

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

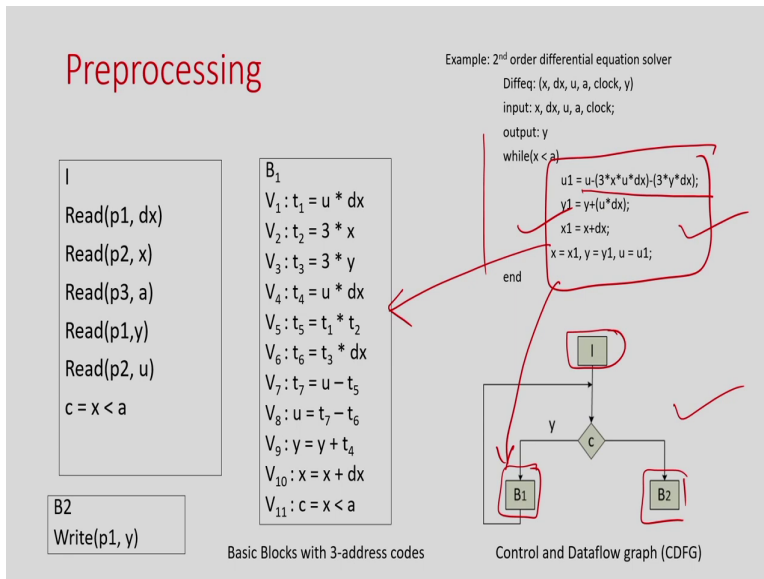
Module - 02
C-Based VLSI Design: Scheduling
Lecture - 06
ILP Formulation of Scheduling

So, welcome student. In today's class, we are going to discuss the ILP Formulation of the Scheduling Problem. So, in the previous class, we have seen that this scheduling is an step where we are actually decide the time step of each operations; that means, if there are set of operations we decide in which time step which operation is going to be executed.

And also we have seen in the previous two classes, that I mean once you are given a program which is consist of a loops, and if else, I mean branch everything what we do we identify the CDFG, control and data flow graph of that. And then we take the each basic block at a time. So, one of the block at a time and try to schedule the operations. This is how I am, we have discussed so far.

And then, for that particular basic block we can schedule the operations and then we can combine the schedule of all basic blocks and then we can actually have a schedule of the complete operations.

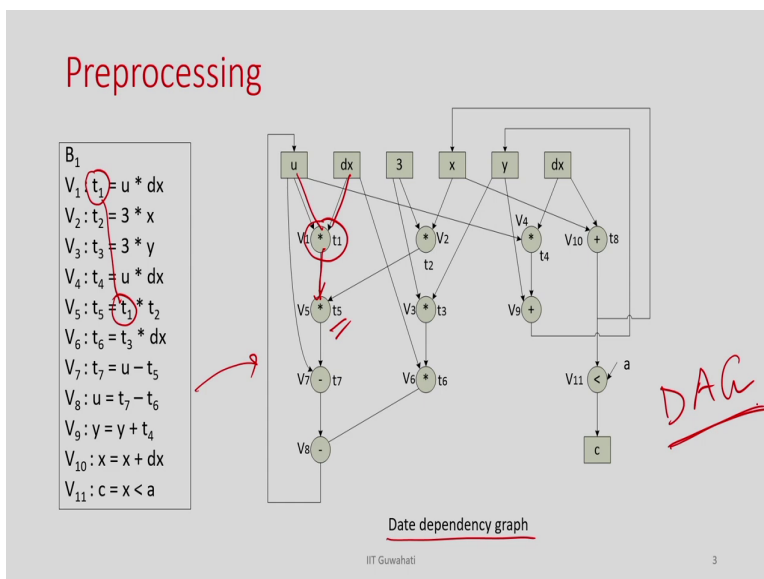
(Refer Slide Time: 01:43)



So, for example, this is the example I have taken in the last class that if you have a while loop like this, so we have extract the CDFG, where this is a basic block, this is the basic block within the while loop, and this is the basic block at the end, and this basic block represent this code.

And in throughout this class we try to take this particular behavior and we will discuss all these algorithms for scheduling and then allocation on binding with respect to this example.

(Refer Slide Time: 02:13)



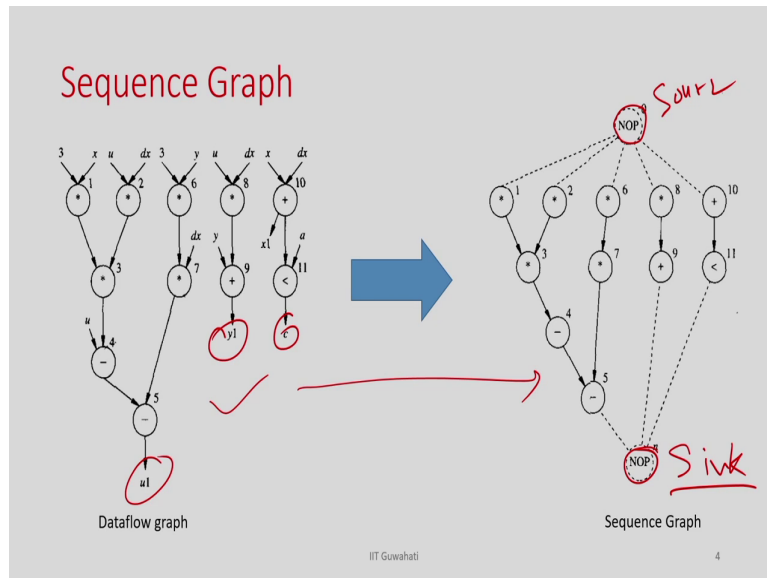
So, and then this is what is the plan. So, we have seen that for now if you consider this particular basic block that whatever the operations are here, they are actually written in the three address form.

So, that means, we split the operation into smaller unit or level operations because otherwise the delay to execute a single operations will be more. So, that is why that is not feasible, so you have to do that. And then what we did is that from this we identify the data dependency. So, this is what is called DAG.

It is basically directed acyclic graph, where we actually represent the data dependency. And what is the data dependency? If some variable is getting defined somewhere and it is used in some different other variable there is a dependency. For example, here t_1 is defined in V_1 and that is getting used in defining t_5 . So, there is a dependency. So, this is my t_1 operation which is doing this $u + dx$ and that particular things is going to t_5 , so where this t_5 is getting calculated.

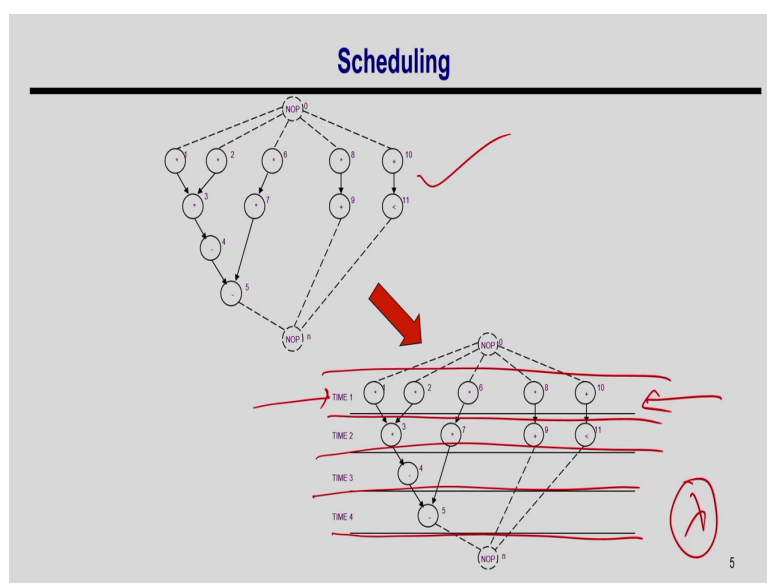
So, this is what the data dependency graph, which we extract from the operations of a basic block. And then what we have seen that and this graph is something is a DAG which is directed acyclic graph there is no cycle, because it is a for flow of data a kind of top to bottom. Because operations are sequential in c code, so there is no kind of cyclic dependency within a basic block because there is no loop nothing. So, this is something I have given.

(Refer Slide Time: 03:44)



So, we have this data dependency graph and then what we convert this into a sequence graph where we just say instead of keeping all these variable names and all because they are not so important during scheduling, what we do we just merge all this input to a single node which is called source and all the output all the output into a single node which is called sink. So, this is the model of a basic block. So, it representing the information of the basic block, the data dependency among the basic block,

(Refer Slide Time: 04:19)



So, now what is scheduling? Given such sequence graph it determines in which time step which operation to be executed. So, here is an example. So, basically you decide that this operation is going to execute in time step 1, this operation is going to execute in time step 2, this operation is going to be execute in time step 3 and this is in time step 4 and so on. So, this is the job of the scheduling.

(Refer Slide Time: 04:42)

Scheduling Problem Formulation

Input:

- Sequence Graph $G = (V, E)$, $|V| = n$
- Delay of each node. $D = \{d_i, i = 0, 1, \dots, n\}$
- Resource or Timing Constraints (optional)

Output:

- The **start time of each node** $T = \{t_i, i = 0, 1, 2, \dots, n\}$
- **Latency**: number of cycles to execute the entire schedule. Difference of start time of source node and sink node; latency = $t_n - t_0$

The start time of an operation is at least as large as the start time of each of its **direct predecessor** plus its execution delay

$$t_i \geq t_j + d_j \quad \forall i, j : (v_j, v_i) \in E$$

So, if I just recap what we have defined the formally the scheduling problem that you have given the sequence graph; that means, the operations and the dependencies within a basic block. And also we have decide discuss that this each operation is not always single cycle.

The operation might take multiple clocks to execute in hardware. So, there is a delay associated with each node. If there are n nodes each node had a delay d_i , node V_i has a node delay d_i ; that means, this particular operation will take d cycle of clock in hardware execution.

And optionally we usually given the resource or timing constant that it is not something in hardware it is unlimited resource, so you might have a limited number of resource, you try to schedule the operations using the resource available to you.

Or sometime you might given this timing constraint or the latency constant that this is the time step is given to you, schedule the operation using minimum number of resource. And

then given this input your scheduling does is you identify the start time of each operation t_i . So, v_i starts at time step t_i . So, when the operation starts.

And also inherently it gives the total latency that is the total number of cycle needed to execute the entire behavior. So, that is something is a scheduling problem. And you have to make sure that whenever you assign time step the data dependency is not violated. So, the data dependence in the sense that if there is a node V_i to V_j and there is an data dependency this must be scheduled and complete its execution, then only I can start the operation.

If this is a 2 cycle operations, if it is time step say starting at 3, so only I can start this operation at time step 5 or more. I cannot start this anything beyond 5. That is what the dependencies.

(Refer Slide Time: 06:42)

Scheduling Problems

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

IIT Guwahati 7

So, this is what we have already discussed in the previous classes. And then based on this constraint or the given you can have a three kind of scheduling problem. One is, first is the unconstrained where there is no resource or timing constant, you just schedule and then you try to identify the minimum number of time step, because since there is no resource constraint your objective is try to schedule the all operations as using minimum number time step.

But the two practical algorithms that are of our interest is the MLRC and MRLC. So, minimum latency or minimize latency under resource constraint or minimize resource under latency constraint.

So, here the resource constraint is given. So, I have given this much resource is available, you schedule the operations, but you try to minimize the number of time step needed to execute the behavior. And this is the dual problem. I have given a latency constraint. I have said that you schedule in say 5 time step, but you use the minimum, you try to minimize the resource usage.

These are the two dual problems. It is just opposite problem the others. And these are the two problems which is actually practically important because most of the time we either we are bounded by the resource constraint or timing constant. So, we have to come up with some schedule, which actually satisfy this given constant. So, and also we have seen that this scheduling problem for general DAG is a NP complete problem.

(Refer Slide Time: 08:09)

The slide is titled "Scheduling under resource/timing constraints". It contains a bulleted list of algorithms categorized into "Exact" and "Approximate/Heuristic based". Handwritten red annotations include underlines, a bracket, and complexity notations.

- NP Complete Problem
- Algorithms:
 - Exact:
 - Integer linear program
 - Hu (restrictive assumptions)
 - Approximate/Heuristic based:
 - List scheduling
 - Force-directed scheduling

Handwritten red annotations on the slide include underlines under "NP Complete Problem", "Algorithms:", "Integer linear program", and "Approximate/Heuristic based:". A red bracket groups "Integer linear program" and "Hu (restrictive assumptions)". To the right, there are handwritten red notations: $O(n)$, $O(n^2)$, $O(n^3)$, and $O(n \log n)$.

So, what does it mean the NP complete? So, NP complete is the problem for which you cannot have an algorithm which can polynomial time, it is a deterministic polynomial time algorithm does not exist which can give us optimal solution. So, that means, we need an exponential algorithm to get the best solution. Say for example, I have given a DAG or the

dependency graph, and I say I you have this say this much of resource; you identify the minimum number of time step needed for this using this resource.

So, for generic DAG this problem is NP complete. I mean what is the intuition? I am not going to prove why this is NP complete. But the intuition here is since there is a dependency of the operations is given, so you cannot just statically determine that if I schedule this operations here that guaranteed the rest of the operation will be scheduled using the given time step.

Or if it does not guarantee that in if you just try to schedule this first time step, you cannot determine which are the operations to be scheduled here, so that I can actually schedule the other operation using the minimum number of time step.

So, basically in the other sense since the operation can be scheduled in any size stamp there are many possibilities of scheduling ideally the exponential algorithm means what is it actually check all possible such mapping or scheduling and give you the best one. So, without exploring all such possibilities having the best solution, identifying the best solution is not possible for NP complete problem.

So, let us not go into the detail of such proofs and all, but you the basic bottom line of this scheduling problem is computationally hard problem. In the sense, you cannot have a polynomial algorithm like if the number of nodes in the graph is n , say order of n or order of N^2 or order of N^3 .

So, this kind of algorithm does not exist or cannot you cannot have an algorithm of this which can give you the best solution. So, to get the best solution you have to go for exponential algorithm. So, that is why this is a hard problem.

And there are two approaches to solve this problem. The first one is the obviously, if you want to get the exact of the optimal solution, you have to go by ILP formulation. We will talk about that in today's class that integer linear programming model which can give you the optimal solution, guaranteed to give you the optimum solution. But in most of the cases in practice because this is a compression hard problem we usually apply the Heuristic algorithm.

So, we actually have some kind of with some efficient algorithm like order of n or say $n \log n$ this kind of algorithm which give you the good solution it may not be the best always, but the solution is good enough for the most of the scenarios. And since this timing and the running the EDA tools are basically of our prime importance, most of the time we usually go for this heuristic based solution to solve the scheduling problem.

So, we are going to discuss about all such problems in this particular course. In today, we try to see how we can model this scheduling problem using ILP, integer linear programming.

(Refer Slide Time: 11:29)

Integer Linear Programming (ILP)

- Given:
 - integer-valued matrix $A_{m \times n}$
 - variables: $x = (x_1, x_2, \dots, x_n)^T$
 - 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:
 - $Ax \leq b$
 - $x = (x_1, x_2, \dots, x_n)$ is an integer-valued vector
- 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

Handwritten notes on the slide include:

- Objective function: $\Rightarrow \text{Min } (x_1 + 2x_2)$
- Constraint matrix: $A = \begin{bmatrix} 1 & -1 \\ 3 & 5 \end{bmatrix}$
- Constraint vector: $b = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$
- Variable vector: $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
- Matrix dimensions: $m \times n$ for A , n for x , m for b .
- Constraint notation: $Ax \leq b$ with $c \in [1, 2]$ and $x_1 - 1.7x_2 \geq 0$.

So, just to go into that differ formulating the problem let us try to understand what is this integer linear programming first of all. So, first let us try to understand from the higher level that. So, what is the objective here, so in this ILP problem? So, if you have a optimization problem either say minimize something or maximize something, you try to solve that. So, what is the thing? So, you have some minimization objective.

So, you have a minimization objective say suppose you have a minimization objective say $x_1 + 2x_2$ should be minimum. So, you have two unknown variable x_1 and x_2 , and you want to find out the value of this x_1 and x_2 such that the expression $x_1 + 2x_2$ is minimum. So, this is say your optimization objective. So, that means, whenever you try to solve try to model this problem as an ILP you have set of unknown variables.

Suppose you have, in general you might have say n unknown variables, say x_1 to x_n . So, this is something unknown variable and we want to identify the value of this variable. And with an objective to minimize or maximize certain expression or some optimization function.

So, it is something the end of it? No. Basically, now these variables can be integer. So, then since integer linear programming we will assume this variable which actually integer.

So, integer can have a range, but this x and x_2 may not take all possible values. So, there will be some constant on these variables. So, that constant is given by a matrix which is called m into n . So, there is m such constant. For example, say suppose you have two variables x_1 , x_2 and you are saying $x_1 - x_2$ should be greater than equal to 0 and say you are saying $3x_1 + 5x_2$ should be greater than equal to 5 and so on, say I am just arbitrating giving things.

So, there are two constant. So, how this matrix will be represented? So, this is one into this, one into this, so it is 1 and minus 1, 3 and 5. So, this matrix A is this. So, which is something we can always represent this and matrix by the coefficient of those expressions. And so, this is basically how many is column will be there? It is basically the number of variables, so this is n . And if there is m constant it is m into n matrix.

And we also have this constant, so the right hand side which is basically my b . So, it is basically represented as like $A * x \leq b$.

So, basically it is nothing, but we have set up expressions and those expression determine certain constant on the variables and those constant expression you can in high level you can assume this way, and we can represent this expression by a this kind of matrix operations because if I just multiply this constant with this matrix with x , x is basically $1 * n$.

So, if I just assume that this is basically $n * 1$, this is the say one the variable is x_1 to x_n . So, it is $n * 1$. So, we will get this $m * 1$ expressions. So, that something is there. So, the bottom line is that there are set of unknown variables, I have a optimization objective and I have given set of constant. And this constant you have to understand what is the linear means is basically this expression should be linear. So, there is no non-linear term.

So, non-linear means what? You do not have a term like x_1 square or x_1 into x_2 into x_3 . See, here it is of every variable is multiplied by a constant. There is no multiplication of two

variables. So, then that will be non-linear expressions. So, this constant must be linear. So, that is the, that is why it is a linear integer linear programming and this unknown variables are integer.

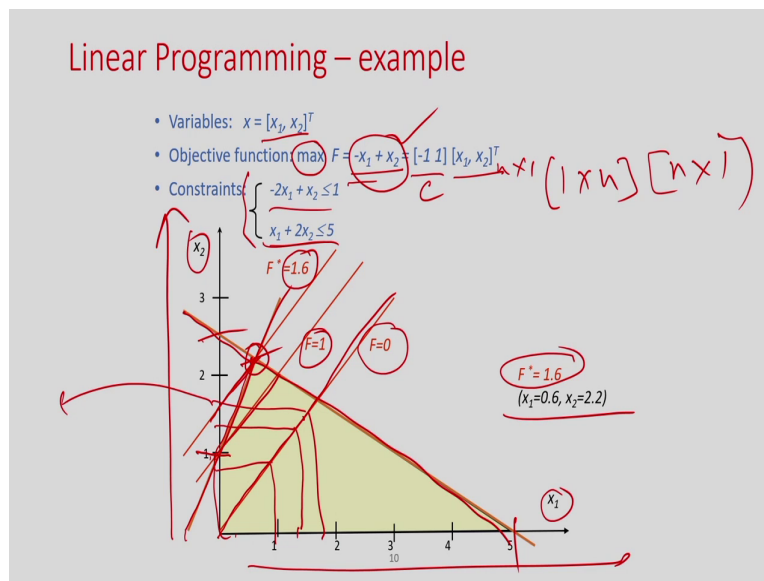
And then what this ILP does? Given such set of constraint and a objective function, it try to give you, it will give you the value of this variable x_1 , x_2 and all those unknown variable x_n , such that the value of this optimization objective is minimum. Whatever the optimum it is it satisfies the minimum optimization objective. So, that is what is the ILP solved. So, again we do not want to go into detail how this ILP works and all because that is something is not the objective of this course.

But, what we try to do in this course is something how can I model this scheduling as this problem because you understand the scheduling is also optimization problem, either you try to optimize this resource under latency constraint or you try to optimize the latency under resource constraints.

So, there is an optimization objective and you try to schedule the operation. So, that means, your operations are the, that schedule time is the variable for us. I do not know which time step, which operation to be executed in which time step.

So, that means, that our time step where a variable going to be executed is kind of a unknown things to us and that is the variable to us and I try to schedule the operation, try to find out the value of those variables, such that this my optimization objective either it is a latency or resource is satisfied. That is what is our modeling of this scheduling problem as ILP. We will go into detail in this today's class.

(Refer Slide Time: 17:13)



So, just to give an example for this ILP, so this is linear programming. So, what is the difference between ILP and linear programming? If the variable is linear; that means, it is a continuous or you can think of a floating value, it can take any value. Then, it is linear programming. And if these variables are actually take only integer value or discrete value 1, 2, 3, 4 and so on then it is an integer linear program.

So, it is a nice example. Say suppose, my two variables are there x_1 and x_2 and then my optimization function is this. So, this x minus x_1 plus x_2 . That I want to do a maximize, it can be minimize or maximize. So, here you see the minimization objective I have written as a $c^T x$. So, that means, this c is a constant. The constant is something; so, for this example the example I have taken it is the 1 and 2. This is my c .

So, c^T means c transpose, so this is my the variable. So, here I can say that this is my c and this is my variable. So, this is basically you can understand that 1 into n , this is the matrix this one and this one is basically n into 1. So, if I multiply I will get this expression, nothing else. So, this is $n * 1$. So, that will give you my expressions.

And say suppose these are the constraint given that this should satisfy. So, we can see here that if I apply this constraint, so basically it restricts the because its x_1 and x_2 can take any values, this x_1 this constraint actually restrict that it cannot take other values. For example,

here if you say this is my x_1 and this is my x_2 , so x_1 can take any value in this range and x_2 can take any value in this range.

But this expression says that your value cannot go beyond this x_1 cannot go beyond 5 and x_2 cannot go beyond some 2.5 or something.

And this expression says that your value of x_1 , x_2 cannot go beyond this, and this basically restricts the other values because it can go this way also. So, this is my possible value set. This is basically the solution space. So, this is my solution space, this yellow part is the solution space for this. And what is my? There are many solutions here; I want to identify the value of x_1 such that this particular function goes the maximum value.

So, now if you try to solve this, so if you just put these values. So, if your x equal to 1 and y equal to sorry x_1 , x_2 is this or if your x_1 is this value and x_2 is this value which is say 1.2 or 1.3 and so on, in all cases you will get your F equal to 0. So, for this all these value in this line will give you F equal to 0. Similarly, if you take the value in this range your F equal to 1. But the maximum value will get, so for these values of x_1 and x_2 your maximum value is F value is 1.6 and which is the maximum value.

And we can see here even for that there are many values possible for x_1 and x_2 and this is the value which you give you the; so, which is basically for this x_1 and x_2 you are getting for this particular value. So, your x_1 equal to 0.06 and x_2 equal to 2.2. So, the idea here is this is just an example that you try to find a value of this x_1 is x_2 , such that you get the value of this, your this objective function maximum.

(Refer Slide Time: 20:57)

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?

$n \times \lambda$ $n \times (\lambda + 1)$

11

So, now move on to our problem. So, our problem is scheduling. In our scheduling, what I want to do is we want to schedule the operations that mean I want to assign the time step. So, for my problem what is my variable? So, that first you have to identify. So, what is not known to us? The operations why did get time scheduled? So, that is not known to me. So, that should be the variable. So, what we did? We for that we take a variable which is called xil. So, what is that?

So, xil is basically 1 if the person v1 starts at time step l. So that means, it is going to execute, it is assigned time steps l. And since it is a multiply it may be the delay d, so it can start execution from l and it can go l plus 1, l plus 2 to l plus d minus 1 because it is delay, it has some delay. But it starts its execution from time step l. So, that is something is not known to me.

So, I am going to get a variable which is basically xil which is something one if this operation vi start its execution from time stamp l, otherwise this value is 0. So, this is the variable because in an ILP formulation I want to have some unknown variable, I want to find the values. So, if I identify the value of this xil for all possible xil then I know what is the start time of each operation. So, let us try to understand how many such variables we need.

So, suppose you have given a latency bound is $\lambda + 1$, so limit is sorry λ and so, since it is a λ , that means, this is sink node will be scheduled and $\lambda + 1$, so we assume that the possible time step is basically $\lambda + 1$.

And there are say n number of operations. So, how many such variables are possible in worst case? Is basically $n * \lambda + 1$ or say $n * \lambda$, $n * \lambda + 1$, because any operations can schedule in any time step ideally. So, I need these many variables.

(Refer Slide Time: 23:19)

ILP Model of Scheduling

- Binary decision variables x_{ij}

$x_{ij} = 1$ if operation v_i starts in step j ,
 otherwise $x_{ij} = 0$
 $i = 0, 1, \dots, n$ (operations)
 $j = 1, 2, \dots, \lambda + 1$ (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?

$\frac{v_1}{\lambda=4}$ $\frac{n_{11}}{=1}$ $\frac{n_{12}}{=1}$ $\frac{n_{13}}{=1}$ $\frac{n_{14}}{=1}$ $n * (\lambda + 1)$

11

So, for example, say for operation v_1 how many variables are there? Say suppose, my λ equal to 4, so how many variables are possible? So, x_{11} , x_{12} , x_{13} and x_{14} . So, what does it mean? So, if the operations v_1 is schedule in time step 1 this then x_{11} is true, if the operation 1 is schedule in time step 2 then this is 1, other variable will be 0. If the operation 1 or v_1 schedule in time step 3, then this will be 1 and rest of the variable will be 0.

Similarly, if this variable operation v_1 schedule in time step 4, then this will be 1 and rest of the variable will be 0. So, that is why because I do not know this where this operation v_1 is going to schedule. So, it can schedule in any time step given. So, I will assume these many variables are there and total number of variable will be this. So, this is obviously, the upper limit of variables. And if you understand in my previous modeling the expressions; so, constant it depends on the number of variable.

So, if the number of variable is more this expression or the constant expression will be more and so, I mean obviously, if you understand if the number of unknown variable is more your solution or solving the problem is difficult.

So, the question here is that obviously, in this model that I can have n into λ plus 1 number of unknown variables can you reduce it that is a question 2. And the concept here is that we can actually utilize the ASAP and ALAP solution that we have already discussed in the previous class to minimize this number of variable.

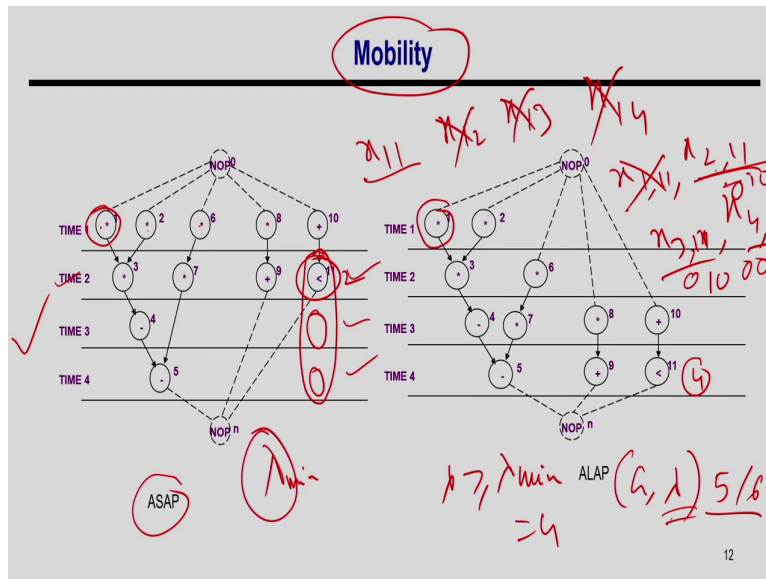
So, let us try to give the intuition. So, for example, here say suppose you have 4 time step is given, so suppose your λ is given 4. So, λ plus 1 is 5. So, if you already know this in this case, since these operations after that there are 3 operations, so this operation never can schedule in time step 4.

Then, I cannot schedule everything in 4 time step because if it is if schedule is time step 4, this will be schedule in 5, this will be schedule in 6, this should be schedule in 7 which is not a valid solution because you want to schedule everything within say time step 4.

So, that means, although logically any operation can schedule in every time step, but because of the dependency, it is not possible to schedule most of the time a variable cannot be scheduled in any time step rather in a span. And that span can be identified by the ASAP and ALAP. Let me try to explain that. Say the ASAP is something as soon as possible. It is unconstrained schedule.

So, it basically whenever the predecessors operations are already scheduled the operations on which it depends is already scheduled, and then what is going to? It is going to schedule the operations. So, if you take that particular graph and try to do an ASAP you will get this solution.

(Refer Slide Time: 26:19)



So, for example, here this 5 operation will be scheduling time step 1 because it depends on the inputs, and I can assume that inputs are available. And I am assuming all the operations are here single cycle, but you can always model this as a multi cycle operation as well. And once these operations are done I am going to schedule the next set of operation in time step 2 and so on. This is already I have discussed in the previous class.

So, this gives you the minimum value of λ . So, this is because it is a unconstrained schedule you never thought about the resource, so the ASAP always give you the minimum number of time step needed to schedule the operations provided there is no resource bound.

So, that something I can always identify the latency bound that you, whatever the scenario we cannot schedule this behavior without less than 4 cycle, that is something my ASAP gives. And it also assign the time, so this is the earliest possible way.

So, because it is an as soon as possible, so it actually give you the earliest possible time phase where this operation can schedule. And the ALAP is the other. So ALAP, what we give? We give the graph and we give a λ , that I you schedule these operations using this sum λ value and there is not necessary that it should be 4 because 4 is given it can give you 5 or 6 anything.

And whatever the λ you are giving say this is my λ minimum, this $\lambda \geq \lambda_{\min}$, whatever I obtained from ASAP.

So, what this ALAP does? It try to schedule the operation as late as possible given the time bound. And if the time bound is actually greater than the minimum I always have a solution. So, ALAP gives the last possible or the latest possible way to schedule the operations. And we have already discussed the time terminology mobility is the time difference between ASAP and ALAP.

So, that ASAP and ALAP, the difference for example, this operation is scheduled here in time step 2 and this operation is scheduled in time step 4, so that means, it can either schedule in time step 4 or 3 or 4. So, that means, the mobility of this operation is actually 2, which is basically from ASAP to s 3 ALAP. But this operation one is scheduled here in time step 1 it is also scheduled in time step 1.

So, that means, this mobility is actually 0. It has to be scheduled in time step 1 if my λ equal to 4. This is the solution where I am given λ equal to 4. So, there is no mobility. So, for this variable I do not needs this say suppose this 1, so I do not? So, x_{11} , x_{12} , x_{13} and x_{14} , these are the Boolean variable possible. The way I have modeled I do not need this because they are already, it cannot schedule there. It only schedule here.

But for this variable 11, so the possible variable is basically $x_{1,11}$, $x_{2,11}$, $x_{3,11}$, and $x_{4,11}$; that means, this operation 11 can schedule in time stamp 1 or 2 or 3 or 4. Here I can see that for this operation cannot scale in time step 1. So, this is not a valid variable. But this 3 is possible. So, I am going to take this. So, the conclusion of this discussion is that although there are n into $\lambda + 1$ number of variable is possible in the worst case.

But in general, if we just identify the ASAP and ALAP timing, I can identify the minimum set of such unknown variables, which is basically the range for each operation I will identify the ASAP value and the ALAP value. And the range from ASAP to ALAP is the variables where this can get scheduled.

And rest of the unknown variable I am going to remove. So, this is something understood, that I have set of unknown variables and which will be determined by the ASAP and ALAP time, and then let us try to understand. So, this is something the variables.

(Refer Slide Time: 30:14)

ILP Model of Scheduling - Constraints

- Start time of each operation v_i is unique:

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

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 :

$$t_i = \sum_l l \cdot x_{il} = 1 \cdot x_{i1} + 2 \cdot x_{i2} + \dots + \lambda \cdot x_{i\lambda} = \lambda_i$$

14

Now, in the ILP modeling, the next phase is the constant. What kind of constant we can have here? So, I am going to discuss this constant without discussing what is my what scheduling problem I am going to solve that will schedule discuss in the pre next class. Here I try to say how will what are the possible constants I will have to model in ILP formulation and how to model them, because here my unknown variable is x_{il} . And I have to model everything in terms of x_{il} .

So, let us try to understand. So, as I already explained in this say for example, in this example that, so this operation 11 can schedule either in time step 2, 3 or 4, but it cannot schedule multiple times. It will always start only one time. So, for example, if this its time starts at time step 2, this must be 0 and this must be 0. If this is 1 if it is execute in time step 3, this is 1, and this is 0 this is 0. And if it is scheduling time step 4, so this is 1, this is 0 and this is 0.

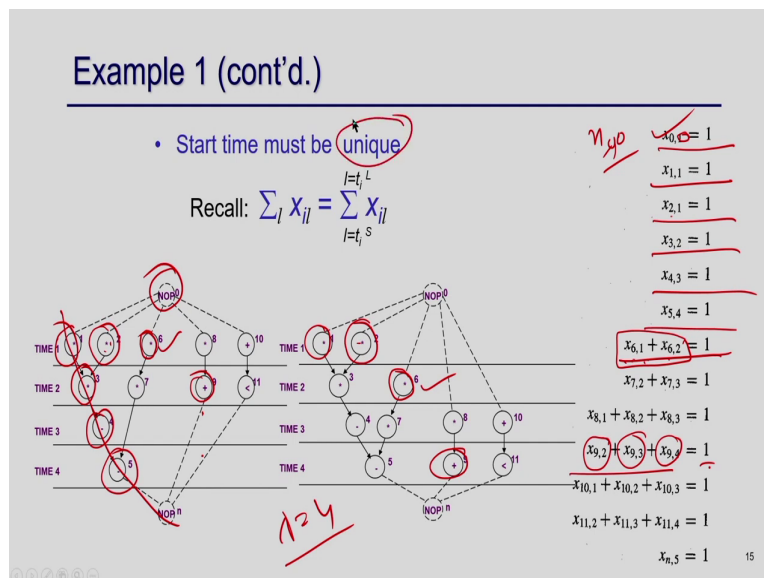
So, it means that only one of the x_{il} will be one for each variable. So, that means, every operation has a unique start time. So, that means, although this I have taken a few variables for each operations, but only one of them will be 1. And that is given by this because, so this x_{il} is the variable, so for each i which is starting from 0 to n , because 0 is the source node and this n is or you can assume that there is $n+1$ which is the sink node which is always schedule at $\lambda+1$.

So, for each variables i the summation of this variable, so for this x_{iy} , so x_{i1} plus x_{i2} plus x_{i3} plus $x_{i\lambda}$ should be equal to 1.

So, that means, only one of the variable value can be 1. And here I as I already discussed that I will not take for every variable for every operations this x_1 to x_λ may not be there. It will be determined by the t_s and t_l . So, this t_i is the time schedule time of the operation i in by ASAP and t_l is the time for the operation i , the ALAP time for operation v_i .

So, this particular constant says that although there are multiple variables for each operations, but only one of them is 1, so which is basically say unique start time.

(Refer Slide Time: 33:14)



So, if you take this example, say for example, if you take this variable this ASAP and ALAP and where my time is 4, so I have already discussed that these operations have only one variable. So, I already know that this operation we are going to schedule in time step 1s.

So, this is $x_{1,1}$, $x_{1,1}$ is basically $x_{1,1}$ equal to 1s and this is basically for the source node. So, source node is always schedule in $x_{0,0}$ which you can ignore actually because I mean this is source node I always know it will be schedule in time step 0.

So, similarly this times operation 2 is only the possibility is one time step because is the ASAP and ALAP time step is same. So, this is also $x_{2,1}$ is also 1, this is operation 2 time

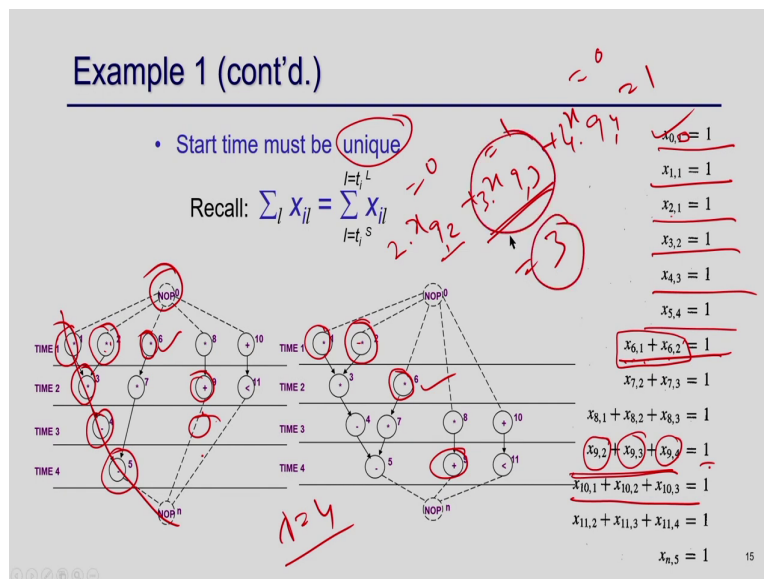
step 1. Operation 3, so these are the critical paths. So, this operation has no options, so these operations will be have a unique value. So, basically I know the start time of this operation, this is already identified for this value if this the λ equal to 4, but for different value they may not be. I can explain with that.

So, this 3, 2 and 4, 3 is 1 and this 1, 2, 3, 4, 5 these are the critical operations. Now, consider this 6, 6 schedule here. And it is scheduled here. So, either it will be schedule in time step 1 or say time step 2 which is given by this. So, either 6, 1 is true or 6, 2 is true. It means that operation 6 can either schedule in time step 1 or operation 6 can schedule in time step 2. But it cannot; it can only schedule one of the time step that is why this plus this equal to 1.

So, now let us take this operation 9, so I will just type it. So, this is schedule in 2, it is in 4, so it can schedule in 2, 3 or 4. So, 9 2, 9 3 and 9 4 summation will be 1; that means, it can schedule one of the time step and this must be unique. So, this is well understood. So, now, let us understand, so suppose it is schedule in time step 1 or say its time t_i because t_i is the start time of the operation v_i that we have already understood.

So, how can I represent this t_i now? Because I cannot introduce another variable t_i . I have to represent everything in terms of x_{ij} because that is the unknown variable for us.

(Refer Slide Time: 35:45)



So, as I mentioned this one of the value is 1. So, for example, say x_1 I am just let me take this 9 1. So, x_2 plus x_3 plus x_4 equal to 1. So, that means, we have already know that this can schedule in this time step.

So, let us say this schedule in time step 3. So, let us say this is schedule in time step 3. So, this is actually 1. So, and this is 0, this is 0. So, what about if I just multiply this with 2; that means, if operation schedule in time step 2, operation 9 times x schedule in time **step 2, I just multiply with this 2.**

So, this expression I am just multiplying with 3, this I multiply with 4, because this is time step 4. This variable deals with time step 4. So, then what will happen say if I just multiply this with this? It will give me the 3. And where it is scheduled? In time step 3.

So, that means, if I multiply this x_{il} with l and take the summation that will give me the time step, where did it schedule. So, as I explained here is that start time is given by this expression that is one of; this is a Boolean variable, one of them is true. And if it is true is first and instead time step l , if I multiply that with l it will give me the l which is the time step. So, then if I just take a summation because the other that other is 0. So, only one of the terms will be 1 and that will give me the time step.

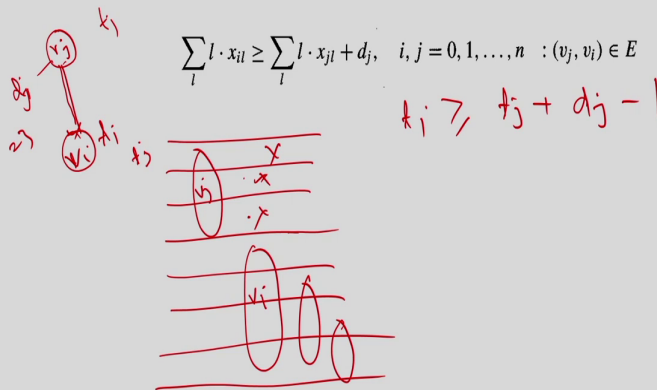
So, this is very interesting that I can represent this t_i as a again summation of l_i into x_l . So, here it is $1 * x_{i1}, 2 * x_{i2} * \lambda * x_{i\lambda}$ it should be. So, this is what is my t_i . So, based on the time step where it is scheduled I can get that l as the time step. So, far we understand that I have a Boolean variable x_{il} for each operations, we have understood the limit where this operation can get scheduled.

And then we identify the first constraint that this the start time of each operation is unique and which is given by this constant. And also we have under, identify how to represent the start time of a variable v_i, t_i in terms of x_{il} . So, let us move on.

(Refer Slide Time: 38:18)

ILP Model of Scheduling - constraints

- Precedence relationships must be satisfied



So, what is the next constant? The next constant is obvious that precedence constraint or the data dependency constraint. So, suppose you have a node v_j and there is a node called v_i and say there is a dependence among them so obviously, so say if this is the time step is t_j and this is t_i and this node has a delay of d_i , d_j . So, what does it mean? So, suppose I have schedule these operations at time step t_j , this is my time step t_j and this is, so say this d_j equal to 3.

When I can start this operation i ? When this operation is finished I cannot start this operation v_i here because then this data is not available. I cannot start my operation v_i here because then this operation is not done, so this data whatever the output of this things are not available. So, I can only start my v_i from here this is v_j . I can start from here. There is no problem, I can start from here, I can start from here, and everything is possible.

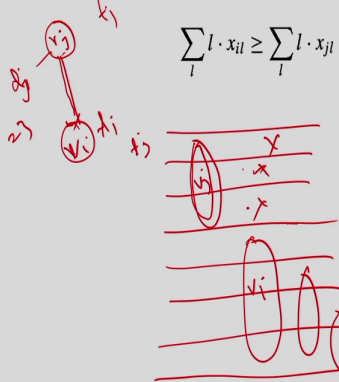
But I cannot start in this time step, I cannot start here, I cannot start here. So, that means, my t_i must be greater than equal to t_j plus d_j minus 1 sorry plus d_j because. So, this should be the greater than equal to this. Because this is if I assume that this is my t_j , so here this is the d_j it will take d_j time number of time step and I can only start after that, so this is obvious.

And now since I cannot represent, I do not have t_j variable in my ILP formulation, I have only x_{il} . So, I can represent this as in terms of x_{il} and which I have already discussed that it is basically l into x_{il} .

(Refer Slide Time: 40:08)

ILP Model of Scheduling - constraints

- Precedence relationships must be satisfied



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

Handwritten notes: $k_i \geq k_j + d_j$

$$\sum_l l \cdot x_{i,l} \geq \sum_l l \cdot x_{j,l} + d_j$$

16

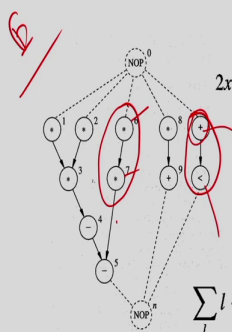
And this is what? This t_j is basically summation of l which is l into x_{jl} and that is and d_j is d_j because a constant value. So, I represent this in terms of this. And here this l varies from is the possible value of λ and this i and j is for all such edges. So, for each such edge I will get a constant like this. And this is called precedence relationship or data dependency constraint.

(Refer Slide Time: 40:49)

Example 1 (cont'd.)

- Precedence constraints

- Note: only non-trivial ones listed



$$2x_{7,2} + 3x_{7,3} - (x_{6,1} + 2x_{6,2}) - 1 \geq 0$$

$$2x_{9,2} + 3x_{9,3} + 4x_{9,4} - x_{8,1} - 2x_{8,2} - 3x_{8,3} - 1 \geq 0$$

$$2x_{11,2} + 3x_{11,3} + 4x_{11,4} - x_{10,1} - 2x_{10,2} - 3x_{10,3} - 1 \geq 0$$

$$4x_{5,4} - 2x_{7,2} - 3x_{7,3} - 1 \geq 0$$

$$5x_{n,5} - 2x_{9,2} - 3x_{9,3} - 4x_{9,4} - 1 \geq 0$$

$$5x_{n,5} - 2x_{11,2} - 3x_{11,3} - 4x_{11,4} - 1 \geq 0$$

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

17

So, if I take this example again and let us say I try to model this say 7 and 6. So, let us say I take this edge and this is here my d_j equal to 1 because these operations are single cycle, but it can be multiple cycles also. So, what is the possible value of this 6?

Because it has mobility 2, even it can schedule in time step 1 or 2, and 7 can schedule either 2 or 3. So, you can say that you can see here that x_{72} and x_{73} . If I multiply with 2, so this is my possible start time 1 for 7. And for 6 it is 6 1 or 6 2. So, this is 1 into x_{61} , 2 into x_{62} .

So, this is the possible value of x term. You can understand if I just put a bracket here, it will be plus. So, this is where this basically t_7 . So, this is the start time of the operation 7, t_7 this is t_6 and it should be and since the operation delay of the operation 6 is 1, so this is my d_7 .

So, this must be greater than equal to 0. So, this is how I can actually take any edge and I can model that way. So, I have to just identify what is the possible value, start time value 1, what is the start time value of this and their difference must be the delay of this node.

And how many constant such constraint will be there? The number of edges. So, there will be e number of such constraint. I am not explaining the others you can actually go through and you can understand, but this is something for each such edges.

(Refer Slide Time: 42:36)

ILP Model of Scheduling - constraints

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

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

$\underbrace{\langle a_1, a_2, \dots, a_k \rangle}_{\substack{1 \quad 2 \quad \dots \quad k}}$

$\gamma(v_i)$

$x_{i,l} =$
 $x_{i,l}$

+, /, *

(k) be (a_k)

18

So, now let us move to the what another type of constraint which is called resource constraint. So, what is the resource constraint? The resource constraint is something you have given some resource bound. So, you can first of all assume that you can have say plus, you

can have division, you can have multiplication, many type of operations, you in the hardware the multiplier may not able to do add addition operation or division operations.

So, you have say k type of operator in say in the behavior. Say k type of operator and the for type k the resource bound is a k . So, that means, if there is say k such operator, so I have the boundaries a 1, a 2 to a k .

So, this is the for operation type 1, this is for operation type 2 and this is operation type k . And given a v_i , node say v_i the type of the operation say determined by the function τ . So, I just if I apply τ this will say this is a plus operations or type operation type 1, operation type 1, operation type 3 and so on.

So, let us say this is what will give me the type of operation because this type of operations all not same. They are having different types, can be plus, minus, multiplications and so on. So, this resource bound supposes I have given, so what is the resource constant? It should say that every type of resource, the number of operation running of that type must be less than of the resource bound. I am repeating again.

So, for a each type of resource because there are k type of resource, so for each type of resource say k , the number of running operation in the time step should be of type k should be less than equal to the a_k . So that means, every time the running operations, the operation that are going to use that this operator type k should be less than of the resource bound. So, let me try to understand this $x \times x$ equation.

(Refer Slide Time: 44:45)

Start Time vs. Execution Time

- For each operation v_i , only one start time
- If $d_i=1$, then the following questions are the same:
 - Does operation v_i **start** at step l ?
 - Is operation v_i **running** at step l ?
- But if $d_i>1$, then the two questions should be formulated as:
 - Does operation v_i **start** at step l ?
 - Does $x_{il} = 1$ hold?
 - Is operation v_i **running** at step l ?
 - Does the following hold?
$$\sum_{m=l-d_i+1}^l x_{im} = 1$$

So, this is the basic in plain English that is the constant, but the first point we have to understand how do I understand what are the running operation in the time step. So, you, so far this x_{il} gives me the start time of the operation i , but it does not talk about when it is running.

Say for example, if my x_{il} is 1, so that means, its time step l , and it is a d , say there are say 3 time 3 cycle operations, so that mean it is running in time step l , it is running in time step l , it is running in time step l plus 1 and it is running in time l plus 2.

(Refer Slide Time: 45:18)

ILP Model of Scheduling - constraints

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

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

$\gamma(v_i)$ $x_{i,k} =$
 $\begin{matrix} k+1 \\ k+2 \end{matrix}$
 v_i

$\{a_1, a_2, \dots, a_k\}$
 \downarrow
 $1 \quad 2 \quad \quad \quad k$

$\frac{k}{k+1}$

$\frac{+}{/} *$

18

So, that is, so although operation start its execution at time step l it is running in 3 all three cycles, and every cycle it needs say this is a multiplier. So, one multiplier will be occupied to execute this operation i or operation v_i because v_i is running in time step l , it running in time step l plus 1, it is running in operation time step l plus 2.

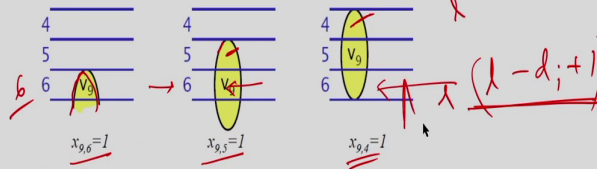
And this in every time step it is going to occupy a multiplier. So, if the type of this operation is multiplier say. So, that means, the start time is one thing that is where it start execution, but identifying the operation that are running in a time step l have to consider the delay of that operation.

(Refer Slide Time: 46:07)

Operation v_i Still Running at Step l ?

- Is v_9 running at step 6?

- Is $x_{9,6} + x_{9,5} + x_{9,4} = 1$?



- Note:

- Only one (if any) of the above three cases can happen
- To meet resource constraints, we have to ask the same question for ALL steps, and ALL operations of that type

So, how to calculate that operation let us try to understand. So, suppose an operation v_9 , it is starting its execution in time step 6. So, that means, it is running in this time step. So, if $x_9 6$ are equal to 1; that means, this operation 9 is starting its execution in time step 6.

So, it is basically running in that time step. If it is started in the time step 5, then also it is running in time step 2. I am assuming this is 3 cycle operation. So, if $x_9 5$ equal to 1, so that means, start its execution here stills it is running here.

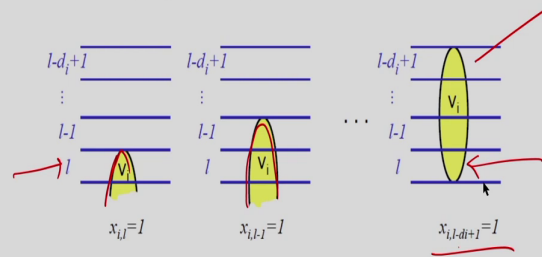
And if say it is $x_9 4$ equal to 1; that means, it start its execution from time step 4, still in time step it is running, time step 6 it is running. So, that means, even if the operation start at say this 6 minus 3 plus 1, so this is my l , this is my d_i and this is 1. So, that means, if my current time step is say l , even if the operation start from x minus d_i plus 1 in that time step still this operation is going to consider is running in this particular time step l .

Because that is my time step where I am actually currently try to identify the number of resource needed, in this time step also this particular operator is going to occupy a resource.

(Refer Slide Time: 47:32)

Operation v_i Still Running at Step l ?

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



So, if you just generalize this, so it is basically you can understand that if the operation is starting in l th time step, l minus 1 time step, l minus 2 time step or l minus d_i plus 1 time step. This is what I have given here l minus d_i plus 1 time step. It is still occupying a resource in time step l . So, if I try to identify the resource requirement for time step l , even if this operation starting if it starting is $x_{i,l-d_i+1}$, d_i plus 1 time step still it is occurring a resource one resource at time step l .

So, this is how I can identify what are the operations running in time step l . So, now, you can understand this expression.

So, this says that all operations of type k because I have to take each type, so where k can vary from all type of resource. Say for each type of k say let us say this is a multiplier, all the operations that are multiplied. So, all the operations which are of time multiplier I am going to consider and from all such multiplication I have to understand what are the operation that are actually running in time step l . And that I can give how?

It is m equal to l minus d_i plus 1 to l , I have already explained that. Even if it start at l minus d_i plus 1 time step, still it is running in l or it is or l minus d_i or l minus d_i minus 1 and so on.

So, this entire time step it is running still it is running in here. So, that particular concept is modeled by this expression. So, that means, for each operations of type k , I will identify how

many of them are actually running in time step 1 their type the total number of such running operation must be less than of the resource bound of type k.

And this is for each type of resource and each time step. So, that means, for each type of resource for each time step I am going to get a constant. So, if there are say k type of resource and there are λ plus 1 number of time steps, these many number of such constant I am going to get. I hope you understand this. So, let me just take an example.

(Refer Slide Time: 49:44)

Example 1 (cont'd.)

• Resource constraints

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

$$\begin{cases} x_{1,1} + x_{2,1} + x_{6,1} + x_{8,1} \leq 2 \\ x_{3,2} + x_{6,2} + x_{7,2} + x_{8,2} \leq 2 \\ x_{7,3} + x_{8,3} \leq 2 \\ x_{10,1} \leq 2 \\ x_{9,2} + x_{10,2} + x_{11,2} \leq 2 \\ x_{4,3} + x_{9,3} + x_{10,3} + x_{11,3} \leq 2 \\ x_{5,4} + x_{9,4} + x_{11,4} \leq 2 \end{cases}$$

22

So, again let us say take this example and say I have say two type of resource one is the multiplier and one is the ALU, and this a ALU can say it can do plus minus and division. So, ALU is the general purpose hardware unit and there is a multiplier. And say there are two multiplier available, and assumed here the operations are actually single cycle. So, what are the operations can live in time step 1?

So, basically as I tell keep for each type of multiplier for each type of resource each time step. So, I have I will have 4 constraint because this is 4 time step operations, 4 into 1 because for time multiplier. So, I will have 4 constraints for multiplier, I have 4 constraints for ALU. So, I will have 8 such constants.

Some of the constant may not be feasible. For example, here we will discuss that. So, for example, let us now takes the multiplier. So, in the time step, so how many multipliers is

possible? This, 1, 2, 3, 4 because they can execute in time step 1. So, this given by $x_1, 1$, this is given by $x_2, 1$, this is by $x_6, 1$ and this 8, 1; that means, this operation 6 can execute in time step 1. So, although there are 4 possibles, since that resource bound is 2, so only 2 of them can be run can run in time step 1 that is given by this constant.

So, that means, ASAP will schedule all these 4 in time step 1, but because if we have a resource bound is 2, only 2 of them can execute in time step 1. Similarly, in time step 2, this operation can schedule, this can schedule, this also can schedule. So, this can schedule here and this can schedule here. So, this can schedule here and this can schedule here. So, that means 3, 6, 7 and 8.

So, this $x_3, 2$; $x_6, 2$; that means, operation 3 in time step 2, operation 6 in time step 2, operation 7 in time step 2, or operation 8 in time step 2, so total number of such operation must be less than or equal to 2 and so on.

So, similarly for the ALU operation I can identify. I am not going into the detail of each expression. But we can understand for each type of resource each time step I will have a constraint which make sure that the number of running operations in that particular time step will be less than of the resource bound.

(Refer Slide Time: 52:06)

ILP Model of Scheduling - Constraints

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

$\sum_i x_{ni} \leq \lambda + 1$

23

So, this is what the resource constraint. You can also have a latency bound that you have given that this latency λ plus 1. So, latency actually I have already explained that λ will be used to execute this part, this will be 0 and this is λ plus 1. So, this 1 to λ will be used to execute the actual behavior. So, what I have to do? I have to make sure that. So, what how do I decide the time step of this? This is n operation, x_{nl} , this is the execution start time of this operation n, and if I multiply this with l, and if I just do a summation over l this must be less than equal to λ plus 1.

So, that means, I am going to schedule everything and make sure that the sink node must scheduling times to l plus 1. So, then if this is scheduling λ plus 1; that means, that indicates that all the operations in the behavior actually schedule within 1 plus λ . So, it ensures that my latency bound is met. So, this is how I can actually express the latency bound. I hope you understand this.

(Refer Slide Time: 53:15)

Scheduling - Objective Function

- Function to be minimized: $F = c^T t$ where $t_i = \sum_l l \cdot x_{il}$
- Minimum latency schedule: $c = [0, 0, \dots, 1]^T$
 - $F = t_n = \sum_l l \cdot x_{nl}$
 - if sink has no mobility ($x_{n,s} = 1$), any feasible schedule is optimum
- ASAP: $c = [1, 1, \dots, 1]^T$
 - finds earliest start times for all operations $\sum_i \sum_l x_{il}$
 - or equivalently:

Handwritten notes and equations on the slide include:

- $c^T \cdot t$
- $c_1 \cdot t_1 + c_2 \cdot t_2 + \dots + c_n \cdot t_n$
- $e_n \cdot t_n$
- A large handwritten equation: $x_{6,1} + 2x_{6,2} + 2x_{7,2} + 3x_{7,3} + x_{8,1} + 2x_{8,2} + 3x_{8,3} + 2x_{9,2} + 3x_{9,3} + 4x_{9,4} + x_{10,1} + 2x_{10,2} + 3x_{10,3} + 2x_{11,2} + 3x_{11,3} + 4x_{11,4}$
- A circled expression: $(t_1 + t_2 + \dots + t_n)$

So far we have understand that I have this variables x_{il} which determine the start time of each operations, and I have 4 type of constraint I have discussed, one is the this start time must be unique, next is precedence constraint or the dependency constraint, the resource bound and latency amount. And now what could be the objective function? What is the objective function?

So, objectives function as I mentioned it is basically given by this $c_i t_i$, t_i is the start time of the operations. So, this t_i is nothing but this expression x_{i-1} into x_i . And if you now try to optimize this latency if you try to optimize the latency or try to reduce this time step, what is the value? I want to schedule this node as soon as possible, as early as possible. So, that means, I can give this my t_n will become this value, and then this constant c should be $[0, 0, 0, 1]$.

Because I want to; so, that means, what? This is basically 0 into, so you can understand this t_i , this is a multiplication of these two. So, basically it is basically $c_1 t_1$ plus $c_2 t_2$, $c_n t_n$. And this t_i is nothing, but this expression, t_i is the start time of operation i .

And if I want to minimize the latency, so that is if this is my objective function how I can get this value minimum? If I put this equal to 0, c_1 equal to 0, c_2 equal to 0, c_3 equal to 0 and this equal to 1 because otherwise this value will be more.

So, if I just put c_1 equal to 0, c_2 equal to 0, everything is 0, except the c_n which is the basically the sink node constant, so then my this optimization function moved to $c_n t_n$ because other values become 0. And then only I can get this expression value minimum, because I want to minimize the latency. So, to minimize the latency I will take those constant whatever I have just discussed, but I will make my latency constant c is this $0, 0, 0, 1$.

And I will just multiply this with this t_i which is the start time of this which can be represented by this in terms of x_i like this, and then I will get this latency value. So, what I am tried to say is that I have those constant and just try to minimize the latency I just take this constant and if I just run the ILP solver, I can get the minimum latency. Similarly, say I want to do ASAP using ILP formulation. So, ASAP means what I want to start this operation as early as possible.

So, then what I have to do? I have to, so if this is my expression, when this value will be minimum? If this value, if this constants are basically 1 because here thus, I want to get the minimum value of each start time. So, what I am going to do? I am going to put c_1 equal to 1, c_2 equal to 1 and so on. So, then this operation become t_1 plus t_2 plus t_n should be minimum. So, I want to get this value as minimum.

So, when this value, what is t_1 ? t_1 is nothing, but say 6 1 and 6 2, 2 into 6 2 because t_1 is given by this. What is a t_2 ? It says t_2 is say 7, so the 7 is given by this. For 8, 8 is given by, this is the possible start time. And what is the 9? So, for 9, it is 9 to this. So, this is the possible start time of this operation nine. So, when this operation will be minimum? If it can schedule here, if it is schedule here it will be 2, if it is schedule here it will be 1.

So, if I just take the summation of this value obviously, the ILP try to schedule this into time step 1 because if it is state at a schedule in time step 2 it will it this value will become 2, if it is schedule in time step 1. Similarly, for this it will try to schedule in time step 2, not in 3, because if it is try to schedule in time step 3 it will be 3, otherwise it will be 2.

So, you can understand that just having the same constraint and say I want to get an ASAP solution with resource the timing constraint, I can just set the value of this c_1 all 1 and I can run the same ILP, it will give me ASAP solution. With this I conclude today's class. So, in today's class what I have discussed? How we can model this scheduling problem into ILP, what are the possible constraint and how to identify the constraint, and what are the number of constraint is there. So, that we have discussed.

In next class, what I am going to do is I am going to say how I can model this MLRC problem as ILP using this constant, or how can I model this MLRC problem minimal latency under resource constraint as a ILP problem using this constant So, that I am going to discuss in the next class in detail.

Thank you.