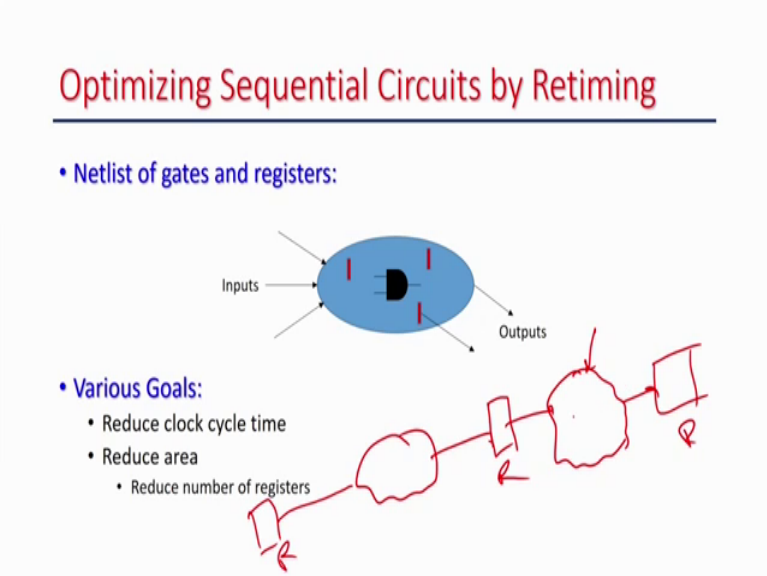
Optimization Techniques for Digital VLSI Design

Dr. Chandan Karfa
Dr. Santosh Biswas
Department of Computer Science & Engineering
Indian Institute of Technology, Guwahati

Lecture – 07
Retiming

Welcome everyone, today we are going to discuss on retiming. We have discuss about several optimization technique to improve timing of a digital circuit in the last class and one of the technique we have get overview is retiming is today we are going to discuss about retiming in more detail. Specifically we are try to show how to solve this retiming problem or how to formulate the retiming problem and how to solve it algorithmically. So, both aspect we are going to discuss today. So, as if you see in a circuit consist of several set of registers and combinational units like gates, multiplexer, demultiplexer and all those terms right.

(Refer Slide Time: 01:11)

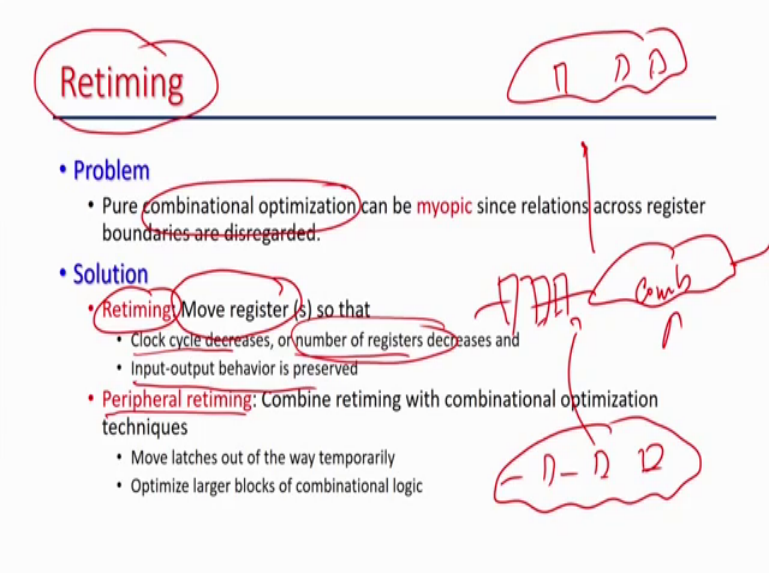


And all objective is always try to improve either timing or area right and if you think about a circuit, we there are 2 options we can go for combinational optimizations or sequential optimizations. So, what do you mean by sequential combinational optimizations we suppose? So, we can think of there are 2 registers and there are some combinational unit in between right. So, you can visualize your your circuit is like this

right. So, there are set of registers and in between there are some combinational clouds right.

So, they set of functionality that modify the outputs right. So, suppose this is something like this and if we can do several kinds of optimizations on the combinational circuit, which is easy where we can try to merge to some set of gates using reduce a size and or several kind of optimization we can do right. But at the end of the day if the registers are very far and the length of this combinational circuit is very long.

(Refer Slide Time: 02:21)



So, whatever the optimization we are going to do combinational optimization we are going to do on this combinational unit between 2 register that does not be sufficient that may not reach a target clock right. So, what I am trying to say is if say this combinational delay is say 200 right millisecond and this is say only 10 say 20 millisecond. So, you can understand that this combinational is unit is smaller than this. So, whatever the optimization going to do you may not you are said the design may not reach the target clock. So, in that case, we have to do some kind of sequential optimizations and one of the sequential optimization is retiming what is this?

We try to move the registers right. So, what we can do instead of put putting this register here, what we can do; we can what we can do here? We can have the smaller combinational unit here and then we put the register here and then another combinational unit here right. So, so this maybe say 100 and this is 100 right.

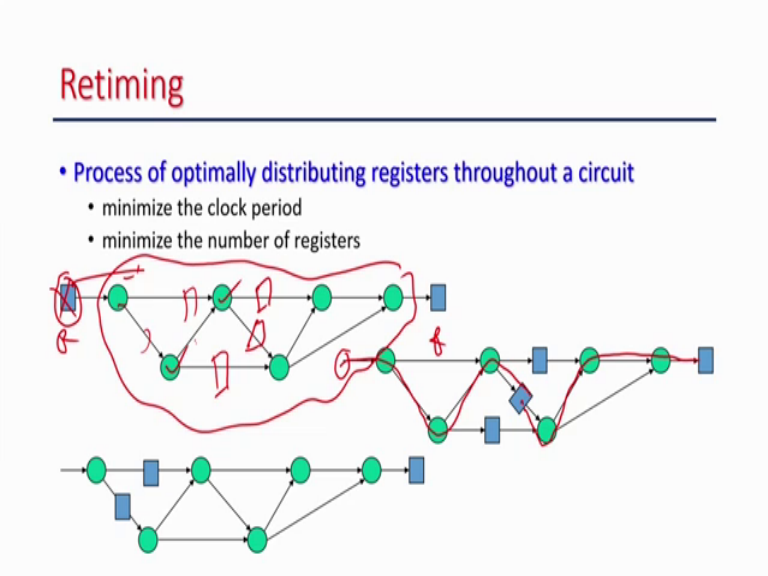
So, now, we can do some optimizations here in this combinational circuit. So, this is the idea that retiming is something we are going to move this registers in the circuit move the registers. So, that we can either reduce the number of clock period; that means, the combinational delay between 2 register is less, as I show here earlier it was 200, and now I just moved the register and place in little bit in between and it become 100 milli millisecond now.

So, this is what is retiming. So, this is a sequential optimizations where we are going to move the register in the circuit, so that we can improve either the clock period or the number of registers right. So, both can be your objective; either reducing the number registers or reducing number of the clock period or maybe both. So, that can be your target right. And we have to be about to make sure that the input output behavior is preserved; that means, the functionality of the circuit does not change because of this particular optimizations right.

So, this is what is called retiming. So, in general moving the register in the circuit to improve the timing of the circuit or improve the number of registers in your design ok. And also you have another type of retiming called peripheral retiming which is basically in combine with combinational optimization it works. So, basically you just place all your registers at the input or output right. So, that you can make your whole circuit is combinational and then you apply all the combinational optimization technique on that combinational circuit. So, that optimization can be better right because if you have bigger circuit the scope of optimization is high. So, we do that and then again you move the registers in the circuit. So, what I am try to say is that, suppose you have a big circuit and there are some registers here and there. So, what we do? We just move the registers at the boundary right. So, we place all the registers here.

And this become a combinational circuit right; now we are going to apply this optimizations in this combinational circuit, and then after make the modification we again move back the registers in the circuit in between. The benefits of is that if we have smaller combinational circuit the scope of optimization is less, but if you have a bigger circuit the optimization scope or possibility is high. So, we have we have done this optimizations in a bigger combinational unit and then again you move the registers back to the actual circuit. So, it is called peripheral retiming. So, it is actually walks with the combinational optimization technique to improve the circuit performance.

(Refer Slide Time: 06:09)



So, here in the example as I mentioned that, we try to move the registers are how we move. So, suppose I have this circuit initially and these are the registers right this is a register, this is a registers and this is that combinational unit what I am talking about right. So, this is that combinational unit in between. Say these circles are either and gate or gate or some gates ok. So, what is happening here you can see that the length of this maximum length here is 1, 2, 3, 4 right. So, we have maximum 1, 2, 3, 4, 5. So, maximum combinational length is this right. So, this is 5. So, number of gates in a path is 5 right and. So, whatever the delay you are going to get that summation of the delay of for this combinational unit right. So, all and gate or gate all them has some delay associated with that. So, we need some processing time that is the combinational delay of that particular node.

So, the maximum clock period will be determine by the 5 this 5 unit of gets delay right and what we can do here by retiming we can move this registers in the inside the circuit right. So, what will happen here? Suppose I want to remove this registers. So, I want to remove this register from here I want to move it inside right. So, what will happen? So, this will be placed. So, when you are going to move inside? It has to place all the fan out right. So, it will be placed here and here and then even I if I want to move this register further, because just reduce the delay by one right. So, now, still you have 5, 4 because I just move it to 1 right.

So, I can move this register again to this. So, what I can do? I can just move this registers inside the circuit. So, if I move this it will. So, I cannot move this registers here, because there is no register here. I have a register here and I do not have any registers. So, I can move only register to the output only when both of the fan in sub registers. So, this cannot be move, but this can be moved right because it is only one fan in to this node and this has registers.

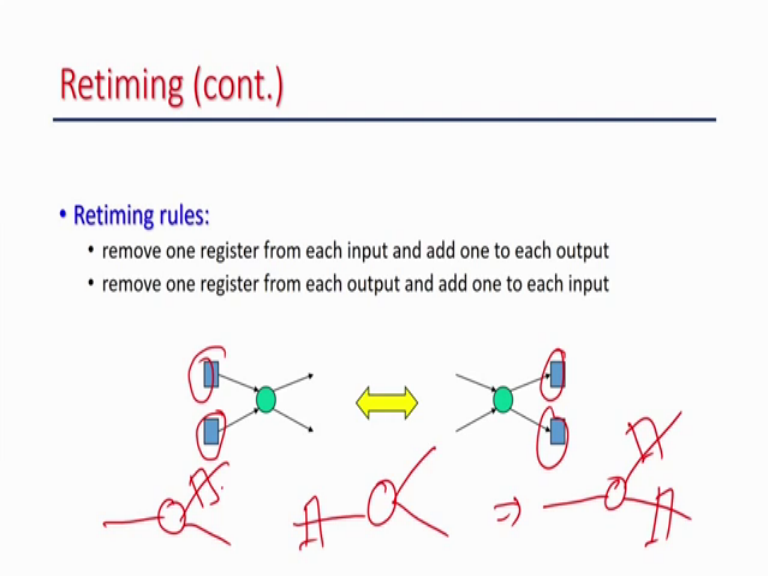
So, I can move this register. So, so I can move this registers to both the output. So, this circuit will become now like this right. Now I can move one register to the output because both the input has of registers. So, I can move this both the registers from here and I can place at the output ok.

So, this is what is shown here right. So, this is after the first step, I move this to register here and then after that I the circuit look like this right. So, now, we can see the maximum delay is 2 right or 3 because from this register to this register I have 3 gates from this from this input to this register I have maximum delay is 3. So, from 5, I have reduced to 3. So, this is what. So, what retiming does its moving the resistors in the circuit? So, that it reduce the combinational delay that we have to keep in mind that the certain rules are there, we can only move register from input to output only when the both the fanins have register then we can move or if I move a register from fan in to fan out it has to move to all the outputs, then only the functionality remain the same.

So, this is the core concept of retiming and you can understand the given a bigger circuit, you have lot of options right lot of possibilities of moving of registers and this circuit is not only have this 8 or 10 gates or only 2 registers. Here circuit may have say 10,000 20,000 registers million gates and maybe more than that also right. So, then doing this things is physically or manually is not possible. So, we have to automate this process. So, given the circuit and the delay of this nodes combinational delay of each node if I give to the tool and then we have to make it automate right. How to automatically do this retiming? So that finally I can reach from this circuit to this circuit.

So, that is what we will going to discuss more detail here. So, we have understood what is retiming and next set of discussion will be going on, how to automate this process how we can automatically does this in a digital circuit that we going to discuss further.

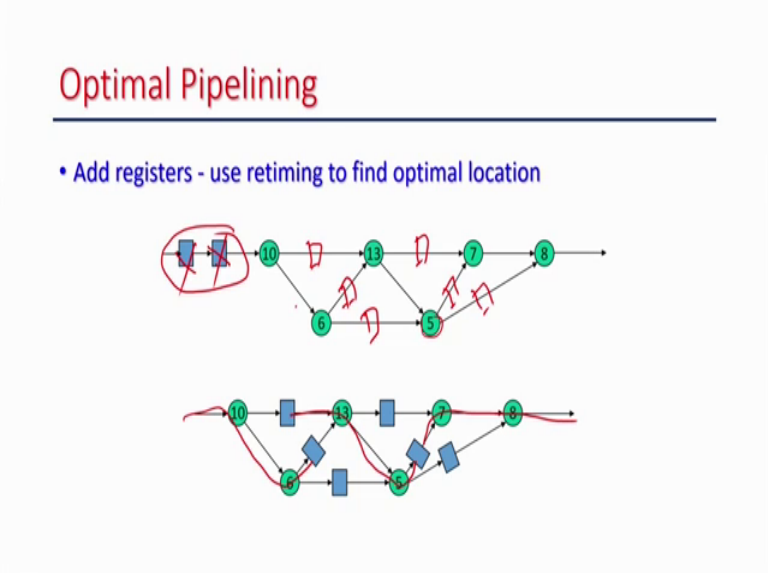
(Refer Slide Time: 10:28)



So, as I mentioned retiming rules is something like this. So, if you have registers in both the input, then only I can move it to output and then if I move to output it will move to both the outputs right. So, suppose you have like this, you have a register here then in the output I have to move both the output right. So, that I have already discussed. Similarly I cannot move if there are 2 fan outs only register here I cannot move it to here because other registers when other path does not have a register. You can understand the functionality of that particular circuit will change if we have move only one register from here to here because this path does not have any register.

So, this is the retiming rules. So, if you have to move all fan in have register I can move them to the output to all the fan out of the register or vice versa right. So, this is the retiming rules.

(Refer Slide Time: 11:24)

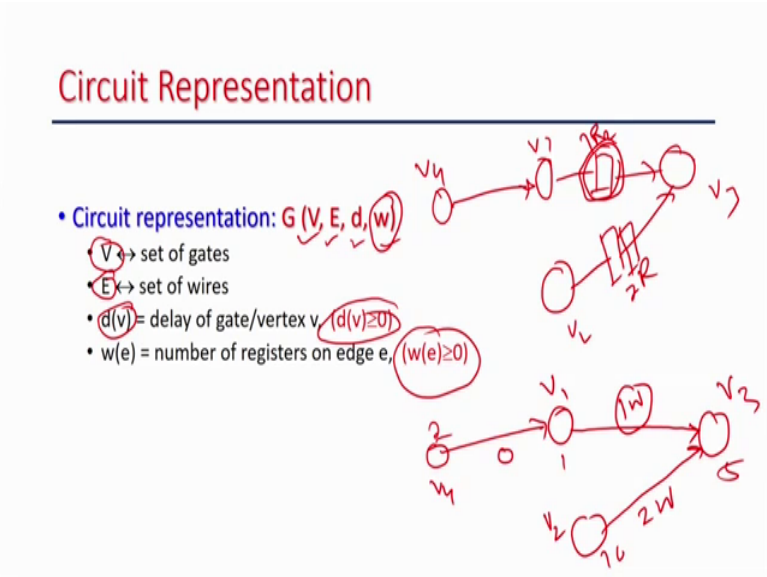


So, again we can use this register, I mean retiming to do optimal pipelining something like this you can place the register to the input and you never know how to place because this maybe a very complex circuit, you do not know what are the parallel paths and all those things and you can utilize that it is a retiming to do this right. So, what we can do I have just shown in the earlier, I can take this register, I can place here and here then I can remove this register I can move this register to the output. So, I can move here right and then since I have both the input have registers. So, I can remove this registers and I can place them in the output right. So, I can place them here output at the output.

Now, this two this two for this node there are both the input have registers. So, I can place I can place them at the output right. So, this I can do. So, I can remove this registers, this is what I have done. So, this is this first set of registers. So, I have removed this register, then I can take this registers and I can move again. So, if I move this register I can place them here similarly then I can move this register to this place right. So, this is the final structure we will get. So, placing the register at the input and then use retiming to move them into circuit you can understand here. So, initially the circuit has the maximum delay of 5 now, you can see the maximum delay is maximum 2 right. So, this is the maximum delay path this is the maximum delay path and this is the maximum delay path.

So, from this 3, I move a from 5 I move it to 2 right. So, this is also we can done. So, pipelining in the sense we place the extra registers at the input and then we can utilize the retiming tool to move the registers in the circuit, to make to improve the pipelining right. So, this is something we can also do ok.

(Refer Slide Time: 13:14)



So, now we are going to talk about the automation part, how to do the retiming automatically. So, for that we have to model that circuit using a graph right how we can do that? So, given a circuit, suppose I have a circuit like this; so this is a gates there is a register here. So, I will just take a simple example, suppose I have this registers. So, this is a node v 1, this is V 2 and this is V 3 and this is a register R 1. So, there are 2 registers here right 2 2 R this is 1.

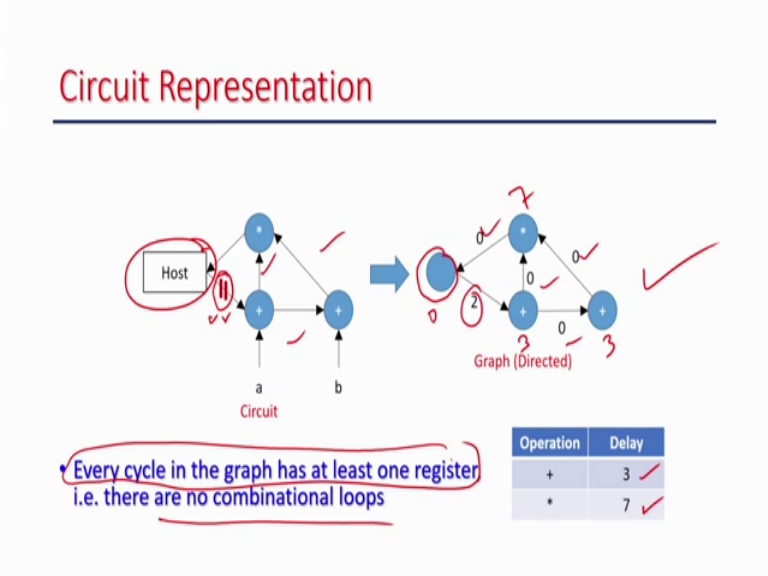
So, what I am going to do here, I am going to construct a graph from this circuit, where I have for each gate I have a node here right. So, I have node for V 1 this is a graph circuit representation graph v 3. So, I am going to have a node for each gate of your circuit right and this registers will become the edge weight ok. So, this registers since I have a register here and this will become 1 w that is the weight of this edge is 1. Similarly there are 2 registers here you can I just mention there are 2 registers between these 2. So, the edge weight of this node will become 2 w right. So, this is what the way we are going to represent the circuit as a graph, I will repeat again. So, will have a node for each gate, I have a node set of vertices set of edges and the weight and d and w I will come into that.

So, for each gate of the circuit, I have a node in the graph and edge is basically the connections between the connection between the gates and this registers become the edge weight and that is the w right. So, for every edge there is a edge weight and if there is no register in between there weight become 0. So, suppose I have another gate here say v 4 to this there is no register here. So, v 4 to and the weight of this will become 0 right. So, I will put weight associated to each edges right and weight is nothing, but the number of register present between these 2 gate and also I have dv delay of the gate right.

So, all gate has some delay associated with that and that will has to be the input of your circuit also, and we have some usually we have a statistical data for a particular target architecture say for a pga say (Refer Time: 15:40) vertex 4 or say vertex 6 or all tera statics 5 or statics 10 or say asic target. So, for every cell are very for every gate, we have some estimation of a delay for each gates right and that value will be associated with a each node right.

So, you have say delay of this node that is a 2, this has a 1, this is 5 this is 10. So, that delay dv for each node will be associated to each node right. So, that will be given by the d which is nothing, but the delay of that particular node combinational delay of that particular node right. So, this is how I going to represent a circuit using a graph right just to elaborate there is an example here.

(Refer Slide Time: 16:21)



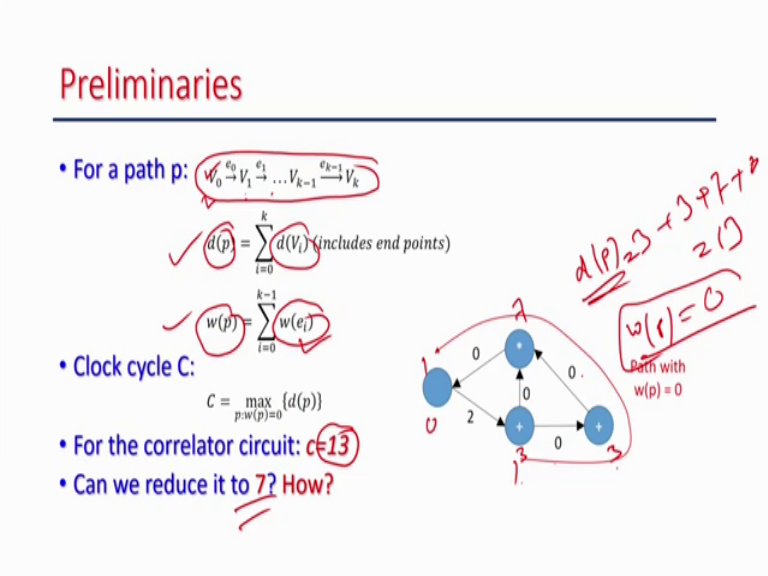
So, suppose this is my circuit, I have other multiplier those in the circuit and these raids are the 2 registers right. So, if I represent into the graph, I have for each edge there is a edge here. So, I have nodes for each gates or say combinational unit and there are 2 register here. So, it will be 2 all other there is no registers in this edges. So, their weight is 0 right.

And we assume that delay of the combinational unit say 3 and multiplication is 7. So, this is 3 dv d of this node is 3 this is 3 this is 7 right. So, this is that graph represent and so, suppose this is a means say input and so I will have a node of input and which is the combinational delay 0 because this is just a starting point of your circuit. So, this is how we are going to represent our circuit digital circuit using a graph ok. So, this is I think clear and we since we do not have a combinational loop or say cycle in a digital design.

So, it should not have any cycle in the graph, which is which has weight 0; that means, you should not have any loop for example, this is a loop right this is one loop, but this loop should not edge weight total edge weight of the particular loop should not be 0. That means, there is no register in that loop because combinational loop is or the combination of cycle is not allowed in a digital circuit, then the v value become inconsistent. So, there is no combinational loop in the circuit. So, this is just you can assume that this is a specification or the requirement for a digital circuit so that that should be there in the graph.

So, this is how I am going to represent the circuit using a graph ok.

(Refer Slide Time: 18:19)



Now we are going to talk about this path. So, path is nothing, but is just transition right of sequence of transition. So, this is a path or say this is a path or from this node this maybe a path right. So, path is sequence of transition. So, this is what it is a; so V e V 1, V 2, V to V k. So, there may be a path of this may be a path from V 0 there is weights to V 1, V 1 there is weights to V 2 and so on final through V k right. So, these are the edges.

So, now if you take a path we can calculate the summation of the delay of that path right. So, this is the summation of the nodes delay of this nodes right. So, for example; so this is a summation of the delay of the nodes this path similarly we can find out the weight. Weight is the weight of the edges right. So, e 1to ek minus 1 e 0 to ek minus 1 all the weight of the edges right so that will be the summation of the weight of that path.

So, whenever you consider a path in a in a graph, we can which represent a digital circuit we can think of the delay of that path which is nothing, but the summation of the delay of the all the nodes of that path and weights w of that path is nothing, but the weight of the edges right, which represent the number of registers in the path right. So, this is the 2 parameter we are going to define for a path right for example, if you take a path from this node to this node what is the total weight. So, as I mentioned addition is 3 and multiplication is 7 combination delay and the host node is 0.

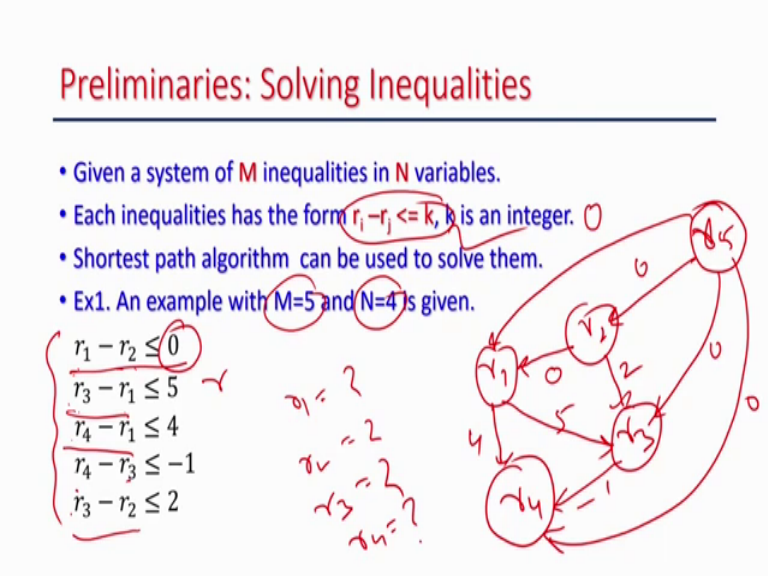
So, for this path total delay would be 13, 3 plus 3 plus 7. So, 3 plus 3 plus 7 plus 0 this is 13. So, this is dp. Dp of this path is 13 because this is 3 plus 3 plus 7 plus 0 right. So, dp is 13 and wp of this path is what? Wp is 0 because there is no register here, there is no register here, there is no register here. So, the wp is 0 for this path ok. If you take another path say from this to this what is the dp? Dp is 0 plus 3 plus 3 right. So, this is 6 and wp is 2 because I have register here 0 2 plus 0. So, this is 2.

So, this is how we can find out the dp and wp for a given path ok. So, for example, the example that is given here is that this is that biggest combination. So, if a if a wp is 0 for a path; that means, this is a combinational path right there is no register in that path and you can see in this particular circuit the maximum length combinational path is 13, because this is the maximum possible combinational path which is the combinational path there is no register incorporate.

So, this is the maximum length of a path, which has wp 0 so; that means, there is no register here. So, that is the. So, and the delay of that particular path is 13 ok. So, this will determine the clock period because this is the maximum possible combinational path and whatever the 13 unit your clock period should be at least 13 so, that this operations can be executed right.

Now, the question is here can we reduce this combinational dp of the maximum dp of that particular circuit to 7 that is we are going to answer and can we do it automatically. So, that is the 2 component here. So, this we have understood this for this particular graph or that this circuit, the maximum length of the delay of a combinational path is 13 and can we make it 7 by retiming or moving the registers, that is something we are going to discuss or can we how we can do this things automatically that we are going to discuss. So, this something the problem statement for now, and will take this example throughout this discussion to see how whole things works the retiming automation part.

(Refer Slide Time: 22:30)



So, before moving to that, the automation part of the retiming we will talk about the how to solve set of inequalities inequalities how we can solve that particular set of inequalities that will discuss, because that will be going to use in later ok.

So, that inequalities if you are look at in the form of this right that it is always like this r 1 minus r 2 less than equal to 0 r 3 minus r 1 less than equal to 5, if you have this form where k is an integer. So, all are integers and say suppose we have set of inequalities is M one 2 3 4 5. So, I have M equal to 5 here and number of variables here r 1 r 2 r 3 and r 4; so N equal to 4. So, if you have you have even a set of such inequalities right and you have to find out the value of this r 1, r 2, r 3 and r 4 which satisfy all this constant right.

So, this is your problem statement that if you have given a set of inequalities over n variables M inequalities over M variables and we have to find out this value of this variable which satisfy all this constant. So, that is how we can solve that problem and we can will discuss how for this particular problem can map to shortest path finding of graph right we try to map this particular problem to shortest path problem of graph and we can solve this particular problem using finding all via shortest path be in a graph or a I mean or single shortest path algorithm to solve this inequalities. So, that is something we going to discuss right.

So, we understood that we have set of inequalities number of variable is N and we have to solve this inequalities; that means, we have to find out the value of the variables which

satisfy all this registers all this constant. So, for that what we are going to do we construct a graph out of it how do we can do? For each variable I will consider a node ok. So, I have 4 4 variables here. So, I am going to consider r 1, r 2, r 3 and r 4 I will consider a node for each variables and I will consider additional node r extra what if there are N plus 1. So, this is 5 for me.

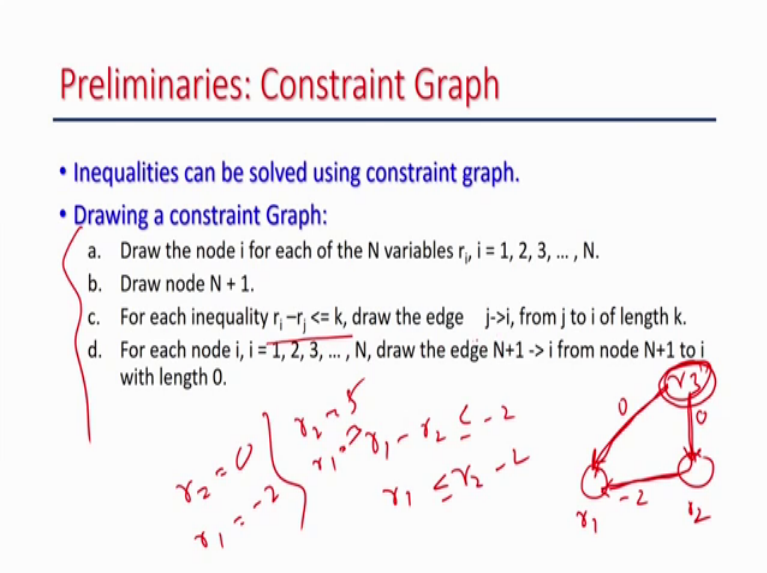
So, this is an additional constant then what I am going to do? I am going to add edges here. So, from this extra node I am going to add a edge to all the node of weight 0 what I am going to do? This is all weight 0 and for each a inequalities like this from r this r j to r 1 r. So, say r 2 to r 1 I am going to add a edge from r 2 to r 1 from this to this ok. So, r 2 to r 1 with weight this ok.

So, for example, this 3 to 1 to 3, 1 to 3 I am going to add a edge of 5, then this is 1 to 4 1 to 4 I am going to add a edge of weight 4 from 3 to 4, 3 to 4 I am going to add a edge of minus 1 and from this 2 to 3, 2 to 3 I am going to add a edge of 2. So, this is that graph representations and now we can show that, if we thus the value of r 1 r 2 sorry this r 2 r 3 and r 4 is nothing, but the shortest path from this node to that particular node.

So, if you just see here. So, the shortest path from this node is 0 right because. So, there are many paths from here I can go through this or I can go through this or I can go through some other path also I have 2 paths for example, for this node I have one path through this I have one path through this right or I have one path through this also.

So, for this path weight is 2 0 plus to 2, for this path weight is 0 and for this path 0 plus 0 plus 5 5. So, shortest path is 0. So, there are 3 possible paths and shortest path is 0 right. So, we can show that if we construct the graph like this from these inequalities that the shortest path from this node is nothing, but the solution of this right and intuitively what is the fact.

(Refer Slide Time: 27:00)



So, suppose I will just take a example r 1 minus r 2 less than or equal to minus 2 ok; so in this case since I have only this constant right. So, this is r 1 and I have to find the value of r 1 and r 2 which satisfy this constant right. So, I have r 3 extra node and this is 0 this is 0 and this is r 2 to r 1 is minus 2.

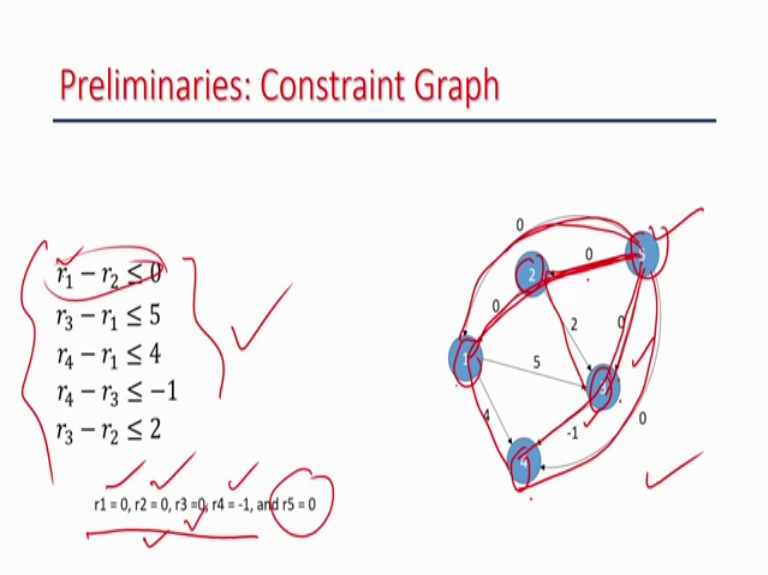
. So, now, think for this node I am only one path. So, r 2 is 0 shortest path is 0 because this is 0 and for this node I have 2 paths. So, this distance is 0 and this distance is minus 2. So, the minimum path is this right this one. So, r 1 is minus 2. So, you can see here that r 1 less than equal to r 2 minus 2. So, whatever the value of r 2, r 1 will be less at least 2 less of that.

So, this is satisfying this right we can have many value solution it can be say this will satisfy say r r 2 equal to say 5 and r 1 equal to 3 that is also satisfy this constant, but we need one solution that is satisfying this condition right. So,. So, you can understand the shortest path actually give you. So, if this is 2, shortest path you can find out from this extra node to this particular node that will give you the solution of this at least one solution because we can have multiple solutions, but we need one solution to this of this inequalities lesser inequalities right.

So, this is how we are going to solve these setup inequalities. So, what we can do? We are going to construct a graph as a constant graph from that, as a show here from the set of constant graph this is my constant graph which I have already drawn. So, I have set of

nodes for this is variable is variable and we have a extra node given from this I have all for to all this node weight 0 node is there and whenever there is a edge from constant like r i to r j I am going to add edge from j to i with weight k ok.

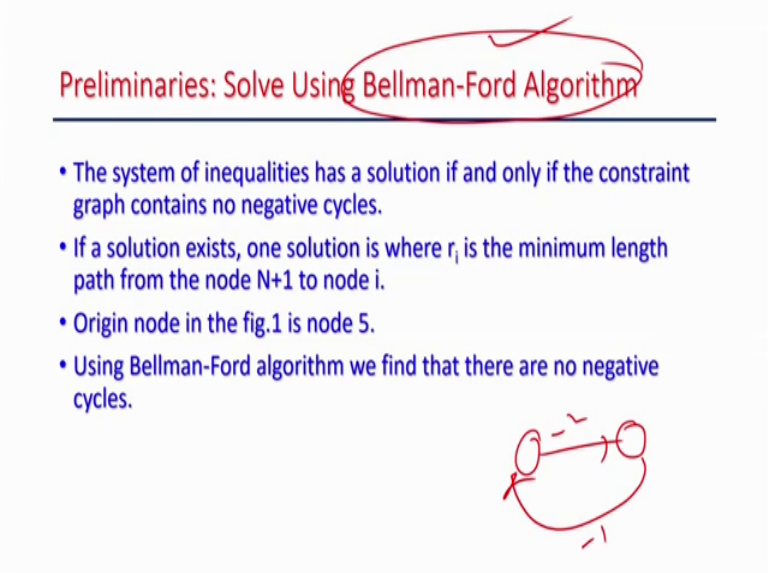
(Refer Slide Time: 29:03)



This is what I have done and discuss and then the solution of this particular inequalities nothing, but the shortest path from this node to all the node. So, if you solve this path we can see that r 1 is 0, this is r 1 r 1 is 0 because I have 2 paths here I just showed one path here one path here both are 0. So, r 1 is 0, for r 2 this is also 0 because there is only path possible. So, r 2 is 0 for r 3 I have seen there are 2 paths, but this is minimum. So, r r 3 also 0, and for r 4 I have this path weight is 0 and this path weight is minus 1 and this path weight is 4. So, this is the minimum one. So, r 4 is minus 1 and r 5 is 0 which does not matter to us. So,. So, this is the solution, and we can see that if you just put this value to here it will satisfy all the constant right.

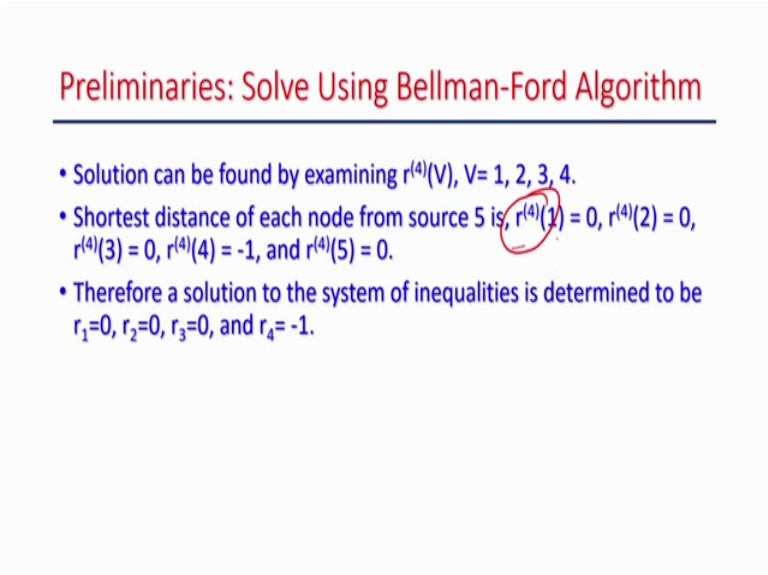
So, this is how we are going to solve a setup inequalities of the form of this. So, this is what we have shown here.

(Refer Slide Time: 30:00)



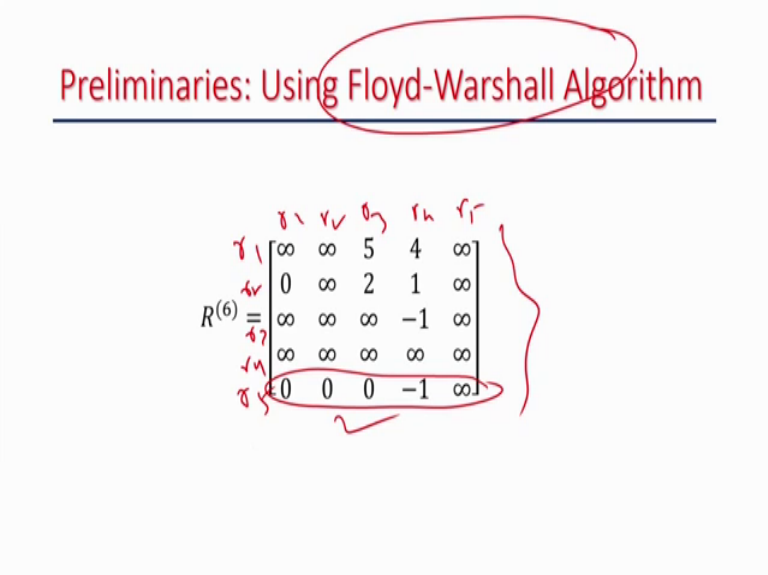
So, we can use bellman ford algorithm single source shortest path problem, which a given a graph we try to. So, we will consider this is the source and you are try to find the shortest path. So, how the bellman ford algorithm works? So, it works iteratively. So, it in first iteration it try to find out the shortest path from this node to some node of length 1 then length 2 and length 4 length 3 and so on finally, length of n. So, it will need n number of iteration. So, finally, I am going to after n iterations because I have n variable here I will get the shortest path ok.

(Refer Slide Time: 30:39)



So, if I just do this. So, I will show this r 4 1. So, this will give you 0. So, the same solution I am going to get ok. So, I am not going to decide of this bellman ford algorithm which we can learn from other sources.

(Refer Slide Time: 30:48)



Similarly, we can apply Floyd Warshall algorithm. So, this is a all pair shortest path it actually calculate using some matrix; so this r 0 r 1. So, if this is r 0; that means, all shortest path from node vi to vj through node r 0 V 0 and then this r 1 means all shortest path using V 0 and V 1. So, if I go for say 5 iteration say using all the nodes right.

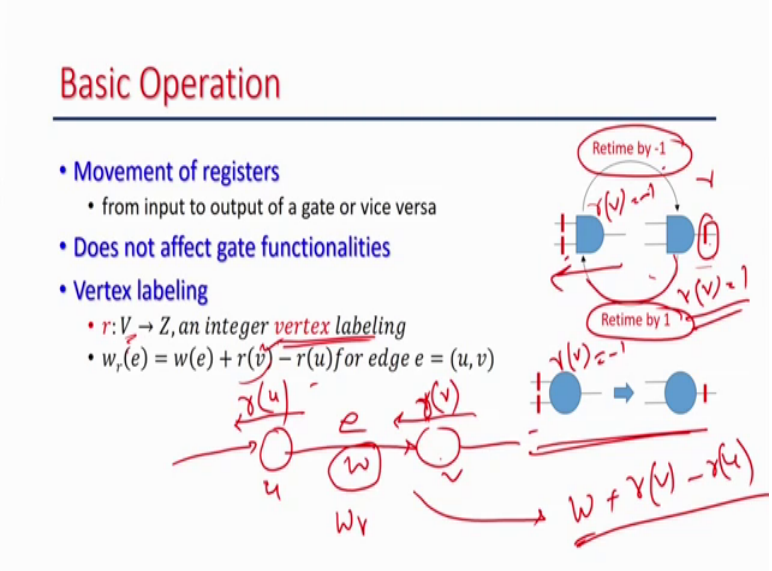
So, since I have 5 nodes here if I go for 6 iteration, it will consider all the nodes from 0 to 5. So, all this r 0 to r 5 will come into picture and then this is something. So, this is r 5 right this is r 1, this is r 1 this is r 2, this is r 3, this is r 4 similarly this is r 1, r 2, r 3, r 4 and this is r 5. So, r 5 is my that source node right. So, from that node; so the last row will give you the solution. So, r 5 to r 1 this is the shortest path length from r 5, r 2 this is the shortest path length from r 5 to r 3 this is the shortest path length and from r 5 to r 4 this is the shortest path length and then r 5 I do not care.

So, this is the solution which we have all also show in the last diagram. So, once we construct this constant graph from the inequalities either we can apply bellman ford algorithm or Floydwarshall algorithm to find out the shortest path ok, but provided there should not be any negative cycle and this bellman ford algorithm actually detect that if I have a cycle negative word cycle. So, for example, suppose I have cycle like this. So,

this is minus 1, this is minus 2 this cycle is minus 3 right if you have a negative cycle. So, there is no shortest path. So, if you rotate one time this minus 3 2 times then minus 6 minus 9 and so on.

So, we cannot find a shortest path if you have a negative cycle. So, using bellman ford algorithm also we can find out whether there is this there is a negative word cycle in the graph or not if there is no then we can solve the problems. So, this is how we can solve the inequalities and we are going to use this particular thing in retiming will come into discussion on that ok.

(Refer Slide Time: 33:00)



So now, we will talk about how to automate this retiming process, ok.

So, you can see that retiming what is happening? We are moving the registers right and how can define that how many registers are moved. So, for that what we are going to do suppose I have a edge like this and say the w is the weight here. So, this is my edge e, this is the node u, this is the v and so, initially their number of registers is w. So, how many registers will be there after the retiming? If some registers move from here. So, it is depends on the number of register move from this side to this side and number of move registers move from this side to this side ok.

So, let us say if the number of registers move from this to this is r v and the number of registers move from this side to this side from for from the output to input of the node u

is r u ok. So, this r is for a node is given by this value like if the number of register move from the output side to input side of that node v right. So, r v means number of registers move from the output side to input side of the r v. So, that the final value of this register w r after retiming what will happen? W plus r v because that is coming and that is going out right r u.

So, the after retiming the number of registers in this particular edge will be given by w e that is this the weight of the edge actual weight then the r v minus r u right. So, number registers coming number of registers go out plus the original registers. So, that will give you the number of registers after retiming ok. So, what it indicates? If we can find out this r value for each node then we can calculate the retiming value right because we are now going to talk about the automation how we can do that.

So, what we are going to talk about? We are going to consider a variable r v for each node v of the registers, which define the number of register that is move from output side to input side of that particular node ok. And if we know the value of that all this r value r value of all nodes, then I can recalculate the recalculate the edge weight after retiming, so that that will define you retiming right.

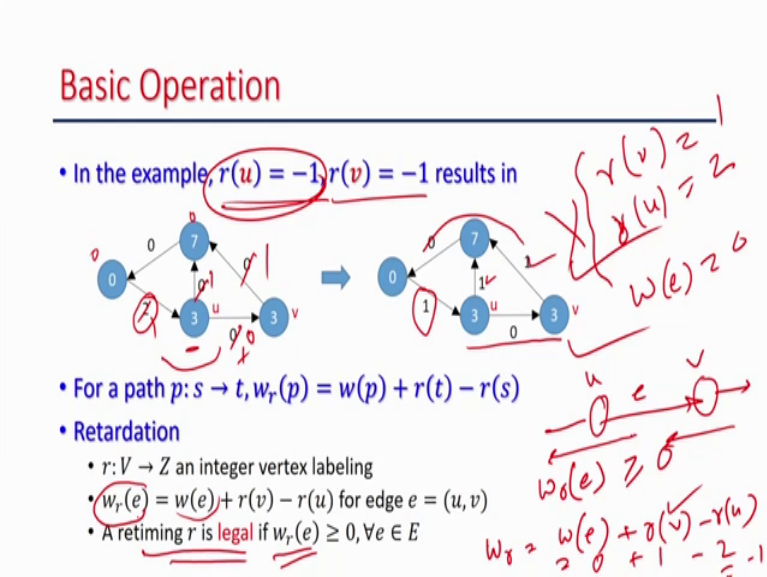
So, this is something what I discuss here. So, retiming is something solving this solving this problem, how to find out this r value for each node right. So, what we are going to do that? We are associated a level vertex level for each node r and which will define the number of register, that is move from the outside side to the input side ok. So, that is r v. So, you have. So, if you have number of nodes in n. So, how many variables are you have? N number of variable r 1, r 2, r 3 to r n right. So, those number of variables are there and if I know the value of all these variable we can find out the number of register in each edge after retiming ok.

So, conceptually this is shown here. So, if there are 2 register at the input and if you move to this register at the output right so; that means, the number of register move from this side to this is minus 1. So, retime by minus 1 right because they are moving the register from in input side to output side. So, the number of register move from output side to input side is minus 1. Similarly if you have register the output and you move them to input then be the number of register move from here is 1. So, retime by 1 right. So, this is how we define the r value. So, for this node r value is if you have move if this

happens. So, this r v value will become minus 1 if this happens from this to this then the r v value will become this is minus 1 this is one because that is what is discuss that is with given the number of register that is move from the input side to the output side ok

So, this is something. So, if you have this if you know this r r v equal to minus 1 then I know there are set of register from input side move to output, if the r v value is 1; that means, I know that the one register from the output is moved to the input right. So, this is what is given by this. So, for this case this is r v equal to minus 1 because number of register move from input to output is 1 ok. So, this is how we can determine the weight of the edges after retiming and we are going to find out these values r values. So, our objective is to find out the r value, the level value register value this r value of each node which with which we can actually identify the retime circuit ok.

(Refer Slide Time: 37:43)



So, here again the one example for that to clarify your things; so suppose this is my u node and this is v node and say r u equal to minus 1 and r v equal to also minus 1 what does it mean? So, if it is minus 1 means what? Number of register move. So, this s r means number of register move from output to inputs, this is minus 1 means effectively 1 register move from the input side to output. So, what is happening? So, one register move from to this. So, this will become 1 and this will become 1, this will become one because that is what given by this r u minus 1 right and then again this r v also says that number of register move from output to input is minus 1, which is means number of

register move from the input side to output side is 1. So, this is move to this. So, this become 0 and this become 1.

. So, this is the retime circuit see if I know the value of r u and r v, I can find out that I actually move this one register from here to this place and this place this is what is given by this value right. So, this is what if I know this value of r u and r v is this, and say this is 0 other are 0. So, this node is also 0 this node is also 0 then I know the after retiming the circuit is like. So, what does what is mean basically? There are 2 register initially here, I move one of the register to this 2 places and I keep one register at the original place. So, this is what is retiming does right and then you can see here the maximum delay will become now 6 right. So, what this node this is 6 and this is also 7. So, maximum delay will become 7.

So, understand that ok. So, this is how we can will do? So, we are we consider a variable r for each register and we tried to find out some set of inequalities now or constant which will satisfy by this r that we are going to discuss and then we try to solve those particular set of inequalities using that shortest path problem, which will give you the value this is the overall approach.

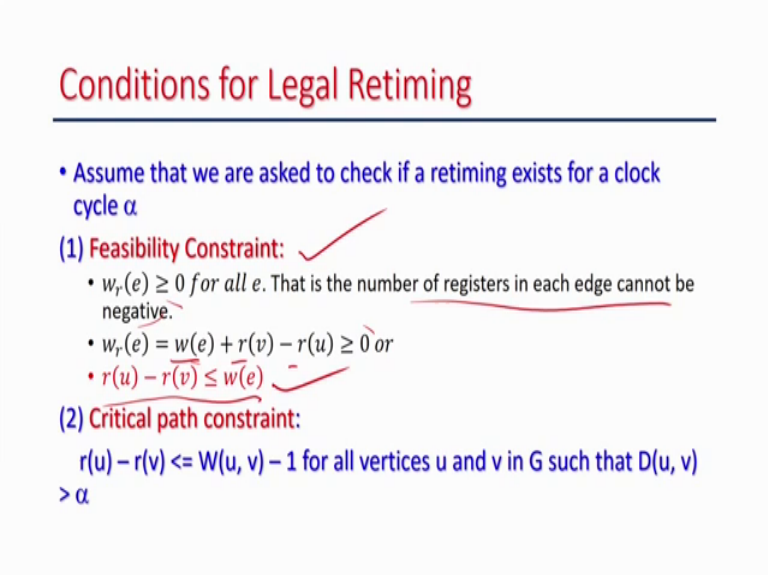
So, what is the set of constant now? The first constant is that, if you have a if you have a path here weight of that path after retiming cannot be negative right. You cannot have a negative weight because your register cannot be negative maximum can be 0 it cannot be minus 1 or minus 2 right. So, the first thing is that, if a retiming is legal; that means, your this w r is given by the weight of the register after retiming. So, w is the weight of the register before retiming and w r of edge e given is given is represent the value of weight of the particular edge after retiming right.

So, what does it say that w r? So, this is the edge e. So, w r of e cannot should be 0 should be greater than 0 because your weight should not be less than 0. So, this is what is the first constant ok. So, for every edge weight should not become negative after retiming because if you put say. So, suppose this w of the edge is say 0 and you. So, this is say u to v and (Refer Time: 41:08) suppose you give the r v is one and ru is 2 ok. So, then what is happening see you are saying that number of register come here is one and number of register come from goes from this register is 2. So, finally, the final value of this register is given by this I know that I already discuss the w plus r v minus r u. So,

that is w w r is nothing, but w of e r v minus ru right. So, now, this will become w is 0 I consider is 0 this is 1 and this is minus 2 this becomes minus 1 .

That we are assuming that after retiming this edge has minus 1 number of weight. So, which is not feasible. So, if you solve these inequalities using this which is wrong right. So, we this is not a valid solution. So, we should have some constant on this variable because any kind of value is not a solution because some of the solution is not illegal some of the solution is not correct or illegal for a circuit digital circuit. So, we have to add set of constant, which is called feasibility constant will the feasibility constant that number of register for each edge cannot be negative right.

(Refer Slide Time: 42:06)



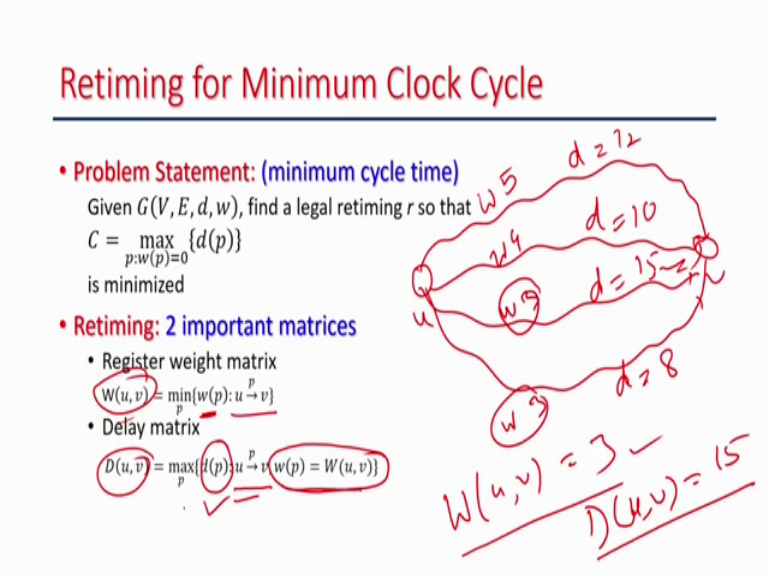
So, each it should be after retiming it should be greater than 0, which is w r greater than equal to 0 which is nothing, but w e plus r v minus r u greater than 0. See if you just re write this it will become formula this r u minus you just re write this I mean put this w this side and you just do this. So, you will get this equation R u minus r v greater than less than equal to w e.

So, this is for each edge which is called feasibility constant. So, you should not have any kind of value of the r. So, that your edge weight become not non I mean negative is. So, that we are restricting this value to this variable so, that we cannot have a solution which result in this ok. So, if you have a set of inequalities like this, which ensures that

whatever the solution you are going to get is it does not result in some edge which has negative edge right

So, that is that will be restricted by this constant ok. So, next is called critical path constant that I am going to come after will discuss ok.

(Refer Slide Time: 43:16)



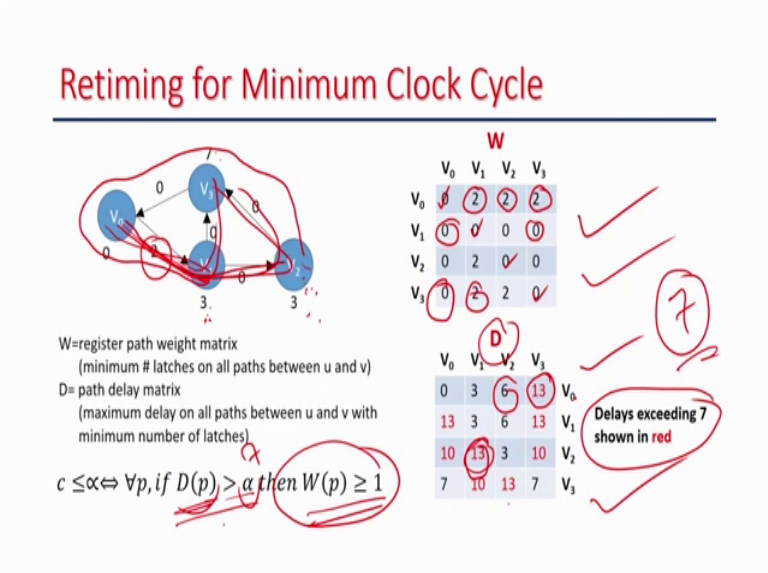
So, before that we are going to discuss two important parameter is this w and d this is used in the second constant. So, let us discuss that what is that. So, if you just consider a path from u to v. So, you can have multiple paths here right you can have multiple paths and say there are 4 paths here and say that weight of this node is say 4, this is 4, this is 3 and this is 3 or say this is 5 the number of registers weight means number of registers in this path. So, number of. So, from a node to another node there may be multiple path not this one path right.

So, I assume that from this node to this node there are 4 paths and the number of weight here is 5, weight is 4 where, weight is 3 here, weight is 3 here and now. So, that is given by the w p ok. So, this is something W u v means for all path from u to v which is has the minimum weight. So, for example, here; so for this particular example W u to v will be 3 because I have 4 paths and I assume the number of register is 5, number of register is 4 here, number of register is 3 here, number of register is 3 here. So, there are 2 paths which has number of register is 3. So, W u to v is 3 ok.

So, W u v. So, for each pair of u v I can find out this value ok. Similarly that d u v says what? So, for all the paths from this u to v for all paths from u to v, if the weight is minimum among the paths what is the maximum delay ok. So, suppose the d of this node is a 12 this is a 10 this is a 15also this is a 5. So,. So, although. So, this is this is 50 ok. So, although these particular paths there are 4 paths, and the minimum delay maximum sorry suppose this value I consider as 8 ok. So, I am going to consider out the maximum delay which is basically this 15, but I am going to consider only the path which has minimum number of registers ok.

So, I am going to consider all the path which has minimum numbers. So, although there are 4 paths, I am going to consider only these 2 path because these 2 path have a minimum number of registers and then I am going to consider the path which has maximum delay value which is 50. So, D u to v will be 15 for this ok. So, this is the 2 parameters this is a matrix right for each pair of u and v, we have to calculate this that maximum sorry minimum number of register between any 2 node and among the parts where we have minimum number of register what is the maximum delay. So, this 2 value. So, example I have taken here w u will be 3 and delay u will be 15 this is how this 2 parameter says ok.

(Refer Slide Time: 46:46)



So, if you just take our same example that example we have considered so for. So, you can see that the maximum. So, if you just consider. So, this is the it will be given by a

matrix, because any pair of paths from any v to v ok. So, v. So, all this V 0 0, V 1 1, V 2 2 will be 0 because this is the number of register between 2 node is 0. So, this all this will be 0 ok.

So, this is given by the w parameter and if you consider any 2 pair V 0 2 say V 3 how many paths are there? This is one path, this is one path and this is another path there are only 2 path. So, this path has 2 register this path also have 2 register because there are two. So, this is 2 ok. So, V 0 to v 2, V 3 is 2 V 0 to V 1 also there is one path weight is 2. So, this is 2. So, V 0 to V 3 also there are V 0 to V 2 there is one path and weight is 2. So, all are 2.

So, now from V 1 to V 3 there are 2 paths both are 0. So, this is 0, V 1 to V 3 there is one path weight is 0. So, this is 0. So, this is how you can represent this w which is giving the minimum number of weight between to any 2 pair. So, for example, say V 3 to V 0 there is only path weight is 0 right. So, V 3 to v 1, V 3 to V 1 I have one path and the weight is two. So, this is 2. So, this way we can actually construct this matrix where any pair v i to V j that will give you the minimum number of register possible in all path among this 2 nodes minimum number registers in that all the paths where we have the number of registers minimum ok. So, this is given by this w ok.

Similarly, now we are going to consider all the path where the weight is given by w u v what is the maximum delay? That is given by this right. So, if you just take again that example this example. So, for examples again this V 0 to say V 2 let us calculate. So, this is weight is 0 this is 3 and this is 3. So, I have 6. So, and I have only one path right. So, V 0 to V 2, V 0 to V 2 6; V 0 to V 3 how much? There are 2 paths this is the path, this is the both the path has minimum number is 2. So, and this path has 3 plus 7 is 10, this path 3 plus 3 this path 3 plus 3 plus 7 which is 13.

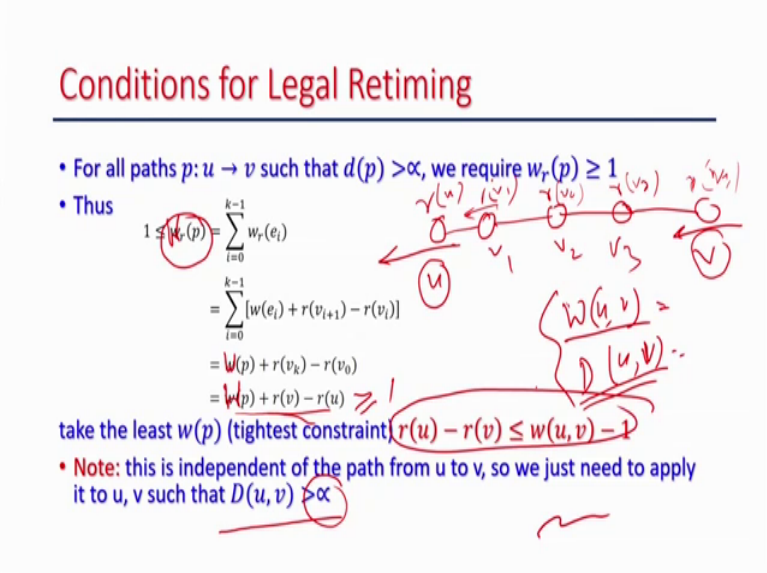
So, I am going to consider the maximum because both the path weight is 2 2. So, this is 13 and you. So, similarly you just takes a V 2 to V 1 V 2 to v 1. So, this is your V 2 the path is this right this is the path and this is 3, this is 7, this is 0 and this is 3. So, 3 plus 7 10 plus 13. So, this is 13. So, this is how we can calculate the D. So, this is suppose this 2 matrix we have to calculate for this which is represent the minimum number of register between 2 any 2 nodes and among them what is the maximum delay ok. So, this is what we got here ok.

So, this is that 2 matrix and now you can see the nodes with red. So, we for this example as mentioned earlier that maximum combinational path is 13, you can see their maximum delay of the combinational path is 13 which we have seen earlier also and our target is to reach 7 right can we reach 7 for this. So, that is what we what we have discussed earlier also that our target is to get 7 here and all the nodes that has red here that is those are the combinational which actually do not satisfy they have the delay more than 7 right. So, for those kind of.

So, they are actually critical path right because our target clock here it is 7 and we are getting here it is 13, 10 those values right. So, those are the critical path because they are not satisfying our clock target clock here and for all such path where w p is. So, all this path where this d p the delay of the p is greater than the target clock peer, which is 7 in our case say this is alpha then those paths should have at least one register after the retiming otherwise it will not break the critical path right. So, we have to consider all the path. So, there are 1 2 3 4 5 6 7 8 there are 8 paths here those path are actually failing or failed to satisfy the critical target clock peer which is 7 in our case and we need at least one register on those path to break that critical path. So, this is given by the critical path constants.

So, for all the paths where the d p, the d of that path is greater than the target clock peer on that path the d W p should be at least greater than, right. So, this is what is given by the critical path constant. So, this is what is given by this. So, how we will get this we will discuss. So, after if you just think about say path here which is very important.

(Refer Slide Time: 51:55)



So, I just consider a path here from u to v and we may have some view on some other node also right. So, this is that node path where w w u v is minimum. So, this is the path there may be minimum many paths now I consider a path where this is minimum w u v is minimum and d p is maximum. So, this is given by one path. So, this is that path ok.

So, now in this path what is happening? So, I know I every node associate with r right. So, this is r u associated here this is r v 1 r v 2 here r v 3 here and so on right r v 4 sorry r v right. So, what we are going to consider is that is the final value of this r v. So, if I know this means the number register move here from this to this right. So, here this means the number register move from the output to input here, this r V 2 means number register move from output to input here right and so on. So, in this case, what is the final value of this w r p.

So, w p is given before the retiming and w r p is given the value which is basically nothing, but after the retiming. So, if we calculate that. So, it is basically the w r of each of this edges right. So, w r of every edges after retiming which is given by w r e i right and now this I can represent by this we already know that this for w r is nothing, but w r r i is where before retiming and this r v I plus 1 to r v.

Because the edge e from v i to v i plus 1 ok. So, this is the difference between this. So, if you just calculate this you can see that I can actually cancel out all the intermediate terms and only this r v k and v 0 comes so; that means, v 0 is u and v k is v. So, finally, r v

value of intermediate node does not matter to us only the r v value of this u and v matter to me right. So, w p plus r v and r minus r u right. So, this is r v this is r u.

So, what is what is the important factor here is that, the after retiming I I should only bother about this r u value and r v value and this will be given by and the w p the weight the minimum weight will be given by w p whatever the weight was earlier plus r v minus r u intermediate thing will be cancelled out each other.

So, now this value should be at least greater than equal to 1 right. So, this value should be greater than equal to 1 ok. So, what we have seen that is for a path which has minimum weight w and d v is w v and maximum weight is d v there we can see that the intermediate r value does not important only important factor is the start node and the end node right.

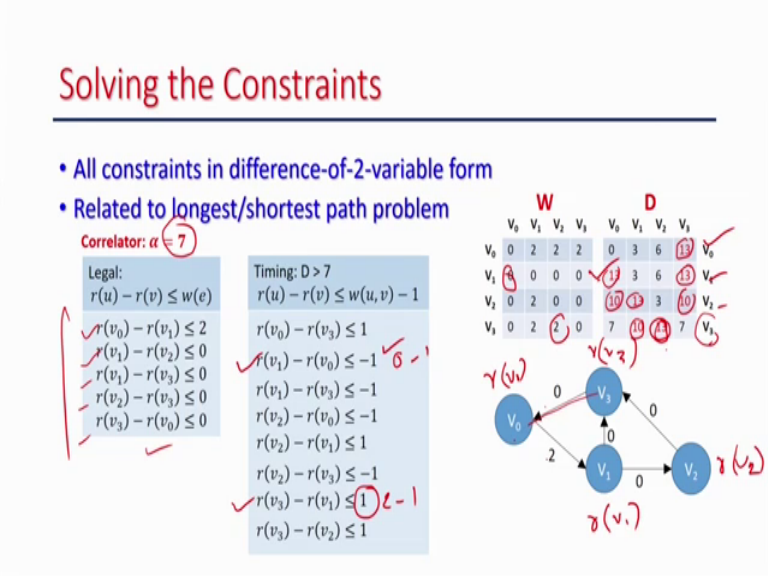
So, the final value of w r p will be given by this is actually w this actual w u v plus the number of nodes come into here and number of register goes from here right. So, this r v minus 1 and what I am saying that since this is e prior d v that D u v is greater than target clock prior what will happen. We need at least one register here to meet the critical path right.

So, we have to see that this value should be greater than equal to 1 and if you just move this values I mean reshuffle this values you will get this right. So, this is what is called critical path constant. So, I have 2 set of constraint first set of constraint says that feasibility; that means, after retiming your age weight should not be non negative so; that means, you should have a at least 0 register between any 2 gates.

And critical clock path constant ensures that for all path where delay actually exceeds the target clock prior which is say alpha, where d p of that particular path is greater than alpha I ensure that that those path all possible path have at least one register so that you meet the critical path. So, if you have these 2 set of register set of constraint and then you try to solve this. So, you will get set of inequalities now right.

So, these are all basically represent by the inequalities of the form we have discuss early right.

(Refer Slide Time: 56:36)



So, here if we take the same example again; so for each age I have this feasibility constant. So, from V 0 to V 1 the form is like this; so r u minus r v less than equal to w v. So, I will have this r 0 r v 0 minus r v 1 should be less than equal to 2.

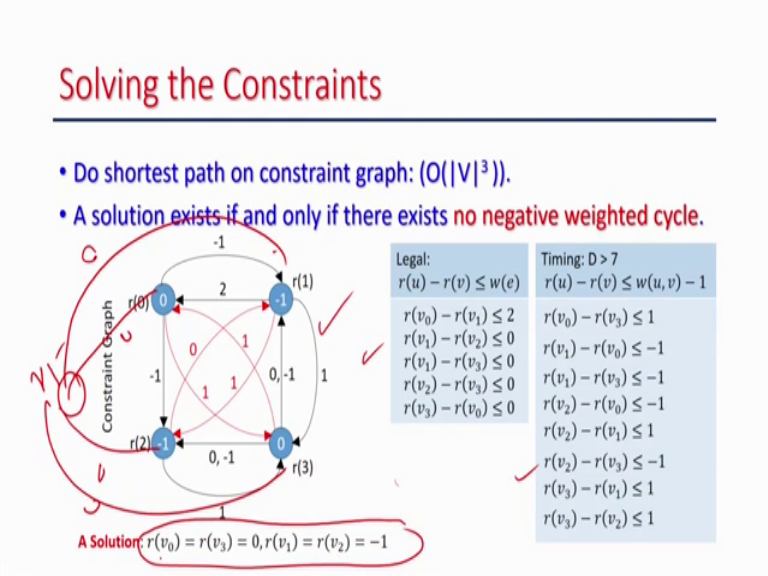
So, I have 1 2 3 4 5 ages. So, I have 5 constant like this 1 2 3 4 5; so for example, r v 3. So, age from V 3 to V 0 V 3 to V 0, I have r V 3 minus r V 1 V 0 should be less than. So, will be 0 you remember that I have assign r v 0 here and r v v 3 here r v 2 here and r v 1 here and our objective is to find the value of this r v 0, v 1, v 2 and v 3.

I am adding constraint the first set of constraint is called legal constraint or the feasibility constant and for each age I have to add this right. So, I have 5 ages I have added 5 constant like this and now I have consider all the path where delay is greater than target clock gate. Because our target clock gate is 7 in this example.

So, I have all this scenarios 3 4 5 6 7 and 8 those are the path which delay is more than 7 because they are more than 7 for those path I have to add a constraint right. So, if I just consider say these value this is V 1 to V 0. So, V 1 to V 0 I have to add a constraint like this r u minus r v less than equal to w u v minus 1 right. So, the w u v value will get by this parameter right. So, if you just consider say r u minus r v. So, if you just consider this say this value ok.

So, this is V 3 minus V 1. V 3 minus V 1 should be V 3 minus V 1. So, V 3 is V 3 V 3 minus V 1; that means, r v 3 minus (Refer Time: 58:39) should be w u v w v or V 3 V 1 is 2 minus 1. So, this will become one right 2 minus 1. So, similarly if I take another value say this one this is V 1 to V 0. So, V 1 to V 0 this is V 1 to V 0 and for this node r v 1 minus r 0 should be w u v is 0 here; so 0 minus 1. So, this is minus 1. So, this is how this constraint array. So, for all this value I am going to add a constraint like this ok. So, I have now set of inequalities. So, this is feasibility constraint and this is critical path constraint. So, now, I am going to solve this constraint if you solve this constraint I will get the value of r v 0, r v 1, v 2, v 3 right.

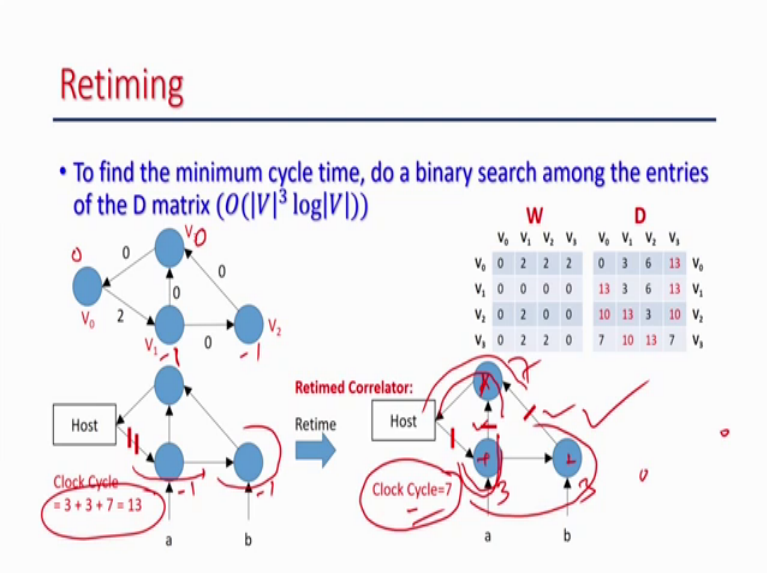
Refer Slide Time: 59:21)



So, this is and for that I have already discuss how to solve the set of inequalities using a shortest path problem. So, what we are going to do I construct a constraint graph as I discuss early from all this constraint I will get this kind of constraint graph and then I apply the shortest path algorithm like single source shortest path bellman ford algorithm or Floyd Warshall algorithm to get this values right. So, that will give you the shortest path value from this extra node. So, constraint graph there should be another node here which is this from where this 0 should be ended.

So, this is 0 0 0 and then this is r 5. So, this r 5 to this node will give you the minimum value right and solution to this problem is this. So, I am not going to detail how to solve this. So, I got the value that r v 0 is r v 0 r v 3 is 0 r v r v 0 is 0 r sorry.

(Refer Slide Time: 60:14)



So, this is say this is V 0 right. So, this is V 0 this is 0 r v 0 is 0 then r v 3 is 0 this is also 0, r v 1 is and r v 2 is minus 1. So, this is minus 1 this is minus 1. So, this is the value we got right and with these value you can actually recalculate the position of the register right.

So, this is my initial position and this is minus 1 right. So, this is minus 1 what is happening here so; that means, one register will move from this means number of register move from this is minus 1; that means, one register will move from this right. So, one will move from here. So, this will come here and here and then this is also minus 1, this is also minus 1 means one register move from output to input.

So, this will move from this to this. So, this will come here and this will be removed. So, you can see that for all other is 0. So, this is the final position of the registers and you can see the maximum delay of the critical path, this is I know this is if you remember this is my if I remember correctly this is multiply and this is adder and this is adder right. So, this is 3, this is 3 and this is 7 right.

So, the delay of this path is 3 plus 3, 6 delay of this path is 7 similarly delay of this path is. So, there is a register this is 3 and delay of this path is also 7. So, the maximum delay of this is 7 and earlier it was 30. So, you can see that using this retiming algorithm, I actually solved this problem right I find out the value of this variable such that this r r 0 r

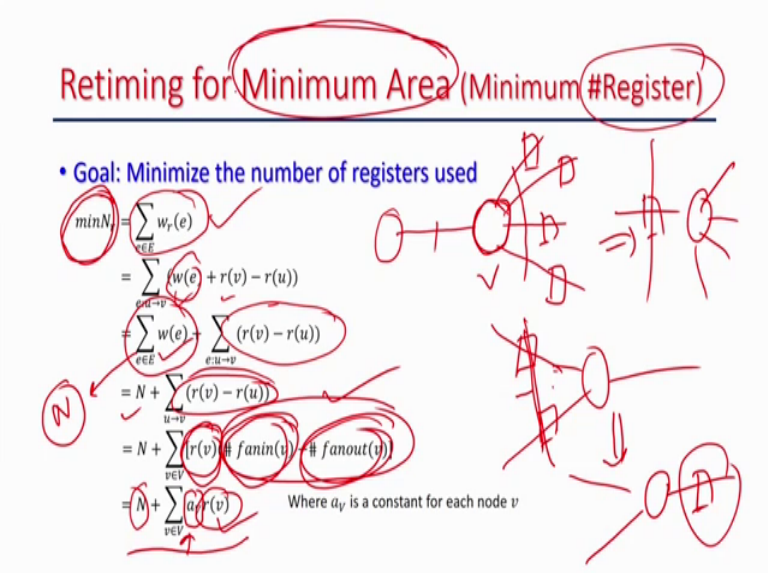
v0 r v 1 and r v 2 such that I actually reach the s r gate which is my target and I can reach the clock of clock period 7 right.

So, you understand the overall flow. So, what we are going to solve here we try to find out the position of the registers automatically what we have done for that I put a value variable I consider variable for each node, which is which define the number of register move from the output to input and we try to find out set of inequalities.

One set of inequalities for the feasibility, one set of inequalities for the critical path and we solve those inequalities using shortest path problem shortest path algorithm like bellman ford or Floyd Warshall algorithm and that will give you the value of those variables the r value and using the r value I can recalculate the register weight for each age and so, the which actually give you the retiming circuit. So, I can see that from this this from this circuit where I have 2 register here I actually using the retiming algorithm, I move one of the register to this places and through which I can achieve the target clock period of 7.

So, this is something is called retiming for clock prior optimization because I try to optimize the clock period. Similarly we can if you do retiming to minimum minimize the number of registers also because suppose if you just consider one example suppose I have a example of this suppose I have say I have 4 output register right.

(Refer Slide Time: 63:25)



So, if I have a register here and suppose it is in other way just a minute. So, I have say I have register here right.

So, number of register is 4 right and I if I do a retiming I just move this register to the input what will happen I have only one register right. So, the number of register reduce. So, this is the objective if you have number of registers at the fanout and if you just move them in the fan in then the number of register may reduce or it may be other way also.

Suppose there are 2 input and one output. So, initially there are 2 registers here and I can move the register at the output also right. So, then the number of register will become 1. So, this is the objective if you our objective is to move the register in the circuit. So, the number of register become 0. So, you remember here the latency remains same because that whatever the input is coming it will take one clock period to I mean one clock period is the output because one set of latency are there here also latency is one, but the number of register is less.

So, latency does not change for the circuit, but the number of register has change ok. So, for this. So, our objective of this kind of retiming is number minimizing the number of registers. So, what will be the objectives? So, suppose this is the total number of register, I want to minimize this number. So, what will be the given value? So, that w r is the modified age weight right. So, for each weight age this is the summation also that will give you the number of register after retiming.

Suppose you do the retiming in the same way and you get this value and which is nothing, but w e plus r v minus r u right. So, we have already seen that. So, this w e is already fix. So, this is the summation of the w e plus the summation of the r v minus r u. So, now this is already in.

So, suppose this is a fix number you cannot change this is the actual number of register in the circuit. So, this is n you cannot change that because in your initial circuit has this number of registers and now what we can do.

So, now this r v and r u associate to each node ok. So, now, you can think of this is basically now you can think of any node and where I can say that the how what is the number of change will happen, this number r v into this. So, this r v is for a node r u

right. So, this is say v node the fan in. So, if we have as I discuss here if the if difference between the fan in minus fan out right.

So, here fan in is 1 and the number of fanout is 4. So, if I move a register from this to this side it will reduce right. So, this is fanout minus fan in the number of register reduce 3 4 2 1 right. So, similarly here if I move a register from out input to output what will happen? This is 2 minus 1. So, sorry one minus 2 this is minus 1 right.

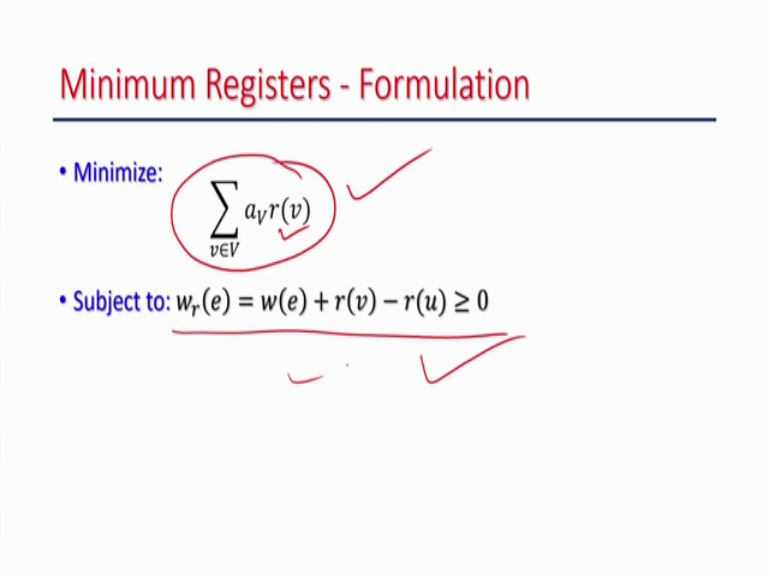
So, basically sorry fan on fan in minus fan out ok; so what is happening here? So, there are fan in is 2 and fan out is 1. So, if you move a register from output input side to output your number of age count will be reduce by 1 right. So, 2 minus 1 is 1. So, this will be given by the r v into number of fan in minus number of fan out. So, for a given node how many fan ins are there and how many fanouts are there their difference will give you the benefits.

So, if the fan in is less and fan out is more; that means, if you move a register from input to output you your actually number of register will increase and if your number of fanout is less and fanout is more fan in is more, then you basically if you move a register from fan in side to fanout side then a number of register will be less. So, this fan out fan in minus fan will give you fan out will give you the benefits for each node and this is a constant for each node.

Because you are not going to change the any circuit structure for each node the number of fan in and fan out is same. So, this is nothing, but a constant for a given node right. So, the objective is here depending at the number N is fix a V is also fix. So, I try to find out some r v such that this summation is minimum right. So, I try to find out the value of r v such a way that this r v value is minimum for this summation ok.

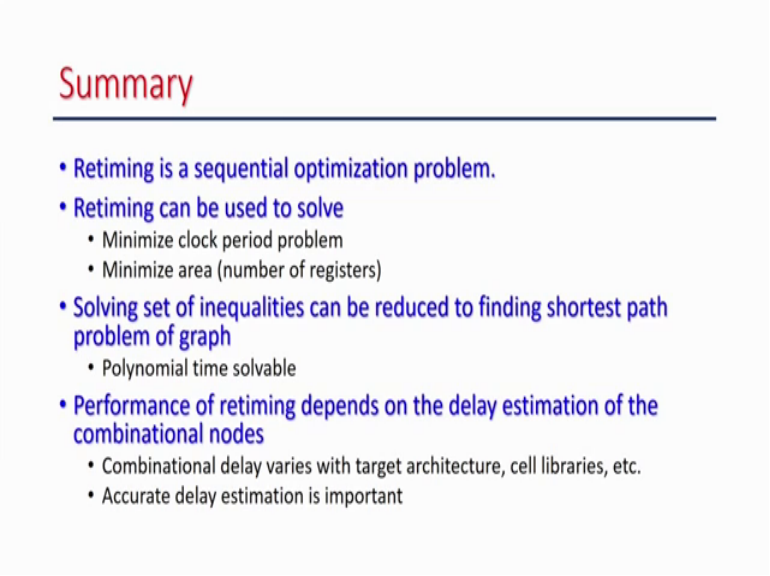
So, here the objective is has to minimize this value summation for each node r v value such that this will become minimum, where I have the feasibility constant right. So, the number of age weight should not less than this right. So, this is the minimum formulation of the minimum area problem. So, if you want to minimize the registers we have this is the objective function and this is the constant ok.

(Refer Slide Time: 68:02)



So, this way we can actually solve the minimum number of register problem as well.

(Refer Slide Time: 68:13)



So, as a summary we have discuss about the retiming, how it will improve this timing or the number of register of a circuit and we have seen that this particular retiming can be map to can be formulate as a set of constant feasibility constant and critical path problem and then we can solve those inequalities using shortest path problem to solve this retiming problem ok. And also we can solve both for minimizing clock period as well as minimizing the number of register and then we one factor is important here is that delay

value that delay of those node, because this is not the actual circuit because we are all doing in I mean this is a virtual kind of circuit right because it is in the physical chip.

So, all the delay value the performance of the retiming primarily depend on this delay value. How accurate is that delay combinational delay estimation of those nodes right. So, if you talk about any synthesis tool for any target architecture they have a file right. So, those file actually mention the delay of each node for that target architecture. So, now, the retiming performance actually depends on that delay value. If your delay is not accurate the retiming that will be done is may not be as per the expectation, but if you have a really very accurate delay estimation for your all the gates of yours design, then retiming actually works wonder right it will give you optimize circuit in terms of number of registers or clock period by automatically right.

So, 2 factor this is a polynomial type solvable problem because all are kind of n q algorithm shortest path problem. So, this is not a exponential or n p n p complete problem this is a polynomial type problem which can be solvable, but performance largely depends on that delay estimation of the nodes of your design ok.

Thank you.