next slot beginning I would be either having a packet to send; if some packet had come in this one plot or I will have no packet to send. So, we are this is beta bar. This is beta.

So, this is this assumption making right I could have made this as no packet is e power minus lambda; one packet is e power lambda lambda e power minus lambda; more than one packet is 1 minus these 2 things and then sent to S 2 also. I get S 0 to S 2 to give little bit more accuracy but I am just making that simplification. So, either only one packet will get generated and simply gets in to the queue. So, now how will we let us say draw this other picture also; I will draw S 2 here. (No audio from 27:46 to 27:58) So, from S 1, S 1 I have a packet to send.

So, beginning of a slot there is the packet to be sent. What will be the next possible state? What other transition? And what are the state transition probabilities? S1 to I can go from S 1,ifpacket is successful and no packet is generated I go back to S naught. So, let us deal with that first. So, what is the probability for that occurring? Beta bar right; packet successful and no packet generated. So, I go to P s in to beta bar I self flow part right if I successfully sent this packet and a new packet is generated I will stay in the same state (No audio from 28:49 to 29:03)then what will can I happen?

Can I go from S 1 to S 2?I cannot go from S 1 to S 2;because S 1 to S 2 within S 2 there is a packet queued. But I think only one packet can come right; I really cannot do that. From S 1, I can go to S 3. So, to go from S 1 to S 3 it means that, the packet was not successful 1 minus and no new packet was generated. So therefore, beta bar and I have one more state; possible that I tried to send, it failed and between another packet also came.(No audio from 29:41 to 29:54) So, that say set of transitions, then some of S 2;let we go to S 3 next. So, I was unsuccessful in the previous slot and no packet right.

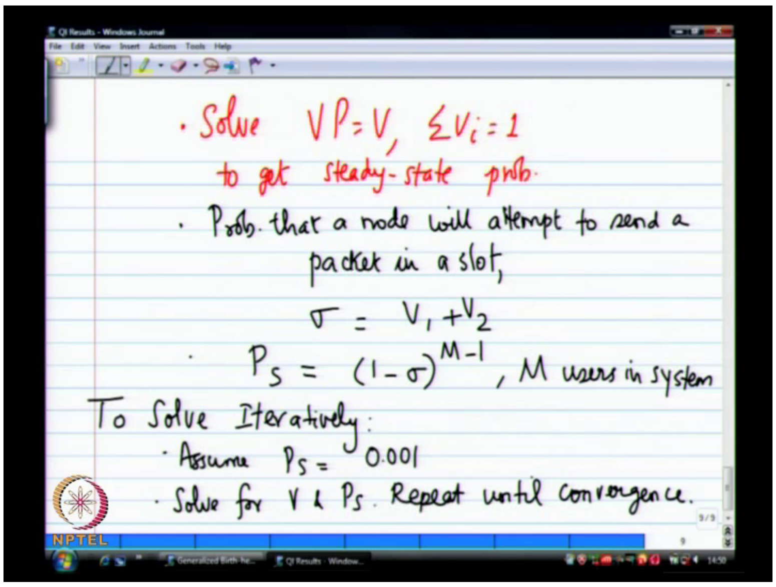
From S 3, what are the transitions? I will go from S 3 to S 3, which means I did not tried back retransmission and no new packet right. So, this is 1 minus P b in to back off probability. P b is may retransmission probability or will call back probability of retransmission and tried and failed. No, I only move from back off to transmit state; this whole, the entire state was in back off; next state, I am going in to the transmit state. So, that will be here. So, this is P b and no packet was generated. Because I am now in transmit mode at the beginning of the slot, but there was no packet in the previous slot. Then we can go from S 3 to S 2.

(No audio from 31:19 to 31:29) I go to S naught with P b in to beta bar, because I was successfully transmitted backlogged packet. No, you are not successful; you are only moving to transmit state. At the beginning of the next slot, you are in transmit state; you did not send right. So, I was in back off at beginning of the slot; at the end of the slot, did I chose to go to transmit state or not and the next slot only, I will send right. So, this entire slot was and here, again I made to sent right. I am trying to send in the next slot and there was a packet generated. (No audio from 32:07 to 32:17) So, have I satisfied all transitions? S 3 not yet; So, S 3 I can stay, I can go to S 4.

If I did not, if I desire it not to retransmit the next slot and one packet get, so alright. So, then in S 4, I can what are the transitions possible at S 4?S 4 in whatever new packets arriving makes no difference; everything is going to be draft. So, either I tried to sent or I do not try; either try retransmission or try to transmission. So therefore, this is simply going to be 1 minus P b. P b alright to transition S 2 I am ready to I am trying to attempting in this particular slot; either I am successful or not successful. New packets do not make any difference, because queue is already full. So, this is going to be if I do not succeed, 1 minus P s and if I succeed, P s.

If I succeed, I go back to transmission with 1 packet right one packet to be sent. I have 2 packets, that is that is no room to store. Say I am only 2 buffers are there right. So, I am at the top of the packet I am trying; in between in that one slot if anything happen; if new packet comes, it simply going to get draft. Therefore, this is no room for store. I think that is essentially right know what do we do? So, we can now try; basically, you want to find out throughput, delay, and things like that. For a particular user, but before that we need to know the steady state probabilities of each state right.

(Refer Slide Time: 34:14)



- Solve $VP = V$, $\sum V_i = 1$ to get steady-state prob.
- Prob. that a node will attempt to send a packet in a slot, $\sigma = V_1 + V_2$
- $P_s = (1-\sigma)^{M-1}$, M users in system

To Solve Iteratively:
- Assume $P_s = 0.001$
- Solve for V & P_s. Repeat until convergence.

So, let us say… So, what we can do is we can solve V P equals V, sigma V i equals 1 to get the… only one catch though P b is specified as an input, lambda is given as the input. These only two things we needed; this P as is actually a system dependent. So, P s will actually depends upon the probability of success. P s Ps actually depend upon the number of users try to compete in the system. So, how do we model that, to be a some of what is happening is, now I am having a iterative process right. I can compute the P s but unfortunately do not know P s and the P s depends upon the P actually.

So, what is the probability that a node will attempt to send a packet in a slot will called is as sigma. For a node for a node to be in transmit state, what is the probability? That it is either in P 1or P 2 right. Say, if I know P 1 and P 2, I can simply say that, I am in transmit state with the probability, sigma. One of those two states, I am sending the packet. Now, there are n users sending the packet right with total of let us say n users are there in the system. Each of them assume that, the system is completely (( )) every node is behaving identically. So, in the longer they will all have the same piece.

So, if I have n users each of them with probability sigma are being in a transmit state in every slot, then what is the probability of success? Simply n choose 1right or where will it is that there is only one state for one process or one user in this transmit state; others are all in the non transmit state right. So, that is our binomial basically. So therefore, I can approximate.

(No audio from 36:38 to 36:48)

This is where, you know will always have confusion, whether this extra sigma is needed or not. Given that, I am in the transmit state. I want everybody else to be quiet; only then I will succeed, and given that there are M users in the system.

I will succeed if nobody else if you going to fight with me whether this sigma it should be there or not yeah yeah, so this this sigma back (( )). In fact, more complicated expression but this is the simplest one. 1 minus sigma M into minus 1 actually will take should take care, but adding that sigma did not work; I have tried it. So, if you want you can; because this is assuming that I will get that this one user in transmit state. What is the probability that, I will be successful is nobody else. Therefore, only other M minus 1 user should be not in the transmit state right any other state.

See, this is our, these are the two transmit states right. If a user is in S 1 or S 2, it means user is sending and the probability of that, where P our P of 1, P of 2 is the steady state probability of being in state 1 and state 2.Let us yeah we said V 1 and V 2 and I have said P 1 and P 2. So, this technically is V 1 and V 2, probability of being in state 1 or state 2. So now, I have P s equals this. So, now I have a set of markov equation which I can simply plug in to right. In fact, if you those days I used actually, manually solved right I will write over these all expressions like you do in your test and then solve it.

We did not did not had mat lab mathematical at that time which I did not really used. I simply solve this equation, then plug those equations in to only the equation (( )). But any way, I can come up with the transition metrics, then I have to initially start with some value for P s right. So, the way to solve is(No audio from 38:58 to 39:07) right Assume P s equals some small value 0.001,I have forget what I used and then solve because I am assuming taking some right then solve for the vector V and P s and then repeat until convergence. We have to have some condition for convergence saying that, ultimately sigma converges to some small right.

Sigma 1 in the first iteration, if I get sigma equals this; the second iteration we get different sigma, then the difference should be some small epsilon right. So, repeat until convergence on some parameter, whether it is P s or sigma; whatever it is that is you want. Once you find that, there is no change in the transition probability and the way that steady state probabilities, then you can stop right repeat right. So, you need P s to start to compute sigma, then you use the sigma to recompute the P s and then do this forever. And what we found is about 50 (( )), in most of the cases converge.

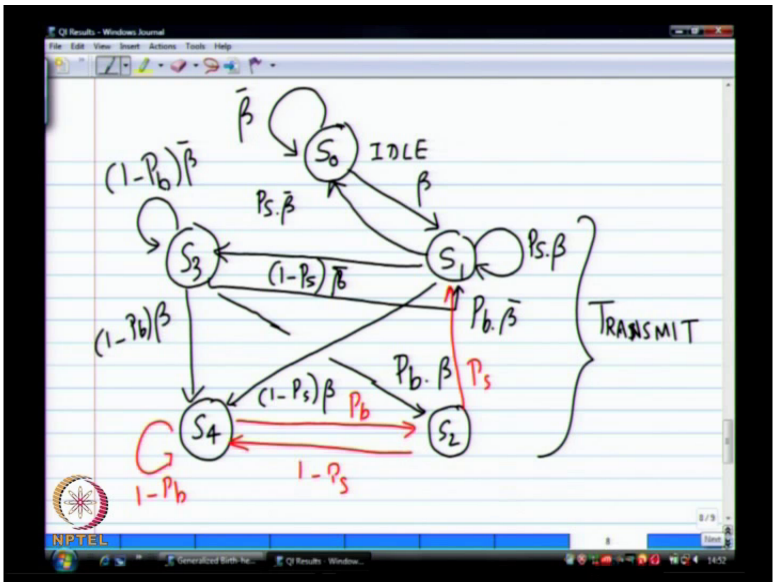
But the lambda is very large; of course, it will not converge; lambda will actually probably it still whatever it is lambda is larger going to go to (( )) 0.36 or something like that. (No audio from 40:30 to 40:48) So, then now I have the steady state probability of being in the nature now what is the throughput of the system? What is the throughput of the system from the user perspective? Successfully transmitted over time slot; so per time per users probability of success is sigma right. So, will be simply sigma sigma into M sigma in to M. So, each user throughput will be a share right, this only per user throughput that M c.

So, it is to be users total system throughput will be sigma into M, because sigma is the probability of sorry P s into M. P s is the probability of successful transmit not sigma right. P s is the probability that, I will successfully transmit and it is going to be Min to P s, yeah some approximations are there, we will see. If not think before next class, because I am going to stop in a couple of minutes and then delay how you do find delay? P sin to M, why is it P s in to M? So, the probability of a node successfully transmitting in a slot is P s; that is my throughput. But now throughput is P m P s right.

P s also the per not through put; P s probability of successful transmissions; yeah given that I am in sigma into P s in to M. Then I should succeed yeah. So, that is why that again that one sigma is there right. sigma into P s, probability of being in a transmission state and probability; so sigma into P s. That into M will come, because I am looking at the system throughput as such. Because each user throughput will be ok that we will come back and continue. And delay per user is easy to find out, how I can compute E of n right; (( )) reward function business.

I simply say the number of customer here is 0; number of packet say here 0; this is 1; this is 1; 2 and 2; look at the probabilities right. I can compute the average buffer occupancy of the queue in the system by using reward function. Once you know the P steady state probabilities, I can compute the average number of packets in this queue that divided by the lambda is my delay. So, this way I can compute delay and throughput in the system. I will continue on specifics of that in the next class, but that is the basic principle right.

(Refer Slide Time: 43:27)



So, now If I give you as aloha system with M users and lambda and P b simply right crunch out this P s to be and so on. Write an iterative equation that will simply solve this. You can write it and see an interface to something else or actually given even solve the closed form equations very simply, if you want to… We will simply have a set of closed form expressions that you will solve with then give you all the values. In this system the user will wait at least one slot before retransmitting. In this particular system at the end of… No I… Then I have to wait one slot yeah before retransmission. Yes, correct.

But in real system you can retransmit in immediate next one yeah yeah you can. So, then if I really want to include that, then my S1 from S 3 to S naught I can go. If I do not want to wait that one slot yeah these are some of the approximations that you try to door in which case S 3 will also be a transmit state that right. The S 3 will If I allow S 3 transmission in S 3, then I will have to change my parameters my transmission probabilities accordingly, but S 3 will then also be a transmit state.

Then, sigma will be equal to whatever P 1 plus S 2 right. V 1 plus V 2 plus V 3 plus V 4if all of them I can have transmissions. So that one extra slot will (( )). So, this is using plain Markov right. Now, what you find is that just 2 minutes I will say and then stop. The self loops are here; this is the right. So, the I can replace all the self loops by simply stating that rather have this system move on a slot by slot basis, I can have the time spent in each state to be n th…This case will be exponential right, I can simply replace this self loop, and replace this beta by 1. I will come back on this. They have your (( )) one problem will we can do this approximation with semi markov.

I will come back in deal with semi markov separately. I can replace self loop basically with the switch on time of each state. I do not need to have right, I do not have to have this extra self loop. I can remove the self loops, and then instead I will say S naught, the time spent in S naught is basically 1over beta right. And then with that I can all the self loop can replace just corresponding switch on time in each of the state. Then I can again re compute this. Then it becomes the semi Markov model, which will come back in deal with on Friday.