VLSI Physical Design with Timing Analysis

Dr. Bishnu Prasad Das

Department of Electronics and Communication Engineering

Indian Institute of Technology, Roorkee

Week - 08

Lecture 42

Full-Netlist Routing

Welcome to the course on VLSI Physical Design with Timing Analysis. In this lecture, we will discuss about how we can do full netlist routing in VLSI physical design. So, the content of this lecture includes how we can do the global routing using integer linear programming which is most popular in industry ILP. We will discuss about it, its formulation and with an example. And integer linear program is very useful for global routing formulation. We will also discuss one example how we can formulate a global routing problem for ILP. So, in this case, we are basically finding a linear program for a set of constraint and objective function. So, if you have any kind of basically optimization algorithm, so you have two things, one is basically objective function and second one is set of constraints. So, here our problem is basically linear in nature. So, it is a linear programming. Linear means basically the degree or power of all the variables is 1. So, here we can have this objective function can be maximized or minimized based on our requirement. So, whenever I have a objective function, it can be maximized or minimized. For example, if I want to basically decrease the wire length, I need to minimize the objective. I need my estimated wire length should be minimized.

So, in that case, this optimization problem is a minimization problem. So, this optimization problem becomes minimization problem. Optimization problem becomes minimization problem whenever I am reducing the wire length, but I want to increase the speed of my design. So, in that case, the speed should be maximized. So, in that case, the optimization problem becomes the maximization problem.

In that case, optimization problem becomes the maximization problem. So, this is the case 1 where we are considering the wire length reduction. So, that is a minimization problem. The case 2 where I am basically improving the speed of a design, so that is a maximization problem. So, the constraint in this case, your constraints and objective because it is a linear program, both are linear in nature.

So, basically the power of the variable should be 1, maximum 1. The constraint form a system of linear equation and inequalities. We will have constraints which will form a system of linear equation and inequalities. The integer linear programming is a linear program where every variable can only assume integer value. So, we have linear programming where the conventional linear programming, the variables can have any kind of floating point numbers, any kind of fractional numbers, but in integer linear programming, each variable should assume integer value.

Integer linear programming, the solution of the program, each of the variable will have integer value. So, here we have inputs, what is inputs to our integer linear programming algorithm? There are three inputs, one is the W cross H of the routing grid, the width and height of the routing grid. Second one is the routing age capacity, the capacity of the interconnects, basically routing tracks. Then we have the third one is the netlist given to us to the ILP program. Then we have two sets of variable.

The first set is a set of k Boolean variables. We have k Boolean variables, x net 1 to x net k for each of which we have a basically we specify for k specific path or route options for each net. For a net we have multiple routing options are there. For a net connection between one pin to the other pin, we have multiple path is there. We are defining each of the path by x net.

So, then we have to choose which one is basically optimal in nature. Then we have basically the second set contains k real variable W net 1, W net 2 to W net k for each of which represents a net weight for a specific route option. So, for any net we have a basically weight or the priority we need to assign how much priority we can give to that net that is given in the second set of k real variables. So, these two variables need to be given as input. So, we have two constraints. The first constraint each net must have a single route. So, since multiple routes are possible only each net can have single route and second to prevent overflow the number of route assigned to each net cannot exceed its capacity and it also should not exceed its capacity. So, what is input to our ILP program? So, we have W, H, W stands for width and H stands for height of the routing area, total routing area G. So, then we have basically G i, j is a grid, one of the grid in that layout which location is i, j in the routing grid G. So, if I can say G i, j so the i stands moving in x direction and the j is basically y direction.

Now so we are defining the grid. Then we are seeing whether I have a routing track available in horizontal direction or not. So, the horizontal capacity, so if I have a horizontal capacity, so I have a grid 1, I have a grid 2, if I have a edge between them, so then the G this grid is basically let us say G i, j and this grid will be G i plus 1, j. So, in this grid, in this basically neighboring grid or adjacent grid whether I have a horizontal edge is there or not. Similarly we can do the same thing, let us say I have a node here, I have a node here, if I have a line between them, so this is let us say my G i, j and this will be G i, j plus 1. So, now it is moving in the vertical direction. So, if that is the case whether there is an edge between them or not. So, that is basically vertical edge. So, we have a netlist is also provided here, we have a netlist is also provided as input. Now this x net 1 to x net k are k Boolean path variable for each net.

So, if I have a net, it has multiple possible connection can be possible, each one of them can be denoted by x net 1 to x net k. Means let us say I have two points are there, it can be connected in multiple ways, so each one of them is denoted by x net 1 and all the possible combinations. Similarly the weights assigned to those nets is denoted by w net 1, w net k is a net weight. Now what is our aim is to maximize the weight such that I can do this optimization. So, here what is happening is that I have a weight assigned to this one, I have a weight assigned to each one of them.

So, what we are doing here if there is a complicated shape is there, then we are giving less priority to that net. If I have a L shape, then we have assigning more priority to that net. So, that need to be maximized. So, here if you can see here, I have some variable. So, this x net 1 to x net k will have 0 and 1. So, either any one of them, the possible solution for each of the nets are either 0 or 1. And there is another constraint is there. So, the nets should be maximum value of net is 1. So, this is variable range and this is the net constraint. So, we have a net constraint for each of the grid. So, it is basically saying the capacity. So, this is basically defining the capacity in vertical direction because here j is changing from j to j plus 1. So, similarly here it is saying the net constraint in the horizontal direction because your i is changing from i to i plus 1. So, this is for capacity in horizontal direction. Now so we have to satisfy those net constraints.

We need to satisfy those net constraints. Now we will take an example to understand how this can be possible. So, we have some nets are given. What are the nets given? Nets are A, B and C. So, this A should be connected to this A and B should be connected to this B. I can change the color. This can connected to this one. Then finally your C should connect to this one such that there should not be any overlap. So, what is the size of your routing grid actually? Your width is 5 here and height is 4 here for the routing grid. And sigma of E is basically 1 for all sigma belongs to each one of the grid will have one value. So, we have another constraint here.

The L-shape routing will have a weight of 1. Then the Z-shape routing has a weight of 0.99 and the lower left corner is basically this is 0, 0 point. So, this is called lower left LLX, LLY. So, LLX is 0 and LLY is 0. Lower left X, lower left Y. Now our task is to write the basically objective function and constraint for the ILP solver such that the nets of the graph can be routed. So, we are basically writing the objective function and constraint which can be given as input to the ILP solver. So, we write the ILP to route the nets in the graph. So, let us we will consider one net at a time.

So, for a net A the possible routes are two possible routes are there A1, A2 in L-shape. So, let us say this is your A1, this is L-shape and this is A2, this is L-shape actually and we have a Z-shape, we have A3 is your this Z-shape, this is the A3 and the dotted one is your A4. So, what is the constraint here is that we have four possibilities there. So, one of them will be 1 and because I need only one interconnection between the net A2 this to this. So, one of them should be 1 and rest will be 0. So, my constraint is basically XA1, XA2 plus XA3 plus XA4 should be less than equals to 1. Now we have variable constraints. So, we have variable, four variables. So, maximum value of the variable is 1 and minimum value of the variable is 0. So, XA1 lie between 0 to 1. Similarly XA2 lie between 0 to 1. XA3 lie between 0 to 1. XA4 lie between 0 to 1. So, these are the variable constraints. These are the basically variable constraints because one of the route is 1 rest will be 0.

So, this is variable constraint and this is basically your net constraint. Now we will do the same thing for all nets. Let us say I have B1 is here that should be connected to this one. So, B1 has basically L shape two possibilities there and Z shape one possibility is there. So, we have three possibilities are there. Let us say this is 1, this is 2 and this is 3. So, what we are doing here is that we are writing XB1, XB2, XB3 is less than equals to 1. Any one of them will be 1 and others will be 0. So, we are analyzing each of the net independently. We are not looking into the other nets at that time. But when the ILP solver will be used, it will check that if I go by this route whether there is a sort with the other net or not. So, that is the job of the ILP solver. The ILP solver will check if there is a sort is there or if there is a basically if there is any kind of violation is there those things will be checked. So, similarly for C net I have this is 1, this is 2, this is 3 and this is 4, 4 possibilities are there. So, I have this is the net constant for C and this is the variable constant for C. Now we will look into this horizontal edge capacity constraints. So, this horizontal edge capacity constraint will look one few example because all of them are given. I will check some of them how this is created. So, we have this capacity if you can see here capacity constraint is there.

So, if I can see these are my grids. So, if I have let us say this is my if I go to this diagram. So, this will be 0, 0 and this will be 1, 0. This will be 2, 0. Y axis is 0. This is 3, 0. Now this will be 0, 1. This will be 0, 2. Now this will be your x coordinate will be same and y coordinate will be increased by 1. So, here it will be 2, x coordinate is 2 and y coordinate is 1. So, this will be x coordinate is 3, y coordinate is 1. Now this will be basically 1, 2. So, now this is a grid. This grid is there. Then what we are going is that we will see that what is the constraint between these two in the x direction. If it is 0, 0, 1, 0 what is the constraint in the x direction? If I can see here there is no constraint in this case. There is no constraint for A net. There is no constraint for B net. There is a constraint for C net actually. So, what are the nets are constant here? This net basically if I can see here this net is going through these two grid actually. So, this is 0, 0. This is basically 1, 0. So, here this is 0, 0 and this is 1, 0. So, in this case what is happening is that what are the nets going through this grid actually? So, my x c 1 is going and my x c 3 is going. So, in this case in the horizontal direction either this x c 1 should go or x c 3 should go, not both. So, if you can see here in this grid I have x c 1 plus x c 3.

So, that should be less than equals to 1. Either x c 1 should go or x c 3 should go. So, this is the obvious case because if I connect C to C either one of them should go. That is a very simpler case. But let us say if I go to this one. This is a complex case because here three nets are going. G 1 1, G 2 1. If I go to this one G 1 1. So, if I go to the G 1 1, G 2 1. So, this point if you can see. If I have this point, so what are the nets going through 1, 1 and 2, 1. So, if I can see this one, first I am drawing the boundary then we can check the nets. In between what are the nets going? In between what are the nets going? So, if I can see here my a 2 is going, a 3 is going. So, here if I look into the horizontal direction, so here it will come x a 2. So, then you have x in the horizontal direction if I can see this line is going and also this line is also going. So, here if you can see x a 2 and x a 3. So, these two is coming in the same grid. So, if I go to b 2 there is nothing there. But if I go to c net, I have which net is going in this grid actually if I can see here in the horizontal direction. So, here if you can see basically it should from here only this x c 4 net will come. Either this two or x c 4 net will go in this area because I am concentrating on this area, because I am concentrating in this area.

So, x a 2 x a 3 plus x c 4, x a 2 plus x a 3 plus x c 4 should be less than equals to my channel capacity in x direction. So, here if you can see x a 2 plus x a 3 plus x c 4. So, similarly I can do the constraint for vertical edge capacity constraint. So, if you can look into this vertical edge constraint graph. So, let us say I am concentrating on this G 0 0 to G 0 1. So, if I go here, so we are concentrating on this area. In this area I am doing in a different color. So, now I am concentrating on vertical constraints. So, vertical constraint means I am looking into 0 0 and this is x coordinate is 0 and y coordinate is 1. So, this area, what are the nets going? So, what are the nets going in this area? So, if I look into here 0 0, 0 0 and 0 1, so the nets going here is basically a 2. So, if I can look into this, in this area, there is nothing is there. In this area there is no routing there. So, in that area there is no routing is there. So, there is no constraint for the a net. So, if I can see here in this area there is no routing is there 0 0 and 0 1.

So, in this area there is no routing is there 0 0, 0 1. So, from b net also there is no constraint is there in this case. But if I go to this c net, I have constraint. So, what are the nets having constraint in this area? is basically the net c2 x c2 plus my x c4, x c4. It is x c2 plus x c4 should be less than equals to 1. So, if you can see here x c2 plus x c4 is less than equals to 1. So, this process will be repeated for all the grids, for all the vertical grids. So, we have basically this horizontal edge capacity constraint. This is one important point. This vertical edge capacity constraint these are also important for the vertical capacity.

So, then we have an objective function. So, this objective function whatever we have assumed is that the L-shape has priority or the weight is 1 and the Z-shape has weight basically w of net is 0.99. If you can see here this a1 a2 x a1 x a2 is a L-shape structure. So, these are L-shape structure L-shape and this b1 b2 is L-shape and c1 c2 are L-shape. Then x a3 x a4 are Z-shape. Similarly you have this x b3 is Z-shape. This is Z-shape and this is also Z-shape. So, whenever I have a Z-shape structure I will multiply that net with 0.99 and whenever I have a L-shape structure I multiply that net with 1. So, what does it mean? What is the requirement of that? It tells the tool that we should avoid Z-shape structure which is not good for routing. However if there is no routing possible we can give a net a Z-shape to complete the routing. So, usually the priority of L-shape structure as higher compared to Z-shape structure. So, this is the main objective function which need to be maximized. So, then we have another formulation which is also popular and it is very useful for global routing which is called rip-off and reroute. This is called rip-off and reroute. This is basically this framework is focused on when there is a problematic net whenever there is any kind of violation those kind of nets this kind of solution is very useful.

So, the problematic net means that some of the nets which is not routable. So, if some of the nets are not routable because of some kind of constraint in the capacity, constraint in the net or physical obstacles. So, in those cases basically we need to use this kind of algorithm rip-off and reroute RRR algorithm. So, the key idea behind this algorithm is that we need to allow temporary violation. Even if there is a violation in some net still will allow that net to be routed so that then all the nets are routed. But what we have to do? We have to remove the nets then iteratively we will remove the sum of the nets which can have less impact on the overall interconnect length.

So, we will route the nets with violation and we will check that which net has basically less routing impact the length of the interconnect is modified by lesser amount those nets would be removed and route them so those nets will be removed and those can be routed again. So, it will use to decrease the number of violations. Let us take an example here of RRR algorithm. So, what is this rip-off and reroute example here is given that this is the initial basically layout we have four nets are there A, B, C and D and the possible routing in this case is this one and if you can see here the total wavelength in this case is 21. So, in this case we basically we do not allow violations. So, there is no violations are allowed in the initial case. So, case one here we routed the net without violation. So, and we found that the total wavelength is 21 and if you can see here what is the main drawback of this routing is that D has a bigger interconnect distance. It is going like this, it is going like this, it is going like this, like this. But what is happening is that we want basically better solution because this detour of the D net will give you more interconnect length.

So, what is another solution is that we will route both D and B together. So, there will be some kind of violation is there D and B will be routed together. But we can say that if I allow the D to be routed from here to here and detour the B the impact is less. Then what we are doing we are deterring the B earlier the shortest distance for B is this much now the distance of B is increased by some amount. However the net D is reduced by large amount the overall net length is reduced. So, in this approach what is the procedure is that we allow the routing with violation. So, the case 2 the routing with violation because we have D and B will be having shorts but what we can see that if we deter the B our total well length is reduced. Now the well length is 19. So, this is a better solution compared to our previous solution. Hence this rip-off and reroute algorithm is very useful for nets with violations and it improves the total basically wire length of the design. So, in this lecture we discussed about integer linear programming and we also discussed about the rip-off and reroute algorithm. These are very useful for global routing phase of the VLSI physical design.

Thank you for your attention.