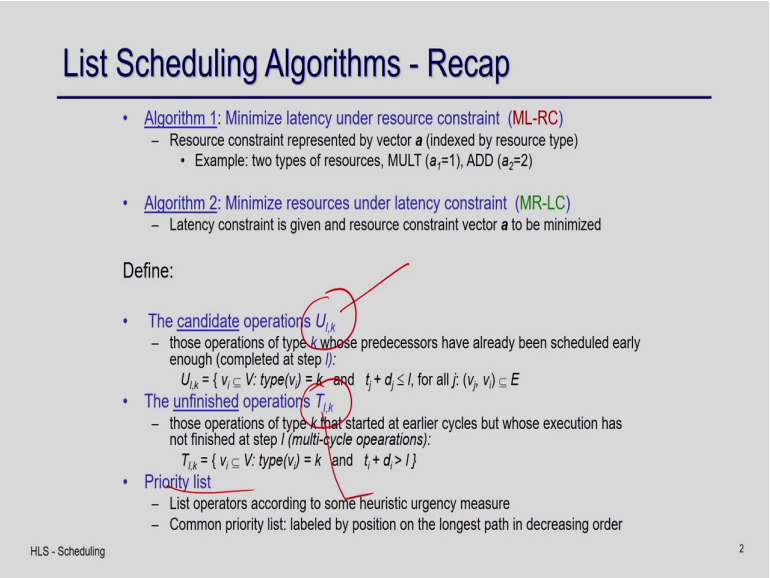
C-Based VLSI Design
Dr. Chandan Karfa
Department of Computer Science and Engineering
Indian Institute of Technology, Guwahati

Module - 03
C-Based VLSI Design: Scheduling
Lecture - 11
Heuristic Scheduling: List Scheduling for MRLC

Welcome students, in today's class we are going to learn about Heuristic Scheduling for MRLC problem which is basically minimization of resource under latency constraint. So, just to recap in the last class we have discussed about the List scheduling algorithm.

So, we have also discussed that the scheduling problem in general is NP complete its computationally difficult problem and so, most of the time heuristic based algorithm actually applied in practical purposes and one of the most important heuristic based algorithm is list scheduling.

(Refer Slide Time: 01:23)



In the last class we have discussed the list scheduling for MRLC problem Minimization of Latency under Resource Constraint; that means, you have given a resource bound and you try to optimize the latency. The basic idea of this list scheduling is very simple you basically

have a list is a priority list right. So, we have all the nodes or the operate operations in the sequence graph and what we do?

We actually identify we make them an order priority order based on some priority function. One of the important priority function is the distance from the sink node and that is what we have been used for MRLC problem that we calculate the distance of the longest path from any node to the sink node and that is what is a priority value and based on this priority value I just make a order of this node I maintain a priority list which is based on the decreasing value of this and whenever and what is the basic idea?

So, we keep scheduling the operations are from time step 1, then 2, 3 and so on in every time step what I am going to see? I am going to see what are the operations are now ready to be scheduled. How do I determine that? Those are the operations whose predecessors are already completed their execution its not only they are started they have also completed their executions. So, that means, all the in data the on which it is dependent is now available.

So, I am going to identify all such operations which are ready to be scheduled and also we I am going to identify the operations that are actually started early, but still running. So, it may be some d cycle operations which have say scheduled sometime earlier some previous cycles previous time step, but still it is running. So, if something is running we need resource for them.

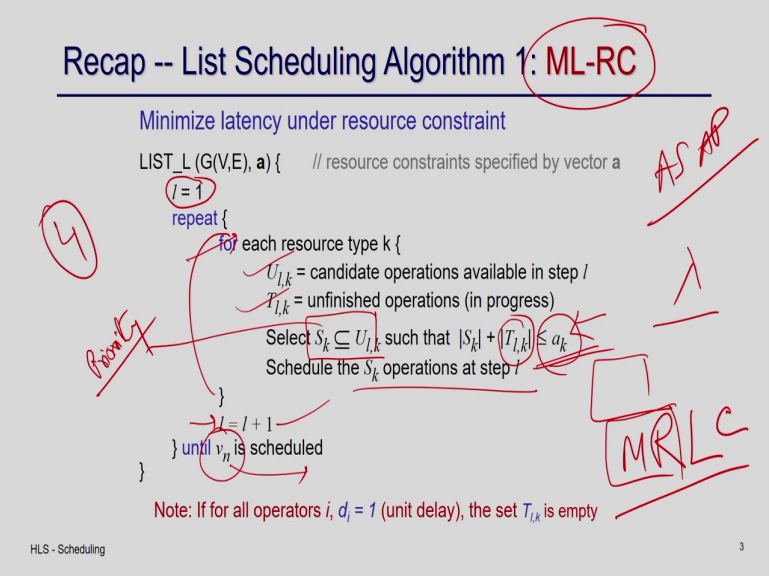
So, identify these two set of operations which are ready to be scheduled and which are still running in time step l and then what we do? We identify we are definitely going to assign resources for these running operations and whatever left I am going to whatever the resource left I am going to select a subset of those ready operations based on their priority.

So, and then I am going to schedule them at time step l. So, that is the basic idea. So, whenever there is a collision; that means, there are many say there are k number of operations ready to be scheduled and there are fewer number of resource available, I am going to choose a subset of those k based on their priority value that is the basic idea and then you are going to schedule them and rest will be wait for the next time steps.

And this way it will go on from time step 1, 2, 3 and so on. This is how we have done and if you remember we have just identified this ready list is a U l, k and T l, k is the set of running operations in time step l for operator type k because you might have different types of operators.

So, this is what type k operator and similarly this U l, k means the rest the set of operation that are ready to be scheduled in time step l of type operator type k. This we have identified and this as I mentioned the priority list is based on some priority value you can have your own priority function, but the function that I have considered is the longest path from any node to the sink node.

(Refer Slide Time: 04:43)



So, and just to recap what this ML-RC problem was that you have given a resource bound and you will try to optimize the latency what you do? You start from time step 1 and for every time step you for each type of resource you identify the ready list and the running list and you schedule the running operations and you find a subset of the ready operation such that your resource bound is satisfied and this selection is based on the priority.

And then I am going to schedule this S k high priority nodes I mean operations in time step l and then I will move on I will do for the next operator and so on once the all operator are done for time step l I will move to the next time step and I am going to keep doing this until I

reach I can schedule my sink node and this is what the list scheduling algorithm for ML-RC problem.

Now, in today's class I am going to discuss the list scheduling problem for the dual problem where we have this latency bound is given. So, latency bound is given resource we have to identify so; that means minimizations of resource under latency constraint.

So, we can do the same thing, but there is a bottleneck here. So, you try to understand that. So, what we are doing? Say suppose we are in time step 1 we identify all the operations to be scheduled in time step 1; that means, they are ready to be scheduled in time step 1.

And since its a time step 1. So, there is no running operation the running operation set is null. So, now, say I have say 4 operation which is of type say multiplier is ready to be scheduled in time step 1, but there is no resource bound given that is what we try to optimize. So, how do I going to select because here we have a resource bound given.

So, I select the subset of operations from this ready list based on this resource bound I should not over source the resource bound, but here there is no resource bound is given rather I have to calculate this resource bound. So, one possibility might say since these operations are ready you schedule them there.

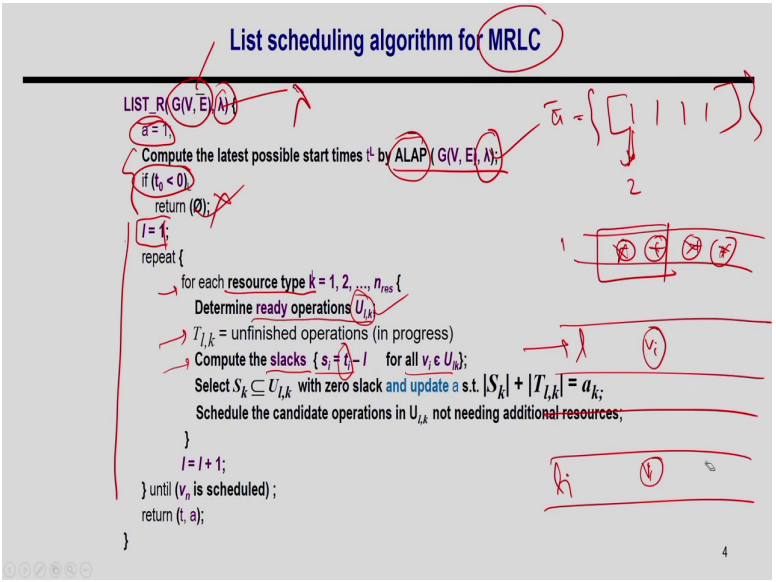
So, then this big algorithm becomes ASAP. So, obviously, you understand with ASAP you never re-get an optimum resource. So, whenever some operations are ready scheduling them in that particular time step will not give you the minimum resource. So, that list scheduling although its a heuristic algorithm cannot go that way.

So, here how we solve this problem because I know there are certain operations is ready in some steps, but because I do not know the resource bound I am not going to take a call which subset I am going to select. I mean I can always select a subset, but the selection of the subset based on some limit.

So, I am going to select a subset what is the number of element I am going to subset of the element I am going to select that something is not known to me. So, for this MRLC problem.

So, I have to resolve that problem. So, that particular resource. So, here is a very nice idea that the way this MRLC algorithm works.

(Refer Slide Time: 08:02)



Let us assume that initially you have all of operator is 1 1 1 1. Say suppose you have four type of operator and they are done at least you should have one instance of them. So, you assume that there are minimum numbers of instance available in the hardware.

So, you have say one multiplier, one adder say one comparator, one say divider and so on. So, there is a four type of orbital. So, initially assume that they have only one instance available and lets start this scheduling by this list scheduling and whenever I found some operations which has to be scheduled in this time step otherwise I cannot meet my latency constraint, then I will increase this bound to I will just increase the bound by those number.

So, not by one always say for example, say in say time step 1 there are algorithm I am talking about. So, there are say 4 multipliers to be scheduled. So, they are ready to be scheduled and now since you assume that in the time step we have only one multiplier available because you just take the minimum instances.

And now, we found that if you do not schedule these two in the time step 1 then you will not be able to meet the target latency. That you have to always keep in mind that I must schedule

my behavior within lambda time step what is the bound latency bound is given because I cannot violate that I can increase my resource, but I cannot violate my latency bound.

So, whenever that scenario come I am going to increase my resource bound or resource count or the resource bound by those many numbers so, that I can actually schedule those critical operations in that particular time step and this is how I am going to keep doing it until I reach the last operations.

So, this is the idea is very simple idea, but this is something you need otherwise you cannot develop this algorithm because resource bound is not there, it has to be calculated and you cannot just take a upper bound rather you take the lower bound and you increase the bound whenever there is a urgency there is a critical operations.

So if you do not schedule it I am going to violate my latency bound let us increase the resource at that moment only not before that or after that. So, let us understand the algorithm which is very simple. So, you have given the sequence graph which is G and you have given the latency bound lambda.

So, now, as I mentioned that you assume that this is your a is the resource bound that is what is what I am trying to say is. So, all kind of instances you have one instance available. So, one thing we have to understand that if the lambda given whether that is a feasible bound or not.

So, how we can do that? I hope you remember that we have discussed that ALAP algorithm and the ALAP algorithm we actually can check whether latency given which is whether its a schedulable whether your sequence graph can be scheduled within the lambda bound or not.

So, that. So, if I just run this ALAP algorithm with this lamda whatever the given value and if we can actually schedule the all operations within lambda ALAP can schedule them then this is something a valid latency bound. So, you remember that for ASAP gives you the minimum latency. So, your latency whatever the latency you are giving it must be greater than of a latency that is obtained by ASAP algorithm.

So, if it is less than that this ALAP will tell it is not possible to schedule less than within the given lambda. So, that something we can actually first check. This is just to check that the

latency bound is given its a valid latency bound. So, that I just check here and if the if it is less than 0; that means, I cannot schedule my first operation t0 and time step 0 so, that mean this is not a feasible latency bound I will say [FL] its not possible.

So, I will just say [FL] return phi and say it is and if it is possible to schedule this behavior in within the latency bound given lambda then my actual algorithm starts here. So, it is similar to my MLRC algorithm because they are also its a this same as list scheduling algorithm. So, what I am doing? I just start from my time step 1 I will and for every time step I am going to do it for all resource type k.

So, I am going to do it for all type of resource and what I am going to do it now? I am going to identify the ready list which is the operation that is ready to be scheduled in this time step that I have already discussed. So, when an operation is ready to be scheduled in this time step when all its predecessor has completed this execute before l before this current time steps l.

So, I am going to identify this and then this is the most important thing this is where the priority determines. So, here I am going to identify the slack of each node in this ready list I am going to discuss that.

So, what is this slack? And if I found I am going to say I just calculate this slack. So, let us explain what is slack. So, slack is something is this, say for every node if I will I am now actually in time step l. So, I am in time step l. So, this t i is giving its t ALAP time step which is actually calculated by ALAP.

So; that means, this t i is the last possible way of possible time step where this can be scheduled that is what ALAP give you the last as possible this is the last possible time step where these operations can be schedule, if I want to meet this lambda. So, if I want to meet this lambda latency bound this is the last possible time step. So, now I am in l time step hence and you are say suppose this is the t i say this is the t i value so; that means. So, there are say two more steps there.

So, that means, I am in this time step, but this operation which is I am now currently considering it can be still scheduled here so; that means, this is not a critical operation yet because it has possibility to be scheduled in another time step here, here or here without

violating my latency bound. So, since this operation does not become a critical one till this point what I am going to do? I am not going to increase my resource bound.

So, since this operation does not become a critical operation, but say this is your t i say in some case it might happen that you are in time step l and the t i which is obtained by the ALAP algorithm for this node v i is t i so; that means, my I am in the step which is the last possible state where this operation can be scheduled to meet this latency bound. So, I have no choice I have to schedule this operation here then only I can meet my latency bound.

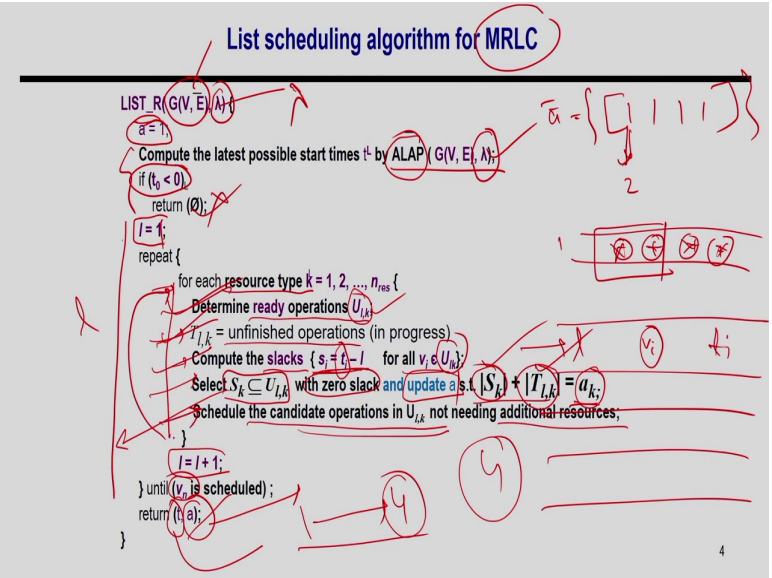
So, that mean this is now must become a must to schedule this operation in this time step. And if the resource does not adequate resource is not available I am going to increase my resource bound because this is this operation is now become critical and hence I have no other choice than scheduling this operation in this time step and hence if I need any more resource I am going to increase it. So, that is the idea.

So, if you just understand here that I will identify my ready list I will identify my running operation. So, what are the operations are running and then I for all the operation in the ready list I identify their slack value and then I am going to select those S k subset of this U k whose slack value is 0 it has a 0 slack; that means, if there are operations whose this is the last possible time step that t i minus my current time step is 0 that means, this is 0 slack that it has to be scheduled here I identify all such states.

And then I am going to increase my resource bound if so, that my running operations which is running in till this time and this S k which is become critical the total should be a k.

So, the; So, if we my current the current value of a k is less than this I am going to increase it just for as the example that just given few minutes back that if it is say 1 and say I found there are two operation will become critical and there are say one operation is running. So, 3 plus 1 is 4. So, I will increase my resource from 1 to 4 because 3 operations become critical they have to be scheduled here and one is running.

(Refer Slide Time: 16:30)



So, 3 plus 1 is 4. So, I will increase my resource from 1 to 4 in this time step and you might have a scenario that in some state you already have 4 resources available, but there are no such critical operations.

So, then in that time what do you do? You just select a subset of this operation from U k and you schedule them there. So, because in some time step if we just increase the resource from 1 to 4, you can assume that from the rest of the time step these 4 resources are available.

So, once you decide that I need 4 resources they are available in the hardware. In the next time step there may be no critical operations only say 3 running operations. So, if there are 3 running operations and there are no critical operations, I have still one resource available. So, I can actually take any such any operations which is not so, critical now.

And that time I can select based on the priority which is given by the longest path from that node to sink node. So, I can utilize these two priorities. So, priority one is the criticality, the slack the operations user is 0 slack there must to be scheduled and once I have resource available, but there is no not enough critical operation then I can actually such select a subset of ready operations based on their priority. So, that is the overall idea.

So, for each type of resource I will identify the ready list I will identify the running operations, I will identify the slack of all ready operations and I will identify those operations

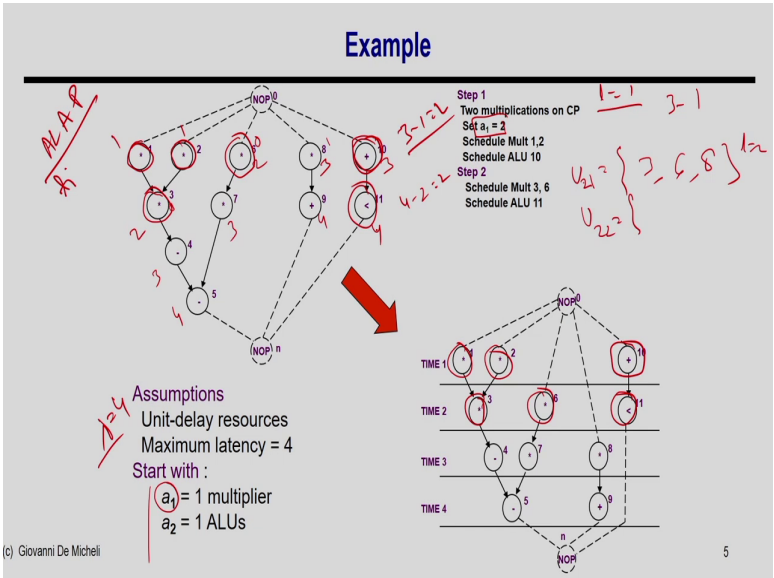
which is critical; that means, there may 0 slack and then I will make sure that I will if I have to increase my resource bound I am going to do it so, that my running operation have a can run and all the critical operations can be scheduled in this time step.

In some scenario if there is a not enough critical operations, but there are some resource available because it has been increased in previous some time steps, I will select a subset of those ready operations based on their priority. So, this ones I done for a one type of resource and then I will do for next type of resource and so on.

And once for this time step l all resource has been considered, I will just increase my time step to the next time step; that means, I am going to go to the next time step now. And I will keep doing it till my sink node is scheduled. So, that is the overall idea and once the schedule is done. So, I have the resource bound calculated.

So, this is the resource bound is this particular algorithm calculates. So, its not only give you the schedule the schedule is given by t, t is the time step t i. So, t i is the start time of every operations operation v i. So, it will give you the start time of each operations as well as the resource bound that it identified by this algorithm. So, I hope you understand the algorithm is very simple, but the basic intuition of increasing the resource from one to gradually something interesting to understand from here.

(Refer Slide Time: 19:32)



So, what I am going to do it? I am going to take that our favorite example the DIFT example and I will just explain this algorithm for this. So, let us say I have given the latency bound equal to 4 and you already know that for this algorithm if all operations are single cycle we I can schedule them in 4.

So, if I run ALAP it will say yes I can schedule this graph can be schedule in 4 time step. Now let us try to show how will this this MRLC algorithm works for this. So, initially assume there are two type of resources one is multiplier and one is ALU which is capable of performing this plus minus and greater than lesser than operation.

So, I have initially as I say I have only one instance of them and let us say let me try to give them ALAP value for this node. So, otherwise it will be difficult to explain. So, if you know. So, if there is lambda 4. So, I am going to schedule. So, this will be 4 I can schedule them in time step 4 I can schedule this. So, this is the ALAP value this is 3, this is 3, this is 3, this is 2, this is 2, this is 1, this is 1.

So, this is the ALAP time step. So, this is basically ALAP t i. So, the time step where this ALAP would have schedule this operations given my latency bound is 4. I hope you understand because it will take the behavior from the sink node all the nodes whose successors are already scheduled it will be scheduled in this time step 4 and then it will move to time step 3.

So, and it can schedule all the operations whose successor already scheduled and so on it will keep doing it and this will the time step I am going to get it. Now, let us start from time step 1. So, now, I am time step 1. So, in time step 1. So, you can understand that. So, you are in time step 1. So, my l equal to 1 and you can understand that for these two nodes TLAP value is also 1.

So, if we make 1 minus 1 it will 0. So, that mean these two node have a 0 slack it has to be scheduled in this time step so; that means, I have to increase my a 1 to 2 so, that I can actually schedule these two operation there and that is going to happen for multiplier.

So, this is how I am just I just increasing my resource from 1 to 2 in time step 1 because these two operations become critical. So, this is for operation time multiplier, then I am going to

take the operation time ALU and I have one operations available and which is not critical its a its I mean time step 1 and it can be scheduled in 3. So, if we just make a 3 minus 1. So, it is basically 3 minus 1 is 2 it has slack 2, but still this operation is available I have enough resource I am going to schedule it based on this idea.

So, I am going to schedule based on this idea [FL] if I have resource available and although there is no critical operation, but let us schedule it because this operation is available. So, I am going to schedule this node also in time step 1. So, in time step 1 I am going to schedule 1, 2 and 10 and I am assuming all operations are single cycle for simplicity now we will move on to the next step.

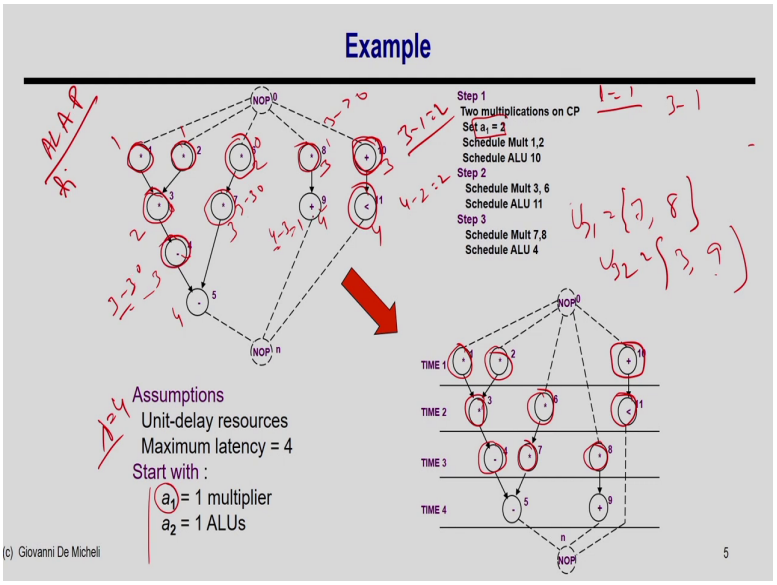
So, in next step what are the operation which is ready to be scheduled? So, I am just write it. So, U time step 2 operator time 1 is multiplier. So, you can see here 3, 6, 8 3, 6 and 8 is ready to be scheduled. So, among them and I am my l equal to 2. So, you can understand 2 minus 2 this slack will become 0, this slack will also becomes 0 and this slack will become 1; 3 minus 2.

So, these two operations become critical and I have to schedule this two operations. So, I just scheduled 3 and 6 here and I can actually delay this operation I do not want to increase my resource bound because my I have two multipliers. So, I can schedule them without increasing the multiplier bound, but say although I have one operation is available and can be ready to be scheduled, but since I do not have enough resource I am not going to schedule it here.

So, similarly if you calculate 22; that means, in time step 2 the operator type 2. So, only this this node is available and for this it is the 4 minus 2 is 2. See it is not a scheduler critical operation, but still since I have 1 ALU available. So, I am going to schedule it here. So, in the time step 2 I am going to schedule 3, 6 and 11.

So, now, I will move on to move on to time step 3. So, in the time step 3 you can understand that the multiplier which is available is this, and this and here you can understand 3 minus 3 equal to 0 it become critical, here also 3 minus 3 equals to 0 because its ALAP value t i is 3 and I am in time step 3.

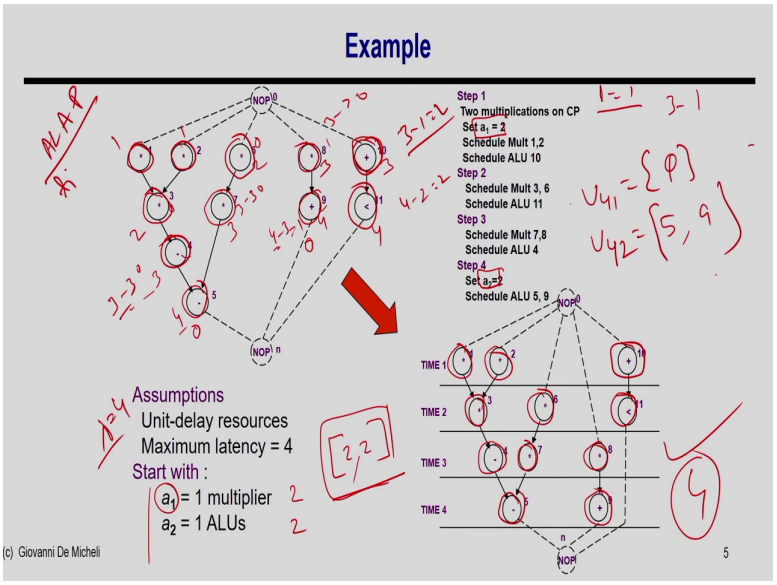
(Refer Slide Time: 24:20)



So, in this time step 3. So, U 3 1 is have 7 and 8 two multiplier both are critical I have two multiplier I am going to schedule both 7 and 8. What about these other two operation 3 2? I have this is available. So, 3 and 9 and I have 3 and 9 and if you I am in time step l.

So, this becomes critical. So, 3 minus 3 equal to 0. So, this has to be schedule, but this is still not critical because it is 4 minus 3 4 is the TALAP time and 3 is my current time step is 1.So, this is not critical operation. So, although there are two ALU operations ready to be schedule and I have one ALU available and only one of the operations is critical. So, I am not going to reschedule both I am going to schedule only 4.

(Refer Slide Time: 25:34)



So, I have just schedule it 4 here. So, this is also scheduled. So, this is not schedule. So, now, I am going to move on to time step 4 in time step 4 there is no multiplier. So, my 4 1; U 4 1 equal to null. So, there is no multiplier to schedule. So, although there are two multipliers they will be idle in this time step and then you just calculate the U 4 2 operation, type 2 I have 5 and 9 5 and 9.

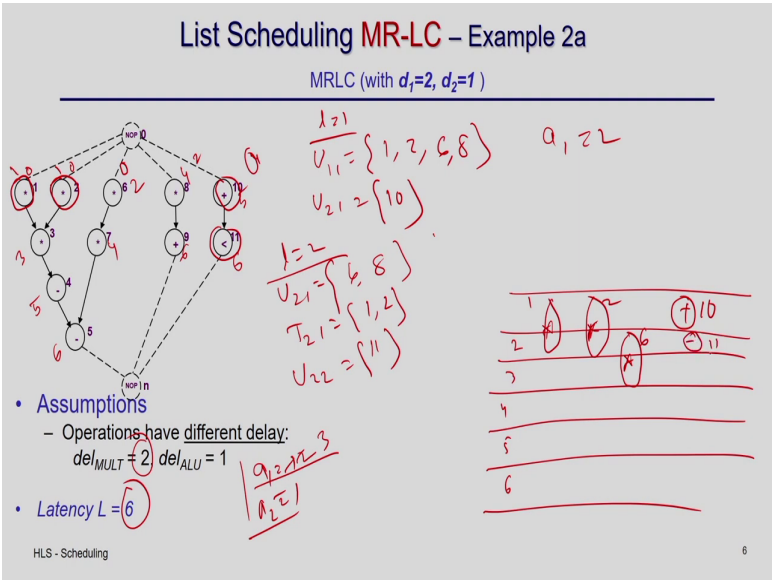
So, you can understand I mean time step l which is 4. So, it is a ALAP time is also 4. So, it is also become critical this also become critical. So, now, I have to schedule both of them, but I have only one ALU.

So, I will increase my ALU to 2. So, I will increase my ALU to 2 and I am going to schedule this and this in time step 5 and 9 so; that means, I have started with 1 1, but I end up having 2 2. So, it will say that I can schedule this behavior in time step 4, four time step and my resource become it is 2 2.

And since this is this is basically a tree I mean that I have already discuss in previous classes and this is the optimal solution. So, although this MRLC does not guarantee to give you the optimal solution, but I can prove even it can be proved that for this graph you need at least 2, 2 instances of resource to schedule this behavior in 4 time step you cannot do it less than this.

So, for this it give you the minimum, but minimum resource or optimal resource, but it does not guarantee to be give you that things for all scenarios.

(Refer Slide Time: 27:07)



So, let me now take the same example, but you assume that I have now a latency bound is given 7 and my multiplier is two cycle operations and this ALU is operation basically a single cycle operation. So, let me try to put the ALAP value. So, this will be 6, this is 6, this is 6 and this is 5 this will start at 4 because 4 5 it will complete its execution at 5 because its successor is have to complete at 6.

So, I have to schedule it such a way that it can complete its execution before 6. So, it has to be scheduled in 4. So, similarly this will be schedule in 4, this will schedule in 5 because it is a single cycle operation. Now, for this it will be scheduled in time step 1, this will be schedule in time step 2 this is a TALAP value I am calculating and this must be 1, 1. So, this is the TALAP value that I have calculated using the TALAP algorithm.

Now, let us try to understand that how I can schedule. So, if we just take l equal to 1 time step 1 and you calculate the U 11 the operator 1 and you can understand that there are a operation 1, 2, 6 and 8 are schedulable and you have given initially my a 1 equal to 1, a 2 equal to 1 I have 1, 1 resources.

So, I can found that there are 4 operation is ready to be schedule 4 multiplier operation ready to be schedule and out of them you are two are critical because it is 1 minus 1 is 0, 1 minus 1 is 0 for these two this is not critical. So, I will increase my 1 to 2 and I am going to schedule both 1 and 2 in this time step.

So, let me try to put. So, 1 and 2 this will be scheduled in time step 1 and since this is a. So, this is a single cycle operation. So, it will be like this. So, I can just. So, you have this 1 it will be scheduled in two cycles and I need two resource here. So, this is a multiplier this is 2. So, this is for this multiplier type.

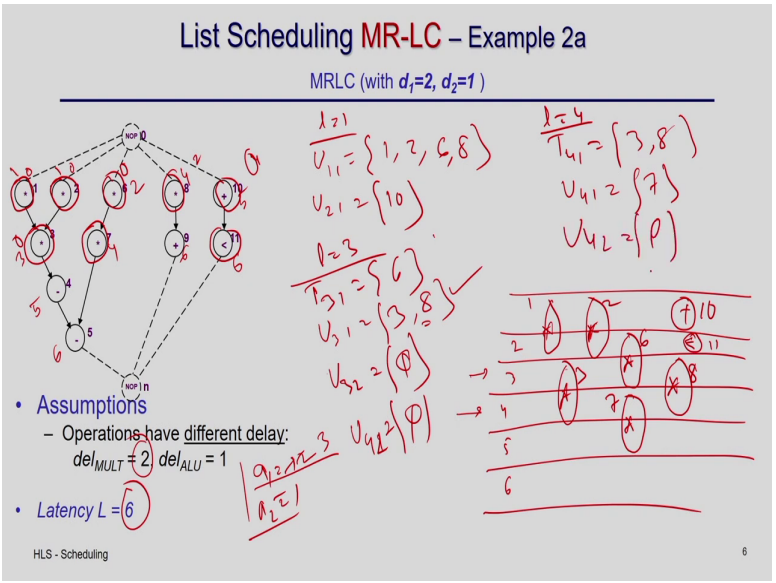
So, now I am going to calculate this U 21; that means, for ALU type and I am only 1 you know that and this is not a critical operation it has lot of slack, but I am in time step 1 it can be 5. So, it is it has a slack of 4, but since this resource available I am going to schedule it here also. So, 10 will be scheduled here.

So; that means, this is done, this is done and this is done in l equal to 2 I have U 21 I have what are the operation. So, you have to understand that this two operation is running here. So, this is not available here they are running in time step 2. So, U is again 6 and 8 right 6 and 8 and my T l k. So, T 21 which is running operation, which is 1 and 2.

So, I have already two operation is running and I have two multiplier available because I increase it to 1 to 2 and now let us calculate the slack. So, for this 6 it is now I am in time step 2 also. So, its the slack becomes 0 and for this slack is 2. So, still this is not critical, but 6 become critical. So, I have two a multiplier available and both the running operation 1 and 2 utilizing. So, I have to increase the resource bound to 3 because I cannot delay my operation 6 further.

So, I have to schedule my operation 6 here which is a multiplier and my resource multiplier become 3. So, this is very interesting and now for this U 22 which is this 22 is basically the ALU, I have only one operation is available because this is still not available because predecessor is not done and I have one resource available. So, I can schedule this. So, this 11 will be scheduled here. So, 11 is a less than operation. So, this is what happened after time step 2. So, let me now take time step 3.

(Refer Slide Time: 31:33)



So, in time step 3, l equal to 3 what are the running operations? Now, 1 and 2 completed. So, t e T 3 1 is only 6 because 1 and 2 is completed I mean time step 3 and what are the ready operations? U 31 once in two is completed. So, 3 become ready now 3 become ready and 6 is running. So, 4 cannot be ready because 6 is still running here, but 8 is still waiting. So, still it waiting.

So, I am in time step 3 and I found that one is running operation is there I have 3 multiplier available and among these two operations 3 is critical because I am in time step 3. So, 3 minus 3 is 0. So, this is critical I have to schedule it. So, I schedule 3 here 6 is running. So, 2 multipliers is getting occupied and I have one more operation which is not critical, but I have one multiplier available.

So, I am going to schedule 8 as well this is interesting. So, although 8 is not critical, but because it has a delay 4 minus 3 equal to 1, but since I have one multiplier available. So, I am going to schedule it as well. So, these 3 multipliers this all 3 multipliers. So, 3 and 8 both will be scheduled in time step 3.

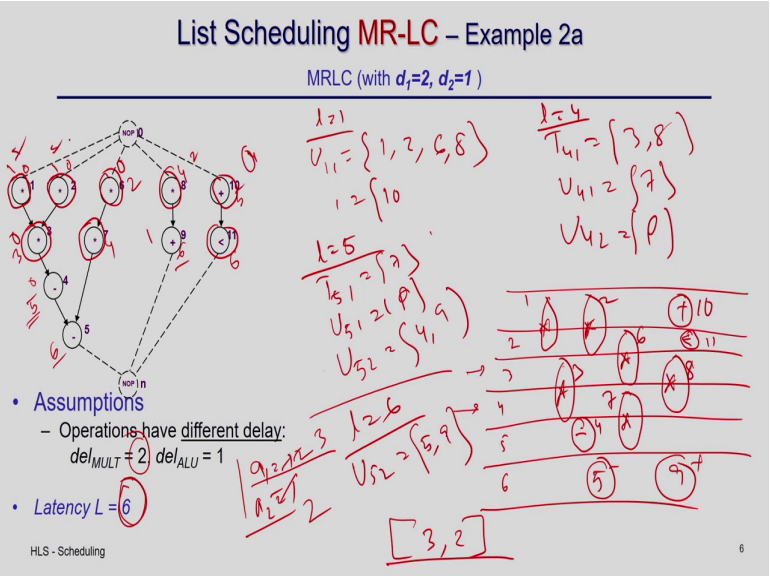
So, this is schedule. So, now, for the other type there is no t that running operations. So, U 3 2 what are the operation is running? So, there is no it is null, because this operation is just

schedule 6 is running and 3 is just schedule. So, this is null. So, nothing to be done. So, I just schedule 8 and 3 in this time step. In time step 4 similarly I can just calculate it here.

So, l equals to 4 what are the running operations? T 4 1 I have 3 and 8 both are running and since 3 and both are running, 6 has completed now. So, U 4 1 will be 7. So, 7 can be 7 can we scheduled now. So, 7 is there. So, you can understand that I do not have to check whether 7 is critical or not because 3 multipliers available.

So, I am going to schedule 7 here as well 3 multipliers available. So, I am going to schedule 7. So, 7 is done and other two is running and in this time step since in you are in time step 4 there is still I cannot my U 42 is basically null there is no ALU type operation here. So, U 42 is null. So, that mean this is what happened in time step 4. So, now, let me quickly schedule in time step 5 and time step 6.

(Refer Slide Time: 34:29)



So, now, in l equal to 5 in l equal to 5 these two multiplier is completed, but this T 5 1 is 7. 7 is still running, but there is no multiplier left I do not have to do anything. So, U 5 1 equal to null. I do not have to do anything for this, but since 8 have completed its execution here and 3 have also completed its execution in time step 4.

So, now, this t sorry this is U 5 2 there is no running operation per operation type 2 which is now 4 and 9 both are schedulable sorry 4 and 9 4 and 9. Out of that I can see that this is

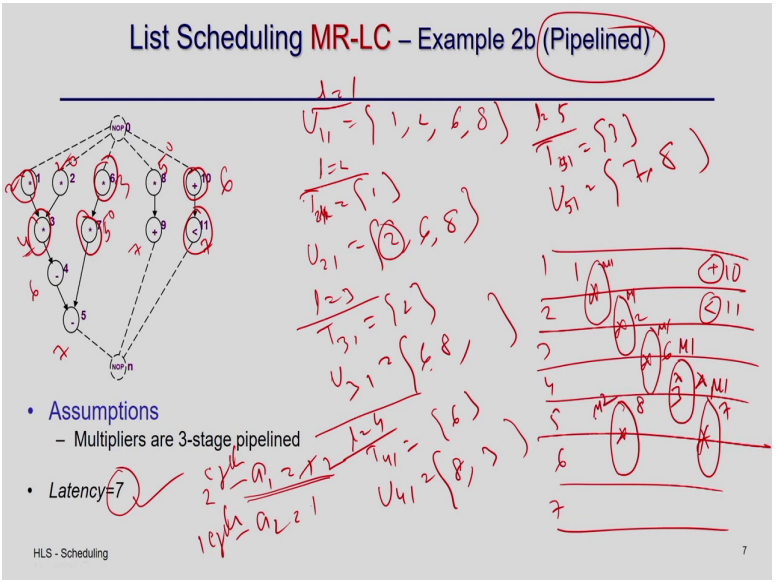
critical because I am in time step 5 and it is l is 5. So, it becomes critical. So, 4 must be schedule here. So, 4 is schedule here 4 is schedule here.

But for this node it is not critical because I am in time step 5 and this is 6 its ALAP value 6. So, it has slack one. So, I am not going to increase my ALU bound. So, I am going to just schedule 4 here and then time step 6, l equal to 6 there is no multiplier because all are done, but I have two operation U 5 2 which is basically this 5 and 9.

So, both 5 and 9 is critical I cannot go beyond that because both have 6. So, I have to increase my ALU bound to 2 and I have to schedule 5 and 9 here. So, 5 and 9 to schedule here. So, the final resource bound I got 3 2.

So, you can understand that to schedule this behavior in 6 time step when my multiplier is two cycle operations I need at least 3 resources 3 multipliers and 2 ALUs. Again, I can show that this is the minimum all because this is a tree its not a generic graph where there is multiple path from the one node to sink node.

(Refer Slide Time: 36:32)



So, this is how you can calculate and this is what the another one which is pipeline. So, pipeline I will just talk about only two step and then I will done. So, what I can do? I just take the latency value 7. So, this is 7, this is 7, this is 7, TALAP value this is 6 this is 5 because it

is two cycles. So, let us say assume that this multiplier is two cycle, this is 6, this is 5 and is this is basically 4. So, this is if it is times started 5.

So, it should be 3 and then if it is 5 this would be 2. So, one interesting point here there see since I have already discussed in the previous cycle that if the latency is 6, then these two operation become critical in time step 1 because they have to scheduled in time step 1 because I have given I just relaxed my latency a little bit.

So, there is no critical operation in time step 2, which was not the case in previous cycles. So, I can quickly do the job. So, in the first time step U 11 the operation again 1, 2, 6 and 8 available and say a 1 is 1 and a 2 is 1 one multiplier and one ALU and this a this a 1 is 2 cycle and this is 1 cycle single cycle operation. So, there are 6 multipliers available I am just going to quickly do it, but none of them is critical because I am in l equal to 1, I am in l equal to 1, but none of them is critical.

So, I am going to schedule one of them let us say I can here since there are 4 I am going to schedule 1 which has the highest distance the distance from the sink node and; obviously, 1 and 2. So, say I am going to schedule 1 here 1 will be schedule here because I do not have there is no critical operation. So, I am done with and; obviously, there is a ALU operator 6 which is basically same one. So, I am going to schedule this 6 as well.

Sorry, this operation 10 operation 10, but it has it can be scheduled till 6, but I am in time step 1, but since this is available I am going to schedule it. So, this is I am done with time step 1. So, in time step 2. So, my T l k is 1, 1 is running here, but its a pipeline one. So, you have to understand this.

So, what is my U 21? What are the operations? So, 2, 6 and 8 available the others are not available because the now I have these two which become critical because now, I am in time step 2 these become critical. So, two become critical and this 6 and 8 is not critical, but since this is a pipeline one. So, I can actually do this using the same multiplier.

So, I am going to use the same multiplayer because this is my multiplier 1 this is also multiplier 1 because this is the pipeline circuit this is what I am going to do. So, I am not

increasing my a 1 this is very interesting to note that I do not increase my resource bound because although there is a critical operations I can pipeline with the previous multiplier.

So, I am just doing that. So, I will not do anything. So, I will just do that and similarly I will just schedule this v 11 here for ALU 2. So, this is done in l equal to 3 my running operation is T 3 1 is 2, one is done and U 3 1 what are the operations? 6, 8 and now since this is done 3 is also available. So, in you are in time step 3 and only 6. So, this is 4. So, this is not critical this is critical 6 become critical. So, 6 become critical and so, 5 is also not 8 is also not critical.

So, I am going to schedule 6 again only one critical operation I am not going to increase my bound. So, 6 again I am going to pipeline in the same multiplier. So, this I am going to do and there is no ALU operation available. So, I am not going to do that. So, 6 is done in this time step. So, in l equal to 4 now, l equal to 4 my running operation is 6. So, T 4 1 equal to 6 and what are the ready operation? U 4 1 8 and 3 is already available.

So, 8 and 3; 8 and 3 is available and 6. So, 2 is completed. So, I am actually made a mistake here. So, in time step 3 this 3 is not available because the 2 was running that time sorry so, but it does not change anything 6 will be scheduled there. So, now, since two is done in time step 4, now 3 is available. So, 3 is available here. So, 8 and 3 is available you are in time step 4. And in time step 4, 3 become critical because its ALU value is 4 and 6 is already running.

So, 6 is already running. So, I can pipeline this operation 3 using the same multiplier. So, this is again 3 I am going to put it here again multiplier 1. So, these are all pipelined. So, in l equal to 5, you can understand that 6 will complete in time step 4, there is no ALU operation nothing will be scheduled.

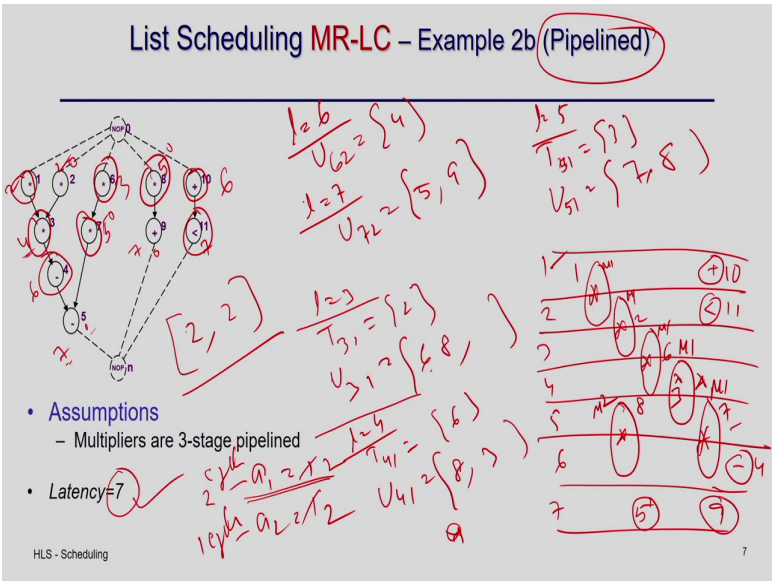
In time step 5, my running operation is 3. So, T 5 1 equal to 3 and U 5 1 what are the ready operations? 6 is done. So, 7 become ready now 6 is done. So, 7 become ready and 8 is still not scheduled. So, 7 become ready, 3 is running. So, I can again schedule this 7 here. So, what I am going to do it 7 here. So, that mean it is very interesting. So, I am just still using one multiplier I am able to do it.

And 7 I am going to do it because you are interested for 5 and this is critical. So, this is critical. So, 7 and 8 both are available. So, now, 7 and 8 both available you are in time step l

and both are critical because slack is 0 slack is 0 TLAP minus this. So, I have to use one more multiplier now. So, this is my multiplier 2 where I am going to schedule 8. So, I will increase my bound to 2.

So, this will start with time step 5 and it will complete in I am in time step 5 sorry. So, I am in time step 5. So, it will be scheduled like this. So, this is my 8 multiplier and this is m 2 because this multiplier is already scheduling this. So, I have to do it in other. So, 8 and 7 is done here now if I just do for l quickly finish because not much left. So, now in l equal to 6.

(Refer Slide Time: 44:08)



l equal to 6 since in my I mean l 6. So, now, 3 is done. So, 4 can be scheduled and others cannot be scheduled because 8 is still running. So, I am just going to do in time step 4 there is no multiplier will be scheduled and I can actually do it with 1 ALUs because there is 11.

Now, if you take l equal to 2. So, here I am just have this U 6 2 which is basically only 4 and it will be scheduled and then l equal to 7 I have no other multiplier available, but in l equal to 7 now 8 and 7 both are done. So, now, I can schedule 5 and 9. So, both have 7. So, my U 7, 2 has 5 and 9 and both are become critical because you are in time step 7 and both has slack value ALAP value 7. So, both are critical.

So, I have to increase this by 2 and I am going to schedule both 5 and 9 here. So, you can understand that I can actually do it in 2 multipliers and 2 ALU I can schedule this operation

using 2 multiplier and 2 ALU in 7 cycle. So, if my latency is not 7 probably I would have taken more multiplier, but since I have some relaxation in the latency I have used less resource than this one. So, here I have used 3 2 here I have used 22 and here the very interesting thing is that one multiplier was pipelined continuously from time step 1 to 5 and its just executing different multiplier and its pipeline 1.

So, with this I hope it is clear to you how this MR-LC will work with this. So, with this I conclude today's class. So, we have completed the list based scheduling algorithm for both MRLC and MRLC problem. So, in the next class I am going to talk about another heuristic which is called force directed scheduling.

Thank you.