Real -Time Systems

Prof. Dr. Rajib Mall

Department of Computer Science and Engineering

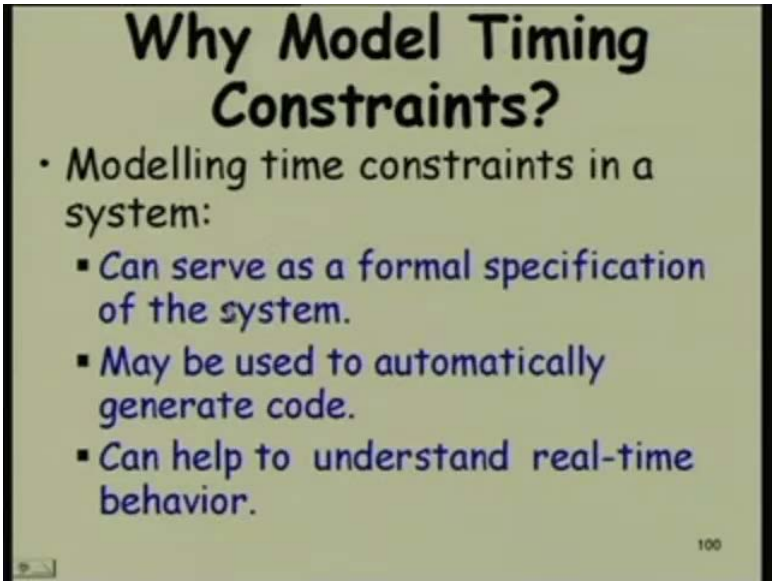
Indian Institute of Technology, Kharagpur

Lecture No. # 05

Modelling Timing Constraints (Contd.)

Good morning, let us get started from where we left last time. So, we were seeing basically, what are the different types of timing constraints that we have to deal with when we try to develop this real time system? And we had seen that deadline is a popular constraint very frequently occurs, but we will also have delay constraints and duration constraints. And we had said that we need to model this for various reasons; one is to understand these constraints better. The second is that we can generate code from this and I was saying that there are many tools being used in industry, where the model timing constraints automatically generates code for them. So, let us see now, how do we really construct the models.

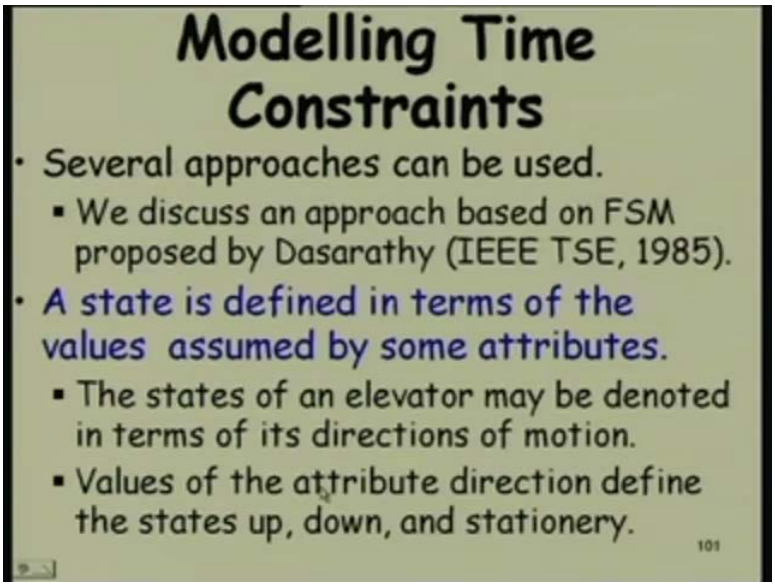
(Refer Slide Time: 01:14)



So, we we had seen that it serves as a formal specification can be used to verify the behavior and to specify the behavior of the system and the environment. Accurately can

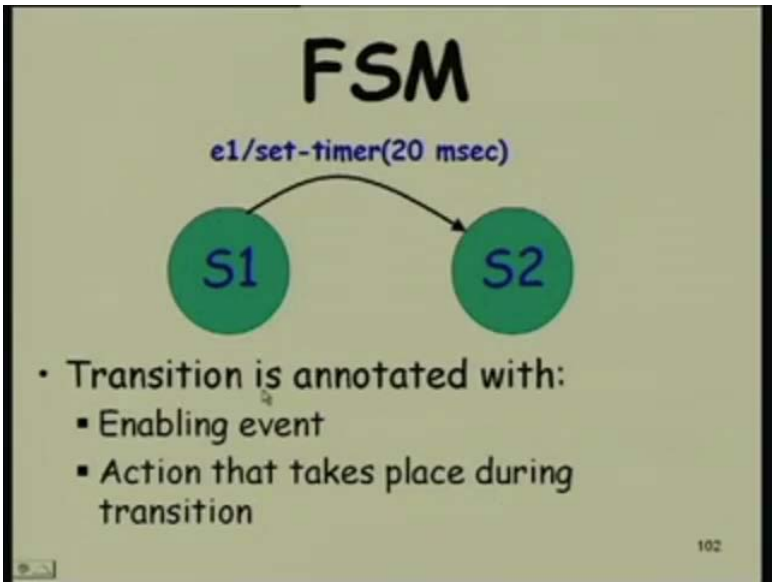
be used to automatically generate code and also, it helps us understand get insight into the behavior of the the real time behavior.

(Refer Slide Time: 01:40)



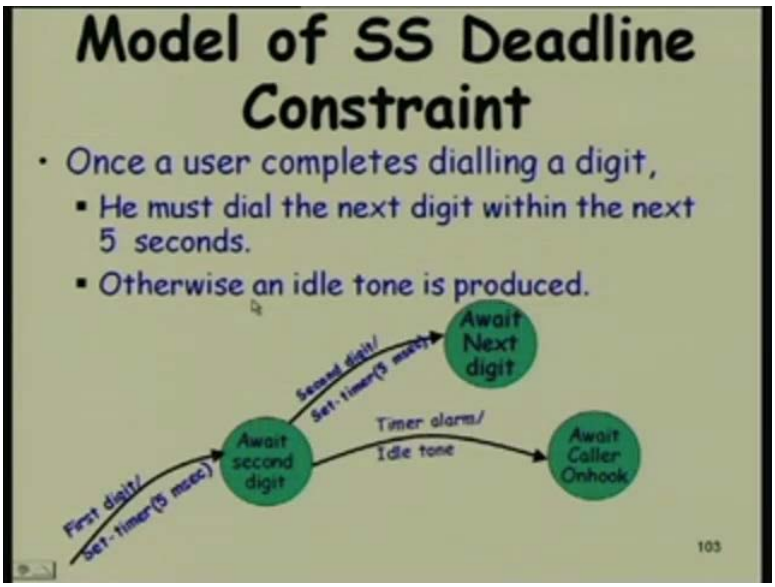
And we had last time said that we will use extended finite state machine for modeling the timing behavior.

(Refer Slide Time: 01:51)



So, this was the basic syntax that we had mentioned that two states if there is a transition, it is annotated with the event, which enables the transition to take place; and then some actions that take place while the transition occurs.

(Refer Slide Time: 02:11)



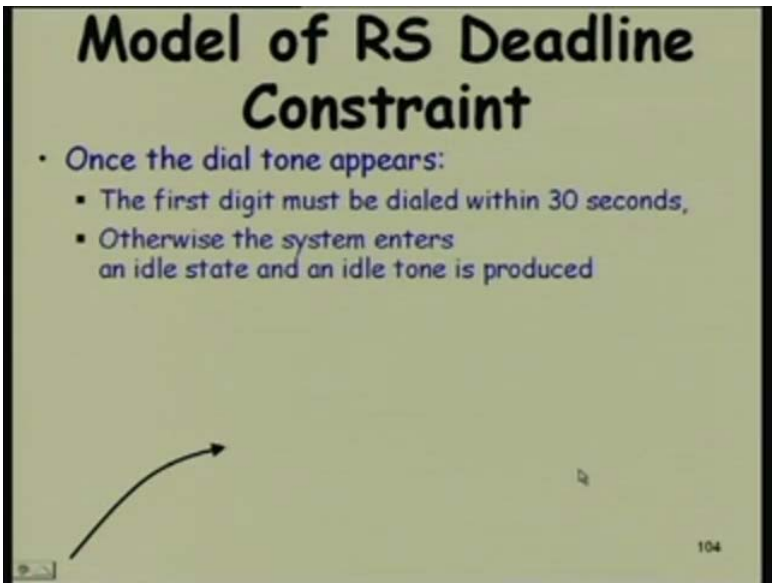
So, the first thing, we are trying to model last time is a stimulus stimulus deadline constraint or the deadline existing between two stimulus events. So, the example, we had discussed from the telephone domain is that once the user completes dialing a digit, he should dial the next digit within the next 5 seconds. Otherwise, idle tone will be produced.

So, let us see how the model will be constructed? I think, we had just seen this last time that once the first digit is dialed, set a timer for 5 millisecond, and then we will await the second digit. I think, somebody mentioned this just corrected here written last time first digit corrected it to await second digit. And then once the second digit comes, it awaits the next digit and then the next one and so on, and some of you had mentioned that we can use a loop here. But I was telling that if we since the digit is are finite just 8 or 10, we can even model them the different states here.

But if the timer alarm comes, before the second digit could be dialed that is 5 milliseconds has expired 5 second actually, it is not millisecond; 5 seconds have expired before the digit is dialed. Then as soon as the 5 second expires, we will get the timer

alarm and it will transit to await caller on hook and produce the idle tone. So, the idle tone will keep on producing, until it enters the on hook or the handset handset is replaced on this telephone.

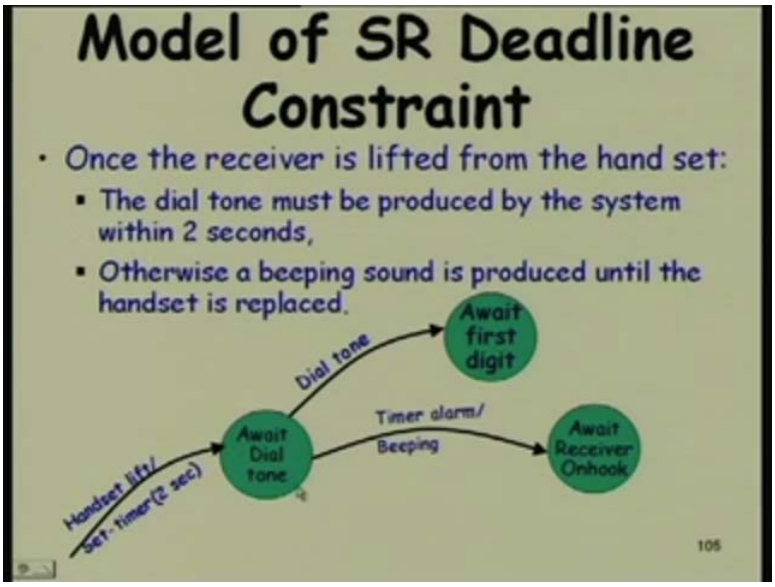
(Refer Slide Time: 04:17)



Now, let us see, how deadline constraint between a stimulus and the corresponding response can be modeled. So, the example that we are considering is that as soon as the dial tone appears. You lift the handset and then after sometime the dial tone appears, and as soon as the dial tone appears, then you must dial the first digit within 30 seconds. Otherwise, it will just enter into an idle state and beeping tone will be produced. So, very similar the dial tone is produced and it enters the await first digit state and set the timer for 30 seconds and here there is again a problem. It should be first digit. So, yeah first digit this should be the first digit; it is not the dial tone; the dial tone has produced and that is why it has entered first digit and once the first digit appears. It should go to next await next digit.

So, this should be first await the first digit and if the first digit is not is not dialed and 30 seconds have passed, then the timer will fire, the timer alarm will event will occur, and it will produce beeping until the caller replaces the handset.

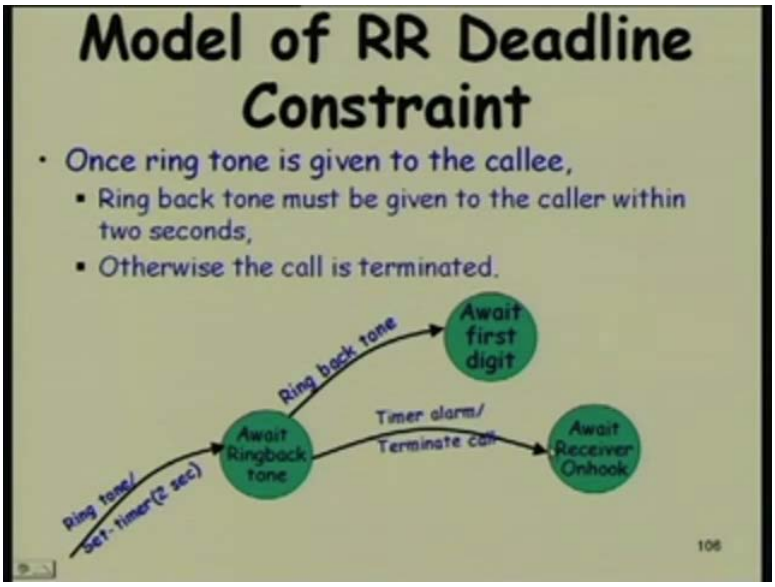
(Refer Slide Time: 04:17)



Now, let us see the other example, we had considered that is deadline of a response from the corresponding stimulus event. So, here the once the receiver is lifted from the handset then the system must produce the dial tone within two seconds of lifting the receiver. And in case, the system cannot produce the dial tone, then it will produce a beeping sound and the handset has to be replaced and dialed again.

So, the receiver sorry, the caller has lifted the handset and then this is the event, handset lift event and two second set timer. So, we are modeling all timing constraints through timers and we will see that when we do programming also, we will use something similar real time programming. So, it is awaiting the dial tone and if the dial tone appears, then it enters the first digit state and if the timer expires before the dial tone could be produced by the system, then the system produces a beeping sound and the until the caller replaces the handset.

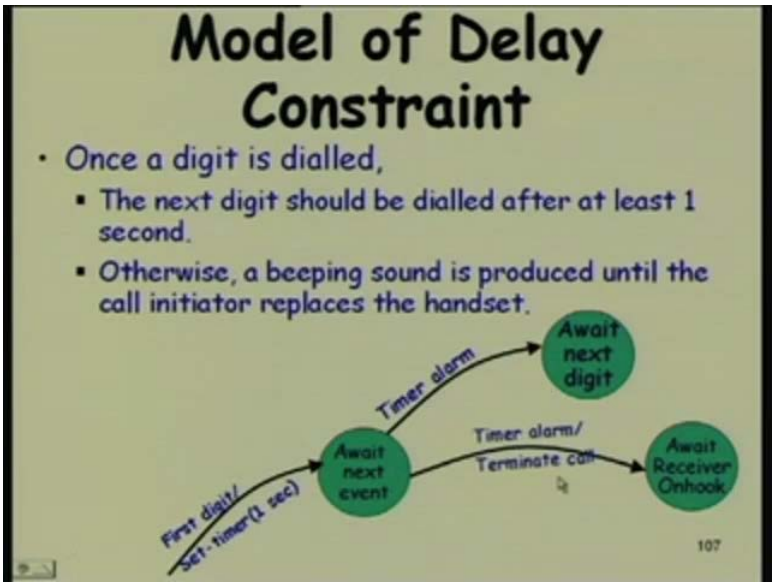
(Refer Slide Time: 07:23)



Now, let us see the other example of a deadline between of one response event from another response event. Once, the ringtone is given to the collie. The ring back tone must be given to the caller within 2 seconds, otherwise the call is terminated - this is the example, we had seen. So, after the digits have been dialed, the ring tone is given to the caller and once the ringtone has been given to the caller the system timer for 2 seconds is set; and it enters the await ring back tone state. And if the ring back tone is produced by the system, then it is not await first digit actually, there is a mistake here. It just enters the call state right ongoing call state. There is one problem here actually, I had just copied the other diagrams and just modified them in each slide. So, I had forgotten to modify this one.

Now, if the ring back tone could not be given, then the call should be terminated. So, the timer alarm has fired, because two seconds have passed and the ring back tone could not be given to the caller. And it will terminate the call, and it will also produce an idle sound possibly we could have written here. Callee's are no, it is not mentioned here, but if we had mentioned here that call is terminated and also a beeping sound is produced, then we could have also written here timer alarm and then the actions are terminate call and beeping. And it continues in this action continues, until the receiver is replaced on the sorry the handset is replaced on the phone.

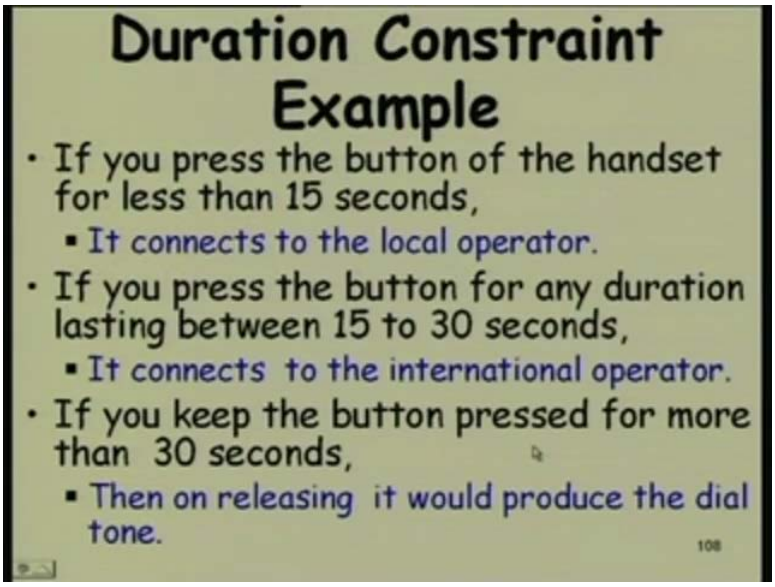
(Refer Slide Time: 09.33)



Now, let us see the delay constraint modeling, the example that we had taken for delay constraint is that once a caller dials one digit on the telephone the next digit should be dialed at least after 1 second. If it is too quickly dialed, then there will be error and beeping sound will be produced, until the caller replaces the handset. We were just saying that in our conventional phones, it is difficult to really dial too fast within 1 second. It is not within the human response time anywhere only problem that might occur is that you press two keys at the same time or something or may be in very quick (( )) just one after other.

So, let us see how this will be modeled the delay constraint again, we use a timer here, so, the first digit has been dialed and we set a timer of 1 second right, and then it awaits the next event or awaits the next digit. And here also there is a problem actually not written it properly. So, the second digit comes so if the second digit timer alarm is alright, the second digit comes here this should have been second digit. If the second digit comes here, before the alarm, and then it should go to await receiver on hook, it should produce an it should not terminate call, it should actually produce a beeping sound. I am sorry; here both are these incorrect just did it quickly. If the timer alarm it should be have been the second digit and the terminate call should have been beeping.

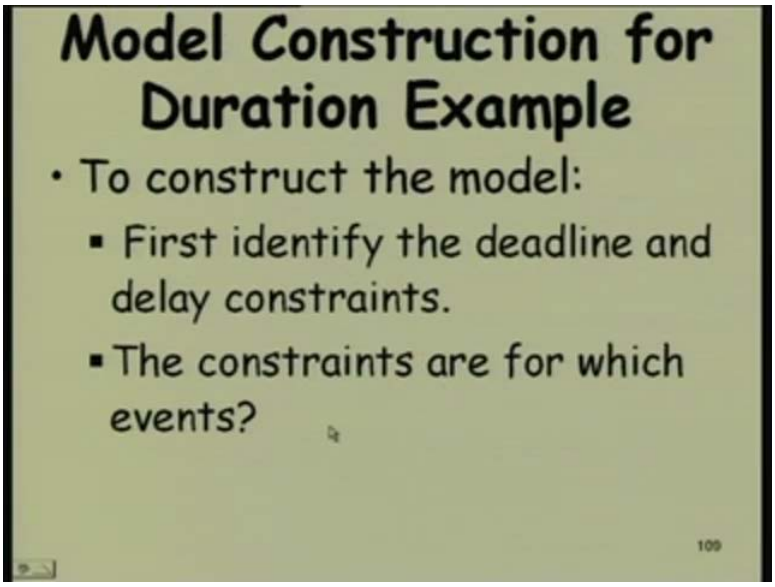
(Refer Slide Time: 11:38)



Now, let us look at that the duration example, this is actually more involved, because there are several types of deadline and delay constraints involved here. So, the example of the duration that we are discussing from the telephone domain is that if the button is on the handset is pressed for less than 15 second, it will connect to the local operator, and if you keep on pressing the button for more than 15 seconds up to 30 seconds, then it will connect to the international operator and any pressed more than 30 seconds; if you keep on pressing for more than 30 seconds, then it will produce the dial tone.

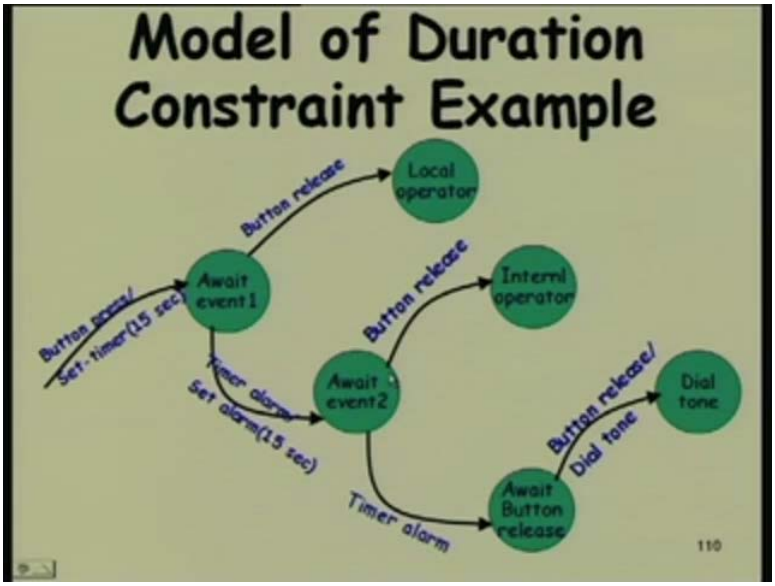
So, this the duration that between 0 to 15 connect to the local operator, between 15 to 30 duration, it will connect to the international operator, and 30 more than 30, it will connect to the it will produce the dial tone.

(Refer Slide Time: 12:44)



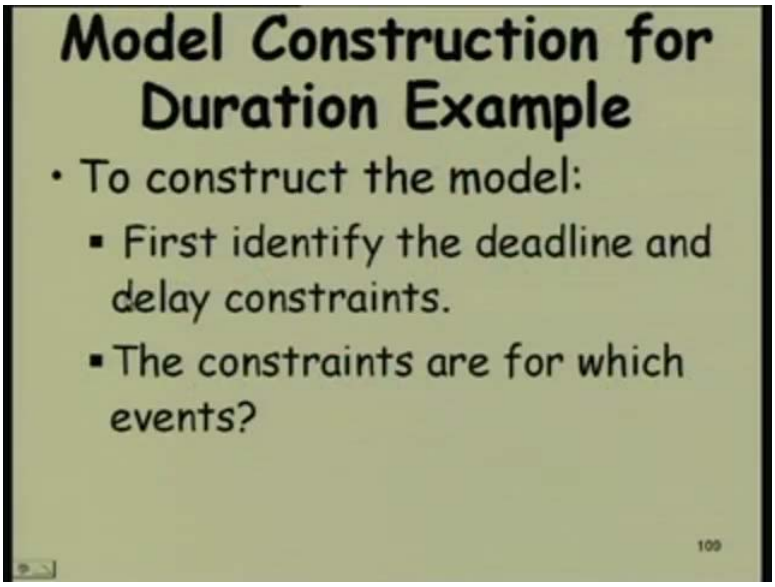
But how do we model this duration constraint? We have to first identify the deadline and delay constraints involved here, and then we will have to identify the constraints are for which events. So, let us look at the model, and then try to answer these questions.

(Refer Slide Time: 13:08)



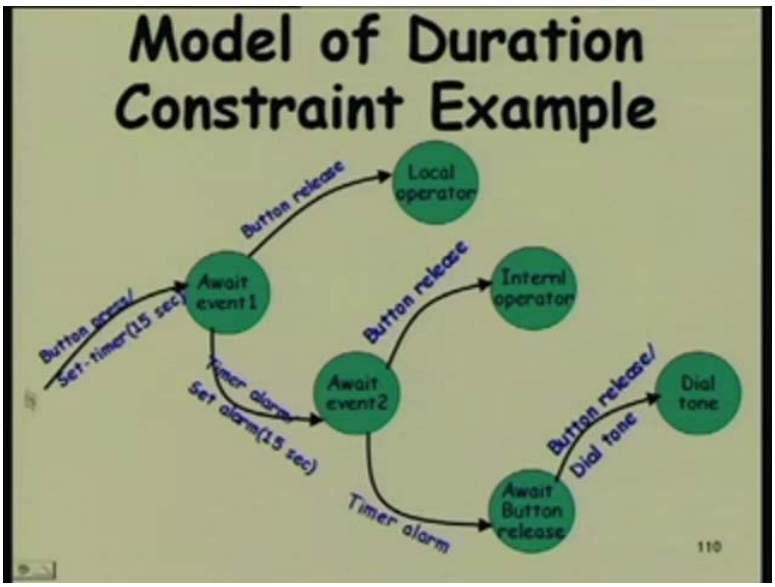
So, see here I am sorry; again there is a mistake here. It should have been button release right instead of button press. See you release the button and the handset that is what is written here also.

(Refer Slide Time: 13:25)



I am sorry; yeah it is ok. So, button is pressed. So, you have lifted the handset and you are pressing the button and if you keep it pressed for less than 15 second and try to listen, it will connect you to the local operator between 15 to 30 international operator and more than 30 is the dial tone will be produced.

(Refer Slide Time: 13:08)



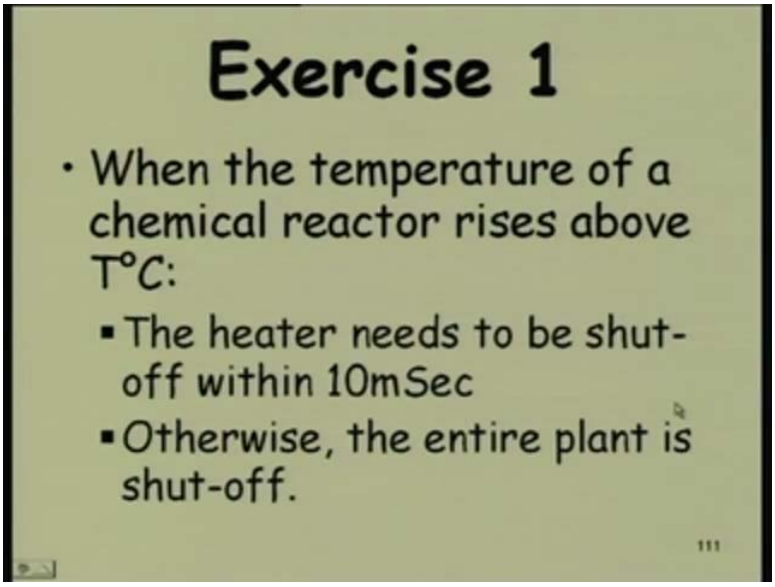
So, let us see the model. So, this is alright you have you are pressing the button and setting the timer for 15 seconds. So, this if the button releases occurs, this is the deadline

the button release occurs between 15 less than 15, then it will connect to the local operator, local operator state will be reached.

But the timer alarm comes, that is a delay of 15 seconds; after the delay of 15 seconds, it will reach the await event 2; and in await event 2, we have a deadline here, just see here by before 15 second the button release occurs, the deadline is 15 here for it to reach the international operator state.

Now, if the timer alarm occurs that is again another delay of 15, then if the timer alarm comes here before the button could be released, then it will await the button release for anytime that you keep it pressed. There is no limit that is what we had said more than 30. So, keeps on waiting here until you release the button, and the dial tone is produced after you release the button and it (( )) to the it goes to the dial tone state. Now, let us try to do few small examples. Let us see, if you can do, please do, it neatly I will just show it here.

(Refer Slide Time: 15:33)



So, first example simple one the temperature of a chemical reactor rise, when the temperature of a chemical reactor raises above some threshold T degree centigrade, let us say. Then the heater needs to be shut off within 5 milliseconds of this event occurring otherwise, the plant is shut off.

Just a simple problem, modeling problem, let us see using the syntax and the concepts that we had discussed, the timing is modeled using a timer. Let us see, if you can do this please try. So, is this a deadline or a delay constraint? What do you think?

Deadline constraint

It is a deadline constraint.

The heater must be shut off within 50 milliseconds; it is a deadline constraint. And if the deadline cannot be met by the system, then the plant is shut off.

So, is it a performance is it a performance constraint or is it a behavioral constraint, what do you think?

(( ))

It is see there is nothing to I mean really think of too much, because it is on the system event that whether it can shut it off, some event has occurred in the environment; environment is basically temperature exceeded for some reason. So, can the system be shut off within 10 or can the computer there controlling, it produce the event to shut it off within 10 millisecond. So, it is a constraint on the system event event produced by the system. So, it is a performance constraint clearly.

Now, let us see whether you are able to draw the model. So, as soon as you draw it please give it here anyone, who completes first? I will just display it. So, that the others can see and give their comments, if it is correct or if there is any difficulties there.

So, I hope the problem is clear, simple problem. Done? Great, let us see yeah what is your name?

Bhanu prasad

Sorry.

Bhanu prasad

Bharath

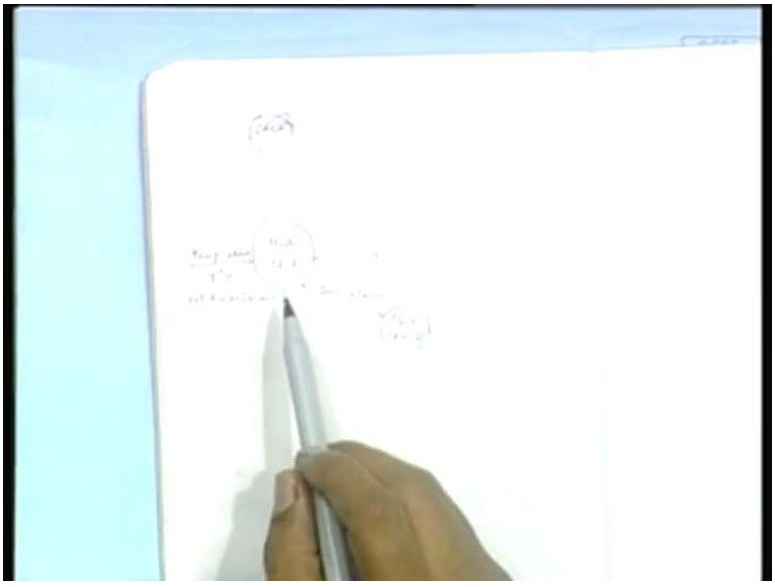
Bhanu Prasad

Bhanu prasad

So, let us see, what Bhanu Prasad has done?

Can it be projected?

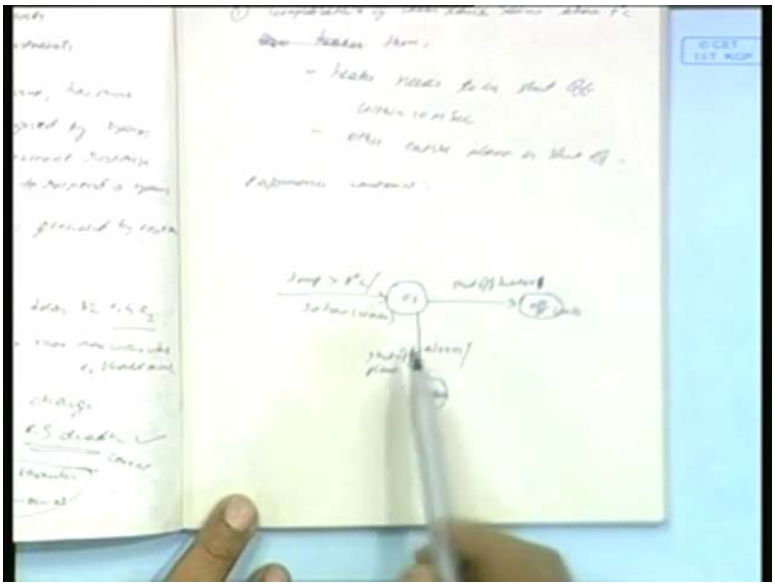
(Refer Slide Time: 19:20)



Let us see, if it shows, it is actually showing very small. But still let me just read out, so that you can see here that the way he has done it here is that the event is temperature above T degree centigrade. And then he has a set timer the action is set timer and is set it for 10 milliseconds, and then what is this is not labeled anything? So, does it go to a state or is it just a this is nothing here that is what he says?

So, if the timer alarm comes then the plant is shut off, but what about I mean until, the timer alarm it stays in the header. What is this heater shut down it stays in heater shut down. And then once the timer event comes the plant is shut off does not look alright, I mean what do you think? So, what is the problem here anybody has done it different way? Yeah please yeah give it. So, let us see here actually for this to be projected may be, you should draw it slightly larger.

(Refer Slide Time: 20:42)



So, let us see this I am sorry;

What is your name?

Sreekanth

Sreekanth

Let us see, what sreekanth has done? So, if the temperature is greater than T degree centigrade, but this is a condition know it is not a event.

(( ))

Event of temperature greater than T degree centigrade. Set timer to 10 millisecond and then it enters a state called as e 1 and then if the shut off heater event occurs, then it off state

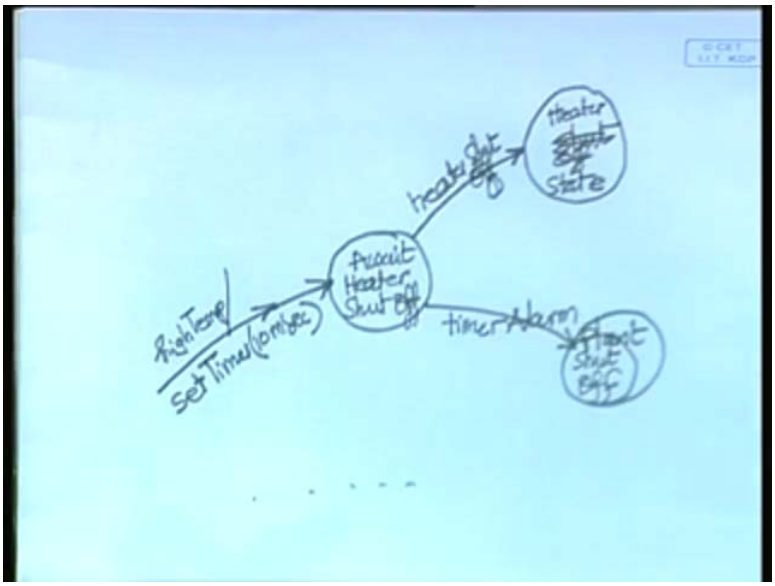
Heater of state is it heater of state and.

Alarm (( ))

On alarm, it shuts off the plant plant shut off event looks more acceptable, But one thing is that see the states, you should give meaningful name we could have, we could have

given await heaters shut off that might have been. Let me just redraw his what he has drawn there let me just redraw that here.

(Refer Slide Time: 21:50)



So, one is the event is high temperature; high temperature is the event, and the action is set timer to 10 millisecond. And once the high temperature is sensed, it awaits heater shut off, await heater shut off. And if the timer alarm comes before the heater could be shut off, the timer alarm event comes, then it went to it goes to the shut off state.

On the other hand, if the heater shut off signal could be generated to the corresponding sensor, then it goes to a heater shut off state, heater shut or something does it appear. I am sorry; the other one is plant shut off; plant shut off, plant yeah right Yes, plant shut off yeah plant shutoff yeah that is a more meaningful name here. Plant shut off and this is heater shut off, but we have given the event name here heater shut off. So, I will just try to give a different name here. Off state, Heater off state yes heater off state that is a better name here.

Heater off state this is just a simple deadline like we had, we were discussing the other examples telephone domain is something very similar to that. Right Is that looks or anybody has any difficulties here in this model, because if you are actually developing real time systems programming many times, you will have to actually develop this using some tools.

For example the stimulant state flow, I think he will tell you about that [they/there] there you model this states and on the events on which the state transitions occur, the timing all those are modeled and that actually generates the code.

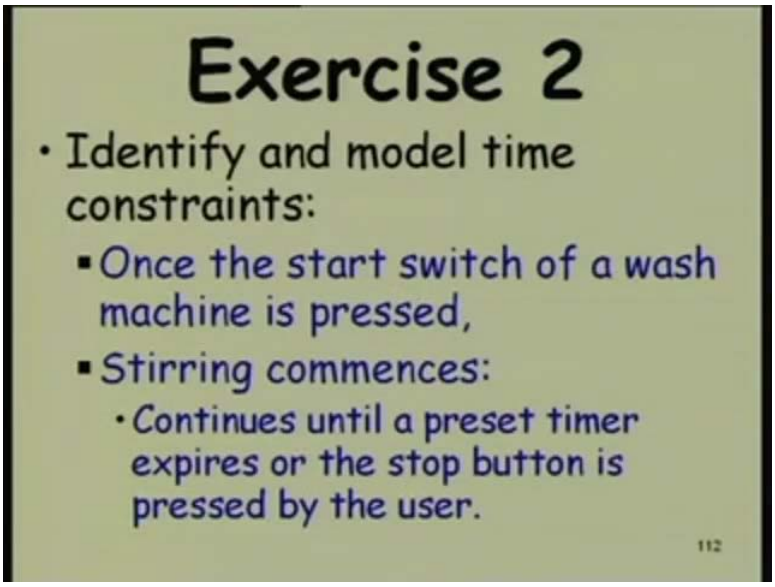
So, there are many tools like that which once you create the model. It will generate the code for you not. (( ))

Yes.

Let us see here this question. He says that high temperature how does this capture that this event is generated, when the temperature is greater than T degree centigrade possibly. We can mention here that high temperature is the event that gets generated the event occurs once the temperature is (( )) to be more than T degree centigrade.

So, this is the name of the event and temperature exceeding T degree centigrade is actually the condition temperature, exceeding temperature greater than T is a condition. And if that condition is satisfied then high temperature event is generated. Now, let us look at another example, let us see if you can do that possibly slightly more complex than this one please take your note books.

(Refer Slide Time: 25:56)



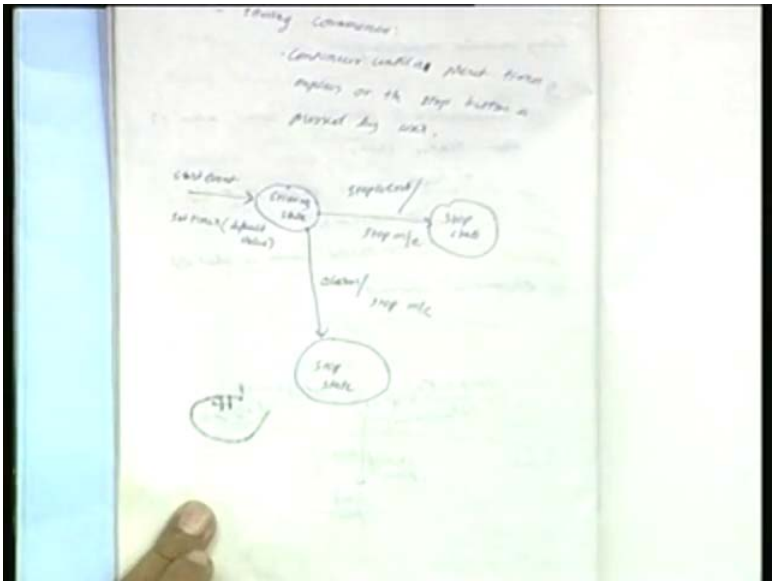
So, this is an example of a wash machine here. We need to identify the time constraints and model them. So, the behavior of the wash machine is as follows, if you press the

start switch on the wash machine then the stirring will commence. So, you put clothes etcetera. Water is filled up and then once you press the start switch stirring commences it continues until a timer expire. So, before starting pressing the start switch, you should have pressed the time. You should have set the timer. Otherwise, possibly it will be a default value and then it continues stirring until the stop button is pressed by the user.

You could have also alternate example like the water filling in the wash machine occurs, as soon as you press a water fill and the water filling continues until, the high level is sensed at the water in the wash machine or user presses the stop button.

So, please try this one, because many times even the concepts and the terminology is simple unless, we do one or two problems, we do not really get a feel of it very similar to the previous problem just small modifications the model please try. So, please draw slightly larger, so that we can project it. So, sreekanth again, has completed earliest let us see, what he has done?

(Refer Slide Time: 28:50)



So, again not very visible, but let me just read it for you. So, once the start event occurs. It occurs on pressing start switch so, start event is the event that occurs when start switch is pressed that is alright. Now set timer for the default value. We we can write some pre set value T or something T t is a preset value and T is either a default value or it is the one set by the user. And then if the stop event occurs or the stop switch is pressed then.

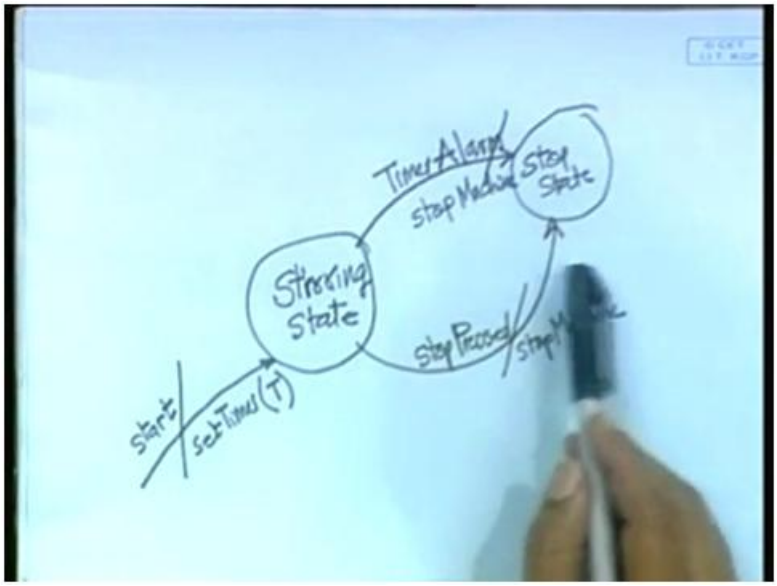
Stop the machine action is stop the machine and state will be stop state. The action is stop machine and it enters in a stop state alright. Now, if the timer alarm occurs, before the stop could be pressed and then stop machine signal is generated. So, this actions occur by the system is not it stop machine is action by the system and this the event stop, event is by the user. And here it again, enters the stop state we could have also used the same state.

Two states yes.

(( )) Timer alarm. The events

Ah here stop machine and timer alarm is it. It is a stop or timer alarm the event the two events. No, no no that is I am saying, that state can be one and we can have two transitions occurring here possible. In 1 transition we can keep 2 events. (( )) no that will be confusing, I mean how do you have two events on one transition that you cannot do each transition is enabled by 1 event. So, let me just redraw that since it is not looking very clean here.

(Refer Slide Time: 30:59)



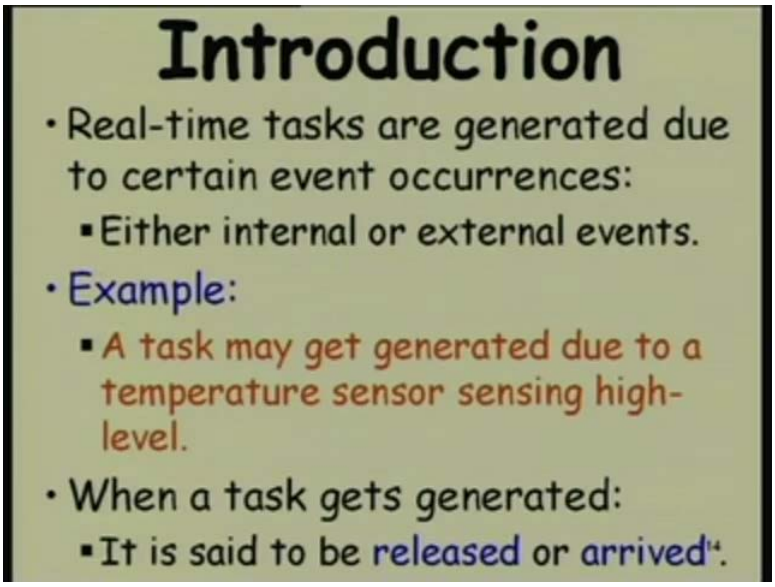
So, let me just copy his model here see 1 is it enters the stirring state once the start event occurs. And the action that takes place, before it enters the stirring state it set timer T, where T is the either the preset I mean the one set by the the user or it is a preset. It is a

default value. And then if the timer alarm comes, then it goes into the stop state and you can write the action is some stop machine signal is generated.

And it also enters the stop state, if the stop switch is pressed stop pressed and again the stop machine signal is generated by the system, that the event is by the user where as the action is by the system. And then it again, enters the stop state does that appear alright any difficulties here. So, let us proceed further. So, let us not spend time practicing more problems, because if you are able to do simple problems then possibly we can do more complex problems little bit of effort.

We have to just identify the timing constraints and then identify the states, and then model the I mean find out, which transition event causes the transition and then the action that takes place and that is about it.

(Refer Slide Time: 33:29)



Now, let us proceed further now let us look at some very basics of tasks scheduling. We have looked at the timing constraints. And we are saying that the main means adopted by any operating system real time operating system is tasks scheduling to be able to meet the timing constraints that are specified.

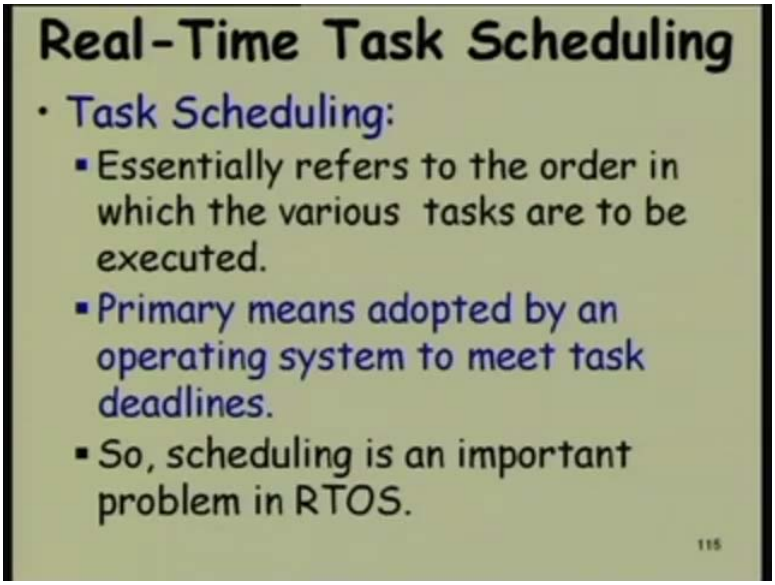
So, let us see what are the tasks scheduling techniques here, and how are these different from the tasks schedulers. You had studied in a simple; I mean, first level operating system course.

So, we had seen that the events that we are concerned about are either the ones that are generated by the environment or those which are generated by the system. We will call them as internal or external events.

And we had just seen the example of a event getting generated on a temperature exceeding some value and this is the event, which is generated by the environment. And when the event gets generated the operating system, it will enable some task right tasks are have been programmed written the program and there are handler events, we will see later.

And that will just enable the task right. So, as soon as the event occurs, the task will be enabled temperature, high temperature handler task will start. And as soon as the task starts, we call it as the task has been released or task has arrived, we will use this terminology. See, every area has it is own terminologies and we will use this kind of terminology that the task is released or arrived.

(Refer Slide Time: 35:29)



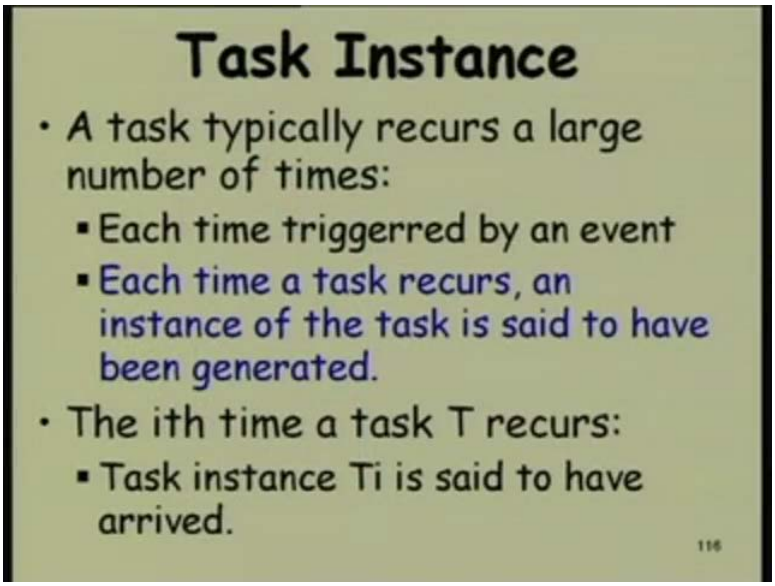
Now, by task scheduling what we really mean is that the operating system or the scheduler part of the operating system, we will find out, which task to be executed next or in other words. What is the order in which various tasks have to be executed? So, that is the basic task scheduling problem you have a set up tasks which keep on arising or arriving at different points of time due to events. And as soon as the events occur, the

tasks are the tasks arrive and what is the order they will be executed that is the basic problem.

We will see there are many ways this can be done one, as he says is that some priorities can be assigned that is just one way. Let us see and as I was saying you earlier that this actually for a real time operating system is a crucial component and in any course on real time operating system or a real time system. You will see that task scheduling is a major portion of the course, various types of scheduler suitable for various applications.

And analysis of tasks behavior task characteristics to find out a suitable scheduler. So, these are important problems. First is to understand the different types of schedulers to find their characteristics the scheduler characteristics and the situations, where they become applicable. And to find out which type of tasks will be handled, I mean which type of situation tasks will be handled by which type of scheduler and we will do some analysis that whether it can really meet the timing constraints of these set up tasks and So on. Quite a bit of theoretical background is also involved here. Let us let us look at that.

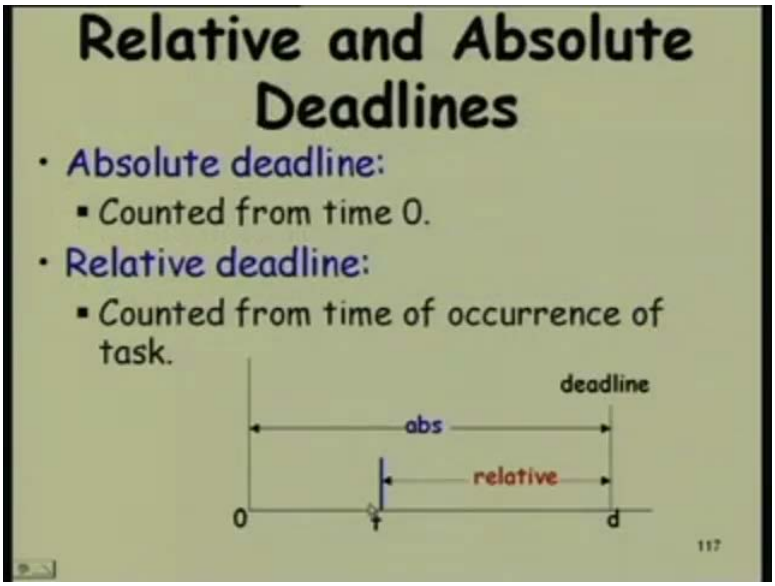
(Refer Slide Time: 37:31)



So, this is another term that we will use is tasks instance each task occurs a large number of times. For example, the temperature sensor might sense high temperature at various points in time and therefore, the same task will be fired again and again and each time the task is fired. We say that a task new instance of the task has occurred each time the

task recurs a new instance of the task is said to have been generated. And we will use the terminology, if T is the task the I time it recurs, we will use T i.

(Refer Slide Time: 38:22)



Now, this is another terminology and concept here, that we will be using there are 2 kinds of deadlines that we will talk of when you discuss about the schedulers. One is the absolute deadline and the other is the relative deadline. So, this is the task has got released here or arrived here. See here T is the time at which the task has arrived or is released and the deadline is d and this is the relative deadline is between the task arrivals to the instant of time at which it needs to complete.

So, t minus d or we will call it r is the relative deadline whereas, the absolute deadline for this tasks is d, so that is d is computed from 0. So, we will distinguish these two terms sometimes. T time 2 whether it refers to the some like when we power on the system.

Exactly.

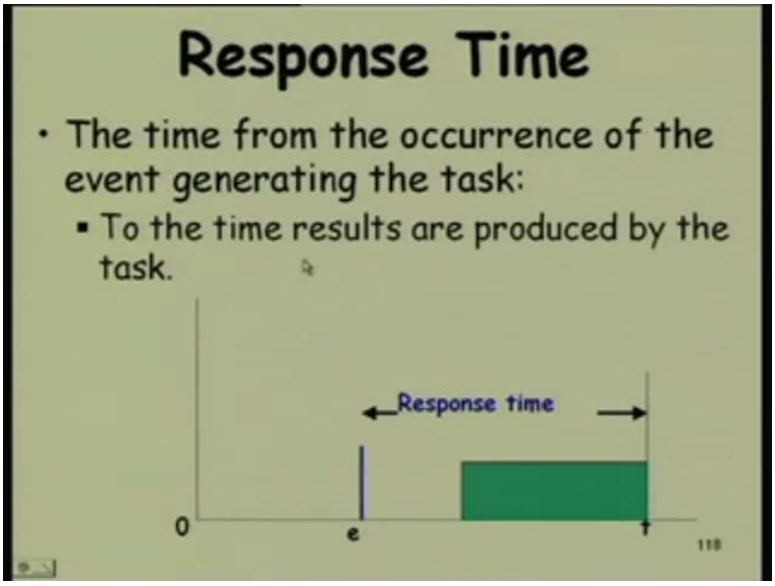
Exactly.

The zero time refers to when the system was started, and that it continues for long time may be, day, months, but zero is the time of system start. When it is actually powered on.

Exactly.

A system becomes operation that is the type 0 type. So, is absolute and relative deadline, relative is with respect to the task arrival and absolute is with respect to 0 time.

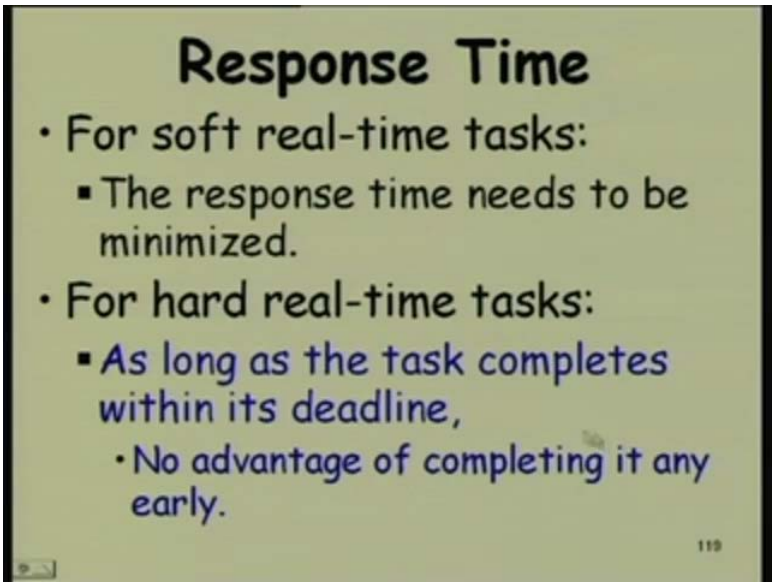
(Refer Slide Time: 40:03)



Now, we will also use this term response time, which you had also possibly come across in operating system course. That when the task has arrived here and then after sometime, it has started executing and it had completed by t, let us say. The task arrived due to a event e at. Let us say time e although, it is not a very satisfactory name here e is the time you should have written t or something anyway.

So, if t let us say that it is t, if that task starts at t the tasks is released at t e or arrives at t e and actually starts executing sometime later, because the operating system will start the task at some time later. And then it continues executing, let us says without getting interrupted and finally, completes at t. So, the response time is the time from where it arises or arrives and the time it completes looks.

(Refer Slide Time: 41:17)



For the operating system basic operating system course, we had seen that…

Excuse me sir.

Yes

Previous one.

Yeah please. This time is it also for the turnaround time. What we are not able to (( )), when we submit the task?

And (( ))

Yes.

(( ))

Yes

Turn around

(( ))

No, no

The entire thing, See you submit the job and the time it completes is the turnaround time, but that is also equivalent terminology. But here we with respect to the event, we are (( )) sir. No, there also, See event is the one which actually like he says it submit is the job right here. Also…

The

Exactly

Here also event e has created this task. But that we are starting from a bit late know sir.

No, there are also.

See you submit a job like let us say you pressed a key to start a task now the job has been submitted by pressing the key and then it takes some time to complete. It is the turnaround time. He says for the task. And we use the response time, because we I mean equivalent terminology, but what we are trying to mean here is that once a event occurs, what is the response of the system by which it produces the action.

The task is let us say the temperature, high temperature is sensed here and then it takes up the task of handling the high temperature some time, because some other task might be running here cannot immediately, start it starts it after some time. And then as soon as the handler task completes it finds that the shut down switch should be shut down signals should be generated and at this point the shut down signal gets generated.

So, the system took e to t e t minus e time to respond to the event right. So, this is the terminology, we will use, but there are other equivalent terminologies in operating systems. We will we will not use those terminology in a traditional operating course. Sir, but the time to respond is immediately know sir respond immediately, but the time it took is that much t minus e. To respond immediately, no, we are we are requiring the system to respond within certain time and then it responds in it is own time, which may be less than the expected time then it is doing fine.

If it is more than the expected time or the deadline, then the system has failed or it has we need to do some handling of the exception that arises due to that. Sir (( )) response

time when (( )) till the completion see after the completion only then the action will take place. Let us assume a simple case, where the action takes place only after completion.

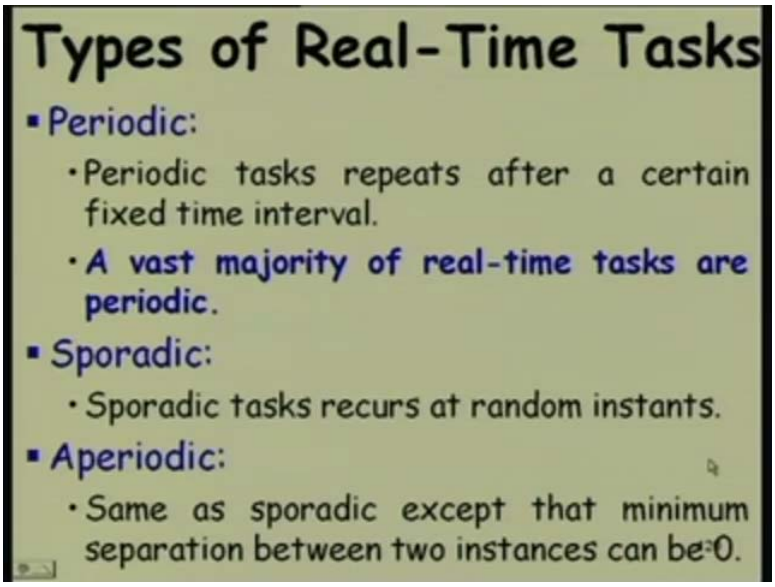
So, if the action can take place like let us say once the task is running and it takes actions at different points of time when it is running. So, then that is a separate case, right more complicated case that we are not talking here just considering a very simple case. That once the task starts, it does some computation based on the computation. It generates a signal and the times from the event that occurred to the task that ran and completed produce the signal that is the response time.

Now, in the traditional operating system course implicitly without I mean, we discussing there that what kind of tasks, we are running there those are all soft real time tasks or interactive tasks right the user submitted a job like he said. So, in those situations one of the objective of the operating system is to reduce the response time or minimize the response time.

And that is the main principle by which all the scheduling algorithms for a traditional operating system like Unix or windows is designed to reduce the response time for the tasks. On the other hand, the kind of scheduler that we will see here for hard real time tasks is that we do not gain anything by completing it very fast.

Minimizing the response time is not really very important here; as long as we are able to do it within the deadline. We are as good as completing it very fast. So, no advantage of completing any task early need to complete it by it is deadline that is the only requirement. So, we are not worried about minimizing the response time. So, that is one important point must remember, as we proceed further.

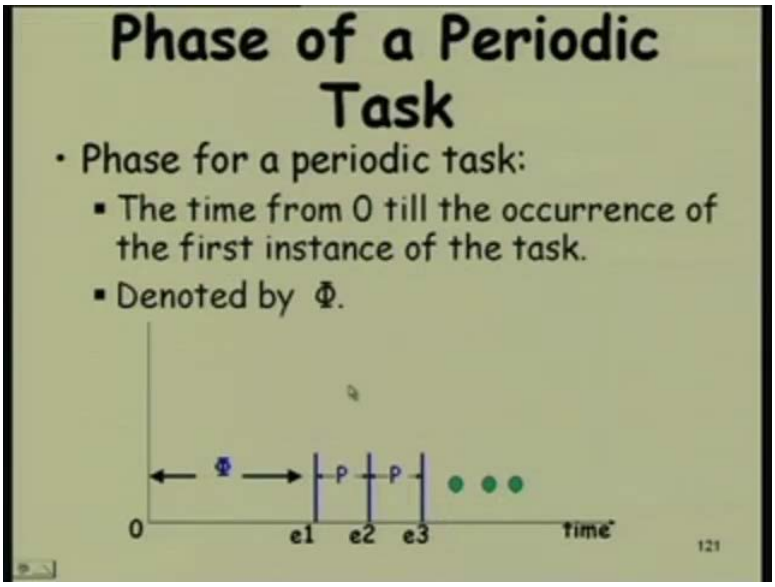
(Refer Slide Time: 46:30)



We had also seen that the tasks are periodic sporadic or Aperiodic there are 3 main types of tasks that we will be talking of. Periodic occurs after a time interval somehow, some clock set gives a interrupt and most of this events periodic events are in nature. And we will see that a majority of the task that will be concerned are actually, periodic tasks and in simple systems like, let us say a air conditioner. So, it sampling the temperature, periodically let us say.

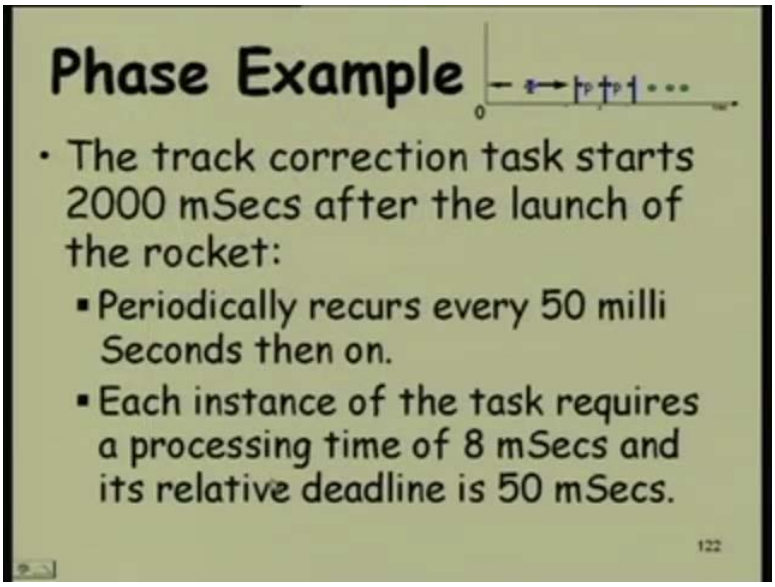
And then as soon as it exceeds some value it switches on right. So, these periodic events are many and many examples of that is there and in simple systems most of the almost all events 100 percent of the events in some systems are periodic in nature. Whereas, we will have few systems, which are few tasks which are sporadic occur at random instants. And Aperiodic which again occur at random instant, but the minimum separation can be 0.

(Refer Slide Time: 47:46)



Now, let us look at for a periodic task, what you mean by the phase of the task this is also a term that we will use very frequently. Once this system starts, the system is switched on then after a delay of certain time a periodic task starts and keeps repeating with a period of p right from this point the first instance of the task starts. And then every p it keeps on recurring. So, this 5 the time phi for the first instant of that instance of the task to start from the 0 time is called as the phase of the task is that we will use this term frequently.

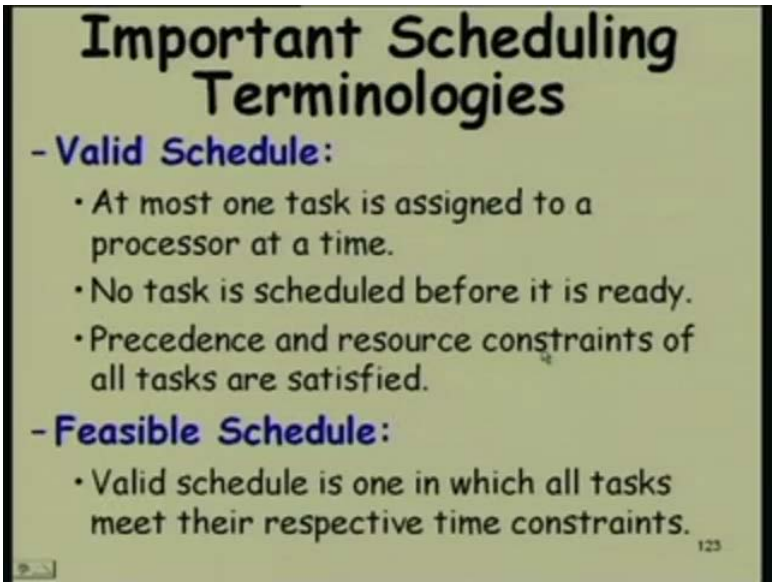
(Refer Slide Time: 48:45)



Let us consider an example of a phase, let us say that a rocket is launched and for 200 2000 milliseconds nothing happens, I mean this task is not started the track correction start does not start until 2000 millisecond. After the blast off the rocket. The track correction does not occur within 2000 millisecond, but then this track corrections task recurs periodically every 50 milliseconds from that point 2000 millisecond onwards. And then this task might have some characteristics. It takes some 8 millisecond to run deadline 50 millisecond. So, this is the characteristic of the track correction start task.

But till 2000 milliseconds, which is the phase the task does not start the first instance starts only at 2000 millisecond. And then at every 50 millisecond the track correction task is released or arrives.

(Refer Slide Time: 50:00)



Now, few more terminologies, because as we are saying that in any area whether you are talking of a doctor or you are talking of a computer person or you talk of a mechanical person, mechanical engineering person, everybody has his own terminology every area. So, if we get familiar with the terminology, we will understand that topics well and if we read a literature or a book we will be able to appreciate that.

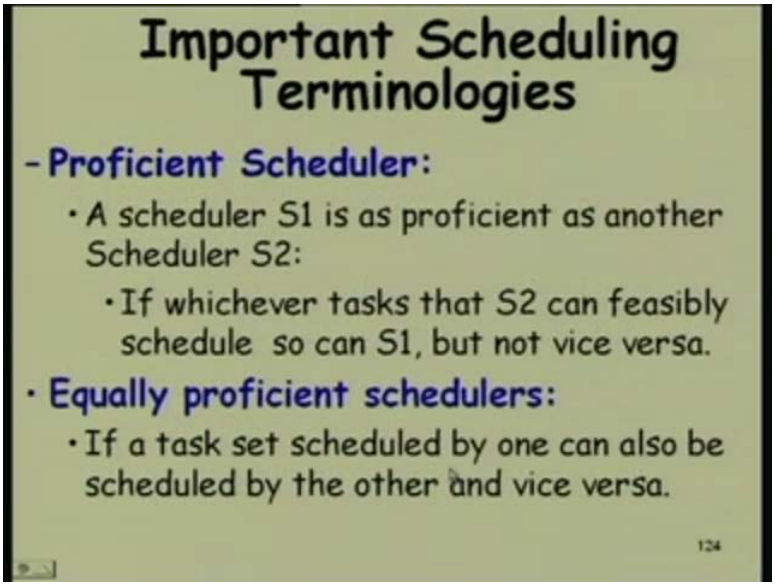
But of course, terminologies are a bit boring. You know, you have to remember, what are this terms? What is their meaning one is a valid schedule? A valid schedule is a schedule

produced by the scheduler. We will call it as a valid if at most one task is assigned to a processor, if a single processor not a problem, if we have a multiple processors.

Even in a single processor, if at the same time two tasks are given to the processor right that is not a not a valid schedule. And also no task can be scheduled, before it is ready. So, first the event should occur, and then we have a valid schedule and all the precedence and resource constraints. We have not discussed this till. Now, we will come across this as we proceed.

The precedence and resource constraints are satisfied then we will call it as a valid schedule, but a valid schedule may may not be a feasible schedule. A feasible one is where the timing constraints are met, see here we (( )) talked about timing constraints, if the timing constraints are met, then we will call it as a feasible schedules there can be many feasible schedules.

(Refer Slide Time: 51:49)



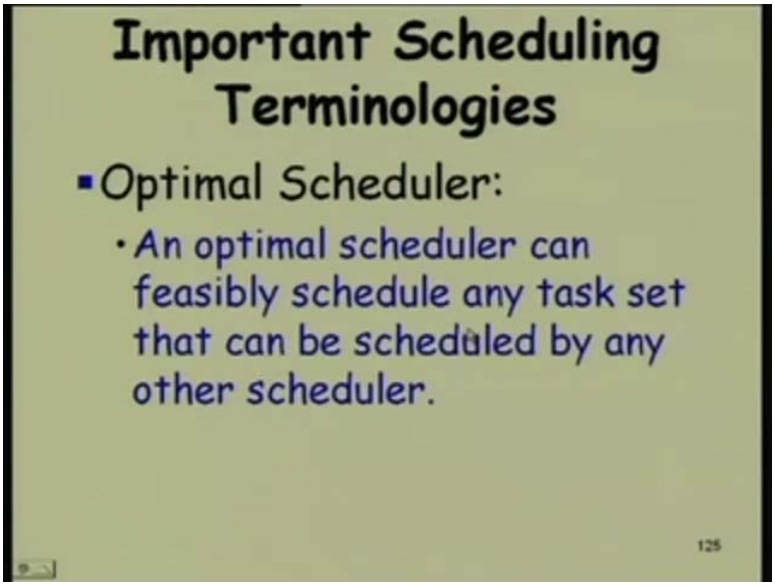
Now, as of these we have to select the best schedule, right and then in this case, we have the proficient scheduler. It produces a better schedule a proficient scheduler produces a better schedule. So, a scheduler s one is as proficient as another scheduler S 2. If the set up tasks for which S 2 can schedule it S 1 can also schedule it. So, S 1 is as proficient as S 2 so that means, whatever S 2 can schedule S 1 can also schedule and vice versa. Not vice versa (( )) then they are not vice versa.

No, no.

It should have been vice versa and vice versa, because it is as proficient. It is not vice versa, it is more proficient. I think, there is a the term is lightly misleading, we if we had said more proficient then this problem would not occur. So, proficient means it is more proficient. So, here is the main problem here s 1 is more proficient. It is not (( )) right there is a mistake here. As proficient as it at least. (( )) S 1 is at least as proficient as another or S 1 is more proficient than another it is at least as proficient. I think, that is correct term to use here.

So, S 1 is as proficient as another schedule or S 2, if S 2 can handle a set up task produce a feasible schedule for a set up task then S 1 also can produce. And equally proficient scheduler on the other hand is one where, if S 2 can produce a feasible schedule. So, can S 1.

(Refer Slide Time: 53:56)



And we can extend the terminology talk of a optimal scheduler. This terminology, we will use very frequently. Here an optimal scheduler is one which can feasibly schedule any set up tasks that any other scheduler can any other scheduler can produce a feasible task then the optimal scheduler should be able to produce the feasible schedule. (( )) it is a concept of optimal. (( )) valid optimal. It is a valid and also feasible; that means time

constraints are also met or in other words, if any scheduler can produce a schedule by which the time constraints set up task is met the optimal scheduler can definitely do that.

But sometimes some scheduler will fail to produce a schedule for some set of tasks some combination of tasks, but the optimal scheduler in any case, will produce the schedule. If any other scheduler is able to schedule, I am not saying that the optimal scheduler will be able to handle all cases. No, many of the tasks it may not be able to handle right may not be able to produce a feasible schedule for many types of tasks. But it is at least as good or it can at least produce the feasible schedules. What the other schedulers that are known can produce is that; so let us, stop here, we will continue in the next class.

.