

Design Verification and Test of Digital VLSI Designs
Prof. Dr. Santosh Biswas
Prof. Dr. Jatindra Kumar Deka
Indian Institute of Technology, Guwahati

Module - 2
Scheduling, Allocation and Binding
Lecture - 3
Scheduling of Algorithms-2

So, welcome to the third lecture of module 2, that are the scheduling algorithms. So, in the last class, what we have discussed in the last class we have seen that there are several algorithms possible for scheduling problems. So, in the last two last lecture that is a one lecture before previous one. So, we have discussed the scheduling allocation in mining problem. So, in that 1, we discussed that the fast type in high level synthesis is actually scheduling process, in which case for each the operation we assign a control step.

So, there can be several constraints driven synthetics, I mean constant driven scheduling, like we can see that we can have a timestep given constant, like that is if you complete all the (()) 4 or 5 times steps. Then we have also discussed there can be some kind of resource given constraint. Like we can have 2 multiplex, 3 adders for 4 subs tractor, kind of a thing.

Now, there can also be a resource and a time constraint both or there can be unconstrained time schedule also. So, that was then we have defined the different type of scheduling problem. And then we said that the focused of the course main course this course, the main focus of this courses therefore having cad tools and algorithms which are automatically solve the problem first. Then in the last lecture, we have seen some of the algorithms like as soon as possible as late as possible, then first directed scheduling we have discussed this algorithm, which can automatically given sub constraints? They are all time constraint measuring schedules. If you give that certain time constraint then as soon as possible as late as possible or in the other case first directed scheduling, So these are the some of the automated tools or automated algorithms you can say, which will given a time constraint and operations to be schedule and a time and a time limit say 4 steps or 5 steps or k number of steps.

So, these algorithms will tell you that: within this time stamp if I want to schedule your stuff or schedule your operations, then what are the minimum numbers of resources

required. So if you try to minimize the number of the resources subjected to the subject to the constraint of time, as well as subject to the constraint of some dependency that is very obvious. So 1 thing I should maintain that even, if I do not mention this that this constraint a, constraint b so, I can say may not say but, if we remember this always there; that is actually data dependency constraint. That is 1 operation has to be done after that.

So, no were whatever you do you cannot schedule both the operations together? That is very important, or if you say that operations that I depends on the operation g a so, always operation g a has to be scheduled after I. you cannot schedule g with I or before I. So, that is actually derived dependency constraint. And even if I do not mention you met this is constraint always there.

So, based on time constraint; so, you have to schedule the resources in using the algorithm as soon as possible as late as possible or first directed scheduling. So, they will give some steps so the operation that the overall time required overall the resource requirement is minimized.

Then we have seen, as soon as possible, as late as possible or f, d, s, first directed scheduling all this drastic algorithms that is, they take much less time than the exponential over a time complexity. Over the time limits time constraints time required to solve the problem. Time complexity is much lower they will always not give much optimized solution? Their solution will be near optimal so, for some reasons we have the last lecture that fast as soon as possible gives a good result, compared to as late as possible for some cases as late as possible will give a better result than as soon as possible.

Then we have seen that first directed scheduling is nothing but, the hybrid of these 2 algorithms. It takes the advantages of both the and gives the better result. I mean, what you called a better result in most of the cases compared to this individual algorithm taken by 1 at a time?

But, in today's class we will see some more algorithms, and we will see 1 exact name and also, we will see 1 exact algorithm. But, this they will always keep the optimal solution but, there complexity will be extremely high, that is exponential or even more than the exponential complexity will be there.

Today we will discuss: 1 such algorithm that is the linear programming based algorithm. We always give very optimal or the optimal solution the best possible solution. In case of scheduling and then so that is very difficult problem to solve because, we know that 0, 1.

If it is a linear programming in the complete problem so, these are very difficult or the take long time to solve. So, which that is, why we go to the rustics. Which we already discussed in the last lecture, then also we will show you some examples; like in the last example, we have found out he first directed scheduling is a very good algorithm, in which it is giving the very optimal results? Where as soon as possible as late as possible were not giving? But, in the in the but, the end of today's lecture, we will also see some cases.

Where first directed scheduling is also not giving a very optimal solution? As there giving a optimal solution. So, in that case I these are linear programming, these are very complex problem, complex it gives a it takes a long time already it gives time to take. It is a very high time complexity so, the problem but, it still gives the most optimal solution in all cases this is in, actual or these are going to learn today, ok.

(Refer Slide Time: 04:49)

List Scheduling

- All the scheduling algorithms we discussed till now were heuristics based on time constants, in terms of number of control steps. Now we discuss another heuristic scheduling algorithm which is resource constrained— List Scheduling.
- Unlike ASAP, ALAP or FDS scheduling, which process operations individually in a fixed order, list scheduling handles each control step individually (in increasing order).
- List scheduling works by trying to schedule "maximum" number of operations in the control step, subject to resource constraints and data dependency.
- During the scheduling process, list scheduling uses a ready list (hence the name) to keep track of data-ready operations subject to data dependency.
- The ready list in a control step comprises those unscheduled operations that can be scheduled into the current control step without violating the data dependency.
- As long as there are operations in the ready list that meet the resource constraints, operations are chosen from that list and scheduled into the current control step.

So, let us start so, now as mentioned in the last class, we have learnt about the f, d, s as soon as possible as late as possible. They are time constraint based solution. Now in this now in this now we are going to see some kind of scheduling ,which some kind of

scheduling algorithms? Which can solve the problem? For, what we call resource constraint? It is actually called the list scheduling first. It is the list scheduling as the name says so.

What it says as the name list scheduling means. We have a list so we have beam. So, we have a list or a beam, so it says in a beam, as we have so many numbers of multiples, so many numbers of what do you say they are called dividers. So different, I mean different resources will be there. In a beam there will be lost this actually called a ready list.

For what do you mean by a ready list? Ready list this are the operations which can be scheduled right now, how can you say that the operations scheduled right? Now that is all the data on which they are dependent on or have already been scheduled like operation is dependent on inputs.

Then, you can schedule in the first step if an operation is dependent on data. I and data g operation I and operation g operation I and operation g has to be scheduled before the operation, we are talking about has to be scheduled. So, based on this data dependency constraint, you can find out these are the operations which can be scheduled right now.

Then actually compared is the ready list and, we have got beam of have got beam of hardware. So, we have to find out the, we have to try to schedule all the ready operations till now. If, you can do that using the beam. Since, we require the 3 multiplication operations to be scheduled now. Because, now they are ready to be scheduled but, we have only 2 multipliers in the bin then we have to use 2 and once the operation will be left we schedule in the next time slide. Now which 1 of them, that will be used some priority functional that we will see.

Now, that is basically the idea of resource scheduling in this. We given a actually we are basic idea is that we have a maximum number try to schedule. Maximum number of operation control step subject to resource constraints and data dependency. So, here data dependency is taken care in the. What do you call this? The list when we are preparing the list, we prepare the list based on the data dependency but, actually a bin which stores the maximum number of operations. So, based on that, we have to try to schedule as many operations as possible in the time.

(Refer Slide Time: 07:14)



List Scheduling

•If more than one operation from ready list can be scheduled in a control step, but violates resource constraints, then choice among the ready operations is made by a priority function.

One common priority function is “ $ALAP - ASAP$ ” (i.e., range where the operation can be scheduled). Operations with smaller ranges (i.e., smaller mobility) are given higher priority, since there are fewer possible control steps into which those operations can be scheduled, and since delaying them to a later control step would more likely increase the overall length of the schedule.


NPTel

As there a list because, we are taking a ready list into a account. So, we name list scheduling. Now, how long have to go as long as the operations in the ready list we have to go about including the time 1 by 1. So you have scheduling the stuff so we can understand that this is not a time constraint based scheduling it is a resource constraint based scheduling.

Because, there we have a bin of resources and we have to keep on scheduling the operations. Until you have schedule all the operations and subject to. Whatever hardware you having? You been so, that means; you may take a very ling number of time stamps also that is also possible in this case. Because, we are not having any we are thinking about the time constraint only, thinking about the resource constraint and we are saying that keep on scheduling all the elements from the ready list as much as possible subject to the availability in the bin.

And, you have to keep on doing it as long as the all the schedule all the operations schedules. So, you have to keep on using the time. this the basically a ready this using in a resource constraint bin schedule.

(Refer Slide Time: 08:00)

List Scheduling

• If more than one operation from ready list can be scheduled in a control step, but violates resource constraints, then choice among the ready operations is made by a priority function.

One common priority function is "ASAP-ASAP" (i.e., range where the operation can be scheduled). Operations with smaller ranges (i.e., smaller mobility) are given higher priority, since there are fewer possible control steps into which those operations can be scheduled, and since delaying them to a later control step would more likely increase the overall length of the schedule.

NPTEL

Bin
1

1 2 3

So, these also say that more than operations 1. Operation which can be schedule in a control? But, cannot do that because, of resource constraint in any 1 of them has to be scheduled. So that you can, I mean find out based out some priority function.

So, 1 what can what can be lot of priority function? So, what can be what the common types of priority function are? So, 1 very common type of priority function that is range in as soon as possible, as late as possible. So, you can say that say for example; this is a time stamp. So, 1 operation can be scheduled here. 1 operation can be schedule here 1 operation can be schedule here. So, this range is 3 so, you can be scheduled any 1 of them.

But, there can be saying 1 operation so, which has to be scheduled only here, that is the. I can say this can be scheduled any wards. when though this but, his 1 operation say this is so say 1, 2, 3 and 4 operations say which can be scheduled only in this cycle? there is no other choice for this ok.

Sorry ,this is also 1 ,this is also 1 and this is also 1. this is o 2 basically so, 1 can be scheduled in 3 steps 1, 2, 3 but, o 2 can be scheduled only in time stamp 1. So by as soon as possible as late as possible. by as soon as possible you will schedule to o 1 at step 1 and o 1 at as in as late as possible. you will schedule o 1 at but, in this case for o 2, you have to schedule o 2 in time stamp 1 in both as well as soon as possible as late as

possible. Because, there is no other option available. Now, if you have to say both of them are say adders.

Now, you say in a bin, you have only 1 step bin name having only 1 adder. now ,what will do so? you can schedule either 1 or either 2. Because, both of them say is only dependent on the primary input so both of them can be scheduled. So, in the ready list you will have both 1 and 2. Now, which 1 you will schedule? Will schedule any 1 of them? that is actually a priority function comes into play. So, if I know that there is only 1 bin so.

And say I mean say some other operation. So, over here that also means that, if it is I mean the range is not that much flexible here. So, they will order further dependencies for o 2. So, it is as soon as possible as late as possible. Its position is only becoming o 2. Because, of some of the stuff over. Here now you have the question whether to schedule 1 or either I schedule 2.

You can schedule either any 1 of them both. Because, both of them are ready now, the priority function is there if you have some range in as soon as possible as late as possible. That range 1 can be scheduled between this no problems. Then but, in o2 because, of some data dependency it has to be scheduled over by here. These range any better schedule o 2 here and go about it the modulation behind this.

Because, o 2 goes on lot of flexibility so, you can prefer it any other place but, o1 is actually hardcoded over here or this range is very less compared to o1. So, you better schedule it I mean give parental of scheduling this 1 over schedule o 2. So you give preference to schedule o 2 then compared to o 1 scheduling at time stamp 1. Now this 1 kind of a priority schedule so 1 cannot priority function so there are.

So, I mean operation is small ranges are given priority. Because, I means less number of I mean if we operate less flexibility is less probable. that you should be you can be able to force o 2 to this table, even if you are want to do that so, the time schedule will be increased like. If we schedule if we drop this 1 here all the other operations will be again shifted down. These are possibility that, what should happen is that?

So the whole time may increase but, basically what they are saying that a priority function is what priority is a if there is a if there is conflict (()) conflict if there is a multiple choice you are having at no you can schedule operation to only you can schedule any 1 of them. Because, of the resource constraint then you take the 1 which has a very which has a low flexibility because low flexibility means there is a very high probability that this scheduling with time stamp. Now, we have we have what all optimal result but, if there is a lack of flexibility here so better you can wait till the next step now this is 1 kind of a priority function ah.

(Refer Slide Time: 11:38)

List Scheduling

Algorithm 4: List Scheduling

Input: Operations Q , Maximum number of operators of type k k_{max} .

Output: Control step for each operations.

Steps

Prepare READYLIST (ready list for first step), which comprises all operations whose predecessors are input variables.

for each operation $o_i \in READYLIST$,

DO

BEGIN

Schedule all operations $o_i \in READYLIST$ in control step1. If there is violation in resource requirement (i.e., number of operations of type k in READYLIST, is more than k_{max}), then schedule according to priority function.

MPTEL
END

There can be many other priority function sometimes we think that dependency say 1 operation it controls n number of other operations like operation I it is actually say some operation I the operation j k l I x y z all are dependent on I.

So, let us schedule operation I to operation m on this nobody is dependent only 1 or 2 operations are dependent if you find there is 1 operation on which many other operation other operations are dependent. So, you schedule it now the priority to that compared to any other which is having less number of operations dependent. So, there can be a lot of priority functions to be redundancy I always told you there are heuristic based functions ah.

So, what you can do you can find out different types of optimal functions operative functions. Then: you can find out, which is giving the best? Actually in this you are having a resource constraint and you are trying to minimize the number of time stamps.

So, just find out this are the resource constraint which priority function is giving you a better solution in terms of number of time stamps. So, there is can be still you can find out different types of priority functions based on the requirements.

So, as I told you they are heuristic. So, it is not guarantee that you will get optimal solution but, it is highly possible that we are going to get a solution, which is optimal? Now let us see the list scheduling algorithm formal terms and then, we will go about for an example; so, what is scheduling operations? you have to take and maximum number of operators of this type.

So as I told this is actually this is actually a resource constraint base schedule, it is a resource based scheduling constraint. What is happening? So ,we are not giving any constraint on time there we are saying that; k_{max} that is of I mean k_{max} for each type of operations like adders multipliers subs tractor dividers etc.,

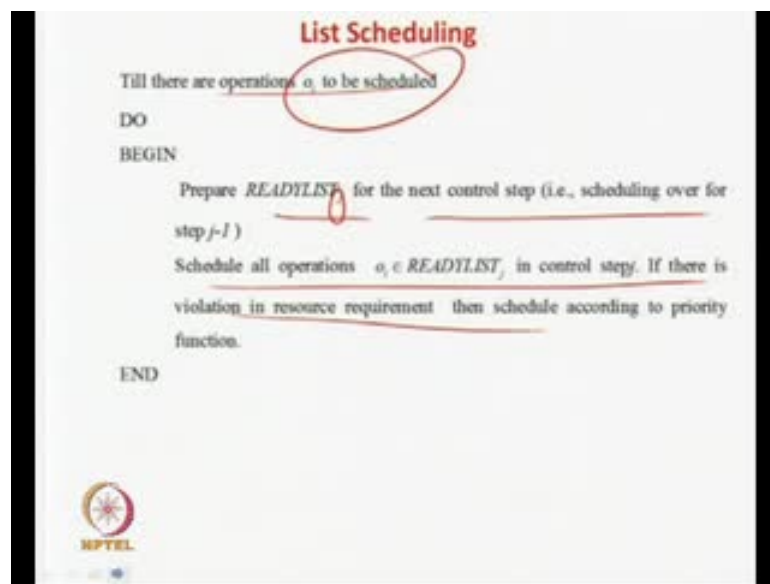
We have defined a maximum number for each 1 of that, we can say that the 2 multipliers, 2 dividers , 2 subs tractor, 1 adder kind of a thing and then you have 2 control step for all the operations. In the scheduling successful, as well as the failure, now what is this? what you have to do constraint to prepare ready list? So, what is that 1 ready list 1 means the entire all the variables. Which can be scheduling time stamp 1? Now, which operation say scheduling time stamp 1? if you think all operations?

Whose predecessors are input variables like, you have some operations like whose dependent only on the primary input all operations. Which are dependent? on the can be scheduled right now. So, this ready list 1 now what do you do? you take each operation from ready list 1 and try to schedule all elements of ready list in 1 control step 1.

If there is a if there is a no violation in the resource requirement. If, there is a violation in the resource constraint then you have to schedule any 1 of the using a priority function. But, in case no resource constraint then a based on this k_{max} value. Then, what you can do? you can schedule all the operations in ready list 1 and that is d_1 but, if there is if you

find out only 1 of them or 2 of them can be scheduled. Then, which 2 of them can be scheduled right? Now, then you have schedule according to priority functions. that is, what thing you have to do in the first list based on the first list? we have scheduled whatever is based on the requirements or based on your prior constraint on the bin that is maximum number of hardware available you schedule some of the operations see time stamp 1.

(Refer Slide Time: 14:31)



Then, what you do? Then, you have to repeat this till all the operations since time stamp 1 is scheduled is obvious. Now, what you do now? you go for time stamp preparation of ready list 2 dot dot dot you have prepare a ready list as long as all the operations are scheduled.

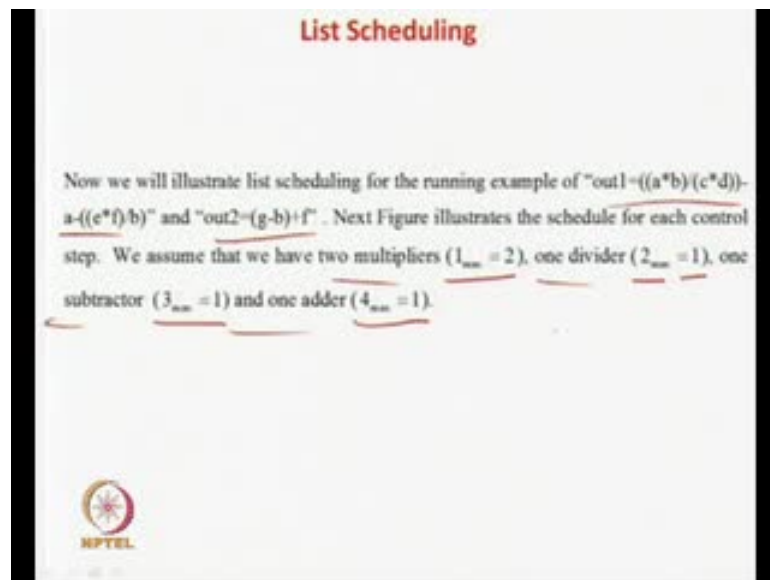
So, what you do in the ready list? 2 in ready list 2 what do you learn all the operations? which were dependent all the variables? Are, all the operations which are now been scheduling in time stamp 1. Now, there will be variable there will be some operation which were dependent on the operations which are no scheduled in the time stamp 1.

Now, you can schedule those elements which were dependent on the operations scheduling time stamp 1 now they gave ready also, me of the operations which you would not schedule in the time stamp 1. Because, of resource constraint.

We also get in the ready list for the time stamp to now you have repeat this based on the some constraint of hardware available we schedule whatever we have available in the resource step 2. So, whatever available in the ready list 2?

Usually, some of them again we will not be able to schedule resource constraint. Then, we have to do that to third list and also the third list, those elements will come in picture. Which were dependent on data dependent on the elements? Which are now schedule in the time stamp 2? So, we have to keep on doing it for all lists. we have to keep on doing till all the operations have been scheduled in general, we say ready list I for the next control step.

(Refer Slide Time: 15:54)



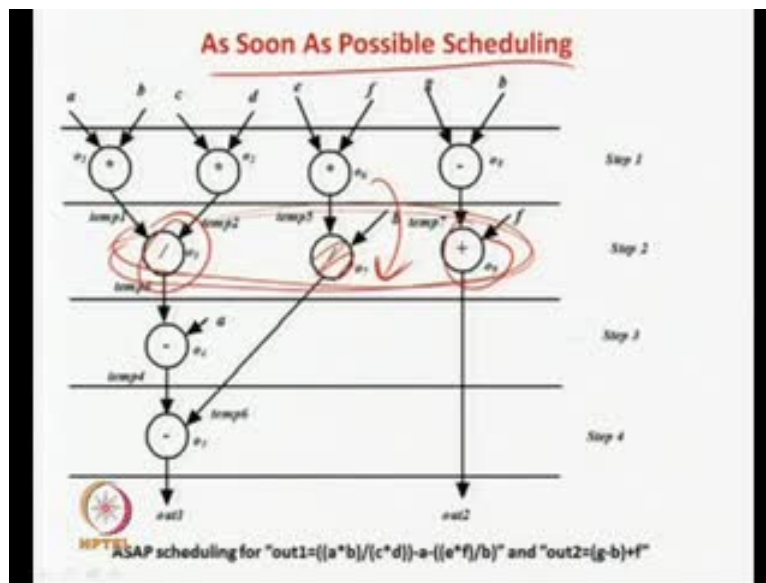
Schedule all operation in this. If there is no violation in the resource requirement and you keep on doing repeat this till all the operations have been scheduled. this is the idea.

So, is this very simple you actually prepare a ready list 1 schedule as many as possible based on the resource constraint whatever is remaining you make it in the resource list to as many as. Whatever else you have to put in the resource? list 2 all the operations. Which were data dependent on the operation? which are now scheduled in the time stamp 1? will get into time stamp 2 and you have to keep on repeating it as long as you have scheduled all the operations .

So let us take our on example; that is this 1 this 1 we are talking about and this called this is our running example in the last 2 lines just so will try to schedule them already e have scheduled this operation out 1 and out operations in. if, f b s as soon as possible as late as possible.

Now we will try to using ready list scheduling. Now, according in the previous case we have given a time constraint of 4 and we try to do that. Now we are here to give cannot given a time constraint this is a resource constraint scheduling. So, we have to give a resource constraint. So, in this case we are saying that 1 max, 1 is to multipliers ,2 multiplier, 2 is divide, we have 1 divided, 1 subtracted type 3 is subtracted and 1 adder that is 4 is 1 and this is the thing. So, we are try to try to go for a list schedule. So what you remember e have to multipliers plus 1 divider 1 sub tractor and 1 adder.

(Refer Slide Time: 17:00)



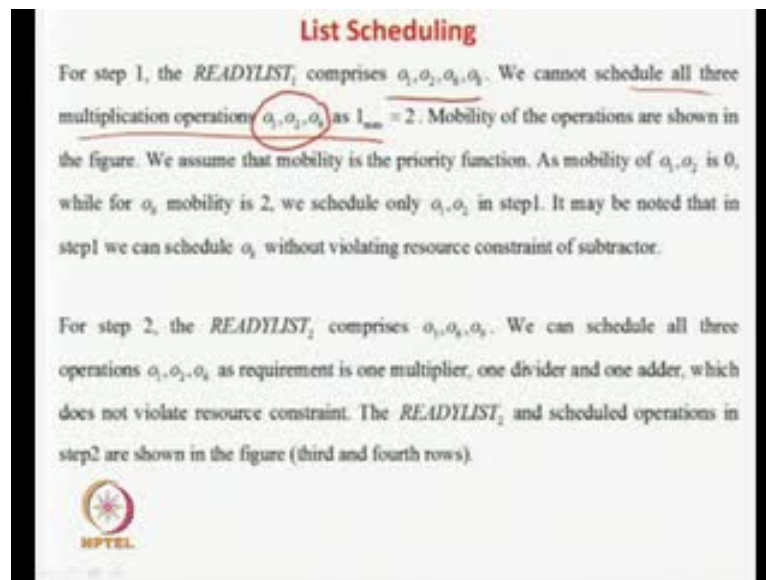
Ok, so this actually is as soon as possible scheduling for this time stamp may given it as a view. So, that you can recall so, you can remember o 1 o 2 o 6 and o 8 can be scheduling time stamp 1 .because, they are dependent on the primary input. So, this 1 is actually belonging to ready list 1.

Now, if you can say all of them. Now, in the first time stamp and this 1 will become a ready list 2 but, for some reason say we cannot schedule operation 6 in a time stamp 1. Actually this die will become too ready list number2 and the ready list number of 2 will

comprise all the it will not comprise all the. Because, the dependency is remaining it will comprise this it will not all this has been scheduled.

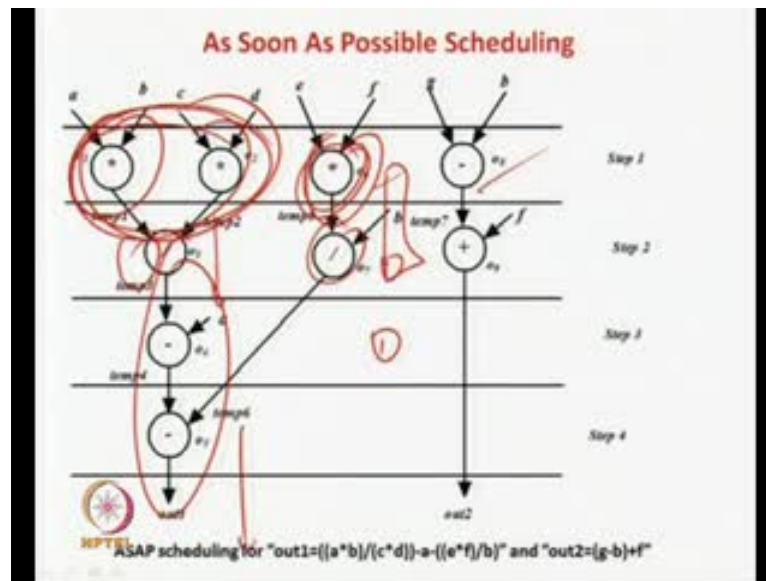
This is also comprised in schedule this time. we join from the ready list 2 ready list 2 because of some time constraint it was not avail for schedule because of this resource constraint o, sorry o, 6, o, 6 was not been able schedule in time stamp 1 so it will join time stamp 2 and so on so we have to keep doing it over everything is scheduled or not. So, this is just an action I mean so there about explanation example for the time being remember the o ,1, o to o, 6 and o 8 can be possible to schedule in time stamp. 1 or the in the ready list 1 because they are already dependent on primary input now let us see.

(Refer Slide Time: 18:08)



So, u in ready list 1 present o 1 o 2 o 6 and o 8 all this can be scheduled no we cannot all the multiplication operation because 1 x is 2 so o 2 o 1 o 2 o 6 only 2 of them can be scheduled now which 2 we have to schedule. So, again let us go back to a priority function so you can see this.

(Refer Slide Time: 18:26)



That if, you are going to as soon as possible as late as possible stuff you can see that if, you remember the last lecture these 2 multipliers are fix their range are very less very less mobility .But, this guy and they can be scheduled here. also this is dependent this is the stuff so this guy having a mobility of 2 this can be scheduled here that is up possible.

But, these things get over if, you try to push these guys down so time stamp will definitely start increasing so according to priority. we should schedule over o 1 and o 2 and they differ o 6 because o 6 can be scheduled uses of flexibility. So, you can push it over here and so this is the case but, now should have been the case assumes that.


So, we have to take any 2 of them and do scheduling so for example, we may referred this guy so referred this guy what is going to happen is that so this guy will come here and all this thing will come 1 step down. So, you will have addition increment in the time number of time stamp. better we go for the multi function we take so in this case is flexibility on o 6 so you will differ o 6 over here so that is what is say.

(Refer Slide Time: 19:32)

List Scheduling

For step 1, the $READYLIST_1$ comprises o_1, o_2, o_3 . We cannot schedule all three multiplication operations o_1, o_2, o_3 as $l_{mul} = 2$. Mobility of the operations are shown in the figure. We assume that mobility is the priority function. As mobility of o_1, o_2 is 0, while for o_3 mobility is 2 we schedule only o_1, o_2 in step1. It may be noted that in step1 we can schedule o_3 without violating resource constraint of subtractor.

For step 2, the $READYLIST_2$ comprises o_1, o_3, o_4 . We can schedule all three operations o_1, o_3, o_4 as requirement is one multiplier, one divider and one adder, which does not violate resource constraint. The $READYLIST_2$ and scheduled operations in step2 are shown in the figure (third and fourth rows).



(Refer Slide Time: 19:50)



So, the mobility shown in this figure: so, we are discussed so o 1 and o 2 0 while this 1 mobility is 2 so this schedule o 1 and o 2 time stamp 1 we will see later I mean what we can do about it. So, 1 o 1 o 2 scheduled over here and o 6 is therefore, so in ready list what will come in ready list 2 you will get o 3 because this processes a schedule. now o 7, will not come in ready list 2. Because, this guy is not yet schedule but, o 8 can be scheduled in time stamp 1. Because, we have already 1 multiplier also your resource is 1 multiplier, 2 multiplier, if you remember 2 multiplier, 1 divider, 1 sub tractor and 1 adder.

So, you could use 1 sub tractor for here so you can easily schedule in the first time stamp that is not a problem.

Now in the next time stamp or in the list 2 you will have o 3. Because it produces a schedule o 9 will be there. Because, it produces a reschedule and o 7 will not be able to schedule here. Because, it produces a not yet scheduled now this guy will join the ready list 2.

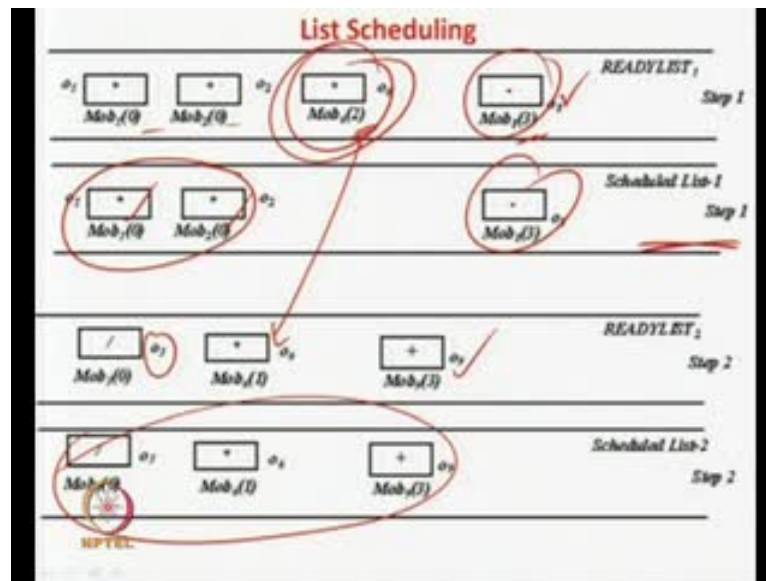
In ready list 2 you will have o 3 o 6 and o 9 now here in schedule all of them. Because, we require 1 divide 1 multiplier and 1 adder. So you can easily schedule them and similarly, we can go about the go about doing it so let us look at this example at pictorial manner.

(Refer Slide Time: 20:42)



So, what is the ready list do ready list 2. We have o 1, o 2, o 6 ,o 8. So, here mobility is 2 here mobility is 3. So, here mobility is 0 means we cannot move this guys from here. and here compared to as soon as possible as late as possible. Here, the mobility is 3 because you can put it here you can put it here. So, 1, 2, 3 this guy can be here and here so this guy is so, 1 ,2 ,3, 1, 2, 3 kind of a thing.

(Refer Slide Time: 21:05).

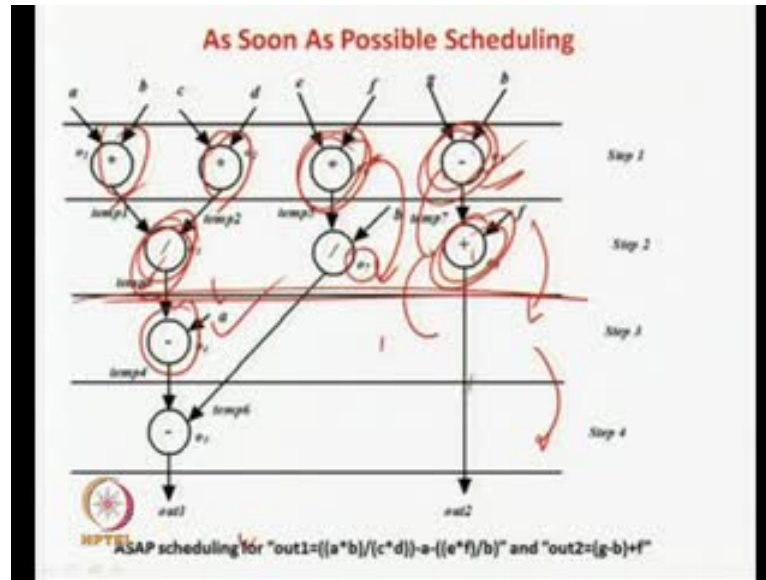


And now So, you are hiding mobility, 3 mobility, 2 mobility, 0 and mobility 0. Now, if you remember already we have only 2 multipliers so we have to better schedule this 2 because, of the mobility 0 we differ it and this 1 subtractor.

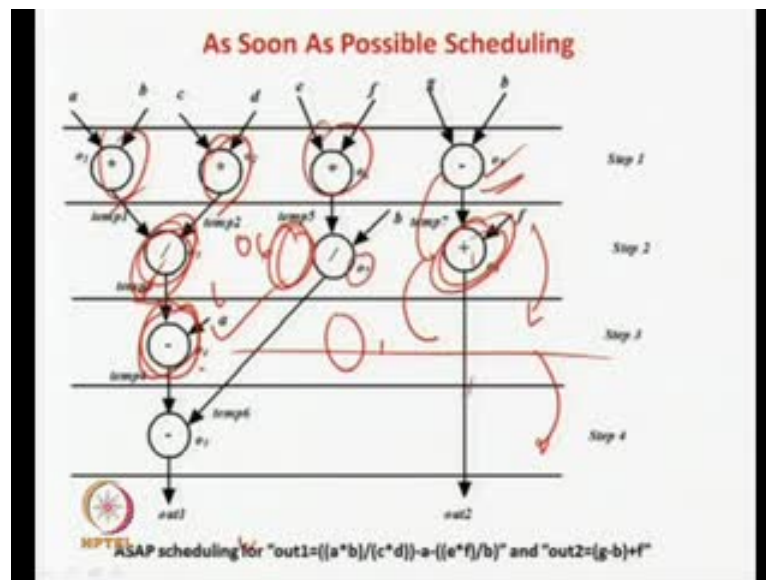
So, you should in time stamp 1 we have scheduled all this 3 operations excepting this. 1 now in the ready list this guy is going to join and o 3 will come. Because, it produces a, o, 1 and o, 2 which have been scheduled coming over here. And o 9 is also here in the ready list 2 because it produces a o 8 is schedule. Now you have a here. We require 1 divider 1 multiplier and 1 adder so it is for your resource constraint. So, you can schedule these 3 guys over here and it is your ready schedule list 2.

Now, what happens now what do you get in ready list 4 in ready list 4. If, you remember you see over here. So, till now we have scheduled up to this already has been scheduled. So, in ready list is for this 1 is to be scheduled. Because, it produces a 10 c and a has been scheduled a is primary input and it has already been scheduled.

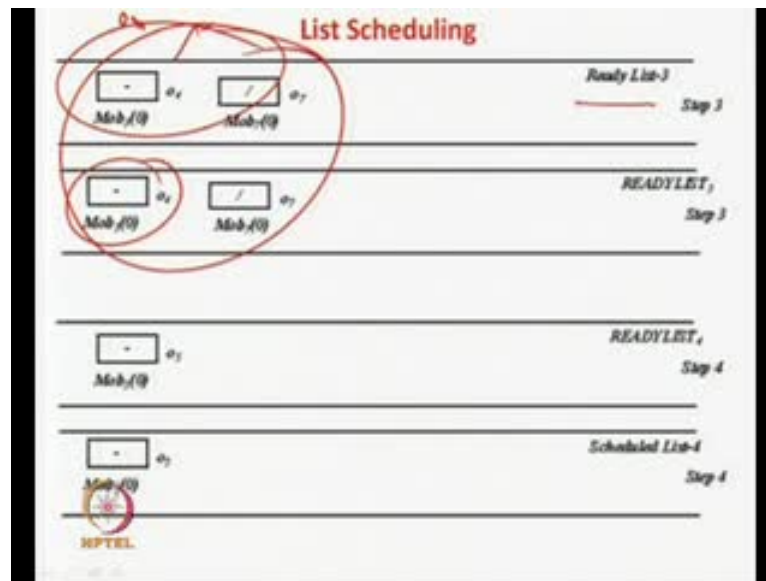
(Refer Slide Time: 21:45)



(Refer Slide Time: 22:12)



(Refer Slide Time: 22:42)



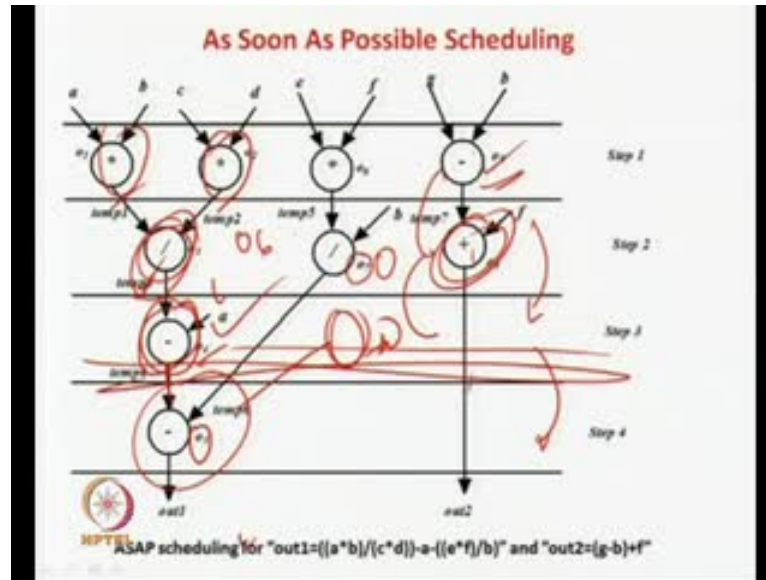
Ok, so that actually we will come over time stamp 4. So, it can see sorry there are 2 operations and this 1 is there and also this guy has been scheduled here.

So, if you see that we could not schedule p_6 over here. Now, we have scheduled p_6 over here has been scheduled p_6 scheduled over here. This guy has been scheduled over here. So, in time stamp 3 that is in ready list 3.

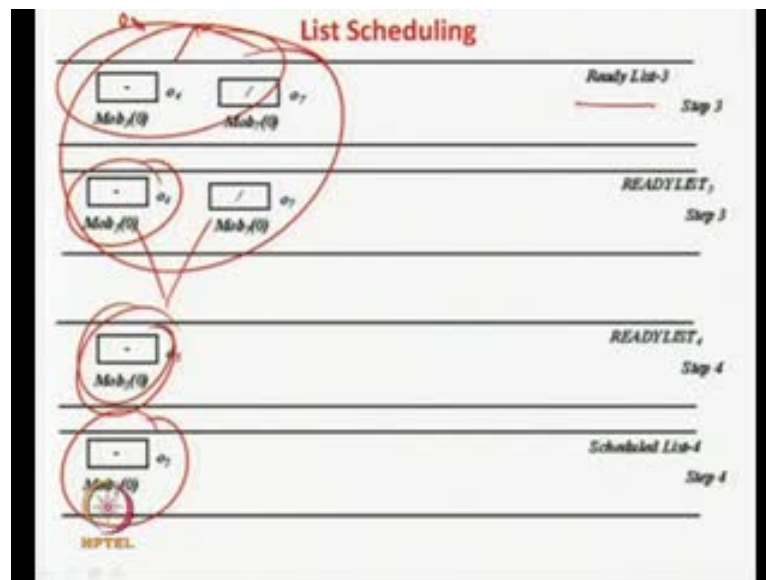
We have this guy because this produces a temporary schedule as well as there is primary input and for p_7 this produces. A, p_6 So, that has also been scheduled in the time stamp 2. So, in this 3 we have p_4 and p_7 so actually this is what is been shown over here. So, in this case we have p_4 whose precedence are sorry, we have this is your ready list 3 So, p_4 whose precedence was p_3 so is specialty is d_1 and a is also it is p_1 .

For this p_3 and depends on this guy can be scheduled over here. and p_7 I d dependent on p_6 so p_6 actually controls this depend on p_7 now p_6 is already scheduled in the time stamp 2 so p_7 can be scheduled over here. I mean it comes in the ready list. So, in other words the precedence are p_4 the precedence are p_7 as already been scheduled in time stamp 2 so, this 2 guys will come in ready list 3. Now, we have 1 subtracted 1 divider we have that so both of them will be scheduled in the time stamp 3.

(Refer Slide Time: 23:29)



(Refer Slide Time: 24:28)



So, now we have completed up to this much this 1 up to all the operations have been scheduled this 2 guys has been scheduled here. and here so they are not scheduled over here. this 2 o 6 and o 7 are now here. and here so, this what have been came is and now only thing is remaining that is what you o 5 now we can be in the ready list number 4 because it is dependent on o 7.

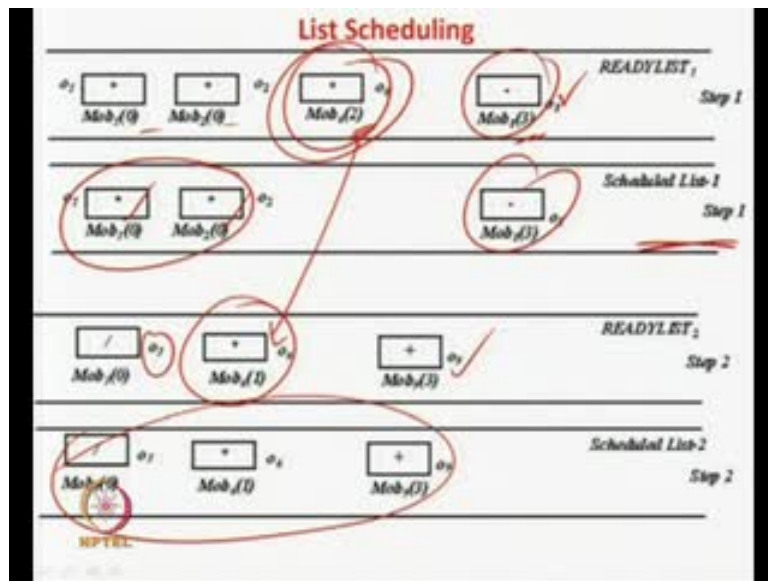
Which is now scheduled in o 3? sorry in the time stamp 3 so, o 5 is dependent on o 7 is now already scheduled in tie stamp 3 as well as also dependent on time stamp 4 tells

which is the output of σ_4 and σ_7 has been scheduled in sorry σ_4 has been sorry σ_4 and σ_7 .

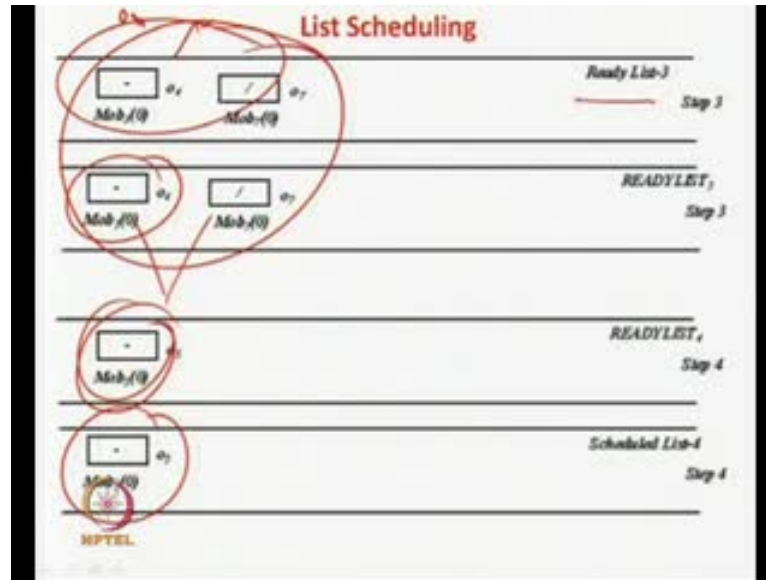
Which are actually controlling σ_5 they have been scheduled in time stamp 3 we will so σ_5 so σ_5 will actually be now in ready list number 4. So, other words the precedence are σ_5 that is σ_4 and σ_7 has been scheduled in time stamp 3. So, σ_5 will be joining the ready list 5 now it's you can see that ready 4 actually σ_5 that is actually dependent on σ_7 and σ_4 which is already been scheduled in the time stamp 3.

So, this will be coming in the ready σ_5 will be coming ready list 4. And you can see that it requires sub tractor to be implemented. So, this guy will be scheduled over here and finally, what do you get you get a schedule.

(Refer Slide Time: 24:46)



(Refer Slide Time: 24:56)




In which case in first step you schedule o 1 o 2 and 3 in second schedule o 3 o 6 o 9 and in the third time stamp schedule o 3 and o 7 finally, you schedule o 5 you can o 4 kind of a thing so this is a list schedule so you require a 4 time stamps and your schedule is d1. Basically, so here actually you see that I mean in this case you we have stamp 4 because of constraint was.

(Refer Slide Time: 25:13)

List Scheduling

For step 1, the READYLIST₁ comprises o_1, o_2, o_3 . We cannot schedule all three multiplication operation o_1, o_2, o_3 as $1_{mul} = 2$. Mobility of the operations are shown in the figure. We assume that mobility is the priority function. As mobility of o_1, o_2 is 0, while for o_3 mobility is 2, we schedule only o_1, o_2 in step1. It may be noted that in step1 we can schedule o_3 without violating resource constraint of subtractor.

For step 2, the READYLIST₂ comprises o_3, o_4, o_5 . We can schedule all three operations o_1, o_2, o_4 as requirement is one multiplier, one divider and one adder, which does not violate resource constraint. The READYLIST₂ and scheduled operations in step2 are shown in the figure (third and fourth rows).



(Refer Slide Time: 25:19)

The slide is titled "List Scheduling" in red. The text on the slide reads: "Now we will illustrate list scheduling for the running example of 'out1=((a*b)(c*d))-a-((e*f)b)' and 'out2=(g-b)+f' . Next Figure illustrates the schedule for each control step. We assume that we have two multipliers (1_{mul} = 2), one divider (2_{div} = 1), one subtractor (3_{sub} = 1) and one adder (4_{add} = 1)." There are red handwritten annotations: a large circle around the text, a diagonal line with "FDS" written above it and "LIST" written below it, and a large red arrow pointing from the text towards the bottom right. The NPTEL logo is visible in the bottom left corner.


2 multipliers 1 adder 1 divider than 1 subtractor kind of a thing you can also try like. If, you can look this as 1 then you can find out the you can much more number of timestamp to solve the problem. So, this constraint was very similar this resource constraint was very similar to what required to find out that in case of f, d, s. So, you think that in f, d, s we started with 4 time stamps and finally, we said that we require this many number of staff and now in the case of a list.

We are choosing we started with this resource so after that and we find out that we require 4 stamps so we in directly I mean 1 case we started where resource constraint and we landed in a time and the other case we started with a time constraint and we ended into a number of resources. So that both of them are equivalent this is not a very important thing to be noted but, the idea here is that I with this many number of we can solve in 4 amount of time and in 4 amount of time or 4 times you can solve using this many amount of resources that is what is the idea.

(Refer Slide Time: 26:13)

Integer Linear Programming based Scheduling

- Integer Linear Programming (ILP) have been used to solve a wide range of constraint based optimization problems
- ILP formulation for optimally solving the synthesis problem.
- The biggest advantage of using ILP is the quality of the solution; unlike the heuristics based scheduling algorithms, described earlier, an ILP solver is guaranteed to find an optimal schedule from these formulations. However, this guarantee of quality comes at a price — ILPs cannot, in general, be solved in polynomial time. Thus, the tradeoff is between the guarantee of solution quality and a guarantee of quickly finding a solution.

 NPTEL

But, we can try at home or you can try on yourself by doing the multiply equal to 1 and you find that number of steps required are much higher. So, 1 you our constraint is basically that is based on resources. So, we have a set of resources and any timestamp we schedule as much as such possible based on the availability of resources.

If, you find out the is not there and you have differ and how do you differ which 1 you differ is based on some priority function? This case our priority function was range you can take any other priority function like how many operations it's been controlling and so on therefore, so till now whatever the algorithm.

We discussed like as soon as possible as late as possible. first directed and then this 1 for this 1 will be scheduling they are heuristic based now we will see the exact algorithm. Which is actually called integer programming based scheduling? In which case you are always guaranteed to test are possible solutions? In this case of some example, we get some good example for example, we don't get an optimal solution, therefore, in the session today we will see 1 example in which case f d s will not give optimum solution.

But, in case of I l p all always going to get a optimal solution that is a guarantee but, now what is the problem in this step you know that integer linear programming is a complete problem. So, this time complexity will be exp1ntial in a number of time stamps. Number

of operations also for so you can assume that. If, I have hundred time stamps the complexity will be to the hundred.

If, I around 5 hundred operations the complexity will be around 2 to the power of 5 hundred plus. So, this a very difficult problem we solve this in a feasible time in a computer. So, that is why we always go for whatever heuristic we have learnt but, again I have told you that formally know that scheduling problem is a very difficult problem.

So, I v e so what we will do is in this case. We try to map the scheduling problem to a well known problem which is empty hard kind of stuff.

So, in this case, what we will do, we try to show that a scheduling problem can be mapped to 0 1 integer programs which is known that it cannot be solved in programming time, which proves that there cannot be any polynomial type solution to integrate linear as there is no polynomial types. 0 1 i l p so, you cannot develop a very good or you cannot develop a polynomial time solution for the scheduling problem, therefore, you have to save time and you have to give a reasonably good answer, you have to go for heuristics. So, all the algorithms we learnt till now are heuristics and same people can develop better and better heuristics.

Which can serve better? So, what is the job of the heuristic heuristics job is run in a reasonable amount of time and we have to give a solution which is as close as to the optimal as possible? So, there is a what is a chase amount so we always try to find a better and better priority function.

So, we can find better and better problem in what heuristic algorithms which will take less time to run and yet a reasonable amount of time to run and it give you a solution is very close to the optimal solution and sure be like that heuristic is for 1 class of an algorithm.

If, it is not working good for another class of it should not be the case and is when you should try to develop, a heuristic covers a wide range of circuits. It should give a reasonable amount of time it should give a solution which is very close to the optimal that is the basic b1 of developing a heuristic algorithm.

Now let's come to a integer linear programming based stuff so in this case is actually well known to be sort of is to be solved for a very wide range of problems. Which are based on optimal constraint based optimization?

I will not only let to V, L, S I it can be led to products, biotechnology can be related to engineering physics any size domain of engineering or whatever. You want or business or finance economy. Very resources and some constraints or wherever some constraints and you have to decide the constraint and reach to some optimal value.

There you can use engineering programming to solve of this stuff so we see how integer linear programming for optimally solving the synthesis problem.

And high level synthesis problem in the scheduling step. So, it is the biggest advantage of I l p of quality of solution so unlike heuristics bound to get the best possible solution but, the time taken is of polynomial because I told you I l p is known to be there is known.

Well known there is known that till now problem now till now you do not whether there exists a polynomial problem solution or I l p or any of those any type of problems. So, that therefore, we know that there is no well known or very difficult, almost impossible. If, you think since a open problem in computer science., so that p is equal to n p or p not equal to n we say but, as of now still nobody knows a polynomial type of algorithm complete problems.

So, you can think that is an almost impossible and as we can now, easy or less complex solution or less complex type solution for the 0 1 I l p and hence, the scheduling problem. So, there is a actually a field of, how fast the exits solution? How close you want to go to optimality.

(Refer Slide Time: 31:03)

Integer Linear Programming based Scheduling

•Now we will formulate the scheduling problem as ILP. Unlike the discussion on other scheduling cases (above), we will not give a generalized algorithm to formulate an ILP for a scheduling problem. However, we will consider the running example and formulate ILP for the same and the discussion will give a generalized idea of the procedure of such a formulation.

•In the ILP for scheduling, we have four sets of equations namely,

- (i) to capture the range ([ASAP-ALAP]) in which an operation can be scheduled,
- (ii) requirement that there is no violation of resource constraints,
- (iii) data dependency and
- (iv) optimize the resource requirements.

The slide features a red scribble over the list items and a small diagram of a rectangular box with internal lines and a triangle, possibly representing a task or resource allocation. The NPTEL logo is visible in the bottom left corner.

Ok, now we will formulate the whole problem whole integer linear or whole. What do you call this or scheduling problem into integer linear programming? But, unlike our previous discussion where you have given a general algorithm? in this case, We are not going to give a general algorithm, but, let us take an example and then we will try to see.

How a problem how are scheduling problem can be map to 0 1, I, l, p we are not going for a general algorithm because, there are very well known tools will actually solve this i l p and no 1 actually uses integer linear programming tool to solve this scheduling problem.

Because, of the complexity, so telling your very general and complex mathematical algorithm is not at all purpose in this course. Because, you just want to focus. So, that you can land how you can map a general problem to a empty complete problem polynomial based problem. So, say for example, you have given problem x now you are asked.

How can you solve this problem? Or can I develop a polynomial type of a problem for this, or can I or is this not possible for polynomial type problem for this. How do people solve the problem? Is not like being that I have tried for 1 hundred days and I could not find the polynomial type algorithm. So, there is no polynomial type algorithm for this now. what will you take an argument? So, how do you know that you have shown that I

proved formally at this problem is equivalent to a very well known integer linear programming kind of a staff.

It is very well known to 1 of the well known empty complete problems. So, as there is no well known polynomial type solution to this. So, you know there cannot be not any polynomial type solution, for my problem also. what do we do? We try to map our problem to a very well known we try to map our problem to a very well known.

What do you call empty complete problem? Empty, complete problem well known, empty complete problem. And we can prove it so but, we not good or what do we say that, it is not very I mean realistic in this course to formulate the algorithm you have to do that because, this is starting foundation of computer science so better. What do we do? Is that will just show how we can or scheduling problem is the 1 I I p and then prove that they are difficult problem to solve or it is a it takes a long time to solve. So, what do you do with I I p. We have to capture some constraints and we have to and we have to optimize some stuff so let's see.

So, what do we do so? we will try to optimize the resources that is what the thing so let us take it as resource constraint b s I I p also you can formulate for time constraint based there also can be there so let us see how it will do so? What do we do will capture the range? In which the operation can be scheduled?

We first we will say that what are the time say, we are some kind of a time this then actually ,we are trying to find out what is the flexibility of the operation. So, in what range you can schedule, your operation next we see that there is no resource any no resource no violation of any resource constraint. That is our resources are constrained and obviously there are depending. Is mainly to satisfy that if resource I operation I is dependent of say j. So, I operation j is an operation I am dependent on operation say j so, j as to be scheduled before I that data dependency has to be matched and then you optimize the also in this case.

You can say that that is there is no violation of resource constraint so you can, I optimize this resource constraint. So what is the these 2 say so actually you have to say that I have 10 adders time multipliers 10 sub tractor kind of thing. So you should not violate this.

In then what do you optimize? So you take as less a resource constraint as possible. So, that is the idea so, let me just tell you in some errand of the stating it. what do you say given an equation? Or how do you formulate the problem? So we first say that how to represent the what is the mobility of the operations like we take, we first take a we take first given a we generally take a time constraint type say. I mean we are actually formulating the time constraint based stuff.

So we say that there is 4 types 5 types what is the equivalent at the first drop whatever you have to do you have to I mean sorry I mean.

Whatever you have to do, you have to I mean see I told you like this is actually data means resource constraint based I am in constraint based. See actually it is not that so what will be landing in this case, is our it is a time constraint scheduling or scheduling will be time constraint and what will be optimize will be the optimizing resource. So initially sorry, for what I have told you, so in this I I L P the time will be fixed that is your time is constraint. So this constraint schedule and what you are going to optimize, you are going to optimize the resources, you are going to take as minimum resources as possible that is like a a sap as soon as possible as late as possible or in the other words or F D S kind of a thing. So first you fix up your time and then you find out the range. So how what is the flexibility? So this things we have represent using integer linear programming so, that we will see how you can do.

Next you see that you give some resource constraints. Now, what is a resource kind of thing say 5, 5, 5 multiples tell you can give a high amount of resources also, that is not at all a problem. So, there is actually we because we are not you can also say that our resources are very high 10 adders 10 multiples sub tractors all those things. You can give this is a lot of relight constraint.

You just give some kind of resource constraint using the equations will see that, and you have to obviously represent the data dependency that is I is dependent on j. So, I cannot be scheduled before the other kind of a thing. So, I is dependent on j. So, I cannot be scheduled before a j kind of a thing that is how and you will again represent the optimize the resource constraint .

So, what is this so here you are saying that I by 10 multiples of 10 subtraction of some numbers you give and then, what you do automatically this? I I L P will actually take as minimum resources as possible given this time period that is what we are going to do. So this is the some constraint you have to fix because otherwise I mean we may not land mean we may not able to formulate the I I L P. So, you can also give a high amount high number of resources that is not a problem.

(Refer Slide Time: 36:58)

Range of scheduling

Let $\alpha_{i,j}$ denote the scheduling of operation α_i in step j . $\alpha_{i,j}$ is a variable for the ILP.

For scheduling we consider 0-1 ILP, where, in the solution we can have only 0 and 1 values of the variables. In the running example, operation α_1 must be scheduled in step1.

The ILP equation capturing this fact is $\alpha_{1,1} = 1$, which implies that for the ILP solution, variable $\alpha_{1,1}$ can only have the solution as 1.

However, for operation α_2 the equation is $\alpha_{2,1} + \alpha_{2,2} = 1$, which captures the fact that α_2 can be scheduled in step1 or step2. In the ILP solution either, $\alpha_{2,1}$ will have the value 1 and $\alpha_{2,2} = 0$ (implying that α_2 is scheduled in step1) or $\alpha_{2,2}$ will have the value 1 and $\alpha_{2,1} = 0$ (implying that α_2 is scheduled in step2).

Automatically we will I I L P will optimize the number of resources that are required to be taken. So, that is the idea so if you can remember as soon as possible. We never thought about the resources what we have d1 we have taken time to be 4 or 5 and ,we have tried to use as many as resources as possible that is actually I I L P sorry , that is equal to as soon as soon as possible as late as possible or F D S so given a time constraint we are not putting any kind of resource constraint but, automatically the algorithm is trying to use as minimum resources as possible that is what being d1.


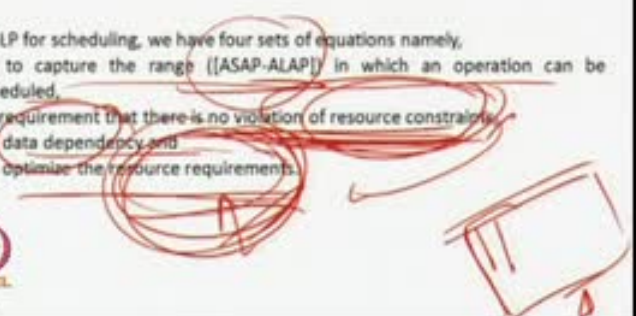
(Refer Slide Time: 37:22)

Integer Linear Programming based Scheduling

• Now we will formulate the scheduling problem as ILP. Unlike the discussion on other scheduling cases (above), we will not give a generalized algorithm to formulate an ILP for a scheduling problem. However, we will consider the running example and formulate ILP for the same and the discussion will give a generalized idea of the procedure of such a formulation.

• In the ILP for scheduling, we have four sets of equations namely,

- (i) to capture the range ([ASAP-ALAP]) in which an operation can be scheduled,
- (ii) requirement that there is no violation of resource constraints,
- (iii) data dependency and
- (iv) optimize the resource requirements.



(Refer Slide Time: 37:44)


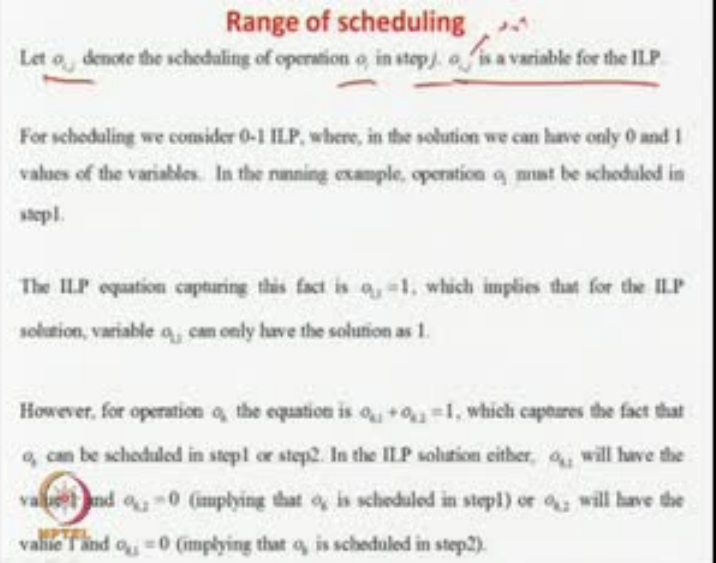
Range of scheduling

Let α_{ij} denote the scheduling of operation α_i in step j . α_{ij} is a variable for the ILP.

For scheduling we consider 0-1 ILP, where, in the solution we can have only 0 and 1 values of the variables. In the running example, operation α_3 must be scheduled in step1.

The ILP equation capturing this fact is $\alpha_{31} = 1$, which implies that for the ILP solution, variable α_{31} can only have the solution as 1.

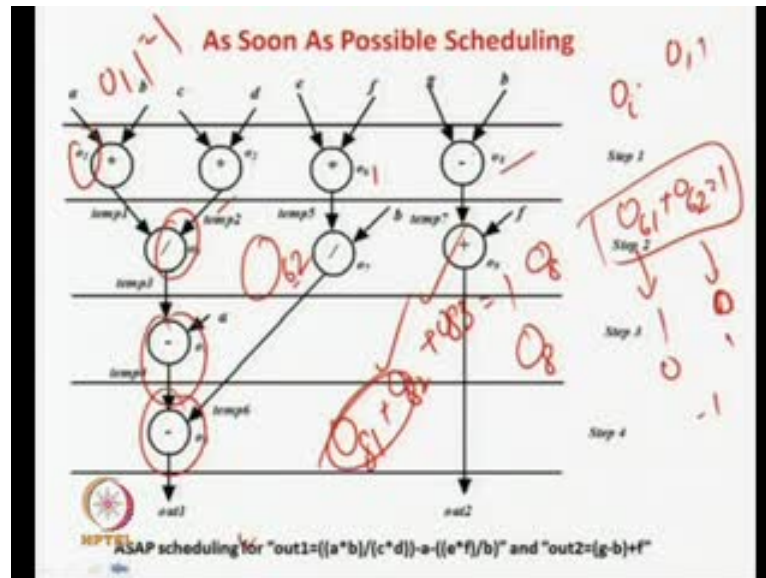
However, for operation α_4 the equation is $\alpha_{41} + \alpha_{42} = 1$, which captures the fact that α_4 can be scheduled in step1 or step2. In the ILP solution either, α_{41} will have the value 1 and $\alpha_{42} = 0$ (implying that α_4 is scheduled in step1) or α_{42} will have the value 1 and $\alpha_{41} = 0$ (implying that α_4 is scheduled in step2).



So, in this case also we were giving some resources constraint, we that is, we can put into a relaxed manner also and if you make this 1 very tight. Then you can think that, you can become resource time constraint scheduling it will come into picture, but actually we first generally in case of online time base stuff time base scheduling were time based scheduling will we can just even wide range of resources and then will try see that as minimum resources as taken so it will automatically optimize.

So, what is the formalism will be using so OIJ that means scheduling operation I means step j so this is actually a variable in ILP it is a 0 1 ILP so it can take the value only 0 and 1. So, for scheduling first we have to specify the range s in 0 1 ILP. This is how to go about it so in this case if you see,

(Refer Slide Time: 38:12)



Or stuff sorry same L result will be taking. So ,this you see this is same algorithm we have taken. So, for this 1 we are already landing. So, this is the 1 remember that this is the same example; we will be taking. So, first we have to represent some excuse me from constraints.

Since, 0 1 ILP so, first considering the range. Now, you can see that in this case we have taken 4 time constraint base schedule. So, you know that o 1 cannot move, o 2 cannot move, o 3 cannot move o 4, cannot move and o 5 cannot move that is the basic problem this 1 can move 2 this. 1 can move 3 so, that you have to represent by some equation and not only the labels of 0 1 ILP OIJ that means, what O I means higher the operation in step j so, if you write o 1 1 equal to 1 then, what it says? it says that variable o 1 1 has to be a 1 in the ILP. if, you write o 2 1 equal to 1 that means it says that variable o 2 1 has to be 1 in the 0 1 LP what does it imply it implies that o operation 1 has to be scheduled in time step 1 then that is actually 1 that means op operation 1 must be scheduled only in time step.

1 and nowhere else similarly, if you write $x_{21} = 1$ it says that operation 2 has to be scheduled in time step 1. and that is mandatory because, we say that $x_{21} = 1$ now let us just see for this 1 then it will make you stuff clear. we write $x_{61} + x_{62} = 1$ sorry $x_{61} = 1$ what does it mean it means that x_{61} can be scheduled over here or $x_{62} = 1$.

That means, x_{61} we scheduled x_{62} is scheduled over 1. Then, this is equal to 1 and if, x_{61} is scheduled into then x_{62} will be equal to 1 and this will be equal to 0. So, this 1 is the case so if sheds.

Operations are scheduled say, x_{61} will be equal to 1 and x_{62} will be equal to hence, we find equation and other way if x_{61} operation 6 scheduling time say 2 then. what is going to happen is x_{62} will be equal to 0 and x_{61} has to be 1 to make it a 1 as I already told you 0 1 I LP so all the variables will have a value of 0, and 1 you can have values like 2, 3, 4 or point 5 or point 7 anything like that.

So, what is the solution to this? There is only 2 solutions 1 0 or 0 1 for the variables you cannot have 0, 0 or you cannot have 1 1. So, what are the says that? There either 6 will be scheduled in 1 and or 6 will be scheduled in 2 for the equation $x_{81} + x_{82} + x_{83} = 1$ that means, 8 can be scheduled in 1 and then this 1 will be 1 other 2 will be 0. If, 8 is scheduled here.

Then, this 1 will equal to 1 this 1 will be equal to 0 or 8. If, it is in scheduled time then this guy is 1 and this 2 are equal to 0.

(Refer Slide Time: 41:00)

Range of scheduling

Let $o_{i,j}$ denote the scheduling of operation o_i in step j . $o_{i,j}$ is a variable for the ILP.

For scheduling we consider 0-1 ILP, where, in the solution we can have only 0 and 1 values of the variables. In the running example, operation o_3 must be scheduled in step1.

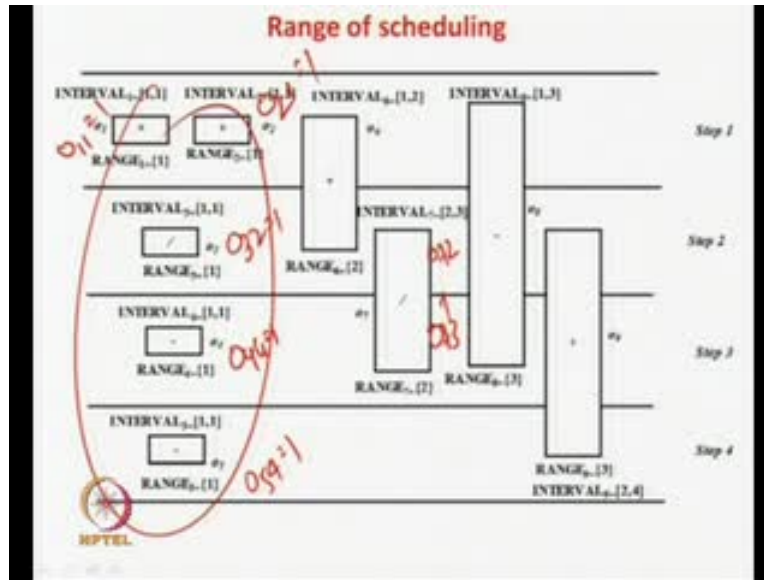
The ILP equation capturing this fact is $o_{3,1} = 1$, which implies that for the ILP solution, variable $o_{3,1}$ can only have the solution as 1.

However, for operation o_6 the equation is $o_{6,1} + o_{6,2} = 1$, which captures the fact that o_6 can be scheduled in step1 or step2. In the ILP solution either, $o_{6,1}$ will have the value 1 and $o_{6,2} = 0$ (implying that o_6 is scheduled in step1) or $o_{6,2}$ will have the value 1 and $o_{6,1} = 0$ (implying that o_6 is scheduled in step2).

So, if can write like this then the first set of constraints that is the range capturing the range of the operation is actually d1. So, you can see we write o 1 as 1 it implies that for the I L P o 1 can be only have the solution. As, 1 that is operation 1 can be only, scheduling time step.

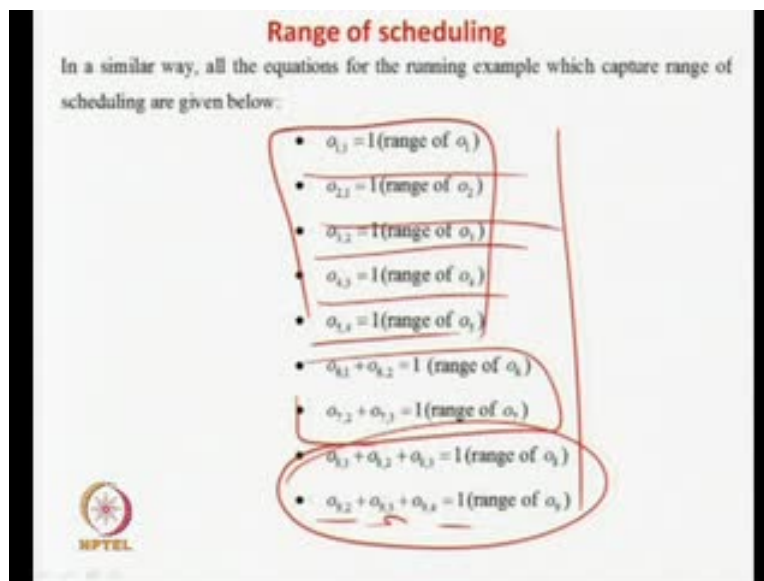
We, write o 6 1 plus o 6 2 equal to 1. that means, fact that o 6 can be captured in time step 2 and 3 and so forth so in this case I as I told you either this 1 this solution or this 1 is this solution and nothing else. So, that means either 6 end should be 1 6 a schedule in 2.

(Refer Slide Time: 41:30)



So, this is actually range for all these guys you will in this case you will have 0, 5, 1 equal to 0, 0, 5, 4 equal to 1. In this case we have 0 4 4 equal to 1 in this case 0 3 2 equal to 1 and in this case 0 1 1 equal to 1 and 0 2 1 equal to 1. So, in this case its 2 range so 0 6 1 plus 0 6 2 will be equal to 1 and 7 it will be 0 7 2 plus 0 7 3 will be equal to 1. So in this case we change ranges.

(Refer Slide Time: 41:55)



So, this are basically your equations. So, it says that 0 1 equal to 1 0 2 is 1 3 4 5. So, this is the chain corresponding to this 1 this is already fixed.

Now, these are further operations of 6 and 7. So, you can move about this 2 guys 2 2 steps. So, it says that 6 1 plus 6 2 is 1 7 2 plus 7 3 is 1 and these are these corresponds to the two sorry, these 2 guys. So this 1 can move between 1 2 and 3 this 1 can move between 2 3 and 4. So, this can move in 2 3 and 4. This is what? So this equation actually just checks the range of scheduling.

(Refer Slide Time: 42:30)

Integer Linear Programming based Scheduling

•Now we will formulate the scheduling problem as ILP. Unlike the discussion on other scheduling cases (above), we will not give a generalized algorithm to formulate an ILP for a scheduling problem. However, we will consider the running example and formulate ILP for the same and the discussion will give a generalized idea of the procedure of such a formulation.

•In the ILP for scheduling, we have four sets of equations namely,

- (i) to capture the range [[ASAP-ALAP]] in which an operation can be scheduled,
- (ii) requirement that there is no violation of resource constraints,
- (iii) data dependency and
- (iv) optimize the resource requirements.

(Handwritten annotations: red circles around (i) and (iv), a red box around (ii) and (iii), and a red diagram of a resource constraint box with a triangle inside.)

NPTEL

(Refer Slide Time: 42:50)

Requirement :no violation of resource constraints

In the running example, at control step1, we can have three multiplication operations and a subtraction operation.

However, this requirement should not violate the resource constraints.

In the example we have multiplier, divider, subtractor and adder. Let 1_{max}, 2_{max}, 3_{max}, 4_{max} denote the maximum number of multipliers, dividers, subtractors and adders, respectively.

So for the first control step, equation $o_{1j} + o_{2j} + o_{3j} \leq 1_{max}$, represents that multiplication operations o_1, o_2, o_3 can be scheduled in step1, however, their number must be less than maximum allowed multipliers (1_{max}).

NPTEL

Now, what is the next 1. So, next 1 is the requirement there is no violation of resource constraints. So, in this case as I already told you that this equation and this ILP actually

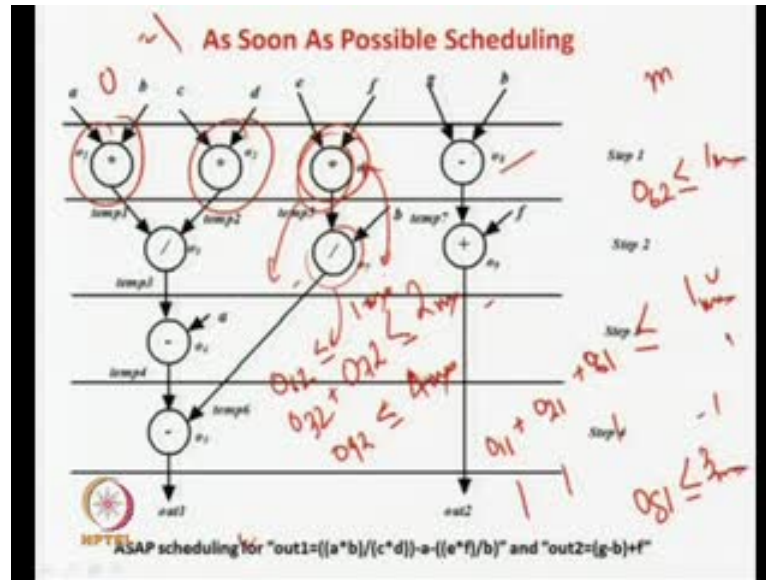
based on optimize your resources based on the time constraint. Time constraint measurably so, you can give some resource constraints you can put some basically.

You can put some higher end also, in this case or again we have to say 1 max, 2 max, 3 max and 4 max there is a 4 4 or n different f operations you have. So, 1 can be your multiplier 2 can be your divider subtraction and adder. So, this multiplier divider subtracted adder. So, whatever type of operations you have. So, those different types will have you can define some maximum.

Ok, now what is the case now? You can allow these are some values. So, generally we write it write this values. So, this values we can will see when this will be optimized later. So you just write this as a symbolic value right now. For the time being we are not putting any kind of values.

As I have told you, we can think that there can be very large in number. So, otherwise you may be getting confused while looking at this that we have some requirement there is no violation of resource constraint. and We optimize resource constraint. So, that is how we told you that writing this here. Means, that, you can write some values and you can think that they are very high value, or you can specify a very high value 1 thousand 10 thousand or something like that. For the time being we just that is why we actually write them in a very symbolic manner like 1 max, 2 max, 3 max and 4 max and will try to optimize this.

(Refer Slide Time: 43:50)



Now, you see 1 thing so what you see in this case how do you write the values. So, in this case you see it has come back lets go back to our old stuff. This 1 is the old stuff so, you can see in time step, what you can schedule in time step 1? you can schedule for the multipliers.

That is type 1 1 max. So, what you can schedule? you can schedule 0 1 1 0 2 1 and 0 6 1 so this are the multipliers which can be scheduled so 0 1 1 plus 0 2 1 plus 0 6 1. So, that should be less than or equal to 1. max that means what I am saying so, 1 1 1 means so 1 multiplier is scheduled over 0 2 1 means 1 another multiplier scheduled over 0 6 1 equal to 1 means another multiplier over this scheduled 1 here .

Schedule all the 3 multipliers in 1. So, it will be 1, 1, 1 and then it will be 1 plus 1 plus 1. So, it will be 3 multipliers and that should be less than equal to 1. Max that is it should be less than the total number of max multipliers. you are having in your some case so initially. it can be very high and then you can think it is very high and you can think and you can think coat and coat think.

It is a very high in number, and then you are optimizer or 0 1 ILP optimizer based take as minimum as possible now you can also think in another case that is 0 8 can also be scheduled over here so you can say that 0 8 1 should be less than equal to say in this case 3 corresponds subtraction so you can write that this less than equal to 3 max.

Now, let us go to time step 2 so in time step 2. What is going to happen? So, there can only be 1 multipliers, which can be scheduled over? only this 1 can, be scheduled over here. There is no multiplier so in this case you can write it as $0 \leq 2 \leq 1 \max$. Now, you consider about in time steps about the dividers. So, you can have 2 dividers over here.

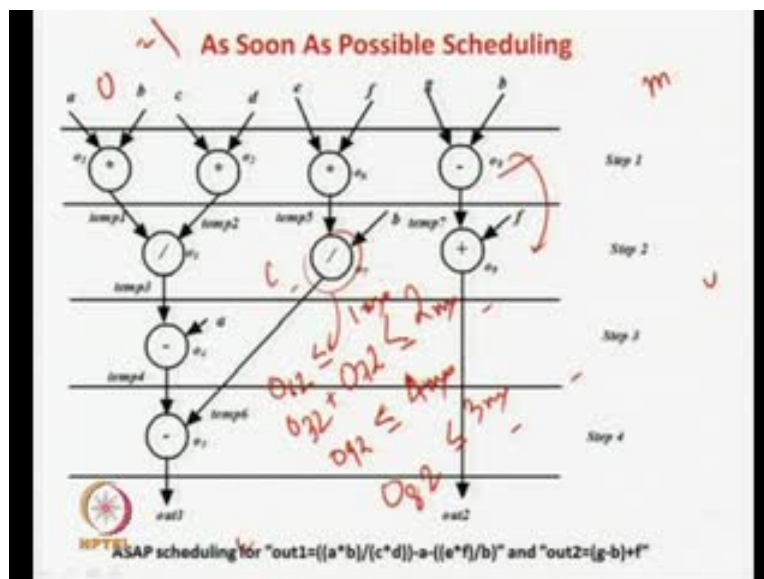
If, obviously if 6 is scheduled over here. 7 will go down but, for the time being you know that there is a possibility if 6 is scheduled over 7 can be scheduled over this is the possibility. So you write $0, 3, 2$ plus $0, 7, 2$ if, you add this 0 to dividers so, less than equal to 2 max because sorry $0 \leq 2 \leq 7$ this is because dividers.

So, the number of dividers if $0 \leq 3 \leq 1$ and this 1 divider consume $0, 7, 2$ equal to 1 then another divider consume. So, 2 dividers if you schedule that should be less than equal to 2 max that is 2, 2 max 3 max maximum numbers of dividers which is available.

Similarly, there is also add 1 adder can be scheduled over here. So, you can also you have to also write that $0 \leq 9 \leq 2$ should be less than equal $2 \leq 4 \max$. Where, 4 is the number of adders also there is 1 more thing for the time step 2. If, this guy can be scheduled over here, and this guy also can come over here. that can also be a possibility.

So, and there is only 1 multiplier that is so, you have to also write $0 \leq 6 \leq 2$ is less than equal to 1 max.

(Refer Slide Time: 46:39)



So, what do I actually say that for time step 2. What is the possibilities, for time step 2. You can see for time steps 2. you can see that this divider can be scheduled possibility. It may also go down but, possibility is there so you have to write $o_3 \leq 2$ plus $7 \leq 2$ is less than equal to number of dividers similarly, this multipliers can also come down there is a possibility.

So, you write that $o_6 \leq 2$ is less than equal to 1 max because, there is no more are dividers similarly, 9 can be scheduled over here so we write say that $o_9 \leq 2$ is less than or equal to number of.

What you call us at the maximum similarly, this subtraction can also come over here. So, you write that $o_8 \leq 2$ is also less than equal to 3 max. So, this is how we are actually writing the resource constraint equation. So, let us again go back and see in more formal. what we are saying?

(Refer Slide Time: 47:28)


Requirement :no violation of resource constraints

In the running example, at control step1, we can have three multiplication operations and a subtraction operation.

However, this requirement should not violate the resource constraints.

In the example we have multiplier, divider, subtractor and adder. Let $1_{max}, 2_{max}, 3_{max}, 4_{max}$ denote the maximum number of multipliers, dividers, subtractors and adders, respectively.

So for the first control step, equation $o_{1,1} + o_{2,1} + o_{3,1} \leq 1_{max}$, represents that multiplication operations o_1, o_2, o_3 can be scheduled in step1. however, their number must be less than maximum allowed multipliers (1_{max}).




Ok. So, in the first time step we are saying that $o_1, 1, o_2, 1, o_3, 1$ represents that 3 multiplication operations that can be scheduled in 1 however this less should be less than equal to 1 max.

(Refer Slide Time: 47:41)

Requirement :no violation of resource constraints

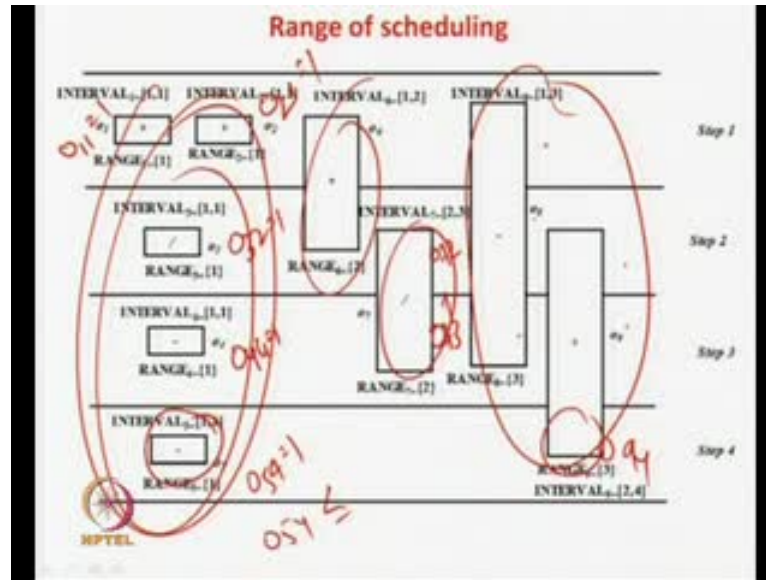
- $a_{1,1} + a_{2,1} + a_{3,1} \leq 1_{max}$ (multipliers in step1)
- $a_{1,1} \leq 3_{max}$ (subtractor in step1)
- $a_{1,2} \leq 1_{max}$ (multiplier in step2)
- $a_{1,2} + a_{7,2} \leq 2_{max}$ (dividers in step2)
- $a_{1,2} \leq 3_{max}$ (subtractor in step2)
- $a_{1,2} \leq 4_{max}$ (adder in step2)
- $a_{1,3} \leq 3_{max}$ (subtractor in step3)
- $a_{1,3} \leq 2_{max}$ (divider in step3)
- $a_{1,3} \leq 3_{max}$ (subtractor in step3)
- $a_{1,3} \leq 4_{max}$ (adder in step3)
- $a_{1,4} \leq 3_{max}$ (subtractor in step4)
- $a_{1,4} \leq 4_{max}$ (adder in step4)



So, this was is actually multiplied in step 1 also. we saw that there is also 1 sub tractor in step 1. So, you have to write this then now this are all about step 2. So, in step 2 we see that c 6 that is the operation 6 can also come in step 2. So, you are writing the o o 6 2 less than equal to m i less than equal to 1 max. because, there is can be only 1 multiplier possible in step 2. That is 6 but, there can be 2 dividers in step 2 that is 3 and 7.

So, if 6 is scheduled in step 1. Then, there is a possibility that 7 will be scheduled in step 2. So, we write that o, 3, 2 plus o 7 2 is equal to less than max and there was only 1 subtract or possible that is 8. So, we write that o 8 2 is less than equal to 3 and there was only 1 adder that is 9 and that could also be possible to schedule in step 2. So, we write that actually o 9 is less than equal to less than equal to the number of adders similarly, you can write this for step 3. and similarly, you can find out that how to write a wordings in step 4.

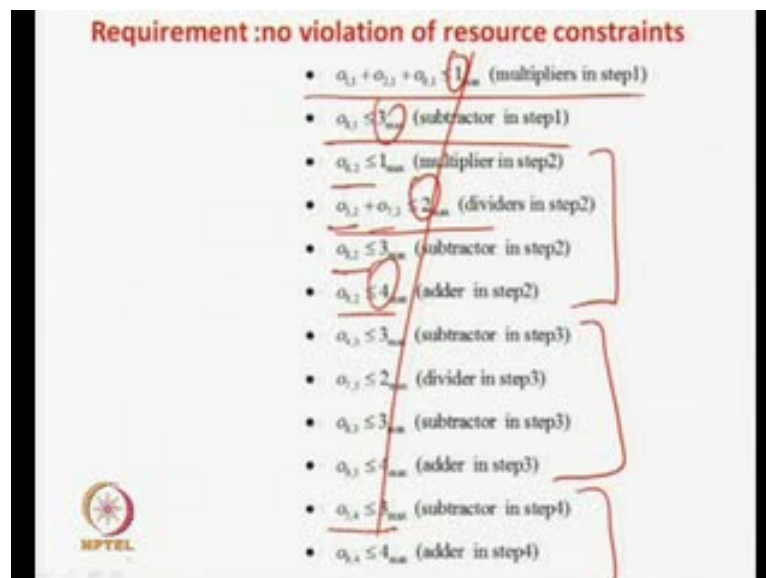
(Refer Slide Time: 48:37)



So, I mean if you look at it just 1 more example will see for the time step 4 in time step 4 you can see that what are the possibilities of schedule only this guy can be scheduled that is 0 5 4 that is subtracted less than equal to number of sub tractor max and this adder can also be scheduled over here. that is over 9.

So, 0 9 4. It should be is less than equal to total number of adders so if you look at time step 4 0 5 is less than equal to number of subtractions and 0 9 for is less than equal to number of adders.

(Refer Slide Time: 49:10)



So, in this way you can always find out all the steps. So, till now what you have done we have written this set of equations. Which represent the ranges? and these are the 6 equations, which actually says that your resources should not over should be number of total number of resources but, I have told you again, this will be this number will be automatically minimize as much as possible for the ILP. So, the now time being you can think that they are very high in number.


(Refer Slide Time: 49:25)

Data dependency

It may be noted that there are some operations in the running example, whose position are fixed namely, o_1, o_2 in step1 and o_3 in step2.

It may be noted that there is data dependency between o_1, o_2 and o_3 ; o_3 can be scheduled only after o_1, o_2 .

However, as the positions of o_1, o_2, o_3 are fixed, we need not write explicit expressions in ILP for their data dependencies. The three equations representing the range of scheduling of o_1, o_2, o_3 ($o_{1,1} = 1, o_{2,1} = 1$ and $o_{3,2} = 1$) capture that dependency relation; o_1, o_2 is scheduled in step1 and o_3 is scheduled in step2, thereby satisfying " o_3 can be scheduled only after o_1, o_2 ".



Ok, now very important is the data dependency as I already told you. Even if I do not say anything data dependency is mandatory. like you can see that o_1 and o_2 are fixed in step 1, and o_3 is whose collisions are fixed like o_1 and o_2 are fixed in type step. 1 and o_3 is fixed in type step 2. So, like how do you say that

(Refer Slide Time: 49:49)

Range of scheduling

In a similar way, all the equations for the running example which capture range of scheduling are given below:

- $a_{1,1} = 1$ (range of a_1)
- $a_{2,1} = 1$ (range of a_2)
- $a_{3,2} = 1$ (range of a_3)
- $a_{4,3} = 1$ (range of a_4)
- $a_{5,4} = 1$ (range of a_5)
- $a_{4,1} + a_{4,2} = 1$ (range of a_4)
- $a_{5,2} + a_{5,3} = 1$ (range of a_5)
- $a_{3,1} + a_{3,2} + a_{3,3} = 1$ (range of a_3)
- $a_{5,1} + a_{5,2} + a_{5,3} = 1$ (range of a_5)

HPTTEL

So, we say that as you look at it. What do you say? We say that $a_{1,1}$ is equal to 1 that means it is fixed at 1 $a_{2,1}$ is 1 that means, it is fixed at 1 $a_{3,2}$ is actually fixed at 2. That means what $a_{1,1}$ $a_{2,1}$ the output will be joining? and you are going to getting as $a_{3,2}$.

So, this is all are fixed at $a_{3,2}$ then $a_{4,3}$ will be scheduled in 1. that is 1 that is actually $a_{4,3}$ is scheduled in like say .1 2 3 and similarly, $a_{5,4}$ is actually scheduled in time step 4. So, this guys are time dependencies are already fixed. So, $a_{3,2}$ is dependent on $a_{1,1}$ and $a_{2,1}$ so this 3 step service 5 is that similarly, $a_{4,3}$ is dependent on $a_{3,2}$ so $a_{4,3}$ is scheduled in 3 scheduled in 2 and then 4 is scheduled in 3.

So, by these 2 equations these dependencies are fine and this 1 and this 1. So, you know that 4 depends 4 controls 5. So, 4 is scheduled in 4 is scheduled in time step 3 and 5 is scheduled in time step 4. So, this 2 equations basically controls the dependencies. That means, this 5 equation control the dependency automatically takes care of the dependency of this graph.


(Refer Slide Time: 50:53)

Data dependency

It may be noted that there are some operations in the running example, whose position are fixed namely, o_1, o_2 in step1 and o_3 in step2.

It may be noted that there is data dependency between o_1, o_2 and o_3 ; o_3 can be scheduled only after o_1, o_2 .

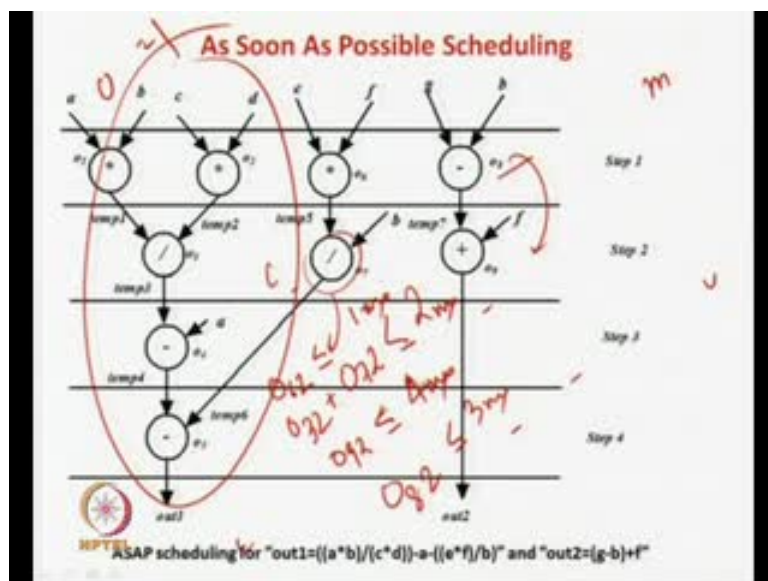
However, as the positions of o_1, o_2, o_3 are fixed, we need not write explicit expressions in ILP for their data dependencies. The three equations representing the range of scheduling of o_1, o_2, o_3 ($o_{1,1} = 1, o_{2,1} = 1$ and $o_{3,2} = 1$) capture that dependency relation; o_1, o_2 is scheduled in step1 and o_3 is scheduled in step2, thereby satisfying " o_3 can be scheduled only after o_1, o_2 " .



On this sub part but, there is a problem like will see. So, that means in this case so o_1, o_2, o_3 and all are actually scheduled after o_2 . So, o_1, o_2 and o_3 all are been taken care by the equations. So, not only the o_1, o_2, o_3, o_4, o_5 . So, all these things are actually taken care by these first 5 equations.

So, these first 5 equations you take they take care of this first 5 equations. If, you take they take care of this dependency that is dependency that already been taken.

(Refer Slide Time: 51:21)



So, this is the dependency that already been taken care in this case ok. Now we will see about this so, there is a big problem over here. So we will find out what is the problem.


(Refer Slide Time: 51:36)

Data dependency

However, for operations like o_6 (which can be scheduled in step1 or step2) and o_7 (which can be scheduled in step2 or step3), equation $o_{6,1} + o_{6,2} = 1$ states that o_6 can be scheduled in step1 or step2 and equation $o_{7,2} + o_{7,3} = 1$ states that o_7 can be scheduled in step2 or step3.

However, these two equations cannot guarantee that o_6 cannot be scheduled in step2 along with o_7 ; this will lead to inconsistency as there is dependency between o_6 and o_7 .

So for such flexible operations we need equations in ILP representing the dependency.



Now, you see this is 1 equation $o_{6,1} + o_{6,2} = 1$ plus $o_{7,2} + o_{7,3} = 1$ that is it says that 6 1 is scheduled in 1 and 6 2 is scheduled in 2 but, and also it says that 7 can be scheduled in 2 and 7 can also be scheduled in 3. So, the 2 equations are (()) but, there is a big problem over here. So, it says that 6 can be scheduled in 1 and 6 can be scheduled in 2 and this equation says that 7 can be scheduled into 1 7 can be scheduled into but, there is a miss link over here.

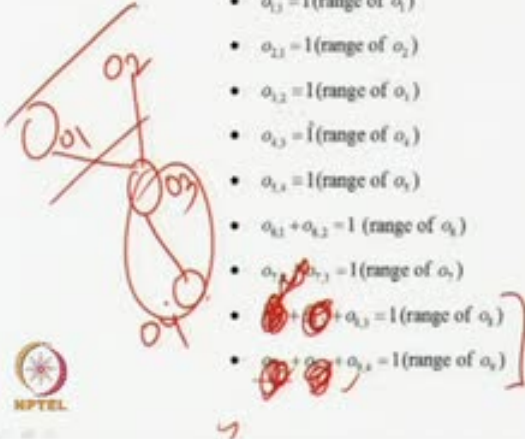
So, there should be a link says that each 6 is scheduled in 1 then only 7 can be scheduled in 2 or 7 can be scheduled in 3 but, if 6 is scheduled in 2 then 7 cannot be scheduled in 2 it has to be scheduled in 7 3 that is that is not been told by this 2 equations. Which are defining your range of operations that is a big problem over here. similarly, you can also find out that this is also big problem over here so this 2 equations if you think,

(Refer Slide Time: 52:24)

Range of scheduling

In a similar way, all the equations for the running example which capture range of scheduling are given below:

- $\alpha_{1,1} = 1$ (range of α_1)
- $\alpha_{2,1} = 1$ (range of α_2)
- $\alpha_{3,1} = 1$ (range of α_1)
- $\alpha_{4,1} = 1$ (range of α_1)
- $\alpha_{4,2} = 1$ (range of α_2)
- $\alpha_{4,1} + \alpha_{4,2} = 1$ (range of α_4)
- $\alpha_{7,2} + \alpha_{7,3} = 1$ (range of α_7)
- $\alpha_{8,1} + \alpha_{8,2} + \alpha_{8,3} = 1$ (range of α_8)
- $\alpha_{9,2} + \alpha_{9,3} + \alpha_{9,4} = 1$ (range of α_9)



It says that equation 8 operation 8 can be scheduled in 1 2 and 3 and this 1 is saying that the operation 9 can be scheduled in 2 3 and 4. but, if and only if 8 1 is scheduled in 1 then only 9 can be scheduled in 2 3 and 4 but, if 8 is scheduled in 2 then 9 cannot be scheduled in 2 it can be scheduled only in 3 and 4 similarly, 8 is scheduled in 3 then 9 can only be scheduled in 4.

So, that is not being represented by these equations. So, we have to put some more equations, which will actually satisfy this data dependency constraint. So, what is that equation? So will just see for this 6 and 7 then it can be automatically extrapolated.

(Refer Slide Time: 53:00)

Data dependency

$(6\alpha_{k1} + 2\alpha_{k2}) - (2\alpha_{k2} + 3\alpha_{k3}) \leq -1$ captures this dependency; the mechanism is explained as follows.

There are four solutions to the schedule of α_k and α_s , after satisfying the equations representing their ranges ($\alpha_{k1} + \alpha_{k2} = 1$ and $\alpha_{s2} + \alpha_{s3} = 1$).

(i) α_k in step1 and α_s in step2.

(ii) α_k in step1 and α_s in step3.

(iii) α_k in step2 and α_s in step2.

(iv) α_k in step2 and α_s in step3.

Among the four cases only (iii) is to be avoided; it may be noted that $\alpha_{k2} = 1$ (so, $\alpha_{k1} = 0$) and $\alpha_{s2} = 1$ (so, $\alpha_{s3} = 0$), will not satisfy " $(6\alpha_{k1} + 2\alpha_{k2}) - (2\alpha_{k2} + 3\alpha_{k3}) \leq -1$ ". However, will satisfy $\alpha_{k1} + \alpha_{k2} = 1$ and $\alpha_{s2} + \alpha_{s3} = 1$.

So, what we write now we write 1 into 6 0 1 plus 2 into 6 0 2 minus 2 into 7 0 2 and 3 into 7 0 3 less than equal to minus 1. So this 1 more equation we have added and we show that it actually satisfies this data dependency mechanism. So, what we have time so, this is nothing but, your stay so this is a 6 1 6 2 7 2 and 7 3.

So, what we are doing? we are producing actually this are the 2 equations that is 6 6 and 7 data dependency should be satisfied. So, what we do 6 can be scheduled in 1 and 2. So, we write 1 into 6 0 1 2 into 6 0 2 7 can be scheduled in 2 and 3. So, we write 2 into 7 0 2 and 3 into 7 0 3 and that will be less than equal to less than equal to minus 1.

So, will not tell you formerly or it will not prove that this equation is theoretical or not going to show that this equation will take care of the data dependency but, it can be very easily illustrated and very easy into deeply shown that it can be d1 now you see that these are the equations.

So, what are all the possible solutions? Solutions are like 1 0 0 1 and in this case also it is 1 0 0 1 that is what the possibilities? So, what are the possibility? 6 can be scheduled in step 1 7 can be scheduled in step 2 correct 6 can be scheduled in step 1 7 can be scheduled in step 3 6, can be scheduled in step 2 7 can be scheduled in step 2 6 can be scheduled in step 2 7 can be scheduled in step 3. so in these are the 4 solutions. which are going to satisfy this equations.

Now, you can see that only there is a problem in this O_6 is scheduled in step 2, and O_7 is scheduled in step 2 and O_6 is scheduled in step 2 and O_7 is scheduled in step 2 this is actually not allowed other things are possible O_6 in 1 O_7 into fine O_6 is 1 O_7 in 3 is fine O_6 in 2 O_7 in 3 is fine. So, only 1 problem is with this O_6 into and O_7 instances that are not possible. So, how it can be avoided? Now you have said that.

So, in this case now we are looking at this now we have writ¹⁰ this equation additional equation we have writ¹⁰ that O_6 1 into this 1. So, if you see that if we are writing that O_6 1 in step 2 that means, what actually this is going to be? a 1 so, O_6 2 is equal to 1 and in this case also to be scheduled in step 2. So, this guy is to be 1 so 1 and a 1 so O_6 2 is a 1 and O_7 2 is also 1 in this case others are 0 this guy is 0 and this guy is 0. So, in this case 2 into 1 is a 2 and again 2 is 2 into 1 is a 2 so 2 minus 2 is 0. So, 0 is less than equal to minus 1. So this is this not valid and the equation (()) false.

So, in other words what I am saying that I am not going, we are not going to discuss formally. How it is actually helping you? Out but, the only thing is we just saying the process scheduled the details are out of the scope, out of this scopes but, individually you can think that you will do your job. So, what we are doing?

So, we are actually multiplying we are so, O_6 can be scheduled any operation which can be scheduled in step 1 2 3 4 5. So, we are writing it as O_6 1 into 1 O_6 2 into 3 and if there more possibility we write 3 into O_6 3 dot dot minus other operations. with this data dependency conflict can be possible. So, we are writing O_7 2 into 2 O_7 2 into 3 dot dot dot. If, it is possible and that is equal to minus 1. Now, you can find out that only 1 is a problem where there is a overlap. So, to make this is a overlap the answer is this 1 is to be a 1 and this 1 has to be a 1 this has to be 0 and this has to be a 0. So, it is 0 1 and 0 1 sorry 0 1 and 1 0.

(Refer Slide Time: 56:44)

Data dependency

So, equation $(1o_{s_1} + 2o_{s_2}) - (2o_{s_2} + 3o_{s_3}) \leq -1$ could incorporate the data dependency in the IPL.

To generalize, the equation $(1o_{s_1} + 2o_{s_2}) - (2o_{s_2} + 3o_{s_3}) \leq -1$, is formulated as follows.

The parts of the equation, " $1o_{s_1} + 2o_{s_2}$ " and " $2o_{s_2} + 3o_{s_3}$ " are taken from the range equation for o_{s_1} and o_{s_2} , respectively, after multiplying the terms with the corresponding control step number. Then we subtract the sub-equation of the successor from that of the processor and the result should be less than or equal to -1. "less than or equal to -1" corresponds that o_{s_1} is to be predecessor of o_{s_2} .

Data dependency equations for the running example are as follows:

- $(1o_{s_1} + 2o_{s_2}) - (2o_{s_2} + 3o_{s_3}) \leq -1$
- $(1o_{s_1} + 2o_{s_2} + 3o_{s_3}) - (2o_{s_2} + 3o_{s_3} + 4o_{s_4}) \leq -1$

0 1 and 1 0 is actually reading to the problem what you feel it 0 1 and 1 0 in this equation additionally this will not satisfy 0 1 constraint and so, this 1 will not be taken into picture. So, with this addition set of equations like this 1 and for o 8 it will be 1 8 can be scheduled in 1 2 and 3. So, we write 1 into 1 8 1 2 into o 8 2 3 into o 8 3 minus 9 can be scheduled in 2 3 and 4 so we write 2 into 9 2 3 into 9 3 and 4 into 9 4 that should be less than equal to 1.


So, if you write these additional equations. What you will find that data dependency based constraints? Are also satisfied that is in this case. what can be the violation the violation can be? If, 8 is scheduled in 2 and 9 is also scheduled in 2. Then, you are in a big problem then this is 0 this is 0.

So, this is cut it is 2 into 1 minus 2 into 1. again it is not less than equal to minus 1 so, it is violated. So you can easily verify that I mean if you write this equation this manner. So, this is this how you generalize how you should write this equation that is the range whole range you write and you product with the number or step. number and you subtract with the other part then you get and you write less than equal to minus 1. So, this equation will automatically check that there is no violation in data dependency.

(Refer Slide Time: 57:45)

Optimize the resource requirements

This equation represents the optimization criterion to minimize the resource cost.
In the running example, the equation is
Minimize " $1_{max} + 2_{max} + 3_{max} + 4_{max}$ ", subject to satisfying all the equations for range, resource constraints and data dependency given above.



(Refer Slide Time: 58:16)


Final Solution of the ILP

If we solve the 0-1 ILP using any standard method, we get the following solution:

- $a_{1,1} = 1$
- $a_{2,1} = 1$
- $a_{3,1} = 1$
- $a_{4,1} = 1$
- $a_{5,1} = 1$
- $a_{6,1} = 0; a_{6,2} = 1$
- $a_{7,2} = 0; a_{7,3} = 1$
- $a_{8,1} = 1; a_{8,2} = 0; a_{8,3} = 0$
- $a_{9,2} = 1; a_{9,3} = 0; a_{9,4} = 0$

$1_{max} = 2; 2_{max} = 1; 3_{max} = 1; 4_{max} = 1$

the best



So, if you write all these equations all the concepts are taken care range data dependency as well as you have to solve the you have to constraint or you have to restrict the number of resources. So, it is 1 max 2 max 3 max 4 max that can be initially assume that it is very high in number already. it is d1 now ILP. If, you run automatically it will minimize 1 max plus 2 max plus 3 max plus 4 max that is the minimum number of resources. You, require that will be minimized this is the this you have to minimize.

Now, if you run ILP I am not giving the details of how you solve the ILP these are well known under graduate course.

Now, you solve this so you will find that this 1 is going to be your answer, that is these are these were you are fixed and you will find that 0 6 1 6 2 is equal to 1 that is 6 is scheduled in 2 7 will be scheduled in 3 8 will be scheduled in 1 9 will be scheduled in 2 were equal to multiplexer 2 multipliers 1 divider 1 adder and 1 subtractor.

So, automatically this value will come into picture. So, in this case we are seeing that ILP is going to give a solution. Which is the best? So, this is the best. Now, you can also observe that first added scheduled has also given the same solution least scheduled has also given the same solution.

So, you are you may ask why I L P actually now, you will see with an example; that ILP will always going to give you the best solution all cases I LP is going to give you the best solution but, the time taken or to solve this a problem, and that is to solve this all. How to solve? So, all the equations satisfied actually is a 0 1 linear programming problem.

Which is known to be a NP high NP complete? Which is going to an exponential number of type? In this number of this variables so, we are saying that these exponential number of steps exponential in number of operations basically formally speaking this exponential the number of variables 0 1 0 2 dot dot dot.

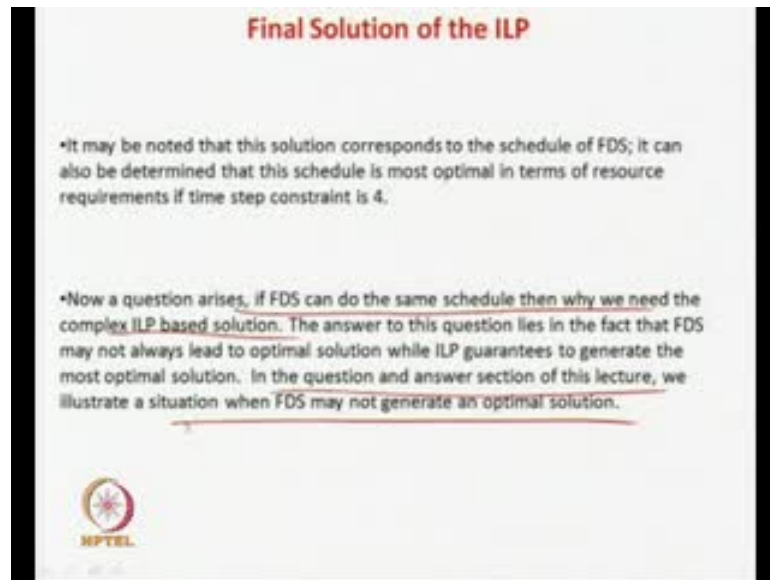
All the variables you have taken is exponential in that number. So, it is going to be a huge you know because, it is the variables are depending on the number of operations as well as the number of steps.

So, you will take a very very long time to solve. But, it will give you the best solution so in this example it was by chance that force scheduling and list scheduling has given the same answer. But, in the example; we will see the both answer session that is not the case there are some examples; where FDS is not going to give you the best solution? But, ILP is guaranteed to give you the best possible solution.

Because, it cares of all possible permutation and combinations and then it generates the results so how is that you can think or you can read some extra material which actually says that this ILP actually takes care of all possible permutation and combination it takes


it tries internally when internal equations are written in such a way. So, all possible permutation and combinations are tried and then it actually solve the best possible solution but, the time taken is so high.

(Refer Slide Time: 60:18)



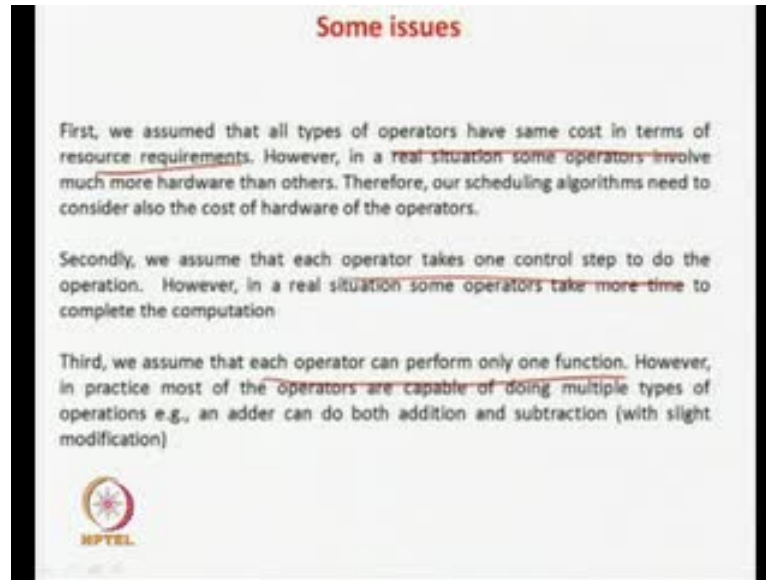
Final Solution of the ILP

- it may be noted that this solution corresponds to the schedule of FDS; it can also be determined that this schedule is most optimal in terms of resource requirements if time step constraint is 4.
- Now a question arises, if FDS can do the same schedule then why we need the complex ILP based solution. The answer to this question lies in the fact that FDS may not always lead to optimal solution while ILP guarantees to generate the most optimal solution. In the question and answer section of this lecture, we illustrate a situation when FDS may not generate an optimal solution.


NPTEL

That is why we are actually go we are bound to go to few restricts. that is, what I was saying that? FDS same schedule is given the ILP then why I I L I LP? Why not FDS? Are some other few restricts. So, that will be FDS I mean so, in this question also we can find case where FDS is not giving you the most optimal solution but, I guess ILP is always going to give you the best possible this thing. So, before we go to the question answer session there are some small issues that we have kept this lecture very very simple gives you various results.

(Refer Slide Time: 60:51)



The first reason is we have assumed that all the operators are same cost in terms of resources required. Thus, we are assuming that all the adder is taking same hardware multiplier is actually same cost in hardware divider is same cost in hardware. So, we thought we are thinking that we have to minimize adder minimize multiplier minimize subtracted all things you have to minimize at equal priority.

But, the answer is no actually multipliers are and dividers are very high in hardware. Hardware cost so an adders are structural less in numbers our schedule algorithm is not very simple scheduling problem as well as scheduling solution. Here, we give priorities that if you can reduce the multipliers or adders multipliers or dividers are high cost hardware then, please go on do that and in that process. If, you keep on increasing the adders and the reverse of adders and the subtractions and say comparators are a uh hardware itself are not very expensive number.

We allowed to do that is we want overall hardware requirement reader then overall total hardware requirement reader then equal treatment. like in this case we have thought that we have to reduce add adder multiplier subtracted divider all equal priority.

But, that is not in really the case if we try to reduce important 1s like multipliers dividers factorial. Computers we directly take more area in their implementation and more

complex. So, we try to reduce those numbers even if, in that process the number of adders subtractors are some simple hardware increases.

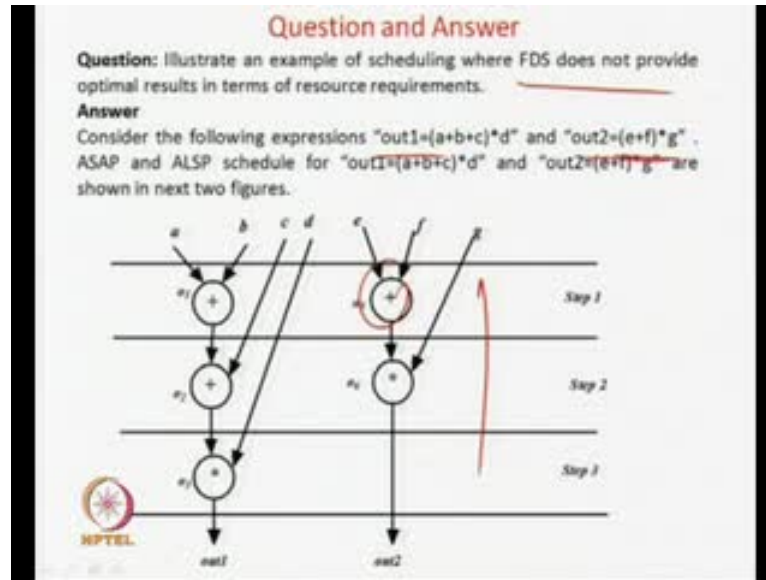
So, that is all not considered in this lecture. So, you can understand that it will make a problem as well as the more complex then we take that we assume that all the operations are d1 in 1 state but, that is also not true multipliers. refer may be referring 2 steps to do the multiplication but, adders and subtractions are may do it in 1 step.

So you may have multiples step or variable size step or multiple size multiple times step based optimization for scheduling and. it will again make the problem very complex to solve. thirdly we assume that all the operation can do 1 function only but, that is also true adder can do multi adder can do addition as well as.

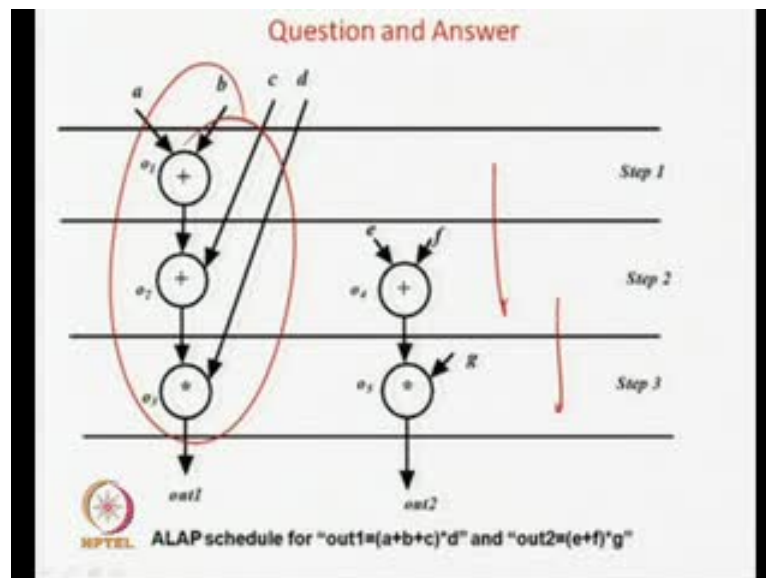
It can do subtraction and you know that 2s complement addition and subtraction hardware are very very similar similarly, add multipliers or subtracted hardware also looks very much similar. So, 1 operation can do multiple operations. So, you know that so if I require o in 1 time step 2 multipliers and 1 divider and in the next time step say all dividers. So basically you can reuse the multipliers for division and vice versa.

So, actually all this are many more steps. we cannot we are not bringing into picture but, you can feel that if I bring all this important parameters into picture or important factors into picture the scheduling problem becomes a very very complex problem. if you taking into care the multiple functionality multiple time steps then cost of each hardware may be different. So, if I bring in all this issue into picture. So, really scheduling is a very difficult problem and have complex algorithms like I 0 1 ILP or future exact problems are almost impossible to be applied so we keep on apply new heuristics. So, that you get a good solution near optimal solution but, it may not be the waste optimal 1.

(Refer Slide Time: 63:42)



(Refer Slide Time: 64:42)



So, that is that about the so let us come to the question answer session which you are waiting for a longtime as well as mean time? We are saying that we give you an example; where first ratite scheduling, were not provide you the best result? in the examples; we discuss in the lecture FDS was giving the best result as well as equivalent to ILP.

It was very much equivalent to particularly scheduled but, generally speaking. it is an example; where FDS is not going to give you the best result and in fact you can find out

So you can easily think that, what will be the best optimization? So, I mean and also thing will be coming into picture multiplier is must here in cost stuff compared to an adder that is also is a very well known thing, and everybody known's this. So, that factor also was into picture. So, what is the best possible stuff?

So, I will put this adder over here and I will put my multiplier over here. That is actually this schedule.

So this will be will be two adders and 1 multiplier then, we know that multiplier cost is much less but, if you take this case where we actually go for this schedule. in which case we actually say we scheduled this adder over here. Because, we know that this adder we are scheduling over here as, we know that this adder can be scheduled end. then free then this will actually schedule this multiplier and this multiplier will be scheduled over here.

Now, we require 2 multipliers and 2 adders. So, that is actually a big problem. So, you require 2 multipliers and 2 adders. So, it cost becomes very very high now adder cost is also increased multiplier cost is also increased. So, that may lead to a problem. So, we are not sure that FDS can actually give you the solution.

So, now the thing is that so, FDS is that you have to take 1 operation. At a time 1 operation at a time find out the probabilities and then you have to schedule. It now in this case let us assume that, we take adder first no where it is mentioned that you have to take multiplier first or adder first. or which hardware you have to take first?

Let us say anything about it says that you take any operator and try doing with it. So let us check the adder soothe probability will be 1 plus 0 point 5. Because, this probability is half and it is actually going to 1 point 5. Here, also it is 1 plus point 5. So, it will equal to 1 1 plus point 5. So, it will be equal to 1 point 5 for the adder here. we do not have any adder so the probability is 0 now. So, you think that if I schedule the adder over here.

So, it probability will become 2 and if I schedule the adder over here. it probability will be 2 now nobody says that no. Where I have to schedule. So, both of the cases the probability will become 2. So, let us just take some time and let us think that I schedule it to 1 nobody says that you look at for the multiplier.

You look at for some other operator or you can take some other things into constraint your FDS are not say that, it says that if the both of them are equivalent you take arbitrarily but, something is 3 and something should not go for the lower 1. So, in this case nobody says that you look at for the multiplier. What constraints you are applying. If, you schedule the adder nobody says. So, in FDS. So, we schedule the adder over here.

(Refer Slide Time: 67:24)

Question and Answer

INTERVAL₁ [1,1]
 ϕ_1 +
 RANGE₁ [1]

Step 1

INTERVAL₂ [1,1] INTERVAL₃ [2,2]
+ ϕ_2 ϕ_3 +
 RANGE₂ [1] RANGE₃ [1]

Step 2

INTERVAL₄ [1,1] INTERVAL₅ [3,3]
+ ϕ_4 ϕ_5 *
 RANGE₄ [1] RANGE₅ [1]

Step 3

FDS starts of "out1=(a+b+c)*d" and "out2=(e+f)*g" after ϕ_1 is scheduled in step2

(Refer Slide Time: 67:36)

Question and Answer

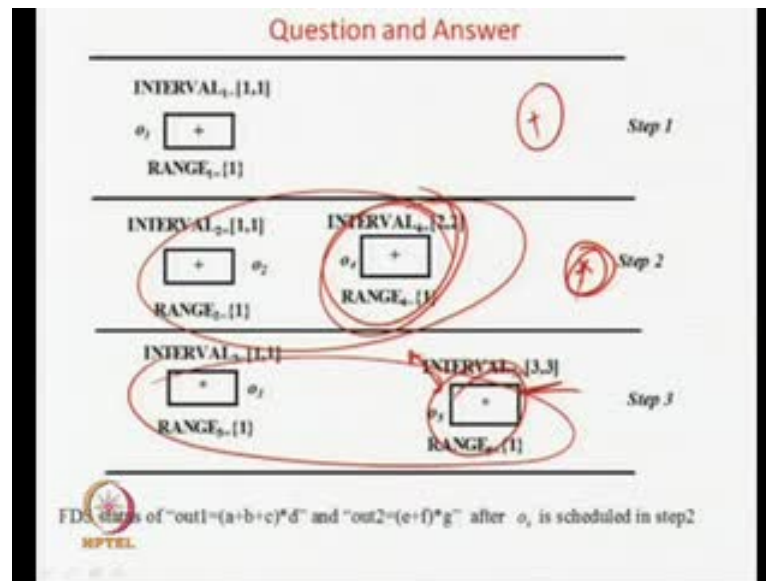
As FDS algorithm does not specify which type of operator to start with, let us consider adder first.

From FDS algorithm it may be noted operation ϕ_1 can be scheduled in either step1 or step2, because both will lead to same cost in terms of resource requirements (number of adder is two).

So let the operation ϕ_1 be scheduled in step2, which implies that operation ϕ_2 will be placed in step3.

Now the total resource requirement is two adders and two multipliers.

(Refer Slide Time: 67:46)



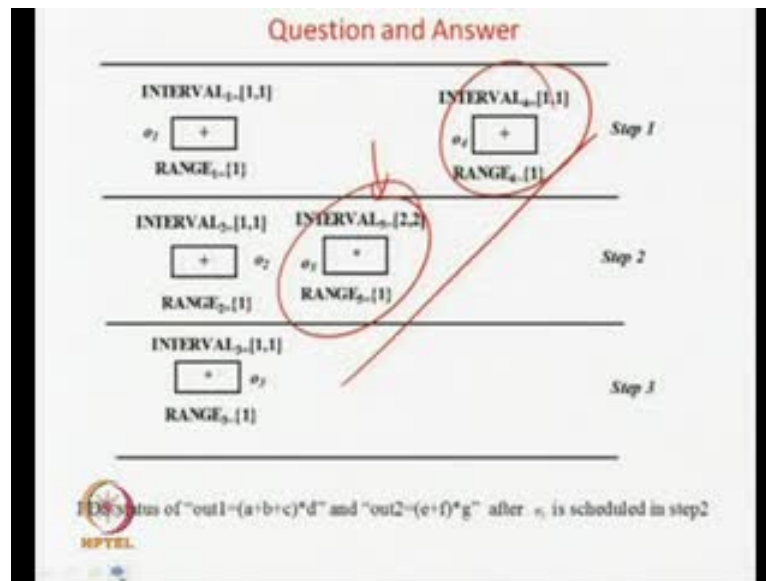
Now what happens in the multiplier by default you have to schedule into varied below the range is actually not 3 3. So, you require 2 multipliers and 2 adders which is not the optimal 1. So, that is what I told you that FDS can lead to a problem. Because, in this case we consider the adder first and we did not have any lucent. If, I put the adder over here then what is going to be the problem?

But, in if you take multiplier first then obviously the multiplier will be scheduled over her. it will not be scheduled over here. And then your adder will be scheduled over here. If you take multiplier first but, pure FDS does not say that. So, that you can think that. I can improve on my FDS by that by saying that.

I will take the operators first, which are high in cost and then I started multiplier and then i started the other somehow. You can say that even if we start with the arbitrary sequence. Then, I have to keep a look at that. If, I put adder here then multiplier will be restricted.

Then what will be the case and all so all this things are FDS plus other heuristics that is how research goes on but, if more the amount of intelligence you put it more amount of complexity will be putting and more and more towards ILP. you will be going if you put a very very complex algorithm then that is actually nothing but, ILP.

(Refer Slide Time: 68:30)



So, we have shown an example so that this was the best solution actually you put multiplier first and then you put the adder. If, you started multiplier that is what the FDS plus you were actually having an emphasis on starting with the operators. Which are high in cost like multiplier and divider, and then start with this 1.

So, in an actual what we found out that. So, F D S or as soon as possible, as late as possible. May not give optimal solution in many of the examples; but, we can keep on modifying our algorithms like look at giving some priority and all. So, that we can good or near optimal solution or optimal solution in many of the cases but, the time that is much much less compared to. What do you call your? What do I say about 0 1 I I p take exponential amount of time in the number of variables.

So life is like that for most of the algorithms are problems in this course are all n p complete n p hard cannot have a well known componet level algorithm, say you have to heuristics and you have to modify and heuristics like. In this case we modified f d s plus priority with the amount of cost of the hardware.

So for keep on modifying heuristics and you keep on getting better and better algorithm. Which will give a near optimal solution in the next course? In the next class actually what will be looking we will be looking at some kind of automated algorithms like posideurs for the binding allocation and binding problems. So, the scheduling is already d1

algorithms now what the schedule is d1 we will see how we can go about this binding?
what do you say about this allocation? and binding problem.

Thank you.