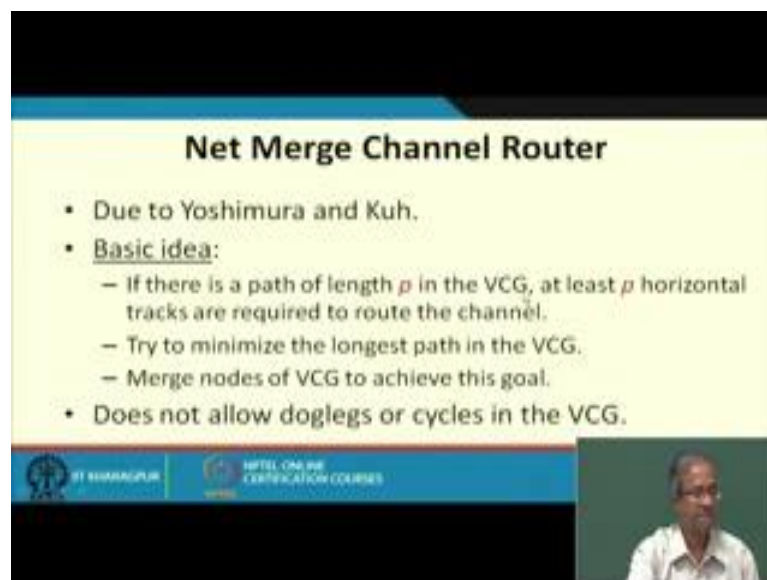


VLSI Physical Design
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 22
Detailed Routing (Part III)

So, in our last lecture if we recall we had talked about the basic left edge algorithm for channel routing and some of its variants, where we extended the algorithm move to handle the vertical constraint and also to allow for the doglegs.

(Refer Slide Time: 00:39)



Net Merge Channel Router

- Due to Yoshimura and Kuh.
- Basic idea:
 - If there is a path of length p in the VCG, at least p horizontal tracks are required to route the channel.
 - Try to minimize the longest path in the VCG.
 - Merge nodes of VCG to achieve this goal.
- Does not allow doglegs or cycles in the VCG.

The slide also features a video inset of Prof. Indranil Sengupta in the bottom right corner and logos for IIT Kharagpur and NPTEL Online Certification Courses at the bottom left.

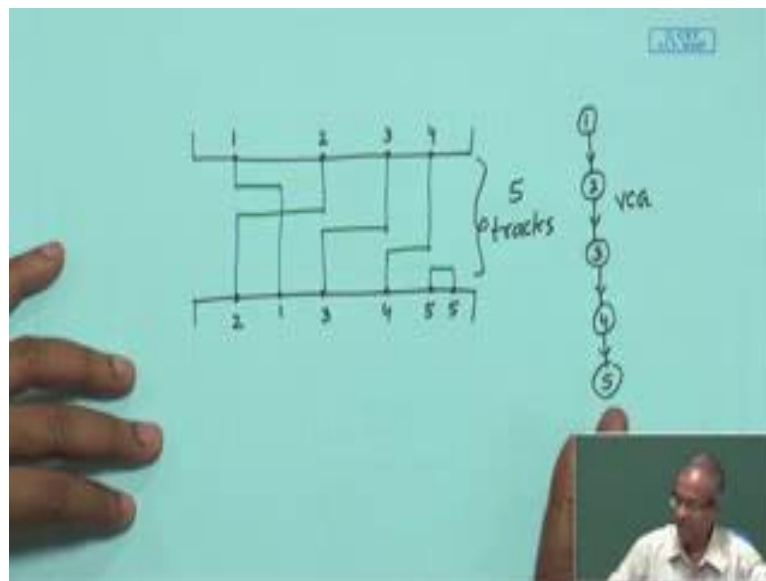
So, we continue our discussion and we shall first talk about something called net merge channel router. This algorithm was due to means Yoshimura and Kuh it is sometimes also called Yoshimura Kuh algorithm, you see net merge channel routing the idea is somewhat like this. You have multiple nets 1 2 3 4 that way you all one to route in a channel. Suppose you have 2 nets i and j , there is a process we shall be discussing that if you feel that i and j are in some sense compatible, in the sense that they can be put on the same track then you try to merge them. So, once you merge 1 and 3 let us say, you merge the 2 nodes or vertices 1 and 3 together to form a one super vertex.

Let us super vertex will indicate that 1 and 3 where ever you move they will all move together. In this way 2 or more nets you try to group them in the same track and move that entire cluster up and down, depending on the vertical constraint that exist between

them. So, this is the basis idea behind net merge channel router, where we shall see that we shall be handling something called zones, the relationships upon each other and also some graph based means processing once the zones are indentified and defined.

So, let us look at the basic idea the first point is important. In a vertical constraint graph if you see that there is a path of length p , then you will need at least p horizontal tracks to route the channel why it so? Let us take an example.

(Refer Slide Time: 02:40)



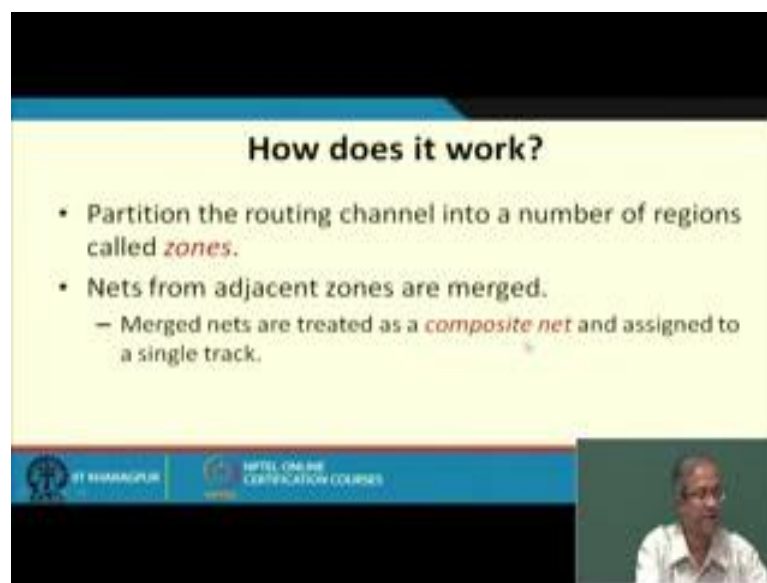
Suppose we have a vertical constraint between 1 and 2, say between 2 and 3, 3 and 4 and say 4 and 5. So, here though there is no cycle, but there is a long chain in the vertical constraint graph 1 to 2, 2 to 3 and 3 to 4.

Now, if you want to route this channel. So, your solution will look like this, let say let us take one more terminal for one, say here 1 here also and you so another terminal 5 here also let say. So, when you want to route 1, you try to put them in this first track because 1 is the vertex which no incoming edges. So, 1 will have to lay out first then you go to 2. So, 2 naturally there is an overlap you have to put it on the second track, 3 - 3 again there is an overlap here. So, 3 have to be put on the third track; 4 will again there is a overlap forth have to be put on the fourth track of course, here there is another up to 5, 4 to 5 also this is a constraint and 5 again there is an overlap. So, you have to use another track.

So, you can see that if we had a long chain in the vertical constraint graph, this is a bad scenario; because of which you need 5 tracks in the solution 1 2 3 4 and 5. So, intuitively we try to minimize the length of the longest chain in the VCG as much as possible, with the intuitive idea that if we are able to do it this will also reduce the number of tracks. And if you are able to reduce the number of tracks, it will mean that we are reducing the height of the channel which implies minimizing the area the routing area all right. So, as I have said the basic idea is to try and minimize the longest path in the VCG.

So, to do this we try to merge some nodes to reduce this longest path, this we shall see. And again the basic net merge channel routing algorithm does not allow doglegs or cycles in the VCG, but this is the more sophisticated kind of an algorithm that leads to a better solution as compared to the basic left edge algorithm.

(Refer Slide Time: 06:22)



How does it work?

- Partition the routing channel into a number of regions called **zones**.
- Nets from adjacent zones are merged.
 - Merged nets are treated as a **composite net** and assigned to a single track.

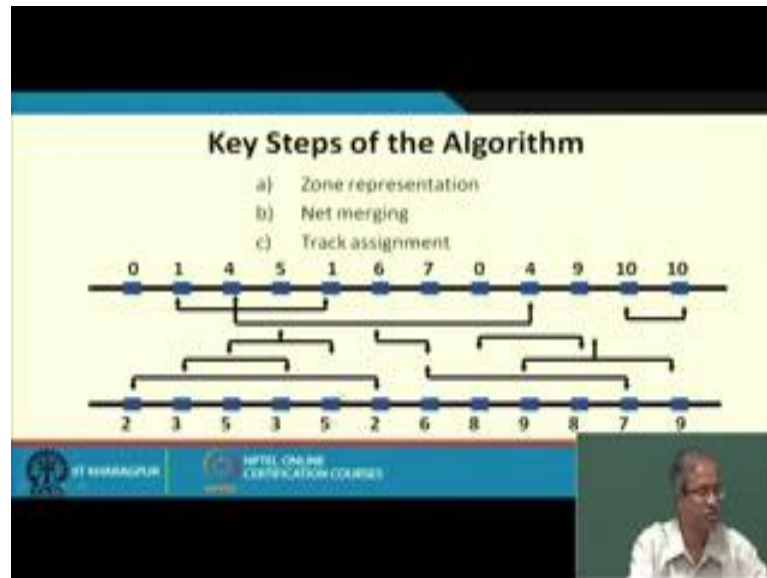
ST KIRANAGURU | NPTEL ONLINE CERTIFICATION COURSES

So, as I said we have the concept of zones, the total routing channel is divided into a number of zones which are the regions. In adjacent zones you look at the nets which belong to them, they can be merged as and when required, and once they are merged you get something called a composite net.

So, initially we had the individual nets. So, you define something called zones, among the adjacent zones there are some rules using which you can merge some of the nets, and you get a composite net. Now once you get a composite net, now you are saying that if a composite net consists of 2 nets i and j ; both i and j must be laid out on the same track that

is the constraint you are imposing additionally. So, regarding the steps of the algorithm there are 3 main steps: zone representation, net merging and track assignment.

(Refer Slide Time: 07:22)



So, we shall be illustrating these processes with the help of an example channel like this. You see that here we have a example channel with 10 nets, which are number from 1 to up to 10 and these horizontal lines show this span of each of the nets. this is the span of net 1, this the net of this span of net number 4, this is the span of net number 5, with connections here, here and here this is span of net number 9 so on. So, this allows multi terminal nets also. So, you see from this span of the nets you can create the horizontal constraint graph, which will tell net 1 and 3 are over lapping, 1 and 4 are over lapping, 1 and 5 are over lapping and so on and 1 and 2 is also over lapping.

So, in this way you can create some information about along every column; column 1, column 2, column 3, which are the nets which are over lapping; for example, in the first column you have only net 2, in the 2nd column we have net 1 3 and 2, in the third column we have 4 and 5 also there are 5 nets, in the fifth column also there are 5 nets, 6th column there is 1 4 5 and 2, 3 is no more 3 has stopped here. So, in this way along every column you try to find out which are the nets which are crossing that column. So, this is the basic idea behind zone representation.

(Refer Slide Time: 09:22)

Step 1: Zone Representation

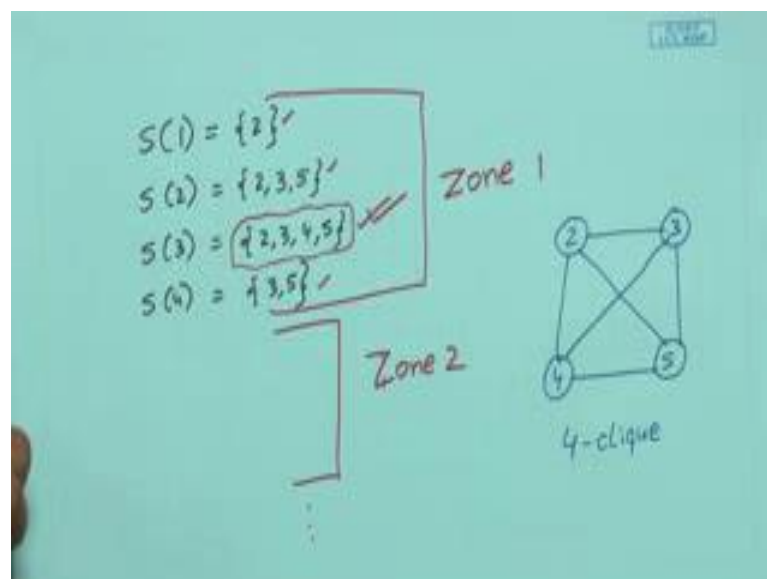
- Let $S(i)$ denote the set of nets whose horizontal segments intersect column i .
- Take only those $S(i)$ which are maximal, that is, not a proper subset of some other $S(j)$.
- Define a zone for each of the maximal sets.
- In terms of HCG / interval graph, a zone corresponds to a maximal clique in the graph.

ST BHARAGURU NPTEL ONLINE CERTIFICATION COURSES

(Video inset shows a man speaking)

So, as I said along every column of this channel like here column 1 2 3 4 like this, you define a set S_i , which will denote the set of nets which cross column i . So, we will be getting S_1, S_2, S_3, S_4, S_5 many sets.

(Refer Slide Time: 09:55)



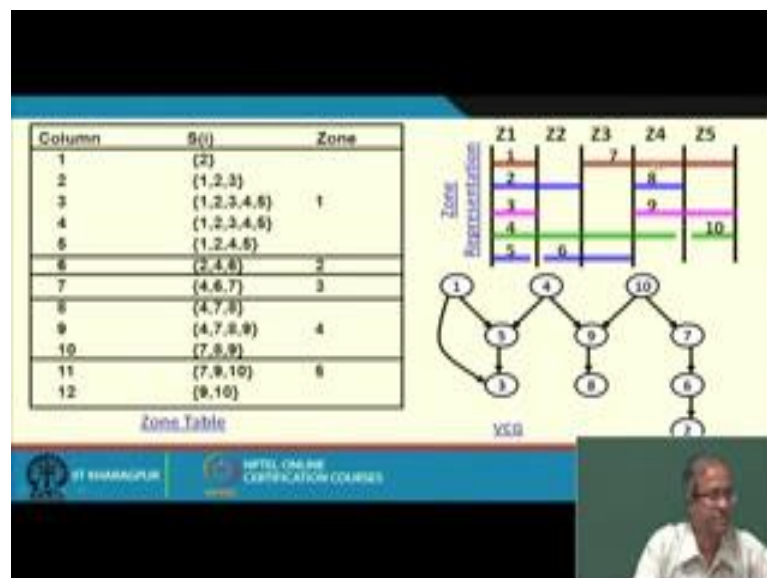
Now, among them you can find for example, you can find that your S_1 consist of only let say 2, let say S_2 consist of let say 2 3 and 5, let suppose S_3 consist of 2 3 4 5, this S_4 consist of let say 3 and 5 let say this is an example.

So, if you have set of such you can say zones, you can identify the maximal zone. So, how I identify? 2 3 4 5 is the maximal set, the others are subsets of these. 2 is the proper set 2 3 5 is also proper subset 3 5 is also proper subset. So, you ignore this subsets, you only take the maximal set and this will define one zone, this will be your one zone. Similarly you continue defining the other zones, this maybe this will be zone 1, then we will be having zone 2 and so on this is the basic idea.

So, the idea is that you take only those S_i that are maximal as I have illustrated, and for each of those maximal sets to define a zone. For horizontal constraint graph zone means it corresponds to a maximal clique in the graph, what does this means let us again come back to this. See 2 3 4 5 means what? In the third column the horizontal segments of 2 and 3 are intersecting like if you have a graph representation, where the vertices indicate the nets. So, these 4 belong to a set means there is a horizontal constraint between every pair. So, this is a clique 4 vertices, this is called a 4 clique.

Clique means maximally connected graph, every pair of vertices have edge between them. So, for every such maximum set you can define such a clique. So, a zone will indicate a maximal clique in the graph; let say.

(Refer Slide Time: 12:29)



So, for the example that you have given here, if you just workout in the first column you have only 2, second column 1, 1, this is 3, this is 2, third column it is 4, 1, this is 1, this is 4, this is 5, this is 3 this is 2, and so on if you just to work it, out you will see that you

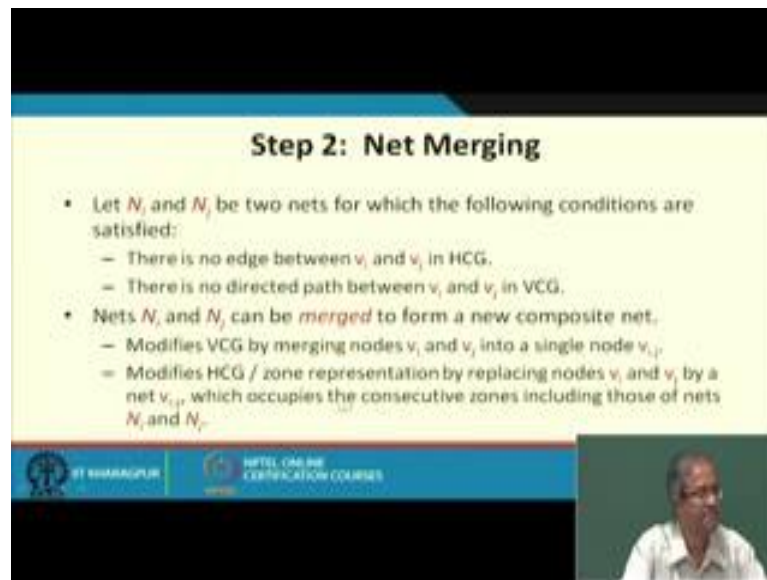
will be getting sets like this. Column 1 and the corresponding sets, column 1 it is 2, 1 2 3, 1 2 3 4 5, 1 2 3 4 5, 1 2 4 5 then 2 4 6, you see your sets are increasing, increasing, increasing here you are getting a maximum set 1 2 3 4 5.

It is decreasing, but still it is the subset 1 2 4 5 you take, but once you reach here where a new element 6 is coming, which is no longer subset of this you move to the next zone. In this way you define the zones, you continue doing this, this entire thing will correspond to one zone, you consider the maximum set here 1 2 3 4 5, then you go on continuing in sixth column you will see 2 4 6. You cannot include anything is 2 4 6, because in the next column your 4 6 7 and new nets 7 is coming in then you have 4 7 8.

Here of course, 4 7 8 9, 7 8 9 these 3 can group together because 4 7 8 9 is maximal these 2 are subsets of this, then you have 7 9 10, 9 10 this is the subset of this. So, for this particular problem you can define zones like this and you can identify that there are 5 zones, and this is just a zone representation we shall see later how you can manipulate this zone representation. This says that I have 5 zones, in the first zone I have net numbers 1 2 3 4 5, I am showing it like this 1 2 3 4 and 5 this is in zone 1, zone 2; zone 2 I have 2 4 6. So, 2 continues in zone 2, 4 continues, and a new net 6 start that is why a disconnection is showed here 6, zone 3: 4 6 7, 4 continues, 6 continues and a new net 7 comes here, zone 4: 4 7 8 9.

Similarly, 4 continues, 7 continues, 8, 9 and 7 9 10, 7 continues, 9 continues, 10 this is the zone representation and from this problem again you can define the vertical constraint graph, 1 on top of 3, 4 on top of 5, 5 on top of 3 and so on. So, I am showing the VCG here you can verify. So, we have the zone representation, we have the vertical constraint graph right. So, we have identified these zones.

(Refer Slide Time: 15:42)



Step 2: Net Merging

- Let N_i and N_j be two nets for which the following conditions are satisfied:
 - There is no edge between v_i and v_j in HCG.
 - There is no directed path between v_i and v_j in VCG.
- Nets N_i and N_j can be **merged** to form a new composite net.
 - Modifies VCG by merging nodes v_i and v_j into a single node $v_{i,j}$.
 - Modifies HCG / zone representation by replacing nodes v_i and v_j by a net $v_{i,j}$ which occupies the consecutive zones including those of nets N_i and N_j .

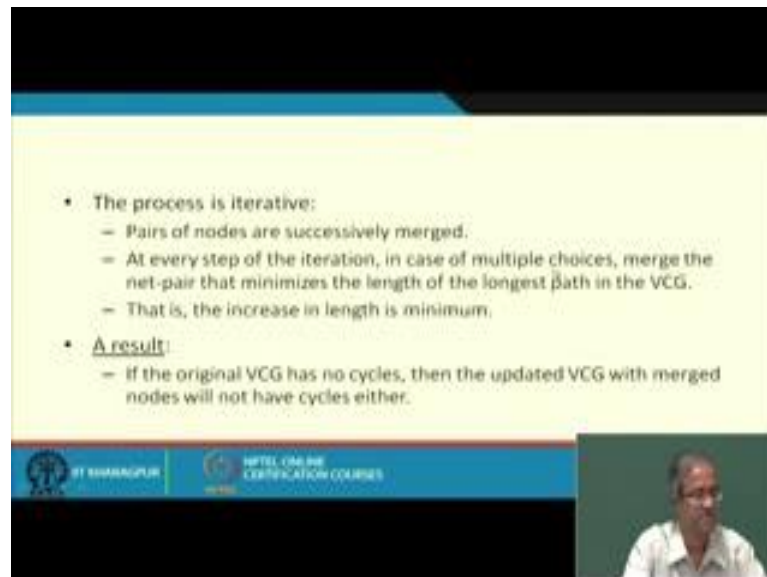
Step 2 says now we move to merge some of the nets. Now again going back here, so I have the nets 1 2 3 4 5 6 7 8 9 10. Now one thing you see just intuitively I am showing, when you see net number 5 and 6 they can share a track across zones 1 and 2, because they do not have anything in common there is a disconnection; similarly 1 and 6, 3 and 6 they can share right. So, the idea is something like this, if there are 2 nets N_i and N_j for which we have these 2 conditions true. First there is no vertical constraint between them, because if there is vertical constraint you cannot merge them into a same track right and also there is no vertical constraint and also this is no horizontal constraint.

The first one is horizontal constraint, second one is vertical constraint. So, means there is no horizontal overlap, there is no dependency in VCG, then only you can possibly merge them in a single track. So, if you have 2 nets for which such conditions hold, then you can merge the 2 nets together to form a so called composite net. So, once you form a composite net what you do in the VCG? This V_i and V_j which you have merged, you delete V_i and V_j , and replace it by 1 node $V_{i,j}$ it is a composite node; and you see the horizontal constraint graph and this zone representation actually 2 alternate ways of representing.

You see the HCG also talks about the same thing, in zone 1 there is a 5 clique 1 2 3 4 5; zone 2 there is 3 clique 2 4 6, zone 3 there is a 3 cliques 7 4 5. So, you can either represent it the HCG as a graph, or you can show it as a zone representation both are

equivalent. So, also modify HCG by replacing V_i and V_j by a node net V_{ij} , which means that now I have merged V_i and V_j in to a composite net, which will occupy consecutive zones corresponding to N_i and N_j and later on they will be placed on the same track this is the basic idea behind net merging.

(Refer Slide Time: 18:28)

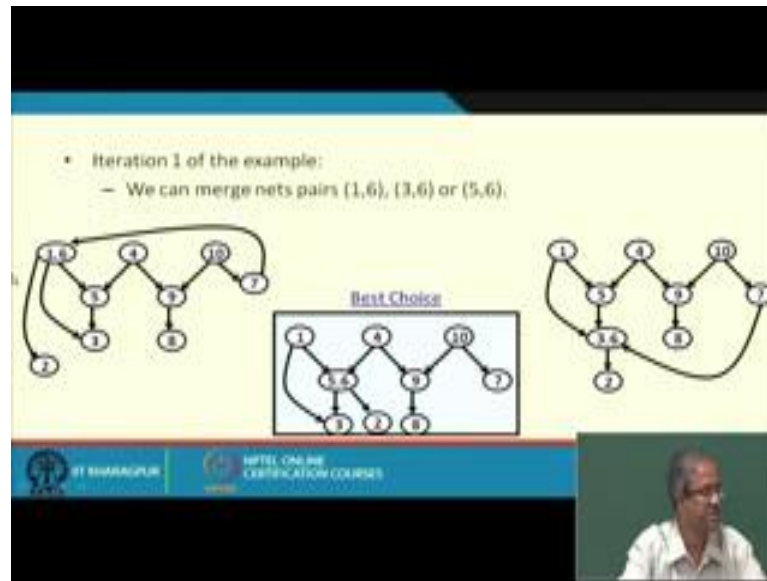


- The process is iterative:
 - Pairs of nodes are successively merged.
 - At every step of the iteration, in case of multiple choices, merge the net-pair that minimizes the length of the longest path in the VCG.
 - That is, the increase in length is minimum.
- A result:
 - If the original VCG has no cycles, then the updated VCG with merged nodes will not have cycles either.

Now, I shall be illustrating this with an example, the process will be alterative; that means, you continue doing this, at every step to find out a pair of nodes which can be merged and merged them like for example, between Z_1 and Z_2 as I have said you can merge 1 and 6, you can merge 3 and 6, you can merge 5 and 6, but 2 and 6 you cannot merge because there is a constraint horizontal. 4 and 6 you cannot merge there is a horizontal constraint. So, there are 3 choices 1 6, 3 6, 5 6 in this graph you can merge 1 and 6, 3 and 6 or 5 and 6.

Now, you have several choices, now which one to use there is a heuristic. It says that if you have multiple choices, you merge that pair of nets which minimizes the length of the longest path in VCG. Because you recall one of our objective here is also to minimize the maximum length path in VCG, because that will imply reducing the number of tracks. So, you merge those pair of nets which will lead to the maximum reduction in the maximum length in VCG. So, there is a result of course it is easy to verify, if your original VCG has no cycles which have assumed, that even after merging you cannot have cycles appearing. So, that will also have no cycles.

(Refer Slide Time: 20:10)

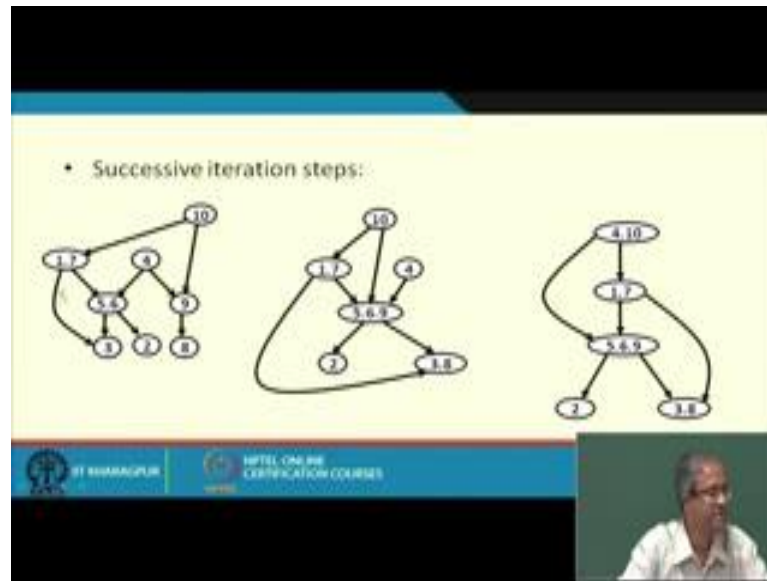


So, let us take an example; when iteration 1 as I have said this was your VCG right, and I said I can merge 1 6, 3 6 or 5 6 in the first step. So, between Z 1 and Z 2, then we will see Z 2, Z 3 then you will see Z 3, Z 4 and so on. So, in the first step I have 3 choices 1 6, 3 6, and 5 6; suppose I merge 1 6. So, my VCG gets transform in to this, if I merge 3 6 you can verify these, my VCG gets modify to this; and if I merge 5 6 my VCG gets modify to this. Now let us see the maximum paths; in the first case what is the maximum path? You see from 10, I can get a maximum path 10 to 7, 7 to this, this to 5, 5 to 3.

So, there is a 4 length path and here, here you have maximum is I think 1 to 5, 5 to 3 6, 3 6 to 2 3. So, this is better than the first one, but the best choice is this merging 7 6 because here you get 2 only 2. So, this example shows if you have multiple choices, you explore all the alternatives and select the best one and there is namely merging 5 and 6.

So, at every step you are doing this; that means, when you are en counting zone 1 and zone 2 conservative zones, to find out which nets across these conservative zones can be merged select that select the best one, then you move on to the next pair Z 2 and Z 3, across them which is the best pair of nets which can be merged you repeat this iteratively. So, I am showing you the results of the iteration, so in the next step you can find that you can merge 1 and 7. So, you can just go back to that zone representation.

(Refer Slide Time: 22:09)



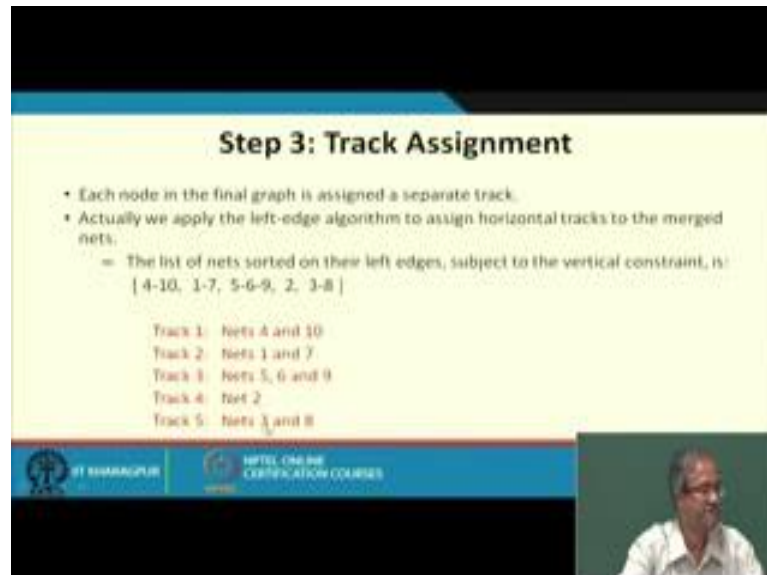
Between next steps you can see that 1 and 7, Z 3 I have 7. So, 7 can be merged with one of the pervious nets, and 1 and 7 gives you the best reduction. Because this was your initial think, if you merge 1 and 7 your length remains same or it reduces. So, I am showing the best solutions merge 1 and 7 in the first step, then merge 5 6 and 9 you get a composite node as 5 6 9, than merge your 3 and 8 also merge 3 and 8 also you get this. So, this 1 7 was pointing to 3 now it will be pointing to 3 8. 9 was pointing to 8, 9 will be pointing to 3 8, the last step you merge 4 and 10 so you get this.

So, you have a solution here where you have merged all the nets that are possible to merge, and you can see that now the maximum length of the path that you get is 1 2 and 3, and see length is not the issue, you see how many such composite nodes you have. You see I have 1 2 3 4 5 now I cannot proceed any further if. So, I have arrived at a graph with 5 vertices, now if I see that I cannot merge any further pairs of vertices from this graph, what does it mean? Either this pair of verities that may correspond to an individual net or a composite net, they either they are over lapping horizontally; that means, they cannot be put on this same track or there is a horizontal constraint, I have to put 1 and top of the other they cannot be put on the same track.

Now, I have arrived at a solution where you cannot merge anything else any further. So, this is your ultimate solution after merging the nets, what this graph tells you? This graph

tells you that 4 10 can be merged in a track, 1 7 can be merged in a track, 5 6 9 can be merged, and 3 8 can also be merged.

(Refer Slide Time: 24:49)



Step 3: Track Assignment

- Each node in the final graph is assigned a separate track.
- Actually we apply the left-edge algorithm to assign horizontal tracks to the merged nets.
 - = The list of nets sorted on their left edges, subject to the vertical constraint, is:
 $\{ 4-10, 1-7, 5-6-9, 2, 3-8 \}$

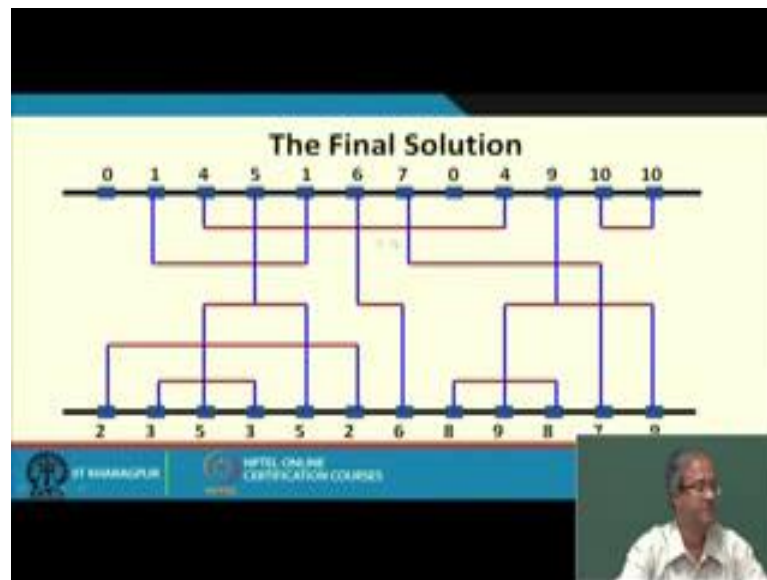
Track 1: nets 4 and 10
Track 2: nets 1 and 7
Track 3: nets 5, 6 and 9
Track 4: net 2
Track 5: nets 3 and 8

ST KAMARAJU
NPTEL ONLINE CERTIFICATION COURSES

So, last step track assignment you do it according to that. These are the vertices in the graph first track you see now you just order by the vertical constraint graph, you have to take 4 10 first no incoming edges, then 1 7, then 5 6 9 these 2 can be done in any order 2 and 3 point at any order.

So, track 1 I put this, track 2 I put this, track 3 this 4 5. 5 I can put both 3 and 8 together 3 8 and 2 and 3 I need I can change this track 4 and 5 can be swapped if required.

(Refer Slide Time: 25:27)



So, I get my final solution like this; first, second, third, fourth and fifth track. So, this is the basic idea behind the Yoshimura Kuh algorithm that analyzes this zones that tries to minimize the longest part in the HCG and in general provides much better solution as compare to the standard extended left edge algorithm, all right.

(Refer Slide Time: 25:51)

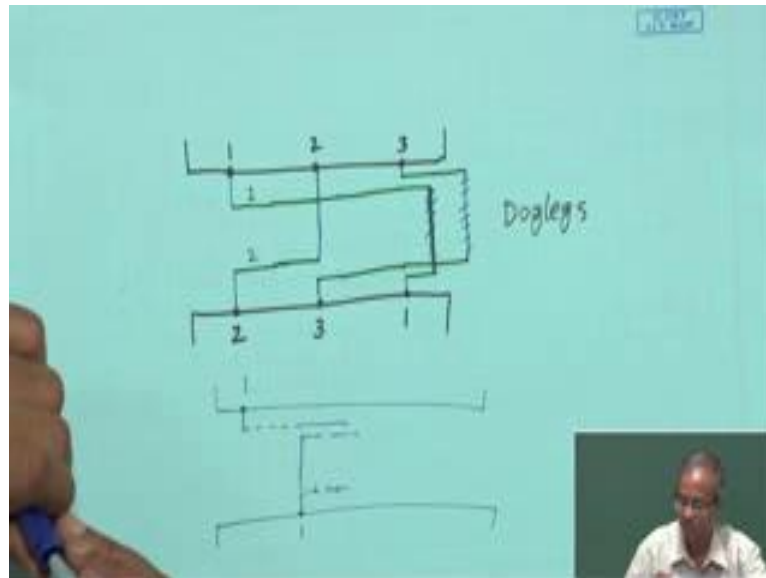
Greedy Channel Router

- The routing algorithms discussed so far route the channel one net at a time.
 - Based on left-edge algorithm or some of its variation.
- The Greedy Channel Router algorithm routes the channel column by column starting from the left.
 - Apply a sequence of greedy but intelligent heuristic at each column.
 - Objective is to maximize the number of tracks available in the next column.
- Can handle problems with cycles in VCG.
 - May need additional columns at the end of the channel.

Next we come to a method which is greedy. Greedy channel router means I am not looking at the whole problem at the same time; I look at the problem incrementally. Specifically I start from the first column I go on moving to successive column first to

second, second to third, third to fourth and I incrementally try to route the nets as I move along. Greedy means at every step I am trying to find out the best solution, but overall it may or may not lead to a good solution.

(Refer Slide Time: 26:47)



Now, the advantages that here you can handle arbitrary situation including cycles in the VCG; intuitively let me give an example which can give you the idea. Suppose I have a simple example with a VCG cycle; say 1 to 2, 2 to 3, 3 to 1 see the methods that we have seen so far is unable to handle this, now let us see how we do this here. Here we start with the first column, column by column we move, here there are only 3 columns let see. First column I see that 1 is on top of 2. So, I have; so what I do? I start net for 1, I start a net for 2, there will be a set of rules I shall we shortly talking about so I have done this.

So, now, I move on to the second. Second one what I see that already I am continuing with 1, I have already continuing with 2, but I cannot here 2 and 2 I can connect no problem. So, 2, I can connect to this 2 because there is a terminal for 2 and 3 new connection is coming. So, 3 will start on a third track and 2 is already finish. So, 2 we need not extend, but 1 we have to extend.

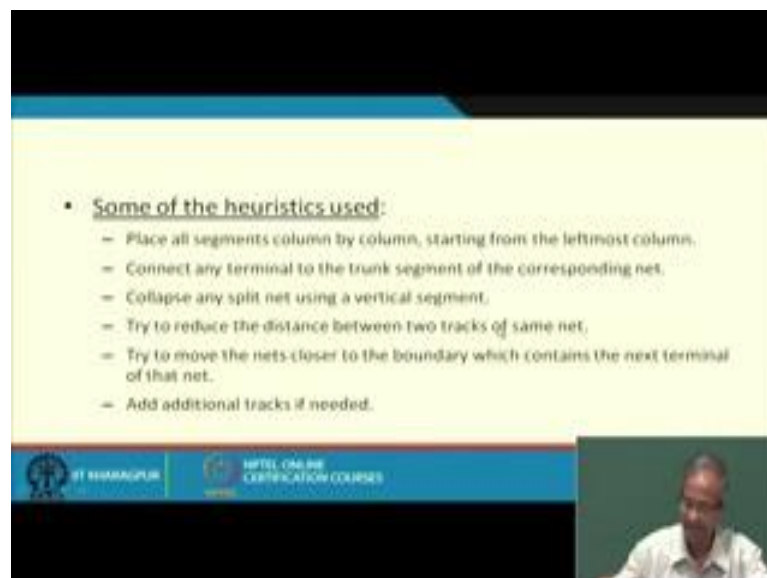
Now, we move to the third. So, this 1 comes here, 3 comes here. We see now we have a problem, if we connect this net to 3 and this net to 1, there is a short circuit coming in because this as to be connected to the bottom terminal, this have to be connected to the top terminal. So, both if you connect by blue there is a short circuit. So, what you do if

such a scenario occurs, we do not connect them here what we do? We again start another net from here and another net from here.

So, now, this will also be 3, this will also be 1. So, now, you move to the next columns. So, it may require additional columns. So, what you do as you move to the next column, you connect this 1 with 1 sorry in this blue, you connect 1 with 1 in blue, you move this again further connect 3 with 3. So, you make the connection something like this, which means 2 things that you are allowing doglegs, because that when you are connecting say for example, 3 with 3 we have 2 horizontal segments, 1 with 1 we have also 2 horizontal segments. So, you are allowing doglegs, but you may require some additional columns in the process as this example shows, right.

So, this is the basic idea behind the dogleg channel router. So, I shall be telling you the basic rules that were followed. So, the greedy channel router as I have said it routes the channel column by column starting from the left, and there are set of rules or heuristics I shall be talking about, this can handle arbitrary problems with any kind of constraints in the VCG including cycles, but I have shown this may required additional columns at the end of the channel, right.

(Refer Slide Time: 30:33)



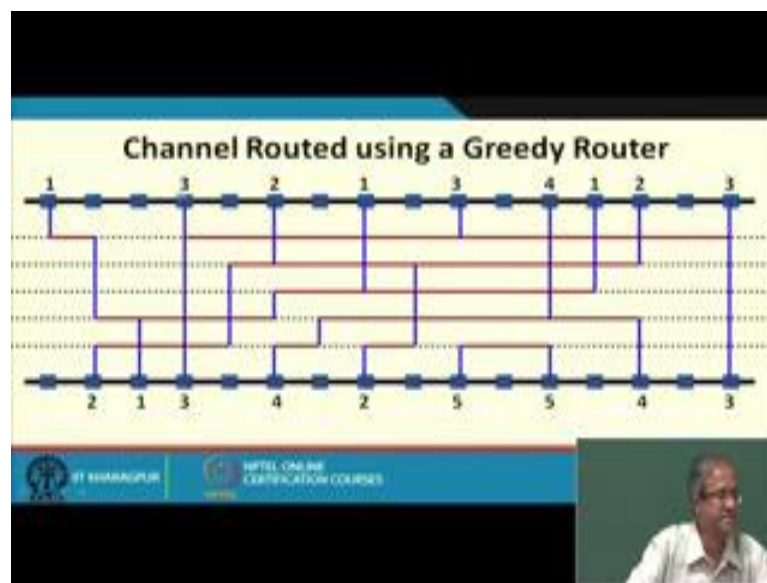
- Some of the heuristics used:
 - Place all segments column by column, starting from the leftmost column.
 - Connect any terminal to the trunk segment of the corresponding net.
 - Collapse any split net using a vertical segment.
 - Try to reduce the distance between two tracks of same net.
 - Try to move the nets closer to the boundary which contains the next terminal of that net.
 - Add additional tracks if needed.

So, some of the heuristics are like this, place all segments column by column starting from the left. Connect any terminal to the trunk segment of the corresponding net whatever you find, collapse any split net using a vertical segment which we are doing at

the end here. See here at the end these 2 split nets where connecting the vertical segment, here also we are using vertical segment, this is what I mean by collapsing 2 horizontal segment by connecting it in a vertical segment, and try to reduce the distance between 2 tracks of the same net.

Like the idea is that if you see that 1 is here also like something like this. If you see that you have this 1 here and there is another 1 coming from somewhere, this 1 net you have a drawn like this, 1 also it is coming like this you try to move it as closer to this as possible, this is says to bring to segments of the 2 like here, 3 and 3 are running in parallel, 1 and 1 we are running in parallel, you try to move them as closer together as possible here there are quite further about it says, and try to move the nets closer to the boundary which contains the next terminal. If I see that my next terminal is here, I try to move it below down below as much as possible.

(Refer Slide Time: 32:09)



And of course, we will need additional tracks if required, and if you follow this heuristics this is just an example which has been worked out I suggest that you work it to out by hand, and see how it works we do it like this from left we go on starting from one assign to the same track.

Second one you start track 2, and using heuristic the next terminal for 1 is below, that is why you bring it as much down below as possible. So, you bring 1 here. So, in this way you proceed again this terminal 2 is coming, but next terminal for 2 is up. So, bring it as

much up as possible. So, you use all the heuristics and you get a solution like this see that therefore, some of the nets you get doglegs, here also you get a dogleg.

So, with this we will come to the end of this lecture, so we shall be continuing with our discussion in the next lecture.

Thank you.