

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

**Lecture No #29**  
**Backend Design: Part – XV**

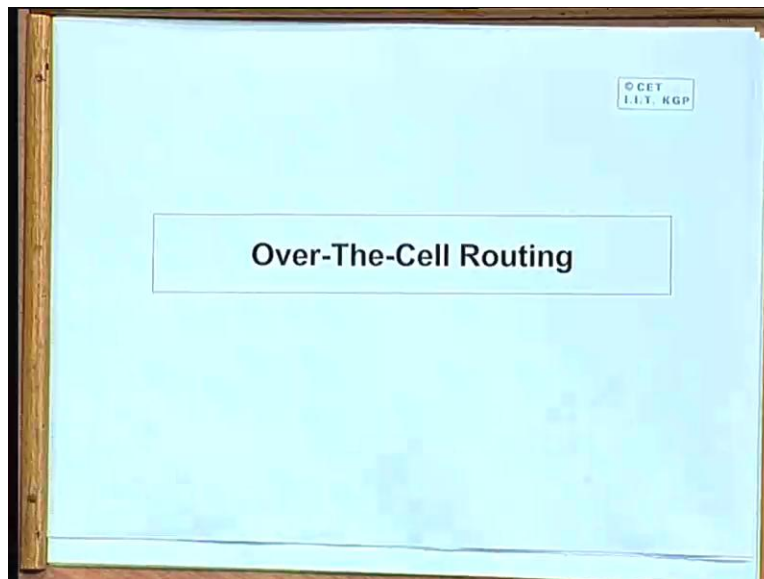
In this class we shall be talking mainly about two things. I told you there is something called over the cell routing which is applicable to standard cell based designs.

(Refer Slide Time: 01:00)



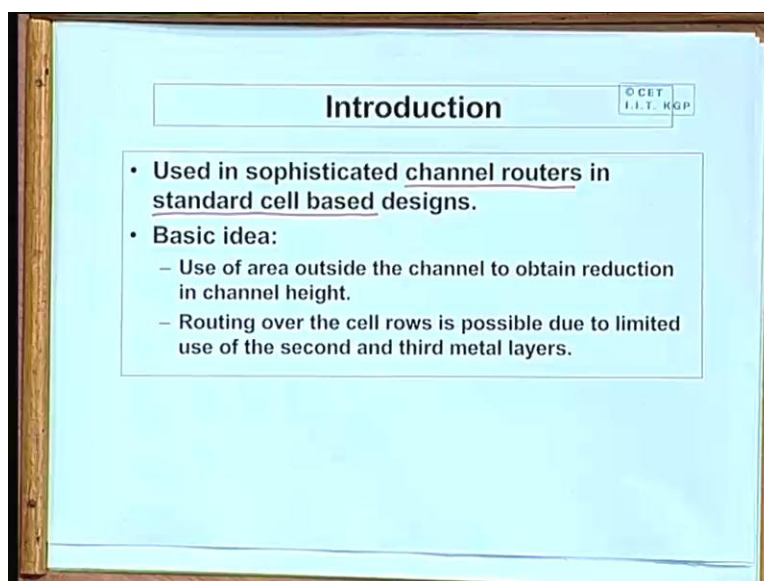
We would be talking about the basic idea of the over the cell routing and how people normally do it. And after that we shall also be talking about layout compaction which is typically a post processing step after the routing is completed.

(Refer Slide Time: 01:20)



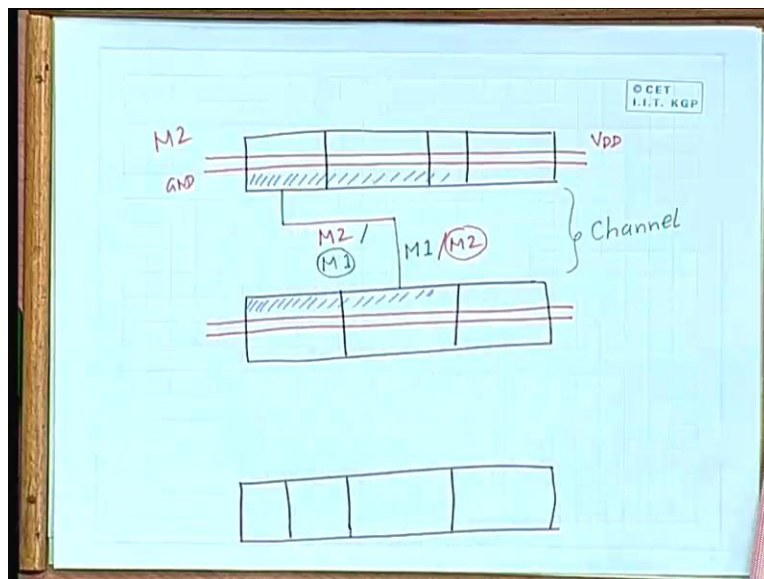
So let us start with our discussion on the over the cell routing. First let us understand what this over the cell routing problem is then we shall try to see how well we can go about solving this problem.

(Refer Slide Time: 01:36)



Now this over the cell routing there are two things we should remember one is that this is used in channel routers specifically in standard cell based designs. Because this concept of channel also arises in general custom based ASIC's but here we would be concentrating only on designs which are based on standard cells. The concept is that in a standard cell based design we typically use the area between the cells to complete the inter connection. But in over the cell routing what we do is that, we use some of this space which is above the cell area. So we use some area outside the channel outside the channel means over the cell rows. This, we are use this we are using for routing in order to reduce the total height of the channel and in terms of the number of tracks you need. Now routing over the cell this becomes feasible because we have very limited use of the second and third metal layers. Well I am trying to explain what this means.

(Refer Slide Time: 02:53)



Well in a standard cell based designs as you know that the cell rows are placed like this. So in a row you can have several cells placed which can have varying width but the same height. Now normally this space which is between the rows this one this is called the channel. This channel is used for interconnection. But in over the cell routing we use a slightly different concept. See here inside the channel what happens is that well we have tracks or we have interconnecting nets

which are running like this on the tracks. So as we had mentioned the vertical and the horizontal layers typically run on two different metal layers.

Usually the vertical layers run on the first metal layer which sometime is called m1 and the horizontal segments sometimes run on the second metal layer or vice versa. This can be m2, this can be m1. For standard cell based routing typically the third layer is not required. Well if you use a channel router which can take care of three layers we can provide some reaction in channel height. Now the idea here is that well with respect to the cell rows the  $V_{DD}$  and the ground they are routed through horizontal layers which are located at exact coordinate with respect to their heights.

Typically  $V_{DD}$  and ground lines will run in parallel like this. This is one possible approach other approaches are also there.  $V_{DD}$  and ground lines are running like this. Well, let us take a specific case. Suppose we say that well the horizontal lines are running on m1 the vertical lines are running on m2 and the  $V_{DD}$  and ground. This is the  $V_{DD}$  and this is ground these are also running on m2. So you see that there is some space. Below this  $V_{DD}$  and ground lines here I am referring to the space here and also the space out here where the second metal layer m2 is unused. So this space we are not using.

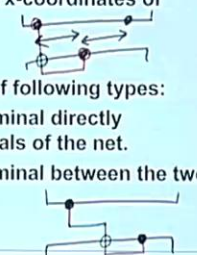
So if required we can utilize this space to complete some interconnection not through the channel but in the region over the cells either here or here, this is the basic concept behind the over the cell routing. Because in the area over the cell the second metal layer typically m2, m2 is not used because  $V_{DD}$  and ground lines are running parallelly in the horizontal direction. So here there is no connection on m2 fine. Now over the cell routing problem has been tackled by much research as in fact there are many algorithms I am referring to one simple approach which people have tried and in fact this approach works pretty well.

(Refer Slide Time: 06:44)

© C E T  
I. I. T. R G P

### Basic Steps in OTC Routing

- **Step 1: Net decomposition**
  - Each multi-terminal net is partitioned into a set of 2-terminal nets (defined based on x-coordinates of their left ends).
- **Step 2: Net classification**
  - Each net is classified into one of following types:
    - **Type 1:** There is a vacant terminal directly opposite to one of the terminals of the net.
    - **Type 2:** There is a vacant terminal between the two terminals of the net.
    - **Type 3:** None of the above.



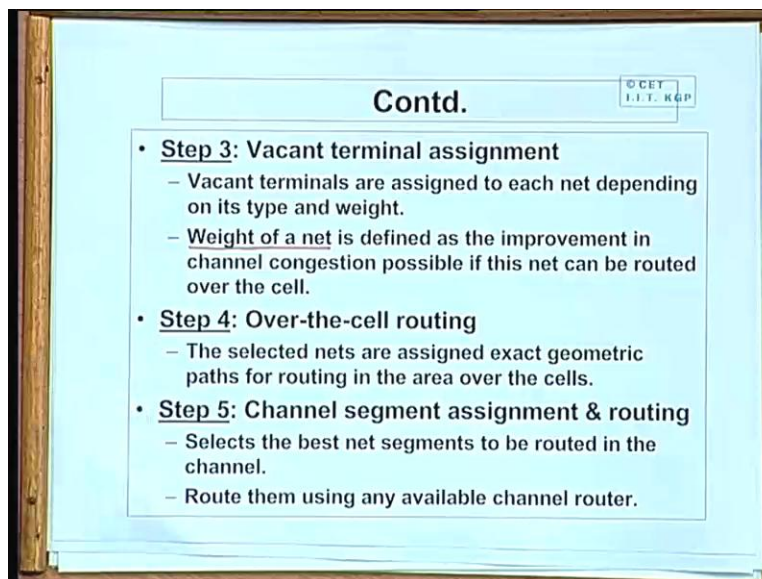
This is a very simple approach in a sense. First step is that well if we have multi terminal nets you break it up into two terminal nets. Now this breaking up is done with respect to the x coordinate of the left ends. Suppose you have a net this is a channel, suppose you have a net where one terminal is here one terminal is here one terminal is here. This you breakup into one net spanning from here to here and another net spanning from here to here based on the x coordinates you do this. So after doing this breaking up, so each net is now a two terminal net.

Now you classify the nets which are existing in terms of whether there are vacant terminals available through which you can route it over the cells. So we will explain when we will just understand when we take an example. Nets can be classified into three types. Type 1 says that there is a vacant terminal directly opposite to the one terminal of the net. For example in this first case when you are considering, this net connecting this point to this point, well it is possible that the terminal point directly opposite to this here. This is a no connection this is a vacant terminal. So if the need arises we can take a line through this over the cell down here. So if the need arises we can utilize this.

So type 1 is desirable in terms of over the cell routability. Similarly type 2 says that well the vacant terminal is not directly opposite, but it is somewhere in between. Somewhere in between also can be advantageous like we have a scenario like this. Say we have a terminal you have you have to connect this point. With this point and there is a vacant terminal say out here. So you can have a connection like this. For instance you have a vertical segment like this, a horizontal segment like this and the remaining connection you can utilize this over the cell.

Now you recall, you just, you notice one thing in the region over the cell. We are not switching between layers because it is difficult to have a wire connection over the cell. So we are doing the entire connection on a single layer of metal. Type 3 is the most difficult type which is none of the above and type 3 nets are most likely. This will be best routed using the tracks in the channel itself. But if we have either type 1 or type 2 you can try to use the region over the cells to minimize the number of tracks required. So you classify the nets depending on the availability of the vacant terminals then you assign the vacant terminals of the nets.

(Refer Slide Time: 10:08)



**Contd.** © CET I.I.T. RGP

- **Step 3: Vacant terminal assignment**
  - Vacant terminals are assigned to each net depending on its type and weight.
  - Weight of a net is defined as the improvement in channel congestion possible if this net can be routed over the cell.
- **Step 4: Over-the-cell routing**
  - The selected nets are assigned exact geometric paths for routing in the area over the cells.
- **Step 5: Channel segment assignment & routing**
  - Selects the best net segments to be routed in the channel.
  - Route them using any available channel router.

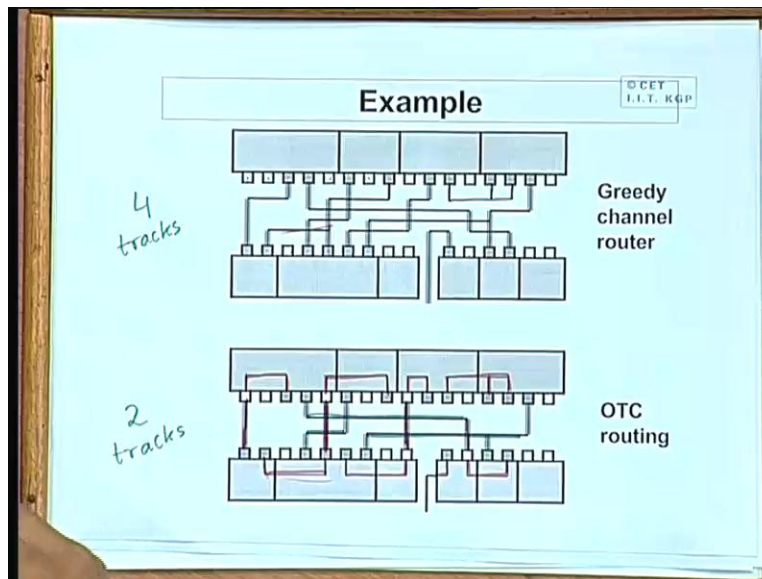
Some of the nets will be using some vacant terminals you make that assignment. Well I am not going into details of this. There is there are some heuristics whereby you can give priority to the

nets. Some of the nets which otherwise lead to congestion in the channel you try to give them higher priority. See if you can route them over the cells so as to reduce the congestion. So you can identify your define some kind of a weight of a net depending on what kind of channel congestion it is creating in a conventional you can say route provided by a channel router. So after you complete this, well some of the nets you are utilizing the vacant terminals for connecting. I am not going into the detail of this.

Well but a conceptually you are doing this. And after you do this assignment some of the net segments you have to route over the cell you complete that here. Now when you are doing this over the cell routing you have to remember or means you will have to have the information that how many tracks you can use over the cell in each direction. Typically it is 1 or 2 not more than that. Because the space is limited. Yes [Students Noise Time: 11:29] Step three says that you have the terminals which you want to connect there are some vacant terminals where there is no connection. So some of the net segments it maybe advantageous to take it through those vacant terminals.

So as to reduce the tracks in the channel. So you try to assign some weights to the nets in terms of the amount of congestion it is creating. So try to give priority to those nets which create more congestion and try to assign it to a vacant terminal first so that part of the net can be routed over the cell ok. And finally step five well the over the cell routing was done this stuff which is left we use a convention channel router to complete it. Well I am taking an example because I am not, I have not gone through the details of the algorithm. Just the broadly the steps I have to, I have talked about.

(Refer Slide Time: 12:28)



So I have an example here. So if you look at this example, you will understand what this means. Well I have only shown one channel in a standard cell we are designed there are two cell rows with channel in between. Well I have not numbered the terminals. But you can understand the connections by seeing the tracks. This is a solution which is obtained using a greedy channel router. Now if you check a greedy channel router requires four tracks 1, 2, 3 and 4, so here you need four tracks. But you see that these white blocks are the vacant terminals there are so many vacant terminals. So there are some nets which are of type 1 there is a vacant terminal just opposite to it. There are some nets which are of type 2 like this net there are some terminals in between some are type 3.

So after going through all the steps, I am showing you the final solution this is a solution which you get using OTC routing over the cell routing which requires only two tracks. Now you look at the vertical segments. Say this net was like this. So now this entire net you take on the second metal layer like this. There is no horizontal segment in the channel. Well so here you are taking advantage of the fact that this is type 1 net. Similarly there are some other connections which are going like this. This is also a type 1 net there is a vacant terminal opposite to one of the terminals. This you take like this. So here the connection was like using doglegging you did like



this. But here there are two segments of the net which are going over the cell and in the channel you require only one vertical segment nothing else.

Similarly other line like this, this is going. Similarly there is well here there is an example where there are two vacant terminals one opposite to the other. You are utilizing that to take the line from this side to the other. Well here I have not shown one connection there are three nets like this. These are all connected. So here you make connections like this. Similarly this connection will go like this. This connection will go like this. So this is a solution you will get using over the cell routing. Here I am assuming that only one row or track you can use on either side over the cell and if you can use it judiciously. As you can see that the reduction in the track can be quite significant many of the nets you can route over the net. [Students Noise Time: 15:39] Yes. [Students Noise Time: 15:41]

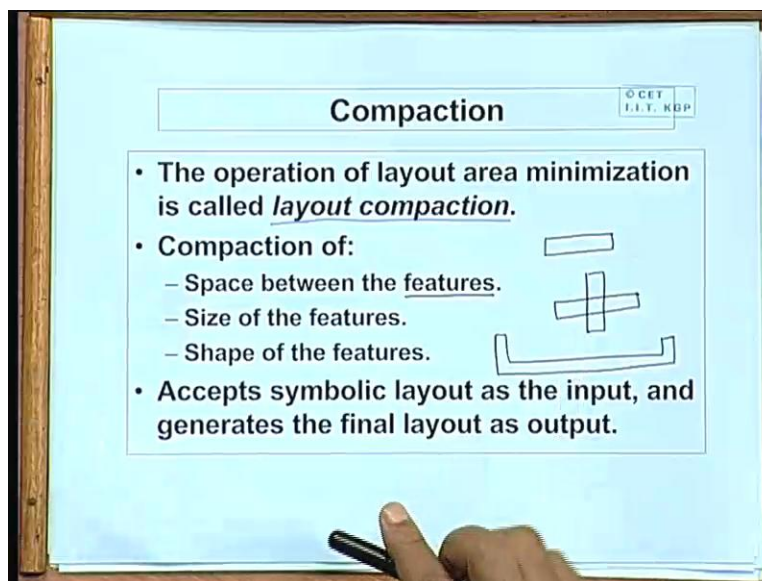
No horizontal segments let me show it in a different color. These are [Students Noise Time: 15:53] over the cell or in the same layer, this is in the same layer. Because over the cell there is already some devices laid out. You cannot have a wire connection in any place you want that can interfere with the other layers which are running below it. So all connection over the layer over the cell will be planer connections. There will be no switching between those ok. So this OTC over the cell routing is also a type of layout compaction. After you have, we have obtained a problem of channel routing. You use this to reduce the number of tracks required some kind of compaction. But in general after completing any kind of a design in terms of the layout you can go for layout compaction in general.

(Refer Slide Time: 16:49)



Because layout compaction is a general step using which you can obtain significant reduction in the layout area it can be as high as up to 20 to 30 percent.

(Refer Slide Time: 17:06)



So when you say compaction we are actually talking about layout area minimization or layout compaction. Layout compaction. So we are saying that we are completed the layout. We have these cells in place we have the interconnection all done so after that we try to make some compaction in the layout. Now when you say compaction we are talking about compacting the space between features. I will explain what features is size of the features and in some cases also shape of the features. See here we are talking at the layout level. So at the layout level we do not have gates on the high level compound. We have only the layers and their interconnections diffusion poly silicon metal. So at the level of layout, we have some feature these are called features which are all rectangular in shape. You can have a feature like this.

This can actually represent a segment in the poly silicon layer. There can be another feature like this. This can be a segment in the diffusion layer. There can be two intersecting features one on top of the other. Well even an interconnection line this can also be treated as a feature because it will also have a finite width. So at the level of layout everything reduces to rectangular blocks and the relative position they are all geometrical shapes. They are all typically rectangle nature well means we are ignoring lines or tracks which run at any other angle other than 90 degree than ((word not clear: 18:54)). There are some routers which also they can also handle 45 degree lines. But when you say features we represent these individual rectangles which can run on diffusion poly silicon or metal layers.

Now there are some conservative design rules which if followed will give you a correct design. Like for instance when there are two polysilicon lines which represent signals which have been carried with respect to two different nets they are running in parallel. There you will be having two different kinds of constraints one is the minimum width of the lines that can run and also the minimum separation of the lines. Now these width and separations are typically measured or countered in terms of the basic feature size which represented by lambda. Lambda is well roughly speaking; lambda is half the channel length of a transistor channel of a transistor, MOS transistor fine. So diffusion metal poly silicon all these geometric you can say this geometries features they are all defined with respect to their minimum width their separation and the reality position.

For example whenever you want to create a transistor, there has to be a diffusion layer on one surface and a poly silicon layer on top of it they must intersect and the minimum area of intersection will be  $2\lambda$  by  $2\lambda$ . These are all design rules which are specified. Now if you need a bigger transistor with a bigger current driving capacity. It can be more than  $2\lambda$  by  $2\lambda$ . But  $2\lambda$  by  $2\lambda$  is the minimum. Similarly when we have two metal layers running in parallel the minimum separation has to be  $3\lambda$ . These are some rules which have been framed out of experience design experience. So here when we do compaction we must honor all those design rules in mind. We cannot compact arbitrarily, so while compacting we must make sure the two metal lines don't come closer than  $3\lambda$  for example fine. So when the problem formulation you can treat this problem mathematically.

(Refer Slide Time: 21:29)

C E T  
I. I. T. R G P

### Problem Formulation

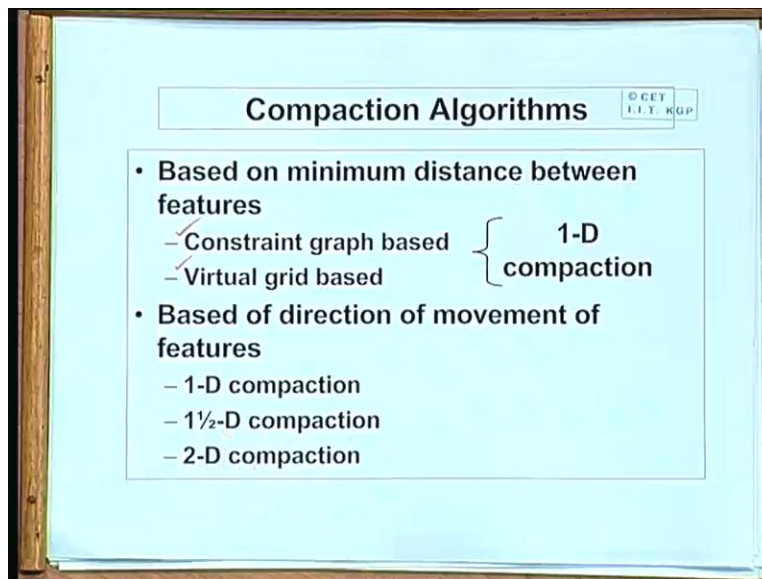
- **Given:**
  - A set of geometric features  $M = \{M_1, M_2, \dots, M_n\}$ .
  - The minimum feature size,  $s(M_i)$ , for all  $i$ .
  - The minimum separation between features  $M_i$  and  $M_j$ ,  $d(M_i, M_j)$ .
- **Objective:**
  - Minimize the layout such that
 
$$\text{size}(M_i) \geq s(M_i)$$

$$\text{dist}(M_i, M_j) \geq d(M_i, M_j)$$
 where  $\text{size}(M_i)$  and  $\text{dist}(M_i, M_j)$  are size of  $M_i$  and distance between  $M_i$  and  $M_j$  after the compaction, where  $1 \leq i, j \leq n$ .

Say, you can say that you can have a set of geometric features. These are all the rectangles I am talking about  $M_1$   $M_2$  to  $M_n$ . The minimum feature size  $sM_i$  for all  $i$  what is the minimum size in terms of the width and length both. The minimum separation between a pair of features depending on which layers it is running the minimum separations separation is also defined. Now objective is to minimize the layout such that the actual size of a feature is of course it should be greater than equal to the minimum specified size and the actual distance between a pair

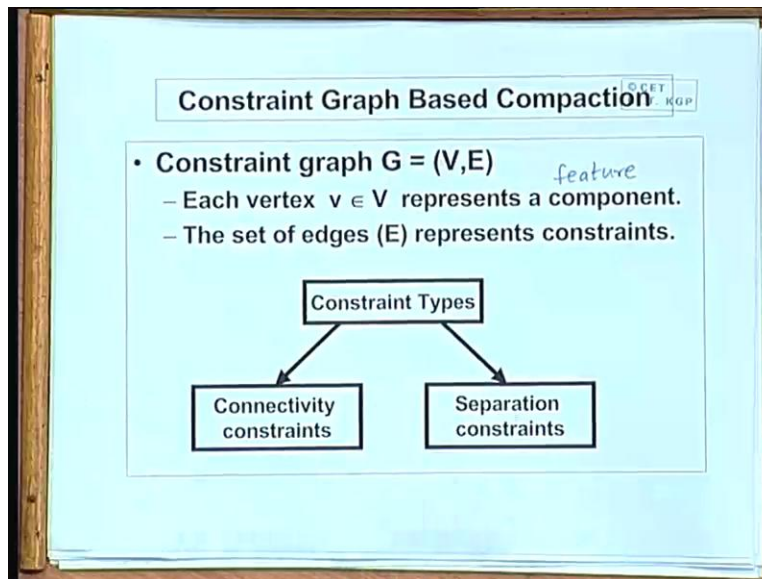
of feature must also be greater than equal to the specified minimum subject to these to constraints you can do as much compaction as you can. [Students Noise Time: 22:21] They are rectangles, they are all defined with respect to different segments of rectangles. That is how they are defined. Now the compaction algorithms well broadly you can classify them.

(Refer Slide Time: 22:44)



As those which are based on these lambda as I have just mentioned based on minimum distance between features. So under this you have constraint graph based and virtual grid based, we will be talking about these. Both these methods well these essentially try to compact a layout in one direction. That is why it is some times called one dimensional compaction. But compaction algorithms you can also classify depending on which direction you are trying to compact. Well one d compaction these fall under one d well you can generalize. The best you can do is you compact in both the directions together. That will possibly give you the best result but usually 2D compaction is difficult to implement. So what you do not have an unconstraint two dimensional compaction. But you compact in one direction with some limited movement in the other direction. This is sometimes called one and a half dimensional compaction. This is something which is more widely used. Well first let us talk about constraint graph based this is of very simple approach. This constraint graph based compaction says.

(Refer Slide Time: 24:05)

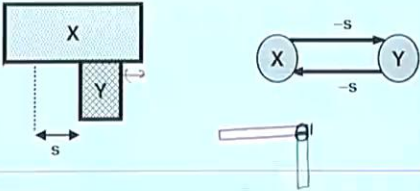


That you first construct a graph which is called the constraint graph. Now in this constraint graph each vertex is a feature or a component whichever you call this component means features same and two features which have some kind of a constraint between themselves will have an edge connecting them. Now constraint which are the edges in this graph can be with respect to connectivity can be with respect to minimum separation fine. So let us say what these two constraints are and how these are represented in the graph.

(Refer Slide Time: 24:51)

**Connectivity Constraints**

- If two features X and Y are required to be within a distance s of each other.
  - A physical connection can be represented in the graph as a pair of edges between X and Y, each with weight  $-s$ .



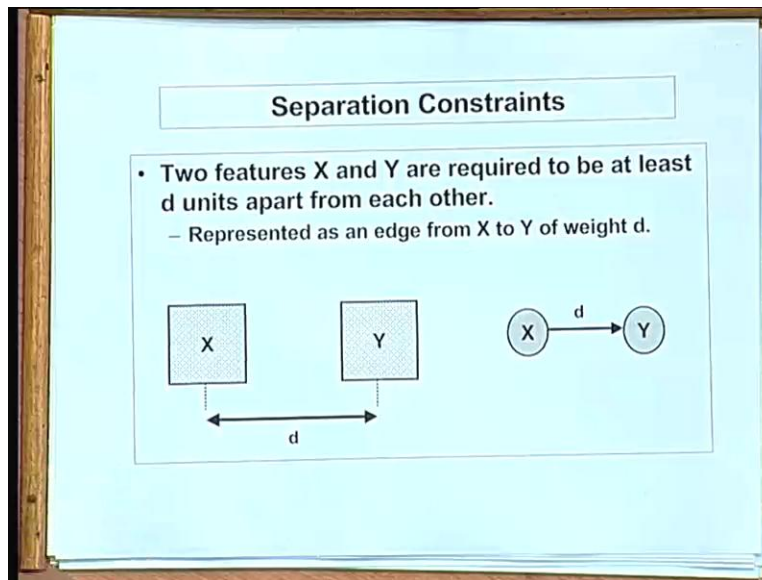
Well the connectivity constraint says sometimes you need to connect two features because you may have a single net. There can be a part of the net running on one layer there can be another part of the same net running on some other layer. One can be metal; one can be poly silicon for instance. But in order to have a faithful connection again the design rule says that you must have a minimum spacing in terms of the width and the height in this overlapped area. So if you can provide this minimum overlap only then the connection will be a faithful one. It will be a reliable one. So the connectivity constraint says the two features are required to be within a certain distance  $s$  of feature. This can be arising out of conditions like this. Well this diagram will be slightly confusing. What I mean to say is that if  $s$  is the minimum distance of if  $s$  is the minimum amount of overlap that we need between two features.

Then you can say that well differently **we can say that** we can say that this feature must be minimum distance  $s$  with respect to this edge of it or minimum distance  $s$  with respect to the other edge of it. In this way you can specify with respect to certain edge the minimum you can say required to be within this much distance not separate. I am not talking about separate. I am saying I am saying they must be within distances. And if you have some constraint like this, the two features must be within a distance in certain distance, then with respect to the constraint graph this  $x$  and  $y$  are two vertices you include two directed edges in both directions with a



weight minus  $s$ . This is a constraint graph the positive weights will indicate separation in constraints negative weight will indicate proximity constraints they must be within a distance of  $s$ . This is for connective similarly for separation you can have something like this.

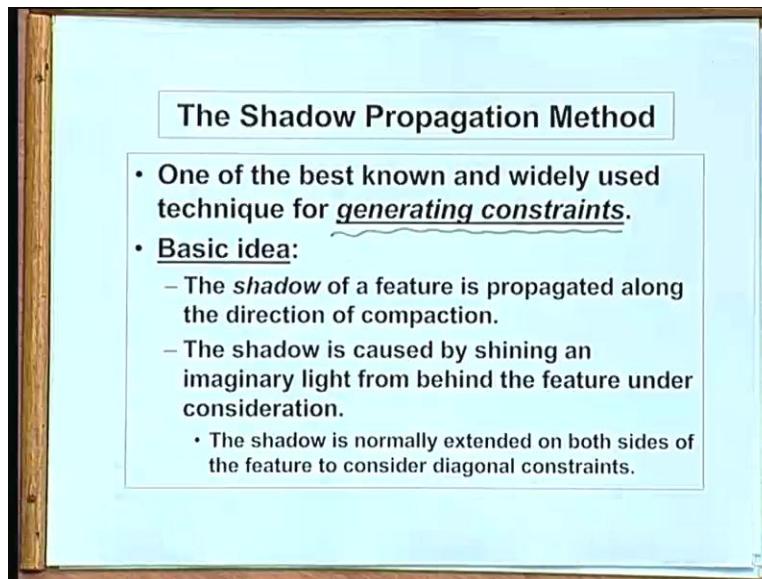
(Refer Slide Time: 27:10)



So if you have two features  $x$  and  $y$  with a minimum distance  $d$  between them you represent it. Well you can represent it by one arrow either in either direction need not be in both directions with a weight  $d$ . This  $d$  will tell that this is the constraint between  $x$  and  $y$  they must be separated minimum by a distance  $d$ . Now this constraint based method what it says are that you first create or construct. This constraint graph this constraint graphs will tell you that what is the relative you can say the minimum distance between separate constraints? What are the different constraints that exist between sub graphs? If there are certain parts where there are no constraints you can compact them without any worry. Now this graph is actually utilized by some other compactor. But you to use it here we are basically interested in creating the graph. In fact there are many ways to create the graph well one of one of the popular way is to use a method called shadow propagation.



(Refer Slide Time: 28:23)

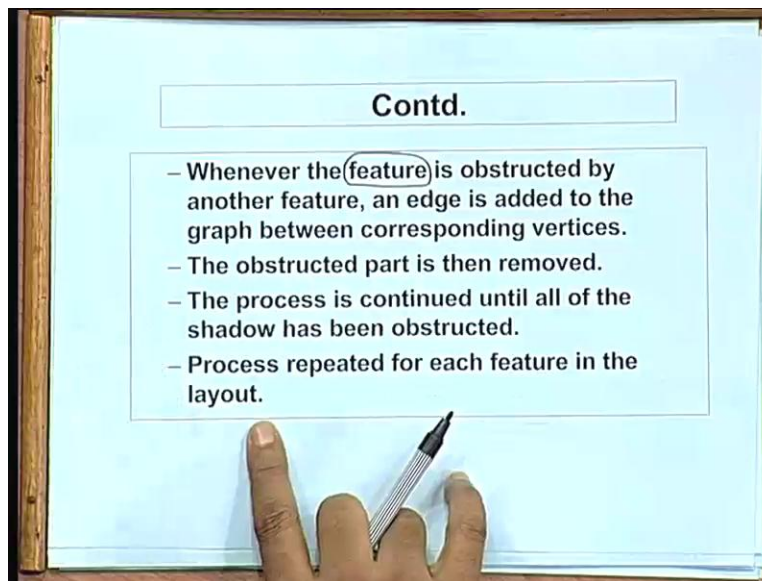


Shadow propagation method is basically an approach to generate the constraints. Because typically in a layout there are millions of features. We cannot just go on comparing pair wise like that and try to find out that which of them or in they have a constraint between them it will be too expensive. So there are some simplified approaches, one such simplified approach is to have some kind of a shadow propagation you create a shadow and try to propagate the shadow across the layout. Well here you take one feature at a time you propagate the shadow of that feature in the direction of the compaction. Suppose you are doing  $x$  compaction and the feature is here you try to propagate the shadow of the feature in this direction. And this shadow whatever we are talking shadow is created by shining an imaginary light. On the other side we are assuming that there is an imaginary light on the other side of the feature which is casting a shadow which is falling on the other side and the shadow might fall on other obstacles it can fall on other features.

Now if the shadow directly falls on another feature then there will be a constraint between the feature and the consideration and that new feature. Because this feature can see the other feature during compaction it is possible that they might come in close proximity and touch each other. Because we are not moving it up and down we are only doing it  $x$  compaction. Suppose I can see you in the  $x$  direction this means if someone compact you can come near me right. So there will

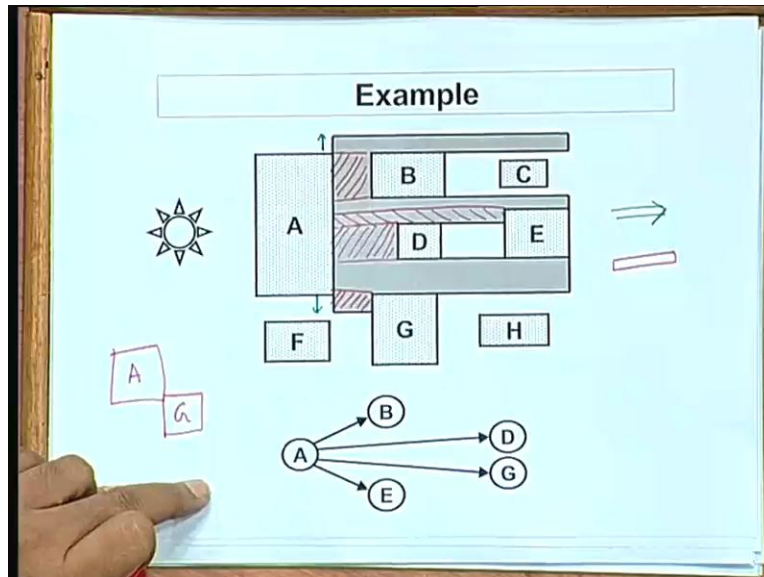
be a constraint between them if we need to remain separated by a minimum distance. So on whichever feature the shadow falls there will be a constraint edge that will be generated in the graph. This is the idea. Well the shadow is normally extended a little bit on both sides. I will take an example to show it to consider diagonal elements also in the layout. I will take an example.

(Refer Slide Time: 30:46)



And the idea is that whenever the feature means, I mean the shadow of the feature is obstructed by another feature. You add an edge as I had mentioned and after you add an edge the portion of the shadow which was obstructed that portion you remove because that portion has been taken care of. The obstructed part of the shadow is removed. You continue this process until all of the shadow has been taken care of and you repeat this process for all features throughout. Say I have an example. So if you look at the example you can understand better take this ex yes ok fine.

(Refer Slide Time: 31:32)



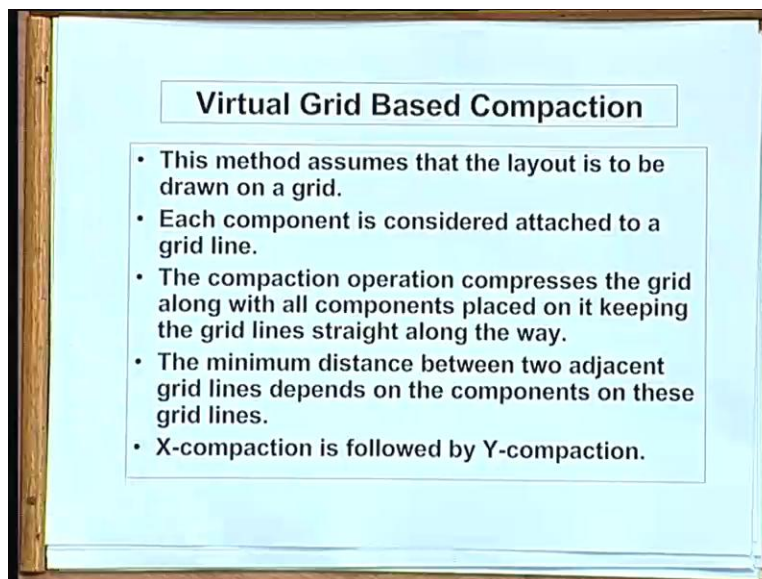
So you take a layout where the features are a, b, c, d, e, f, g, h, these are the features and I am considering this feature a and I use an imaginary light source on the back side and a shadow is caused in this direction. And I mentioned that in order to take care of diagonal constraint this shadow is slightly extended in the two directions you see this is extended because there can be a block here and sometimes you may also have some diagonal constraint. Now this extension can take care of that diagonal constraint. So now when this shadow is caused using some geometrical methods you try to find out that which the obstacles that are hit are by the shadow. First is b so a part of the shadow is obstruct by b which is this region. Similarly a part of the shadow hits d it is this region a part of the shadow hits e it is this region and a part hits g the remaining part hits maybe the boundary of the chip and they gets deleted in that way.

They do not have any obstacle in between they extend till the boundary of the chip or if there are any other obstacle it will hit them. So you do this in this way till all portions of the shadow gets deleted or gets taken care of like this. Now you see that a gets or the shadow of a hits b, d, e and g. So you will be adding four such edges in the graph from a. The weights of the edges will depend on which layer these are and what are the constraints if the polysilicon to polysilicon. It can be  $2\lambda$ . If it is polysilicon to metal, it can be  $3\lambda$  and so on. [Students Noise

**Time: 33:45]** No. These below but since there are diagonal it is possible that while compacting this, a and g can come as close as this. So I am trying to avoid this. That is why I have just extend this shadow little bit in both directions. So the two blocks do not get touched at the corners fine.

This method works very well but as you can just understand very clearly this is computationally very intensive. We have to generate so many constraints you have to cast so many shadows and process it. **[Students Noise Time: 34:24]** Yes. **[Students Noise Time: 34:30]** That is how **your you your** your idea is that if if for example we have a some distance like 5 lambda that is the maximum separation considering about ever we will not be considering anything beyond that. But you just imagine a situation where this e this block e instead of a square block like this. This was a thin block like this. Say although it was more than 5 lambda away, but while compacting it come very near to a that also need to be checked whether it has a free passage in between up to a. There are some other approximations.

(Refer Slide Time: 35:30)

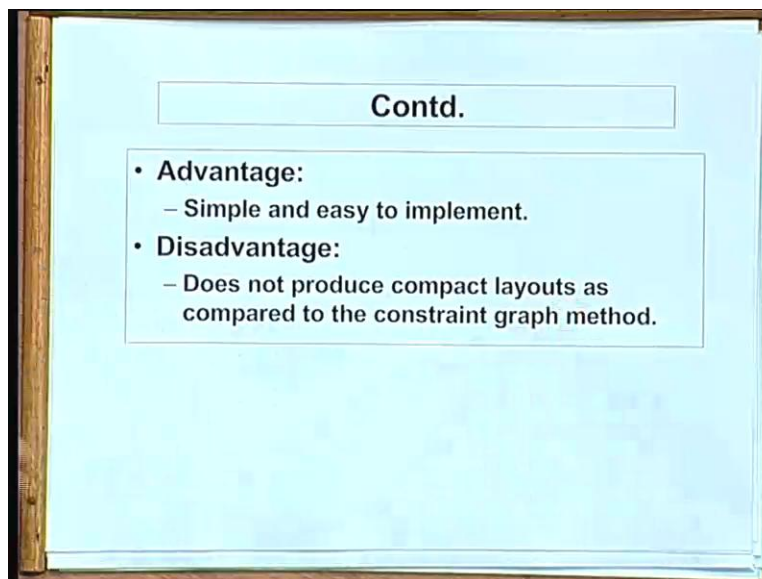


Well this of course does not works as well as the shadow propagation. This is called; see this shadow propagation is a way to generate the constraint graph using the graph there are a variety

of approaches to do compaction. Virtual grid based compaction in fact it can also utilize the constraint graph for the purpose. Well here as the name implies there is the concept of a grid we are assuming that layout is drawn on a grid. I will take an example. An each component on the layout that is a feature is assumed to be attached to one of the gridline. And when you do a compaction you are moving the gridlines and all the components which are attached to gridline moves along with it.

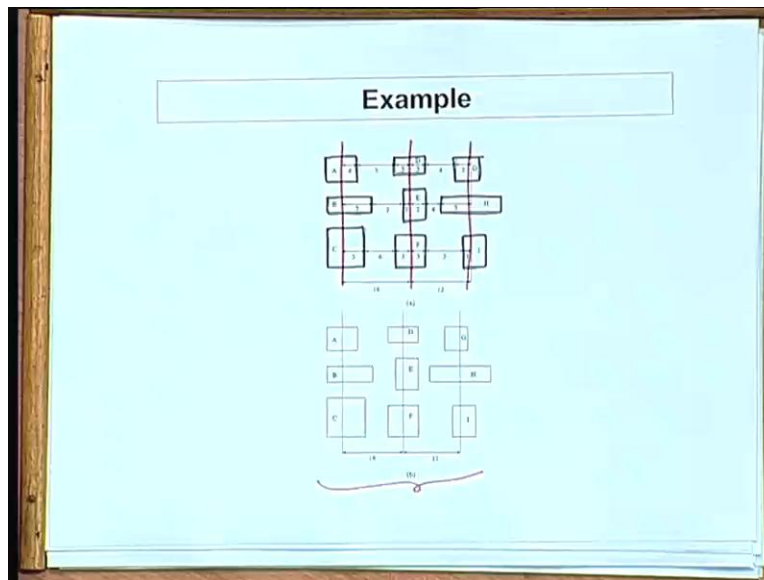
So the components cannot move individually when you move a gridline if there are five components attached to that gridline all of them will move together right. So when you compute the minimum distance between two gridlines. It will be the maximum of the minimum distances between every pair of components which are attached to those gridlines because you are bringing all of them together. So whatever is the maximum constraint that means that that much separation has to be there. So typically this approach compacts in one direction at a time. Usually we first do x compaction then do y compaction first in x direction then in y direction.

(Refer Slide Time: 37:10)



The advantage of this approach is that this is much easier to implement and computationally much less intensive. Drawback is that the quality of the layout you get is not as good as the other approaches where you take care of all the constraints and do it. Well I have an example well.

(Refer Slide Time: 37:30)



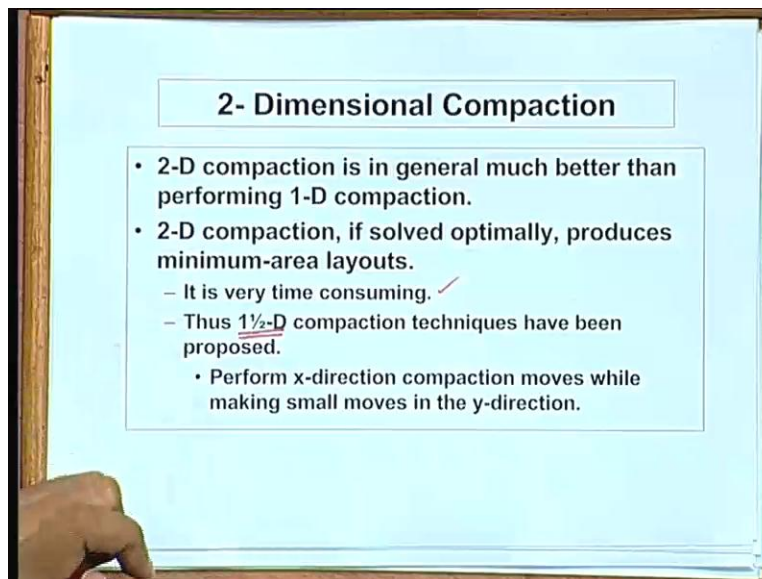
So here the blocks are little small. I am showing you these are the features and suppose in the layout these are the feature. You see here let me tell you one thing also. So when you do compaction some of the compactors also work at a slightly higher level like instead of working at those rectangles represented by polysilicon diffusion and metal layers segments of them. They can even consider the layout of a transistor as an entity the entire layout of the transistor can be moved. That is also an approach which is called. Because here the virtual grid is better suited for the later approach where each component is a macro component. It is a compact layout of a slightly higher subsystem.

It can be a gate it can be a single tra it can be a gate. I mean gate or a transistor also fine. So you see that here there are 9 components and you assume three gridlines one here, one here and one here. Now depending on the types of this components and the interaction some minimum distance between them will be defined you try to compact the grids. Well here a compacted

layout is shown. But actually you try to bring these two gridlines close together subject to this minimum separation you should not violate the minimum separation whichever you specified here. So you try to bring them as close together as possible. But the constraint is that whenever you move a gridline you are carrying all of them together.

But possibly if this central block you could have pushed it individually you could have got a better layout. But here that is the limitation you are not allowed to do that. That is why the quality of compaction you get is usually not very good. So I had mentioned two dimensional compaction is the ideal thing. We compact in both directions together. But the problem is that the optimal two dimensional compaction is computational very hard. It is not feasible to try and have an optimal two dimensional compaction it is very time consuming.

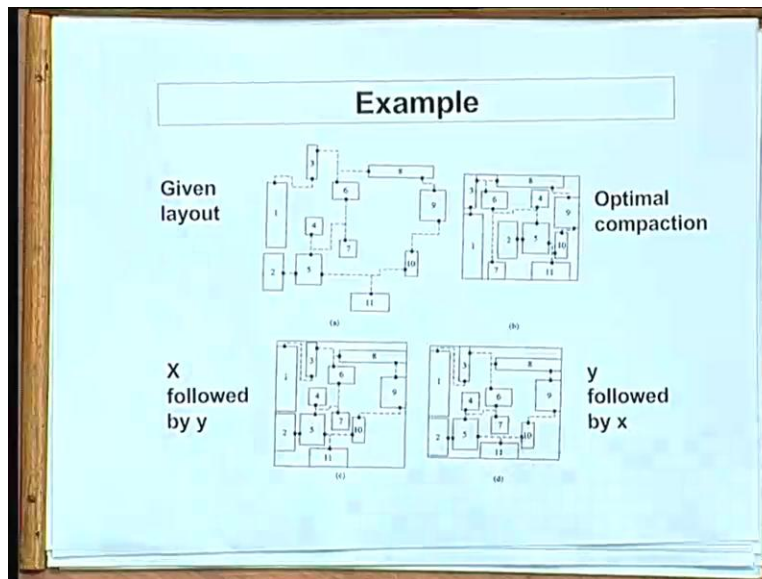
(Refer Slide Time: 40:22)



So I have mentioned due to this normally people use one and half dimension compaction. Now in order to get a feel of two dimensional compaction I have an example like just I am showing the example.



(Refer Slide Time: 40:43)



Fine. This is a given layout these are the different features of the components. This is the given layout, this is the layout which we get by first applying an x compaction then an y compaction. See after you have the constraint graph with you x compaction is easy you bring each of the component as close together to left edge as possible without violating the constraint it is a greedy approach that with respect to the x coordinates you go on bringing them as far left as possible. So you first do an x compaction followed by an y compaction. You get a layout like this. But if you do it the other way around first and y compaction and then x you see you get a layout whose height is less. But width is slightly more. But here I am also showing the optimal compaction where well it is neither x followed by y nor y followed by x you are doing both x and y together. Then we can get a much better layout which has **which is a which has** a much less area as compared to any of these.

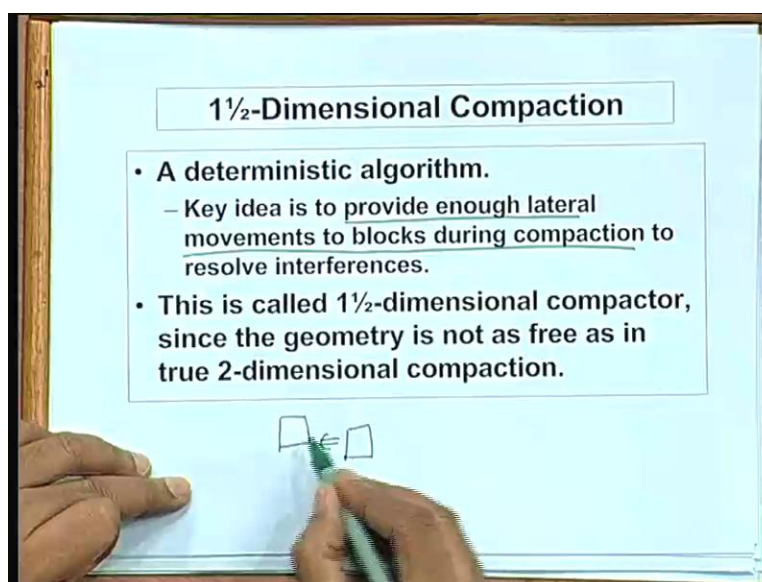
**[Students Noise Time: 41:59]** relative placement. Yeah. Relative placement facts have been changed. Yeah. So you can also shift around no see in an optimal placement you can also shift things around. **[Students Noise Time: 42:16]** No routing. No. **[Students Noise Time: 42:20]** See **[Students Noise Time: 42:22]** No these are all we mind for example this I means I will just show you with the with that one and half d half d compaction. You see a block you are trying to



bringing it to the left you see that you cannot bringing it to the left. Because there is no obstruction but there are lot of free spaces if you move a block slightly on above it can fit. So this kind of small moves can be done during compaction. [Students Noise Time: 42:48] Yeah. [Students Noise Time: 42:51] Robustly out no this can be done only for custom based layouts. [Students Noise Time: 42:57] Routing will disturb. See this compaction I typically done only at a very local level.

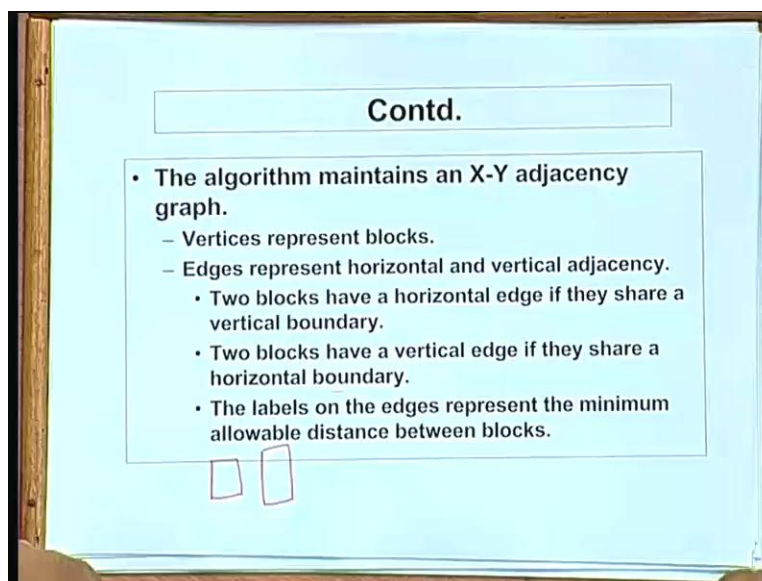
You do not do it very globally because there are so many components which are already densely packed in the chip. So the whatever technique we apply the small changes here and there that applies only a only to a small portion of the layout. Maybe the placement which you are done earlier you are disturbing that. But here you are working at a much lower level you are not talking about the functional blocks. In placement you are talking about the higher level functional blocks and placement them. But here possibly you are talking of a gate or a transistor much lower level. Moving the low level components slightly here and there that will not matter much in terms of your circuit performance. But again telling you this. This circuit compaction is an optional step you may go for it you may not go for it. So let me tell you in slightly detail the basic concept behind the one and a half dimensional compaction method.

(Refer Slide Time: 44:16)



Well one and a half dimensional compaction this is a deterministic algorithm and it is called one and a half dimension. Because here basically you are trying to compact in one direction and you are allowing limited movement of the blocks in the other direction during the process. You provide enough lateral movement to blocks during compact just what I have mentioned that if a block which you say a block was here and a block were here which you are trying to move in this direction. This block will be hitting this. But if you find that this block can be slightly move up. Then this block can possibly be pushed down here. So you are allowing limited lateral movement of the blocks during compaction the x direction because the lateral movement is limited it is not free. That is why it is called one and a half dimensional compaction. You are not really compacting in the other direction. But allowing some movement of blocks in that direction. Now this algorithm maintains some kind of an adjacency graph.

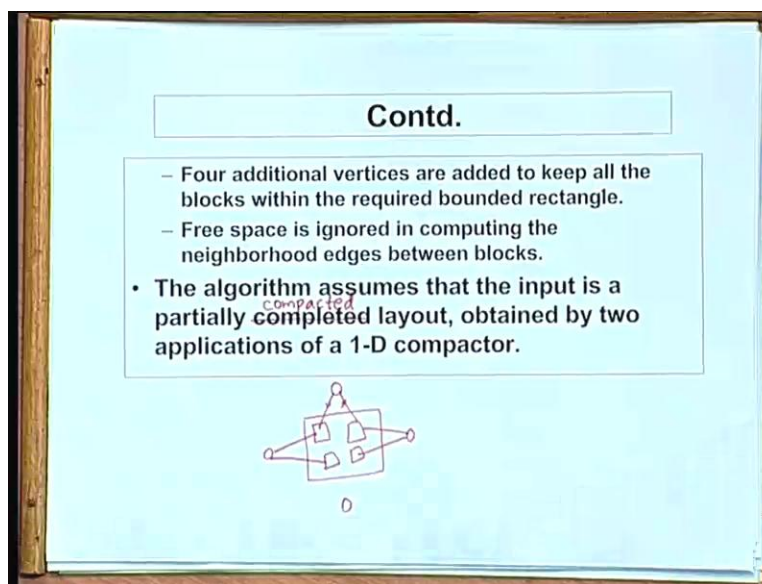
(Refer Slide Time: 45:41)



This is an x y adjacency graph x y. Means one in the x direction other in the other in the y. Direction vertices in this graph will represent the blocks the layout. The edges will represent horizontal and vertical adjacency. Say if there are two blocks one here, one here where there is some kind of a vertical boundary between them. If you bring them together they will hit there

will be there will be an horizontal edge. Similarly if you have one below the other there will be a vertical edge. So one in the x direction other in the y direction. In both direction you try to represent the proximity of the block in terms of the edges in the graph. The labels on the edges will indicate the constraint. That is the minimum allowable distance minimum separation. So the edges will have labels and you will have to honor that minimum separation whenever you are doing compaction and I will take an example.

(Refer Slide Time: 47:02)



**Contd.**

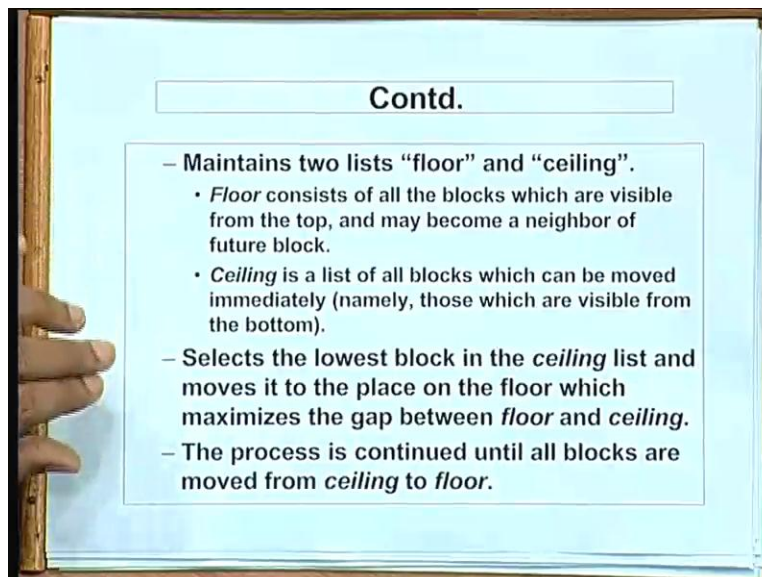
- Four additional vertices are added to keep all the blocks within the required bounded rectangle.
- Free space is ignored in computing the neighborhood edges between blocks.
- The algorithm assumes that the input is a partially <sup>compacted</sup> completed layout, obtained by two applications of a 1-D compactor.

The diagram shows a rectangular layout of blocks with four additional vertices (top, bottom, left, right) and edges connecting them to the blocks.

Well you say you have a layout there are blocks and with respect to the blocks you draw the graph. Now you have four additional vertices you add them. These are hypothetical vertices one on top one on bottom left and right. These actually represent the boundary of the rectangular region. Because the blocks which are visible from top they will be having an edge from this node. One with the visible from here they will have edges from here this will have edges here and here. So these four additional vertices these are added to keep the blocks within the required bounded rectangle because you can put appropriate weight to this edges so that they will never come near there minimum separation will always be there.

And here whenever we are constructing the graph we are ignoring the free space we are not keeping track of the free space and this algorithm takes as input a partially completed layout. Well this partially completed layout maybe a layout compacted once in x, once in y, using a using a conventional compact which is not a very good compact but that can be starting point of this algorithm. So you [Students Noise Time: 48:35] Well this should be a compacted you are right partially compacted. This is not complete compaction. But we have done we have used some simple algorithm to do some compaction that you take as a starting point. Well I will just, I will demonstrate some steps of the algorithm. For an example.

(Refer Slide Time: 49:02)

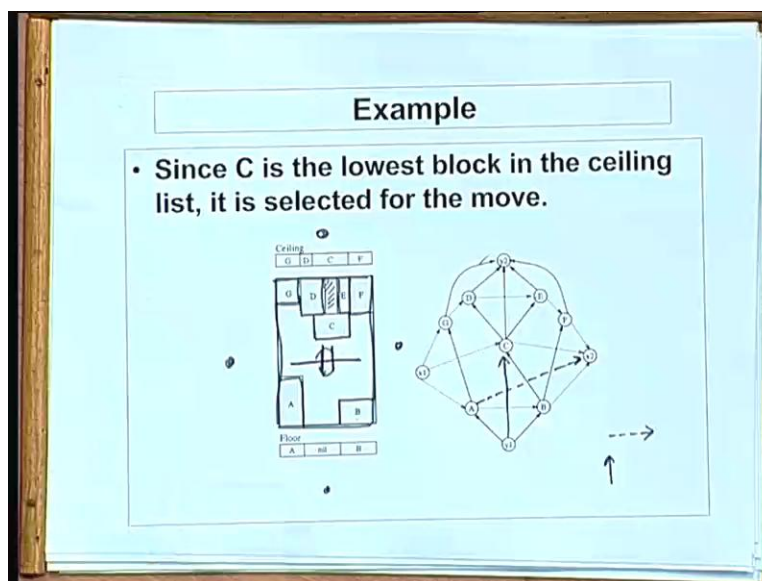


But before that if we talk about some kind of data structure which is maintained. The algorithm maintains two lists floor and ceiling floor consists of all blocks which are visible from the top. See whenever you pick from the top what you see you see the floor. So whenever you look from the top whatever you see down below the blocks which are there visible to you they will belong to the list floor. Similarly ceiling is something when you look up you will see some blocks they will belong to the ceiling list. Ceiling is the list of all blocks which can be well we can say which are visible from the bottom which can be moved down. Visible from the bottom means what there is no obstruction in between it can directly fall on my head I can bring it down. So the idea

is that you select the lowest block in the ceiling list and move it to the place on the floor which maximizes the gap between floor and ceiling I will take an example again.

Ceiling list is the block which you can see from bottom. You bring it down and place it on the floor. Well you do not bring it down straight down bully. You can move it slightly to the left or right place it at a point where the distance between that block and the ceiling is maximized. This is the heuristic which is used and the process is continued until all the blocks are moved from ceiling to the floor. The idea is that we have some block on the floor I am slowly moving blocks from the ceiling to the floor. That means I am compacting in this direction, I am starting in one direction I am slowly moving the blocks down, down, down and while moving the block down, I am allowing certain lateral movement of the block. I am putting a block in a place which will be maximizing the distance between the ceiling and the floor. [Students Noise Time: 51:12] Top and bottom I am taking an example take an example here.

(Refer Slide Time: 51:21)



This is a partially completed layout. Well I have shown it this way in order to explain it properly these are the blocks. This is empty space this is a block out here this is a block. Now this ceiling well the letters are, so this is a, b, c, d, e, f and g ceiling will consist of the list when if you from

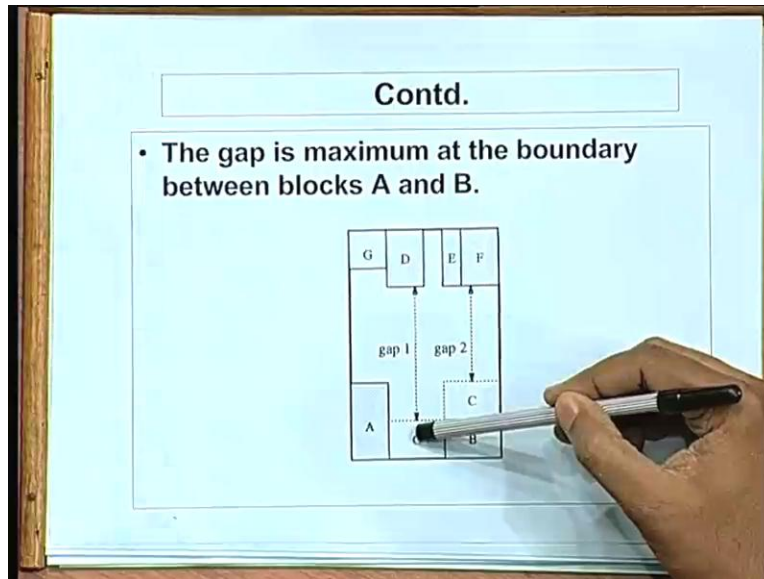
down if you look towards the top. Say from down if you look towards top you can see g you can see d, c and f but e is not visible. So ceiling will consist of the set g, d, c, f. Similarly when you look down you will see a and b so the floor list will contain a and b. This is the graph this graph will consist of all these edges see a and b there is a vertical edge between them this dotted arrows are the horizontal and the solid arrows are the vertical. So if there is a vertical edge we add a dotted arrow between a and b.

Similarly between g and d there is a vertical arrow between d and e there is an arrow between e and f there is an arrow similarly. Now similarly for the other direction from between c and d c and e there are arrows and we assume 4 4 hypothetical vertices on 4 sides they represent the edges of the layout  $y_1, y_2, x_1, x_2$ , this  $y_2$  represents the top end. From say from top this  $y_2$  has connection with g, d, c, is also visible there is a part e and f there will be edge to all 5 of 5 edges. Similarly down here there will be an edge to a and b. Similarly for this f, c b and also a [Students Noise Time: 53:50]. Yeah c is there in the list. C is [Students Noise Time: 53:58] No. [Students Noise Time: 54:01] Downward means see I have started to place the blocks on the floor down means whatever I have placed you look from above that, it does not mean that you just pick from here you can see c hanging on top.

No c is a list which I have not brought down to floor. They were on the ceiling. I am slowly taking one by one bringing down to the floor this is the concept. Looking from the top means, I am just looking from the point here some are here there are some blocks which are already placed on the floor. I am just looking from a point just above that. [Students Noise Time: 54:39] Ok. Ok. Right. Right. Right some edges are missing you want to see this missing. Similarly this, a to x two is missing. Now fine some edges are missing ok fine alright. Now the idea is that see you select a block from the ceiling to move down. Now you select the one which is closest to the floor here it is c. So you take c has the candidate block to move down. Now I have another slide to show. Suppose you have selected this block c to move down.

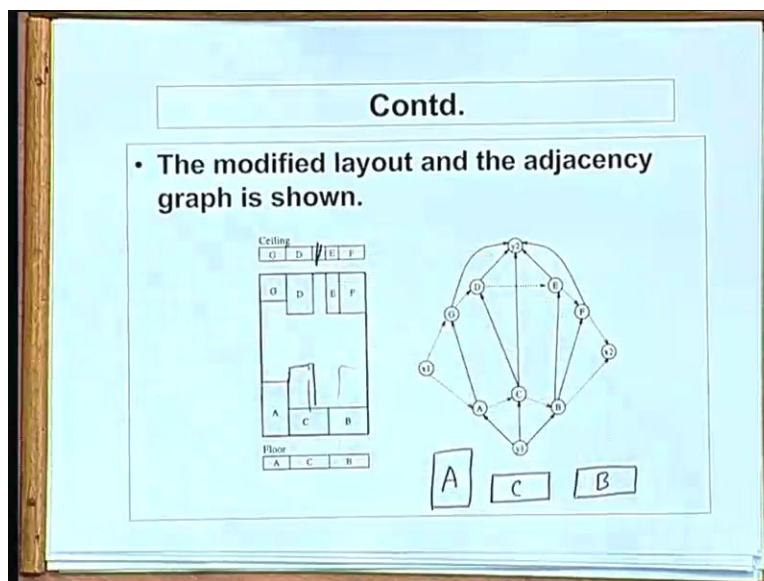


(Refer Slide Time: 55:39)



Now here you have an option c you can place on top of b, c you can place between a and b. Since the gap out here is more you take this as the final position of c and not this. This is the heuristic which is used. So after this first step.

(Refer Slide Time: 56:00)



Your c will come here the floor and the ceiling list will get modified and also this graph will get modified this process will continue. Next I am possibly you will take d. D can be put anywhere. It can be put here, it can be put here, and anywhere same distance will be the same. No. This will not be the same we will have to put it here, here. So you take the blocks and do it in the same way bring them down try to put it move it laterally. So as to maximize the distance and do it in a greedy fashion one after the other. [Students Noise Time: 56:38] Graphs see here although I have shown it like this. But actually there will be some separation between a and c, and b and c the graph will tell you that what is the minimum separation will have to be maintained. So this is just for the sake of illustration actually this a and c will have some minimum distance between them gap, some gap b will be here. This graph will give you that information what is the minimum separation of the gap that need to be mentioned. [Students Noise Time: 57:19] c is in the ceiling list because c, No. No. No. C will not be in the ceiling list c will get deleted. You are right. C has already been brought to the floor c will not be in the ceiling list. So with this we finish our discussion on backend cad tools. Thank you.