

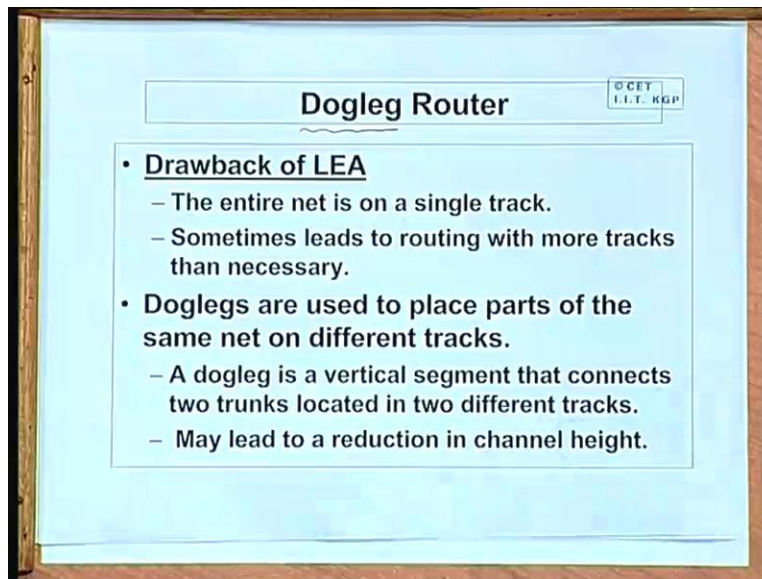
Electronic Design Automation
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture No #26
Backend Design: Part – XII

In the channel routing algorithms in our last class, so if you recall the algorithm that we had looked at are the basic left edge algorithm and one of its extensions. Now in the basic left edge algorithm the nets were considered to be assigned to tracks starting from the left most boundaries of the nets and in that sorted order the nets are considered. Now in the basic left edge algorithm we had assumed that there are no vertical constraints. But in practice there will be vertical constraints so means one particular terminal of one net may lie just above the terminal of another net. So in general there will be vertical constraints and we had looked at an extension or a modification of the basic left edge algorithm where vertical constraints were also present. Now there, the idea or the consideration was that if there is an edge in the vertical constraint graph from net I to net j then net I must be considered first and then net j.

So in terms of the vertical constraint graph just by looking at the direction of the edges we can also find out an ordering. Just not necessarily only based on the left edge of the nets also on the direction of the arrows that which of the net has to be considered before the other. So with this 2 modifications we are able to route a net where there where some vertical constraints. But one restriction was there in the solution that we had obtained is that, well there, whichever the algorithm we are using any one of the two. Say a particular net was assigned totally to a single track. So there was no concept that 1 net would be broken up and the 2 or 3 verticals 2 or 3 horizontal segments of that net would be assigned to different tracks that was not the concept. The entire net was assigned to a particular track ok. So the algorithm that would be considering first today that can relax that restriction. There you may allow breaking up of the horizontal segments of a net across several tracks ok.

(Refer Slide Time: 03:30)

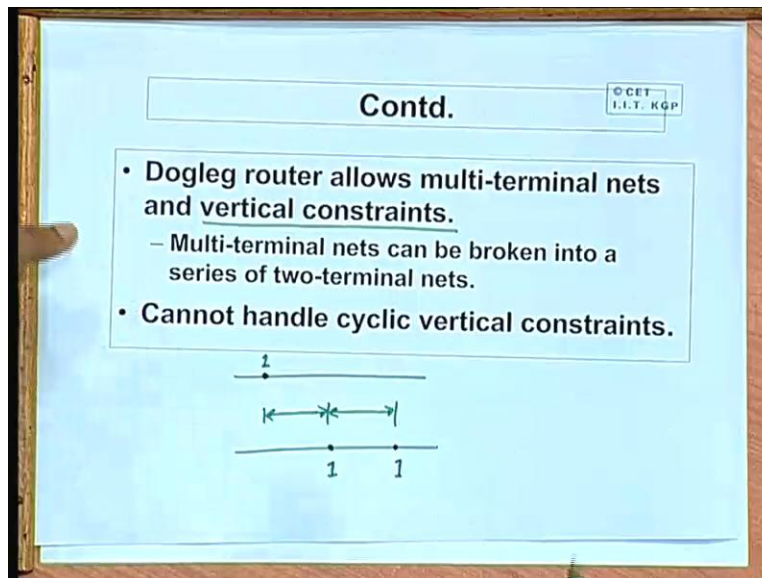


So this routing algorithm that we would be looking at this conventional is called dogleg router. Because whenever we break a horizontal segment of a net across more than one track, it is called a dogleg. So the basic dogleg router tries to overcome the drawbacks of the left edge algorithm that we have just mentioned namely the entire net is assigned to a single track. But in fact it can be easily proved that in the absence of any vertical constraint this basic left edge algorithm gives you the optimal solution in terms of the number of tracks required. But if there are vertical constraints then optimality cannot be guaranteed that we will see though an example shortly. So in the presence of vertical constraint you may get a routing solution using the left edge algorithm which takes more number of tracks than the optimal solution.

So this is one motivation where we can possibly look for an algorithm which can break a track a net across tracks. Say a particular net instead of routing like this, we may route it like this maybe. So this is what is called a dogleg we had mentioned earlier. So in a dogleg the basic idea is you are placing or allocating parts of the same net on different tracks. So when you have for parts of the same net on different tracks then there will be some vertical segments on the other layer which will be joining these 2 obviously. So actually these vertical segments are called the dogleg connections. Doglegs are the vertical segments of the wire which will be connecting these 2

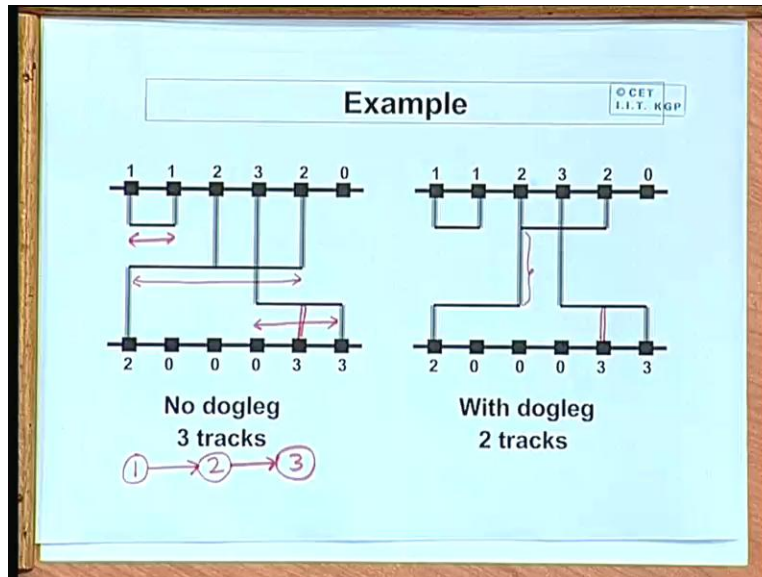
trunks which are located on 2 different tracks. Now we will see through an example that this may sometimes lead to a reduction in channel height or the total number of tracks will require to complete the routing.

(Refer Slide Time: 06:03)



So the routing algorithm that we would be looking at, this allows multi terminal nets. Of course the left edge algorithm also allows multi terminal nets. But here also we allow multi terminal nets and vertical constraints as well. Now in the algorithm if you have a multi terminal net it gets broken up into a series of 2 terminal nets. For instance if you have a case where you have a terminal say 1 out here 1 here and 1 here. This will be broken up into actually 2 nets. First one will be starting here and ending here the second 1 will be starting here and ending here. Starting from the left most boundary, as you go on sequentially scanning the terminals of the net from left to right. So you take the 2 terminal nets which come in the way and yes break it up in that way. So if there are 4 terminals there will be 3 such 2 terminal nets one after the other. But 1 restriction of the algorithm that will be presenting is that it cannot handle cycles in the vertical constraint graph ok well. So now let us first take an example which will show you that how the use of doglegs can reduce the number of channels required tracks required. So a very simple example.

(Refer Slide Time: 07:40)



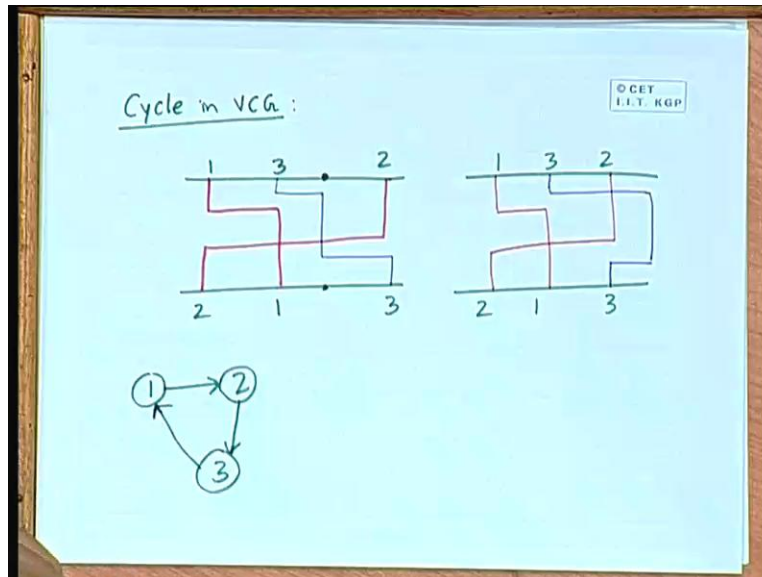
So let us take a channel routing problem like this where there are 3 nets one, 2 and three. So as you can see the net 1 starts from here and spans up to here net 2 it starts from here and span still here and net 3 starts here and ends here. And if you look at the vertical constraint there is a constraint between 1 and 2 so there will be 1 arc between net 1 and net two. There will be another arc from net 2 to net three. So there is a directed path in the VCG from 1 to three. Well now if we allow the extended left edge algorithm honoring this ordering of the VCG then obviously 1 has to be done first, then 2 and then 3 you have no other choice because there are directed arrows from 1 to 2 and 2 to three.

So first when you assign 1 you assign to the first available track 2 well you can only assign the second track to a. Because it is already occupied 3 three again well you can see you well there is another terminal here I am not shown in this there will be a connection here also. So due to this connection you cannot assign the net 3 to track 1 you have to assign a new track to it right. So this is a solution which requires 3 tracks. Now 1 result follows from this example is that if you have this kind of a directed path in the VCG and if you say that I will not breakup a net across tracks then the length of the path maximum length of the path in the VCG that also will give you a lower bound to the number of tracks required.

Well, the 1 bound comes from the horizontal constraint graph you take the size of the maximal sub graph the clique that will give you the number of tracks required minimum. And again if you look at the path length in VCG that will also give you a lower bound. But the same solution if you allow a net to be split across tracks then we can get a solution like this. Here net 1 as usually is placed on track one, net 3 is as usually placed on another track. But net 2 instead of placing as a whole on 1 track in which case a new track was required. If you break it up like this then you can say you can do with 2 tracks also. So a part of net 2 is assigned to the same track as 3 and a part of net 2 is assigned the same track as one. So here you have a dogleg this is the dogleg junction.

So doglegs can reduce the density of the channel in terms of the number of tracks. This simple example illustrates. Yes. [Students Noise Time: 11:06] In this example you can have 2 tracks without having [Students Noise Time: 11:14] this one. [Students Noise Time: 11:17] Yes, yes. [Students Noise Time: 11:19] In that case there will be configuration 2 and 3 there is connection to 3 and 2 also there will be a short circuit here just due to that. So if this 3 was not there you could have done that ok fine. Now ok this example gives you the motivation why dogleg is used for. Means how it can reduce the number of tracks. Well this is from the view of optimization of the solution. But there are certain situations where dogleg is mandatory but although the algorithm we would be considering right now that does not consider that scenario but let me also talk about that scenario.

(Refer Slide Time: 12:07)



Where there is a cycle in VCG. Now the basic dogleg algorithm we have considering right now that cannot handle cycles. So let us take a simple routing problem a channel routing problem with nets placed like this. So as you can see in VCG there is a path from 1 to 2 there is a path from 3 to 1 there is path from 2 to three. So there is a cycle. Now in terms of the solution if you try to apply the basic left edge algorithm then well you just ignore the VCG for the time being. Suppose you try to assign to the tracks say you first route net 1 then say you route net two. But while routing net 3 will land up into a problem.

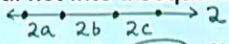
You cannot assign net 3 into a single track because there will be either conflict otherwise short circuit with this net or there will be a short circuit with this net. So what you can do there actually 2 alternatives you can either have if you have a spear column in between, then you can either have a solution like this with another track and a dogleg. So in this case you require 3 tracks 1 2 and 3 or if you do not have a sphere column in between. [Students Noise Time: 13:50] Ok. Sorry forth track for this 2, 3 and 4 right. So there are 4 tracks which are required. This you can do provided you have a free terminal column a free column in between.

But if the pins are already packed there is there is no free column in between then you will have to do something like this 1, 3, 2, 2, 1, 3. Then the first 2 nets will be routed as usual 1 to 1, 2 to 2 ok. There will be dogleg again no doubt but the dogleg will now be put like this. There will be an additional column which will taken up at the end of the channel. So if we have a space in between you can use that or you will have to extend the net towards the right or towards the left and reach the end of the channel and use up an additional column of their ok. So there are some algorithms that will be seen later which can take care of this kind of a scenario. But the basic dogleg algorithm we would be looking at now that does not take into accounts cycles in VCG right ok.

(Refer Slide Time: 15:22)

© CET
I.I.T. KGP

Dogleg Router: Algorithm

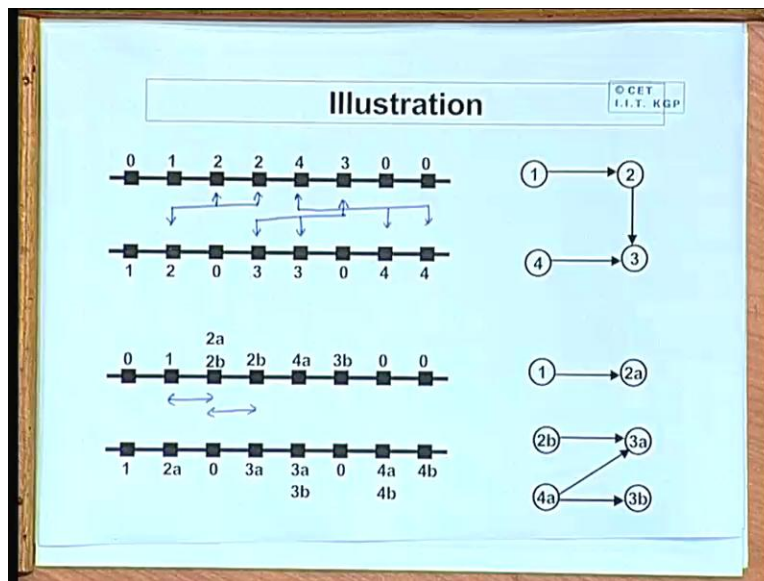
- **Step 1:**
 - If cycle exists in the VCG, return with failure.
- **Step 2:**
 - Split each multi-terminal net into a sequence of 2-terminal nets. 
 - A net 2 .. 2 .. 2 will get broken as 2a .. (2a 2b) .. 2b.
 - HCG and VCG gets modified accordingly.
- **Step 3:**
 - Apply the extended left-edge algorithm to the modified problem.

So the basic algorithm works like this. Well of course if there is a cycle the VCG this algorithm cannot handle it, so you will be returning with failure you cannot route it. Step 2 as I had mentioned if you have a multi terminal net you break it up into a sequence of 2 terminal nets. Well the way you do it is like this. Suppose you well just in terms of the horizontal direction I am showing suppose this is the horizontal span of a net and these are the different terminal positions either on top or on bottom say here, here, here and here. Suppose this cons, this if the span corresponding to net number two. So in the original sub problem all this terminals were

labeled as 2, 2, 2 and 2. Now when you split it up you call this net segment as 2a call this as 2b, call this as 2c.

Which means for the first segment the pair of terminals will be labeled as 2a and 2a for the second segment we label as 2b and 2b. So some of the common terminal will be some times called 2a some times called 2b similarly the third segment will be called 2c and 2c. So if you have a multi terminal net you break it up into a sequence of 2 terminal nets. So 2, 2, 2 will be broken up into 2a, 2a, this 2a and 2b are actually the same terminal they are assigned to given. I will give an example shortly and after you do this you apply the extended left edge algorithm. See once you have broken this up like this, what you get. Well, let us take an example then I tell you.

(Refer Slide Time: 17:22)



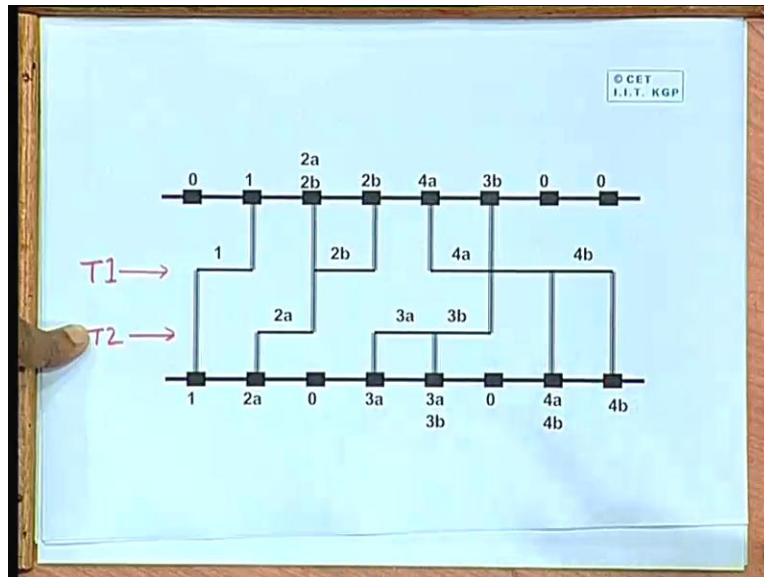
Take this this is a simple example where the vertical constraint graphs look like this. This is an edge from 1 to 2, 2 to 3 and 4 to 3 this is the VCG. Now here you see that there are some nets which have which are multi terminal 1 is this net number 2 where you have a terminal out here, you have a terminal out here you have a terminal out here. This is your net number two. Similarly you have net number 3 which has a terminal out here a terminal out here and a terminal

out here and net number 4. Net number 4 is like this here, here and here. So here all these nets, 2, 3 and 4 these are multi terminal nets. So as far as our strategy we break these nets up into individual 2 terminal nets like this net 2, 2 and 2 will breakup into 2a, 2a and 2b, 2b actually there are now 2 different nets 1 is 2a like this 1 is 2b like this.

Of course 2a and 2b touch at a terminal obviously. Similarly, net 3, 3a and 3a and 3b and 3b. Similarly 4; 4a and 4a, 4b and 4b. Now as we do this renumbering our VCG also gets modified. So now the constraint will be 1 to 2a, 2b to 3a, 4a to 3a and 4a to 3b both that is all. So our VCG also gets modified. But once you do this what you gain is that, now since we have split this net to into 2 different segments 2a and 2b and these 2 net segments are treated independently. So when you apply the left edge algorithm, you can possibly assign 2a and 2b to 2 different tracks that is the idea. But here although I have not shown the horizontal constraint graph, normally if 2 nets are like this which start and end at a particular column position.

There will be a horizontal constraint between then they cannot be put on the same track. But since 2a and 2b actually belong to the same net there will be no horizontal constraint between 2a and 2b that is the only 1 change you make in the HCG. So I am just noting it down. In HCG in horizontal constraint graph there will be no edge between say 2a and 2b as an example I am taking similarly between 3a, 3b and between 4a, 4b. Because if the need arises 2a and 2b can also be placed on the same track ok. But if they belong to 2 disjoint nets you could not have done that ok fine. So once you do this for this problem if we apply the extended left edge algorithm the solution comes like this.

(Refer Slide Time: 21:07)



Start from left to right you just again look back at the constraint graph just to see which in which order we have to consider. Look at the constraint graph 1, 2b and 4a are the ones which do not have any incoming edges ok. But once you finished 1 you can also start with 2a. So here from the left edge net 1 comes first. So you assign net 1 to first track this is your track one. Well once net 1 is completed net 2 is 2a is ready. 2a cannot be assigned to the same track because there is a horizontal constraint between 1 and 2a they belong to different nets. So 2a will be assigned to the second track, this is track 2. Well once you have done 2a you can also do 2b because 2b also does not have any constraint 2b can be started. Now according to the left edge algorithm you will be assigning 2b to the first available track. So 2b will not be assigned to track 2 rather it will assign to track one, 2b will be assigned here. Next will come is after 2b you see once 2b is done you can start with 4a because 3a and 3b cannot be started 4a is the first we have to take. So after 2b we will have to take 4a this is the first net you have to take. So this will again we assign to the first available track t1 4a up to here. Well 4b does not have any constraint 4b can be taken up anytime. But now you can take up 3a and 3b.

3 cannot be assigned first track because 4a has already been laid here 3a has to be assigned to track 2 similarly 3b. 4b there is no constraint you put it on the first available track t1 then you

join the nets. This will be the final solution. So this in the final solution some of the nets are not split across tracks but some of the nets are. So the basic dogleg algorithm tries to minimize the number of channels by putting the different segments of the same net across tracks. This is the basic idea ok. So in the absence of a cycle in the VCG this method works pretty well right ok. But the problem in this method is that well if we have many nets with multi terminal pins then the number of nets we have to consider in the modified problem that goes up quite considerably the number of these split nets become pretty high. Because ultimately we are applying the left edge algorithm the number of nets we have to consider there be high large. So there is another algorithm which we will be presenting now that gives a good solution. But that does not consider a dogleg without dogleg that gives you a fairly good solution. This is a pretty well-known method.

(Refer Slide Time: 24:33)

© CET
I.I.T. KGP

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.
- **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.

No doglegs

1 2 3

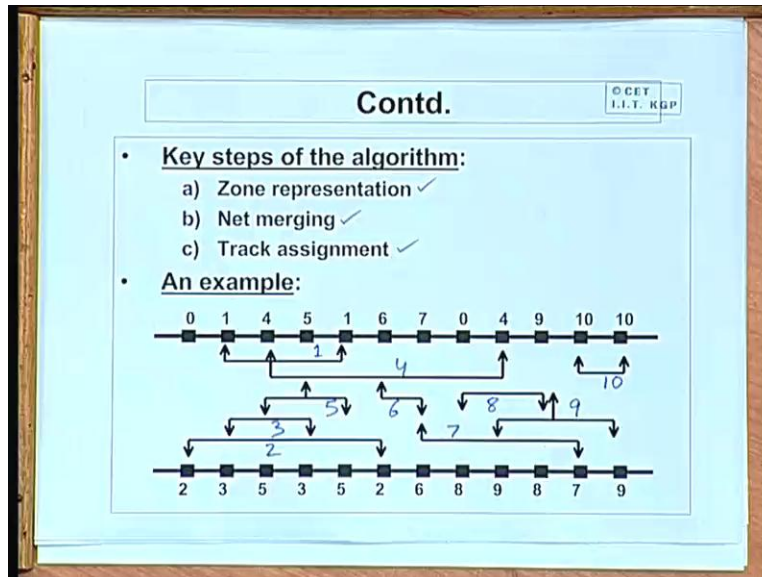
This is called net merge router this was initially proposed by Yoshimura and Kuh. Well now as the name implies net merge if the idea is that some of the nets were trying to merge well what is the basic idea. Suppose I have net i and net j . There are some constraints, if net i and net j satisfy those constraints we can merge them into to 1 composite net or super net whatever you call. So after merging net i and net j will become a single composite net. So in this way we would be

continuing to do this merging as far as possible. Now after merging what will happen is that a composite net will have to be placed on a single track. So instead of considering the individual nets and let the left edge algorithms spend a lot of time to assign them to the tracks. Now a composite net will be assigned to a track. Now when we assign a composite net to a track all the individual nets that belong to that composite will automatically be assign to the same track this is the basic idea behind the method.

So let us see how it works. Now this starts from the primary switch I have just mentioned sometime back. That if there is a directed path in the VCG of length p then in any solution were you are not using dogleg you will need at least p number of tracks. This of course provided you are not allowing doglegs but if you allow doglegs it can less. Now in the net merge router we are trying to merge nets in the way we have just mentioned. But we are always trying to minimize the longest path in the resultant VCG because when we merge the VCG also will get modified. So we are trying to minimize the longest path in the VCG and the process successively merges the nodes of the graph in order to achieve this object. So as I had mentioned this does not allow doglegs or cycles in the VCG. So this you can consider this is another extension to the left edge algorithm basic left edge algorithm ok. So the way it work is likes this this will partition the routing channel into a number of regions which are called zones. Routing channel means this is the routing channel there so the pins are located on top and bottom.

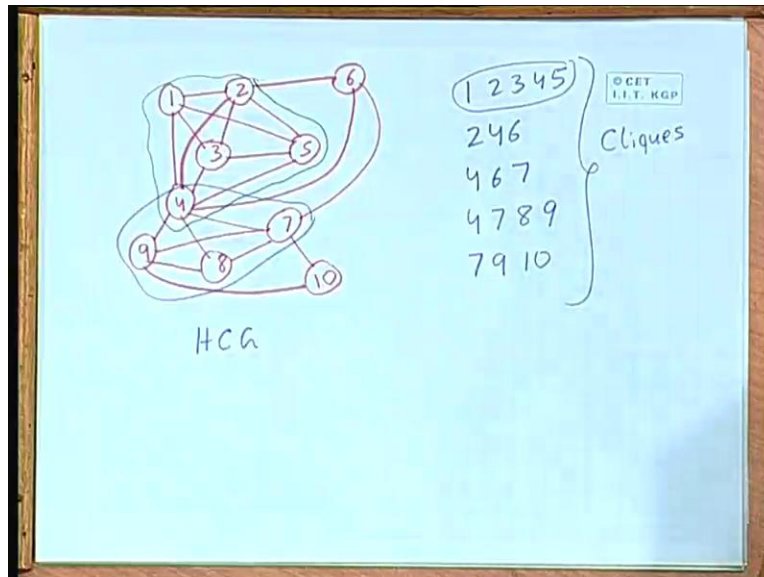
So we can make some vertical slices of this sub problem and this you can call zone1, zone2, zone3 like this. And when you are saying that we are merging nets, well we cannot merge 2 nets which belong to the same zone. Because they will lead to a horizontal constraint, then nets will be belonging to the same zone. If there is horizontal constraint between them, they are left edge and right edges or the spans overlap. So we can merge 2 nets which belong to adjacent zones right and as I had said once you merge them the merge nets will be treated as composite nets. And when you **allo when when you** allowed them to the tracks the composite net will be assigned to a single track. So now let us see how the zone definition is carried out and how we can use this to merge the nodes and assign them to tracks ok. So in this algorithm there are 3 steps essentially.

(Refer Slide Time: 29:00)



The steps are of course the first step is to identify the zones zone representation and once you identify the zones, you can start merging the nets and once you have finished merging them, you can assign the composite nets into tracks. These are the 3 steps in this [Students Noise Time: 29:25]. Well a composite net will always consist of a number of nets in its entirety. You cannot breakup a net and say that half of this belongs to this and half of this belongs to this. Well this is because we are saying that we are not allowing any dogleg a net will always be put on a single track ok. Now as an example we take a slightly complex example where there are you can say there are 10 nets which are placed like this and these lines show the span of the nets. This is the span of net number 1, this is net number 2, this is net number 3, this is net number 4, this is 5 this is 6, this is 7, 8, 9 and 10. So you see that you have so many nets the 10 nets. Now the first thing is that let us carry out an exercise that from this example, let us try to construct the horizontal constraint graph. You just look at the overlap say from node number 1.

(Refer Slide Time: 30:52)



This is the span. So it overlaps with 2, 3, 4 and 5. So there will be edges to 2 to 3 to 4 and say to 5. So from net 1 these are the edges from net 2 in addition you have edges with 6. So 2 will be having an edge with 3, with 4, with 5 and with 6 also. Net 3 will be having edge with 2. Ok. 2 already you have taken 4, 5 nothing else 4 and 5. Net 4 will be having an edge with 5 and with 7, 8 and 9, 7, 8 and 9, 6 also. Ok. 6 also. So net 4 will be having an edge with 6 also. Similarly 5 we have already taken 6 will be having an edge with 7, 6 and 7. 7 will be having an edge with 8, 9 and 10, 8, 9 and 10. 8 will be having with 9 not with 10 8 and 9 and 9 and 10 ok. So this is the horizontal constraint graph for the problem. Now if we analyze the horizontal constraint graph you will observe one thing that in this graph there are certain maximal complete sub graphs or cliques. For instance this sub graph consists of nodes 1, 2, 3, 4 and 5, this is a clique.

All these nodes are connected to all others. Similarly you see well I think I have missed 1 edge there will be an edge between 4 and 6. So I did not draw that between 4 and drawing, drawing. So you see 2, 4 and 6 this will be another clique so the 1 clique is 1, 2, 3, 4, 5, maximal complete sub graph. See 2, 4 and 6 there is a triangle. Here this is another maximally complete sub graph. There is another maximally complete sub graph consists of 4, 6 and 7. This is another triangle 4, 6 and 7. There is another consists of 4, 7, 8, 9, this 1, 4, 7, 8, 9, and lastly there will be 1, 7, 9 and

10 ok. So for this HCG these are the different cliques or maximal complete sub graphs. Now the idea is that if we have a maximal complete sub graph say 1, 2, 3, 4, 5, then nets 1, 2, 3, 4 and 5 must be placed on different tracks. Well if you decide not to break them ok well. Now the way we are defining zones it comes directly from the horizontal constraint graph representation these cliques indicate the zones ok. So let us see how this is the zones are defined.

(Refer Slide Time: 35:08)

Step 1: Zone Representation ©CET I.I.T. KGP

- 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.

The diagram shows a channel routing problem with five columns labeled 1, 2, 3, 4, and a dash. There are three horizontal lines representing nets. The first net spans columns 1, 2, 3, and 4. The second net spans columns 2, 3, and 4. The third net spans columns 3, 4, and the dash.

Well we have a channel routing problem. There are a number of columns ok. These are the columns which corresponding to which corresponding to the terminal locations say, column 1, column 2, column 3, column 4 and so on. So whenever you look at a particular column for example, let us take that example once again.

(Refer Slide Time: 35:31)

Contd. © CET
I.I.T. KGP

- **Key steps of the algorithm:**
 - a) Zone representation ✓
 - b) Net merging ✓
 - c) Track assignment ✓
- **An example:**

The diagram shows two horizontal tracks with 11 columns each. The top track contains net numbers: 0, 1, 4, 5, 1, 6, 7, 0, 4, 9, 10, 10. The bottom track contains net numbers: 2, 3, 5, 3, 5, 2, 6, 8, 9, 8. Arrows indicate connections between nets in the top track and nets in the bottom track, labeled with numbers 1 through 10. For example, net 1 in the top track connects to net 2 in the bottom track, net 4 connects to net 3, net 5 connects to net 5, net 1 connects to net 3, net 6 connects to net 2, net 7 connects to net 6, net 0 connects to net 6, net 4 connects to net 8, net 9 connects to net 9, and net 10 connects to net 8.

Say you look at this particular column, column 4. Column 4 you see your column number 4 intersects net 1, net 4, net 5, net 3 and net 2, 5 nets. So each column will be intersecting some nets.

(Refer Slide Time: 35:54)

Step 1: Zone Representation © CET
I.I.T. KGP

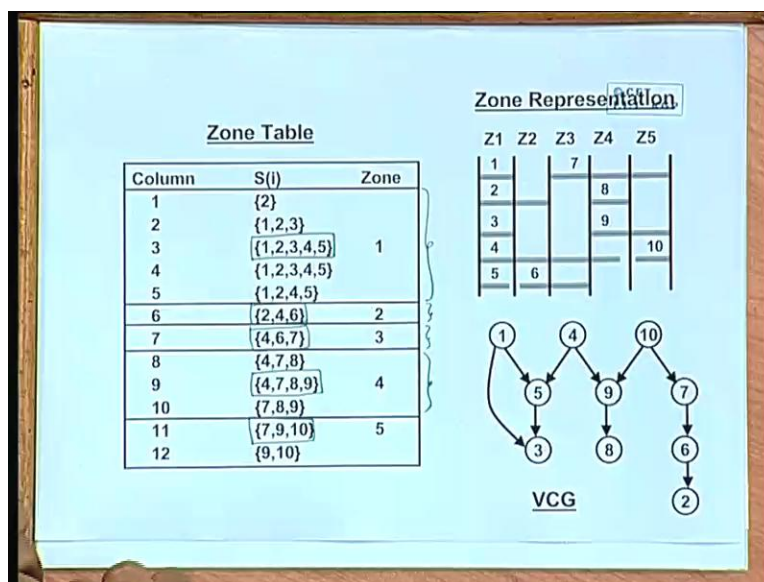
- Let $S(i)$ denote the set of nets whose horizontal segments intersect column i .
- Take only those $S(i)$ which are maximal, that is, not a proper subset of some other $S(j)$.
- Define a zone for each of the maximal sets.
- In terms of HCG / interval graph, a zone corresponds to a maximal clique in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

The diagram shows a grid with 5 columns and 4 rows. The columns are labeled 1, 2, 3, 4, and a dash. A hand is pointing at the grid with a blue pen.

That we denote by the set s_i . This s_i will denote the set of nets whose horizontal segments will intersect column i of the channel ok . Now as we go on computing s_i we will find that well some of the s_i 's can be subset of others. So we take only the maximal s_i 's which are not subset of others. You take only those s_i 's which are maximal not a proper subset of some other s_j . Now if 2 s_i 's are identical you take 1 of them delete other and if only the proper subset delete that proper subset take only the maximal once. So after taking the maximal once whatever remains will be these, this is the idea. Say if you look at this problem once more, say as you scan from the start column 1, 2, column 2, 1 and 3 now sorry 1, 2 and 3, column 3, 1, 4, 5, 3, 2, in this way it goes on. So I will just show this.

Now once we have identified this maximal s_i 's, you call a treat each of them as a zone and assign a number to the zone 1, 2, 3, 4, 5, in that order. Now from the horizontal constraint graph we have identify these cliques and this cliques indicate the zones. There is an alternate representation of horizontal constraint graph we shall show it shortly this is called an interval graph. Now in the HCG as I had just shown a zone corresponds to a maximal clique well a clique is maximal of course. Now let us see what interval graph is and the zone definition for the problem were taken ok.

(Refer Slide Time: 38:14)



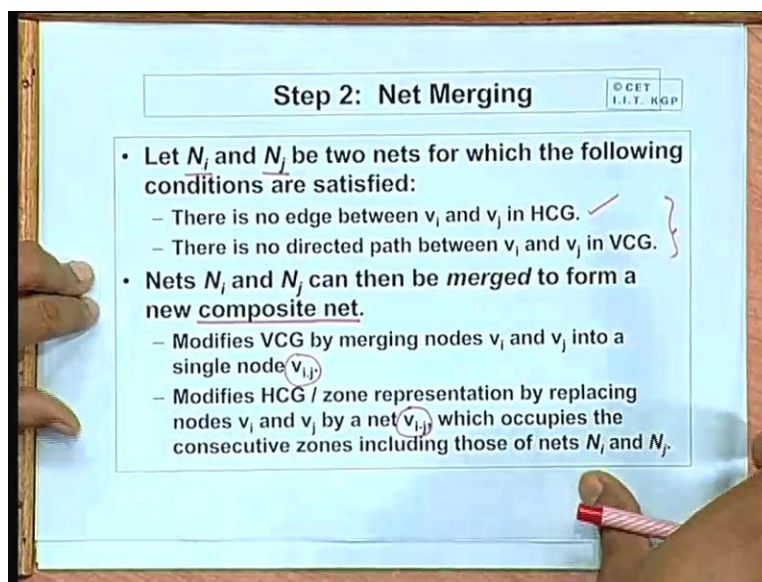
This particular slide will show you that as we go along the columns of the problem for the example we have just taken what are the values of s_i 's. This if you can correlate to that example later but I am just showing this. In column 1, only net 2 is crossing column 2, 1, 2, 3, column 3, 1, 2, 3, 4, 5, in this way 1, 2, 4, 5, column 6, 2, 4, 6, in this way up to column 12 you get. Now once you have this list, you will see that across a set of consecutive columns all the sets will be subset of some set there and you identify that bunch as a zone. This will be your first zone this 1, 2, 3, 4, 5, is the maximal subset out here. Now as we moved from 5 to 6 you will see that a new net is encountering. So there is no subset relationship between this and this. So here a new zone will start. Similarly for column 7, some other 7 is coming in here. So neither of this can be a subset of the other. So this in itself will be a zone similarly this in itself will be a zone. From 8 again another net 8 starts. But 8, 9, 10, well this 8 remains a new net gets added, so it is a subset of this.

So here this is the maximal this is your zone 4. Similarly for the last 1, 7, 9, 10 and then 9, 10. So this is maximal. So there are 5 zones which get defined corresponding to the maximal s_i 's corresponding to the different you can say set of consecutive columns. Now this information you can depict in the form of a zone table this is also called an interval graph. Now in this table you define the zones 1, 2, 3, 4, 5. Now each zone you show which are the nets which belong to that zone. In zone 1 you have nets 1, 2, 3, 4, 5. 1, 2, 3, 4 and 5. In zone 2 you have 2 and 4 and 6 is the new 1. So 2 gets extended 4 gets extended and 6 is a new net which starts. But 1 and 3 does not extend to z 2. Similarly zone 3 you have 4 and 6 but not 2 two will not extend. But 4 and 6 will extend 7 is the new node which starts new net which starts. So in a similar way you can complete this representation. Now this is another way of looking at the HCG and these and the zone information.

So from here you can know that we that in each particular zone which are the nets which are in conflict in terms of the horizontal spans. Zone 1 contains nets 1, 2, 3, 4, 5, this means that 1, 2, 3, 4, 5, these 5 nets must be allocated to 5 different tracks. Zone 2 says 2 4 6 which means 2, 4, 6 must be allocated to 3 different tracks zone by zone, this will give you an idea. The maximal information from each zone and we use that information in taking the decision because anyway

we have decided not to split a net right. Yes [Students Noise Time: 42:21] 2 and 4. Yeah, 2 and 4 gets extended yes. And for the problem the vertical constraint graph is also shown for the same problem. Here as you can see that the maximal path length is 3. So according to the VCG minimum 3 tracks are required of course HCG demands more tracks. But according to VCG minimum 4 tracks are required 10 then 7 then 6 then 2 this is a path of length 4 ok. So the first step is the zone definition this is how we define the zones.

(Refer Slide Time: 43:05)

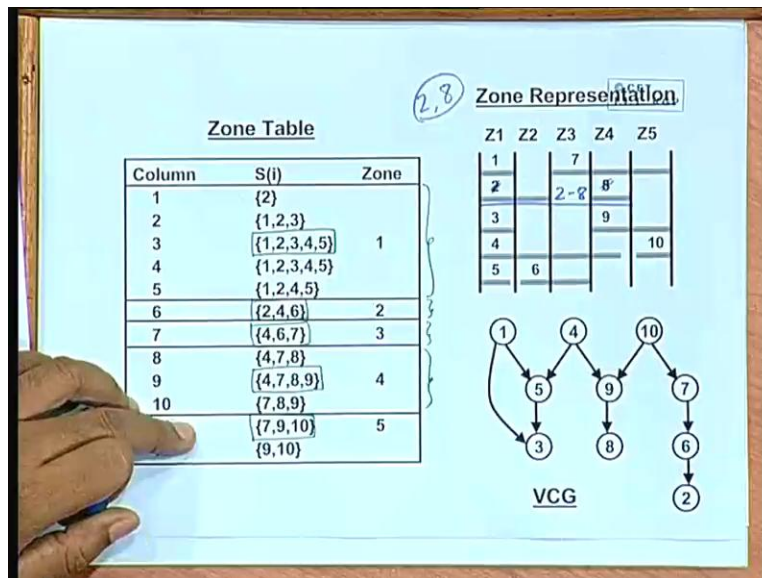


The second step is the merging of the nets. Now after you have defined the zones now systematically we would be proceeding with the merging of the nets. Now let us say how we can merge. Suppose we have a pair nets n_i and n_j , so two nets n_i and n_j can be merged. If these 2 conditions are satisfied, there is no conflict between them in terms of HCG which means they do not belong to the same zone ok. Secondly there is no directed path between them in the VCG. Because in the vertical constraint graph if there is a directed path which means that 1 has to be placed above the other, so there is a constraint.

So if there are no such constraints you can put them merge them. So as to finally you can allot them the same track ok. So if a pair of nets they are satisfied these 2 conditions then you can

merge them together to form a so called composite net. Now once you have merge them you will have to modify the VCG and also the HCG of the zone representation whatever you call. Well I will be showing the VCG step while HCG will also be similar. In the VCG what we do this, the vertices v_i and v_j which corresponded to nets n_i and n_j will now get merged into a single vertex v_{ij} .

(Refer Slide Time: 45:14)



Similarly in terms of the HCG or the zone representation you will be replacing the 2 vertices by a net v_i dash j means i to j . Like what I mean to say is that suppose in this example so lets take this zone representation once more. Suppose in this example we have decided say to merge nets 2 and 8 suppose 2 and 8 are the nets we would be merging. So although there is a gap in between does not matter in the modified zone representation, we will be having a single line these 2 will not be there we will be calling them 2 to 8 ok. So this will represent the composite net which starts here and ends here. This is how we modifying this zone representation ok. Fine.

(Refer Slide Time: 45:58)

Step 2: Net Merging © CET I.I.T. KGP

- 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 then be *merged* to form a new composite net.
 - Modifies VCG by merging nodes v_i and v_j into a single node v_{ij} .
 - Modifies HCG / zone representation by replacing nodes v_i and v_j by a net v_{ij} which occupies the consecutive zones including those of nets N_i and N_j .

So net merging is done for 2 nets which satisfies this condition and this process is iterative.

(Refer Slide Time: 46:06)

Contd. © CET I.I.T. KGP

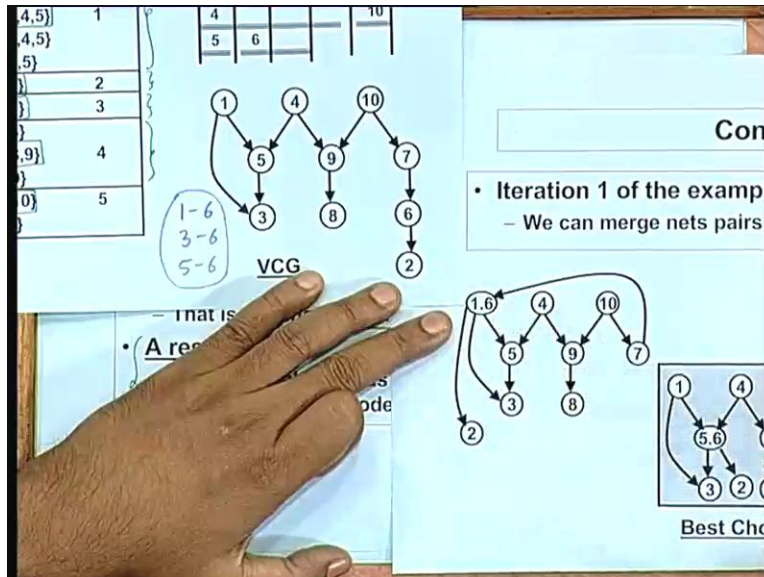
- The process is iterative:
 - Pairs of nodes are successively merged.
 - At every step of the iteration, in case of multiple choices, merge the net-pair that minimizes the length of the longest path in the VCG.
 - That is, the increase in length is minimum.
- A result:
 - If the original VCG has no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

We continue doing it in an iterative fashion where pairs of nodes are successively merged. Well here there is a heuristic. There can be several pairs of nodes which you can merge. But the

heuristic says that you merge that pair of nodes for which the increase in length in the maximal path in the VCG is minimum. Well the ideal case or the ideal scenario will say that you try to minimize the length of the maximum path in VCG. But sometimes you will see that when you merge 2 nodes the length of the path can increase a little bit that you will have to allow. But even if you it increases a little bit you try to take that combination where the increase is the minimum. Because in most practical scenario in VCG the path increase by a little bit can be tolerated because a greater constraint comes from the horizontal constraints like in the example I have shown. The HCG has a clique of length 5 which tells you require minimum 5 tracks. VCG has a path of length 4 which says you require minimum 4 tracks.

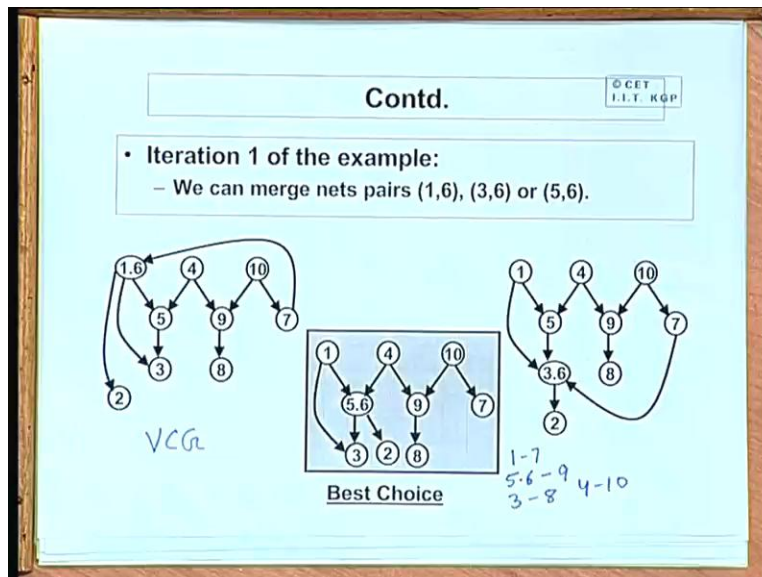
So the you have 1 flexibility you can increase it by 1 without increasing the cost of the solution ok. So if you have multiple choices you merge that net pair. Well that you can say minimizes the length of the longest path in the VCG in fact it can go up in that case you select the 1 for which the increase in length is the minimum I will take an example to show that. But there is an interesting result of course this result is not difficult to just prove it says that if your original constraint graph does not have any cycle. Then even if you merge 2 nodes there cannot be any cycles. Because in the VCG you can merge 2 nodes only if there is no directed path from 1 to the other. So you just think over this problem if you merge 2 such node reacted path there can never be cycles in the resultant VCG right ok. So let us again come back to that example. Let us show the steps of the algorithm ok.

(Refer Slide Time: 48:54)



This was the original VCG just look at the original VCG. Now in terms of these are the horizontal constraints zones and this is the VCG. Now in terms of original constraints the pair of nodes that you can merge here are 1 and 6, I am just showing you 1 and 6 they do not have any constraint horizontally or vertically. 3 or 6, 3 or 6 does not have any horizontal constraints vertically also there is no directed edge and 5 and 6. Similarly, 5 and 6. These are the 3 possibilities we have ok. Now with respect to this graph we can see that how we can merge the node. Let us see. Suppose we have this VCG and we decide to merge 1 and 6, 1 and 6. So if you merge 1 and 6 I am only showing the VCG the VCG will become like this. 1 and 6 will get merged the remaining part remains unaltered but from 6 there was an edge out to 2. So now from 1 dot 6 there will be an edge out to 2. Similarly there was an edge from 7 to 6. Now there is an edge from 7 to this 1 dot 6. So the edge is which are going out of 1 and 6, there will now be going out of 1 dot 6 and the edges which are coming into 1 or 6 will now be coming into this composite node right.

(Refer Slide Time: 50:54)

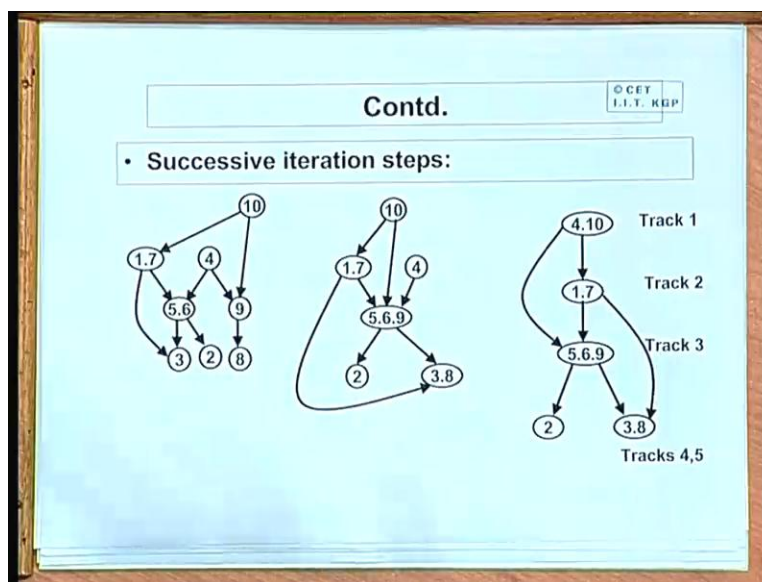


So, in this slide I am showing all the 3 possibilities. Well if you merge 1, 6 this is the resultant VCG well I am showing the VCG ok these are all VCGs. If you merge 3 and 6 this is the resultant VCG and if you res and if you combine 5 and 6 this is the result. Well now we have to take a decision out of these 3 which is the best. Now evaluate with respect to the longest path again in this 1 the longest path will be starting from 10, 1, 2, 3, 4, 4 means there will be a 5 nodes 4 edges. Here again starting from 10; 1, 2, 3, yeah 3 it comprises of 4 nodes 3 edges. Here only 2 edges and 3 nodes. So in terms of our heuristic in terms of our heuristic this is the best choice. So in the first step of the iteration will be merging nodes 5 and 6 ok. Now after merging your VCG gets modified to this your zone representation also gets modified by merging this 5 6 into a composite net. So you repeat this process.

\Now in the second step you will see that you can merge well you can merge 1 and 7 you can merge 5, 6 with 9, you can merge 3 and 8. Well I am not showing all the possibilities now. Here from here you can merge 1 and 7 you can merge this composite net 5, 6 with 9 you can merge 3 with 8. Well in fact you can also merge 4 with 10. There are 4 possibilities. [Students Noise Time: 53:04] Yes. You will have to also to check horizontal I am not showing this. But you will also have to continuously look at this graph after merging what is the condition whether there are

any constraint there are also. Yes [Students Noise Time: 53:15] Ok. [Students Noise Time: 53:22] Yes, yes. [Students Noise Time: 53:26] Merging with con yes, yes. So once we have decided to merge two nets the entire span gets reserved. So now you cannot put anything else in between its true. Well now again you do these things all this 4 possibilities you try to merges and say which is the best. Well I am only showing you the best paths now. So the next step we will find that merging 1 and 7 will give you the best alternative. So I am showing the steps here.

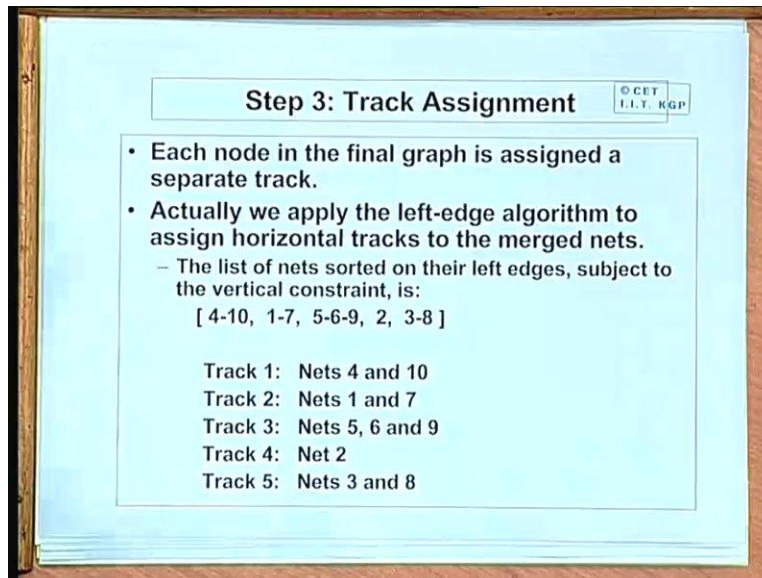
(Refer Slide Time: 53:54)



So in the second step 1 and 7 gets merged. This is the modified VCG. Well here actually in the between I have skipped 1 step. So actually after this step we will find that 5, 6, 9 or 3, 8, they give you the best cost. So you can apply 1 after the other. So I have applied both of them 1 by 1 and this is the resultant VCG. Where 3 and 8 have been merged and also 5, 6 and 9 this is the VCG. Now again in the next step you will see that 4 and 10 can be merged. So after you merge 4 and 10 these will be the resultant graph and here you cannot merge anything any further. There will be constraint coming either vertically or horizontally there is some vertical constraint and between 2 and 3, 8 there is a horizontal constraint these 2 also you cannot merge. So once you have arrived at a graph where you cannot merge the nodes any further. Ok. There you have you have arrived at a possibility by each of this composite nodes can be allocated to a separate track

that is your and based on these arrows we apply a left edge algorithm to allocate the tracks. First 4, 10 then 1, 7 then 5, 6, 9, then either this or this any order.

(Refer Slide Time: 55:36)



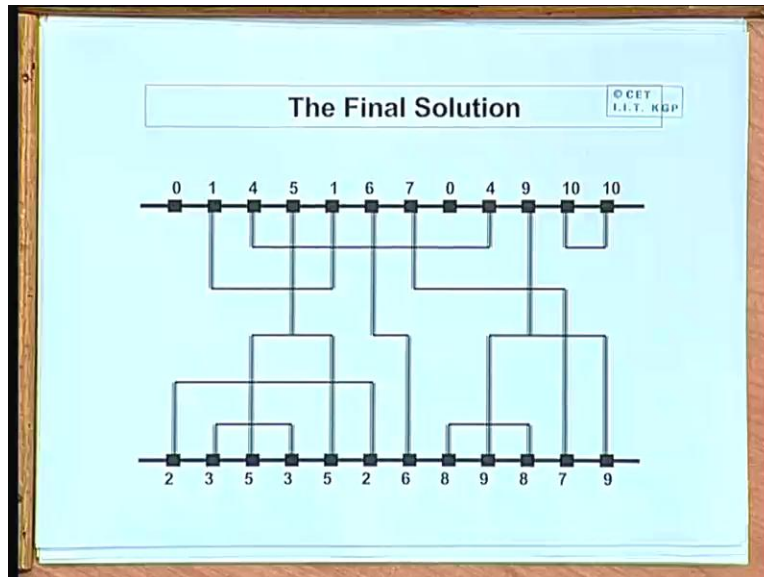
Step 3: Track Assignment CET
I.I.T. KGP

- 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

So the third step is the track assignment, so each node is assigned a track. So actually we apply the left edge algorithm subject to the constraints. This is the ordering and in terms of the tracks just following the graph you allocate nets in this order.

(Refer Slide Time: 55:56)



So I am showing the final solution this gives the final solution. Where in track 1, 4 and 10 are assigned track 2, 1, 7, track 3, 5, 6, 9, track 4, 2, only 2 track 5, 3 and 8 this will be a final solution. So the idea of this algorithm is this. You take your HCG, VCG and you constantly go on merging nodes. Based on some criteria and finally apply the simple left edge algorithm to assign the composite nets to tracks. But in our next lecture we will be looking at an algorithm which can finally take care of cycles in the VCG. We have not yet looked at any scheme where the cycles can be added ok. So this we shall be discussing in next class.