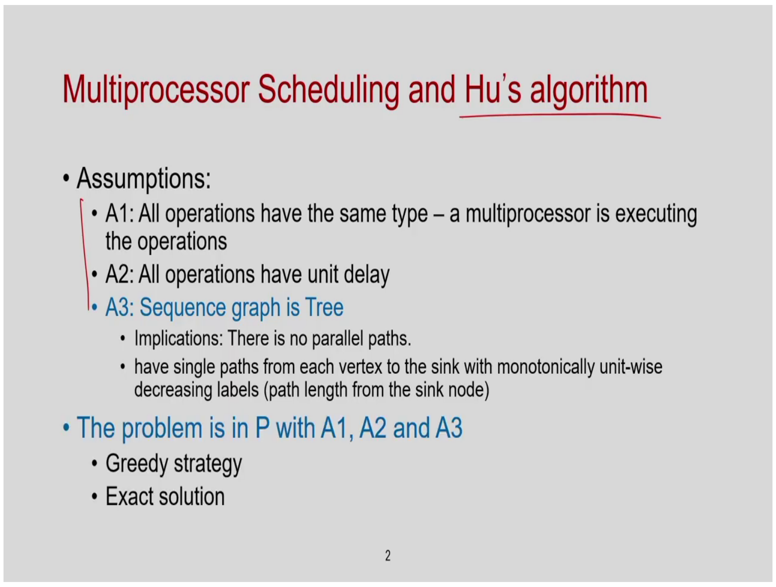
C-Based VLSI Design

Dr. Chandan Karfa
Department of Computer Science and Engineering
Indian Institute of Technology, Guwahati

Module - 03
C-Based VLSI Design: Scheduling
Lecture - 09
Hu's Algorithm for Multiprocessor Scheduling

Welcome students in today's class I am going to discuss Hu's Algorithm for Multiprocessor Scheduling.

(Refer Slide Time: 00:55)



Multiprocessor Scheduling and Hu's algorithm

- Assumptions:
  - A1: All operations have the same type – a multiprocessor is executing the operations
  - A2: All operations have unit delay
  - A3: Sequence graph is Tree
    - Implications: There is no parallel paths.
    - have single paths from each vertex to the sink with monotonically unit-wise decreasing labels (path length from the sink node)
- The problem is in P with A1, A2 and A3
  - Greedy strategy
  - Exact solution

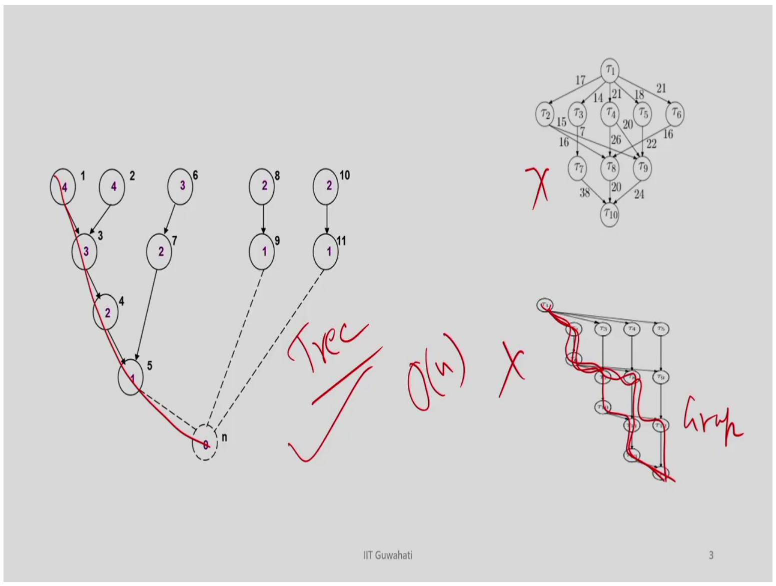
2

As you recall in the previous class I have discussed that in general my sequence graph is a graph where the nodes have a different delay and the type of the operations also delay is different. But if you take some restrictions or some assumptions on this particular input graph, that if the this node type is same that all operations of same type or it can be executed using a multiprocessor.

So, this is my assumption 1 second is that this delay of the operation is all one that means I have a unit delay. And the third assumption that I have taken that the sequence graph is actually tree, the big difference from the tree and graph is that graph can have from any node to sink node there remain many paths. But in a tree from any node to the sink node there is a single path or a unique path.

So, once we have these three assumptions we have argued that this particular my scheduling problem of such graph become polynomial time solvable. What does it mean? I can have an efficient algorithm which is polynomial time deterministic algorithm which can give my optimum results in polynomial time. So, one of this algorithm is called Hu's Algorithm in today's class I am going to discuss about this Hu's Algorithm.

(Refer Slide Time: 02:17)

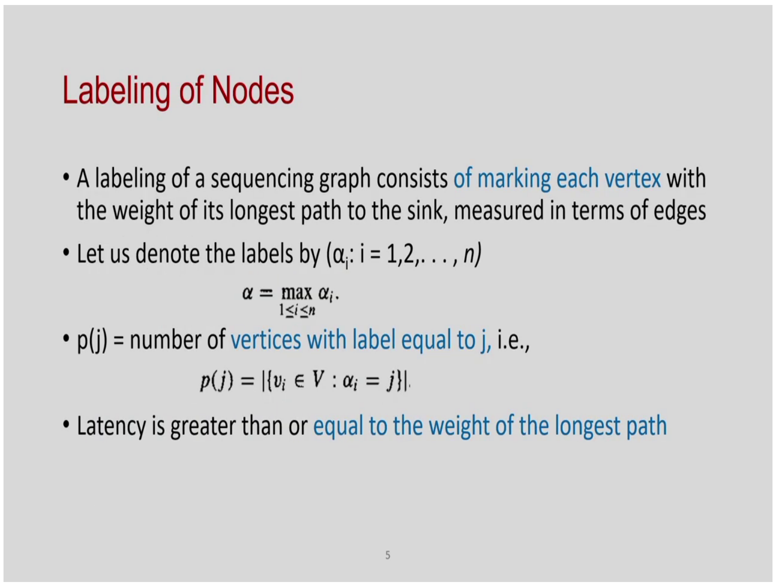


So, I have already discussed so this is the structure called Tree, because from any node there is only 1 path, but these structures are graph because you can have many paths from one node to the sink node you can understand that.

So, today we I will assume that my sequence graph structure is this and the delay of the node is basically all 1 and all are type 1. So, I do not have to bother whether it is a multiplier or a adder or a divider. So, I will assume that all are of same type of operations.

So, the structure of my input sequence graph is basically is this not this or this. So, once this is given what I am going to show that, I am going to explain an algorithm called Hu's Algorithm which solve this problem in very first in order of n time. And I am going to show that this always achieve the minimum latency for a given resource constraint and also for a given resource constraint. It always where is the minimum latency, so that I am going to show the intuitive proof for that.

(Refer Slide Time: 03:31)



Labeling of Nodes

- A labeling of a sequencing graph consists of marking each vertex with the weight of its longest path to the sink, measured in terms of edges
- Let us denote the labels by (α_i: i = 1,2,. . . , *n)*

$$\alpha = \max_{1\le i\le n} \alpha_i .$$

- p(j) = number of vertices with label equal to j, i.e.,

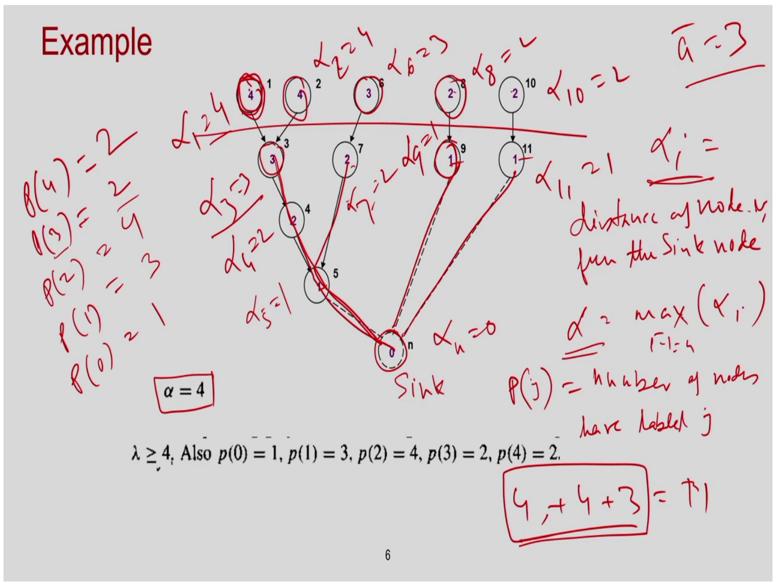
$$p(j) = |\{v_i \in V : \alpha_i = j\}|.$$

- Latency is greater than or equal to the weight of the longest path

5

So, let us discuss the Hu's algorithm now. So, for Hu's algorithm I need to label the nodes. So, let us understand how is to label the nodes.

(Refer Slide Time: 03:39)



Example

$\lambda \geq 4$, Also $p(0) = 1$, $p(1) = 3$, $p(2) = 4$, $p(3) = 2$, $p(4) = 2$.

6

So, we are going to label the nodes using the distance of the node from the sink node, let us try to do that. So, for see this is my sink node. So, the distance from this node is 0 and this is given by alpha, so alpha n equal to 0. So, this is my node 5 and the distance is 1 so this alpha 5 is equal to 1.

For this node distance is to1 and 2, so this is alpha 4 equal to 2 for this node the distance is 3 1 2 and 3. So, alpha 3 equal to 3 and for this node it is 4, so alpha 1 equal to 4. Similarly, for this node alpha 2 equal to 4 for this node alpha 7 equal to 2, because the distance is 2 for this node is distance is 1. So, alpha 9 equal to 1 for this node alpha 11 equal to 1 and for this node alpha 10 equal to 2 and for this node alpha 8 equal to 2.

So, this is how I am going to label each node which is the nothing but the distance from the sink node. So, this is the labelling and we identify. So, this is given by alpha i. So, alpha i is the distance of node vi from the sink node and then I will just take this alpha which is the max of all alpha i, so where i equal to 1 to n.

So, for this node what is the alpha? Because see here the 1, 2, 3 and all, so the maximum value of this distance is 4. So, my alpha equal to 4. So, which is the maximum all such labels and then I am going to define another term p alpha i which is basically or say p j or say p j which is basically the nodes the number of nodes number of nodes have label j, so I have label j.

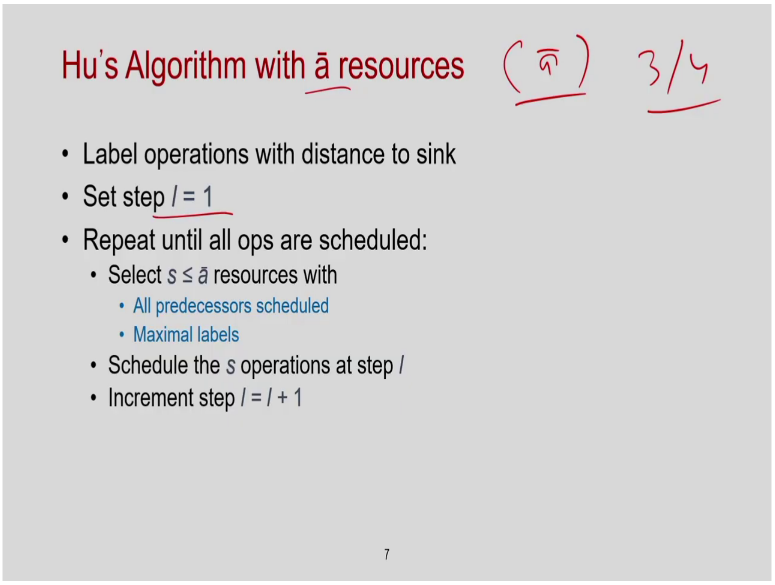
So that means, I have to identify the nodes that has all the nodes of have label 4 similarly label 1 and label 2. So, for this graph how many nodes of p4 how many nodes of having label 4? So, I have a 1 node of label 4 another node of label 4. So, p4 equal to 2, so the number of nodes that has label 4.

Similarly I can identify p3, p3 is basically 1 node 1 node only 1 node and for this node also 3, so alpha 6 equal to 3. So, these are the 2 nodes having distance from the sink node is 3, so there are 2 nodes which has label 3.

Similarly, I can identify p2, p2 how many nodes are there? 1, 2, 3, 4. So, this p2 is 4 and p1, 1 2 3 and p0 is the sink node, so this is where the value. So, this is my initial processing. So, given the sequence graph which I know having satisfying these 3 assumptions that all node have unique delay all of at the same type.

And is it actually tree I can I will identify first the alpha i for each node and then I will find out the what is the maximum alpha i which is alpha and then I also have the p, p is the label and you have to keep an eye on that p j is basically the number of nodes that has label j. So, this timer is very important on the correctness of the algorithm. So, this I am going to do at the start, so this is done. So, this for this information is available with me.

(Refer Slide Time: 07:57)



Hu's Algorithm with $\bar{a}$ resources $(\bar{a})$ $3/4$

- Label operations with distance to sink
- Set step $l = 1$
- Repeat until all ops are scheduled:
  - Select $s \leq \bar{a}$ resources with
    - All predecessors scheduled
    - Maximal labels
  - Schedule the s operations at step l
  - Increment step $l = l + 1$

7

So, now this algorithm is very simple. So, what it is basically does you have given a resource a bar. Because I have only 1 type of resource I have say [FL] I have a bar number of resources this can be 3 4 or something. So, I say I have 3 multiprocessors or 2 multiprocessors.

So, what I am going to do? As I mentioned in the previous class that for this kind of problem, because it is a tree I can actually take a local decision. So, what when you basically if say for example in this graph, so there are 5 nodes. So, I am going to schedule the time step 1 there are 5 nodes which is scheduler and say suppose my bar equal to 3.

So, among these 5 nodes I can take only 3 nodes I can schedule only 3 nodes. So, there are many choice 5 c 3 percent choice are there. So, you can actually take any 3 from this 5 there are many choice which said to be taken.

So, for generic graph this choice is not unique and it is not possible to take such decision at this at the time step 1. But for tree this particular choice is unique and this particular one we can actually take a low voltages and I can actually show that what if I choose that the choice that I have taken at the time step 2 will ensure the optimal schedule in terms of the latency or the number of time step

So, now the question is that the I understand that say I mean for this node there are 5 nodes can be scheduled in time step 1 and say resource bound is 3. So, I can only select 3 of them, but how to select? To select this I am going to use this label, so that is I have calculated this.

So, once I am going to say I have 5 nodes I have to select 3 I am going to select all these 3 which has a maximum label. The sum of label of my selection is maximum among all possible selection and which is very simple that you basically select has the highest value. And why this particular choice is give you the correct results if you try to understand that way, because say for example here I have that selection is 4 4 3 2 2.

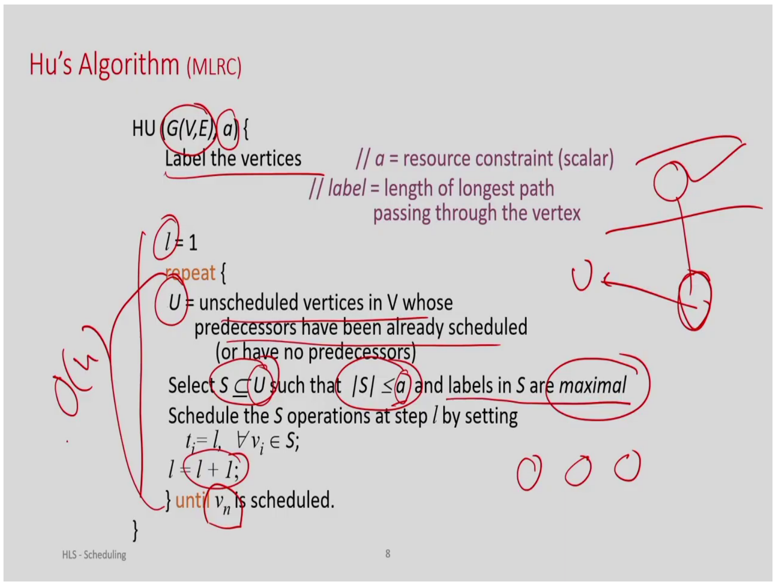
So, I have to select say 3. So, I am obviously going to select 4 4 and 3 because that is summation will be 11 for any other choice this will be summation will be less. And why this is correct? Because this 4 is what the distance of this node from the sink node.

So that means, if I because there are many nodes which is actually line backend or to be scheduled which is subsequent to this node 4 with the label 4. So that means, if I delay this operation further, so that means, this all the scheduling of the all the nodes which is following this node will be delayed.

Here for this node there are 2 nodes to be scheduled after that and for this node there are 4 nodes to be scheduled after that. So obviously, I should select the node which has many nodes to be scheduled after that. So, that means my choice of selecting the maximum labels gives me the best solution, because I am making sure that the subsequent node will get earliest possible time to schedule.

So, that is why this choice based on the label actually gives you the best solution when we can prove that. So, the algorithm is very simple that I will start with my start step 1 and I will identify the all the operations which is schedule label. So, in the sense schedule label means what whose predecessors are already scheduled. So that means, they are already done, I am going to select such set let me just explain here.

(Refer Slide Time: 11:43)



Hu's Algorithm (MLRC)

```
HU (G(V,E), a) {
    Label the vertices        // a = resource constraint (scalar)
                              // label = length of longest path
                                 passing through the vertex
    l = 1
    repeat {
        U = unscheduled vertices in V whose
            predecessors have been already scheduled
            (or have no predecessors)
        Select S ⊆ U such that |S| ≤ a and labels in S are maximal
        Schedule the S operations at step l by setting
            t_i = l, ∀ v_i ∈ S;
        l = l + 1;
    } until v_n is scheduled.
}
```

So, I will start so I have given this sequence graph and some resource bound initially I will do all the labelling. So, this part is done and this is my actual algorithm. So, now I have to identify all the nodes which is actually what was predecessors are already scheduled. That means, these operations are ready to be scheduled now. I will identify such nodes and then as I mentioned that among this nodes, which is actually schedule label I have to select a subset of such nodes such that my resource bound satisfy.

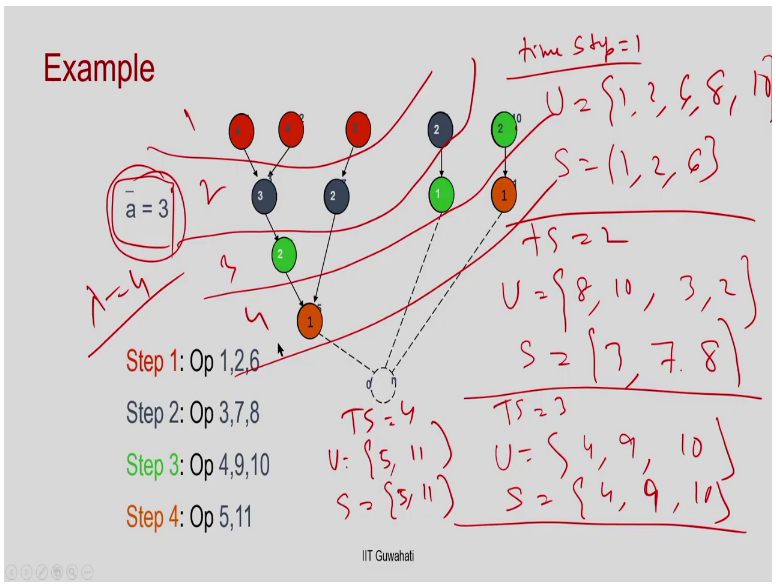
So, I am going to select a subset of this all scheduled label nodes which is S such that by the number of such predecessor node is less than of the resource bound and I can actually. So, how am I going to select this subset above and sorry. So, how I am going to select this subset? The way I am going to select this subset is that this must satisfy the resource bound and the label of the selections is the maximal.

So that means, whatever the node I am going to select from this U the summation of their label will be maximal among all possible selections. And since these nodes have some delay, so I can actually always select the maximum values which is very easy way to do that.

So, whenever I just select this I am going to schedule this S in this time step l and then I will just increase the time step to l plus 1 I am going to do it for next time step and I will keep doing this until I my last node or the sink node is finished. So, you can understand the complexity of this algorithm is kind of order of n, because every time if you select a subset there will be scheduled and the next time I can actually just access there.

So, I can easily if there is a edge like this and if it is scheduled here I can I will just put this node as the in U for the next iteration. So, all nodes will be kind of considering schedule. So, in every time so the number of nodes is n, so this kind of this algorithm will run in or drop n.

(Refer Slide Time: 13:49)



So, let me just give this example. So, say for example for this graph, so the node the number inside the node is the distance or the label. So, these are the labels.

So, for U so I am now in time step 1 time step 1 in time step 1 what is the U? Obviously, I know understand the node 1, 2, 6, 8 and 10. So, these are the nodes which are scheduled label. So, I will just mark them. So, these 5 nodes are scheduled label and say my resource bound is 3. So, which I one I should select? I am going to select these 3 nodes among them whose labels are maximum.

And I have already discussed that for this case I should select 4 4 3. So, then my S will be 1 2 and 6 these 3 node. So, I am going to select these 3 nodes because there summation of the delay is maximum. So, I am going to schedule them in 1, so that means, in time step 1 I just scheduled these 3 nodes.

So, understood so this is what I am going to do. So, this is my time step 1 is done in time step 2 what is my U what are the nodes? So, these 2 has not scheduled so they are still available, so 8 10 and now since these 3 nodes are scheduled now 3 and 2 U also scheduled I mean is available. So, these are the nodes is ready to be scheduled.

So, these 4 nodes, so these 4 nodes are ready to be scheduled. So, I can select 3 so which 3 I should select? Obviously, I am going to select this node because this has this is 3. So, this node is basically say operation is 6 I guess this is node 3 and among this these are all 2 I can select any 2. So, we need can be 7 8 or 7 10 there is you can just select arbitrarily it does not matter. So, let me select say 7 and 8, so that means, in the next clock I just schedule these 3 operations.

So, now this time step 2 is done. So, what is the time step 3? What is my U? So now, this node is yet to be scheduled this is yet to be scheduled and so this was still available from the time step 1. I am not scheduled although it is available from time step 1 I do not schedule it time step 1 or 2 because it has low value low label.

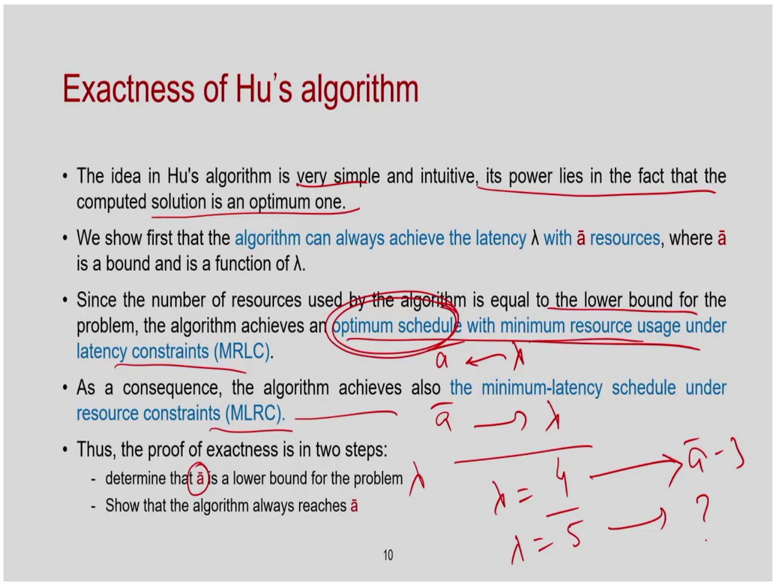
And once you schedule this once you schedule this these 3 nodes are not available, this node is not available because although this data is available. But the other data is not available until all it is predecessors are done I cannot schedule this node.

So, these 3s are available. So, my U is basically 4 9 and 10 and I can select 3. So, there are 3 nodes I have to select 3 I will be going to select all of them. So, I because there are 3 resources available, so I am going to schedule this these 3 nodes in time step 3.

So, once you do this in time step 4 now T s 4 my U is basically only these 2 nodes these are the 2 nodes are now available. So, which is basically 5 and 11 since I have 3 resources I am going to schedule both of them 5 and 11. This is going to be happen in time step 4 and obviously in time step 5 the sink node will be scheduled.

So, that is what is the execution of the Hu's algorithm in this particular example and it is very simple algorithm you understand. But this is very powerful example for this assumptions, it because it always give you the minimum latency.

(Refer Slide Time: 17:45)



Exactness of Hu's algorithm

- The idea in Hu's algorithm is very simple and intuitive, its power lies in the fact that the computed solution is an optimum one.
- We show first that the algorithm can always achieve the latency λ with ā resources, where ā is a bound and is a function of λ.
- Since the number of resources used by the algorithm is equal to the lower bound for the problem, the algorithm achieves an optimum schedule with minimum resource usage under latency constraints (MRLC).
- As a consequence, the algorithm achieves also the minimum-latency schedule under resource constraints (MLRC).
- Thus, the proof of exactness is in two steps:
  - determine that ā is a lower bound for the problem
  - Show that the algorithm always reaches ā

10

$a \leftarrow \lambda$
$\bar{a} \rightarrow \lambda$
λ
$\lambda = 4 \rightarrow \bar{a} - 3$
$\lambda = 5 \rightarrow ?$

So, as I mentioned that this is very simple algorithm, but it is very power lies the fact that it will always give you the optimal solution. That means, once we assume that there are 3 resources available so it give you 4. So, it gives the 4 time step and this is the best for this is the best solution for this. So, this is the 4 time step is given and this is the lambda is equal to 4 you cannot schedule this less than 4 using 3 resources.

So, that something can be proven that for this graph with 3 resources lambda equal to 4 is the minimum possible value. In the other way we can also show that this when this lambda equal to 4 you need at least 3 resource to schedule in 4 time step that is also the other way of thinking this.

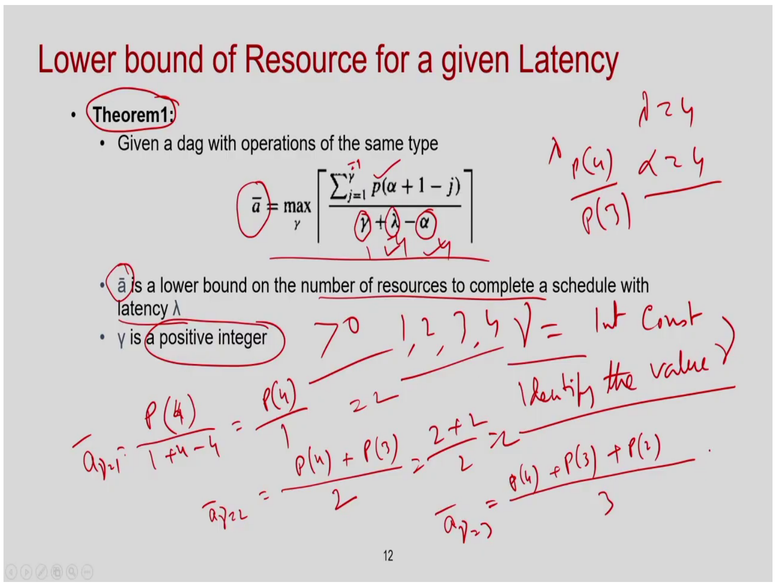
And Hu’s algorithm is something is so powerful because it satisfies both. So, it is basically a one hand it is a MRLC and other hand is also MLRC, because once you given a lower bound of the resource you always optimal schedule or the optimum latency for a given resource.

Similarly, if you just given this is basically I have written here is that this is latency constant. So, if you have given a latency constant lambda it always give you the minimum resource bound and similarly here if you give a resource bound a it always ensure that it actually scheduling minimum number lambda. So, this is what the beauty of this Hu’s algorithm.

So, now what I am going to show is given such sequence graph which is have 3 assumption it is true it is a tree, all operations are unit delay and all of have same type. So, can I identify the lower bound for a given lambda? So, if I say I give you 4 can you have can you identify that for lambda equal to 4 a bar is 3?

So, can you identify this? So, we can actually for any so now if you say if lambda equal to 5. What will be the bar? So, can we have some algorithm proof or some equation which actually can give you the lower bound of the resource for a given latency?

(Refer Slide Time: 20:07)



Lower bound of Resource for a given Latency

- **Theorem1:**
  - Given a dag with operations of the same type

$$\bar{a} = \max_{\gamma} \left\lceil \frac{\sum_{j=1}^{\gamma} p(\alpha + 1 - j)}{\gamma + \lambda - \alpha} \right\rceil$$

  - $\bar{a}$ is a lower bound on the number of resources to complete a schedule with latency λ
  - γ is a positive integer

$\lambda = 4$

$\frac{p(4)}{p(3)}$ $\alpha = 4$

$\gamma > 0$ 1, 2, 3, 4 $\gamma =$ Int Const

Identify the value γ

$\bar{a}_{\gamma=1} = \frac{p(4)}{1+4-4} = \frac{p(4)}{1} = 2$

$\bar{a}_{\gamma=2} = \frac{p(4)+p(3)}{2} = \frac{2+2}{2} = 2$

$\bar{a}_{\gamma=3} = \frac{p(4)+p(3)+p(2)}{3}$

12

So, I am going to show this with this equation. So, this theorem 1 actually captures this, that if you have given a latency bound lambda which is this I can actually identify the lower bound of the resource using this equation I am going to explain it. So, this bar is the lower bound of the resource needed to schedule with latency lambda.

So, this is very important because I can actually given a lower lambda I can always identify what is the minimum number of resource needed to execute this. So, let us try to understand this equation now. So, you can see here that there is a. So, what is given to me I know this alpha value, because alpha was the label of the node at. So, maximum label.

So, in this particular equation let us try to understand what is known to me. So, this is known to me because this is the latency bound is given to me, this alpha is the maximum label of the nodes that is also known to me and this p value is also known to me.

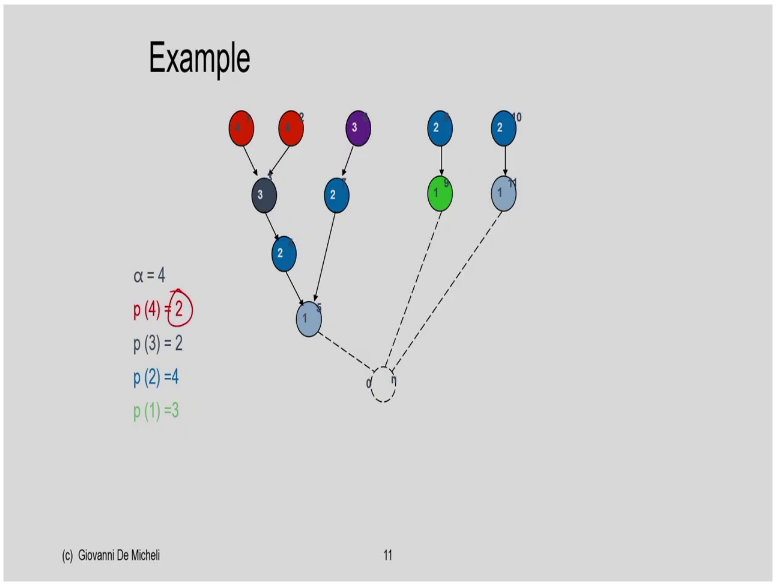
Because p of 4 is basically the number of nodes which has the label 4, p of 3 is the number of nodes which has the label number of nodes which has the label 3 and so on. So, this all are known to me only thing is not known to me is the gamma. So, this gamma is some constant, so it is an integer constant.

So, we need to find out some value of gamma it can be any integer value. So, gamma is actually a positive integer greater than 0, we have to identify the value of gamma such that this expression gives you the maximum value. So, that identifying this gamma is the things that you have to do. So, identify the value of gamma. So, this gamma which give you the max value, so that is the things we have to do.

So, now let us try to go into little bit more. So, since the value of this gamma can be 1 2 3 4 and so on, let us try to put those values. So, let us say I put gamma equal to 1. So, how this equation looks like? If you put gamma equal to 1 this is the summation is basically become p alpha plus 1 minus 1.

So, this become p alpha and if you put gamma equal to 1 here and this so for this example you so I sell my lambda equal to 4 and I know alpha will also 4. So, this is 1 plus 4 minus 4. So, which basically this is alpha equal to 4. So, these become p4 by 1, so that means the number of nodes having label 4 and in my example such node is 2.

(Refer Slide Time: 23:13)



Example

$\alpha = 4$

p (4) = $\boxed{2}$

p (3) = 2

p (2) =4

p (1) =3

(c) Giovanni De Micheli 11

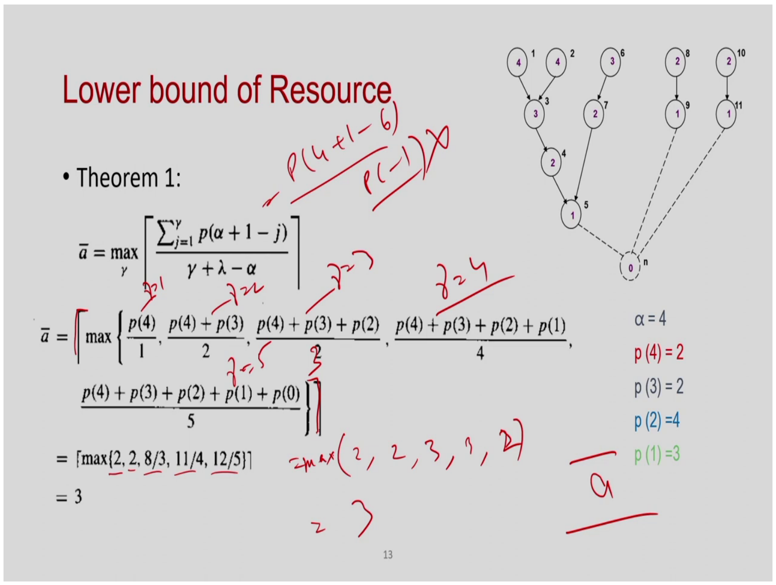
So, this will give me the value 2. So, when this gamma equal to 1 I get a bar equal to 2. So, my a bar equal to 2 when gamma equal to 1. So, now let us take gamma equal to 2. So, a bar when gamma equal to 2. So, if we just put this value so it is a summation. So, when you say summation you have to consider both the thing. So, when gamma equal to 1 I know this is p 4 when gamma equal to 2 it is basically p 4 plus 1 minus, so it is p 3.

And here this is 4 minus 4 it is 2 because this is also 4 this is also 4 it is whatever the gamma. So, what do we understood it is basically the average of 2 the nodes number of nodes which has label 2 and the number of nodes which has label 3. So, in my example it was 2 plus 2. So, this is 2 plus 2 by 2 which is 2.

Now, let us try to put gamma equal to I am sorry this say a bar when gamma equal to 3. So, let us try to understand so when gamma equal to 1 I know it is p 4, when I put gamma equal to 2 in this equation j equal to 2 it is p 3 and if I put gamma equal to 3 it is alpha minus 2 which is 4 minus 2 is p 2 and here it is always this gamma is 3.

So, it is basically the average of the number of nodes have label 4 plus number of nodes having label 3 plus number of nodes having label 2 by 3. So, this is basically identifying all such values and then you try to find out the maximum value of gamma I am going to consider that gamma for which this expression has a maximum value.

(Refer Slide Time: 25:19)



Lower bound of Resource

- Theorem 1:

$$\bar{a} = \max_{\gamma}\left\lceil \frac{\sum_{j=1}^{\gamma} p(\alpha+1-j)}{\gamma+\lambda-\alpha} \right\rceil$$

$p(4+1-6)$ $p(-1)$ ✗

$\gamma=1$ $\gamma=2$ $\gamma=3$ $\gamma=4$ $\gamma=5$

$$\bar{a} = \left\lceil \max\left\{ \frac{p(4)}{1}, \frac{p(4)+p(3)}{2}, \frac{p(4)+p(3)+p(2)}{3}, \frac{p(4)+p(3)+p(2)+p(1)}{4}, \frac{p(4)+p(3)+p(2)+p(1)+p(0)}{5} \right\} \right\rceil$$

$= \lceil \max\{2, 2, 8/3, 11/4, 12/5\} \rceil$ $= \max(2, 2, 3, 3, 3)$

$= 3$ $= 3$

$\alpha = 4$

$p(4) = 2$

$p(3) = 2$

$p(2) = 4$

$p(1) = 3$

$\bar{a}$

13

So, if I take this a graph. So, as I already explained for gamma equal to 1 this is my gamma equal to 1, this is already I have explained gamma equal to 2 this is my gamma equal to 3 and it is gamma equal to 4.

So, this is when I have not shown but you can understand this is how things will go. So and I cannot go beyond and then gamma equal to 5 is this, I cannot go beyond gamma equal to 5 because then this there is no minus value. So, this will become minus and which is not defined.

So, if you go gamma equal to 6 this will become p plus 4 plus 1 minus 6 become p minus 1. So, there is no p minus 1, so I will my gamma also vary from 1 to 5. So, this is not true. So, I have to find out this the value of this. So, if I just put this value because I know the values of these nodes before and all.

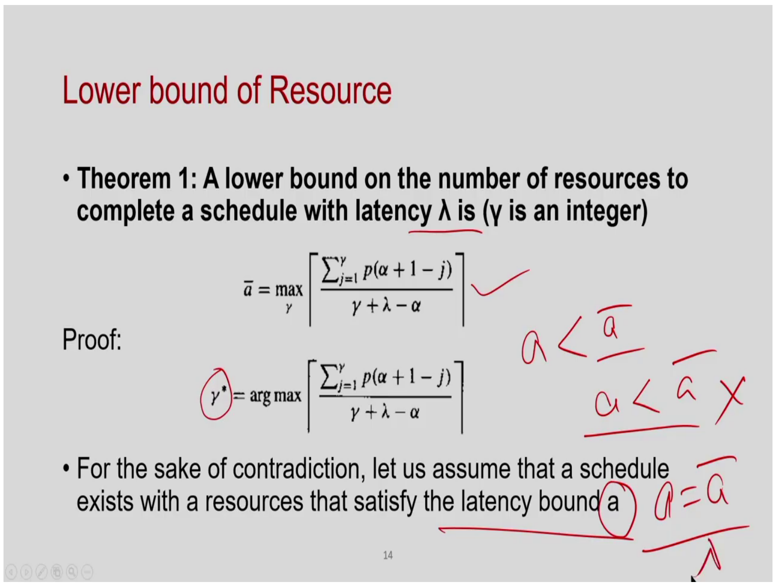
So, if I just put this value it will come the max of 2, 2, 8 by 3, 11 by 4 and 12 by 5 which is since it is a ceiling because you cannot have a resource which is point. So, it is basically 2 you can write this. So, 2, 2 max of 8 by 3 is also 3, 11 by 4 is 3 and 12 by 4 is also 3.

So, or you can actually this is 2 it is fine. So, this is basically come as to be 3. So, I need 3 resources for to schedule these things in 4 time step, so that is what this expression gives you. And intuitively if you understand this is basically either you identify the nodes that having you need to identify the resource needed to schedule 4, the number of nodes which has the label 4 or you identify the nodes which has label 4 or label 3 and you take average.

Or you identify the nodes having the label 4 3 or 2 and you take a average. So, this will be 3 here and or you take the nodes which has label 4, 3, 2, 1 and you divide it by 4 you take an average or you identify the nodes that having label 4, 3, 2, 1 or 0 and you take a average this is what this expression does in available.

So, this is very important in the context of this lower bound. So, here if I given latency I can actually identify the this lower bound of the resource.

(Refer Slide Time: 27:53)



Lower bound of Resource

- **Theorem 1: A lower bound on the number of resources to complete a schedule with latency λ is (γ is an integer)**

$$\bar{a} = \max_{\gamma} \left\lceil \frac{\sum_{j=1}^{\gamma} p(\alpha + 1 - j)}{\gamma + \lambda - \alpha} \right\rceil$$

Proof:

$$\gamma^{*} = \arg\max \left\lceil \frac{\sum_{j=1}^{\gamma} p(\alpha + 1 - j)}{\gamma + \lambda - \alpha} \right\rceil$$

- For the sake of contradiction, let us assume that a schedule exists with a resources that satisfy the latency bound a

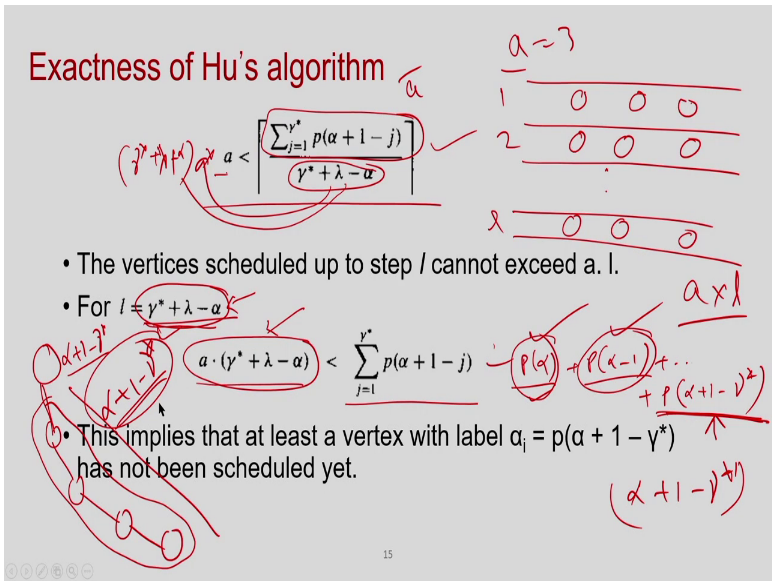
14

So, let us try to prove this why this is this holds. So, I will just give a very quick proof of this; let us say I got the max value of what this gamma star. Because for this example my gamma star is actually 3.

So, because it is there are many possibilities I assume that this will this expression will be maximum when my value of the gamma is gamma star. And let us say for I i want to prove it by contradiction that I want to say that no I even I can actually reschedule the things less than a bar, I can actually I need say need a which is less than a bar.

I need a number of resource which is less than the a bar that is given by this equation, to schedule this behaviour within lambda that I assume. So, as a proof of contradiction I assume that this my resource requirement is a which is less than of this resource bound given by this equation to schedule this behaviour within the latency lambda.

(Refer Slide Time: 28:57)



Exactness of Hu's algorithm

$$a < \left\lceil \frac{\sum_{j=1}^{\gamma^*} p(\alpha + 1 - j)}{\gamma^* + \lambda - \alpha} \right\rceil$$

- The vertices scheduled up to step l cannot exceed a. l.
- For $l = \gamma^* + \lambda - \alpha$

$$a \cdot (\gamma^* + \lambda - \alpha) < \sum_{j=1}^{\gamma^*} p(\alpha + 1 - j)$$

- This implies that at least a vertex with label $\alpha_i = p(\alpha + 1 - \gamma^*)$ has not been scheduled yet.

15

$a = 3$
$a \times l$
$p(\alpha) + p(\alpha - 1) + \dots + p(\alpha + 1 - \gamma^*)$
$(\alpha + 1 - \gamma^* + 1)$
$\alpha + 1 - \gamma^*$

Now, let us see so I have assumed this is my resource recovery which is less than of the maximum this is a bar. Now let us try to understand one thing say suppose. So, since there are a number of resources. So, how many maximum number of person can be schedule in time step 1 is a number of say a equal to say 3. So, I can maximum schedule 3 operation.

Because that is the maximum number of resource in time step 2 also I can schedule up to 3 maximum 3. So, if I just go this way if I mean l time step maximum operation can be scheduled a into l which is very obvious, because in every time step maximum a number of operation can be scheduled and there are l such time step. So, maximum possible operation that is scheduled so far will be a into l.

Now, if we assume this l equal to this which is this part. So, if I assume this then the maximum number of operation will be scheduled till this time step will be this into this. So, this into this which is nothing but I am just brining these things here, I am just putting this gamma star plus alpha minus this, so gamma lambda plus alpha.

So, I am just saying this is the maximum number of operation which is scheduled still these time step and what is this it is less than this I just get it from this expression. Because I just moved this part here, so this will this part will come this side and this is what I have written here.

So, what is this? So, this is basically if you just put we try to write this summation it is p4. So, because my alpha so or instead of 4 I will should write alpha because this is a generic group and if I put gamma j equal to 2 it is basically p alpha minus 1. So, when alpha equal to 1 it is p alpha. So, when it is gamma so it is basically p of alpha plus 1 minus gamma star.

So, these are the labels that are already done I mean this number of operations that are scheduled is must be this label. So, among this label the way I do the Hu's algorithm now if I try to do this must be done, because this will be considered first then this will be considered because their label is highest. If you remember when I do this Hu's algorithm what I am going to do? I am going to choose the nodes that has the highest label that I am going to do.

So, if we assume that this summations this the number of operation that is scheduled still this time step is less than this and among these operations, which is the operation that likely to be unscheduled. This must be this the nodes with this label because these are the highest label this has the higher label this must be done.

So, since this is less than this; that means, at least some of the node which has label this has not schedule it. Understood it is very important point here, I am saying that the number of operation that scheduled till this time step is this which is less than of this and this is nothing but this expression. So, among this expression the so basically what this expression since the number of node having the label alpha or number of node having the label alpha minus 1 and so on.

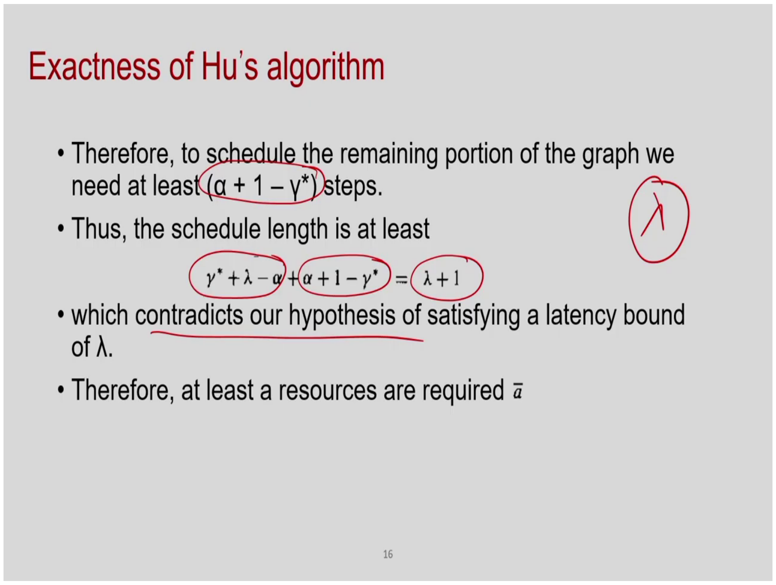
So, alpha is the maximum label, so this is the nodes is maximum label this is the node with the next set of labels so on.

So, among this since these nodes this delay has less than this, so that means among this node I can safely assume that at least 1 node with this label, that label with alpha plus 1 minus gamma star is still to be scheduled because this is not scheduled. So that means, I have a node which label is alpha plus 1 minus gamma star is not scheduled here.

So, what this label means that means the length of the path is this, this alpha plus 1 minus gamma star and if since this is the length. That means, at least to schedule the rest of the behaviour I need at least this many time step, because in 1 time step I can only schedule 1 label I cannot schedule multiple labels.

So, what does it mean? To schedule the rest of the behaviour I need at least these many time step, because this is the length of these things. And so far I have taken my I am actually helical to this so total time will be this plus this.

(Refer Slide Time: 33:41)



Exactness of Hu's algorithm

- Therefore, to schedule the remaining portion of the graph we need at least $(\alpha + 1 - \gamma^*)$ steps.
- Thus, the schedule length is at least

$$\gamma^* + \lambda - \alpha + \alpha + 1 - \gamma^* = \lambda + 1$$

- which contradicts our hypothesis of satisfying a latency bound of λ.
- Therefore, at least a resources are required $\bar{a}$

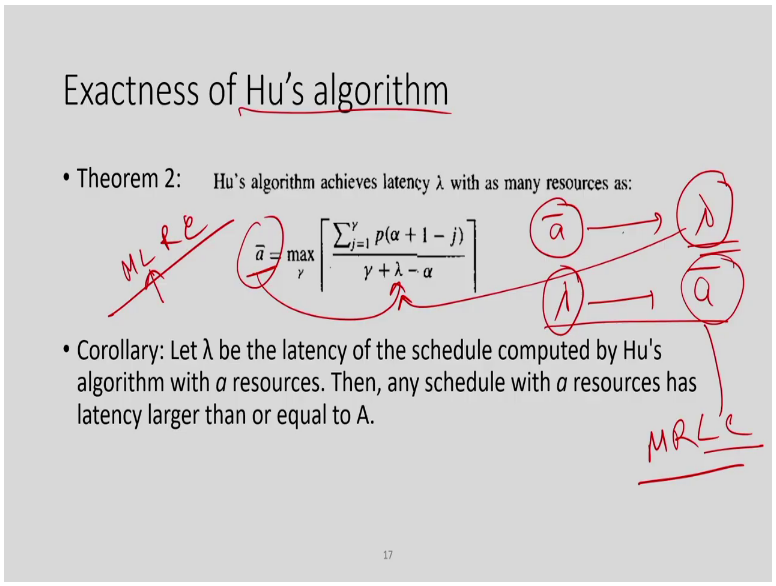
λ

16

So, as I mentioned that there is some node which has label this has not scheduled yet and to schedule them I need this many time step and I have already in helical to this. So, I have total time will be this plus this which I have written here. So, this is the current time step and this is the time step to be needed and if I just do a summation it is lambda plus 1.

So, but then it actually contradicts assumption because I have seen that I can schedule my behaviour within lambda with a. I am assuming a which is less than a bar. But I have shown that if you take any resource is less than a bar your time will be taken is lambda plus 1 which is not lambda. So that means, my assumption is wrong that whatever I am assume that my a is less than a bar is a wrong assumption. So, in a must be a bar then only I can achieve a lambda.

(Refer Slide Time: 34:43)



Exactness of Hu's algorithm

- Theorem 2: Hu's algorithm achieves latency λ with as many resources as:

$$\bar{a} = \max_{\gamma} \left\lceil \frac{\sum_{j=1}^{\gamma} p(\alpha + 1 - j)}{\gamma + \lambda - \alpha} \right\rceil$$

- Corollary: Let λ be the latency of the schedule computed by Hu's algorithm with a resources. Then, any schedule with a resources has latency larger than or equal to A.

MLRC
$\bar{a} \rightarrow \lambda$
$\lambda \rightarrow \bar{a}$
MRLC
17

So, this is how I prove it; I am not going to prove the other correct exactness of the Hu's algorithm, because a lot of theory involves there. So, what this the exactness of the Hu's algorithm is important in the sense that if I give a bar number of resource and I can ensure that Hu's algorithm always give you lambda and you can understand this a bar is depend on lambda.

So, this is basically interdependent for a given lambda I can always identify the a value resource bound by this equation and Hu's algorithm ensures that if I give this resource bound. So, basically MLRC it will always achieve the minimum latency which is lambda for this given resource. And similarly if I given this lambda Hu's can schedule this behaviour within lambda times T using a number of resource; so which is basically nothing but MLRC.

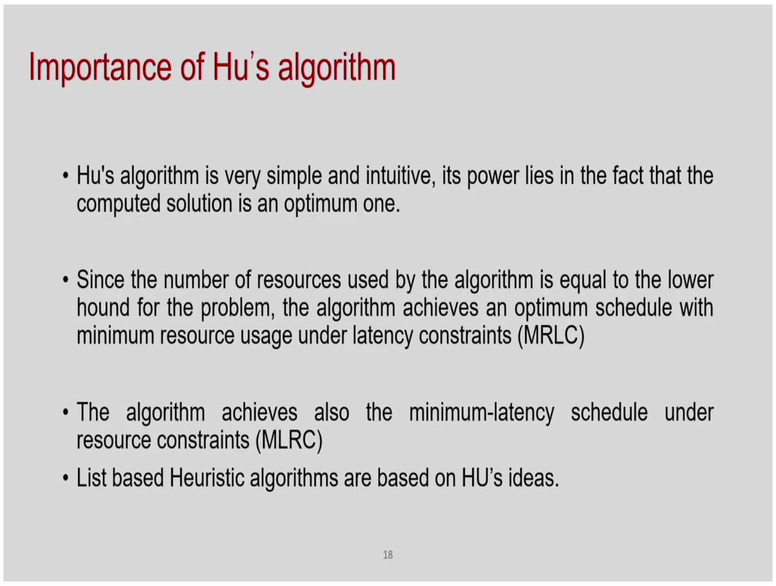
So, in general it is actually given optimal solution for in both contexts either it is a MLRC or MRLC in both contexts. So, this is basically sorry this is actually minimization of the latency under resource constants and this is sorry this is actually MR minimization of resource under latency also. Because I am given this lambda the latency constant and I am ensuring that I can schedule my behaviour within lambda using a bar nominal resource and which is the lower bound.

So, I can actually I can actually able to achieve the lower number of or the lowest number of resource hence is the optimal solution. And similarly here for this MLRC if I given a bar number of resources I am ensuring that I am able to schedule this behaviour within lambda and lambda is something is the target is given and for this lambda this is the minimum resource. So that means, this is actually able to achieve the minimum latency as well.

So, this is why this algorithm is actually important and one of the most another important aspect is that when obviously this my graph will not be tree for generic case. So, this may not be applicable for my practical purposes Hu's algorithm, but as I mentioned in classes that for in practical perspective this is this in scheduling problem is np complete usually we go for heuristic algorithm.

So, heuristic algorithm is something where we apply some heuristic and those and one of the most popular algorithms is called list scheduling. So, this list scheduling actually the idea of this labelling of the nodes from this Hu's algorithm, so that is why I can actually make this Hu's algorithm generic for a generic graph using list scheduling which heuristic by solution which may not give you the best solution always.

(Refer Slide Time: 37:43)



Importance of Hu's algorithm

- Hu's algorithm is very simple and intuitive, its power lies in the fact that the computed solution is an optimum one.
- Since the number of resources used by the algorithm is equal to the lower hound for the problem, the algorithm achieves an optimum schedule with minimum resource usage under latency constraints (MRLC)
- The algorithm achieves also the minimum-latency schedule under resource constraints (MLRC)
- List based Heuristic algorithms are based on HU's ideas.

18

But it actually is very useful and in most of the practical purposes we usually apply list scheduling. So, that is why I covered this Hu’s algorithm in detail and which is the backbone of this heuristic practical algorithm for generic scheduling problem. So, with this I conclude today’s class.

Thank you.