Chapter 10

Over-the-Cell Routing and Via Minimization

The current trend in VLSI design is to develop high performance chips. The main objective of physical design is satisfy the performance needs while minimizing the die size. Historically, the gate delays limited the chip performance. The developments in fabrication process technology in the past two decades have resulted in a phenomenal decrease in feature sizes, and introduced additional metal layers for interconnections (routing). Deep sub-micron processes with five to seven metal layers for interconnections are now available for design of high performance and high density chips. The number of devices in a chip have increased from about a thousand devices in the early 70's to over twenty million devices now. The increase in the number of devices has led to a significant increase in number of interconnections. Interconnect delays, which were considered to be insignificant earlier, have now become comparable, if not more prominent than the gate delays.

With the availability of five to seven metal layers for interconnections, three dimensional routing techniques are necessary to satisfy the performance and density goals. The number of metal layers provide the third dimension. Therefore, for interconnect planning, routing *volume* needs to be considered, instead of just the routing *area.* The space in all metal layers across the entire die, both over active areas and in channels need to efficiently utilized for routing. This concept was first introduced in standard cell designs. Several existing channel routers can produce solutions only one or two tracks beyond optimum for most channels. Despite this fact, as much as 10% of the area in a typical layout was still consumed by routing. Considering a fixed placement, in view of this 'optimality' of channel routers, further reduction is only possible if some nets can be routed 'outside' the channel (as the area allocated for the standard cells is inherently fixed by the circuit design). In particular, the metal layers available over the cell rows can be used for the routing. This technique is called *over-the-cell* routing. The over-the-cell routing style for standard cell designs has become both practical and important as more and more metal layers are

made available for routing.

Over-the-cell routing concept is used across the entire chip and it is not feasible to design complex high performance microprocessor chips (100Mhz to 1GHz) without adopting over-the-cell routing techniques. This book describes basic OTC routing algorithms for standard cell designs and advanced concepts can be found in [SBP95].

After a chip is completely routed, the layout is functionally complete and can be sent for fabrication. However, the layout is usually improved to reduce the possibility of fabrication errors, reduce the total chip area and therefore, improve performance. In most current technologies, two or more layers are available for routing. Most of the existing routing algorithms use a large number of vias to complete the routing. This is due to the fact that most routers use a reserved layer model. However, vias are undesirable from fabrication as well as circuit performance point of view and therefore, the number of vias should be kept as small as possible.

Significant volume of research exists on techniques for reduction of the number of vias in a completed detailed routing by re-assigning the wire segments to different layers. This kind of via minimization is called *Constrained Via Minimization* (CVM). Via minimization has also been considered without the restriction of completed routing. In this approach, the actual layout of wires can be changed and thus offers more flexibility as compared to the CVM approach. This via minimization approach is called *Unconstrained Via Minimization* (UVM) or *Topological Via Minimization* (TVM).

In this chapter, we discuss the problem of over-the-cell routing and via minimization to improve detailed routing solutions. In Section 10.1, we discuss the problem of over-the-cell routing. Both CVM and UVM problems have been considered in Section 10.2.

10.1 Over-the-cell Routing

The total layout area in the standard cell design style is equal to the sum of the total cell area and the total channel area. For a given layout, the total cell area is fixed. Thus, the total area of a layout can only be reduced by decreasing the total channel area. As several channel routers have been developed that complete channel routing with the number of tracks very close to the channel density, further improvement in the layout area is impossible if routing is done only in channels.

Internal routing of cells is typically completed using one metal layer. Therefore, the higher metal layers (M2 and M3) over-the-cell are un-utilized. The area in M2 and M3 can be utilized for routing of nets in order to reduce the channel height. As the number of layers allowed for routing increases, the over-the-cell routing problem becomes important. Since the conventional channel routing problem is known to be NP-hard [Szy85], and the over-the-cell channel routing problem is a generalization of the conventional channel routing problem, it is easy to see that the over-the-cell channel routing problem is also

NP-hard [GN87].

Several algorithms for over-the-cell routing have been presented, and the technique has proven to be very effective [CL88, CPL93, HSS93, LPHL91]. In the following, a review of algorithms for over-the-cell channel routing is presented. We start by describing the physical constraints for over-the-cell routing.

10.1.1 Cell Models

Based on the locations of the terminals there are four major classes of cell models : Boundary Terminal Models (BTM), and the Center Terminal Models (CTM), the Middle Terminal Model (MTM) and the Target Based Cell Model (TBC). Each of these classes contain several cell models based on the variations in other routing parameters, *i.e.,* the number of metal layers and permissibility of vias in over-the-cell areas.

- **Boundary Terminal Model(BTM):** This is the traditional cell model. This was introduced when only two metal layers were available for routing. In BTM, there are two parallel horizontal diffusion rows, one for the P-type transistors and the other for N-type transistors. The first metal layer (Ml) is used to complete connections which are internal to the cells. The power and ground rails are in M2 layer, adjacent to each other, in the center of the cell row. Terminal rows are available in all layers and are located on the boundaries of the cells [HSS93]. This leaves a rectangular, over-the-cell routing area for each terminal row of the standard cells. The number of tracks available for over-the-cell routing is determined by the height of these rectangular areas and may vary depending on the cell library used. The entire over-the-cell area may be used for routing in the third metal (M3) layer. This model is used by most existing over-the-cell routers [CPL93, HSS93, HSS91]. This class of cell models is referred to as BTM or class of Boundary Terminal Models. (See Figure 10.1(a)). BTM contains, 2BTM (2 layer process), 3BTM-V (3 layer process when vias are not allowed in over-the-cell areas), and 3BTM+V (3 layer process when vias are allowed in over-the-cell areas).

- **Center Terminal Model(CTM):** This class of cell models is quite different than BTM in terms of terminal location. In CTM, the terminals are located in M2, in the middle of the cell. The power and ground rails are in Ml near the top and bottom cell boundaries respectively. Connections within the cell are completed in Ml. Thus, M2 is only blocked by terminals, and M3 is completely unblocked (See Figure 10.1(b)). Over-the-cell routers may use two rectangular regions (about thirteen tracks wide) in M2 and M3.

- **Middle Terminal Model(MTM:** This model differs from the BTM and CTM in terms of terminal locations. In MTM, the terminals are located in two rows, one row is located k_1 tracks below the upper cell

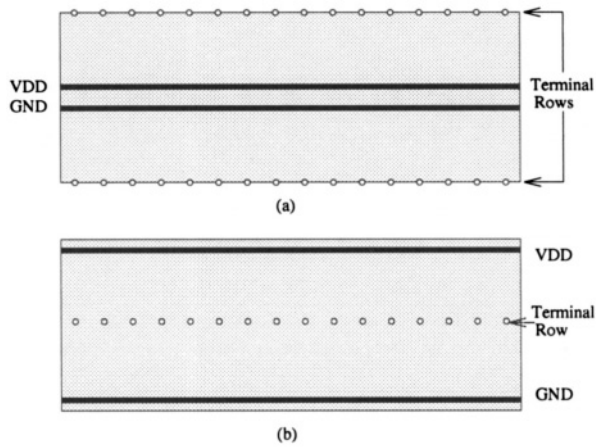

Figure 10.1: Existing Cell Models (a) BTM (b) CTM

boundary and another is located k_3 tracks above the lower cell boundary. As in CTM, in MTM, terminals are available only in M2 and the power and ground rails are in Ml near the top and bottom cell boundaries respectively (See Figure 10.2). Both terminals in a column of a cell are equi-potential. Intra-cell routing is completed in poly and Ml, and does not block M2. As opposed to two rectangular regions in CTM, over-the-cell routers for MTM may use three rectangular regions in M2 and M3 as discussed below:

1. **T area:** k_1 track wide area between the upper cell boundary and the upper terminal row,
2. **C area:** k_2 track wide area between the lower terminal row and the upper terminal row,
3. **B area:** k_3 track wide area between the lower terminal row and the lower cell boundary.

- **Target Based Cell (TBC):** TBC is designed to effectively utilize the over-the-cell areas for routing. The terminals are in the form of long vertical strips in Ml layer, called *targets.* The exact location of the interconnection contacts on the targets is determined by the routing algorithm. The power and ground lines are located in Ml layer at the top and bottom cell boundaries, respectively. The TBC cells have targets of non-uniform heights and are placed arbitrarily, as shown in Figure 10.3. Since the power and ground lines and the targets are located in Ml layer, the over-the-cell areas in M2 and M3 areas are completely unblocked.

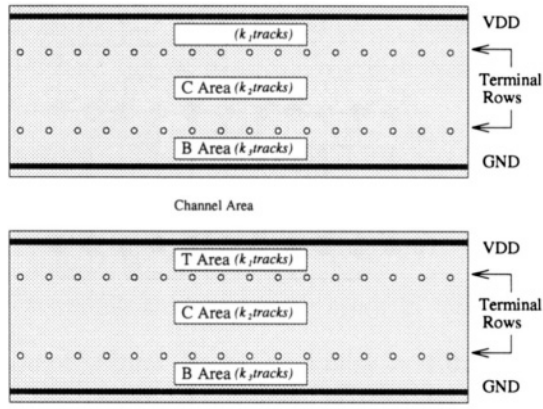

Figure 10.2: Middle Terminal Model (MTM)

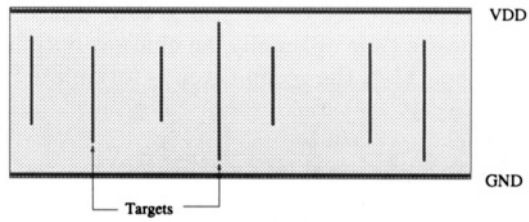

Figure 10.3: Target Based Cell (TBC)

10.1.2 Two-Layer Over-the-Cell Routers

The two-layer routing problem essentially boils down to selection of two planar sets of segments. One of them is routed in the upper over-the-cell area and the other is routed in the lower over-the-cell region. The nets that are not selected are routed in the channel area. In the following, we discuss two algorithms for over-the-cell routing.

10.1.2.1 Basic OTC Routing Algorithm

In [CL90], Cong and Liu presented an algorithm for the over-the-cell channel routing. It divides the problem into the following three steps:

1. routing over the cells,
2. choosing net segments in the channel, and

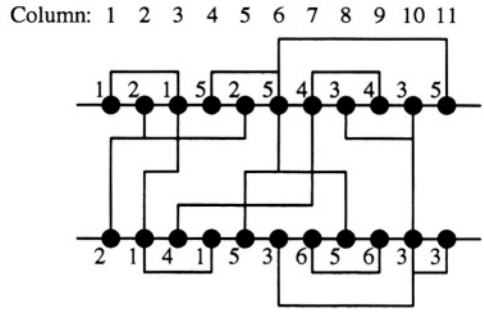

Figure 10.4: A valid over-the-cell routing solution.

3. routing in the channel.

The first step is formulated in a very natural way as the problem of finding a maximum independent set of a circle graph. Since the later problem can be solved in quadratic time optimally, an efficient optimal algorithm is obtained for the first step. Also, the second step is formulated as the problem of finding a minimum density spanning forest of a graph. The minimum density spanning forest problem is shown to be NP-hard, so, an efficient heuristic algorithm is presented which produces very satisfactory results. A greedy channel router [RF82] is used for the third step.

There are two routing layers in the channel, and there is a single routing layer over-the-cells for inter-cell connections. Clearly, the over-the-cell routing must be planar.

The first step of the over-the-cell channel routing problem is to connect terminals on each side of the channel using over-the-cell routing area on that side. The same procedure is carried out for each side (upper or lower) of the channel independently. Let t_{ij} denote the terminal of net N_i at column j. In a given planar routing on one side of the channel, a hyperterminal of a net is defined to be a maximal set of terminals which are connected by wires in the over-the-cell routing area on that side. For example, for the terminals in the upper side of the channel in Figure 10.4, $\{t_{5,4}, t_{5,6}, t_{5,11}\}$ is a hyperterminal of net 5. $\{t_{2,2}\}$ is also a hyperterminal. Obviously, when the routing within the channel step (the third step) is to be done, all the hyperterminals of a net need to be connected instead of connecting all the terminals of the net, because the terminals in each hyperterminal have already been connected in the over-the-cell routing area. Intuitively, the fewer hyperterminals are obtained after routing over the cells, the simpler the subsequent channel routing problem. Thus the first step of the problem can be formulated as routing a row of terminals using a single routing layer on one side of the row such that the number of hyperterminals is minimum.

After the completion of the over-the-cell routing step, the second step is to choose net segments to connect the hyperterminals that belong to the same

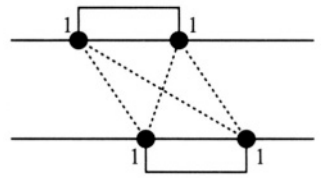

Figure 10.5: Possible net segments for connecting two hyperterminals.

net. A net segment is a set of two terminals of the same net that belong to two different hyperterminals. For example, for the two hyperterminals of net 1 on the opposite sides of the channel in Figure 10.5, there are four possible net segments that can be used to connect these two hyperterminals (indicated by dashed edges), while only one of them is needed to complete the connection. Thus the second step of the problem is to choose net segments to connect all the hyperterminals of each net such that the resulting channel density is minimum.

After the net segments for all the nets are chosen, the terminals specified by the selected net segments are connected using the routing area in the channel. The problem is now reduced to the conventional two-layer channel routing problem. A greedy channel router [RF82] is used for this step. Other two-layer channel routers may also be used.

Net Selection for OTC Routing: The first step of the over-the-cell channel routing problem is to route a row of terminals using a single routing layer on one side of the channel such that the resulting number of hyperterminals is minimized. This problem is called the multi-terminal single-layer one-sided routing problem (MSOP).

MSOP can be solved by a dynamic programming method in $O(c^3)$ time, where c is the total number of columns in the channel. Given an instance I of MSOP, let $I(i,j)$ denote the instance resulting from restricting I to the interval $[i,j]$. Let $\mathcal{S}(i,j)$ denote the set of all the possible routing solutions for $I(i,j)$. Let:

$$M(i,j) = \max_{S \in \mathcal{S}(i,j)} \{\sum_{k \geq 2} (k-1) d_k(S)\}$$

where $d_k(S)$ is the number of hyperterminals of degree k in S. If there is no terminal at column i, clearly, $M(i,j) = M(i+1,j)$. Otherwise, assume that the terminal at column i belongs to net n. Let $x_{n_1}, x_{n_2}, \ldots, x_{n_s}$ be the column indices of other terminals that belong to net n in interval (i,j). Then, it is easy to verify that

$$M(i,j) = \max(i+1,j), \max_{1 \leq l \leq s} \{M(i+1,n_l) + M(n_l,j)\}$$

It is easy to see that this recurrence relation leads to an $O(c^3)$ time dynamic programming solution to MSOP.

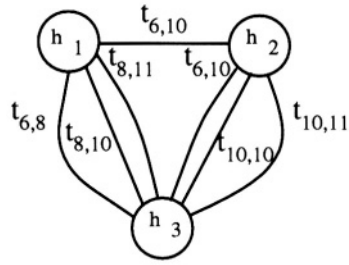

Figure 10.6: The connected component induced by net.

Channel Segment Selection: After the over-the-cell routing, a set of hyperterminals is obtained. The terminals in each hyperterminal are connected together by over-the-cell connections. The next problem is to choose a set of net segments to connect all the hyperterminals of each net such that the channel density is minimized. This problem can be transformed to a special spanning forest problem, as discussed below.

For an instance I of the net segment selection problem, the connection graph $G = (V, E)$ is defined to be a weighted multi-graph. Each node in V represents a hyperterminal. Let h_1 and h_2 be two hyperterminals that belong to the same net N_i. For every terminal t_{ij} in h_1 and for every terminal t_{ik} in h_2 there is a corresponding edge (h_1, h_2) in E, and the weight of this edge $w((h_1, h_2))$ is the interval $[j, k]$ (assume that $j \leq k$, otherwise, it will be $[k, j]$). Clearly, if h_1 contains p_1 terminals and h_2 contains p_2 terminals, then there are $p_1 \times p_2$ parallel edges connecting h_1 and h_2 in G. Furthermore, corresponding to each net in I there is a connected component in G.

For example, the connected component corresponding to net 3 in the example in Figure 10.4 is shown in Figure 10.6. Given an instance I of the net segment selection problem, since all the hyperterminals in the same net are to be connected together for every net in I, it is necessary to find a spanning forest of $CG(I)$. Moreover, since the objective is to minimize the channel density, the density of the set of intervals associated with the edges in the spanning forest must be minimized.

Therefore, the net segment selection problem can be formulated as Minimum Density Spanning Forest Problem (MDSFP). Given a weighted connection graph $G = (V, E)$ and an integer D, determine a subset of edges $E' \subseteq E$ that form a spanning forest of G, and the density of the interval set $\{w(e)|e \in E'\}$ is no more than D.

In [CL90], it was shown that this problem is computationally hard.

Theorem 17 *The minimum density spanning forest problem is NP-complete.*

In view of NP-completeness of the MDSFP, an efficient heuristic algorithm has been developed for solving the net segment selection problem [CL90]. The

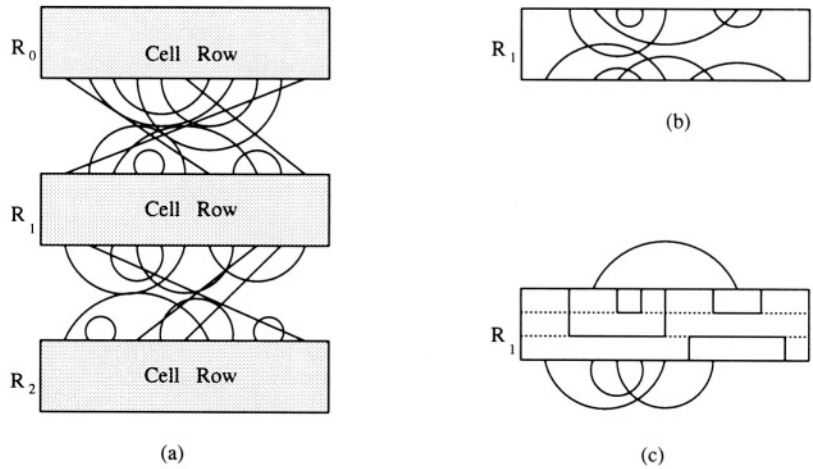

Figure 10.7: An example of the TRMPS problem

heuristic algorithm works as follows. Given an instance I of the net segment selection problem, a connection graph $G = (V, E)$ is constructed. For each edge $e \in E$, the relative density of e, called $RD(e)$, is defined to be $d(e)/d(E)$, where $d(e)$ is the density of the set of intervals which intersect with the interval $w(e)$, and $d(E)$ is the density of the interval set $\{w(e)|e \in E\}$. The relative density of an edge measures the degree of congestion over the interval associated with the edge. The algorithm repeatedly removes edges from E until a spanning forest is obtained.

10.1.2.2 Planar Over-the-Cell Routing

In [DMPS94] Danda, Madhwapathy, Panyam and Sherwani presented an algorithm to select a maximum planar subset of nets in M2 in BTM standard cell designs.

Figure 10.7(b) shows the set of nets that are suitable for routing over the cell row R_1. In a HCVC (Horizontally Connected Vertically Connected) model [CPL93], the main problem in two layer over the cell routing is to select a maximum planar subset of nets which are suitable for routing in a single layer, available over the cell rows. The remaining connections are completed in the channel. Authors call this problem as the Two Row Maximum Planar Subset (TRMPS) problem. Figure 10.7(c) shows the maximum planar subset of nets, that can be routed over R_1, which is an optimal solution, for the instance of the TRMPS problem, shown in Figure 10.7(b). Notice, that the tracks are shared between the top row nets and the bottom row nets, so as to efficiently utilize the over-the-cell area.

The TRMPS problem, is formally defined as follows. Given two rows of terminals $\mathcal{T} = \{t_1, t_2, \ldots, t_L\}$ and $\mathcal{B} = \{b_1, b_2, \ldots, b_L\}$ and two sets of nets

$\mathcal{N}_\mathcal{T} = \{(t_i, t_j) \mid t_i, t_j \in \mathcal{T}\}$ and $\mathcal{N}_\mathcal{B} = \{(b_i, b_j) \mid l_i, l_j \in \mathcal{B}\}$, where $|\mathcal{N}_\mathcal{T} \cup \mathcal{N}_\mathcal{B}| = n$,and k tracks between the two rows, find the maximum planar subset $\mathcal{N}_\mathcal{P} \subseteq \mathcal{N}_\mathcal{T} \cup \mathcal{N}_\mathcal{B}$ of the two sets in k tracks. Authors presented a dynamic programming approach to solve this problem.

Let L denote the total number of columns in a cell row, numbered from left to right. In BTM-HCVC, the terminals are located at the intersection points of the upper or the lower horizontal boundaries of a cell row and the vertical columns. If a terminal is not used by any net, then that terminal is called a *vacant terminal.* If both the upper and lower terminals of a column are vacant, then that column is called a *vacant abutment.* The total number of tracks available in the OTC area of a cell row, for routing, is denoted by k (cell height), and the tracks are numbered from top to bottom. Then, an instance of the TRMPS problem can be formally represented as a 7-tuple $\mathcal{I} = (\mathcal{T}, \mathcal{B}, \mathcal{N}_\mathcal{T}, \mathcal{N}_\mathcal{B}, k, n, L)$. An instance of the TRMPS problem is called as a *Canonical Instance,* if there are no vacant abutments in that instance. If n is the number of nets in a canonical instance $\mathcal{I}$, then the number of columns (L), can be at most $2n$. This is because, in the worst case, each column has at most one vacant terminal, either in the top or the bottom terminal row.

Canonical instances with two terminal nets are considered as input to the problem. A net is denoted by a pair of terminals. A net (t_i, t_j), where $1 \leq i, j \leq L$, is called a top net. Similarly, a net (b_i, b_j), where $1 \leq i, j \leq L$, is called a bottom net. *span* of a two terminal net is defined as the absolute difference between the column numbers on which the terminals of the net are located. For example, the span of the net $N_\alpha = (t_i, t_j)$, is given by,

$$span(N_\alpha) = \mid i - j \mid$$

A region $\mathcal{R}_m$ of a cell row is defined as a rectangular region of the cell row, containing the columns in the range [1,m], where $1 \leq m \leq L$. A net (t_i, t_j) (or (b_i, b_j)), is said to be *completely contained* in the region $\mathcal{R}_m$, if $1 \leq i, j \leq m$.

Let $T(j)$ denotes the optimal TRMPS solution in a rectangular region $\mathcal{R}_j$. The $T(j)$ solution is computed for all j, $1 \leq j \leq L$, using a dynamic programming technique. Finally, the $T(L)$ solution gives the optimal solution, for a given instance $\mathcal{I}$ of the TRMPS problem. In order to compute the $T(j)$ solution, the region $\mathcal{R}_j$ is partitioned into two or three sub regions. depending on the existence of top nets and bottom nets, completely contained in $\mathcal{R}_j$, with one of their terminals at column j, as shown in Figure 10.8.

Let $N_\alpha = (t_i, t_j)$ be the only net with a terminal at column j, and which is completely contained in $\mathcal{R}_j$. In this case, $\mathcal{R}_j$ is divided into an L-shaped region R, and a rectangular region r which consists of a single row of terminals (Figure 10.8(a)). The optimal $T(j)$ solution, may or may not contain N_α. If N_α is included, then the $T(j)$ is summation of the optimal solutions in the L-shaped region R and the rectangular region r, and the net N_α itself. If N_α is not included, then the $T(j)$ solution is the same as the $T(j-1)$ solution. The maximum of the above two solutions, is taken as the optimal $T(j)$ solution.

Let $N_\alpha = (t_i, t_j)$ and $N_\beta = (b_m, b_j)$ be the nets with terminals at column j, and which are completely contained in $\mathcal{R}_j$. Then, the optimal $T(j)$ solution

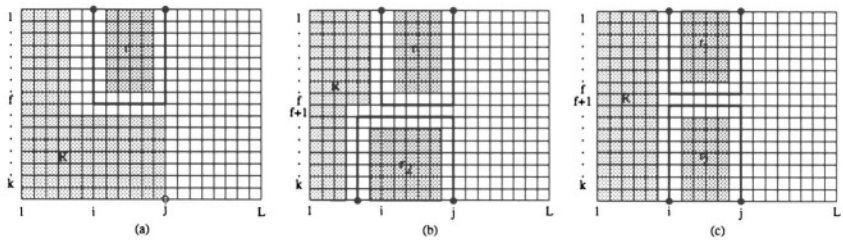

Figure 10.8: Schematic Overview of the Algorithm ALGO-TRMPS

may include

1. **None of the nets N_α and N_β:** In this case, the $T(j)$ solution is the same as the $T(j-1)$ solution.

2. **Only the net N_α:** In this case, the $T(j)$ solution can be computed as shown in Figure 10.8(a).

3. **Only the net N_β:** In this case also, the $T(j)$ solution can be computed as shown in Figure 10.8(a).

4. **Both the nets N_α and N_β:** In this case, if $i \neq m$, then $\mathcal{R}_j$ is partitioned into an L-shaped region R, and two rectangular regions r_1 and r_2, which consist of a single row of terminals (Figure 10.8(b)). If $i = m$, then $\mathcal{R}_j$ is partitioned into a rectangular region R, which consists of two rows of terminals, and two rectangular regions r_1 and r_2, which consist of a single row of terminals (Figure 10.8(c)). Then, the $T(j)$ is simply summation of the optimal solutions in the regions R, r_1 and r_2, and the nets N_α and N_β.

The optimal $T(j)$ solution, is the maximum among all the above four solutions.

From the above discussion, it is clear that, the single row solutions and the solutions in the L-shaped regions need to be computed, before computing the two row solutions. Our algorithm consists of the following three phases.

1. In the first phase, single row solutions of the terminal rows $\mathcal{T}$ and $\mathcal{B}$, are computed individually. Each single row solution of a terminal row, is an (i, j, t) solution, where $1 \leq i, j \leq L$ and $1 \leq t \leq k$. These solutions are denoted as $S_t(i, j, t)$ and $S_b(i, j, t)$ for top and bottom terminals respectively.

2. In this phase, the maximum two row planar subset $T(j)$ for the given terminal rows is computed, where $1 \leq j \leq L$ by using a dynamic programming approach. Here, the $S_t(i, j, t)$ and $S_b(i, j, t)$ solutions, computed in the first phase will be used. As described above, finding the $T(j)$ solution also involves finding the maximum planar subset in L-shaped regions.

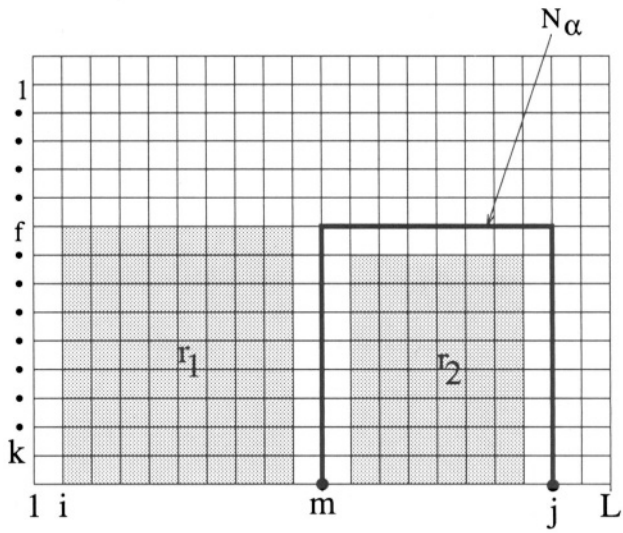

Figure 10.9: Single Row Maximum Independent Set

3. The solution obtained in phase 2, gives the number of nets in the optimal solution for a given instance of the TRMPS problem. In this phase, the actual planar subset of nets in the optimal solution, is determined by backtracking.

From a routing perspective, this problem is equivalent to assigning the maximum number of intervals to k tracks such that, if interval (i,j) is assigned to track f, then no interval assigned to tracks $1, 2, \ldots, f-1$ should intersect columns i and j. Let *MIS(i,j,f)* denote the solution of the OFPR problem resulting from restricting the intervals to be in the range of $[i,j]$ and allowing f tracks for routing, where $1 \leq i,j \leq L$ and $1 \leq f \leq k$. The (i,j,f) solution is computed using dynamic programming. Notice that, computation of $MIS(i,j,f)$ can be any of the following cases.

1. If j is vacant, then

$$MIS(i,j,f) = MIS(i,j-1,f)$$

2. There exists a net N_α with terminals j and m but $m \notin [i,j)$. Then,

$$MIS(i,j,f) = MIS(i,j-1,f) \text{ if } m \notin [i,j)$$

3. There exists a net N_α with terminals j and m such that $m \in [i,j)$, then the following two cases are possible:

(a) Excluding the net N_α in the solution leads to

$$MIS(i,j,f) = MIS(i,j-1,f)$$

(b) Including the net N_α in the solution results in

$$MIS(i,j,f) = MIS(i,m-1,f) + MIS(m+1,j-1,f-1) + 1$$

As shown in Figure 10.9, if $m \in [i,j)$, one has to check if including N_α will lead to a better solution or not. Therefore,

$$\begin{aligned} MIS(i,j,f) = \quad & \max \quad \{MIS(i,j-1,f), MIS(i,m-1,f) + \\ & MIS(m+1,j-1,f-1) + 1 \text{ if } j' \in [i,j)\} \end{aligned}$$

The complexity of this algorithm is given by the following theorem, stated in [CPL93].

Theorem 18 *[CPL93] The two-terminal net OFPR problem can be solved in $O(kn^2)$ time, where n is the number of nets and k is the number of available tracks.*

Using the above algorithm the maximum k-planar subsets S_t and S_b are computed, for the top and bottom terminal rows respectively, and all the intermediate solutions are stored.

Since, computing the $T(j)$ solution, involves computing the solutions in L-shaped regions, let us discuss a scheme to represent an L-shaped region.

Figure 10.10 shows two types of L shaped regions. For instance, an L-shaped region shown in Figure 10.10(a), is denoted by the 3-tuple (i,j,f), where

1. i is the column number of the terminal t_i, which is the rightmost corner of the L-shaped region, in the top terminal row.
2. j is the column number of the terminal b_j, which is the rightmost corner of the L-shaped region, in the bottom terminal row.
3. f is the track, that forms part of the horizontal boundary of the L-shaped region (See Figure 10.10(a)).

The maximum planar subset in the L-shaped region, shown in Figure 10.10(a), is denoted by L(i,j,f). Following the same convention described above, the inverted L-shaped region, shown in Figure 10.10(b) is denoted by (j,i,f), and the solution in this region is denoted by L(j, i,f). The method of computing solutions in L-shaped regions will be described later.

While computing the $T(j)$ solution in the rectangular region $\mathcal{R}_j$, the algorithm deals with the following three cases.

Case 1: There exists a top net $N_\alpha = (t_i, t_j)$, which is completely contained in $\mathcal{R}_j$ (Figure 10.11(a)).

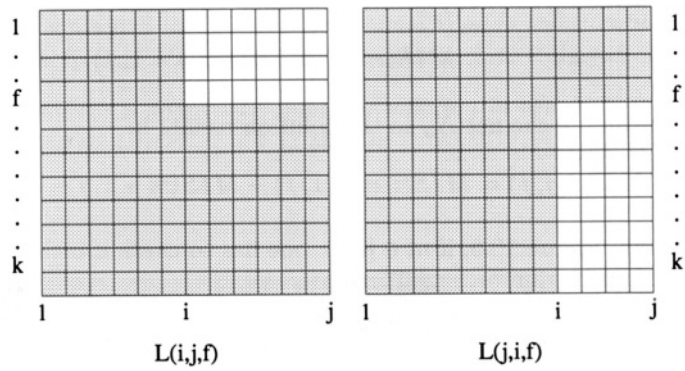

Figure 10.10: L-shaped regions

Case 2: There exists a bottom net $N_\beta = (b_i, b_j)$, which is completely contained in $\mathcal{R}_j$(Figure 10.11(b)).

Case 3: There exists a top net $N_\alpha = (t_l, t_j)$, and a bottom net $N_\beta = (b_m, b_j)$, which are completely contained in $\mathcal{R}_j$(Figure 10.11(c)).

Let us consider each of the above listed cases in detail.

Case 1: Depending on whether the net N_α is in the optimal $T(j)$ solution, or not, the algorithm has to deal with the following sub-cases.

Case 1(a): Excluding the net N_α leads to

$$T(j) = T(j-1)$$

Case 1(b): If the net N_α is included, such that, it is assigned to a track f, $1 \leq f \leq k$, then the following solution, which is denote by $T'(j)$.

$$T'(j) = S_t(i+1, j-1, f-1) + 1 + L(i-1, j-1, f+1)$$

By considering all possible track assignments, the track to which N_α can be assigned is found, so as to maximize the $T(j)$ solution. Then, the $T(j)$ solution obtained by choosing N_α, which is denoted as $T''(j)$, is given by,

$$T''(j) = \max_{f=1}^{k}\{T'(j)\}$$

The optimal $T(j)$ solution will then be the maximum of the two solutions obtained by including and excluding the net N_α. Therefore,

$$T(j) = \max\{T(j-1), T''(j)\}$$

Case 2: This is symmetric to Case 1.

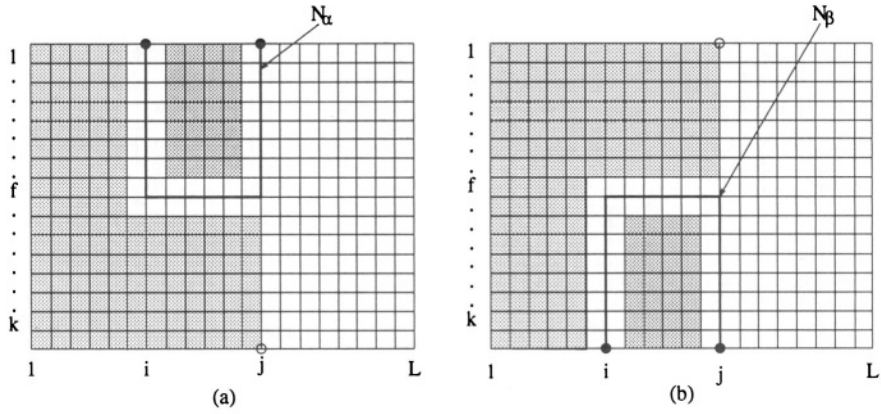

Figure 10.11: Cases 1 and 2 in ALGO-TRMPS

Case 3: Here, the following three sub-cases are possible as shown in Figure 10.12.

Case 3(a): $Span(N_\alpha) > Span(N_\beta)$

Case 3(b): $Span(N_\alpha) < Span(N_\beta)$

Case 3(c): $Span(N_\alpha) = Span(N_\beta)$

For each of the above three sub-cases, the following four solutions are computed.

$W_0(j)$: Two row solution of $\mathcal{R}_j$, which does not consist of the nets N_α and N_β

$W_1(j)$: Two row solution of $\mathcal{R}_j$, which consists of only the net N_α

$W_2(j)$: Two row solution of $\mathcal{R}_j$, which consists of only the net N_β

$W_{12}(j)$: Two row solution of $\mathcal{R}_j$, which consists of both the nets N_α and N_β

The maximum of W_0, W_1, W_2 and W_{12} solutions is the optimal $T(j)$ solution. If both the nets N_α and N_β are included in the optimal solution $T(j)$, then a simple observation, regarding the track assignment of the nets N_α and N_β, is stated in the following lemma.

Lemma 1 *If $N_\alpha = (t_l, t_j)$ and $N_\beta = (b_m, b_j)$, are two nets, which are completely contained in $\mathcal{R}_j$, and the optimal $W_{12}(j)$ solution has the net N_α in track f_1, and N_β in track f_2 such that $1 \leq f_1 < f_2 \leq k$, then,*

1. *if $span(N_\alpha) > span(N_\beta)$, then, the solution in which, the net N_β is assigned to a track $f_1 + 1$ is also an optimal $W_{12}(j)$ solution.*

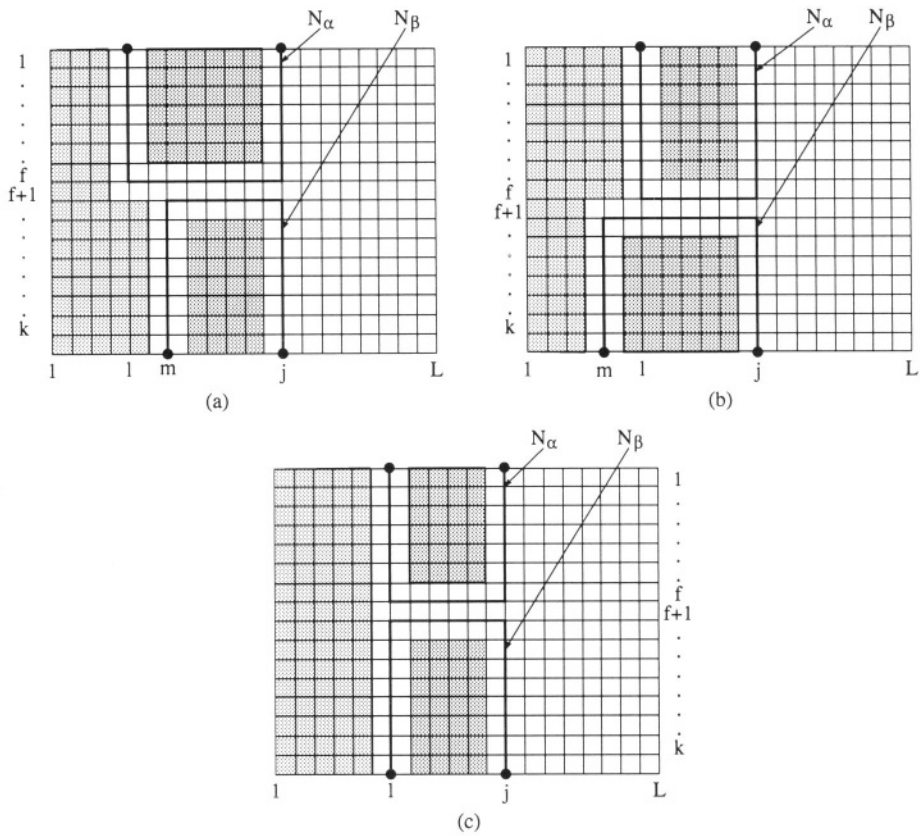

Figure 10.12: Cases 3(a),(b) and (c)

2. *if* $span(N_\alpha) < span(N_\beta)$, *then, the solution in which, the net* N_α *is assigned to a track* $f_2 - 1$ *is also an optimal* $W_{12}(j)$ *solution.*
3. *if* $span(N_\alpha) = span(N_\beta)$, *then, the solution in which, the net* N_β *is assigned to a track* $f_1 + 1$, *and the solution in which, the net* N_α *is assigned to a track* $f_2 - 1$ *are also optimal* $W_{12}(j)$ *solutions.*

Let us now consider the three sub-cases listed above, in detail.

Case 3(a): In this case, since $span(N_\alpha) > span(N_\beta)$, column l is to the left of column m (Figure 10.12(a)). The $W_0(j)$ solution, in which both the nets are excluded is given by,

$$W_0(j) = T(j-1)$$

The $W_1(j)$ solution can be computed as follows. Suppose, the net N_α is assigned to track f, $1 \leq f \leq k$, then, the following solution, which is

called as $W_1'(j)$.

$$W_1'(j) = S_t(l+1, j-1, f-1) + 1 + L(l-1, j-1, f+1)$$

By trying all possible track assignments, the track to which N_α can be assigned is found, so as to maximize the $W_1(j)$ solution. The $W_1(j)$ solution is given by,

$$W_1(j) = \max_{f=1}^{k}\{W_1'(j)\}$$

The $W_2(j)$ solution can be computed in a similar manner as $W_1(j)$.

The $W_{12}(j)$ solution can be computed as follows. From lemma 1, it is clear that in the optimal $W_{12}(j)$ solution, the nets N_α and N_β are assigned to adjacent tracks. Suppose, the net N_α is assigned to track f, and N_β in track $f+1$, then the following solution, which is called as W_{12}' is obtained.

$$\begin{aligned} W_{12}'(j) &= L(l-1, m-1, f+1) + S_t(l+1, j-1, f-1) \\ &+ S_b(m+1, j-1, f+2) + 2 \end{aligned}$$

The adjacent tracks, to which N_α and N_β can be assigned is found, so as to maximize $W_{12}(j)$.

$$W_{12} = \max_{f=1}^{k-1}\{W_{12}'(j)\}$$

Then, the optimal $T(j)$ solution will be the maximum of W_0, W_1, W_2 and W_{12} solutions. Therefore,

$$T(j) = \max\{W_0(j), W_1(j), W_2(j), W_{12}(j)\}$$

Case 3(b): This is symmetric to Case 3(a).

Case 3(c): In this case, $span(N_\alpha) = span(N_\beta)$. (Figure 10.12(c)). Here, the $W_0(j), W_1(j)$ and $W_2(j)$ solutions are the same as for Case 3(a) and Case 3(b) However, the W_{12} solution differs slightly. According to the Lemma 1, the nets N_α and N_β can be assigned to adjacent tracks (say f and $f+1$ respectively). Then the W_{12}' will be

$$\begin{aligned} W_{12}'(j) &= T(l-1) + S_t(l+1, j-1, f-1) \\ &+ S_b(l+1, j-1, f+2) + 2 \end{aligned}$$

By trying all possible track assignments, one can find two adjacent tracks, on which N_α and N_β can be placed so as to maximize the W_{12} solution. Therefore,

$$W_{12}(j) = \max_{f=1}^{k-1}\{W_{12}'(j)\}$$

Then the optimal solution is given by,

$$T(j) = \max\{W_0(j), W_1(j), W_2(j), W_{12}(j)\}$$

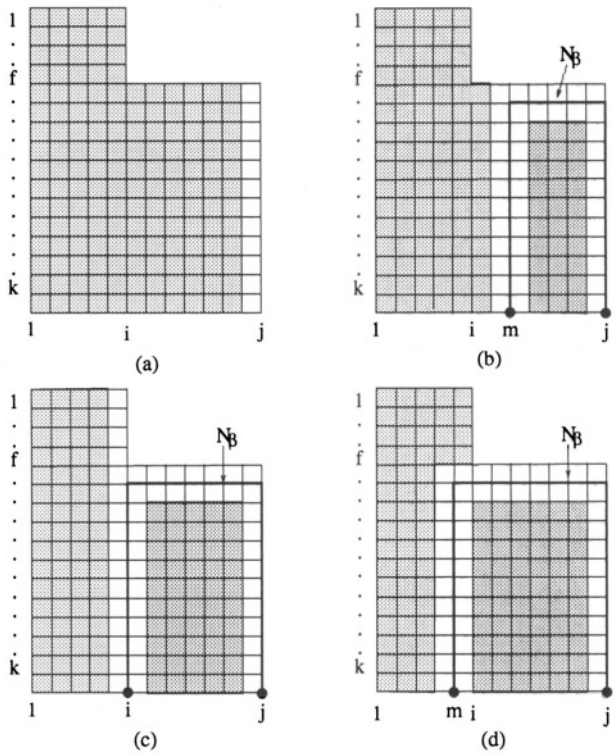

Figure 10.13: The four cases of L-shaped solutions

Authors have the following theorems on the time complexity and optimality of ALGO-TRMPS.

Theorem 19 *The time complexity of ALGO-TRMPS is* $O(kn^2 \times f(k,n))$, *where n is the number of nets, k is the number of tracks available on over-the-cell area and* $f(k,n)$ *is the time to compute solution in each L-shaped region.*

Theorem 20 *Given an instance* $\mathcal{I}$, *of TRMPS problem, ALGO-TRMPS produces an optimal solution.*

In the following paragraphs a detailed description of computing the maximum planar subset in an L-shaped region is given.

The $L(i,j,f)$ solutions can be classified into the following four types depending on the existence of a bottom net which is completely contained in R_j, with b_j as one of its terminals.

Case 1: There is no bottom net, which is completely contained in R_j, with b_j as one of its terminals (Figure 10.13(a)). In this case

$$L(i,j,f) = L(i,j-1,f)$$

Case 2: There is a net $N_\beta = (b_m, b_j)$, which is completely contained in R_j, such that, $span(N_\beta) < (j - i)$, i.e., column m is to the right of column i, as shown in Figure 10.13(b). Excluding the net N_β leads to,

$$L(i, j, f) = L(i, j - 1, f)$$

Let us assume that, the L(i,j,f) solution that includes the net N_β, is maximum, by assigning N_β to track f_1, such that $f_1 \geq f$. Also notice that the optimal L(i, j,f) solution cannot consist of any other nets, that lie entirely in the L-shaped region, represented by (i,j, f), in the shaded area shown in Figure 10.13(b). If any such net exists, then the $L(i,j,f)$ solution, which includes the net N_β, would not be planar. Therefore the $L(i,j,f)$ solution remains maximum, even if $N_\beta$is assigned to track f_2, such that $f < f_2 < f_1$. Therefore, one can assign N_β to track f +1. Now, the L(i, j,f) solution, which includes N_β, consists of

1. the nets enclosed by N_β, which is the single sided solution $S_b(m + 1, j - 1, k - f - 1)$.
2. the net N_β itself, and
3. The solution of the L-shaped region, represented by L(i – l,m –l, f).

The L(i, j, f) solution that includes N_β, which is denoted as $L'(i, j, f)$ is given by

$$\begin{aligned} L'(i, j, f) &= S_b(m + 1, j - 1, f + 2) + 1 \\ &+ L(i, m - 1, f) \end{aligned}$$

The optimal $L(i,j,f)$ solution will be, the maximum of the solutions obtained by excluding and including the net N_β. Therefore,

$$L(i, j, f) = \max\{L(i, j - 1, f), L'(i, j, f)\}$$

Case 3: There is a net $N_\beta = (b_m, b_j)$, which is completely contained in R_j, such that, $span(N_\beta) = (j - i)$, i.e., column m and column i are the same, as shown in Figure 10.13(c). This is similar to the Case 1, except that, the L(i,j,f) solution, which includes the net N_β, consists of the single row solution, in the region enclosed by N_β, the net N_β, and the two row solution $T(i - 1)$. Therefore, the L(i, j,f) solution is given by,

$$\begin{aligned} L(i, j, f) &= \max\{L(i, j - 1, f), \\ &\quad S_b(i + 1, j - 1, f + 2) + 1 + T(i - 1)\} \end{aligned}$$

Case 4: There is a net $N_\beta = (b_m, b_j)$, which is completely contained in R_j, such that, $span(N_\beta) > (j - i)$, i.e., column m is to the left of column i, as shown in Figure 10.13(d). Excluding the net N_β leads to,

$$L(i, j, f) = L(i, j - 1, f)$$

Suppose the net N_β is assigned to track $f_1, f < f_1 \leq k$, then The $L(i, j, f)$ solution, that includes the net N_β, in track f_1, $f < f_1 \leq k$, denoted by $L'(i, j, f)$ is consists of

1. the nets enclosed by N_β, which is the single sided solution $S_b(m + 1, j - 1, k - f_1 - 1)$.
2. the net N_β itself, and
3. The solution of the L-shaped region, represented by $(i, m - 1, f)$.

Therefore the $L(i,j, f)$ solution, which includes N_β in track f_1 is given by

$$\begin{aligned} L'(i,j,f) &= S_b(m+1, j-1, k-f_1) + 1 \\ &+ L(i, m-1, f) \end{aligned}$$

By varying f from $f + 1$ to k, one can find the track, to which N_β can be assigned, so as to maximize the $L(i,j,f)$ solution. Then, the $L(i,j,f)$ solution by choosing N_β, which is denoted $L''(i, j, f)$ is given by

$$L''(i,j,f) = \max_{f_1=f+1}^{k} \{L'(i,j,f_1)\}$$

The optimal $L(i,j,f)$ solution will be, the maximum of the solutions obtained by excluding and including the net N_β. Therefore

$$L(i,j,f) = \max\{L(i, j-1, f), L''(i,j,f)\}$$

The solutions in an inverted L-shaped region (where $i > j$), can also be computed in a similar manner.

The computation of each $T(j)$ solution, involves the computation of solutions in several L-shaped regions. Therefore, the worst case running time of the algorithm ALGO-TRMPS, depends on the the number of L-shaped regions. The following lemma is on the number of L-shaped regions.

Lemma 2 *In canonical representation the number of L-shaped regions is $O(kn^2)$, where k is the number of tracks and n is the number of nets.*

Lemma 3 *Each $L(i,j,f)$ solution ,where $1 \leq i, j \leq L$ and $1 \leq f \leq k$, is computed once and it takes constant time to compute the solution.*

Theorem 21 *The computation time of ALGO-LMPS is $O(kn^2)$, where k is the number of tracks in a cell row, and n is the number of nets.*

Theorem 22 *Given an Instance $\mathcal{I}$, ALGO-LMPS produces an optimal solution.*

```
Algorithm ALGO-TRMPS(N_T, N_B, n, T, B, N, L)
Begin
Compute_SRMPS_l();
Compute_SRMPS_u();
for  j = 1 to L
      case(net_type(j)):
           Type 1: TRMPS(j) = T1;
           Type 2: TRMPS(j) = T2;
           Type 3: case(nets_at(j))
                type a: TRMPS(j) = T3a
                type b: TRMPS(j) = T3b
                type c: TRMPS(j) = T3c
End(For)
N_P = Φ
for  j = L to 1
      if(T(j-1) < T(j))
           if(type = 1) N_P = N_P ∪ N_α
           else if(type = 2) N_P = N_P ∪ N_β
           else if(type = 3) N_P = N_P ∪ N_α ∪ N_β
      End(if)
End(for)
End;
```

Figure 10.14: Algorithm ALGO-TRMPS

Theorem 23 *Given an instance $\mathcal{I}$, ALGO-TRMPS provides an optimal solution to the two row maximum planar subset problem.*

Theorem 24 *The complexity of the ALGO-TRMPS is $O(kn^2)$, where k is the number of tracks available over-the-cell area and n is the number of nets.*

Figure 10.14 presents the algorithm formally.

10.1.2.3 Over-the-Cell Routing Using Vacant Terminals

In [HSS93], Holmes, Sherwani and Sarrafzadeh presented a new algorithm called WISER, for over-the-cell channel routing. There are two key ideas in their approach: use of vacant terminals to increase the number of nets which can be routed over the cells, and near optimal selection of 'most suitable' nets for over the cell routing. Consider the example shown in Figure 10.15(a). Four tracks are necessary using a conventional channel router or an over-the-cell router. However, using the idea of vacant terminals, a two-track solution can be obtained (see Figure 10.15(b)). Furthermore, it is clear that the selection of nets which minimize the maximum clique, $h_{\max}$, in horizontal constraint graph

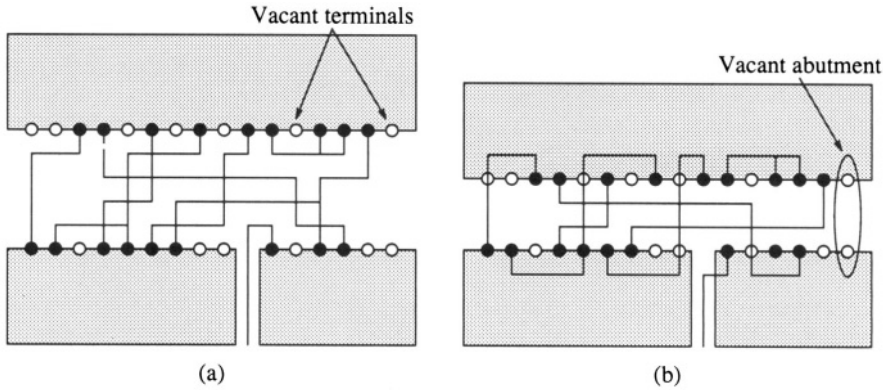

Figure 10.15: Effect of using vacant terminals in layout.

is not sufficient to minimize the channel height. For example, channel height for the routing problem shown in Figure 10.15 is determined strictly by v_{max}, that is, longest path in the VCG (vertical constraint graph). Thus, the nets which cause long paths in VCG should be considered for routing over the cells to obtain a better over the cell routing solution.

An informal description of each of the six steps of algorithm is given below.

1. **Net Classification:** Each net is classified as one of three types which, intuitively, indicates the difficulty involved in routing this net over the cells.

2. **Vacant Terminal and Abutment Assignment:** Vacant terminals and abutments are assigned to each net depending on its type and weight. The weight of a net intuitively indicates the improvement in channel congestion possible if this net can be routed over the cells.

3. **Net Selection:** Among all the nets which are suitable for routing over the cells, a maximum weighted subset is selected, which can be routed in a single layer.

4. **Over-the-Cell Routing:** The selected nets are assigned exact geometric routes in the area over the cells.

5. **Channel Segment Assignment:** For multi-terminal nets, it is possible that some net segments are not routed over the cells, and therefore, must be routed in the channel. In this step, 'best' segments are selected for routing in the channel to complete the net connection.

6. **Channel Routing:** The segments selected in the previous step are routed in the channel using a greedy channel router.

The most important steps in algorithm WISER are net classification, vacant terminal and abutment assignment, and net selection. These steps are discussed in detail below. Channel segment assignment is done using an algorithm similar to the one presented in [CL90]. The channel routing completed by using a greedy channel router [RF82].

Vacant Terminals and Net Classification: The algorithm WISER was developed to take advantage of the physical characteristics indigenous to cell-based designs. One such property is the abundance of vacant terminals. A terminal is said to be *vacant* if it is not required for any net connection. Examination of benchmarks and industrial designs reveals that most standard cell designs have 50% to 80% vacant terminals depending on the given channel. A pair of vacant terminals with the same x-coordinate forms a *vacant abutment* (see Figure 10.15). In the average case, 30% - 70% of the columns in a given input channel are vacant abutments. The large number of vacant terminals and abutments in standard cell designs is due to the fact that each logical terminal (inputs and outputs) is provided on both sides of a standard cell but, in most cases, need only be connected on one side. It should be noted that the actual number of vacant terminals and abutments and their locations cannot be obtained until global routing is completed.

To effectively utilize the vacant terminals and abutments available in a channel, algorithm WISER categorizes nets according to the proximity of vacant terminals and abutments with respect to net terminals. Before classification, each k-terminal net N_i is decomposed into $k-1$ two-terminal nets at adjacent terminal locations. Let $N_i = \{t_{b1}, t_{b4}, t_{t4}, t_{t6}\}$ be a four-terminal net. The notation t_{rx} is used to refer to the terminal on row r (top or bottom) at column x, Net N_i is decomposed into 3 two-terminal nets: $N_{i_1} = (t_{b1}, t_{b4})$, $N_{i_2} = (t_{b4}, t_{t4})$, and $N_{i_3} = (t_{t4}, t_{t6})$. Each two-terminal net $N_j = (t_{r_1x_1}, t_{r_2x_2})$ where $r_1, r_2 \in \{t, b\}$, $x_1, x_2 \in Z^+$, and $x_1 \leq x_2$ is then classified as a type I, type II or a type III net. The type of a net intuitively indicates the difficulty involved in routing that net over the cell rows. In other words, type III nets are hardest to route, while type I are easiest to route over the cells.

Definition 1 *Net* $N_j = (t_{r_1x_1}, t_{r_2x_2})$ *is a* type I *net if* $r_1 = r_2$, *and at least one of the terminals* $t_{r_1x_1}$ *and* $t_{r_2x_2}$ *is not vacant.*

Definition 2 *Net* $N_j = (t_{r_1x_1}, t_{r_2x_2})$ *is a* type II *net if the terminals* $t_{r_1x_1}$ *and* $t_{r_2x_2}$ *are both vacant.*

Definition 3 *Net* $N_j = (t_{r_1x_1}, t_{r_2x_2})$ *is a* type III *net if* $r_1 \neq r_2$, *neither* $t_{r_1x_1}$ *nor* $t_{r_2x_2}$ *is vacant, and there exists at least one vacant abutment a within the span of* N_j, $x_1 < a < x_2$.

The three net types are illustrated in Figure 10.16. A typical channel of a standard cell design contains about 44% type I nets, 41% type II nets, and 10% type III nets.

Observing that type I and type II nets constitute a majority of nets in the channel, one might suggest that it is sufficient to consider only these net types

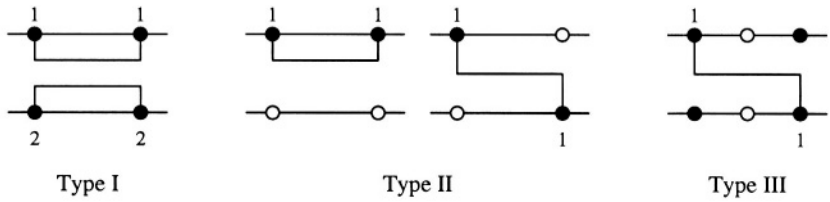

Figure 10.16: Net types.

when routing over the cells rows. However, this is not the case, since removing type III nets from the channel is critical in minimizing the length of the longest path ($v_{\max}$) in the vertical constraint graph.

The basic algorithm given in section 10.1.2.1 for over-the-cell channel routing attempts to minimize only the density due to horizontal constraint graph. Since channel height depends on both $h_{\max}$ and $v_{\max}$, it is clear that using $h_{\max}$ as the sole criterion for selecting nets is not as effective. In WISER, a net weightingfunction, $F_w : \mathcal{N} \rightarrow R^+$, which incorporates both the channel density and *VCG* path length criteria is used to assess the suitability of a given net for over-the-cell routing. The weight of a net $N_j = (t_{r_1 x_1}, t_{r_2 x_2})$ is computed based on the relative density of the channel in the interval $[x_1, x_2]$ and the ancestor and descendant weights of the net *n*. The relative density of net N_j can be computed by $r_d(N_j) = \frac{l_d(N_j)}{h_{\max}}$ where $l_d(N_j)$ is the maximum of the local densities at each terminal location t where $x_1 \leq t \leq x_2$. The ancestor weight of a net N_j, denoted by $a(N_j)$, is the length of the longest path from a node t in the vertical constraint graph with zero in-degree to the node N_j, and the descendant weight of N_j, denoted by $d(N_j)$, is the length of the longest path from N_j to a node s in VCG with zero out-degree. The general net weighting function is given below:

$$F_w(N_j) = k_1 \frac{r_d(N_j)}{v_{\max}} + k_2 \frac{(a(N_j) + d(N_j) - \mid a(N_j) - d(N_j) \mid)}{h_{\max}}$$

where k_1 and k_2 are experimentally determined constants. Since the weight of a net N_j indicates the reduction possible in $h_{\max}$ and $v_{\max}$ if N_j is routed over the cell rows, the 'best' set of nets to route over the cells is one with maximum total weight.

Vacant Terminal and Abutment Assignment: After classification and weighting, nets are allocated a subset of vacant terminals or vacant abutments, depending on their type, to help define their routing paths in the area over the cell rows. It should be noted that type I nets, which have both of their terminals on the same boundary of the channel, can be routed in the area over the cells without using vacant terminals as shown in Figure 10.17. Therefore, the vacant terminal/abutment assignment problem is a matter of concern only for type II

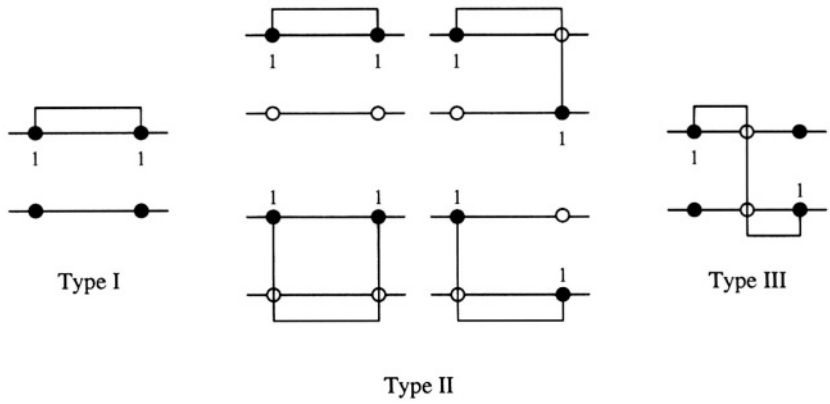

Figure 10.17: Routing of different net types.

and type III nets. For type II nets, the vacant terminals are 'reserved' for a net. That is, only a particular net may use a particular vacant terminal; as a result, vacant terminal assignment for type II nets is actually a net selection problem. On the other hand, a type III net may use any abutment within its span and therefore vacant abutment assignment problem for type III nets can be viewed as a matching problem.

Theorem 25 *The vacant terminal assignment problem for type II nets is NP-complete.*

Using theorem 25, it can be shown that the problem of finding an optimal routing using only k tracks in over-the-cell area is also NP-complete. However, if the value of k is restricted to one ($k = 1$), the problem is reduced to finding a maximum-weighted bipartite subgraph in an interval graph, which can be solved in polynomial time. The complexity of the problem for a fixed k (k being a small constant), however, for arbitrary k, the following result can be established.

Theorem 26 *The vacant abutment assignment problem for type III nets is NP-complete.*

Corollary 3 *The vacant terminal assignment problem for type II nets remains NP-complete when the number of tracks available over each cell row is restricted to k.*

In view of NP-completeness of the vacant abutment assignment problem for type III nets, a greedy heuristic is used. This heuristic is based on certain necessary conditions for the routability of a pair of type III nets. These necessary conditions are depicted in Figure 10.18. These necessary conditions basically check the planarity of pairs of nets. The formal description of the algorithm

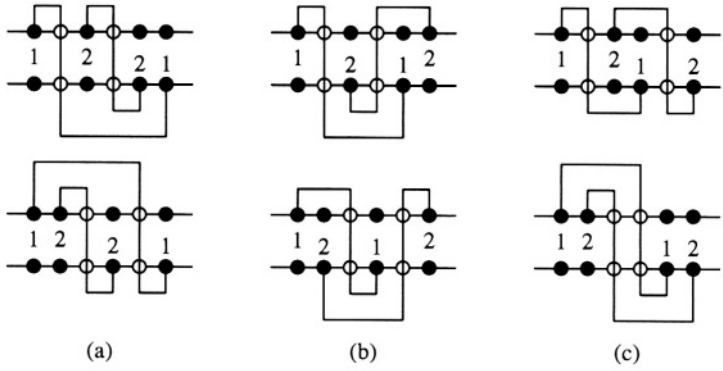

Figure 10.18: Necessary conditions.

is given in Figure 10.19. The main idea of this algorithm is to assign vacant abutments to nets according to their weight. The 'heaviest' nets are considered first. It is easy to see that the algorithm ASSIGN-ABUTMENTS produces a feasible solution in $O(dn^2)$ time.

Provably Good Algorithm for Net Selection: The net selection problem can be stated as follows. Given a set $\mathcal{N}$ of of nets, select a maximum-weighted subset of nets $\mathcal{N}' \subseteq \mathcal{N}$, such that all the nets in $\mathcal{N}'$ can be routed in the area over the cell rows in planar fashion. Algorithm WISER uses a graph theoretic approach to net selection. An overlap graph G_O is constructed for intervals of nets in set $\mathcal{N}$. It is easy to see that net selection problem reduces to the problem of finding a maximum-weighted bipartite subgraph $B_{\max}$ in the overlap graph G_O. However, the density of the nets in each partite set must be bounded by a constant k, which is the number of tracks available in the over the cell region. The problem of computing $B_{\max}$ is known to be NP-complete [SL89a]. As a result, a provably good algorithm is used for net selection. This algorithm is guaranteed to find a solution within 75% of the optimal.

Let P_t, P_b denote the partite sets of the graph B. The vertices of P_t correspond to nets which will be routed over the upper row of cells, and the vertices of P_b represent nets which will be routed over the lower row of cells. It is easy to see that there are several restrictions on assignment of vertices to partite sets. For example, a vertex corresponding to a type I net N_j, which has both of its terminals on the upper cell row, may not be assigned to partite set P_b because nets represented in P_b are routed over the lower row of cells. On the other hand, a vertex corresponding to a type I net N_j with terminals on the lower cell row may only belong to P_b. As noted earlier, vertices representing type II nets may be assigned to either partite set since these nets can be routed over either the upper or lower cell row. A type III net N_j is partitioned into two type I nets at the location of its designated abutment. Each of these nets

```
Algorithm ASSIGN-ABUTMENTS()
begin
(* Sort nets of N_3 according to weight, forming netlist NL *)
      L = Sort(N_3)
(*ASSIGN[j] stores the abutment assigned to net N_j, and
A(N_j) denotes the set of abutments within the span of net N_j *)

(* Initialize array ASSIGN[] to zero *)
      for each a ∈ A do
            for each net N_j = (t_rx1, t_rx2) of N_3 do
                  if (x_1 < a < x_2) then A(N_j) = A(N_j) ∪ N_j;
      for i = 1 to | N_3 | do
            for each a ∈ A(L[i]) do
                  for each N_j such that ASSIGN[j]≠0 do
                        if (CND-OVLP(L[i], a, N_j, ASSIGN[j]) = 1)
                        then ASSIGN[i] = a;
                              break;
                        if (CND1-CONT(L[i], a, N_j, ASSIGN[j]) = 1)
                        then ASSIGN[i] = a;
                              break;
                        if (CND2-CONT(L[i], a, N_j, ASSIGN[j]) = 1)
                        then ASSIGN[i] = a;
                              break;
end.
```

Figure 10.19: Algorithm ASSIGN-ABUTMENTS.

is considered as a separate net in $\mathcal{N}$ and must be assigned to a fixed partite set as in the case of other type I net. The basic idea of the algorithm is similar to that of the algorithm MKIS in Chapter 3 and we call this algorithm FIS. The lower bound of the algorithm is 75% of the optimal solution. However, experimentally, the algorithm typically gives solutions which are at least 91% of the optimal result and in the average case, the performance of the algorithm is very close to the optimal solution (98% of the optimal solution).

Channel Segment Selection and Channel Routing: Channel segment selection is same as that discussed in [CL90]. When channel segment assignment is completed, a channel router is used to complete the connections within the channel. For this purpose, a greedy channel router is used, which typically achieves results at most one or two tracks beyond the channel density [RF82].

The formal description of algorithm WISER appears in Figure 10.20. On PRIMARY I benchmark from MCNC, WISER produces a solution with the total number of track equal to 206 as opposed to the solution with 187 tracks produced by the greedy channel router and 449 track solution produced by the

earlier OTC router.

10.1.3 Three-Layer Over-the-cell Routing

Holmes, Sherwani, and Sarrafzadeh [HSS91] introduced two models for three-layer, over-the-cell channel routing in the standard cell design style. For each model, an effective algorithm is proposed. Both of the algorithms achieve dramatic reduction in channel height. In fact, the remaining channel height is normally negligible. The novelty of this approach lies in use of 'vacant' terminals for over-the-cell routing. For the entire PRIMARY 1 example, the router reduces the routing height by 76% as compared to a greedy 2-layer channel router. This leads to an overall reduction in chip height of 7%.

Wu, Holmes, Sherwani, and Sarrafzadeh [WHSS92] presented a three-layer over-the-cell router for the standard cell design style based on a new cell model (CTM) which assumes that terminals are located in the center of the cells in layer M2. In this approach, nets are first partitioned into two sets. The nets in the first set are called critical nets and are routed in the channel using direct vertical segments on the M2 layer, thereby partitioning the channel into several regions. The remaining nets are assigned terminal positions within their corresponding regions and are routed in a planar fashion on M2. This terminal assignment not only minimizes channel density but also eliminates vertical constraints and completely defines the channel to be routed. In the next step, two planar subsets of nets with maximum total size are found and they are routed on M3 over-the-cell rows. The rest of the nets are routed in the channel using a HVH router.

Terai, Nakajima, Takahashi and Sato [TTNS94] presented a new model for over-the-cell routing with three layers. The model consists of two channels and routing area over a cell row between them. The channel has three layers, whereas the over-the-cell area has two layers available for routing. An over-the-cell routing algorithm has been presented that considers over-the-cell routing problem as a channel routing problem with additional constraints.

Bhingarde, Panyam and Sherwani [BPS93] introduced a new three-layer model for, over-the-cell channel routing in standard cell design style. In this model the terminals are arranged in the middle of the upper and the lower half of the cell row. They develop an over-the-cell router, called MTM router, for this new cell model. This router is very general in nature and it not only works for two- and three- layer layouts but can also permit/restrict vias over-the-cell.

Bhingarde, Khawaja, Panyam and Sherwani [BKPS94] presented a hybrid greedy router for the TBC model. The routing algorithm consists of two key steps; *terminal position assignment* and *2-3-2 layer irregular boundary channel routing.* An optimal *O(KL)* algorithm for terminal position selection is presented. The algorithm determines exact terminal locations on each target in the entire cell row. The routing environment for the TBC Router typically consists of a 3-layer channel area enclosed by two 2-layer non-uniform boundary over-the-cell routing regions. The TBC router generates smaller layouts for benchmarks, primarily due to smaller layout widths. For example,

Algorithm WISER()
begin
(* PHASE 1: Net Decomposition and Classification *)
for each $N_j \in \mathcal{N}$ **do**
$\mathcal{N}' = \mathcal{N}' \cup \{N_x = (t_{ri}t_{ri+1}) \mid r \in \{t, b\}, 1 \leq i \leq k$
where i refers to the i^{th} terminal of k-terminal net N_j $\}$
for each $N_j = (t_{r_1x_1}, t_{r_2x_2}) \in \mathcal{N}'$ **do**
if $(r_1 = r_2)$ (* Type I nets *)
then $\mathcal{N}_1 = \mathcal{N}_1 \cup \{N_j\}$
if ($\bar{r_1}$ is vacant) and ($\bar{r_2}$ is vacant) (* Type II nets *)
then $\mathcal{N}_2 = \mathcal{N}_2 \cup \{N_j\}$
(* Type III nets *)
if $(r_1 \neq r_2)$ and ($\bar{r_1}$ is not vacant) and ($\bar{r_2}$ is not vacant)
then $\mathcal{N}_3 = \mathcal{N}_3 \cup \{N_j\}$
(* PHASE 2: Vacant Terminal/Abutment Assignment *)
for each $N_j = (t_{r_1x_1}, t_{r_2x_2}) \in \mathcal{N}_2$ **do** (* Type II nets *)
$\mathcal{V}(N_j) = \{t_{\bar{r_1}x_1}, t_{\bar{r_2}x_2}\}$
for each $N_j = (t_{r_1x_1}, t_{r_2x_2}) \in \mathcal{N}_3$ **do** (* Type III nets *)
$\mathcal{A}(N_j) = \{a \mid a \in \mathcal{A}, x_1 < a < x_2\}$
ASSIGN= ASSIGN-ABUTMENTS($\mathcal{N}_3, \mathcal{A}$)
(* PHASE 3: Net Selection *)
$\mathcal{N}_3' = \mathcal{N}_3; \mathcal{N}_3 = \emptyset$
for each $N_j = (t_{r_1x_1}, t_{r_2x_2}) \in \mathcal{N}_3'$ **do**
$N_{j1} = (t_{r_1x_1}, t_{r_1A[j]})$
$N_{j2} = (t_{r_2A[j]}, t_{r_2x_2})$
$\mathcal{N}_3 = \mathcal{N}_3 \cup \{N_{j1}, N_{j2}\}$
B = FIS($\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3$)

(* PHASES 4, 5, and 6: Over-the-Cell Routing, Channel
Segment Assignment, and Channel Routing *)
OVER-THE-CELL-ROUTE(B)
C = ASSIGN-CHANNEL-SEGMENTS($B, \mathcal{N}$)
CHANNEL-ROUTE(C)

end.

Figure 10.20: Algorithm WISER.

for PRIMARY I benchmark, all three models TBC, CTM and MTM, generate channelless layouts, however, TBC layout has minimum area due to smaller layout width.

10.1.4 Multilayer OTC Routing

With the advent of multi-layer processes more OTC area is now available for routing and hence, further reduction in the layout height can be accomplished. Bhingarde, Madhwapathy, Panyam and Sherwani [BMPS94] presented an efficient four layer OTC router, for a cell model similar to TBC, called Arbitrary Terminal Model(ATM). In this cell model, the terminals can be placed at any arbitrary locations in the cell. Freed from fixed terminal placement restrictions, cell designers can aim to design with minimum width. Figure 10.21 shows ATM based designs. The routing algorithm is based on the following four key steps; (1) The nets spanning multiple rows are decomposed into net segments belonging to single rows. All the terminals belonging to a single row are connected by a single horizontal metal segment, and a terminal is selected on each segment for completing the net connectivity. (2) Generation of intervals for same row and critical nets. (3) Interval assignment and same row routing and (4) Selection of an appropriate position for placing the same row and critical net intervals in each cell row. This approach was further generalized so that it can be used not only for different cell models, but also for full custom layouts and thin film MCM's.

10.1.5 Performance Driven Over-the-cell Routing

Despite the dramatic performance of OTC routers, a major shortcoming of the existing routers is the increase in the total wire length and the length of the longest net. Careful analysis of existing results shows that the total wire length may be increased by as much as 20% in [CPL93] and 35% in [HSS93]. Although no results on wire length are reported, it is very likely that the net length also increases in case of [LPHL91]. However, it is possible that the net length in [LPHL91] is less than the corresponding net lengths reported in [CPL93, HSS93]. This may be due to the fact that the main objective of their router is to minimize the number of routing tracks used in the over-the-cell area, as well as in the channel.

Natarajan, Holmes, Sherwani, and Sarrafzadeh [NSHS92] presented a three-layer over-the-cell channel routing algorithm (WILMA3) for high performance circuits. This router not only minimizes the channel height by using over-the-cell areas but also attempts to route all nets within their timing requirements. This algorithm is based on two ideas. Firstly, it optimizes the track assignment of each net with respect to delay. It identifies the track bound for each net which ensures that the wire length is no greater than the length of the net if routed in the channel. Using this track bound, nets are selected for over-the-cell routing. Secondly, 45° segments are used to route the nets over-the-cells to further reduce the net length.

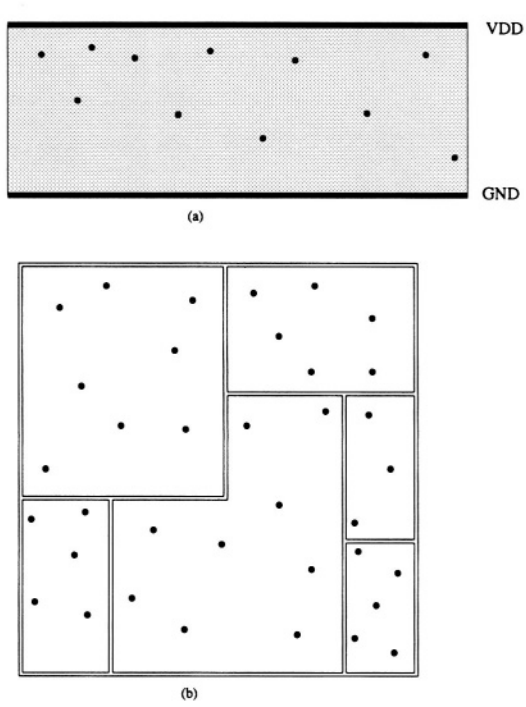

Figure 10.21: ATM based designs (a) Standard Cell (b) Full Custom

The basic idea of the algorithm is as follows: all the multi-terminal nets are decomposed into two-terminal nets and classified. Then weights are assigned to each net. The weight of a net intuitively indicates the improvement in channel congestion possible if this net can be routed over the cells. A channel router is then used to obtain the channel density (d_c) if routed in the channel. For each net N_i, track in which N_i is routed is recorded. An over-the-cell router is used to obtain the channel density (d_o) for over-the-cell routing. For each net N_i, the track bound k_i is computed, which ensures that if the net is routed over-the-cell at a track less than or equal to k_i, it will have a wire length less or equal to the net length when routed in the channel. This is based on the estimated channel heights d_c and d_o. Among all the nets which are suitable for routing over the cells, four (two) maximum-weighted planar subsets are selected, subject to the track bound constraint for the three-layer (two-layer) model. Once the nets are selected, a set of vacant terminals (vacant abutments) in the case of Type II (Type III) nets are assigned to each net N_i depending on its weight. These vacant terminal/abutment locations will later be used to determine an over-the-cell routing for N_i. Over-the-cell routing is done with 45° segments and rectilinear segments. In order to avoid design rule violations, any net N_i routed over-the-cell on track t_i must contain a vertical segment of

length ρ_i before 45° segments can be used. The net segments that have not been routed in the area over the cells are routed in the channel. After the channel routing is done the channel density (d_θ) due to over-the-cell routing of nets is obtained. If $d_\theta > d_o$, d_o is set equal to d_θ and the process is repeated.

The iterative process mentioned above takes place very rarely as for most examples the algorithm can complete the routing using no more tracks than d_o. However, in cases when d_θ is indeed greater than d_o, it has been observed that it is usually one or at most two tracks.

10.2 Via Minimization

Vias are necessary to establish multi-layer connections. Many routers use a simple reserved layer model and produce routing solution with a large number of vias. However, there are numerous reasons for minimizing the number of vias in a layout:

1. In integrated circuit fabrication, the *yield* is inversely related to the number of vias. A chip with more vias has a smaller probability of being fabricated correctly.

2. Every via has an associated resistance which affects the circuit performance.

3. The size of the via is usually larger than the width of the wires. As a result, more vias lead to more routing space.

4. Completion rate of routing is also inversely related to the number of vias.

Despite all these reasons, existing routers and design tools consider the minimization of the number of tracks in channel routing, completion of switchbox routing, and wire length minimization as their primary objectives. Via minimization is either completely ignored or de-emphasized. As a result, via minimization came as an 'afterthought' in routing.

Before discussing the via minimization problem in detail, let us define some related concepts. A *plane homotopy* (also called a *sketch)* consists of a set of simple curves in the routing region. The two endpoints of a curve are the terminals of a net. Two curves may intersect at a finite number of points, i.e., overlap of wires is not allowed. A k-layer homotopy (or simply a homotopy) is obtained by mapping pieces of the curves of the plane homotopy into one of the k layers. Vias are established at points where a curve changes layer and no two distinct curves intersecting on the same layer (see Figure 10.22 for two different homotopies of the same problem). If the topology of the plane homotopy is fixed, then the problem is called CVM. In other words, in CVM problem, we are given a set of wire segments (the placement of wire segments has already been determined by some router) and k layers for routing. The problem is to assign each segment to one of the layers without changing the topology so that the number of vias required is minimized. In UVM problem, the placement and

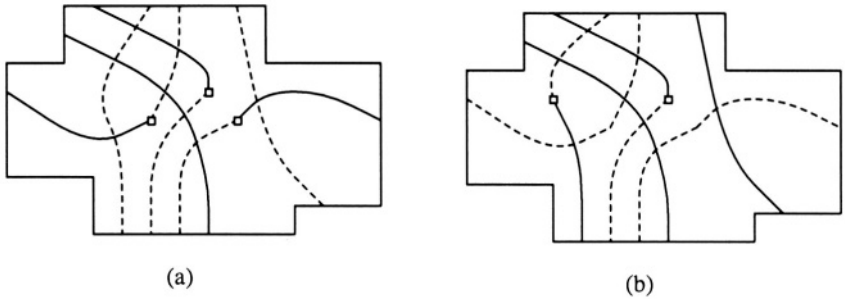

Figure 10.22: Two homotopies of the same problem.

the layer assignment of segments are not given. The problem is to both place the segments and also assign the layers so as to minimize the total number of vias. In other words, UVM is an integrated approach to routing and via minimization.

Intuitively, the UVM problem is harder than CVM problem. This is so because each UVM has many different homotopies, each resulting in a different optimal number of vias. Thus solving the UVM problem requires finding such a homotopy, which leads to a global minimum number of vias. In the following sections, we discuss both the CVM and the UVM problem.

10.2.1 Constrained Via Minimization Problem

In multi-layer routing problems, vias are required when two nets are crossing each other on a single layer. A *via candidate* is a maximal piece of wire, that does not cross or overlap with any other wire, and can accommodate at least one via. A *wire segment* is a piece of a wire connecting two via candidates. A *wire segment cluster* (or simply *cluster* is a maximal set of mutually crossing or overlapping net segments. For example, Figure 10.23 shows an instance of CVM problem. The points other than terminals, where two or more segments of a net meet and are electrically connected are called *junctions.* The number of segments which meet at a particular junction is referred to as *junction degree.* A *crossing* is a point where two net segments of two different nets intersect. A layer assignment is *valid* if no two segments of two different nets cross at a point in the same layer.

A routing solution is called a *partial routing solution* if the physical locations of the net segments is given, however, the layer assignments are not specified. Also, a valid layer assignment must exist for a partial routing solution. A *complete routing solution* consists of a set of net segments, a set of vias, and a valid layer assignment which correctly realizes the interconnection requirements specified by the netlist. A valid layer assignment for Figure 10.23 is shown in Figure 10.24.

Given the above definitions, the CVM problem can be formally stated as

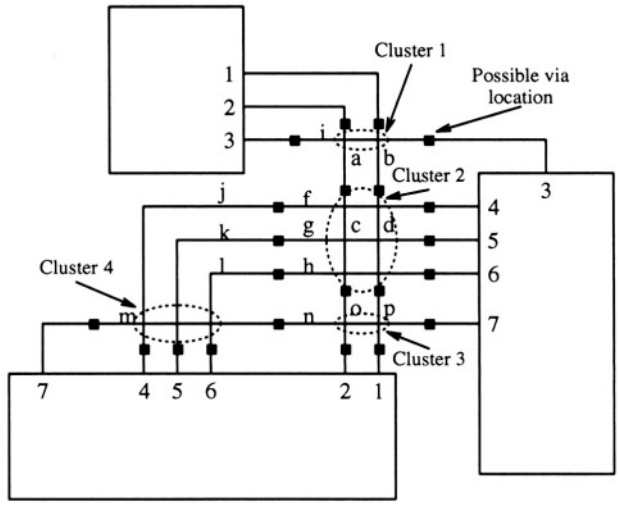

Figure 10.23: A CVM problem instance.

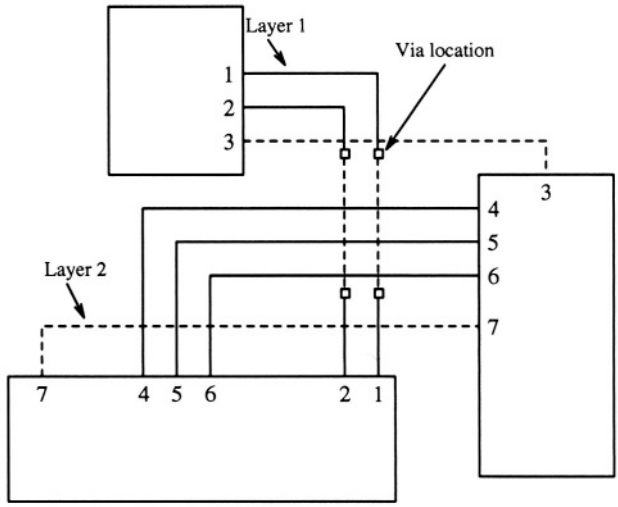

Figure 10.24: A valid layer assignment.

follows. Given a partial routing solution for a particular routing problem on k layers, find a complete routing solution with minimum number of vias for the corresponding partial routing solution. Since the CVM problem is to assign net segments to layers, the problem is also called the *layer assignment problem.* We use the terms layer assignment and complete routing solution interchangeably.

In 1971, Hashimoto and Stevens [HS71] first formulated the two-layer CVM problem as a graph-theoretic max-cut problem. The problem was initially thought to be NP-complete which led other researchers to develop heuristic algorithms [CK81, SV79]. In [SV79], Stevens and VanCleemput used a similar but more general graph model than Hashimoto and Stevens model to develop heuristic algorithm for the two-layer CVM problem. Ciesielski and Kinnen [CK81] proposed an integer programming method for the same problem. Chang and Du [CD87] developed a heuristic algorithm by splitting vertices in a graph. In 1980, Kajitani [Kaj80] showed that the two-layer CVM problem can be solved in polynomial time when the routing is restricted to a grid-based model, and all the nets are two-terminal nets. Kajitani identified the net segment clusters in a layout and showed that the graph in Hashimoto's model is planar. Kajitani's result encouraged other researchers to look for a polynomial time algorithm for more general case. In 1982, Pinter [Pin82] proposed an optimal algorithm for two-layer CVM problem when the maximum junction degree is limited to three.

10.2.1.1 Graph Representation of Two-Layer CVM Problem

In this section, we first describe the graph-theoretic representation of the two-layer CVM problem formulated by Pinter [Pin82]. We also describe the model presented by Naclerio, Masuda and Nakajima. Note that in each cluster, once a wire segment is assigned to a certain layer, layer assignment of the rest of the cluster is forced. Thus there are only two possible ways to assign the wire segments in a cluster to layers. With a prescribed layer assignment, a cluster is said to be *flipped over*, if all the wire segments in the cluster are reassigned to the opposite layers.

Given a (partial) routing problem, a *cluster graph* $G = (V, E)$ can be defined, where

$V = \{v_i \mid v_i$ corresponds to cluster $i\}$ and
$E = \{(v_i, v_j) \mid$ clusters i and j are connected to at least one via candidate$\}$

The cluster graph for the layout in Figure 10.23 is shown in Figure 10.25.

If a complete routing solution is given, the weights can be assigned to the edges of the cluster graph. The weight $w(e)$ associated with each edge $e \in E$ of the cluster graph is defined as follows. Let p be the number of via candidates connecting the two clusters incident to e, and let q be the number of vias introduced by the known layer assignment connecting the two clusters. Then $w(e) = 2q - p$. In other words, the weight indicates the via reduction that can be achieved due to flipping over either one of the two clusters. The weights corresponding to the solution in Figure 10.24 are shown in Figure 10.25.

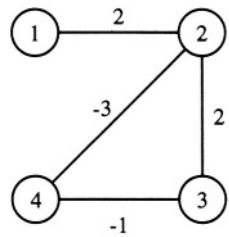

Figure 10.25: Cluster graph.

An arbitrary layer assignment L can be obtained from a known layer assignment L_0 by flipping over a set of clusters. Let X be the set of clusters which are flipped over in L_0 to obtain L. If X consists of just one cluster v, then the change in the number of vias is equal to the weights of all the edges incident on v. In a general case, the net change of vias is equal to the weights of edges between the sets X and $V - X$. This is due to the fact that any two clusters $u, v \in V$ (or $u, v \in V - X$), the via count between the clusters u and v remains unchanged. However, for $u \in X$ and $v \in V - x$, if $e = (u, v) \in E$ then the via count is reduced by $w(e)$. Let $q(L)$ and $q(L_0)$ be the numbers of vias introduced in the layer assignments L and L_0, respectively. Then

$$q(L) = q(L_0) - \sum_{e \in E(X, V-X)} w(e)$$

where $E(X, V - X)$ is a cut separating X and $V - X$, i. e., the set of edges connecting vertices in X and vertices not in X. The above equation is due to the fact that for any two clusters both in X or both in $V - X$, the via count between the two clusters remains unchanged, but for two clusters, one in X and one in $V - X$ via count is reduced by $w(e)$. In order to minimize the via count q(L), we want to find a cut $E(X, V - X)$ which maximizes its weight $\sum_{e \in E(X, V-X)} w(e)$, This problem is equivalent to the max-cut problem. Note that the edge weights $w(e)$ can be positive or negative, but a maximum cut always has non-negative weight since X can be empty and $\sum w(e) = 0$ for $X = \phi$. In case that a maximum weighted cut has weight 0, L_0 is an optimal layer assignment with minimum number of vias. For the cluster graph shown in Figure 10.25, the vertex sets {2, 4} and {1, 3} determine the maximum cut of total weight 3. As a result, three vias can be reduced to produce a minimum via routing by flipping over clusters 2 and 4. The minimum via routing is shown is Figure 10.26.

Note that the cluster graph is planar if the junction degree is at most three. In planar graphs the max-cut problem is polynomial time solvable [Had75]. Therefore, the via minimization problem can be solved in polynomial time if the junction degree is restricted to at most three.

In 1989, Naclerio, Masuda, and Nakajima [NMN89] showed that without

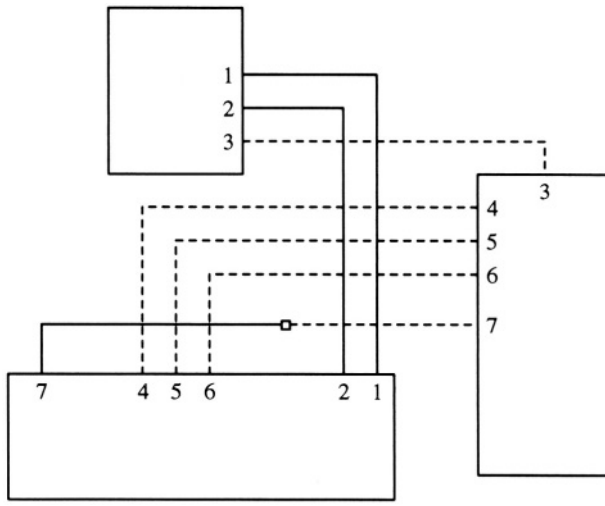

Figure 10.26: Minimum via routing.

any restriction on the maximum junction degree, the CVM problem is NP-complete, by showing a polynomial time transformation from the NP-complete planar vertex cover problem [GJ79]. They also show that the problem is NP-complete, even when one or more of the following restrictions are made.

1. The layout must be grid-based.
2. Vias can be placed only in the junctions.
3. The maximum junction degree is limited to six or more.

In 1987, Naclerio, Masuda, and Nakajima [NMN87] presented a different graph representation of the CVM problem for gridless layouts. In this representation, also the maximum junction degree is restricted to at most three. Given a partial routing solution, a *crossing graph* $G = (V, E)$ is defined as follows: Each vertex $v \in V$ corresponds to a crossing of two wire segments of two different nets in the partial routing. Two vertices $v_i, v_j \in V$ are adjacent only if there is an wire segment connecting the crossings corresponding to v_i and v_j in the partial routing. Figure 10.27(b) shows the derived crossing graph G corresponding to the partial routing of Figure 10.27(a). It is easy to see that the crossing graph defined above is planar. Each face, of the planar crossing graph is a fundamental cycle. If that cycle has an odd length, then we call that face an odd face. Otherwise the face is called an even face. Since each edge corresponds to a wire segment in the partial routing and each vertex to a crossing, the wire segments corresponding to edges that make up a fundamental cycle in the graph must be assigned to alternating layers to obtain a valid layer assignment. For an even face, all the wire segments corresponding to the

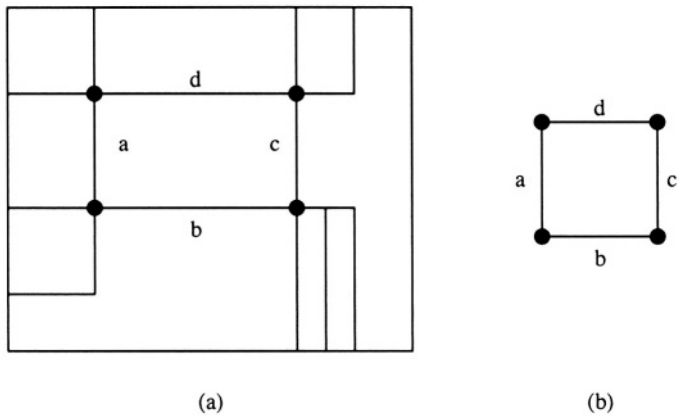

Figure 10.27: Partial routing and corresponding crossing graph.

edges can be assigned to alternating layers, thus no vias are required for for that face. Consider the example shown in Figure 10.27. The graph consists of just one even cycle, and the segments *a, b, c,* and *d* can be alternately assigned to layers 1 and 2 to get a solution with no vias. On the other hand, if the graph contains an odd cycle, then the wire segments corresponding to the edges of that cycle cannot be assigned to alternating layers to obtain a valid routing without vias (see Figure 10.28(a) and (b) for an example of odd face). Note that each odd cycle require at least one via to obtain a valid routing.

Thus, a partial routing solution can be routed with no vias if and only if the corresponding crossing graph does not contain any odd faces. That is if the crossing graph is bipartite. In case the graph contain odd faces, the wire segments requiring vias can be marked and the corresponding edges in the graph can be removed and two faces sharing that edge can be merged. If the remaining graph is odd cycle free, then no further vias would be required to route the wire segments in the remaining graph. Thus in order to find the minimum number of wire segments that require vias, it is necessary to find the minimum number of edges such that the removal of those edges results in a bipartite subgraph.

Note that the problem is also equivalent to find a maximum cut the planar crossing graph. Hadlock's algorithm [Had75] can be used to find the maximum bipartite subgraph from a planar graph. The algorithm presented by Hadlock removes the minimum number of edges from the graph to remove all the odd cycle by forming the dual of the planar graph.

The crossing graph can be extended to handle multiterminal nets as long as the junction degrees are restricted to at most three. In that case, each junction is also represented as a vertex in the crossing graph. The details of the description may be found in [NMN87].

All the optimal algorithms mentioned above are based on Hadlock's maxi-

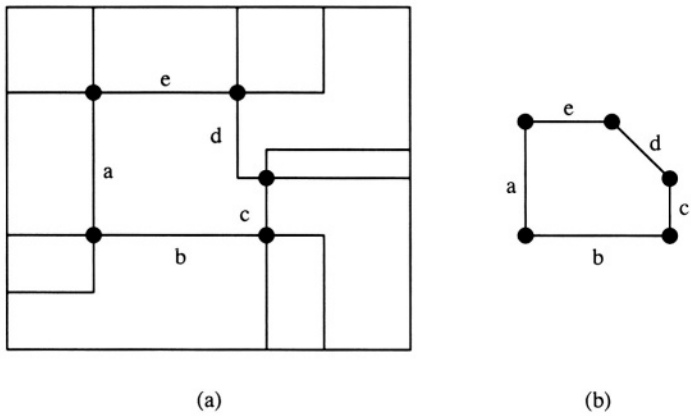

Figure 10.28: Example of an odd face in the crossing graph.

mum cut algorithm for planar graphs [Had75]. Since Hadlock's algorithm requires finding all-pair shortest paths and finding a maximum weighted matching of a dense graph, all the algorithms have time complexity $O(n^3)$, where n is the total number of net segments.

In 1988, Kuo, Chern, and Shih [KCS88] presented an $O(n^{3/2}\log n)$ time complexity optimal algorithm for the CVM problem. The algorithm they proposed was based on Pinter's graph model. In 1990, Barahona [Bar90] also presented a simpler $O(n^{3/2}\log n)$ time complexity optimal algorithm for the two-layer CVM problem.

10.2.2 Unconstrained Via Minimization

As mentioned before, the unconstrained via minimization (UVM) problem (also known as topological via minimization (TVM) problem) is concerned with finding a plane homotopy of wires so that the total number of vias are minimized [CL91, Hsu83b, LSL90, Sad84, RKN89, SL89a, SHL90]. The physical dimensions of the wires, terminals, and vias are not considered in the UVM problem. The general TVM problem in k layers (k-TVM) may be stated as follows. Given a set of nets, number of layers k and terminal locations, find a k-layer topological routing solution that completes the interconnections of all nets using the minimum number of vias. In weighted version of k-TVM problem (k-WTVM), each net is assigned a positive weight which is a measure of the priority of the net. The weight of a via represents the weight of the corresponding net. The problem is to minimize the total weight of vias used in the routing.

The TVM problem was first introduced by Hsu in [Hsu83b], and it was conjectured that TVM problem is NP-hard. Hsu considered a simple 2-TVM problem for two-terminal nets and formulated the problem using circle graphs.

It was shown that the 2-TVM problem is equivalent to finding a maximum bipartite subgraph in the corresponding circle graph. The independent sets of the bipartite subgraph can be routed in two layers without any vias. The remaining nets can be routed using vias. This result established the fact that TVM problem can be solved by routing maximum number of nets without any vias and the rest of the nets using as few vias as possible.

Marek-Sadowska [Sad84] proved that the TVM problem is NP-complete. Following theorem was also proved by Marek-Sadowska for two-terminal net TVM problems:

Theorem 27 *There exists a solution to an arbitrary instance of topological via minimization problem such that each net uses at most one via.*

The above theorem shows that the TVM problem can be solved by maximizing the number of nets that can be routed without any vias (i.e., in planar fashion).

In 1989, Sarrafzadeh and Lee [SL89a] showed that the problem of finding a maximum bipartite subgraph in a circle graph is NP-complete which in turn proves that even a simple 2-TVM problem is NP-complete. As a result, several special classes of the TVM problem have been considered. Sarrafzadeh and Lee [SL89a] and Cong and Liu [CL91] considered the crossing-channel TVM problem. In the crossing-channel TVM problem, the routing region is a simple channel. All the nets are two-terminal nets and no net has both of its terminals on the same boundary. Crossing-channel k-TVM and k-WTVM problems are solvable in polynomial time [CL91, LSL90, RKN89, SL89a].

10.2.2.1 Optimal Algorithm for Crossing-Channel TVM Problem

Note that a crossing channel is equivalent to a matching diagram and its permutation graph can easily be found (see Chapter 3). As a result, the problem of finding maximum independent sets in permutation graphs become a key problem. Sarrafzadeh and Lee [SL89a] showed that the problem of finding a maximum 2-independent set in a permutation graph can be solved in polynomial time. Cong and Liu [CL91] showed that the problem of finding a maximum k-independent set in a permutation graph can be solved in polynomial time.

Given a crossing channel consisting of a set of nets $\mathcal{N} = \{N_1, N_2, \ldots, N_n\}$, the TVM problem can be solved by first finding a maximum k-planar subset of nets. The k-planar subset of nets can be routed in k layers without any vias. Then using Theorem 27, the remaining nets can be routed in any two adjacent layers using one via per net.

We now show how the nets can be routed using one via per net by an example of the 2-TVM problem. Let $\mathcal{N}^*$ be a maximum 2-planar subset of nets for the given problem. Without loss of generality, assume $\mathcal{N}^* = S_1^* \cup S_2^*$, where $S_1^* = \{N_1, N_2, \ldots, N_p\}$, $S_2^* = \{N_{p+1}, N_{q+2}, \ldots, N_{p+q}\}$. Note that any net $N_t \in \mathcal{N} - \mathcal{N}^*$ must cross nets in S_1^* and in S_2^*. Since S_1^* is planar, nets in S_1^* can be assigned to layer 1. The p nets in layer 1 partition the region into $p+1$ subregions called *panels* $X_0, X_1, \ldots, X_p$ from left to right, where $N_j \in S_1^*$ separates regions X_{j-1} and X_j. Similarly, nets S_2^* can be assigned to layer 2

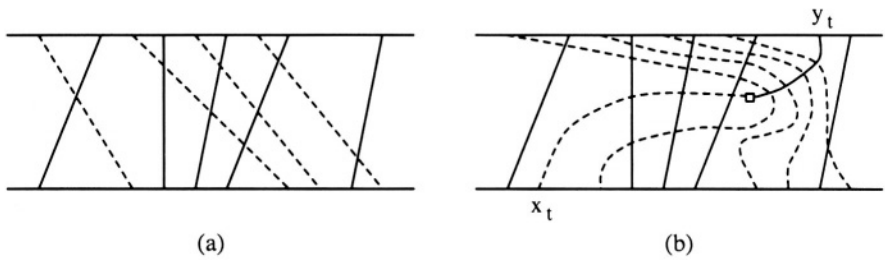

Figure 10.29: A topological routing of a net using one via.

to form q +1 panels, denoted $Y_0, Y_1, \ldots, Y_q$ from left to right. Figure 10.29(a) shows planar routing of two sets S_1^* and S_2^* on layer 1 and layer 2, respectively.

Assume that a net $N_t \notin \mathcal{N}^*$ and $N_t = (x_t, y_t)$ is in panel (X_j, Y_k). Without loss of generality, let us assume that x_t lies in panel X_j and y_t lies in panel Y_k. Consider placing a via v_t in panel Y_k and connect x_t to v_t and v_t to y_t. Let the segment connecting x_t to v_t be denoted by $[x_t, v_t]$ and let v_t be in panel X_l for some l. Without loss of generality, assume that $l \geq j$ and $[x_t, v_t]$ be assigned to layer 1. The nets $N_{j+1}, \ldots, N_l$ on layer 1 that 'intersect' $[x_t, v_t]$ are 'pushed' to right, thereby, enlarging the panel X_j (see Figure 10.29(b)). The segment $[v_t, y_t]$ can be assigned to layer 2 without any difficulty, since it lies totally within panel Y_k. If there is more than one net to be routed, the above mentioned steps can be repeated to route all the nets using one via per net. The nets can be routed from left to right in $O(n)$ time. Since the maximum k-independent set in a permutation graph can be found in $O(kn^2)$ time. Therefore, the total complexity is dominated by the problem of finding maximum k-planar subset of nets. Thus we conclude,

Theorem 28 *An optimal solution to a crossing-channel TVM problem can be found in* $O(kn^2)$ *time.*

10.2.2.2 Approximation Result for General k-TVM Problem

If the routing region is more general than a channel, then the two-terminal net k-TVM problem becomes NP-hard. This is due to the fact that the circle graph must be used to represent the problem instead of simpler permutation graph. The k-TVM problem is equivalent to finding a maximum k-independent set in a circle graph. In chapter 3, we have presented an $(1-(1-\frac{1}{k})^k)$-approximation algorithm for maximum k-independent set in circle graphs. Using that result, the following theorem can easily be proven:

Theorem 29 *Given a set of nets* $\mathcal{N} = \{N_1, N_2, \ldots, N_n\}$ *in a k-layer routing region, let* $\mathcal{N}^*$ *be the maximum k-planar subset of nets in* $\mathcal{N}$*, and* $\mathcal{N}'$ *be the k-planar subset of nets found by taking one maximum planar subset at a time, then* $|\mathcal{N}'| \geq (1 - (1 - \frac{1}{k})^k) \times |\mathcal{N}^*|$.

Based on Theorem 27 and Theorem 29, we conclude,

Theorem 30 *Given a set of nets* $\mathcal{N} = \{N_1, N_2, \ldots, N_n\}$ *in a k-layer bounded region, the k-TVM problem can be approximated with with at most* $(1 - \frac{1}{k})^k \times |\mathcal{N}^*|$ *more vias than the minimum number of vias, where where* $\mathcal{N}^*$ *the maximum k-planar subset of nets in* $\mathcal{N}$.

10.2.2.3 Routing Based on Topological Solution

Since it appears that the CVM problem does not offer enough flexibility for via minimization, the topological routing might offer a good starting point as vias are already minimized. It is easy to see that minimum-via topological routing often uses very long wires for some nets and causes high congestion in the routing region. Since the geometric routing problem has fixed area, it may not be possible to transform a high congestion topological routing solution to geometric routing solution. Therefore, a topological routing solution is needed that is guaranteed to be transformable into an actual geometric routing solution. This can be achieved by allowing some extra via's to keep the topology as close to the actual geometric solution as possible. In this way, the final topological routing solution can be easily transformed into actual geometric routing solution. We denote this problem as *routable topological via minimization* problem in k layers (k-RTVM). The major difference between the solutions of TVM and RTVM problems is that the solution of RTVM problem is guaranteed to be transformable into actual geometric routing.

In [HS91], Hossain and Sherwani presented a graph-theoretic algorithm to solve 2-layer routing problem based on topological solution. The algorithm consists of two different phases. The first phase of the algorithm finds a solution to 2-RTVM problem. In the second phase, the solution to 2-RTVM problem is transformed into actual geometric routing.

The algorithm starts with finding a 2-planar subset of nets. Each planar subset is routed in a separate layer to form panels. If the panels on two layers are projected on a single layer, the panels intersect and form pseudo-rectangular regions. The remaining nets are topologically routed by assigning nets to the regions keeping the topology as close to the actual routing as possible. The topological routing of the nets is done by finding a weighted shortest path in the corresponding *region adjacency graph* defined from the regions. In the region adjacency graph, each vertex corresponds to a region and two vertices are adjacent if their corresponding regions share a boundary. Once the nets are topologically routed, a geometric routing is obtained by iteratively imposing grid onto each region.

10.3 Summary

The layout area for standard cell design can be reduced by minimizing the channel height. Over-the-cell routing has been successfully used to achieve dramatic reductions in the channel heights. In three-layer technology, it is

possible to achieve even a channel-less layout. Several algorithms for over-the-cell routing have been presented. For high performance circuits, an algorithm has been presented which minimizes the layout height without sacrificing the performance. A significant research is needed to develop new cell models and associated over-the-cell routers to achieve the channel-less layouts for high density circuits.

Via minimization is one of the most important objectives in the detailed routing. There are two different approaches to minimize the number of vias. In constraint via minimization problem, the topology of the routing solution is fixed. Vias can be minimized only by reassigning the net segments to different layers. On the other hand, in unconstraint via minimization problem, the objective is to find a routing topology with minimum number of vias. Since the topology in UVM problem is not fixed, the UVM problem allows much flexibility than that of the CVM problem. The UVM approach, however, does not take into consideration the routing constraints; as a result, UVM solutions are not practical. Since the UVM approach allows a significant reduction on the number of vias and as the technology is improving and more and more layers are becoming available, it is expected that topology based routing solution will be more competitive.

10.4 Exercises

‡1. Given, (a) a single layer rectangular routing region R which has K tracks and two rows of terminals; one on top side and another on the bottom side and (b) a set of two-terminal nets $\mathcal{N}$. Give an efficient algorithm to find a maximum subset of $\mathcal{N}$ which can be routed in R.

2. More utilization of the over-the-cell area is possible if we allow an additional net type (type IV). Net $N_j = (t_{r_1x_1}, t_{r_2x_2})$ is a *type IV* net if $r_1 \neq r_2$, neither $t_{\bar{r}_1x_1}$ nor $t_{\bar{r}_2x_2}$ is vacant, and there exists two vacant terminals $t_{r_1x_3}$ and $t_{r_1x_3}$ with $x_1 < x_3 < x_2$ and $x_1 < x_4 < x_2$ (see Figure 10.30(a)). Note that Type IV nets are not constrainted to use abutments, however, they compete with the type II and type III nets for the usage of vacant terminals. Modify WISER to use type IV nets in addition to type I, II and III nets.

3. Further utilization of the over-the-cell area is possible if an additional net type (type V) is allowed. Net $N_j = (t_{r_1x_1}, t_{r_2x_2}), x_1 < x_2$ is a *type V* net if $r_1 \neq r_2$, neither $t_{\bar{r}_1x_1}$ nor $t_{\bar{r}_2x_2}$ is vacant, and there exists two vacant terminals $t_{r_1x_3}$ and $t_{r_1x_3}$ with $x_3 < x_1$ and $x_4 < x_1$ or $x_3 > x_2$ and $x_4 > x_2$. Note that from Figure 10.30(b) type V can be used for taking nets away from the congested areas, however, it increases the net length. Modify WISER to use type V nets in addition to type I, II, III, IV nets.

‡4. Given, (a) a single layer rectangular routing region R which has a height of K tracks, a terminal row on its bottom boundary, and a set of rectangular

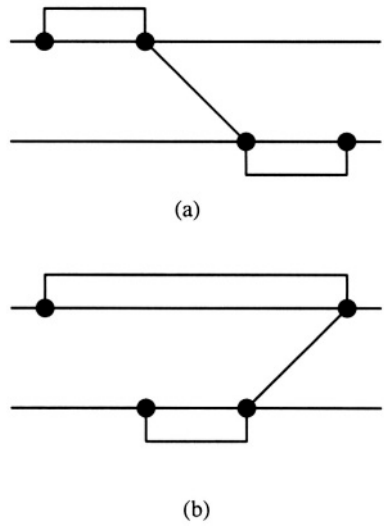

Figure 10.30: (a) A type IV net. (b) A type V net

blockages and (b) two-terminal nets $\mathcal{N}$. Give an efficient algorithm to find a maximum subset of $\mathcal{N}$ which can be routed in R.

†5. In three-layer technology, when vias are allowed in over-the-cell region, then the over-the-cell channel routing is similar to 2-layer channel routing in over-the-cell area and 3-layer channel routing in channel area. For this case, develop a greedy router that can simultaneously perform channel routing as well as over-the-cell routing.

‡6. In many cell libraries the entire metal layer (M2) is not available for routing. Instead, it has several blockages representing the routing within the cell. Also, the terminals may not be aligned in a row. In this case, the nets that are to be routed in the channel need be brought to the boundaries of the cell using the available routing regions in M2. Develop an algorithm for this problem.

†7. In [DMPS94] planar over-the-cell routing algorithm for two terminal nets was presented. Extend the algorithm to multi-terminal nets.

†8. Prove that 2 layer planar over-the-cell routing problem is NP hard.

9. Prove that the time complexity of algorithm presented in [DMPS94] is $O(kn^2)$, where k is the number of tracks available in over the cell region and n is the number of nets.

10. Given the partial routing in Figure 10.31, do the following:

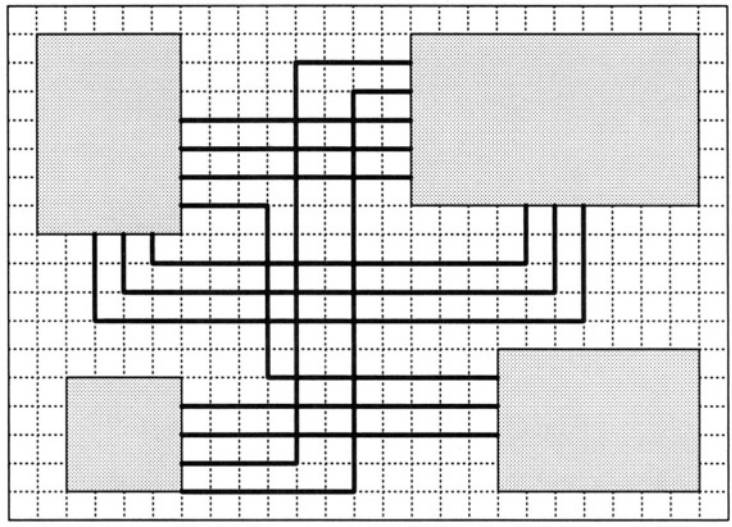

Figure 10.31: A routing instance.

a. Find all of the via candidates. Note that if a segment spans more than one gridline, a via can be placed in that segment.

b. Find any valid layer assignment.

c. From the layer assignment created in c, develop the cluster graph.

d. Find the max-cut of the cluster graph derived in c.

e. Reassign the layers to find the minimum via routing.

11. The performance of a chip can be improved by minimizing the number of vias per net. Develop an algorithm which routes nets with one via per net.

‡12. Develop a coloring based algorithm to 3 layer constrained via minimization.

13. Develop an algorithm that minimizes the vias in a routing, by making local changes in the routing with the use of maze patterns.

14. Develop a router for two-layer crossing channel routing problem to route all the nets with at most one via per net. The basic idea of the algorithm is the same as topological routing solution for crossing channels, however, instead of finding topological solution the router should find actual detailed routing. The algorithm should first find a maximum 2-planar subset of nets and route them on two different layers. Then route the remaining nets using as many columns and tracks required.

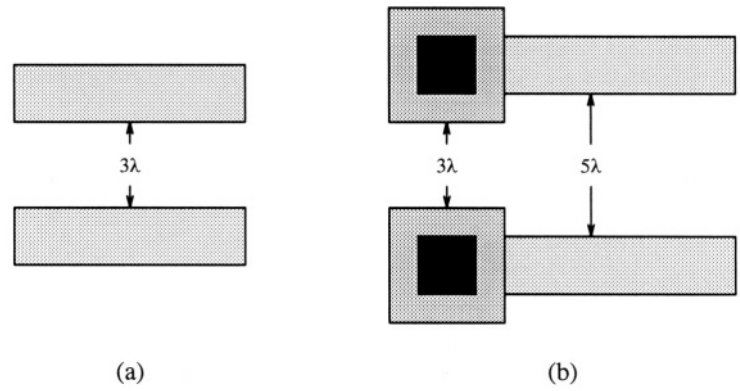

Figure 10.32: Spacing between tracks.

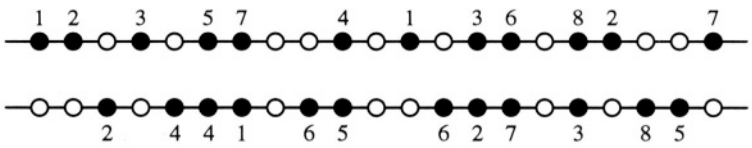

Figure 10.33: An instance of MSOP.

†15. Develop a two-layer routing algorithm for crossing channel routing problems based on topological solution. The algorithm should first find a maximum 2-planar subset of nets and route them on two layers to form a pseudo grid. The remaining nets are to be routed on the pseudo grid using a modified maze routing technique.

16. According to the design rules, the spacing between two adjacent metal tracks needs be 3λ. If the vias on adjacent tracks are aligned in a column, the spacing between the tracks increases to 5λ (see Figure 10.32). Develop an algorithm that offsets the aligned vias to compact a channel.

17. Solve the instance of MSOP in Figure 10.33 for $K = 3$.

Bibliographic Notes

The concept of OTC routing was first introduced by Deutsch and Glick in 1980 [DG80]. In [CL88], a symbolic model for over-the-cell channel routing was presented together with the algorithms for each stage of the entire routing process. Lin, Perng, Hwang, and Lin presented a linear programming formulation to select a set of nets to route in the channel in order to reduce the channel density [LPHL91]. In [NESY89], a new design style employing OTC routing techniques called *Quickly Customized Logic* (QCL) was introduced for

fast turn around times. An over-the-cell gate array channel router was presented in [Kro83]. Katsadas and Kinnen [KK90] presented a multilayer router using over-the-cell areas. In citeDLMPST96, algorithms for selecting maximum planar subset of nets which are suitable for planar OTC routing were presented. In [Kan96], presents a new triple-layer OTC Channel router for OTC routing in an irregular cell area.

A detailed description of OTC routing concepts and techniques can be found in a specialized book on this topic by Sherwani, Bhingarde and Panyam [SBP95].

In 1983, Chen, Kajitani, and Chan [CKC83] a polynomial time optimum algorithm for grid-based layouts when the junction degree is also limited to three. However, they restrict that vias can only be placed at junctions. In 1987, Naclerio, Masuda, and Nakajima [NMN87] presented an algorithm which has the same complexity as Chen *et al.* [CKC83]; however does not require that the layout should be grid-based or restrict the via locations.

In [SL89a], an $O(n^2)$ time complexity algorithm is presented for the crossing-channel 2-WTVM problem, where n is the total number of nets. An optimal $\Theta(n \log n)$ time complexity algorithm for crossing-channel 2-TVM problem is presented in [SL89a]. crossing-channel k-WTVM has been solved in $O(kn^3)$ time [CL91]. This algorithm was improved to $O(kn^2)$ in [SL90]. Multi-layer TVM problem was considered in [SHL90] and it was shown that if the terminals are pre-assigned to layers, then the problem can be solved in $O(kn^2)$ time, where k is the maximum number of terminals of a net in a single layer and n is the total number of terminals.

Chang and Cong [CC97] presented an efficient heuristic algorithm for the layer assignment and via minimization problems for multilayer gridless layouts.