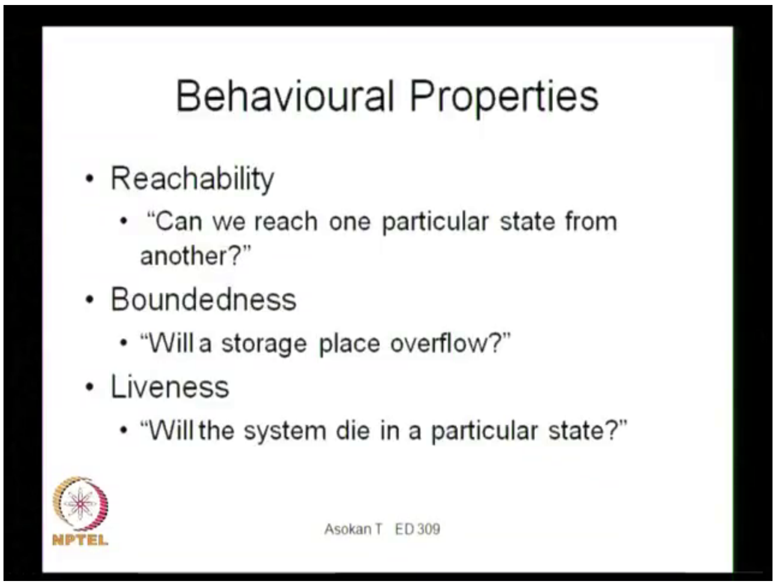
Principles of Engineering System Design
Dr. T.Asokan
Department of Engineering Design
Indian Institute of Technology, Madras

Lecture - 26
Graphical Modelling Techniques

Dear friends welcome back once again to the lecture on system engineering. The last class we discussed about the modelling techniques especially on the behavior modelling how do I see the dynamics of the system or the behavioral dynamics of the system and how do we model it using graphical modelling techniques. We discussed about the petri net, which is one of the important tools for modelling the dynamics of the system and we saw the basic elements of a place transition and the arc to represent the dynamics and we saw one simple example also.

I will take you through few more lectures were actually will try to see some complexities concurrencies in the system and then activation or some control elements control signals to start their process or the start their transition. So, we will go through some of the examples and then show you how the petri nets can be used for complex systems and modelling the dynamics of complex engineering systems.

(Refer Slide Time: 01:14)



Behavioural Properties

- Reachability
  - "Can we reach one particular state from another?"
- Boundedness
  - "Will a storage place overflow?"
- Liveness
  - "Will the system die in a particular state?"

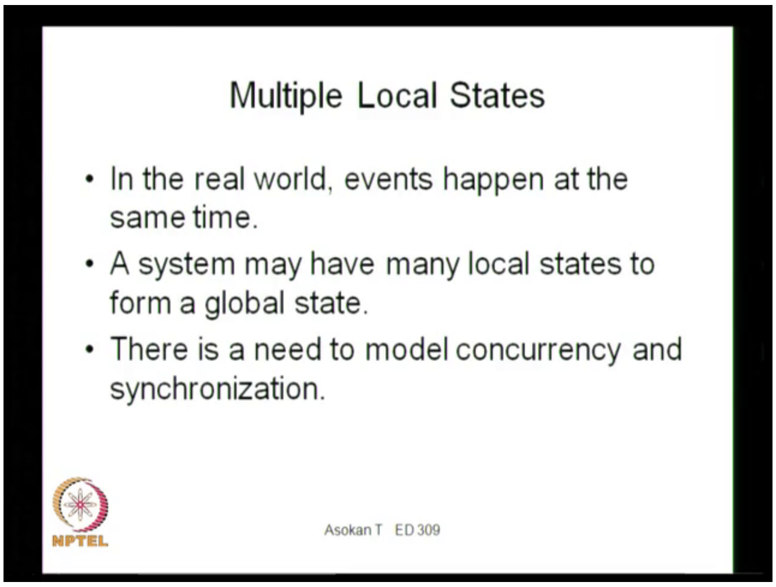
Asokan T ED 309

Let me take a simple example. So, before going to that let me just explained you this point once again because in petri net we used the petri nets to understand the

Reachability of a system whether we can reach a particular state from a an initial state. So, what are the transitions needed to reach that state. So, that can be analyzed similarly the Boundedness of a particular state. So, to what extent we can continue to load that particular state or when we do should we actually stop that transition to that particular state because there may be an overflow of data or information.

So, how do we actually check that boundedness of a system with this also can be tested using petri net then we have this Liveness of a system, whether the system transition will lead to a death of a system in terms of whether system will stop giving an output or their will not be an output, when you reach a particular state these are all can be a model using petri net.

(Refer Slide Time: 02:08)



Multiple Local States

- In the real world, events happen at the same time.
- A system may have many local states to form a global state.
- There is a need to model concurrency and synchronization.

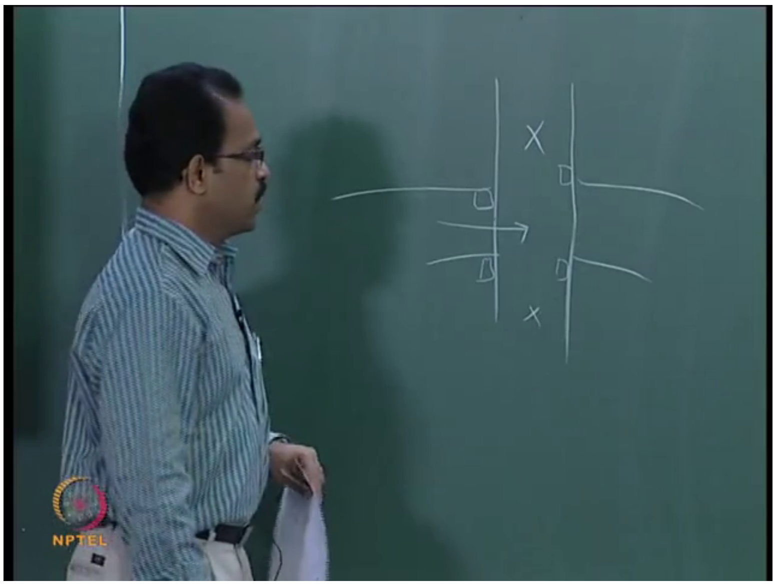
NPTEL

Asokan T ED 309

Apart from this we can actually see that there are multiple local states which actually lead to a global state.

So, that also can be understood using petri net we can actually model those behaviors using this kind of petri nets the states and transitions. So, you need to explain these situations I will take a few examples and then show you how petri net can be used for the behavior modelling of a systems. I will take a very familiar example which is a traffic signal basically we all know that traffic signals are to be synchronized with other signals suppose you have a junction.

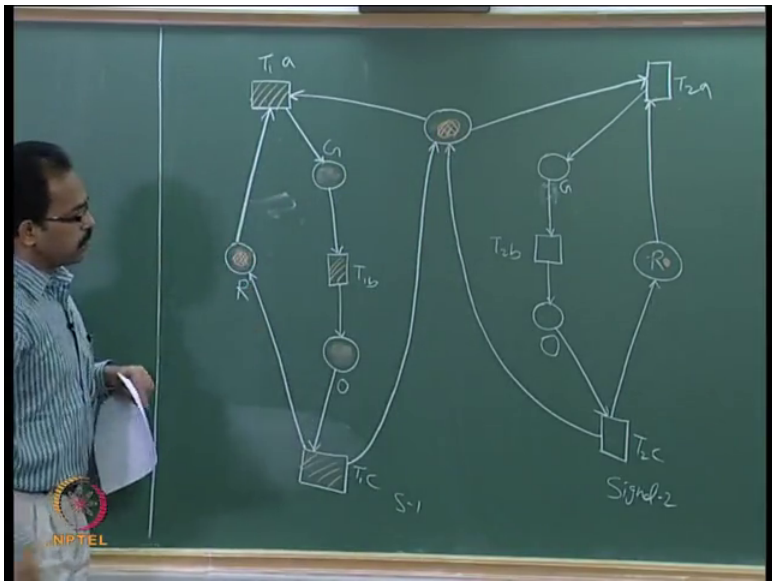
(Refer Slide Time: 02:47)



And then you want to show the signals on both sides so you will be having a signal this point and signal here of course, here also. So, this signals need to be a synchronized because then is a green signal here, then this always should be a red signal on both sides and when this should change this red should not changed to green unless this green has changed to red. So, this kind of concurrencies and synchronization need to be brought into the system and we need to test that synchronization and then the concurrency using transition.

So, what kind of transition and what kind of control need to be put in place to ensure that there will not be any problem with the signals or there will not be any issues of synchronization. So, this can be modelled using petri net by showing different states and places as well as the transitions. So, we can actually represent the signals.

(Refer Slide Time: 03:45)



So, you can show that one side signal that is you have a red signal and you have a green signal and an orange signal. So, this is the or 3 signals. So, red, green and orange. So, these are the places. So, we put sorry these are orange I will tried it outside because we need to show the tokens inside. So, this is red this is green and this is a signal one and similarly we have a signal 2 also, we have red signal then we have the green signal and then we have this orange signal. So, red, green and orange and we need to have the transitions.

Because when one to change from red to green you need a transition so we will represent that transition here then green to orange one transition. So, I will let me represent it here because it is easy to represent the transitions. So, this is the T 1 a which actually shows the transition from red to green. So, you have a transition from red to green. So, it will be like this. So, this is the flow of data. So, this is transition a T 1 a over the first signal transition one for this and then you have a transition here.

T one b which actually shows the transition from green to orange and then you have one more transition from orange to red. So, let me show this transition here T 1 C, which actually shows this transition from here. So, this actually represents the a transition and the places of a signal you have a red green and orange and you have this transition one a, one b, and one c, which actually represent the transition from one state to other state and we can have the tokens to represent the present state of the signal.

In the same way you can have this transition here also. So, here you have the T 2 a and you have this T 2 T 2 b which is which in T 2 a green 2 this is T 2 b and then you have this T 2 c so this again the transition. So, you have this red to green and then green to orange, then the transition from orange to red. So, these are 2 independent signals. So, this is signal 2 this is signal 1. What we need to do is T 2 show the synchronization of the signals. So, we cannot have a 2 green signals at the same junction. So, we need to may ensure that when the signal is red here then only that should be green otherwise if this green that should never become a red signal.

So, how do we actually represent that situation using the tokens or the place values? So, here this is the transition which actually moves from red to green. So, red to green cannot be done unless we ensure that this is red. So, we need to have some kind of another input over here or another place which actually connects to this and when we have tokens on both these red and the other one then only this can actually change to green. So, we will provide a another place here is actually connects to this T 1 a. So, you can see that this T 1 a the transition is connected to 2 places and both are actually coming to this T 1 a; that means, there should be token available at this point and this at this place and this place in order to activate this transition.

So, that is the first condition the same way for this also for T 2 a which actually transfers from red to green that also should have to because when this is green this should not change to green. So, this some red to green it is not happen. So, therefore, we need to ensure that this transition has to check what is the status of this particular signal? So, we have another connection over here. So, this transition has got 2 places connected and they both should have the tokens in order to make a transition from red to green. And then we need to ensure that once this has become red then this has got a value because when this is red then only this can change to green.

So, we will have a place value connected to T 2 c also similarly T 1 c which actually gives a token back to the this particular place, that is whenever this red as this oranges going to red when there will be a token available here. Similarly when just become oranges become red that token will be available here; that means, the particular when the signal is red then only there will be a token over here; that means, can have a change from red to green the other one.

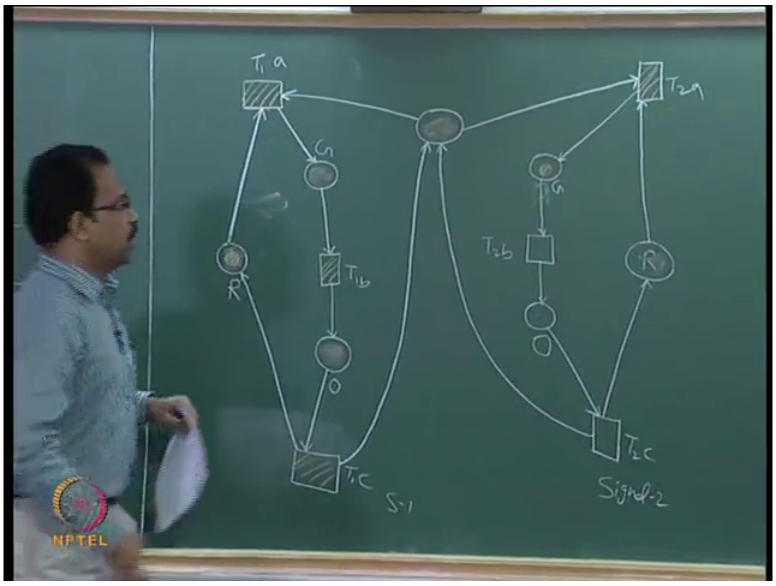
So, we will represent this here suppose you have an initial token available here suppose this is the initial situation you have a token available here and the token available here; that means, both are red when you have a token available here; that means, both are red signal now. So, you have a red signal here and the red signal here. So, you have a token over here also now there is a option that you can actually have a T 1 a transition or T 2 a transition. So, one of the signals can actually go to green not both you have this T 1 a activated suppose you activate this T 1 a the transition is activated then immediately this token will be given to green and this token will be released from here.

Because this only one output. So, this token will be released from here to this point. Now this is green and there is no token over here therefore, T 2 a cannot change. So, T 2 a that please transition cannot happen; that means, this red cannot become green unless they token comes back to this position. So, now, this green is there. So, the present position is a one signal is red the other one is green. Now we have a transition here suppose we go for the transition here this token will be move to here; that means, orange will be activated. So, this place has got a activation now. So, the signal is orange now.

Still it cannot change to write green because there is no token available here and there will be a token available only when you have the next transition. So, when we have this transition suppose we activate this transition now, then this token will be given here move to here as well as to this point this place because now you have this transition is connected to an output to here. So, whenever there is an input coming to this both this places will be getting a token.

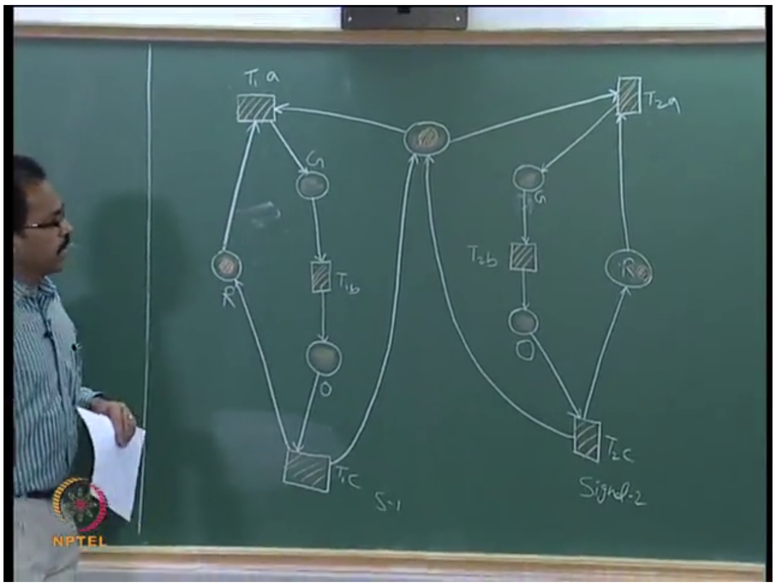
Now, this has got a token. So, the token is there and this signal is red. So, at this point you can see that there is a token available here and token available here also here also there is a token available therefore, we can actually activate this transition. So, this signal is red and therefore, we can activate this transition.

(Refer Slide Time: 11:42)



Again this token will be removed and this token will be passed to this one. So, the token from red will be passed to green you can see that the now one signal is red the other signal is green. So, you can the same way you will do the transition.

(Refer Slide Time: 11:54)



So, this transition the token will be passed to this and when this transition is made the activate this transition this token will be moved from here it will go to here as well as to here. So, when we activate this transition you will be getting the token at here and here now the both the signals are red and therefore, we can have the next transition from here.

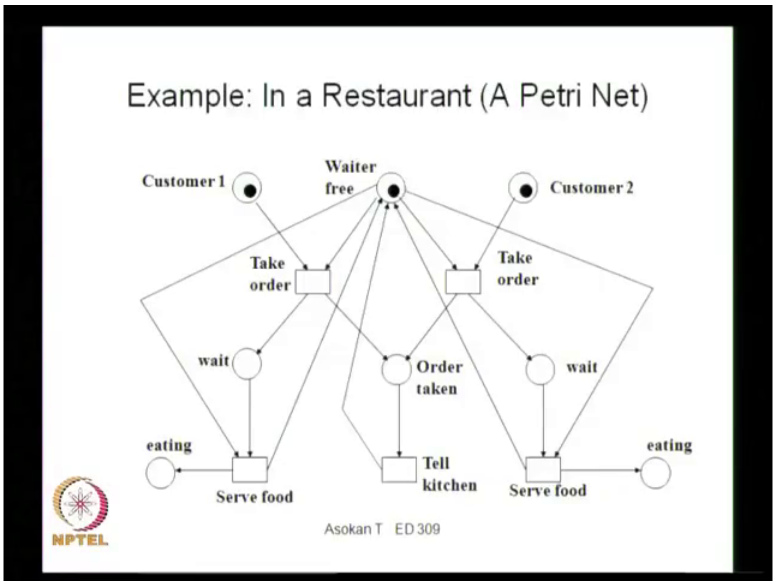
So, unless this token comes back from here this cannot activate and this red signal cannot change to green.

So, this actually shows that there is a synchronization between these 2 signals in order to ensure that there is no issues with the synchronization basically when one signal is red then only the other will change to green. So, this is one way of representing this synchronization using petri net. So, this is a traffic signal where we have different signal of course, you can actually have multiple signals if you have more signals here I have shown only straight direction motions. So, if you have a turning to right or turning to left and then taking you turn all the signals are there.

We can actually incorporate all those scenarios and then see what kind of synchronization we need to ensure. So, depending on that we need to include additional places. So, that the tokens can be distributed in such a way that there is proper synchronization and there is no issue of overlapping of signals or there is no other problem associated with the concurrency and synchronization. So, this is one way of modelling the dynamics of this particular behavioural dynamics of the signal using petri nets.

The same way you can actually represent many other situations also.

(Refer Slide Time: 13:43)



Example: In a Restaurant (A Petri Net)

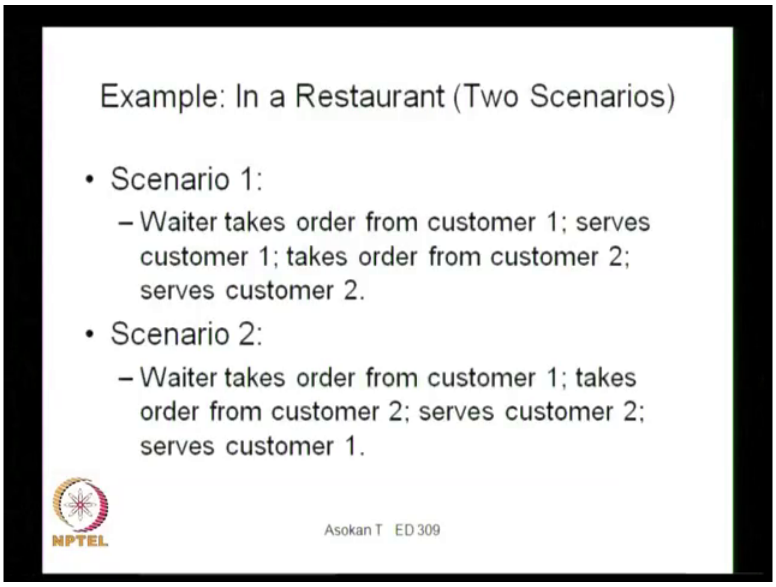
Asokan T ED 309

You can show you another situation where we have a restaurant and customers and basically people who actually serve, how do we actually synchronize the behaviour of this customers and servants and how do we actually optimize the use of these people by having proper synchronization of their activities as you can see in the figure over here. So, we have shown that there is a one waiter and the 2 customers waiting for a customer and. So, that is a Waiter free.

So, now if you want to represent this situation we can have different situation that the waiter can actually take the order from one customer and then you can give the order to the kitchen and then wait for the order to come and then he can serve the customer. And then order from the next customer or you can take order from customer one and place the order to the kitchen and then he can take the order from customer 2 and then place the order and whenever the food is ready he can serve customer one and customer 2.

So, how do we actually represent this kind of scenarios using petri net? So, as I mentioned petri net can actually shows all the dynamic behavioural dynamics and can show the concurrency and the synchronization the same way we can use the situation also we can represent using petri net.

(Refer Slide Time: 14:58)



Example: In a Restaurant (Two Scenarios)

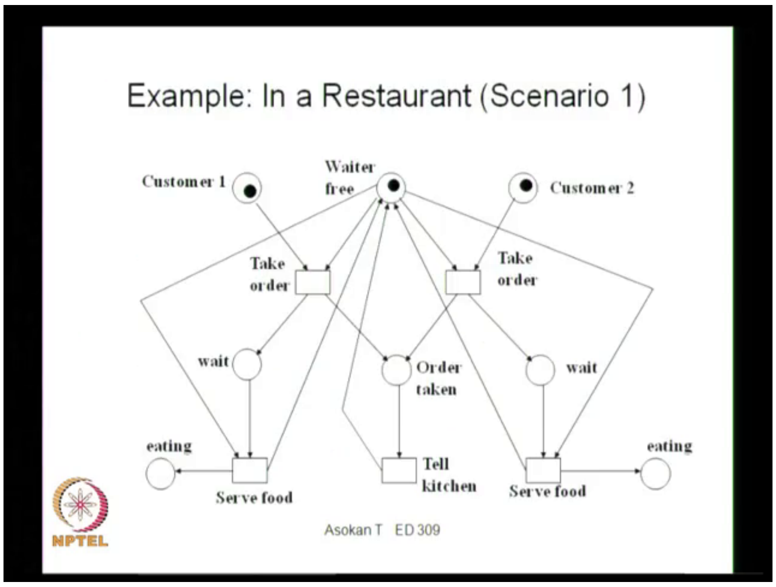
- Scenario 1:
  - Waiter takes order from customer 1; serves customer 1; takes order from customer 2; serves customer 2.
- Scenario 2:
  - Waiter takes order from customer 1; takes order from customer 2; serves customer 2; serves customer 1.

Asokan T ED 309

As you can see here so we have 2 scenarios we are representing the scenario one is that the waiter takes order from customer one and serves customer 1, takes order from customer 2, serves customer 2, customer 2 will be waiting till the customer one is served.

The second scenario is the waiter takes order from customer 1, takes order from customer 2, serves customer 2 and then serve customer 1. So, this is the another scenario of course, you can have many other scenarios of various situation and then try to see how do we actually get those scenarios represented using petri nets just for example, I am showing 2 scenarios.

(Refer Slide Time: 15:34)



Example: In a Restaurant (Scenario 1)

So, this is the scenario 1. So, initially both the customers are there you can see the tokens at these places. So, you have this token customer one and customer 2 and then there is a waiter who is free here.

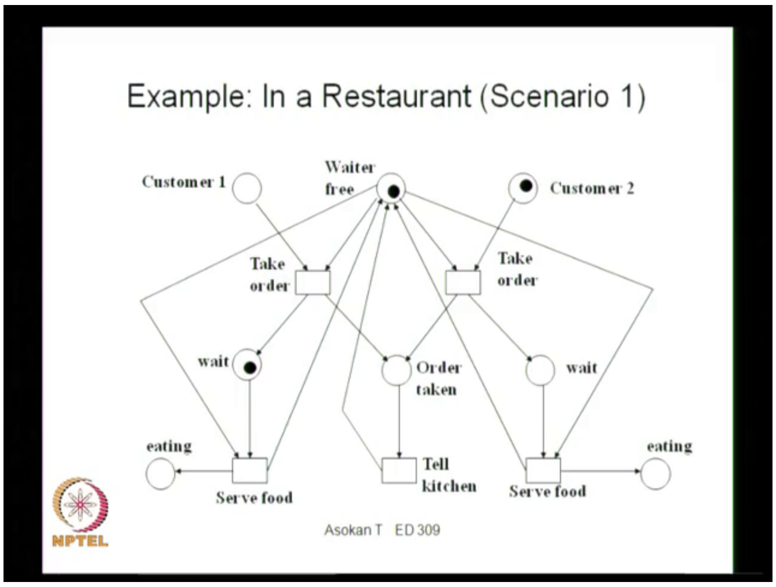
This transition that is taking order is shown by this transition and then this taking order from customer to his first shown by this transition. So, you can see here this transition has is connected to 2 places. So, customer one and waiter free. So, this transition can take place only if this there are tokens in these 2 places that is customer one and waiter there are tokens available, then only it can be transition can take place similarly for the customer 2 also you need to have this transitions and when this transition takes place there are 2 tokens coming over here. So, when this transition take place as we can see it is connected to 2 places that is wait as well as order taken.

So, this 2 places you will be having tokens and once you connect this one you can see the serve food is connected 2 tokens again waiter free and food is available. So, these 2 are needed for this transition or the customer is waiting and the waiter is free then only the transition can take place. So, this the relationships actually what particular relationship is needed for the transition can be represented using this arrows and the tokens.

Now, let us see how this is taking place. So, you can see that the order is taken. So, therefore, the transition is taking place customer is waiting here and the order is taken. So, we have these 2 places with tokens now, because this transition take place you have this customer waiting here after the order is given and the order is taken from the customer that is completed. And now once the order is taken that has to be informed to the kitchen or it has to be placed to the kitchen. So, that is this transition. So, the waiter you need to give this order to the kitchen.

So, he will be giving this order to the kitchen that is this transition. So, we can see here. So, once you placed the order the actually the token goes back to the waiter.

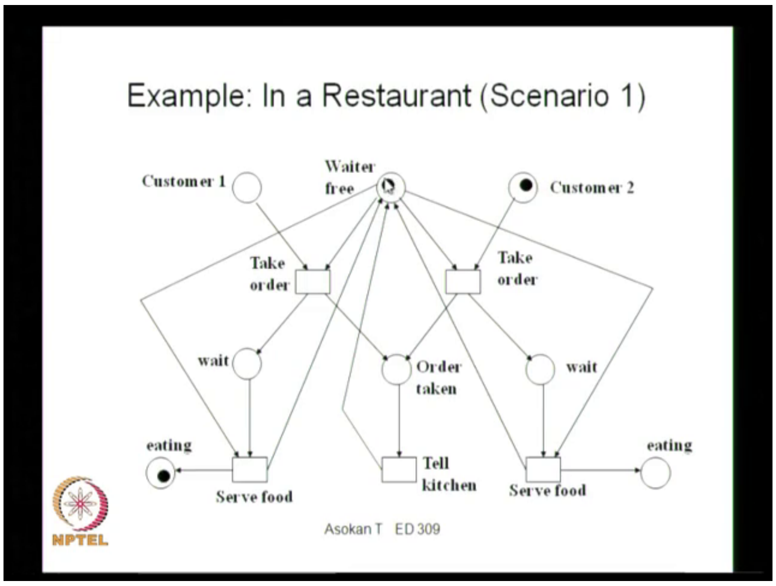
(Refer Slide Time: 17:37)


Example: In a Restaurant (Scenario 1)

So, once he ordered the item to the kitchen then the waiter is free here. So, here now the waiter is free this customer is free and this customer is waiting. So, the waiter has got an option either to take the order from customer to or to wait for the item to be ready. So, that he can serve food. So, you can see that to serve food for this transition you can see this is the waiter should be free and then there will be food should be ready.

Now, waiter has option to take the customer 2 order or wait for the food to be served. So, here in the first case it is actually serving food.

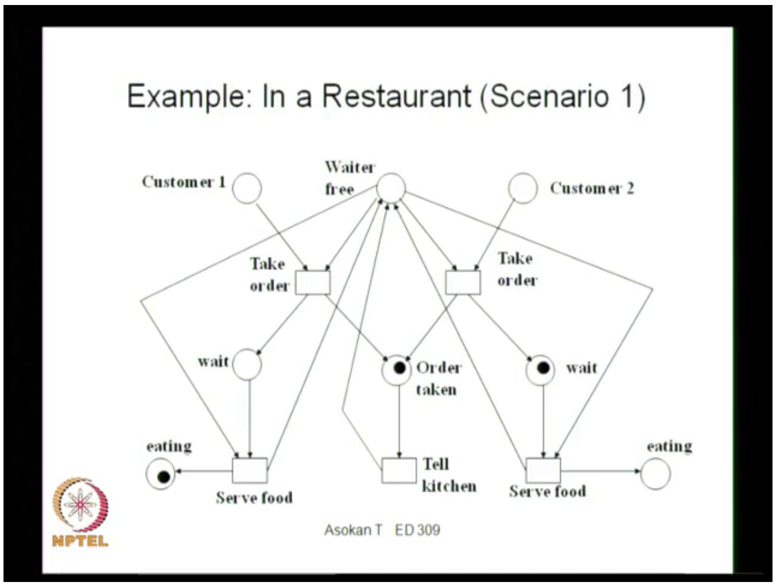
(Refer Slide Time: 18:19)



Example: In a Restaurant (Scenario 1)

And the customer is eating for the place is taken the token is given there and therefore, the waiter is again free after the serving of food. So, the customer is free now he can take the order second order. So, he is taking the second order customer 2.

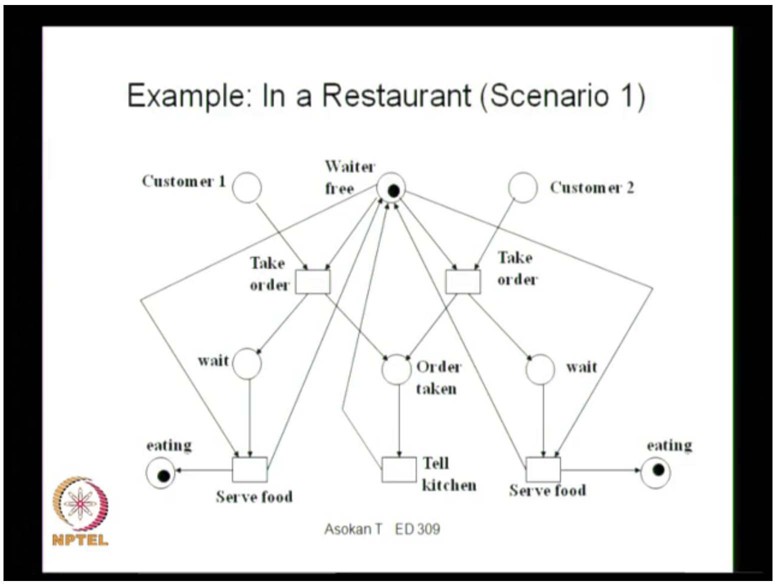
(Refer Slide Time: 18:33)



Example: In a Restaurant (Scenario 1)

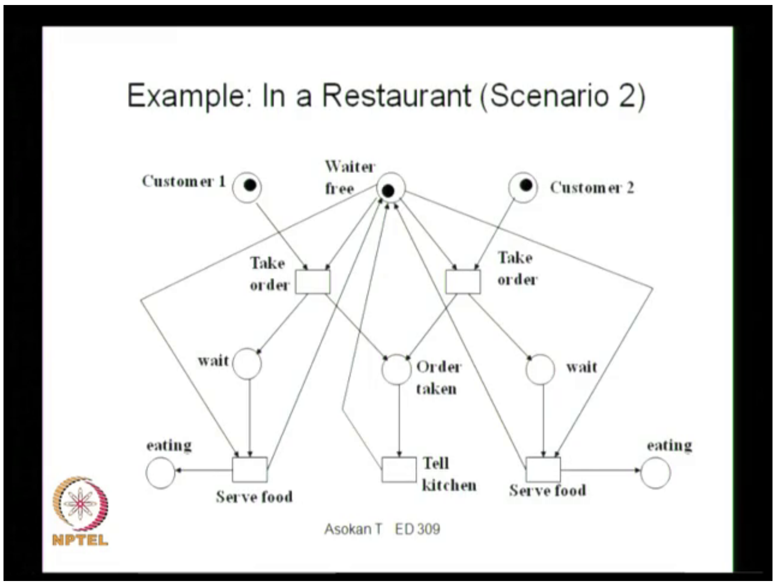
And you can see the tokens are moving here order is taken and the customer is waiting order is taken and then tell the kitchen about the order. So, this is activated. So, he will be getting the transition and then you will get the serve food and the customer is eating and the waiter is free now.

(Refer Slide Time: 18:47)


Example: In a Restaurant (Scenario 1)

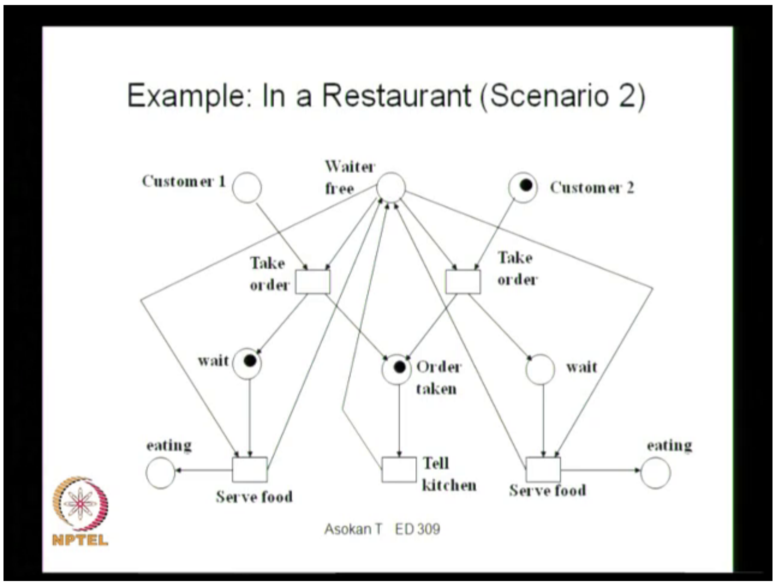
So, both the customer is are having food and the waiter is free. So, this is one scenario where we can actually represent the dynamics or the behavioral dynamics what is happening inside the in that particular scenario. So, we can have any number of customers and any number of waiters. So, we can actually show the complexities also with some waited arcs that I will explain later, but in this case we can see that is a scenario can be represented using petri nets and the place values and the tokens.

(Refer Slide Time: 19:23)


Example: In a Restaurant (Scenario 2)

So, in the scenario 2 it is almost similar except that the situation changes. So, here the waiter and customers are free and then actually it takes order from customer 1. So, customer is waiting and the order is taken and that is inform to the kitchen and the waiter is free.

(Refer Slide Time: 19:38)

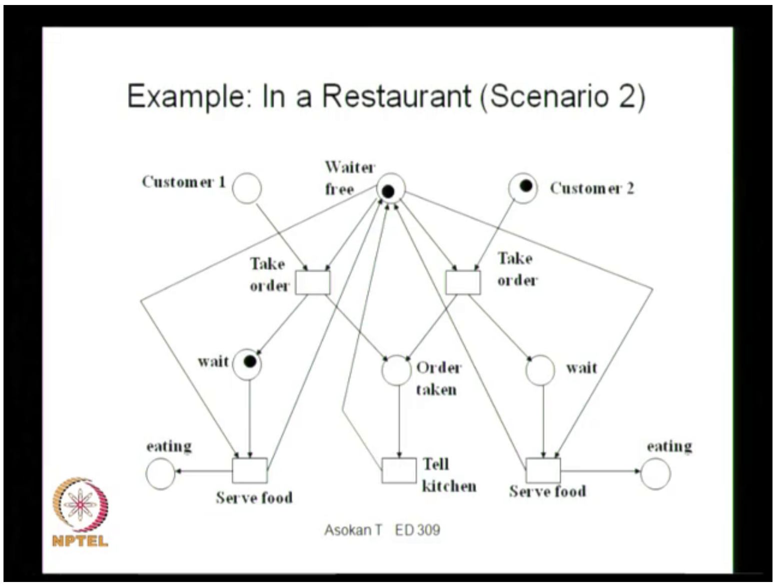


Example: In a Restaurant (Scenario 2)

Asokan T ED 309

And one customer is waiting for food the other customer is waiting for to place the order.

(Refer Slide Time: 19:44)

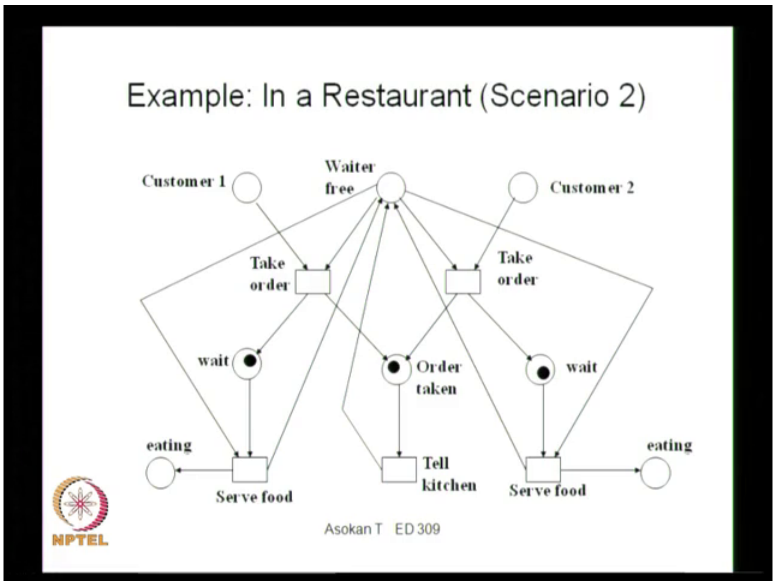


Example: In a Restaurant (Scenario 2)

Asokan T ED 309

Now, you actually take the second situation the waiter is taking order from customer 2.

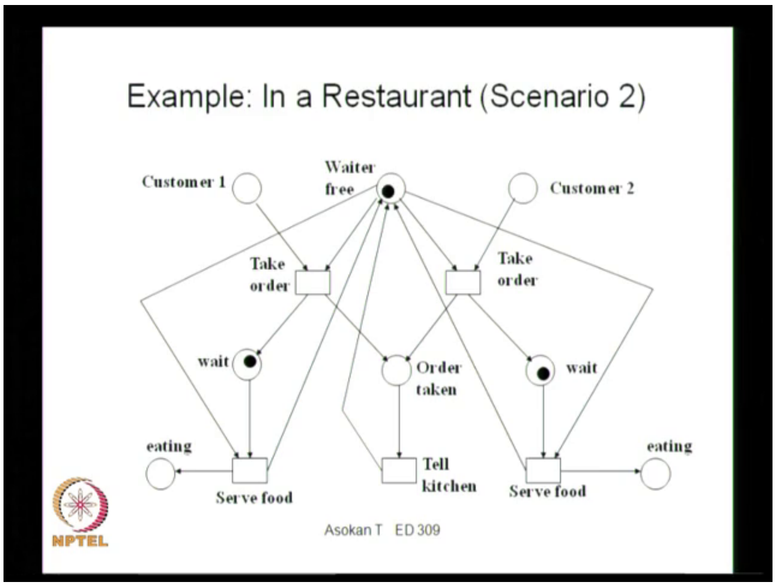
(Refer Slide Time: 19:59)



Example: In a Restaurant (Scenario 2)

And then places the order to the kitchen and then both the customers are waiting and the waiter is free. So, that he can actually serve food.

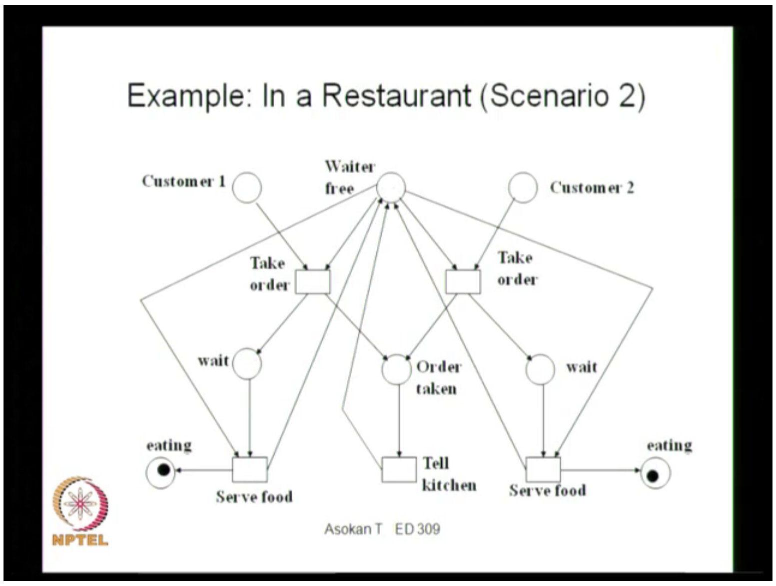
(Refer Slide Time: 20:05)



Example: In a Restaurant (Scenario 2)

Now, the waiter is free. So, he will serve the food to one customer. So, serve food the customer 2 and then serve food to customer 1.

(Refer Slide Time: 20:11)

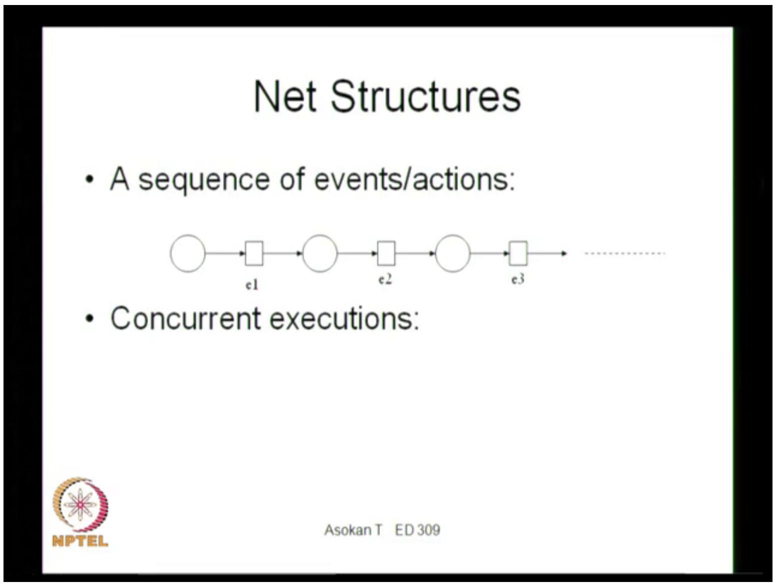


Example: In a Restaurant (Scenario 2)

Asokan T ED 309

So, this is way how you actually represent the scenario. So, any scenario can be shown like this way. So, we can actually show the what kind of concurrency is needed and what kind of synchronizations are needed, there can actually be represented using this kind of ah petri net. So, that was one example and there are few more example which actually shows the complexity of the petri net.

(Refer Slide Time: 20:45)



Net Structures

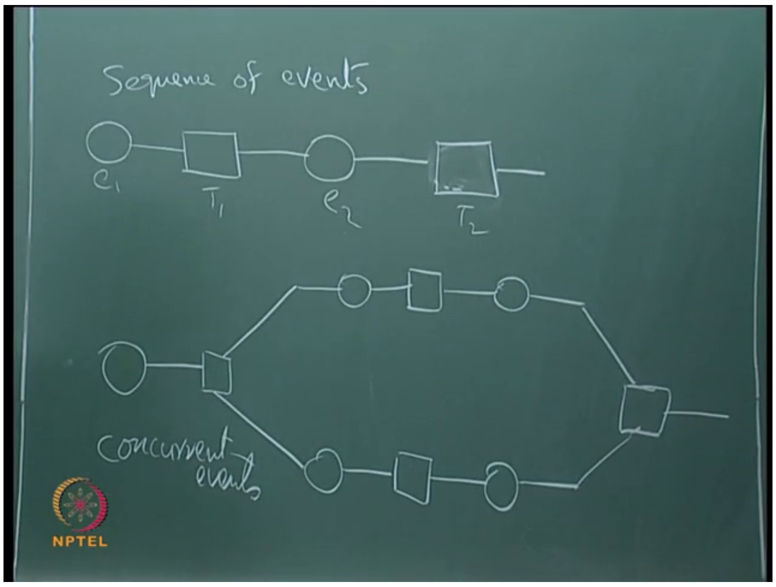
- A sequence of events/actions:

- Concurrent executions:

Asokan T ED 309

Here you can see as I mention the petri net can be used for various scenarios. So, you can actually represent a sequence of events as you can see here you can have events e one e 2 e 3 represented and the places and transitions as we have the places and transitions you can actually represent it using in the serial way.

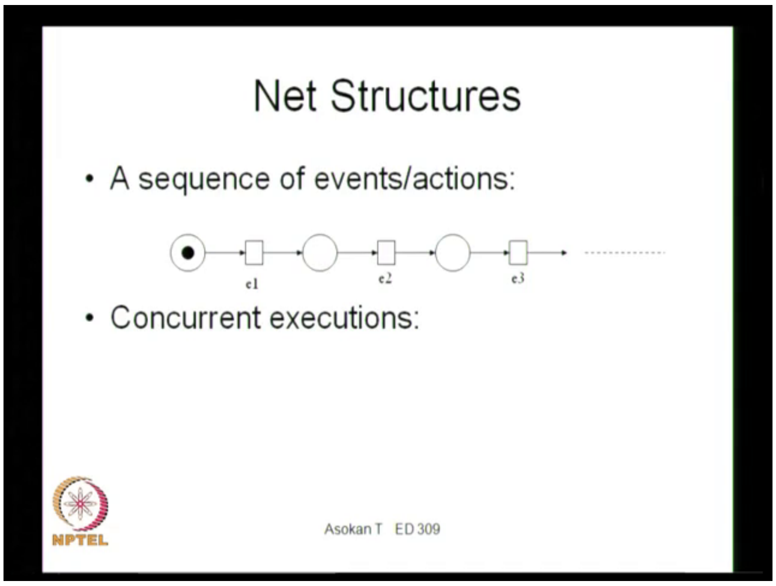
(Refer Slide Time: 21:05)



So, we have these places or the events. So, you can have any event one or the places then you have a transition then you have an event so this kind of serial activations sorry.

So, this is e one e 2 and transition one transition 2. So, this is basically a serial sequence of events. So, that can actually be represented using this kind of petri nets or you can actually go for the concurrent executions. So, this actually shows the executions how the place values are changing.

(Refer Slide Time: 21:36)



Net Structures

- A sequence of events/actions:

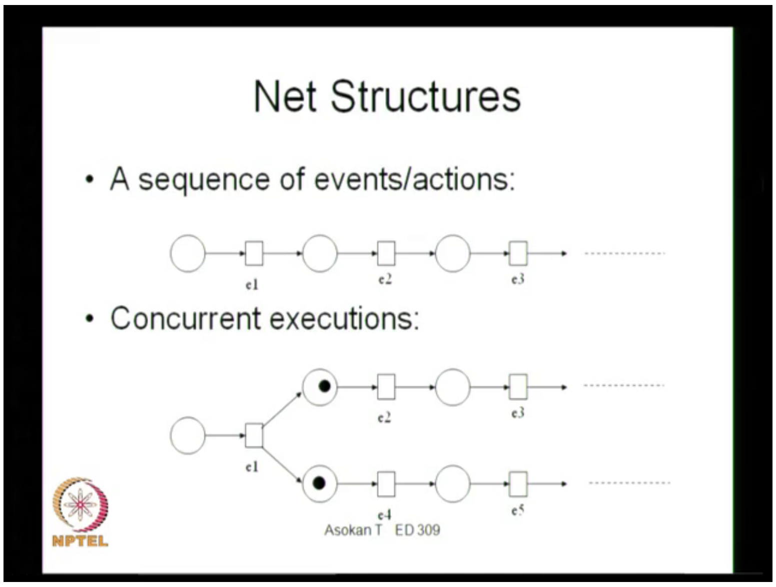
- Concurrent executions:

Asokan T ED 309

So, if you go back you can actually see this. So, when you have the transition event is happening the token is transferred to the next one then you have this token next one Trans happening then again the token is represent transferred using every transition.

So, this is one way of showing the sequence of events and then the concurrent executions.

(Refer Slide Time: 22:05)



Net Structures

- A sequence of events/actions:

e1
e2
e3

- Concurrent executions:

e2
e3
e1
e4
e5

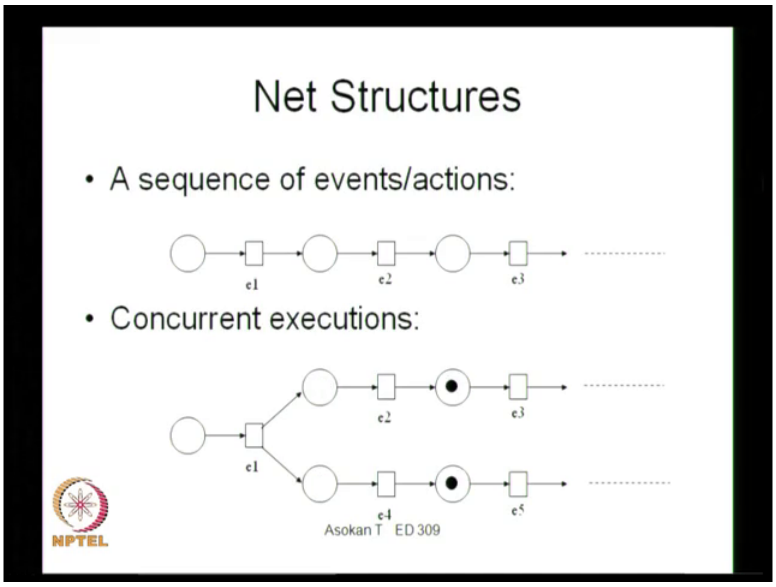
Asokan T ED 309

You can see in the picture here the concurrent executions also can take place by concurrency mean that you have this transition. So, we have actually this concurrent activities you can have this kind of transitions and events. So, the place transitions can actually happen like this. So, this actually shows the concurrency of the events.

So, you have one event and then you have many events going the same time and of course, you can actually if the come to one state at a later state then actually you can have the same output from here. So, these events will happen concurrently and once these 2 are ready then only this transition will take place and to finally, till in an output. So, that is the sequence or concurrent execution. So, this is the serial one for sequence of events and this is the concurrent event.

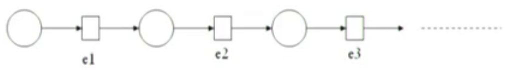
The concurrency of events or concurrency of executions same way this can actually be again found through the animation you can see the e one happening. So, the tokens are given to the next 2 parallel execution change there is a concurrent events and then e 2 and e 4 take place.

(Refer Slide Time: 23:31)

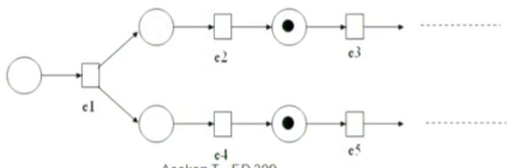


Net Structures

- A sequence of events/actions:



- Concurrent executions:

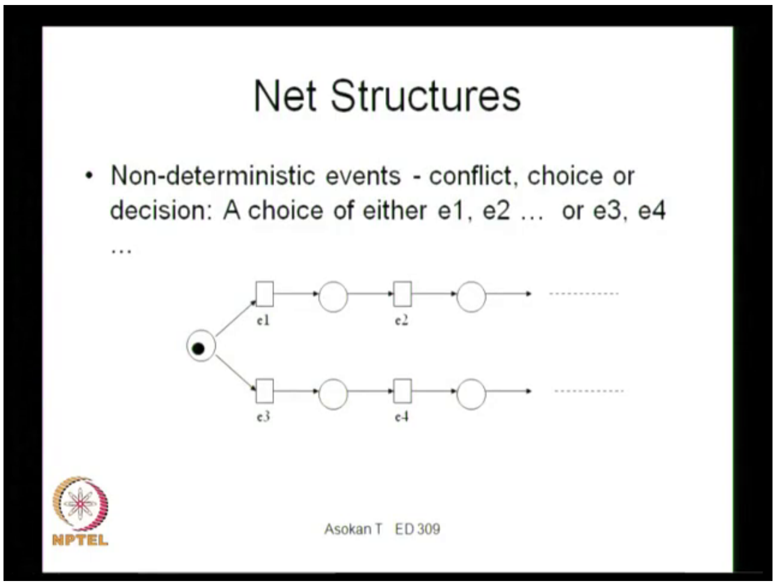


Asokan T ED 309

Then pass the token then again e 3 happen e 3 and e 5 pass the token. So, like this you can have concurrency of events.

These are the sequence of events and the concurrency of events.

(Refer Slide Time: 23:41)



Net Structures

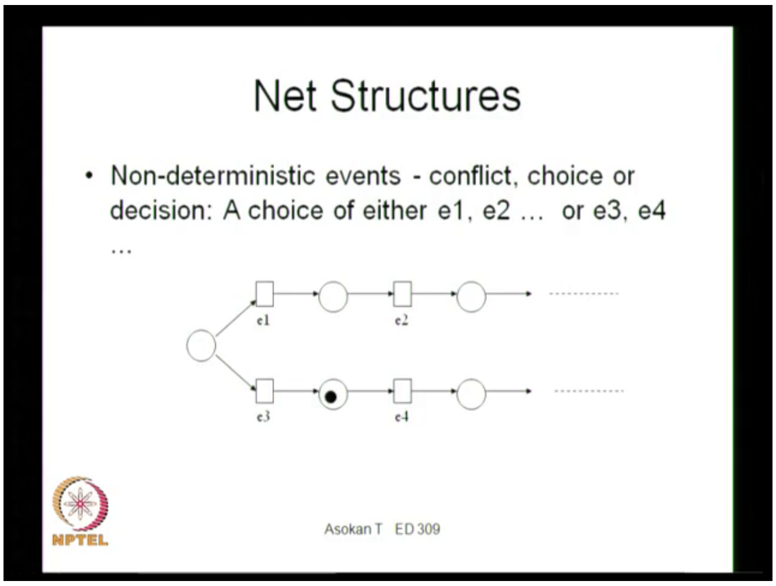
- Non-deterministic events - conflict, choice or decision: A choice of either e1, e2 … or e3, e4 …

Asokan T ED 309

And apart from this we can actually have a nondeterministic events like conflict choice or decision. So, this also can be represented using this kind of petri nets that this you can have a choice of either e one or e 2 or e 3 or e 4 like that. So, you can actually choose an e one it is not particular conditions you can actually make a choice also that also can be represented using petri nets.

So, here you can see that you can have a for e one or e 3 e one e 2 etcetera or you can go for e 3 e 4 and that level also. So, this also can be represented. So, in this case if you pass the token to e one or e 3 can be executed. So, in this situation what is shown here? So, only one of this transition can take place then this will happen e one e 2 etcetera will happen or if this is started then e 3 e 4 etcetera will start. So, not the concurrently this will happen it is only one group will be executing. So, as you can see here.

(Refer Slide Time: 24:37)



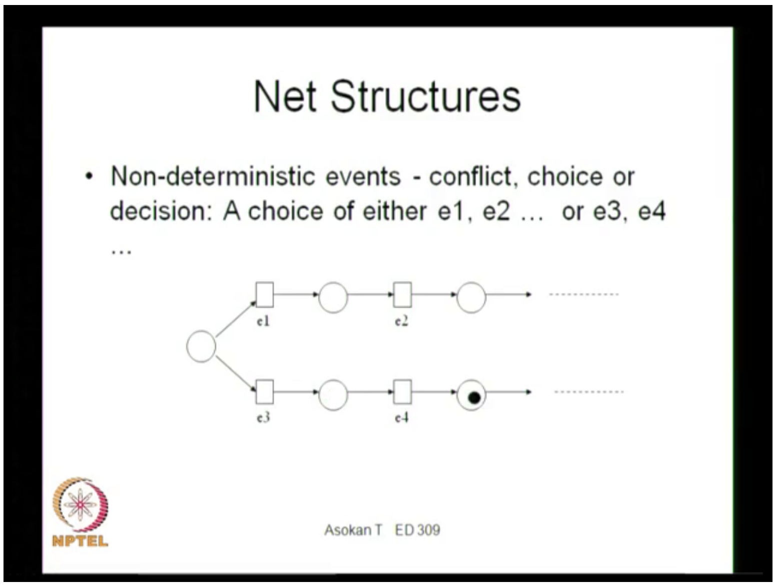
Net Structures

- Non-deterministic events - conflict, choice or decision: A choice of either e1, e2 … or e3, e4 …

Asokan T ED 309

So, if e threes event is happening then e one cannot happen, because there is no token available here. So, e one cannot happen unless the token comes back to this position. So, e 3 will happen then e 4 will happen and then it will continue like this unless.

(Refer Slide Time: 24:48)



Net Structures

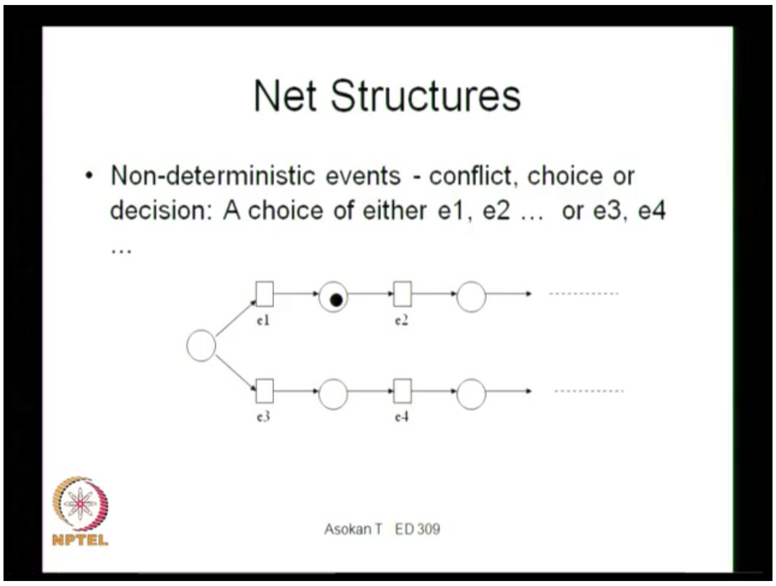
- Non-deterministic events - conflict, choice or decision: A choice of either e1, e2 … or e3, e4 …

Asokan T ED 309

Until when it reaches one point where the token from here is return back to this position e one cannot start. So, that is the choice for a particular situation.

So, this actually shows the second one. So, once e one is activated.

(Refer Slide Time: 25:02)



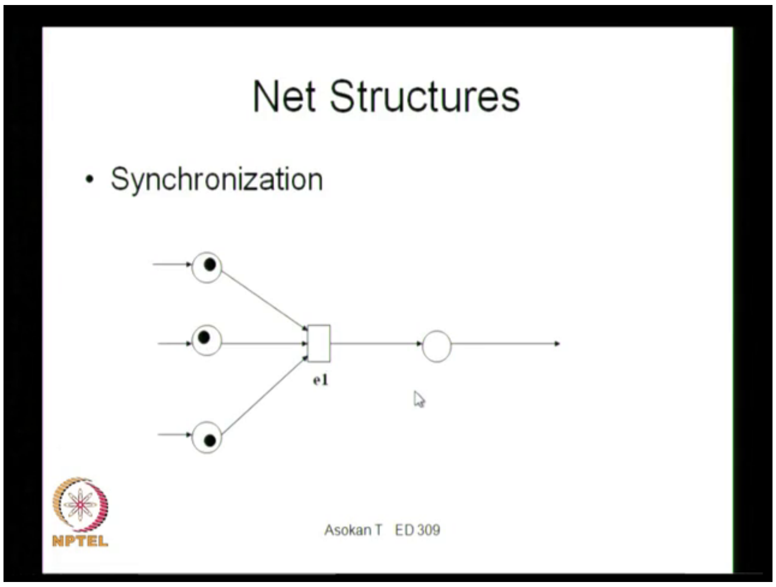
Net Structures

- Non-deterministic events - conflict, choice or decision: A choice of either e1, e2 … or e3, e4 …

Asokan T ED 309

Then e 3 cannot happen because the token is not available therefore, e 3 will not happen then e 2 will happen and then continue. So, that is about the structures were you can have choices and various options to execute.

(Refer Slide Time: 25:15)



Net Structures

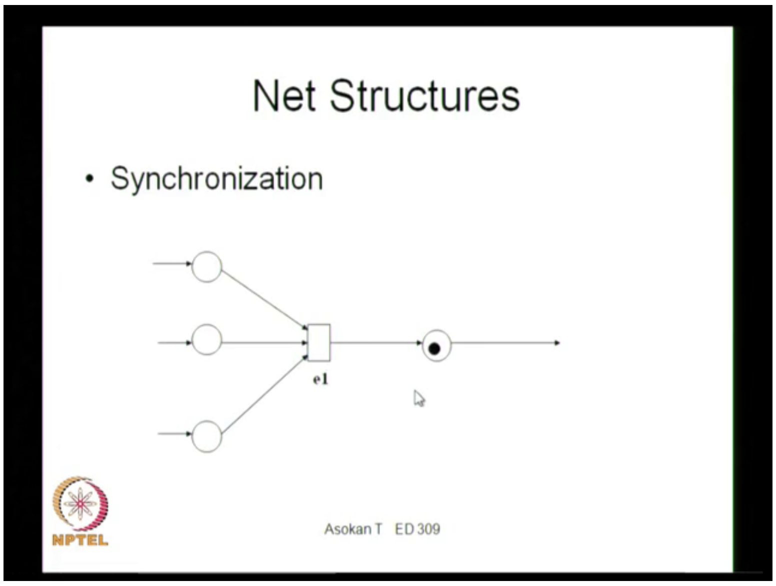
- Synchronization

Asokan T ED 309

Now, there are different other net structures also as you can see here in this case this is a synchronization which I already explained. So, the synchronization will happen only when this e one can happen only and there is a tokens are available in all these cases.

So, if there tokens are not available here e one cannot happened; that means, this e one will be synchronized with the happening of all these 3 incidences all these places should have token in order to activate this. So, that is the 3 are ready then only e one can be executed and it will be going the token will be move to the next one.

(Refer Slide Time: 25:50)

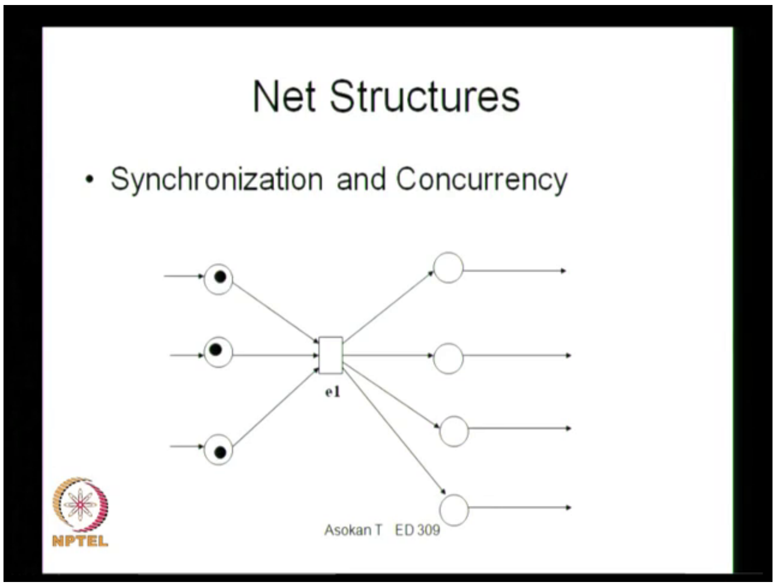


Net Structures

- Synchronization

So, that is the synchronization structure. And the next one is the concurrency so synchronization we saw then asynchronization concurrency also can be introduced through structures. So, this is the synchronization and concurrency structure.

(Refer Slide Time: 25:57)

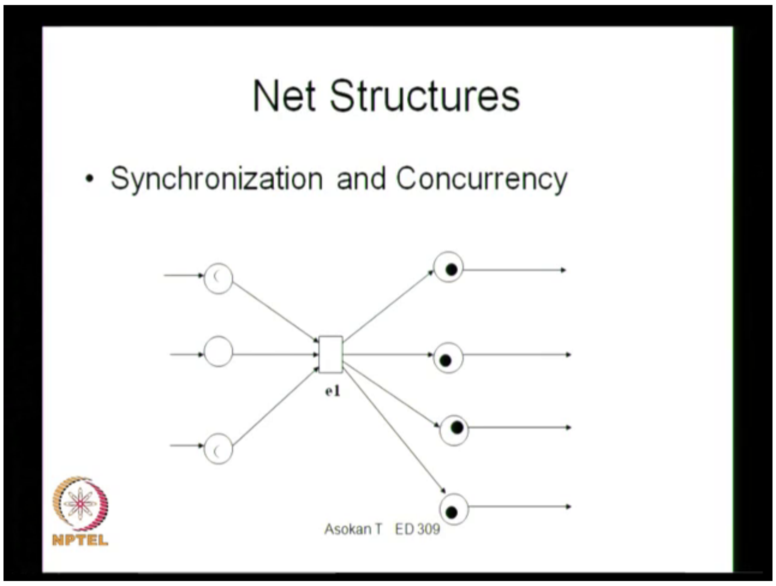


Net Structures

- Synchronization and Concurrency

So, this actually shows the synchronization. So, this e one we will synchronize with the happening of all the 3 and the concurrency when e one is activated all this will concurrently start all this will be having the places. So, that is the synchronization and concurrency of the system again you can see here. So, you have the transition e 1.

(Refer Slide Time: 26:32)



Net Structures

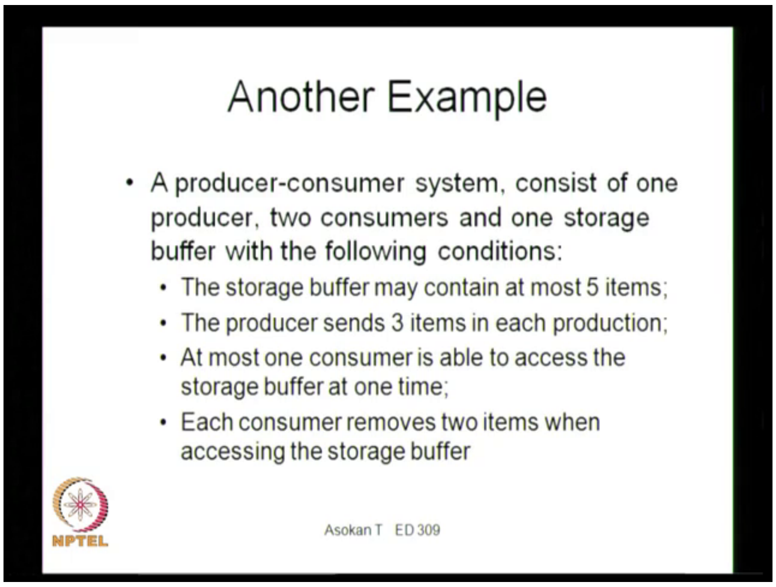
- Synchronization and Concurrency

Asokan T ED 309

This will go to all the places will be having the token value and therefore, you will be getting activation of this and that will be the concurrency of the system.

So, that is how the synchronization concurrency can be represented using petri nets.

(Refer Slide Time: 26:45)



Another Example

- A producer-consumer system, consist of one producer, two consumers and one storage buffer with the following conditions:
  - The storage buffer may contain at most 5 items;
  - The producer sends 3 items in each production;
  - At most one consumer is able to access the storage buffer at one time;
  - Each consumer removes two items when accessing the storage buffer

Asokan T ED 309

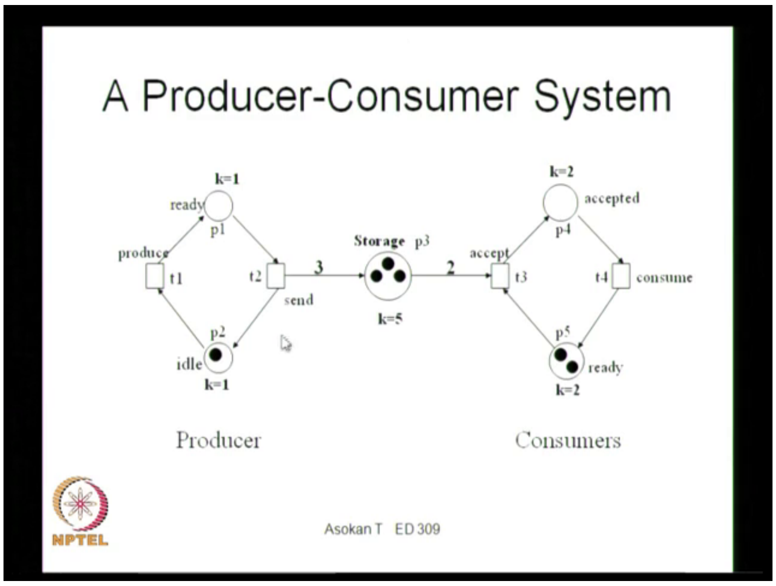
Now, we will take another example where we can show all these concurrency sequence of activation as well as concurrent activation and synchronization. We will take a another example where a producer consumer system which consist of one producer and 2 consumers and one storage buffer with the conditions, this case we have a producer consumer system where we have one producer and 2 consumers and one storage buffer.

So, how do we actually synchronize the activities of the producer the consumer as well as the storage, we need to ensure that the producers enough for the consumer and the if the consumer is not taking it is it going to the storage and a storage is not overflowing. So, how do we actually represent these a concurrency and the synchronization of this events that can be represented using petri net. So, the conditions for this are the storage buffer may contain at most 5 items. So, storage buffer cannot have more than 5 items and the producer sends 3 items in each production.

So, whenever the producer makes one production it will be having 3 items and buffer can have only 5 items stored and the at most one consumer is able to access the storage buffer at one time. So, only one consumer will be able to access the storage buffer at one time and each consumer removes 2 items when accessing the storage buffer. So, whenever a consumer access the storage buffer he can take 2 items, but only one consumer can access it at one time and one producer can produce 3 items at a time and the storage buffer has got a maximum capacity of 5.

So, how do we actually ensures that there is no overflow of the buffer as well the storage as well as the producer is not producing more than what can be stored or consumed and how do we ensure that consumer gets the items whenever it is there or the whenever the customer wants he get some item. So, how do we ensure this condition we can actually represent that kind of a scenario using petri net?

(Refer Slide Time: 28:45)



A Producer-Consumer System

Asokan T ED 309

So, the petri net for the system is shown here. So, here you can look at the petri net. So, this is the producer, this is the storage, and this is the consumer. So, we have consumer place this one and this is the consumer 2 and you have this producer then it is a ready for transferred to the storage.

So, this is the storage with the place value of 5. So, here we can see that the storage can have a maximum 5 items. So, that is why k is given as 5. So, this another kind of petri net where you have the value for the places as well as for the arcs you can see here these are the arcs which actually represents the transition from this state to this state. So, the values given as three; that means, you can send a 3 at any time to the storage.

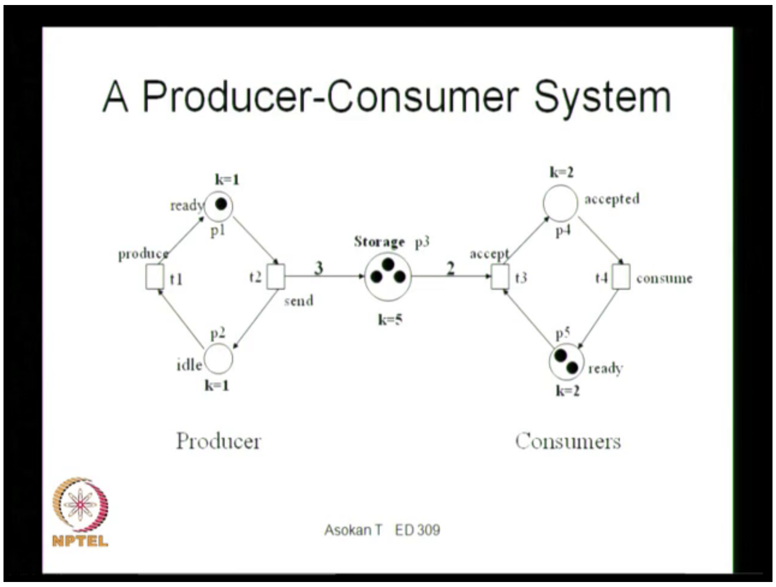
Similarly, the consumer can actually take 2 at any time that is why k is equal to 2 in both the cases. So, here as k is equal to 2 and only 2 can be removed from here by the consumer that is why this value is 2. So, whatever any transition takes place only 2 will be moved from here though there is a value of 5. So, 3 can come at any time and 2 can be removed and these 2 can actually take 2 each, but only one at a time and it can be consumed.

Once we consumed again they can take the items. So, this is the situation you have the producer producing items and then they going to the storage and then it is consumed by the consumer. Now look at this how do we actually show the dynamics or behavior of the system you can see here. So, there is one producer that is k is equal to one it is one producer who actually produces 3 items. So, this is k is equal to one producer and this is 3 k is equal to one is this producer one and this is 2 produ k is equal to one is another producer.

So, he can actually produce this is 3 items at a time now there are 2 consumers are free and the producer is also a free. So, we have the producer here idle and the customer consumer here also ready to consume. Now the first thing is basically the producer will produce some item. So, it will actually producing the item and then the item is ready for taking. So, that is the token is transferred from here to here. So, the producer is given one item it is ready for transport and then that actually transferred back to the storage space.

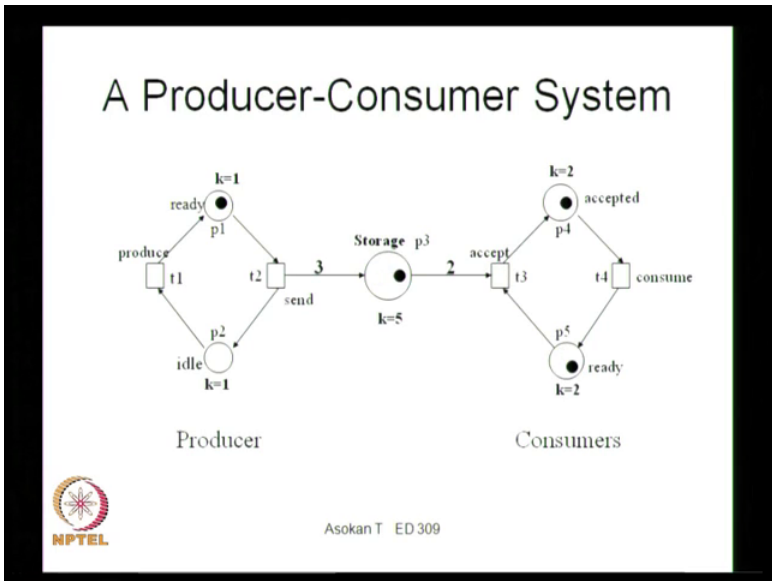
So, this one producer actually produces 3 items that is why once he sense the item to the storage there is 3 places available 3 value of this place is 3 now and the producer is again free to produce. And now he can produce more, but it cannot be sent here because next time if he sends it will be another 3, but it can take only 5 therefore, we need to ensure that before he sends this item to here this has to be accepted by the consumer. We will see how do you actually represent the scenario.

(Refer Slide Time: 31:48)


A Producer-Consumer System

So, once we have this he can actually produce further and then it will be ready, but you cannot be send you can see here this cannot be send because if you send it will overflow. So, though he is ready to send he cannot send it because in a overflow therefore, one consumer has to accept the product.

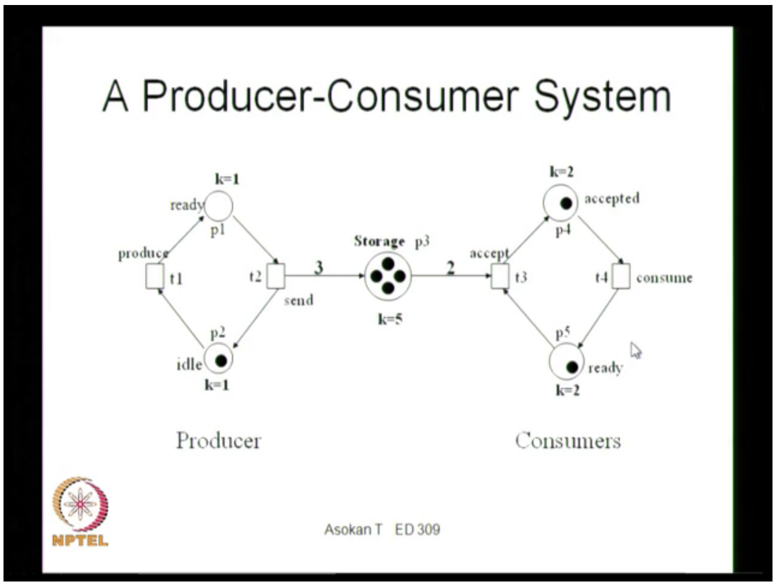
(Refer Slide Time: 32:05)


A Producer-Consumer System

Therefore we have to accept the products. So, the one customer has accepted it or consumer accepted it and that is why he has accepted 2 items here and then one place values one here one here because one consumer already accepted the item in an another consumer is free. And now he cannot accept further because there is only one item. So, he cannot accept this cannot be accepted because there should be 2 node to activate this one there is only one at this place therefore, it cannot be activated.

So, we need to activate the producer again. So, the producer should actually produce it further. So, that is why it already produced and that should be sent to the storage space.

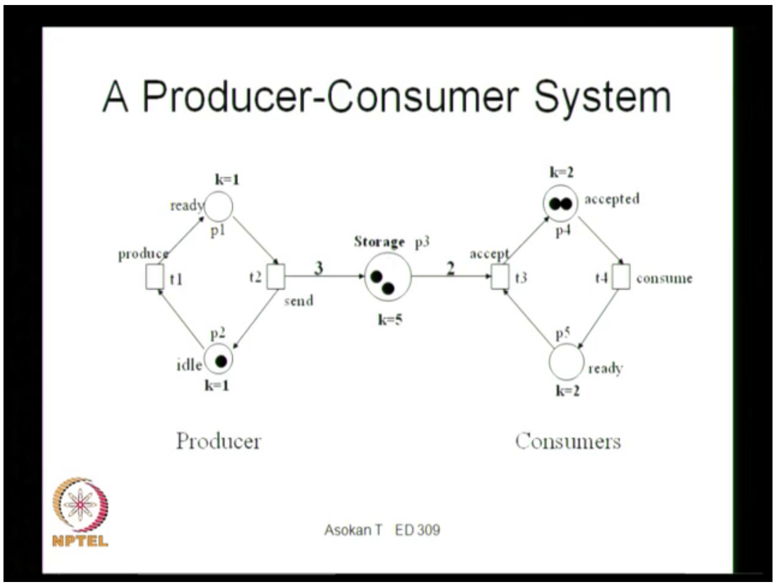
(Refer Slide Time: 32:43)


A Producer-Consumer System

Asokan T ED 309

So, it actually goes to the storage and then storage has got 4 now. Again we can have one another consumer accepting it now we can actually have 2 consumer accepting it because there are 2 plus 2 4 items available, but there is only one consumer ready for accepting it is only one consumer is available. So, he can actually accept it.

(Refer Slide Time: 33:02)

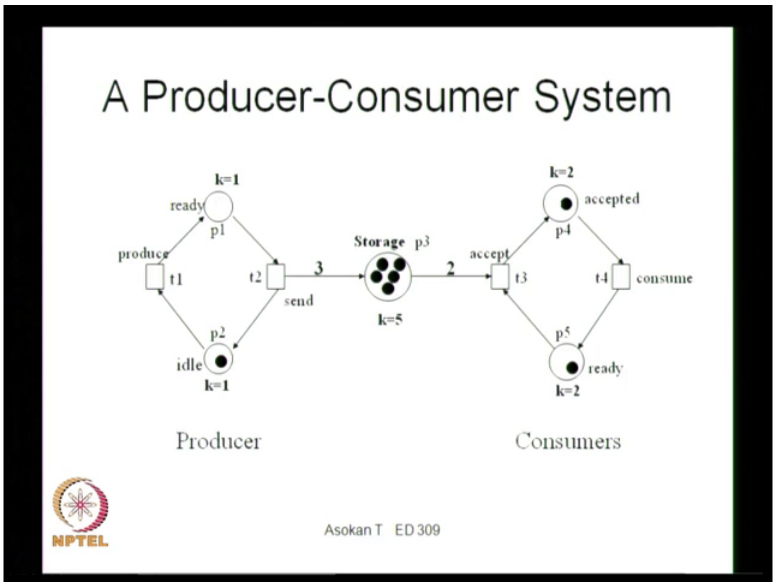

A Producer-Consumer System

Asokan T ED 309

So, it has gone back to the accepted and then this both the consumers accepted the item.

So, you have this both consumers accepting the item and there are 2 items already available and now the producer is still able to produce 3 and send it because we can have another 3 in the storage buffer. So, we can actually either producer can produce all the consumer can actually consume in accept it also. So, the producer is again producing one more set. So, he is ready and he can actually send it to the storage.

(Refer Slide Time: 33:33)



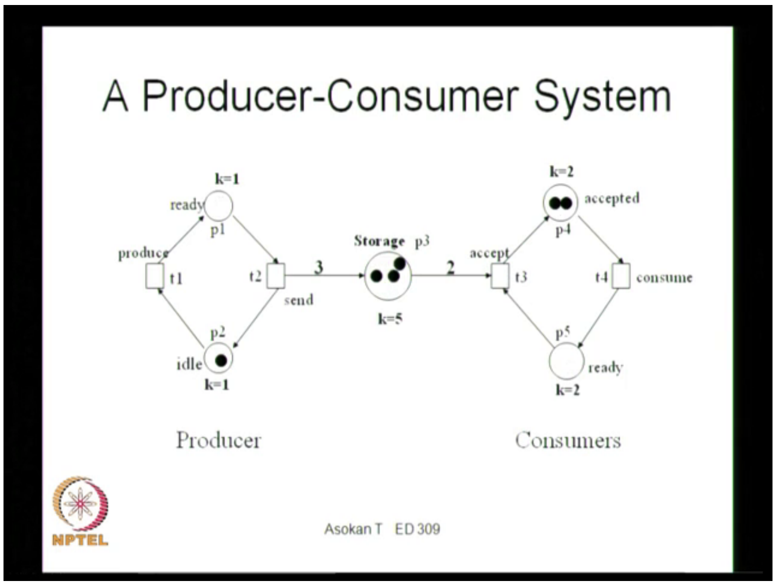
A Producer-Consumer System

So, we can see that now the storage is full with 5 items already filled in.

Now, producer can still produce, but he cannot be sent, but there are no consumer is also available to accept it. So, now, it is a full condition consumers are not available to accept it because you can see here the consumer now tokens available. So, this transition cannot take place. So, unless the consumer consumes the product will not be taken from the storage space. So, we need to ensure that at this situation the consumer consumes or the producer stops producing the item.

So, here see that producer is free, but we are actually making the consumer consume the item and then consumer is made available. So, now, the one consumer is free to accept another 2 item therefore, you can actually make sure that this consumer accepts further item from the storage before the producer goes for it is production. So, you can see again 2 are accepted by the consumer 2 items are removed from the storage. So, it actually accepted and then can consume and the you can continue the cycle again.

(Refer Slide Time: 34:30)


A Producer-Consumer System

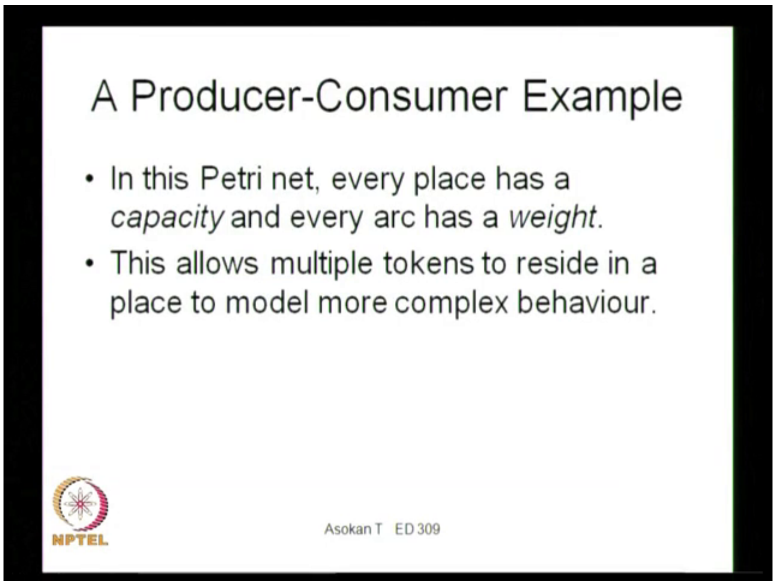
Asokan T ED 309

So, here the place values are given. So, that is one important point about this particular petri net you can see that the place values that is one producer produces 3 items. So, that is the place value here whenever he sends he sends 3 items. So, this is the 3 value and then from sending accepting only 2 can be accepted. So, that is why you have 2 place value this actually shows that the number of tokens to be handed or to be moved is equal to the number of with the value of kth shown here.

So, 2 will be transferred and then I will be having the accepted also you can actually have accepted 2 and that will be consumed and then the consumer will come back.

So, this way the scenario of concurrency as well as the sequential activations can be represented using petri net. So, this is one place where we can have complex scenarios presented using petri nets, where we use the weighted arcs as well as the place values to represent the capacities of the buffer or the producer or the consumer how much they can accept from the particular storage and how much they can produce and what will be the upper limit of storage space all those things can be represented using this kind of weighted arcs as well as place values, that is the importance of having this kind of petri net.

(Refer Slide Time: 35:59)



A Producer-Consumer Example

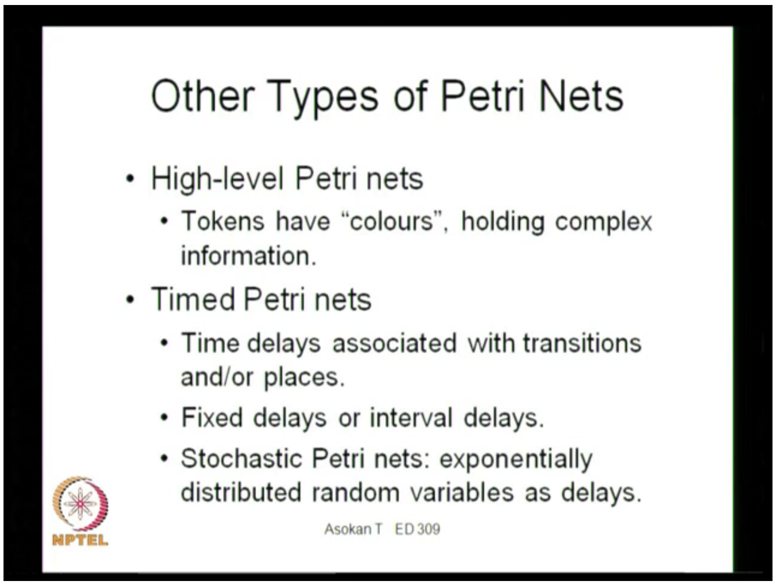
- In this Petri net, every place has a *capacity* and every arc has a *weight*.
- This allows multiple tokens to reside in a place to model more complex behaviour.

Asokan T ED 309

So, as you can see here this petri net every place has a capacity and every arc has a wait. So, we have a capacity as well as a wait for the arc and this allows multiple tokens to reside in a place to model more complex behavior. So, this actually helps us to have multiple tokens to reside in a place to model the complex behavior of the system as you saw in the previous one there were multiple tokens in the picture you can see these are the multiple tokens in the storage space we can have up to 5 tokens that actually represents the capacity of that particular place.

So, this can actually go maximum 5 and then we have to remove few tokens before we actually fill this with token ok.

(Refer Slide Time: 36:38)



Other Types of Petri Nets

- High-level Petri nets
  - Tokens have "colours", holding complex information.
- Timed Petri nets
  - Time delays associated with transitions and/or places.
  - Fixed delays or interval delays.
  - Stochastic Petri nets: exponentially distributed random variables as delays.

Asokan T ED 309

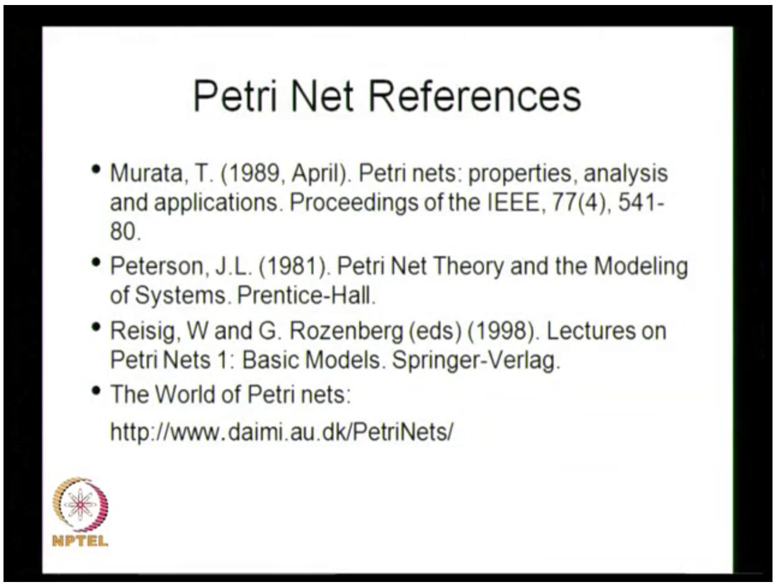
Now, there are various other petri nets also I explained it simple petri nets we will not be having the weightage or the place values there is now wait for the arc or the place values for number of tokens for the places and in the some other or the more complex petri nets you can actually have the arc waits as well as the place values the number of tokens, but then if you want to have more complex system behavior to be modelled using petri net we need to go for the complex petri nets where we have we called just high level petri nets.

In this case there will be colors holding complex information. So, we can have different colors for the place value or the token. So, you can have black token red token or green token which actually represent different scenarios. So, there can actually be incorporated in the petri net and apart from these we can have timed petri net also. So, in the previous cases we did not include the timing of the system. So, how much time it takes to make a production and how much time it takes for the consume the product or how long we can keep a product in the storage. So, that is not incorporated in the petri net.

But in more advance petri nets we can actually incorporate those values also the timings also. So, time delays associated with the transitions and or places can be incorporated and fixed delays or interval delays can also be incorporated and the stochastic petri nets which exponentially distributed random variables as delays can also be incorporated in the petri net. So, basically it shows that any complex system with any variations in in terms of behavior or in terms of the capacities or in terms of time can actually be a model using petri nets.

So, these are very useful for system design because in complex systems may encounter with various scenarios or some scenarios we will be having the situation where we need to represent the behavior without worrying too much about the timings, but in some scenarios we were more worried about the time especially when you are looking at something which actually to be executed in a particular time or there is a an urgency to execute some of the task within a particular time interval. So, in the such situation we need to look at the time involvement also. So, all this scenarios can be easily represented using petri net. So, this actually will be a very good useful tool in the design of systems.

(Refer Slide Time: 39:00)



Petri Net References

- Murata, T. (1989, April). Petri nets: properties, analysis and applications. Proceedings of the IEEE, 77(4), 541-80.
- Peterson, J.L. (1981). Petri Net Theory and the Modeling of Systems. Prentice-Hall.
- Reisig, W and G. Rozenberg (eds) (1998). Lectures on Petri Nets 1: Basic Models. Springer-Verlag.
- The World of Petri nets:
  http://www.daimi.au.dk/PetriNets/

If you want to know more about the petri nets you can actually see this references there are lot of references available. So, we can go through any of this a references Murata 1989 and then this Peterson 81 then of course, Reisig and Rozenberg at 1988 and of there are lot of internet resources available also you can actually search for petri net you will get lot of examples and lot of case studies. So, you can actually learn more about the use of petri nets for in the system design or practical examples can be easily obtained from this kind of resources.

So, with that we have come to a conclusion for the graphical modelling techniques. So, there are few other techniques also actually we talk about then we talk about the modelling one is the graphical modelling techniques basically looking at the behavioral dynamics of the system. So, they do not really take into the actual system dynamics or more into the engineering dynamics will not be taken into account.

So, there is 3 methods what we discuss the data modeling, the process modelling and behavior modeling, you are mainly to look at the behavior of the system and how the data transition take place how the relationships are taking place and how the behavioral dynamics takes place, these modelling methods can be used for analyzing those behaviors, but if you want to know more about the system dynamics the time response of the system and how fast the system will respond for a particular input and what are the physics involved in the system and how do you model the physics behind the system.

So, this can actually be modelled using system modelling and simulation techniques there are various modelling and simulation techniques to look at the system dynamics and those methods and the methodologies for system engineering we will discuss in the next class. So, till we meet and goodbye to all of you.