

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

Module - 02
C-Based VLSI Design: Scheduling
Lecture - 07
ILP formulation of MLRC and MRLC Scheduling

Welcome students. In today's class we are going to learn ILP formulation of MLRC and MRLC Scheduling. So, in the last class we have seen that the how we can model the scheduling problem using integer linear programming.

So, just to recap what we have learned in the last class that the scheduling is a problem where we want to schedule the operations; that means, we want to assign the time step to each operation.

(Refer Slide Time: 01:21)

Integer Linear Programming (ILP)

- Given:
 - integer-valued matrix $A_{m \times n}$
 - variables: $x \in (x_1, x_2, \dots, x_n)^T$
 - constants: $b = (b_1, b_2, \dots, b_m)^T$ and $c = (c_1, c_2, \dots, c_n)^T$
- Minimize: $c^T x$

subject to:

$$\begin{cases} Ax \leq b \\ x = (x_1, x_2, \dots, x_n) \text{ is an integer-valued vector} \end{cases}$$

- If all variables are *continuous*, the problem is called linear (LP)
- Problem is called *Integer LP (ILP)* if some variables x are integer
 - special case: 0,1 (binary) ILP

2

And, what is the ILP? The ILP is basically we have set of unknown variable say x and I have set of constant on this variable x and that is given by this matrix A and this con A and b . So, which is given by these relations and you have set of unknown variables and you have an

optimization objective. You want to optimize certain expression which is basically a linear expression about these variables.

And, the ILP solver actually give you the values of those unknown variable which meets all this constraint and, give you the minimum value of the objective function. So, this is what, is the ILP.

(Refer Slide Time: 02:00)

ILP Model of Scheduling

- Binary decision variables x_{ij}

$x_{ij} = 1$ if operation v_i starts in step l ,
 otherwise $x_{ij} = 0$
 $i = 0, 1, \dots, n$ (operations)
 $l = 1, 2, \dots, \lambda+1$ (steps, with limit λ)

Q1: How many Binary variable do we need?
 Q2: Can we use ASAP and ALAP scheduling information reduce the decision variable?

And, then what we have seen in the last class that to model scheduling using ILP, we so, we have to define that unknown variable first. And what we have seen that this basically the since this our unknown things is that I do not know the time step where this operation is going to be schedule.

So, I just define a variable x_{il} which basically says that if a variable operation v_i , scheduling time step l schedule in the sense starts its execution start time then this x_{il} will be 1 for that variable, this operations other it will be 0. So, in general if there are n such operations and l number of time steps is available x_{il} this l into n number of variables are needed.

Then we have discussed that we can use the as ASAP and ALP bound because that is the range where this operation can be scheduled. So, by that way I can reduce the number of unknown variables.

(Refer Slide Time: 02:58)

ILP Model of Scheduling - Constraints

- Start time of each operation v_i is unique:

$$\sum_l x_{il} = 1, \quad i = 0, 1, \dots, n$$

Note: $\sum_l x_{il} = \sum_{l=t_i^S}^{l=t_i^L} x_{il}$

where: x_{il}

t_i^S = time of operation i computed with *ASAP*
 t_i^L = time of operation i computed with *ALAP*

Start time for v_i :

$$t_i = \sum_l l \cdot x_{il}$$

4

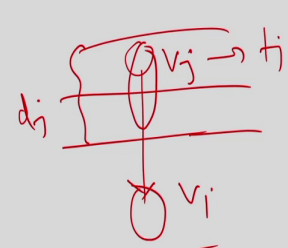
So, once we have developed the ILP model with these unknown variables then we have to see what are the kind of constraints which are applicable for scheduling and we identify four such constants, the first one is the start time. So, we specify that each operation should start exactly in one time step.

So, that means, although the ILP model by these variables says that it can start in any time step, we use the start time constraints to ensure that it has actually scheduled only one time step. So, it can start only in one time step and which is called unique start time.

(Refer Slide Time: 03:30)

ILP Model of Scheduling - constraints

- Precedence relationships must be satisfied

$$\sum_l l \cdot x_{il} \geq \sum_l l \cdot x_{jl} + d_j, \quad i, j = 0, 1, \dots, n : (v_j, v_i) \in E$$


The diagram shows a node v_j with a duration d_j and an arrow pointing to a node v_i , illustrating a precedence relationship. The duration d_j is shown as a horizontal bar below the node v_j .

5

So, and then we have discuss about this precedence relations. So, basically the idea is that if you and dependency between the node v_j and v_i . So, and if it is executing in t_j then this v_i can only start once this particular operation is finished. So, if it is the operations the time taken by this operation is d_j , the delay of this node and then only I can schedule this operations. So, this is the precedence constant.

(Refer Slide Time: 04:01)

ILP Model of Scheduling - constraints

- Resource constraints must be met
 - let upper bound on number of resources of type k be a_k

$$\sum_{i:T(v_i)=k} \sum_{m=l-d_i+1}^l x_{im} \leq a_k, \quad k = 1, 2, \dots, n_{res}, \quad l = 1, 2, \dots, \bar{\lambda} + 1$$

6

And, then we have this resource constant as well that at every time step the number of operations running must be less than of the number of resource available. And, here because it's the number of resource can have a different type so, it is basically for each type of resource the number of operations that are using this resource in a particular time step should be less than equal to the resource bound.

(Refer Slide Time: 04:34)

Operation v_i Still Running at Step l ?

- Is v_i running at step l ?
 - Is $x_{i,l} + x_{i,l-1} + \dots + x_{i,l-d_i+1} = 1$?

$x_{i,l} = 1$ $x_{i,l-1} = 1$ $x_{i,l-d_i+1} = 1$

So, this we have understood and basically here there is one more concept comes into picture that although I identify the start time of the operation because it's run for multiple cycle. So, even if it starts in this time step it will use the same say this is a multiplier. So, this is going to use the multiplier in all four time step. So, one multiplier will be required in all four time step because this is something a 4-cycle say operation.

So, the concept here is that the its although this operation is starting in a particular time step, it will occupy a multiplier in every clock where it is actually running.

So, that is something was given by this equation that all the operations of type k that are running in a particular time step l that you can identify by this equations and then I say that so, of type k the number of operations that are running in a particular time step must be less than of the resource bound and this is for each type of resources. So, this is what we have on the resource constraint.

(Refer Slide Time: 05:30)

ILP Model of Scheduling - Constraints

- Latency bound must be satisfied
- $\sum_i l x_{it} \leq \lambda + 1$

8

And, then we have also found the there is a another possible constraint is the latency bound suppose you have given a particular bound on the latency and you must ensure that the sink node must schedule on or before of the latency bound. So, this is actually given that x_{n1} will give the start time of the node the sink node and we are saying that sink node should must be less than of the latency bound.

(Refer Slide Time: 06:01)

Scheduling Problems

- Minimum Latency Unconstrained minimum-latency scheduling problem (Unconstraint) - ASAP
- Minimum latency under resource constraints (MLRC)
- Minimum resource under latency constraints (MRLC)

IIT Guwahati 9

So, this is what the constraint that we have identified and then in today what we are going to learn is something we will so, we will see how we can model specifically this minimize latency under resource constraint or minimize resource under latency under scheduling problem using ILP.

(Refer Slide Time: 06:21)

Minimum latency under resource constraints (MLRC)

- Given a set of ops V with integer delays D a partial order on the operations E , and upper bounds $\{a_k; k = 1, 2, \dots, n_{res}\}$ on resource usage:
- Find an integer labeling of the operation $\varphi : V \rightarrow \mathbb{Z}^+$ such that :
 - $t_i = \varphi(v_i)$
 - $t_i \geq t_j + d_j$ for all i, j s.t. $(v_i, v_j) \in E$,
 - $|\{v_i | T(v_i) = k \text{ and } t_i \leq l < t_j + d_j\}| \leq a_k$ for all types $k = 1, 2, \dots, n_{res}$ and steps l

and t_n is minimum

sink time sink node

10

So, let us just understand or recap what was the MLRC, is basically is a minimize latency under resource constraint; that means, you have given a resource bound. The resource bound is available and then you try to schedule the operations using minimum number of latency, that was the MLRC problem.

So, what was given to us? We have given a set of nodes the operations and it has a delay. So, each node has some specific delay and then with their dependency is actually given by the h. So, we already have the sequence graph that is actually capturing these nodes and their dependencies and the delay is something is dependent on the how many clock cycle it needs to execute that operations.

And, you can have many each type of operator has a different delay and also we have given a upper bound of the each type of resource. So, here I just again try to explain that you can have a multiplier type of resource divider, adder and so on. So, for each type of resource what

is the number of resource available? So, that is given. So, if there are n res type of resource so, this is a_k is give the type of resource for kth type of I mean k type of resource.

So, now this what this minimum this MLRC scheduling does? It ensures that it actually assign time step to each node. So, the t_i is the start time of the node v_i such that it obviously, if satisfy the dependency constant and such a way this is going to assign the t_n which is the start time of the sink node start time of the sink node. So, it will be minimum.

So, if you understand that if this sink node start time is minimum if this is minimum and because the rest of the nodes actually must be scheduled before this because of the data dependency and so obviously, this operation must be scheduled before this time step and if this is minimum, this is the minimize this is the best solution which actually has the minimum latency.

(Refer Slide Time: 08:31)

Minimum Resource under latency constraints (MRLC)

- Additional constraint: Latency λ
 - Latency bound must be satisfied
- Resource usage is unknown in the constraints (a_k is unknown)
- Objective is to minimize resource usage such that
 - Dependency constraints are satisfied
 - Latency constraint is satisfied

$c = [0 \ 0 \ 0 \ \dots \ 1]$

Min $c^T \cdot k$

$(c_0 \times b + c_1 \times h + c_2 \cdot k_1 + \dots + c_n \cdot k_n)$

IIT Guwahati 11

So, this is what is the MLRC problem. So, now, let us what is the other variance. The dual problem was the MRLC that here the latency bound is given. So, I know that I need to execute things in some lambda bounds and I want to schedule the operation you within the lambda time step use a minimum number of resource our objective is to minimize the resource.

So, here the resource bound that a_k in the previous expression that this k is not known. I need to identify this. So, that was something is the problem here. But, the λ is given, the latency bound is given and you try to find out a solution which actually must satisfy this latency bound and minimize this resource constant. So, that is what is the problem.

(Refer Slide Time: 09:12)

Minimum-Latency Scheduling under Resource Constraints (ML-RC)

- Let \mathbf{t} be the vector whose entries are start times
 $t = [t_0, t_1, \dots, t_n]$
- Formal ILP model
 minimize $\mathbf{c}^T \mathbf{t}$ such that
 $\mathbf{c} = [0, 0, \dots, 1]$
- ① $\sum_l x_{il} = 1, i = 0, 1, \dots, n$
- ② $\sum_l l \cdot x_{il} - \sum_l l \cdot x_{jl} - d_j \geq 0, i, j = 0, 1, \dots, n : (v_j, v_i) \in E$
- ③ $\sum_{i:T(v_i)=k} \sum_{m=l-d_i+1}^l x_{im} \leq a_k, k = 1, 2, \dots, n_{res}, l = 1, 2, \dots, \bar{\lambda} + 1$
- ④ $x_{il} \in \{0, 1\}, i = 0, 1, \dots, n, l = 1, 2, \dots, \bar{\lambda} + 1$

Handwritten notes: $v_i \rightarrow (t_i)$, $x_{il} = 0$, $\lambda = ASAP$, $\lambda = ALAP$, $\lambda = ABAP$, 24 , 12 .

Now, let us try to model this both problem using ILP. So, in the ILP so, we have a objective function. So, let us try to define the objective function for this MLS MLRC scheduling problem. So, if you remember that this t_n is something is the a vector which is actually consists of t_0 to t_n and these are the start times of operation v_i .

So, t_i is the start time of the operation v_i and you remember because in the ILP formulation I do not have any t_i I have only this x_{il} and we have say; we have say told that this l is basically from ASAP to ALAB. So, and if I just multiply this will given my t_i .

So, you remember that is I we have explained in the previous class that we can represent this t_i in terms of x_{il} because that is the variable that I have in the il and we are saying that this x_{il} if I take a summation from this l which is starting from t ASAP to ALAP and if I multiply that x_{il} with l so, that will give me the t_i because it is only so, one of this x_{il} will be 1.

So, because it is it has a unique start time and corresponding l is the value where it actually schedules. So, that is how I can get the t_i . So, this t_i is something is basically we try to

identify and then what is our objective function I want to optimize the latency. So, then I have already explained that if the optimization of redundancy of the whole behavior is nothing, but optimizing the latency the start time of the last node or the sink node.

So, although in the generically ILP this optimization function is given by c^T into T that c^T is a vector of constant and for this particular optimization problem MLRC this constant should be $0, 0, 0, 1$ why? Because I want to mini. So, if this basically c^T is it will be given by this that says you have this c^T into t . So, basically it is c_1 into t_1 plus c_2 . So, I think it should be t_0, c_0 then c_1 into t_1 plus c_2 into t_2 .

And, it will be c_n into t_n this is the my minimization function you try to find the value of this x_i . So, you can think about this t_i , t_1 is nothing, but the expression over x_i .

So, I want to minimize this expression and I have already told that for this MLRC this optimization is nothing, but finding the minimum value of this t_n . So, I can assume this c_1 is equal to 0, c_2 equal to 0, these are all 0 this is the only 1. So, my c is a vector which is all 0 only the last entry is 1.

So, that something is the constant value. So, effectively that this c^T is become that minimize this t_n . So, this is my objective function this become my objective function effectively. So, we understand what is the objective function for MLRC and now, let us try to see what are the constant is applicable for this particular problem. So, obviously, this unique start time must hold.

Because unique start time is something obviously, hold because you cannot execute one operation multiple times, it will only start only once. So, the unique start time constant must hold. Obviously, the data dependency must hold whatever schedule you generate it must hold the data dependency and the resource bound must hold.

So, that resource bound is given by this that every time step the operations the running operation of type k must be less than of the number of resource available in that particular time step. So, what we have understood that this unique start time must hold then data dependency must hold and the resource bound each time step must hold.

So, these are the three constants we have to specify and this is the objective function that you minimize the value of the t_n the start time of the sink node and that is what is the ILP formulation of this if I give it to some ILP solver it will give me the solution.

So, one question it must come to your mind that so, in this case my lambda or the latency bound is not available then where I am getting this lambda. So, I am writing here this lambda plus 1 lambda plus 1 and if you remember that I have defined a variable called x_{il} which is basically i is running from 0 to n and then rl, l is the latency bound which is basically from 0 to lambda plus 1.

So, 0 is for the source node and lambda plus 1 is for the sink node and 1 to lambda is for the operations. So, unless I define or I give a fixed value of the lambda plus 1 I cannot define this x_{il} at all. So, you should ask me this question that why how you then model how this model actually works. So, this is very interesting question.

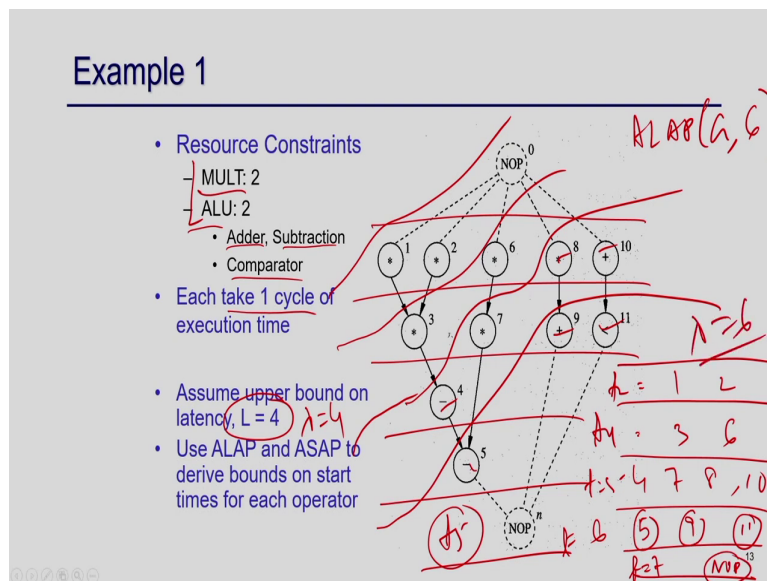
So, to answer this question is basically yes, it is true that this ILP modeling demands a specific value of the lambda and since we are try to optimize the lambda how we can a give a value of a latency. So, that is something we should understand.

Now, let us try to understand for a given sequence graph if I run this ASAP algorithm as soon as possible algorithm it will give you the minimum number of latency. So, it is give the lower bound of the latency because it is using unrestricted amount of resource. So, that means, you cannot have any solution that can schedule this particular sequence graph less than the ASAP time given by the ASAP algorithm.

So, for a given behavior I can actually identify the lower bound and what ALAP does? So, in the ALAP you need may not you do not need to give the lower bound of this what you have obtained from the ASAP rather I can give any value of the lambda.

But, the lambda that I am going to give for as ALAP that must be greater than equal to of the lower bound that is obtained by the ASAP algorithm. So, if I give any such value what will do? ALAP it will actually give the lower bound or the latest possible place where this operation can be scheduled.

(Refer Slide Time: 15:57)



So, suppose you take this behavior. So, you know this it can be scheduled in 4 times and this is the actually the ASAP solution. Now, if I run ALAP with this sequence graph and say with 6. See if I run it with 6 it will give a different solution.

For example, if I run with 6 so, in the 6 clock I can run. So, this is my time step 6, I can execute node 5, 9 and 11 and then if it is t equal to 5 what I can do? So, this and these are done then I can run 4, 7, 8 and 10. So, this I can do in the next time step this and this. So, basically this is my ALAP solution. So, I this is my ALAP solution and then the next clock I can run this and this is my unconditional.

So, if I just do this my t_4 will consist of 3 and 6 and t_2 will constraint 1 and 2. So, this is the ALAP I am going to get. So, you can understand that since I have given 6, but for this particular viewer I need only 4 time step which is given by the ASAP. So, I can have a different bound.

Now, you understand that I have not tried to solve a problem where I give a resource bound. So, its not a unrestricted resource. So, obviously, for a given resource bound your the minimum time step which is basically the obtained by ASAP may not be achieved. So, to achieve this ASAP this the latency bound that is obtained by ASAP you might need more resources.

So, that means, in this particular case because I do not know this what is my exact latency bound what I can do the algorithm the way I can do I can start modeling this MLRC using the latency bound which is similar equivalent to the ASAP.

So, if I have a solution for this latency bound so, that means, this is so, the resource bound that is given to me with that this let minimum latency bound is achievable say. So, which is basically possible?

And, say if I just use a lambda equal to 4 in this example and I run the ALAP and I do not get any solution ALAP says solution. ALAP says that there is does not exist any solution for this modeling with lambda 4; that means, a given resource bound cannot meet this latency bound minimum latency bound. So, I can now take 5 and actually model the whole thing and I can run it. So, this is how I should go.

So, here although this is MLRC resource bound is given I am trying to identify this ml this minimum value of the latency, but I cannot just model the whole thing with a variable bound of latency because then the number of variable will be infinite because I do not know how many variable we need.

So, the approach that should I should take for this particular problem is you take a specific value of lambda and see whether you try to whether able to get a solution or not. If solution exists then that is the minimum this is the minimum solution for.

So, that means, that solution actually meets this lambda what is the latency bound is given to you with the given set of resources. If it does not then you increase the latency bound by say 1, 2 whatever your factor and you again run the ILP and that will give you the solution. I hope it is clear to you.

So, now let me take this example again and say I my resource bounds I have 2 type of resource one is the multiplier and we have the ALU and ALU you can do this addition, subtraction, comparator operations and multiply due to the multiplications. So, in this behavior we have only I have plus, minus, multiplication and comparator. So, I have two type of resource and I am assuming for simplicity that every operation is single cycle operations.

But, you can actually try the same thing with multi cycle operations as homework. And, say my latency bound lambda is given as 4. So, which is the minimum value of for this example that I have already discussed that ASAP will give you 4. So, what I am try to solve by this modeling that can this minimum latency bound can be achieved using two multiplier and two ALU for this given sequence pair.

And if yes, so, if ILP return something it will say yes I can achieve 4 for the latency bound 4 with this resources.

(Refer Slide Time: 20:40)

Example 1 (cont'd.)

- Start time must be unique

Recall: $\sum_{l=1}^L x_{il} = \sum_{l=1}^S x_{il}$

where:
 $t_i^S = t_i$ computed with ASAP
 $t_i^L = t_i$ computed with ALAP

$\lambda = 4$

$x_{0,0} = 1$
 $x_{1,1} = 1$
 $x_{2,1} = 1$
 $x_{3,2} = 1$
 $x_{4,3} = 1$
 $x_{5,4} = 1$
 $x_{6,1} + x_{6,2} = 1$
 $x_{7,2} + x_{7,3} = 1$
 $x_{8,1} + x_{8,2} + x_{8,3} = 1$
 $x_{9,2} + x_{9,3} + x_{9,4} = 1$
 $x_{10,1} + x_{10,2} + x_{10,3} = 1$
 $x_{11,2} + x_{11,3} + x_{11,4} = 1$
 $x_{n,5} = 1$

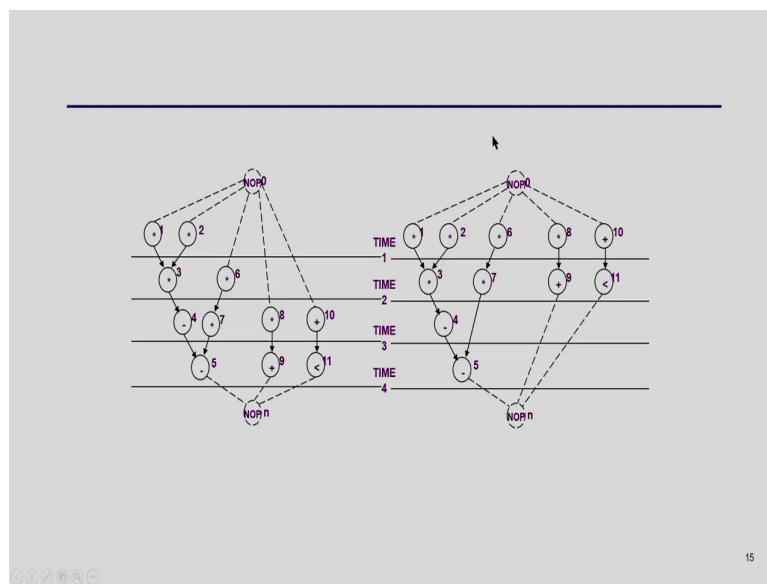
So, let us go quickly. So, this example I have already explained in the previous class. So, I will go little bit quickly that for this we have to identify the unique start time and for that since my lambda equal to 4 you have to always keep in mind that. So, we have to identify what are the time steps their particular variable can execute and you know the mobility of this operation.

This operations are basically 0 because if you have a four time step operation there is no mobility for them. So, this x 0,1; so, that means. So, this is for x 0,0 and this is x 1,1 which is basically for this node; that means, this node must be schedule in time step 1. This is x 2,1; that means, node 2 must schedule in time step 1.

3 must be schedule in time step. So, operation 3 must schedule in times time step 2, operation 4 must schedule in time step 3 and operation 5 must schedule in time step 4 and this 6 can schedule here and here so that is given by this. Either 6 schedule in time step 1 or scheduling time step 2, but only it can schedule one of the places.

So, that is why this plus this equal to 1. Similarly, 7 can schedule here or here. So, it is 7 either time step 2 or 3; operation 8 can schedule here or here. So, either 8 can schedule in time step 1, 2 or 3 and for the others. So, this is how I model the unique start time.

(Refer Slide Time: 22:09)



(Refer Slide Time: 22:11)

Example 1 (cont'd.)

- Precedence constraints
 - Note: only non-trivial ones listed

$$\begin{aligned}
 & (2x_{7,2} + 3x_{7,3}) - (x_{6,1} + 2x_{6,2}) - 1 \geq 0 \quad d_7 \\
 & 2x_{9,2} + 3x_{9,3} + 4x_{9,4} - x_{8,1} - 2x_{8,2} - 3x_{8,3} - 1 \geq 0 \quad d_6 \\
 & (2x_{11,2} + 3x_{11,3} + 4x_{11,4}) - (x_{10,1} + 2x_{10,2} + 3x_{10,3}) - 1 \geq 0 \quad d_{11} \\
 & 4x_{5,4} - 2x_{7,2} - 3x_{7,3} - 1 \geq 0 \quad d_{10} \\
 & 5x_{n,5} - 2x_{9,2} - 3x_{9,3} - 4x_{9,4} - 1 \geq 0 \\
 & 5x_{n,5} - 2x_{11,2} - 3x_{11,3} - 4x_{11,4} - 1 \geq 0
 \end{aligned}$$

$$\sum_l l \cdot x_{il} \geq \sum_l l \cdot x_{jl} + d_j, \quad i, j = 0, 1, \dots, n \quad : (v_j, v_i) \in E$$

16

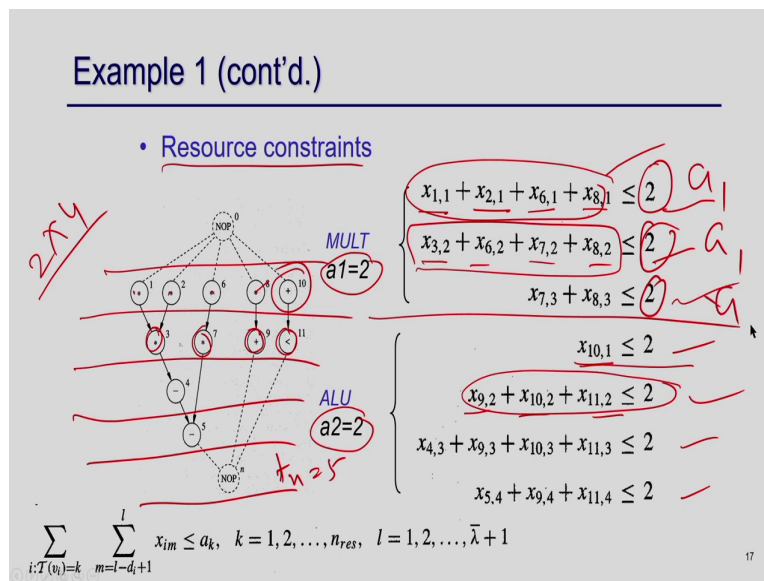
And, then this precedence constraint so, for each edge I have to take one edge at a time and I have to say that this the start time of this node must be greater than equal to the start time of this node plus the delay of this node. Since I here the delay is 1 so, every time I will going to use the delay 1. So, let us take this node. So, the 7 6 6 to 7. So, that means, the start time of this node 7, 2 so, which is basically either 2 or 3 that we have already done.

So, this is I have already discussed in the previous slide that this operation 7 can schedule either in terms 2 or 3 and this is the possible start time of node 6, it is either in time step 1 or time step 2. So, and this is the delay of the node d_6 . So, this is basically t_6 and this is basically t_7 . So, we are this expression saying that the operation 7 can only start when t_6 actually execute complex execution. This must be greater than equal to 0.

So, similarly you can take this say this node say 11 to 10. So, this is actually 10, this is 11. So, what we are saying that this operation can schedule in 2, 3 or 4. So, this is the start time of node 11. So, this is my t_{11} . This is the start time of the node t_{10} . It can schedule either 1, 2 or 3 time step and I am saying that and this is the delay of node this is d_{10} .

I am saying that this constant saying that operation 11 can only start his executions 1, t_{10} is actually complex execution. This is how for each edge I can have a constant.

(Refer Slide Time: 24:04)



So, similarly we have a resource constant because I have multiplier 2 and I have ALU 2. So, what so what I have to say? This is basically for each time step. So, how many resources will be there? So, there are 2 type 2 resources and for each time step 4 time steps are there lambda equal to 4. This many this is a constant will be there for this particular constant.

So, for example, if you take this say time step 1. So, in the time step 1, how many multipliers can be scheduled 1, 2, 6 and 8. So, what I am saying that if operation 1 schedule in time step 1; operation 2 schedule in time step 2; operation 6 schedule in time step 1 or operation 8 schedule in time step 1. So, this 4 is the possibility, but at max 2 can be scheduled. This is nothing, but like this.

Similarly, in the time step 2 either this 3, 7, 6 and also can schedule because 6 can be scheduled here, 8 can also be scheduled. So, either 3, 7, 6 and 8 any of these four can be scheduled in time step 2, but only maximum 2 can be scheduled. Similarly, you can actually add for time step 3 and time step 4. So, similarly I can actually add as for time step 4, there is no multiplier is possible. So, there is no constraint for that.

And, this is for adder. So, if you take this time step 1, only this operation can schedule. So, I am saying that operation 10 must be scheduled. So, these are going to always schedule in

time step 1 because the resource is available. In time step 2 this can schedule, this can schedule, 10 can schedule.

So, 9, 11 or 10 can schedule, but only two of them can be scheduling time step 2. This is how I can actually add the constant in it is time step and this is my resource constant.

(Refer Slide Time: 25:56)

Example 1 (cont'd.)

- Objective function for MLRC: $F = c^T t$
- $F1: c = [0, 0, \dots, 1]^T$
 - Minimum latency schedule
 - When $\lambda = 4$, since sink has no mobility ($x_{n,5} = 1$), any feasible schedule is optimum.

Handwritten notes on slide: $\text{Min}(t_n)$, $\lambda = 4$, and $\frac{\lambda = 6}{\lambda = 5}$.

So, and then what is the objective function I have already told you that my objective function is to just minimize the t_n which is nothing, but you just you just assume my the c equal to all 0, but the last bit is 1, then this is my problem. So, if I just give this set of equations this unique start time, then this resource precedence constant and this resource constant and if you just said this objective function equal to this then you will get a schedule.

Since here this λ equal to 4 equal to be the minimum value I already know this is the minimum latency. So, any feasible solution is optimum solution because the mobility of the node t_n or the sink node is 0 because in this case if there are four timestamp. So, this must be scheduling t_n must be 5; because it cannot go off, but the example that I have taken earlier that.

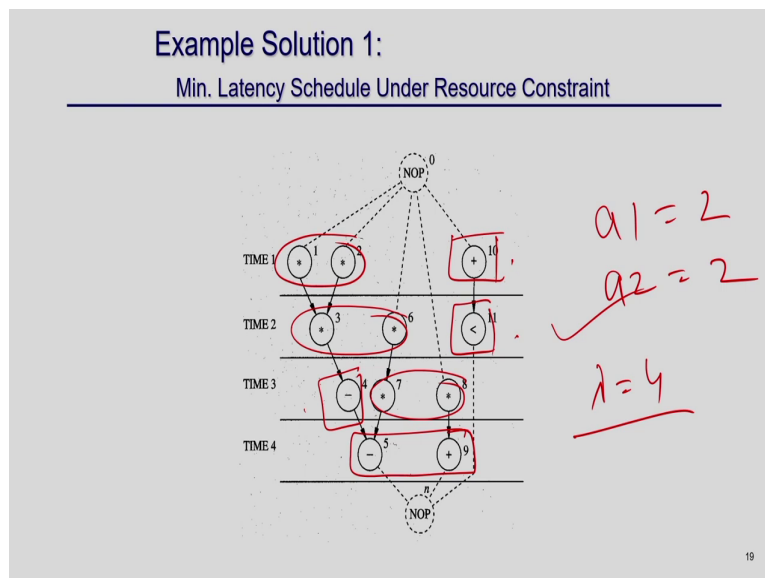
So, here you have mobility for each node because you schedule this 1 and 2. So, in ASAP it is scheduling 0. So, all nodes have some kind of mobility because this t_n will be scheduling

t5 in ASAP and t7 in the ALAP. If you assume that lambda equal to 6. So, in that case you have mobility and that time in it is not the any solution is feasible.

So, but if you have lambda equal to 4 because that is I know the minimum solution and for that there is no mobility of the sink node.

So, any solution, any feasible solution is optimal solution, but if lambda equal to say 6 or lambda equal to 5 you will have mobility for the sink node and that time not all solutions are minimum, but only the solution that will be written by ILP will be the minimum solvency solution so, for the given set of resource bound.

(Refer Slide Time: 27:44)



And, if I apply this constant this is the solution I am going to get you can actually model this in any available I will be solver and you can actually check it and you can see here every time every clock there are two multiplier. So, there are two multipliers, there are two multiplier and also the number of ALU operations each time is at max 2.

So, here one, here one, here one, here two. So, that means, I am able to get a solution where I am actually my lambda equal to 4 for the given resource bound multiplier equal to 2 and adder equal to 2. So, that is what this is talked about. So, what I can suggest you that you take some delay maybe you can consider the multiplier is a 2-cycle operation or 3-cycle operation

and actually do a homework for this whole ILP for this ILP formulation of the MLRC solution where your delay of the multiplier is say 2.

(Refer Slide Time: 28:43)

ILP formulation of MRLC Scheduling

- Minimize resource usage under latency constraint
- Dual of the MLRC problem.
- Resource usage is unknown here.
- The optimization goal is a weighted sum of the resource usage represented by vector \mathbf{a} .
- Hence the objective function can be expressed as scalar product between a vector whose entries are the individual resource (area) costs and the vector \mathbf{a} .
 - Minimize $\mathbf{C}^T \mathbf{a}$.

\mathbf{C} in $\mathbb{R}^{n_{res}}$ is a vector which entry is individual resource cost.

$\mathbf{C} = [c_1 \dots c_n]$

$c_1 a_1 + c_2 a_2 + \dots + c_k a_k$

$5 \quad 2 \quad 3$

$\mathbf{a} = [a_1 \dots a_n]$

$\mathbf{A} = \mathbf{A}$ (known)

$\mathbf{a}_k + \mathbf{a}_{i,j}$

\mathbf{b}_{ind}

20

Now, I will quickly move on to the MRLC which is basically latency bound is given, I want to minimize the resource. So, this is a dual problem of the MLRC problem. So, here the resource usage is not known. So, my \mathbf{a}_K is unknown. So, that is something is unknown in my equations.

So, my objective is to find the value of the \mathbf{a}_K and my lambda is given lambda is not something you have to assume anything here it is given that force say and I want to find the value of the \mathbf{a}_K and also $x_{i,j}$ because $x_{i,j}$ is also not known. So, in this case I want to find the value of this two variables and my lambda is known. So, this is something.

In the previous case \mathbf{a}_K was known and lambda is also kind of known but, it may not work for any value. So, I will guess a value and then I solve this problem, but in general this lambda is something I am going to achieve, but in this case \mathbf{a}_K the resource boundary is also not available. So, I want to so what will be my objective function here objective function is not that minimization or tn. So, here it should be the minimization of the resource bound.

So, if you assume that say there are K types of resource so; that means, a_1, a_2, \dots, a_K say \mathbf{a}_K . So, the objective function is given by $\mathbf{C}^T \mathbf{a}$. So, if you just write this expression is

basically c_1 into a_1 , c_2 into a_2 and c_K into a_K . This is my objective function. Now, what is this value of c here, so, that you should understand.

So, this we can assume if we assume that each particular the area needed for each type of operator is same you can assume all c equal to 1 all 1 if the area requirement for each type of resource is same. But, in general that is not the true for say multiplier we need more area compared to adder or subtractor. So, in that sense this may not be the all 1.

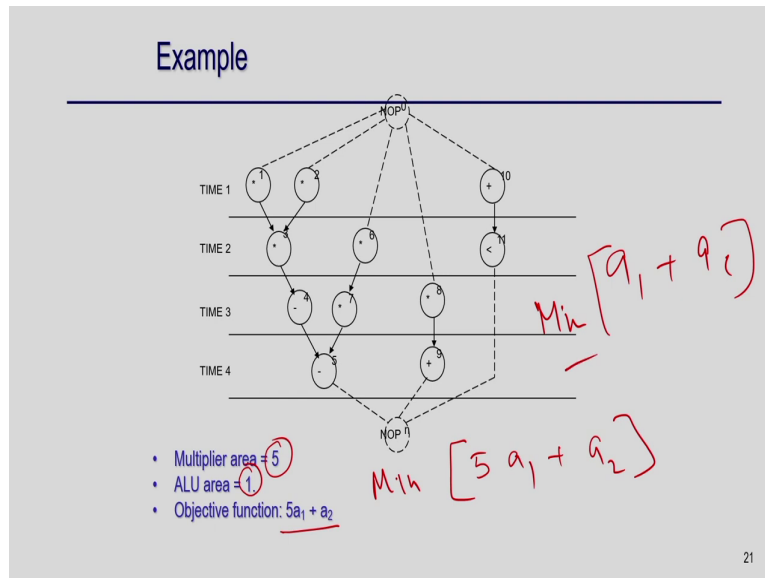
So, what we should ideally consider for this? We should actually consider the kind of some relative value for this say this area for each type of resource. Say you can assume that say this is say 5, this is 2, this is 3 and so on. So, this C is kind of a constant vector say 5, 3 something like this. So, where this number represent the area requirement of that particular module in hardware.

And, if you just try to reduce this number you can assume all 1. Here it will actually reduce the total overhead in terms of area because this is a resource constant. So, we you might have if you can assume that if the number it is not only that if you put two multiplier it is not same as putting two adder into the circuit two multiplier if you assume that multiplier area is 5 times of the area.

So, you can assume that my c_1 equal to 5 and c_2 equal to 1. So, that means, placing one multiplier in the hardware which you needed 5 time more area than an adder. So, this is how you can actually define your cost, but if you are only rely or your intention is to just to reduce the number of resource you need then you just consider all 1.

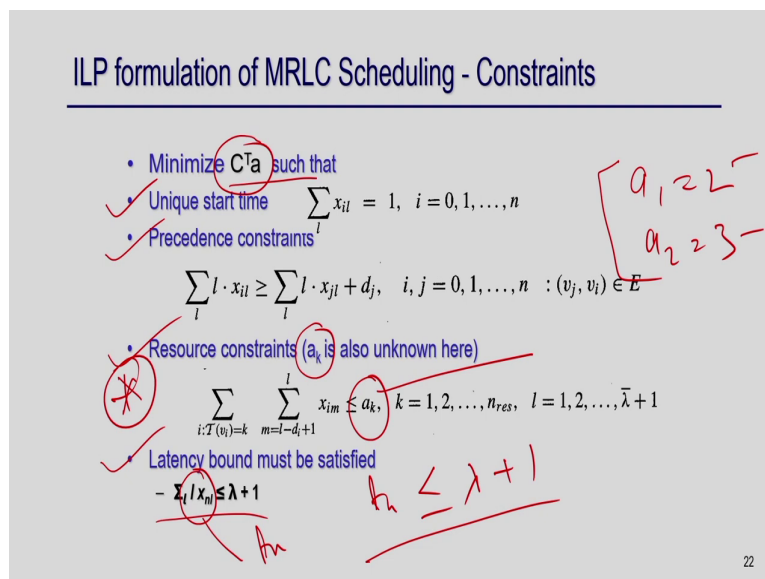
So, in that case there is no area impact will be considered. So, it is up to your objective.

(Refer Slide Time: 32:06)



So, one example given for here that, if your multiplier area is 1 and area ALU area 1, your objective function is basically $5a_1$ plus a_2 ? You try to minimize this value and if you assume that I want to just reduce the number of kinds you just put say I want to reduce a_1 plus a_2 total count. So, this will be my minimization function. So, that is something I think you understood this concept.

(Refer Slide Time: 32:38)



Now, let us try to formulate this. So, I have a minimization function is CT where a is the resource bound. I try to find out and CT is a constant based on either all 1 or the relative area requirement and then let us see what are the constant is applicable here. So, obviously, unique start time must hold because these are all obvious? Then precedence constraint must fall because all data dependency must be satisfied.

And, resource constraint also you should specify I am going to talk about this and you have a latency bound. This is the addition constraint which was not there in the MLRC problem because here there that λ was not kind of given, but it given, but here you have to pass specify that this is basically the start time of the sink node.

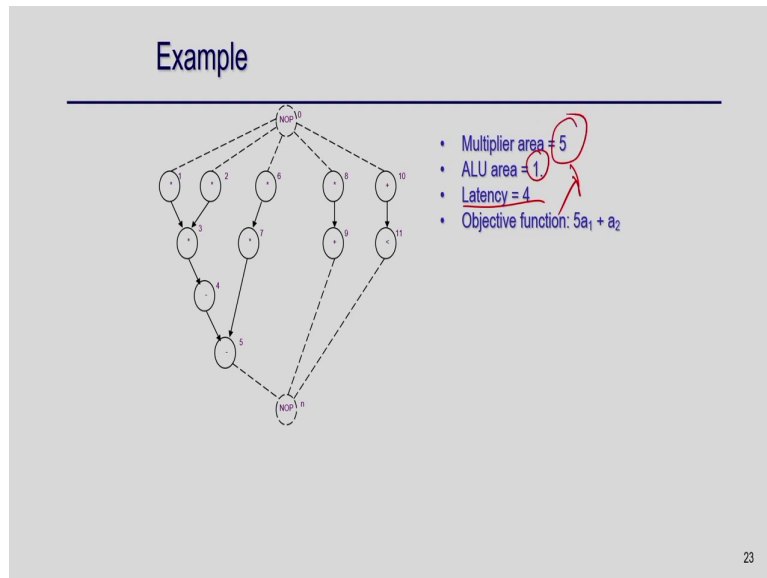
This is my t_n . So, my is basically saying that t_n must be less than equal to $\lambda + 1$; that means, your whole behavior must be scheduled within $\lambda + 1$ times term. And, then now you might ask me why we need this resource constant because this a_k is not known mean to me the answer to your question is we need this because say suppose you determine the value of say a_1 equal to 2, a_2 equal to 3 and say this is what you identify by solving this problem.

But, once you have this you must mention that the number of operation that going to schedule in each time step must satisfy this or specifically in every time step say number of multiplier operation running must be less than 2 and number of say ALU operation running in the 3 must be 3.

So, this unless you specify this particular constraint this must not hold and your ILP must give you some solution where the number of operation running in time step may be more than this.

So, just to avoid this you add this constraint, but here a_k is a unknown variable, but this is already specified in the optimization function. So, effectively this will be taken care, but this actually ensures that once you identify some resource value that must also satisfy the resource constraint each timestamp. So, this is the overall modeling of the MRLC. I hope you understand.

(Refer Slide Time: 34:47)



And, we can quickly take this example. Same, I have given this lambda equal to 4 and then I assume that multiplier is 5 and area 1 is 5. So, which I have already talked about this overall objective function is $5a_1$ plus a_2 . And, then we already so, the for this start time we have to specify the same constraint like this because this is the same thing.

For dependency we have to specify this constant again because this is the data dependency or precedence constraint and for resource constant I am going to specify this, but only thing is that here this resource bound is a_1, a_1, a_1 . So, this is what I have done here.

(Refer Slide Time: 35:27)

Example

The objective function to minimize is $c^T a = 5 \cdot a_1 + 1 \cdot a_2$.

$$\begin{aligned} x_{1,1} + x_{2,1} + x_{6,1} + x_{8,1} - a_1 &\leq 0 \\ x_{3,2} + x_{6,2} + x_{7,2} + x_{8,2} - a_1 &\leq 0 \\ x_{7,3} + x_{8,3} - a_1 &\leq 0 \\ x_{10,1} - a_2 &\leq 0 \\ x_{9,2} + x_{10,2} + x_{11,2} - a_2 &\leq 0 \\ x_{4,3} + x_{9,3} + x_{10,3} + x_{11,3} - a_2 &\leq 0 \\ x_{5,4} + x_{9,4} + x_{11,4} - a_2 &\leq 0 \end{aligned}$$

a_1, a_2

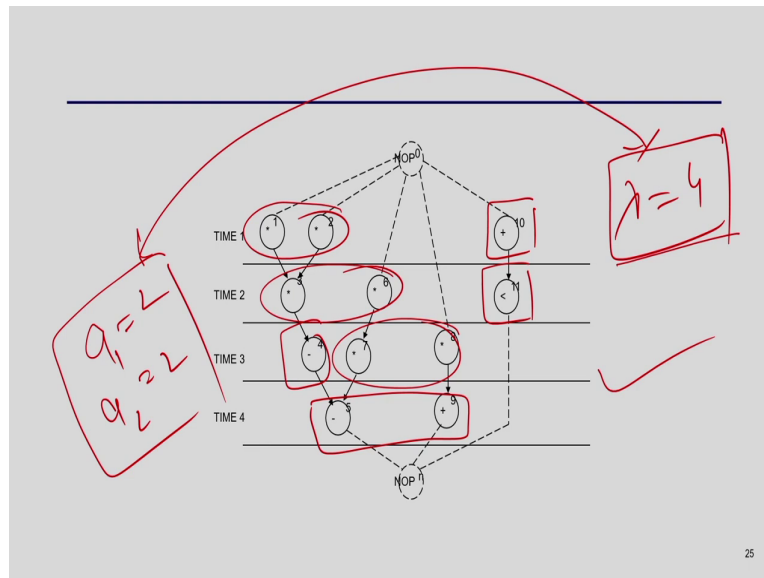
x_{ij}

24

You can see here that the same constraint I have written, but I have just giving this a 1. So, I am giving this is the multiplier requirement and this is. So, the same equation that I have taken from there I just replace them by earlier it was 2, I just put a 1 and here this is the ALU bound for a2. So, instead of 2 I am just putting a2 here.

So, once I give this objective function and this unique start time, precedence constraint and this resource constraint this ILP solves this and it will give the value of this a1 and a2. It will give the value of a1, a2 and also is give the value of x_{ij} which is the start time of each variable because this is also unknown in this formulation.

(Refer Slide Time: 36:13)



And, if you actually solve this will get the same solution whatever you got in MLRC because this is a lambda is the minimum one.

So, for this is the best possible solution. But, it is basically saying that you can actually achieve for this lambda you are getting this and you can actually identify the value of this a 1 is here 2; here 2 multiplier here 2 multiplier. So, a1 is basically 2 and I have one ALU operation here, one ALU here, one ALU here and two ALU here. So, a1, a2 is also 2.

So, you can actually see that so, it is basically dual problem. So, what I say that if I give 2 multiplier and 2 ALU my MLRC says that you can actually achieve lambda 1 and here I am saying that I am if I give lambda 4 I can achieve I mean I can execute the whole thing using two multiplier and two ALU.

Because this is the dual problem your solution is since lambda is equal to 4 is the minimal one so, this is possible. But, if you take lambda equal to 5, 6 or if you take a different resource bound your solution you can end up a different solution by these two formulation.

(Refer Slide Time: 37:19)

ILP Solution

- The ILP formulation of the constrained scheduling problems is attractive for three major reasons.
- First it provides an exact solution to the scheduling problems.
- Second, general-purpose software packages can be used to solve the ILP.
- Last, additional constraints and problem extensions (e.g. scheduling pipelined circuits) can be easily incorporated.
- The disadvantage of the ILP formulation is the computational complexity of the problem.
- The number of variables, the number of inequalities and their tightness affect the ability of computer programs to find a solution.

26

So, with this I conclude, but before concluding I just say what is the advantage of disadvantage of ILP formulations is basically we have already discussed this scheduling problem is a complex problem is NP complete problem. So, we usually the heuristic we do not have any FC and polynomial algorithms we usually apply this heuristic way solution, but this ILP actually solve this problem and give a exact solution.

So, that is the biggest advantage that in the ILP you it actually gives you the exact solution and also there is another advantage here is that you can just add different set of constraint here.

So, you have already three type of a constraint. If you want to say some put some constraint on the reliability, you want to put some constraint on the power or say you want to put some constraint on say pipelining. So, you can actually easily add some constraint and you the same ILP actually give you a solution.

So, adding different kind of constraint is actually easy in this ILP formulation. And, also the things is that see I have not discussed about how to solve this equations. Because this is already tool available and if you just model this and give it to a tool it will give you this. So, you can actually use the general purpose solvers for the ILP. So, this is the advantage of this ILP formulations.

But, this actually works for when your number of operations are actually small for bigger you can understand that when there are many operations this formulation become complex because this try to just find the exact solution. So, for small to medium size of stocks graph or say sequence graph you can actually apply things, but for a bigger task case this ILP may not actually solve in feasible time. It may take lot of problem.

So, but this is something very important study how to model the things in ILP what we have learned in this today's class. So, with this I conclude today's class.

Thank you.