Advanced Operations Research

Prof. G. Srinivasan

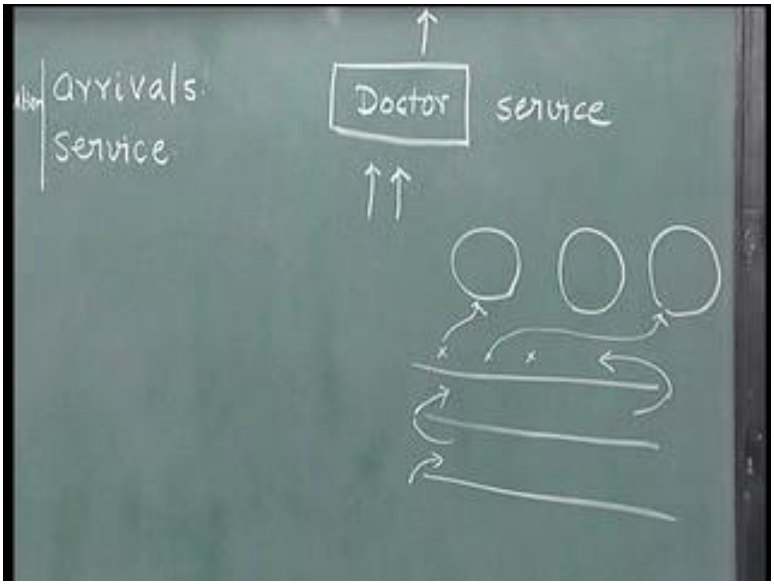
Department of Management Studies

Indian Institute of Technology, Madras

Lecture - 30
Queuing Models

In this lecture, we are going to see basics of queuing models. Queuing models are also called waiting line models. Here, we study about what happens when an individual or a set of people come and join queues. We are quite familiar with queues in our day-to-day life. Common examples of queuing models that we encounter are going to a doctor or going to a barber shop. A queuing system essentially happens when there are entities or people who are called arrivals who require a kind of service from another entity. There is a service and there is a line or a queue where a person is joining this system.

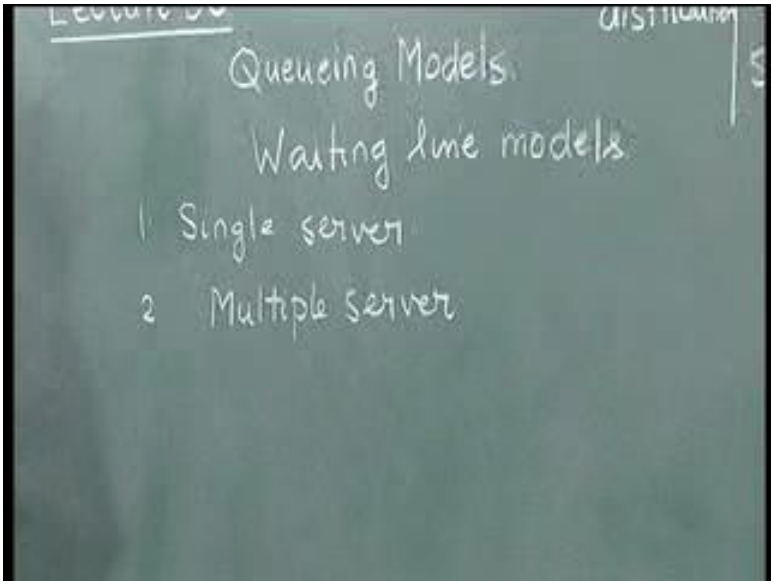
(Refer Slide Time: 01:38)



If you take a typical example of a doctor; people arrive, spend some time, get served and people leave. What characterizes this queuing system or what makes it little different from the earlier models is the fact that these arrivals and service are assumed to follow some distributions. They are not deterministic but they are assumed to follow certain distributions. So, people arrive

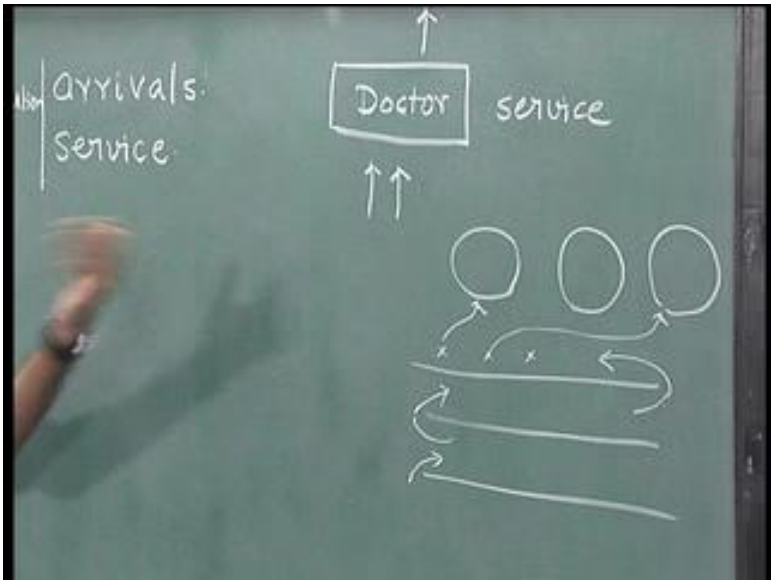
according to the given distribution or according to a certain distribution, the service is also provided according to certain given distributions.

(Refer Slide Time: 02:39)



Queuing models can be of several types. First category is called a single server queuing model where there is only one server. We also have multiple server queuing models where there are multiple servers.

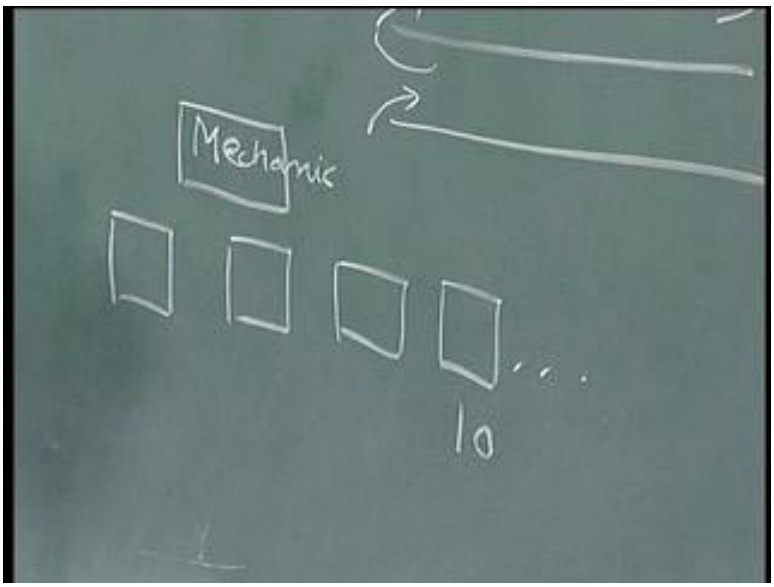
(Refer Slide Time: 03:07)



A good example of a multiple server queuing model is a railway reservation system where we could have several counters where people who come join this line and whichever server is free the first person will go, then the next and then the next person goes and so on. So, we have a multiple server model where there is more than one server. Here, there is a common line and as soon as a server is free, this person will join, get the service and leave.

Within the single server and multiple servers there are two categories. One is called a finite queue length and infinite queue length. The infinite queue length model assumes, that every person who comes joins the line, for example, if already three people are waiting for the doctor, the fourth person will join the line and so on. There is no restriction on the number of people who are actually waiting or there is no restriction on the length of the queue. The queue length can theoretically be infinite so it can go on and on. In finite queue length models we try to restrict the queue length to a certain limit after which we say that if this threshold limit is reached, people who come into the system do not join the system.

(Refer Slide Time 05:14)

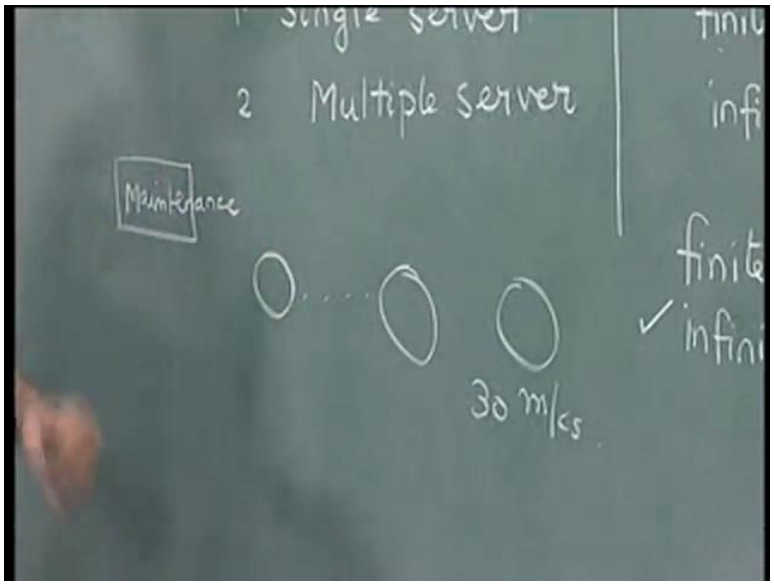


A good example of a finite queue length model is a garage. Let us say there is a garage with a single server or a single mechanic. There is a mechanic here and let us say this garage has space to park, say ten cars. For example, someone is coming into the garage to give his or her car for service if some slots are available which means the number in the system is less than 10.

Remember, these are the spaces for the garage, this is not garage this is the server. These are the places in the garage, if the number of cars in the garage including the one that is being serviced is less than 10, then the person who is coming in will have a space to park the car, the person will leave the car and go. When some person is coming, the garage is full which means all the ten slots have been taken up including the car that is being serviced, then this person does not find a place to park his or her car. The person will leave the system without joining the line without getting served. Such models are finite queue length models, whereas ordinarily we have infinite queue length models.

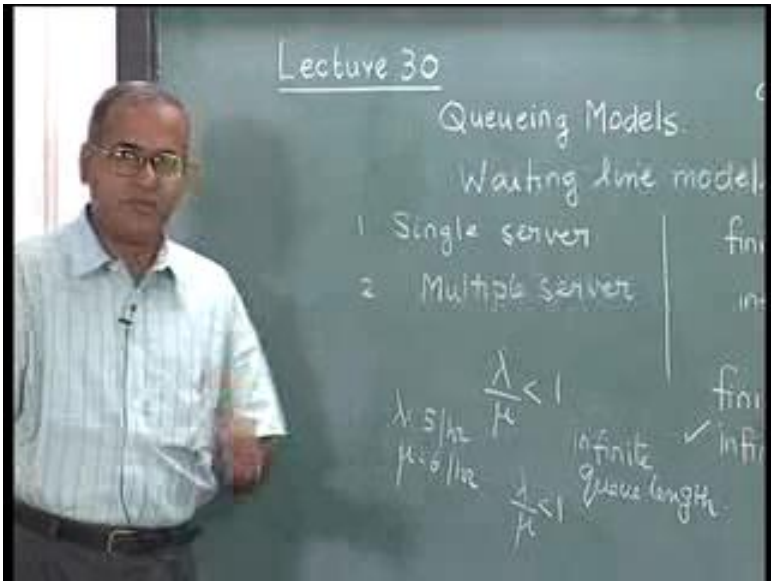
There is one further classification which is called finite population models and infinite population models. If we take the example of the doctor or the reservation system or the car mechanic they all come under the category of what are called infinite population models. This is an example of a single server, infinite queue length and infinite population. This is an example of a single server, finite queue length and infinite population models. How do we categorize something as an infinite population and finite population? In this case, the population simply represents anybody who can come for treatment or service to this doctor and therefore it can be infinite.

(Refer Slide Time 08:14)



A good example of a finite population model is like this. If we have a factory and if this factory has about thirty machines and if there is a dedicated maintenance team, this maintenance team attends to calls every time these machines breakdown. Then we have a system where the breakdown represents the arrival of a job for the maintenance team. The service is the time or the service provided by the maintenance team to attend to these breakdowns. There is a finite population because only any one of these thirty machines when they breakdown, this team will come and attend them. This is an example of a finite population situation. Ordinarily we discuss more of infinite population situation compared to finite population situations.

(Refer Slide Time: 09:24)



The queuing system is also characterized by the distribution of this arrival and distribution of this service. The arrivals and services can follow any given distribution or we can go observe physically, what is happening in the queuing system. From the data that we can collect from what is actually happening we can fit a corresponding distribution. Most of the times it also observed that arrivals follow a Poisson distribution, with arrival rate called lambda per hour.

Lambda usually denotes the arrival rate in a queuing system. It is also observed from practice, that service times are exponentially distributed, at the rate of mu per hour. We also observe that Poisson distribution and exponential distribution are related. Poisson distribution has an important property called the memory less property based on which we will derive some

expressions for the performance of the queuing systems. Essentially, we will assume right through this lecture series that arrivals follow a Poisson distribution with lambda per hour and service times are exponential, denoted by mu per hour.

What is the relationship between this lambda and mu?

We also need realize that lambda by mu is less than 1, particularly, if we have an infinite queue length models. For example, we assume lambda equal to 5 per hour and mu is equal to 6 per hour, then lambda by mu is less than 1. What happens, here lambda is equal to 5 per hour means on an average five people enter the system every hour, which means, on an average every twelve minutes a person enters the system and on an average every ten minutes a person gets served and leaves the system.

What happens when lambda by mu is greater than one?

For example, if mu were not 6 per hour and mu were 4 per hour. Then, every hour five people on an average enter the system and four people on an average leave the system, so, the queue length will automatically increase by 1 every hour and this means that somebody who joins the queue will never get served. For a queuing system to be efficient, particularly, when we have infinite queue length, lambda by mu has to be less than 1.

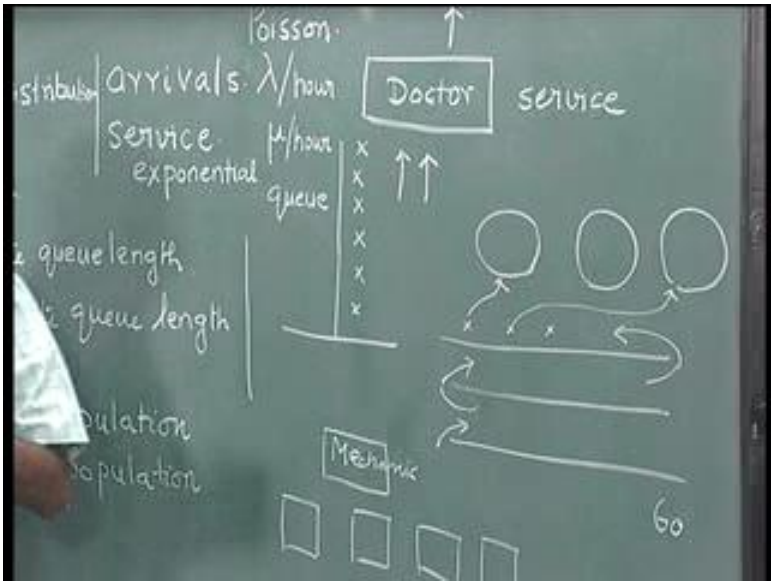
If we have finite queue lengths we may have lambda by mu greater than 1 because in finite queue length models depending on what we have here, people may not join the line or join the system. Therefore, we are not that worried about lambda by mu being greater than one. Since some people do not join the system and go away, everybody who joins the system there will get an opportunity to be served. We have to be very careful and we should have lambda by mu less than 1. Particularly, when we have infinite queue length models and lambda by mu can be greater than one.

If lambda equal to 5, mu equal to 6, lambda by mu is less than 1. This means, every hour on an average five people enter the system while six people will leave the system. We may be tempted to ask a question why should we have a queue at all or a line at all, when five people enter the system and six people can leave? The answer comes from the fact that, this is an average or

expected value of a distribution. This 5 per hour does not means that exactly every twelve minutes a person gets in.

The inter arrival times are exponential; the arrival rate is Poisson with lambda per hour, it means, on an average five people enter the system following a Poisson distribution. It may happen between two consecutive arrivals, it could be five minutes, it could be twenty minutes, but at steady state, if we measure then we will have five people per hour entering the system. Therefore, there will be a line even though lambda is less than mu. There will be a line and we are interested in analyzing the performance of this queuing system or this line. There are a couple of other things that we need to look at, before we start deriving some expressions for the queuing models. There are three other terms which are often associated with queuing and these three are called balking, reneging and jockeying.

(Refer Slide Time: 15:43)



What is balking?

Let us assume that, we are going to book a ticket in a railway reservation system. Let us say that we have these three servers. Let us assume we are in a slight hurry and we enter this system, and we observe that there are already some sixty people waiting in the line. As this person enters the person forms an opinion of how much time it is going to take for this person to finish this service and leave. If this person thinks it is going to take a lot of time, then the person leaves the system

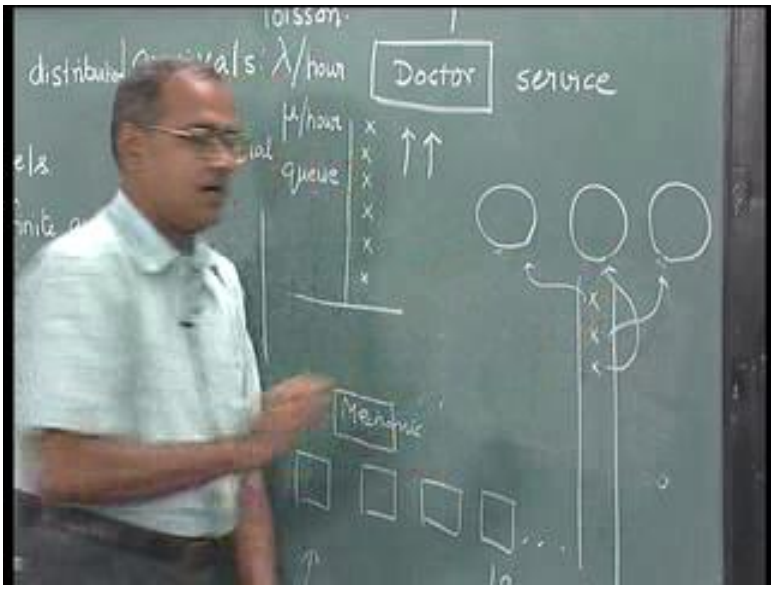
without joining the line. If an arrival does not join the system and leave, then the arrival is set the balk. The person balks when the person does not join the line and leaves.

Balking can again be of two types: Forced balking and an unforced balking. For example, when you enter as a sixty-first person in a line and you think it is going to take a lot of time, therefore, you do not join the line and nobody is forcing you to leave. That is an example of an unforced balking.

On the other hand, if you go take the mechanic example with space only for ten cars, including the car being serviced and you drive your car for service and you realize that all these ten cars are full then you do not have a choice, you leave the system then such a balking that is called forced balking. Whenever we have finite queue length models, then there is a chance that balking occurs. The second phenomenon that happens is called reneging.

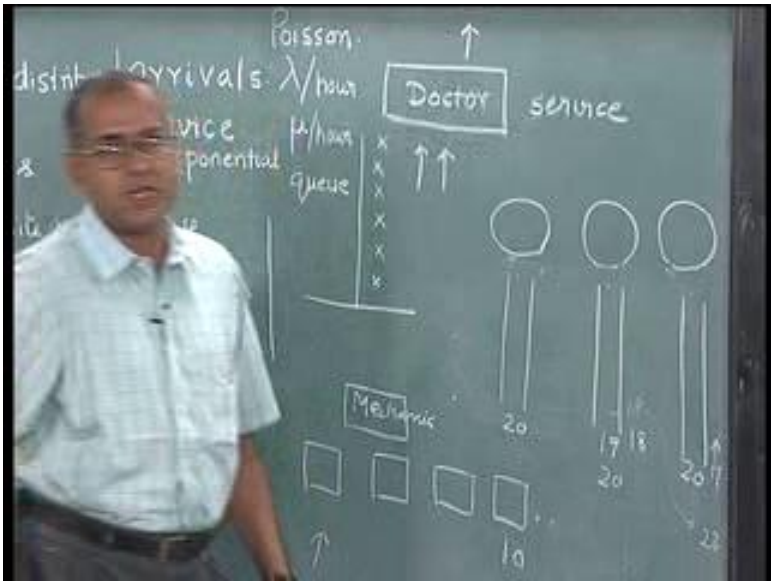
Again come back to this example let us say you enter the system to book a ticket and there are sixty people already waiting. Then you realize you wish to join and say you join as the sixty-first person in the line. When you join as the sixty-first person you expect the line to move in a certain speed. After a while, let us say you observe that there it is actually moving slower than what you thought and your time is running up, then half way through from somewhere in the middle of the line you decide to quit, you just come out the line and you go away. Such a phenomenon is called reneging. The person joins the system, but after some time decides not to continue and simply moves out of the system, called reneging.

(Refer Slide Time: 18:16)



The third one is also an interesting phenomenon. It is called jockeying. For example, if this system, we have multiple servers but there is a common line. People are waiting here now depending on which server is free, the person will automatically go to the corresponding server and solve.

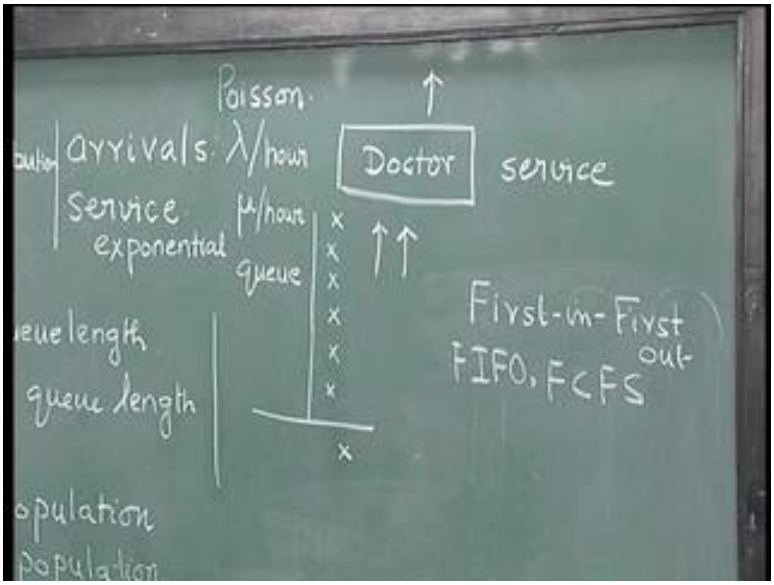
(Refer Slide Time: 18:49)



Instead, if we had a system where there are three servers and each server has a dedicated line let us say you come to the system and you find about nineteen people here and about twenty people here and fifty nine people here including the people who are being served we normally would join here or we would join here, we would join one of these lines. Many times there will be a tendency to join the shortest line. Let us say, you join the shortest line as a twentieth person. All the three lines are moving towards the server and more people have actually come in and joined. After five minutes you realize that, this is moving slightly faster. You have now become the eighteenth person here, but if you realize that, if you actually shift this line, you can become the seventeenth person in this line, because this line is moving faster.

There will be a tendency to shift from one line to another line for a while and come back depending on what we think which is the rate at which people in these lines are moving. Such a phenomenon is called jockeying. Ordinarily, jockeying happens within the first few minutes of joining the line. Once we realise that there are four or five people behind us. If you skip this, if you are the eighteenth person here and then if you jockey, you will become the twenty-third person here then you will not like to jockey. Nevertheless, jockeying happens at the beginning. When we enter we find the queue lengths more or less the same as we enter any queuing system which has multiple servers and dedicated lines. Jockeying is something that is common to multiple server queuing system.

(Refer Slide Time: 21:15)

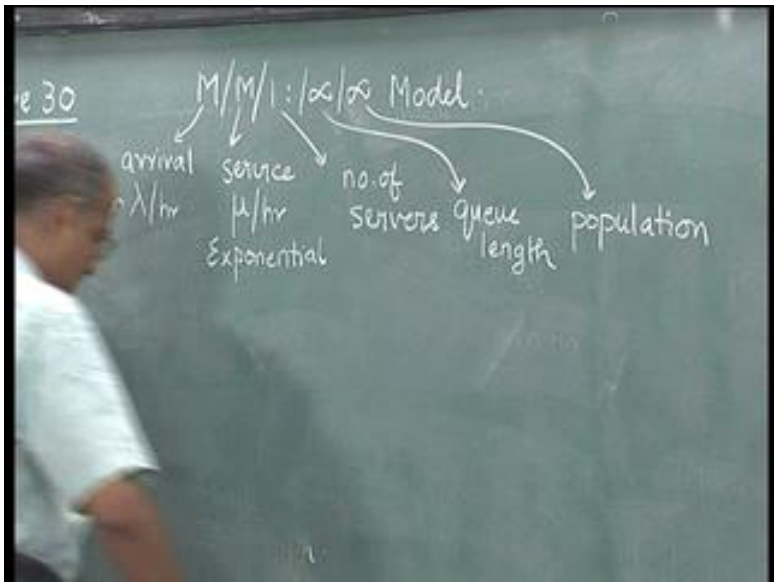


Another parameter that we need to discuss before we derive models is called the queue discipline or the service principle. If we take the example of a doctor that we have seen let us say there are six people already in the line, you join as the seventh person So ordinarily you would know and you would expect the queue discipline to be what is called first-in-first out, also called first come first served, which is either FIFO or FCFS. Ordinarily, when human beings are involved and the service provider is also a human being it is customary to assume a first-in-first out service discipline or first-in-first out rule to send the next person into the system.

When we do not have human being in the system the service discipline can vary or can change. There are times we can have a last-in-first out system and so on. But most queuing models are also derived under the principle first-in-first out system. Steady state models usually are independent of the service discipline whereas transient models involve the discipline and are derived using first-in-first out models. With this kind of an introduction we will look at four different queuing models in this lecture series. We will not consider aspects like balking, reneging and jockeying. We are not going to consider finite population, we will consider single server model, multiple server model and within each of these we will consider finite queue length and infinite queue length.

Two types of servers single server and multiple server models and two types of queue lengths which is finite queue length and infinite queue length, would give us four basic models of queuing theory. We will first look at a single server, infinite queue length model then single server, finite queue length model; multiple servers, infinite queue length model and multiple servers, finite queue length models. This will be the order in which we will study these models.

(Refer Slide Time: 24:14)



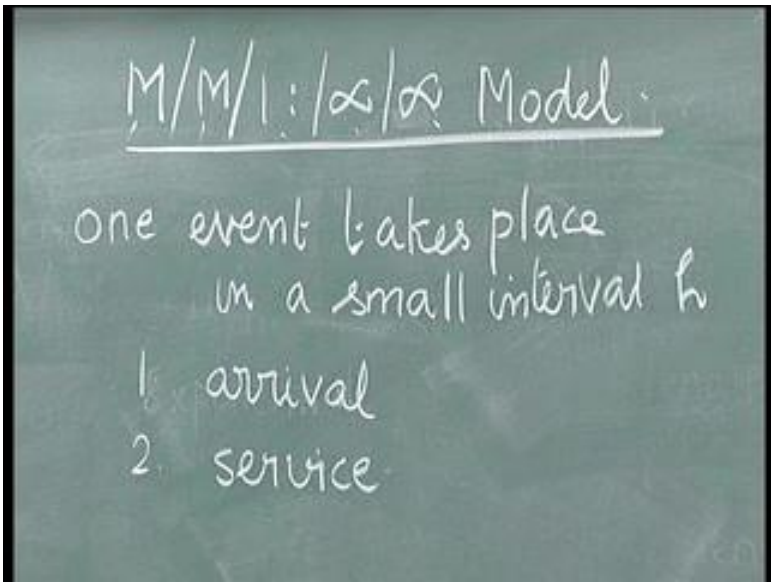
We start our discussion on the single server infinite queue length model. We will represent this model following a certain notation. The notation that we use will be called M/M/1 /infinity/infinity model. This is part of a very familiar notation called the Kendal's notation. We have used a portion of the Kendal's notation. I have left out one of the spaces in the Kendal's notation. This M characterises the arrival, this M characterises the service. The M and M indicates the memory less property of the arrival and the service, or Morkovian property of the arrival and the service. These come here, because of the assumption, that it is Poisson that follows the Morkovian property of lambda per hour. This is exponential with mu per hour. This 1 represents, the number of servers, so, it is a single server model. This infinity is the queue length, so it is an infinite queue length model. This represents the population, so it is an infinite population model.

M/M/1/infinity/infinity model would mean a single server model with Poisson arrival and exponential service following the memory less property, the infinity, infinity represent infinite queue length and infinite population model.

(Refer Slide Time: 26:32)

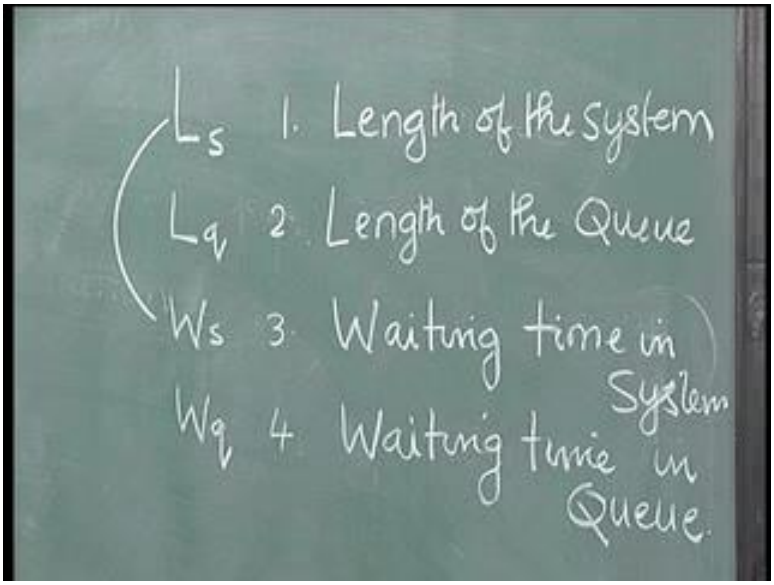
The queuing system will be like this. There is a server, who serves at the rate of mu per hour. There are arrivals at the rate of lambda per hour. In the memory less property, we are going to assume that the behavior of the system does not have any memory. Therefore, it does not take into account the earlier states of the system in order to define the present state of this system.

(Refer Slide Time: 27:22)



The memory less property also helps us in one important property which is an important assumption here, that during a very small interval only one event will take place. One event takes place in a small interval h. An event is either an arrival or a service. What we want to study is in a system like this, at steady state, what is the probability, that at steady state this system has zero people, one person, two people and so on. This can go up to infinite. What are all the other things we wish to study in this system? Typically, from a user point of view or from a customer point of view the customer is interested in four important parameters.

(Refer Slide Time: 28:37)



These four parameters are called length of the system, length of the queue, waiting time in the system and waiting time in the queue. What is this length of the system and length of the queue? Length of the system is the expected number of people who are actually in the system, including the person who is being served. Length of the queue is the expected number of people who are waiting for service in this system. Obviously, there is a relationship between these two. A person who enters the system would actually like to know when the person will actually leave the system. The person is interested in waiting time in the system and there are times the person is interested in waiting time in the queue.

Normally, if we are interested in going for specialized services, say a doctor or a legal service then the time taken by the server also matters. There are times we are worried about waiting time

in the queue, we are worried about when our service is exactly going to start So in such a situation we are concerned about the waiting time in the queue.

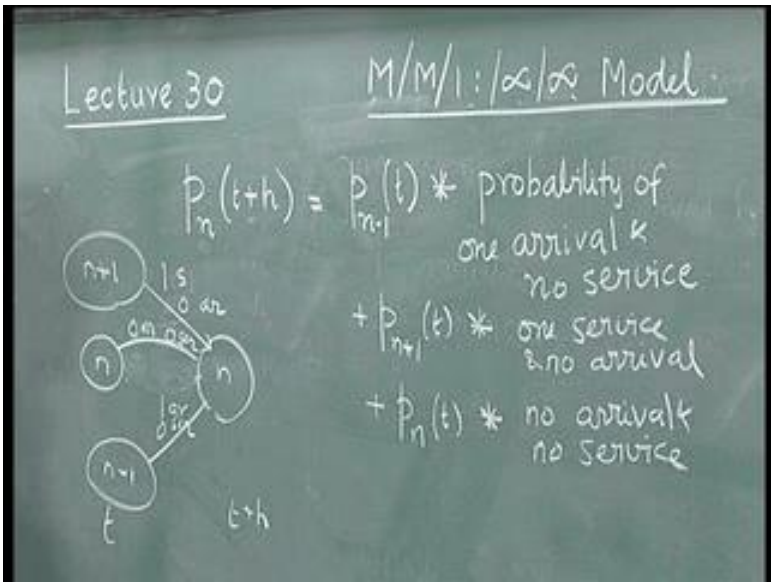
If we have a restriction on the total time that we want to spend in the system, then we look at waiting time in the system. Ordinarily, these are represented as L_s, L_q, W_s, and W_q. It is also obvious that these two are related, these two are related and there is a relationship between L_s and W_s which means all four of these are actually related to each other.

If we derive an expression for one of them from first principles we can actually use the relationship among these parameters to derive expressions for the rest of them. This queuing system finally is going to be measured for equations that define L_s, L_q, W_s, and W_q. These are expected values, expected number of people, expected waiting time and so on. Therefore, these depend on the probability that there are n people in this system. Probability that there are zero people in the system is p_0, p_1, p_2 and so on. This can go up to infinity.

We are interested in getting expressions for p_0, p_1, p_2 etc which represent, the steady state probabilities that there are 0 1 2 etc., people in the system. Based on these probabilities, we will derive expressions for L_s, L_q, W_s and W_q. In order to get expressions for these p_0, p_1, p_2 etc., the three important things that we know are lambda, mu and the number of servers, in this case, this is 1, is called C equal to 1. C represents the number of servers. The input values to the queuing system are lambda, mu and C. The output values are L_s, L_q, W_s, and W_q. The intermediate values are p_0, p_1, p_2, etc.

We start deriving some expressions to get a general p_n, the probability that there are n people in the system in terms of lambda, mu and C. Once we get an expression for p_n, we can substitute for various values of n as 0 1 2 and so on. we can get these from which we can go back and get these expressions.

(Refer Slide Time: 33:31)



Let us start deriving some expressions for this; p_n (t plus h) means, the probability that there are n people in the system at time (t plus h) where h is small. There will be n people in the system at time (t plus h), if we had (n minus 1) people at time t and during this small period h, one person has arrived and no person has left. So, p_n (t plus h), probability that we have n people at time (t plus h) can happen, when there are (n minus 1) people at time t and in the small period there is one arrival and no service. This can also happen if there are (n plus 1) people at time into there was one service and no arrival plus there where n people into no arrival and no service.

This is obvious that we have n people here. That can happen if we had (n plus 1) at time, this is time t, this is time (t plus h). One way it can happen is there were (n plus 1) people here and one person has left. So, one services zero arrival or we could have had (n minus 1) people here and in this small gap we would have one arrival zero service. We could have n people here; we could have had zero arrivals, zero services or one arrival one service.

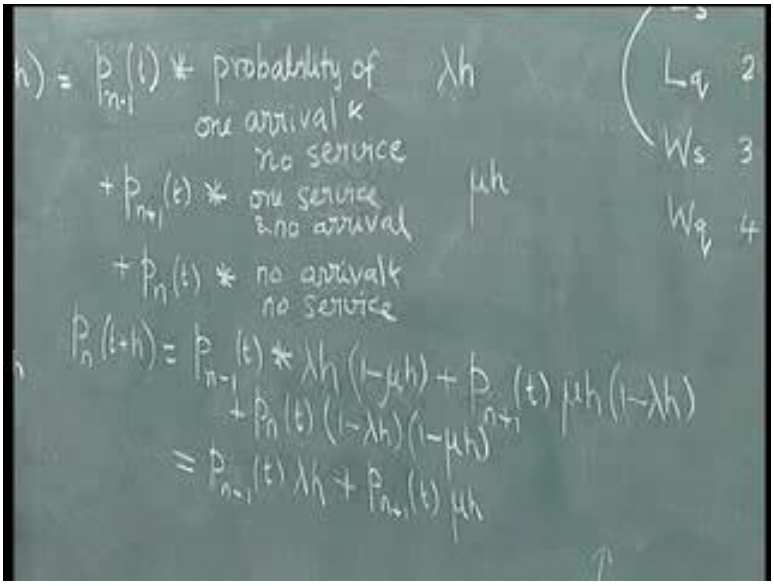
We are not considering other possibilities for the reason that in a small period only one event can take place. That event is either an arrival or a service. Therefore, we are not looking at (n plus 2) people in the system and two people leaving in this small period h. Similarly, we are not looking at any other value other than (n minus 1). We do not say (n minus 2) people were there and there were two arrivals. Now, coming back to this n, two things can happen. There is a zero arrival and

zero service; there is a one arrival and one service. Now, one arrival and one service means, two events. We have made an assumption that we will not have more than one event taking place. So, we do not consider this as part of defining the probability, that there are n people in time (t plus h), using n in time t.

That is the reason I have not written the fourth step, which is one arrival and one service. That means, there are two events and therefore we do not write that. These things come from the memory less property of this system. Which also says, it is not state dependent as such, the system does not respond to what it was doing in earlier periods.

System does not have memory and therefore does not carry whatever it was doing in earlier stages than what we are looking at. Therefore, based on the memory less property which the Poisson and exponential distributions provide us with, we can write this expression. Coming back to this expression, now probability of one arrival during any given period is lambda h and probability of one service is given by mu h.

(Refer Slide Time: 38:16)

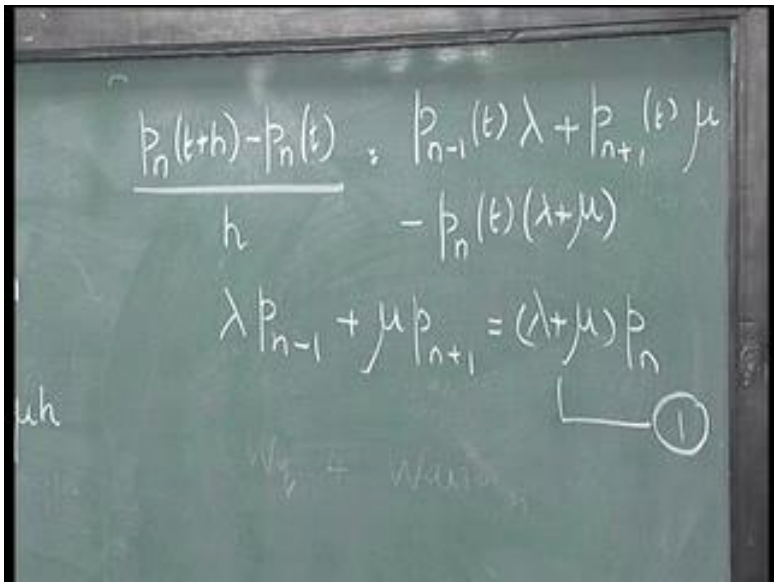


When we when we do this, we can write this as p_n of (t plus h) equal to p_{n-1} at time t into probability of one arrival, which is lambda h. Probability of no service is 1 minus mu h; because mu h is the probability of one service or one person leaving the system. So, the probability of zero people leaving the system is 1 minus mu h.

We do not consider two persons leaving, three persons leaving and so on. Therefore, the probability that zero person leaves plus, probability, that one person leaves is equal to 1. Therefore, probability of no service will become 1 minus mu h, plus p n+1 t into probability of one service and no arrival. So one service is mu h, no arrival is 1 minus lambda h. For the same reason, one arrival is lambda h. Therefore, zero arrival is 1 minus lambda h, plus p_n of t into no arrival and no service, so 1 minus lambda h into 1 minus mu h. This is the expression that comes from this. This we can simplify. We simplify and we leave out the higher order terms. On simplification, this will become p_{n-1} of t lambda h minus lambda mu h square. So lambda mu h square is a second order term. We leave out the second order term.

We get p_{n-1} t into lambda h plus p_{nplus1} at time t into mu h, plus p_n of t into 1 minus lambda h minus mu h plus lambda mu h square. So, second order term is left out, so this into 1 minus lambda h minus mu h.

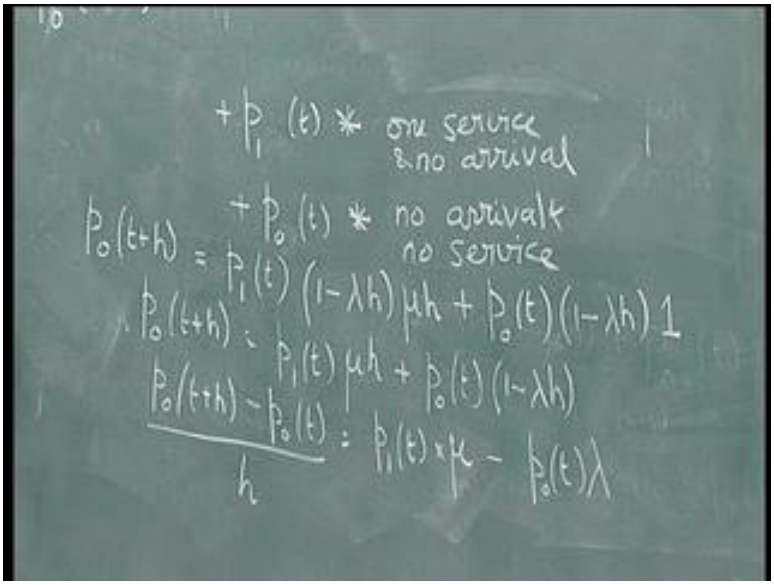
(Refer Slide Time: 41:03)



We bring this p_n (t to the left-hand side. We will get p_n of (t plus h), p_n probability that n people are there and dividing the whole thing by h, h is common in all of them, would give us p_{n-1} at time t into lambda plus $p_{(n\ plus\ 1)}$ at time t into mu minus p_n time t into lambda plus mu. This 1 will come to the other side divided by h. From this equation, if we apply the steady state condition which means the change of p_n by t to the interval h, will be 0 at steady state.

At steady state the probability is not going to be time dependent. When we apply the steady state condition, this portion becomes 0. We have p_{n-1} into lambda or lambda p_{n-1} plus mu p_{nplus1} equal to lambda plus mu p_n. If p_{nplus1} represent the steady state probability, that there are (n plus 1) people in the system, p_{n-1} represents the steady state probability, that there are (n minus 1) people in the system. Now, p_n represents the steady state probability, that there are n people in the system. Then the equation that is joining them is lambda p_{n-1} plus mu p_{nplus1} equal to lambda plus mu into p_n. We call this as equation number 1.

(Refer Slide Time: 43:17)



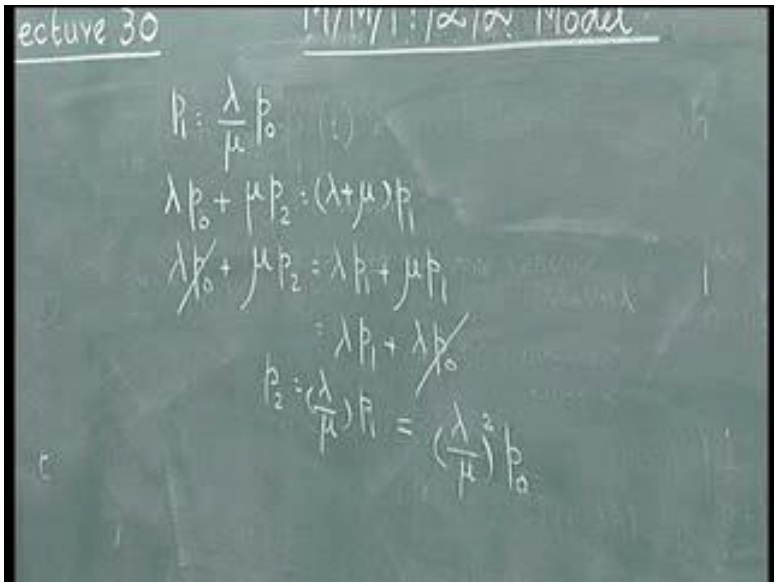
We go back and do this. Make a slight difference here. Then, we derive one more expression here, which is this. Probability, that there are zero people in the system at time (t plus h) will be probability, that there is one person in the system at probability that there are 0 people in the system at time (t plus h) is probability that there is one person in the system at time t. There is one service and no arrival plus probability that there are zero people in this system, there is no arrival and no service. This, when we expand, we will get p_0 time (t plus h) equal to p_1 of t into no arrival and one service, which is 1 minus lambda h into mu h. Probability of one service is mu h, probability of one arrival is lambda h, so probability of no arrival is 1 minus lambda h plus p_0 t into probability of no arrival, is 1 minus lambda h, but probability of no service is 1, because there are already zero people in the system, so there is no service. So, probability of no service is 1. Please note this difference; earlier when we had other than zero, then there could be a service.

So, probability of service was mu h, probability of no service is 1 minus mu h, because we have zero people in the system; probability of no service is 1.

When we expand this, p_0 (t plus h) equal to p_1 (t) expanding and leaving out the higher order terms into mu h, plus p_0 (t) into 1 minus lambda h. Now, taking to this other side, p_0 (t plus h) minus p_0 (t by h), will become p_1 (t) into mu, minus p_0 (t) into lambda.

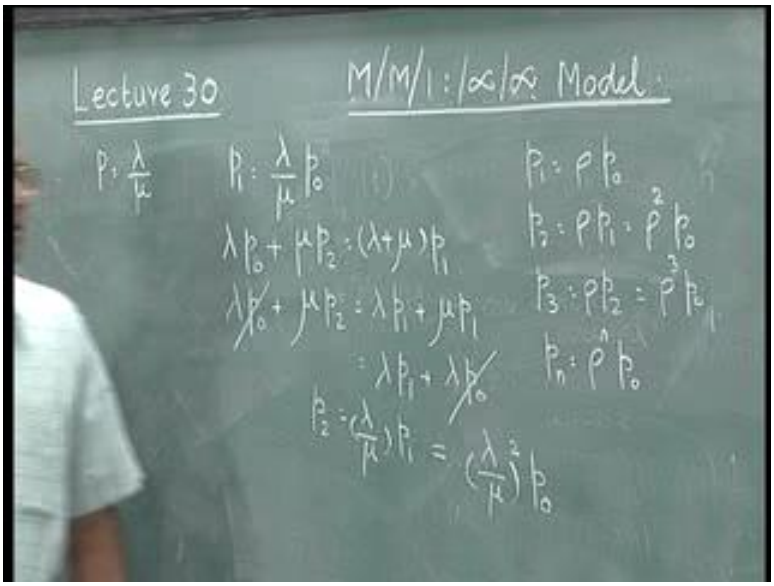
Again, applying the steady state conditions, at steady state, the rate of change of p_0 with respect to h is 0 because we assume that p_0 is a steady state probability. So, there is no change to the respect to h. This will become 0 and p_1 (t) will simply become p_1 and p_0 (t) will simply become p_0. So, this will give us a second equation mu into p_1 equal to lambda into p_0. This is the second equation that we have.

(Refer Slide Time: 47:24)



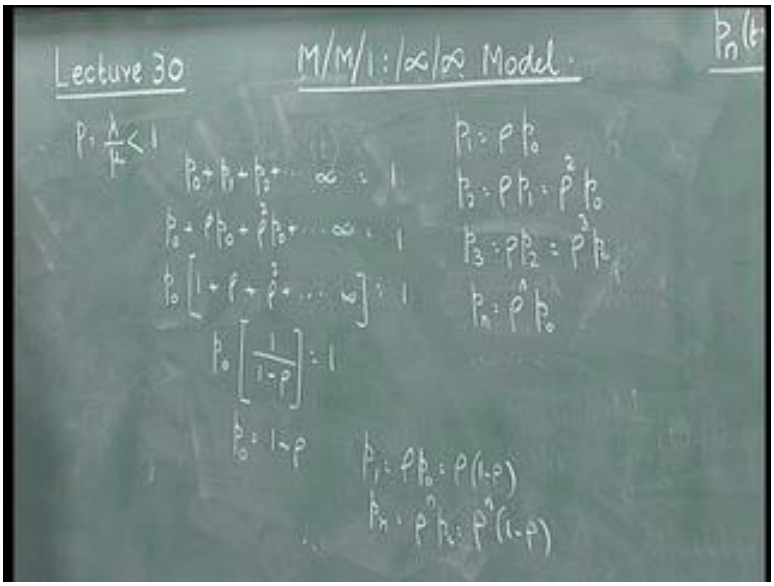
From these two equations, the first thing we can derive is p_1 equal to lambda by mu into p_0. This is the first thing that we can derive. Now, we go back and apply this to this one. We get lambda into p_1, put n equal to 1, we get lambda into p_0 plus mu into p_2 equal to lambda plus mu into p_1. This is lambda p_0 plus mu p_2 equal to lambda p_1 plus mu p_1 when we expand. From this we know mu p_1 equal to lambda p_0. This is lambda p_1 plus lambda p_0 so the lambda p_0 gets cancelled. From this we get p_2 equal to lambda by mu into p_1.

(Refer Slide Time: 49:04)



We already know that p_1 equal to lambda by mu into p_0 so this is lambda by mu the whole square into p_0, p_1 equal to rho p_0; p_2 equal to rho p_1 equal to rho square p_0. In the similar manner, we can derive p_3 equal to rho p_2 equal to rho cubed p_0 and any general p_n equal to rho power n. This we can derive by progressively substituting in the earlier equation. These are the important equations that relate this. At the moment, we know now that p_1 equal to rho p_0; p_2 equal to rho square p_0 and so on. We still do not know the actual values of p_1, p_2, p_3 or p_n because they have all been derived as dependent on p_0. Unless we know p_0, we cannot find out the values of p_1, p_2, p_3 or a general p_n. To find out p_0, we go back to the normal rule, that is the sum of the steady state probabilities is equal to 1.

(Refer Slide Time: 50:16)



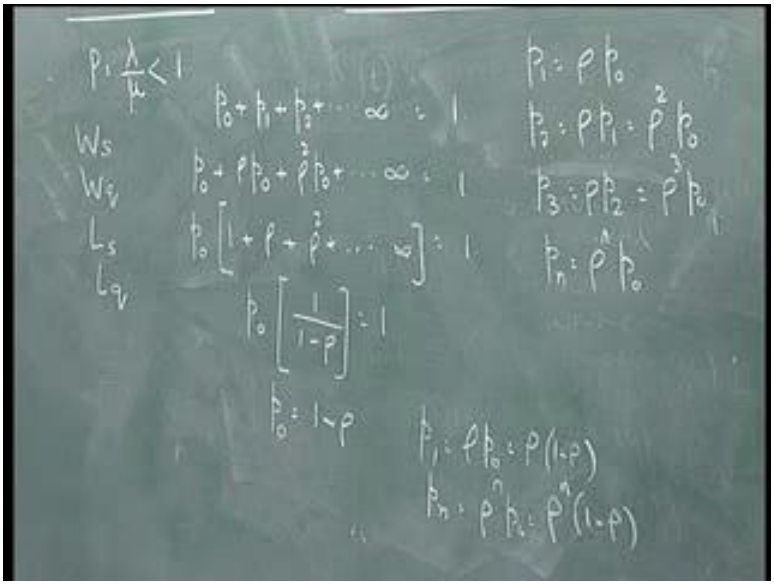
We have p_0, plus p_1, plus p_2, plus up to infinity is equal to 1. This would give us p_0 plus rho p_0 plus rho square p_0 etc, up to infinity is equal to 1. This is p_0 into 1 plus rho plus rho square plus up to infinity is equal to 1. An interesting thing is the infinite series which is the first one. It is a geometric series because you have 1 rho rho square etc, we know the sum to infinity terms of a geometric series as a by 1 minus r provided r is less than 1.

We also know that rho being lambda by mu should be less than 1 for an infinite population model. When lambda by mu or rho is less than 1, we can apply the infinite geometric series summation formula to get p_0 into this will have by 1 minus r. So, 1 by 1 minus rho equal to 1 from which p_0 equal to 1 minus rho. So, p_0 equal to 1 minus rho is an important equation for an M M 1 queuing model or M M 1 queuing system. Once we know that p_0 equal to 1 minus rho, now p_1 equal to rho p_0 which is rho into 1 minus rho. A general p_n is rho power n p_0 which is rho power n into 1 minus p. We know rho as lambda by mu. Therefore we can calculate p_0, p_1, p_2, any p_n. We started off by saying that the inputs are lambda mu and C.

In this case, because it is a single server model, it is lambda mu and 1 or essentially only rho, which represents lambda by mu. It is not even necessary to have the individual values of lambda by mu or lambda and mu, if we know rho which is the ratio of the arrival rate to service rate, which is less than 1, then we can find out the expression for p_0 p_1 up to any p_n ,but it does not

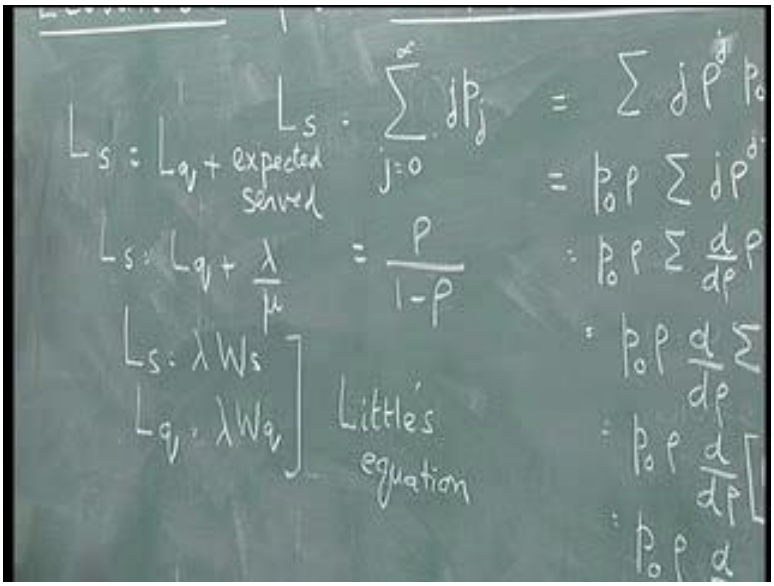
end there because we said earlier, that we are interested also not only in p_0, p_1, p_2 up to p_n, we are also interested in the expressions for W_s, W_q, L_s and L_q.

(Refer Slide Time: 53:02)



We need to derive expressions for W_s and W_q, L_s and L_q in terms of p_0, p_1, etc, which means in terms of rho, we will do that next.

(Refer Slide Time: 53:47)



L_s is the expected number of people in the system and L_s is given by sigma J equal to 0 to infinity jp_j, L_s, is like an expected value. It is j, the number of people multiplied by the steady state probability that so many people are there in the system. This is like saying 0 into p_0 plus 1 into p_1 plus 2 into p_2 etc n into p_n. So this will give us sigma j p_j is rho power n p_0 or rho power j p_0. We take one p_0 outside and a rho outside. This is p_0 into rho into sigma j rho to the power j minus 1. This is p_0 rho into sigma d by d rho of rho power j and then we follow the usual idea that when we have the summation and the differentiation, we can switch this without losing anything in the expression. We get p_0 into rho into d by d rho of sigma rho power j. This will give us p_0 into rho into d by d rho of sigma rho power j, this is 1 plus rho plus rho square etc., up to infinity. This is again an infinite geometric series, with rho which is lambda by mu rho less than 1.

This will be p_0 rho into d by d rho of 1 by 1 minus rho. This will be p_0 rho into 1 by 1 minus rho will give minus 1 by 1 minus rho the whole square into another minus 1. We get 1 by 1 minus rho the whole square. Here again, p_0 is 1 minus rho. This is 1 minus rho into rho by 1 minus rho the whole square. This is rho by 1 minus rho. We get an expression for L_s as L_s equal to rho by 1 minus rho. So, expected number of people in the system is rho by 1 minus rho. We need to write down the expressions for the other ones. Expected number of people in the system is equal to expected number of people in the queue plus expected number of people who are being served plus expected number of people who are being served.

We get L_s equal to L_q plus lambda by mu, lambda by mu or rho, represents the expected number of people who are being served. From this we can get the expression for L_q. Once we know L_s, L_s and L_q are related as L_s equal to lambda W_s and L_q equal to lambda W_q. This is the relationship between L_s and L_q and W_s and W_q. These two are the well known Little's equation, which capture the relationship between the length of the any system and the time associated with this system. With this we can find out W_s, W_q, L_s and L_q. We will work out a numerical example and continue our discussion in the next lecture.