Routing Algorithms for Channels, Switchboxes and MCM's in VLSI Layout Design

Dissertation submitted for the degree of
Doctor of Philosophy
of the
Indian Statistical Institute

Sandip Das

Advanced Computing and Microelectronics Unit
Indian Statistical Institute
203, B. T. Road, Calcutta - 700 035
INDIA

(Revised)

DEDICATION

To my parents

Contents

Acknowledgment

I intend to give vent to my profound feeling of gratitude to Professor Bhargab B. Bhattacharya who evoked in me a deep insight into the way to concentrate to this novel course. His perennial and constant guidance to find the wayout from the entangled labyrinth deserves no less expression.

I shall be lacking in my gratitude in case of any lapse from properly mentioning the contributions of Professor Bhabani P. Sinha for the suggestions and encouragement to overcome as and when frustration and problems overtook me.

No less worth mentioning are the help and assistance I had received from Professor Mihir K. Chakrabarti, Professor Jayasree Dattagupta, Dr. Nabanita Das, Dr. Susmita Sur-Kolay, Dr. Subhas C. Nandy and Professor Malay K. Sen at the various stages of progress of the work.

I do not find proper words to express my thanks for the invaluable assistance I had received from my friends Dr. Krishendu Mukhopadhyaya, Dr. Srabani Sen Gupta, Dr. Debesh Das, Dr. Susanta Chakrobarti, Dr. Rajib Das, Dr. Mallika De and Arijit Bishnu.

I shall prove ungrateful if I fail to mention about Mrs. Sumita Sinha and Mrs. Sarmistha Bhattacharya who had been encouraging me during my long course of research.

Last but not the least, I thank my parents, my brothers, my sister, my wife and my daughter for being themselves and making me what I am today.

January 20, 2000
Indian Statistical Institute
Calcutta

Sandip Das

Chapter 1
Introduction

1.1 Background

The term *Very Large Scale Integration (VLSI)* reflects the capability of semiconductor industry to fabricate a complex electronic circuit consisting of millions of components on a single silicon substrate. The growth of semiconductor technology in recent years has been described by "Moore's law", enunciated in the late 1960's, which projected quadrupling of components in a chip in every three to four years. Several factors contributed to this tremendous growth : (i) reduction of line width of the basic device and interconnection wires due to the development of high-resolution lithographic techniques and improved processing capabilities, (ii) increase in the size of the silicon wafer due to improved reliability of processing, (iii) growth of the accumulated circuit, and (iv) layout design experience. Better understanding of system level design issues leads to improved architectures exploiting the current technology, and helps in designing efficient electronic design automation (EDA) tools for logic synthesis, circuit layout, simulation, verification, and testing.

Microelectronics technology is advancing very rapidly. The design styles and tools are also undergoing a continuous evolution. The custom design style incorporates maximum flexibility; the designers have at their disposal the entire array of design tools, i.e., logic capture, layout, design verification, simulation, timing verification, placement and routing.

VLSI technology plays a major role in the development of complex electronic systems. As a result of continuing progress in semiconductor technology, around 100,000 to several millions of transistors can now be put on a single micro-chip. New design methods for VLSI are therefore of constant need to cope with this increased design complexity.

The VLSI design process spans a diverse spectrum of disciplines in physics, chemical engineering, electrical engineering, and computer science. Because of the diversity of tasks and design issues, a systematic approach to breaking the process into a number of design layers and subtasks is essential. Typically, the top-down design of a chip consists of the following stages:

(i) design specification
(ii) functional design
(iii) logic design
(iv) circuit design
(v) physical design
(vi) fabrication on semiconductor chips.

Each stage consists of synthesis, analysis and verification, and iterations may be necessary in order to eliminate errors. *Layout design* is essentially a stage of physical design where behavioral and structural representations of an electronic system are translated to geometric shapes to be used during fabrication. Several problems arise during the layout design of VLSI chips. This includes *partitioning* the components into homogeneous groups, *placement* of modules on the floor and finally, *routing*, i.e., interconnection (wiring) among the modules.

The work reported in this thesis is mainly concerned with the development of new and efficient routing algorithms. Once a circuit is partitioned and elements are placed on board, it is necessary to implement the connection patterns amongst the modules using signal nets.

The objective of routing strongly depends on the nature of the chip. For general purpose chips, it may be sufficient to minimize the total *wire length* in order to reduce the chip area. For high performance chips, total wire length may not be a major concern. Instead, one may want to minimize *the length of the longest wire* to minimize the delay in signal propagation and, to *reduce crosstalk* to enhance its performance. The presence of large number of *vias* (contact holes across layers) is undesirable from fabrication and circuit performance point of view. So, *via minimization* also plays an important role in designing routing algorithms. Similar other parameters need to be considered depending on the design goals.

Unfortunately, efficient algorithms satisfying all these conflicting goals are in general hard to find. Several researchers identified different sub-problems for example, row routing, channel routing, switchbox routing, global routing, etc., and attempted to solve these problems either in isolation or in totality. Even in many simple situations, an optimal solution is hard to find. In fact, most of these layout optimization problems have been proved to be NP-hard [S93]. Algorithms for solving such problems may give optimum results in polynomial time in some special situations, but in most of the cases, they remain intractable. In this thesis, our goal is to identify new types of routing problems, investigate them critically, and to develop efficient algorithms based on graph theory, combinatorics and related mathematical tools, and finally, to implement them for industrial use.

1.2 Scope of the Thesis

Channel routing is one of the most important and probably one of the most popular phases of physical design for VLSI chips as well as PC-boards. One of the earliest algorithms in this area was due to Hashimoto and Stevens [HS71]. Since then, various methods were extensively studied and applied to many different technologies. It may be noted that with the advent of availability of multiple routing layers, the classical channel routing problem seems to have lost its full significance. However, the basic algorithms are useful for routing in a system-on-a-chip environment, as well as in an MCM. In this thesis, we focus on the following problems, and propose new techniques for solving them:

1. Over the cell channel routing;
2. Via as well as wire length minimization in channel routing problem;
3. Channel routing in manhattan diagonal model;
4. L-shaped channel routing;
5. Switchbox routing;
6. Routing in multi-chip modules (MCM).

Descriptions of these problems are summarized in the following sections. Novel techniques and data structures have been utilized for their modeling. The proposed methods, based on graph algorithms, and computational geometry incorporate many new features, and experiments on benchmarks demonstrate their superiority to the existing ones from the viewpoint of running time, space requirement, and generality.

1.2.1 Channel routing

A channel is a routing region bounded by two parallel rows of terminals. Without any loss of generality, it is assumed that the two rows are horizontal. Each terminal is assigned a number which represents the net to which the terminal belongs to, and it appears on the top and bottom boundaries of the channel. Zeros represent vacant terminals which do not belong to any net. The netlist of a channel is the input to most of the channel routing algorithms.

The horizontal (vertical) dimension of the routed channel is called the channel length (channel width). In channel routing problem, the goal is to find the interconnections of all the nets in the channel in the specified area. Generally, two or more layers are available for routing. Change of layers for nets is done through holes called vias. The main objective of the channel routing is to minimize the channel width. Additional objective functions are minimization of the number of vias used in a multilayer channel routing, minimization of wire length, etc. In grid-based *reserved layer* models, the horizontal wire segments appear on tracks which are vertically equidistant inside the channel. Here, the objective essentially is the assignment of horizontal segments of nets to the tracks. Vertical segments are used to connect horizontal segments of the same net in different tracks with the pins appearing along the boundaries of the channel. In the *unreserved layer* model for two-layer routing, vertical and horizontal wires may pass through both the layers. In this situation, though the width of the channel reduces drastically, the routing becomes complicated.

The first theoretical bound of the track requirement is given by Kawamoto and Kajitani [KK79], where they suggested a column-by-column routing algorithm that guarantees routing with an upper bound on the number of tracks equal to the density plus one, where the density of a channel routing instance is the highest congestion over all columns. The congestion of a column is the number of nets passing through that column. In this method, additional columns may occasionally be needed to complete the routing. The first algorithm developed for channel routing was the left-edge algorithm, proposed by Hashimoto and Stevens [HS71] which uses a reserved layer model and does not allow doglegs and any vertical constraints. Ho, Iyenger and Zhenq [HIZ91] developed a simple but efficient heuristic channel routing algorithm which is greedy in nature and can be generalized to switchboxes and multi-layer problems.

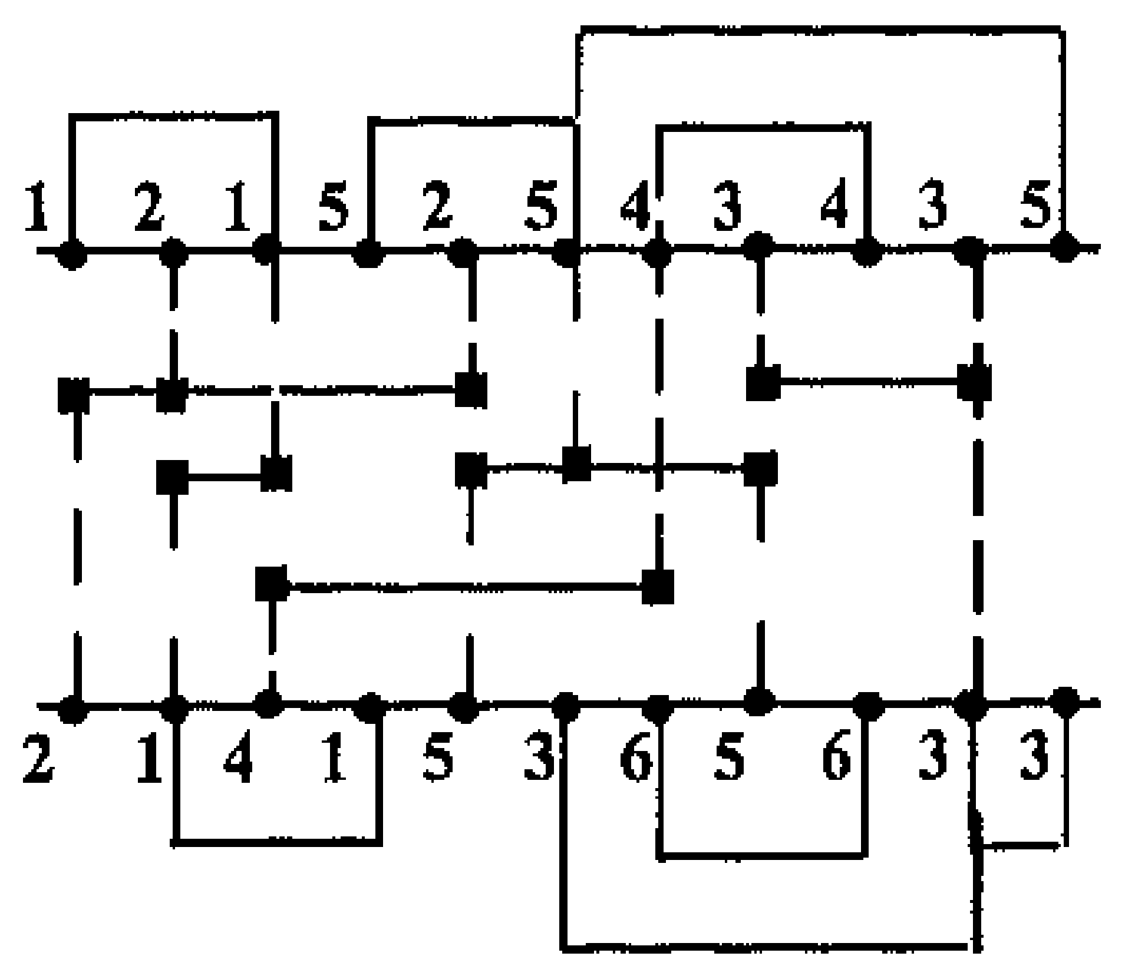
1.2.1.1 Over-the-cell routing

The total 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. Conventionally, in two-layer layout, cells appear in a single metal layer. Routing is done through both the metal layers in the channel area. As several channel routers have been developed that complete routing with the number of tracks very close to the channel density, no further improvement in the layout area is possible, if routing is done only through channels.

With the advent of two-and-half layer technology used in the layout design of standard cells, several channel routers use the extra routing area available outside the channel for interconnections. It is already mentioned that no cell is placed in the channel area in either region, and cells are placed in a single layer in the cell area. So, in a two layer model, the cell area in one layer remains unused. Reduction in layout area is only possible if the unused portion of the cell area is utilized during routing. Such routing methodology is referred to as *over-the-cell* routing in the literature. An example of over-the-cell routing is shown in Fig. 1.1.

In this context, the routing problem is to realize the interconnection of nets by means of paths, such that two paths of distinct nets are not allowed to intersect in the same layer. Terminals are accessible only at the edges of the cells. Given a netlist, the objective is to generate a routing with possible over-the-cell interconnections which minimizes the number of horizontal tracks inside the channel i.e., the channel width.

Over-the-cell channel routing problem is known to be NP-hard [S93]. The best known heuristic algorithms for over-the-cell channel routing appeared in [CL88, CPL90], which consist of three major phases in the following sequence : (i) selecting the net segments for routing over-the-cell, (ii) choosing terminals of the nets for routing within the channel and (iii) completion of routing through the channel.



# of tracks inside the channel = 3
# of tracks over the cell = 4
# of vias = 12

Fig. 1.1 : An example of over-the-cell routing

It is easy to observe that the first two steps are highly interrelated. Routing a net over-the-cell may force some other nets to be routed within the cell through the heavily congested area. Thus, maximizing the number of over-the-cell candidates as suggested in [CL88] may not lead to a good solution. In our work [DNB91], we have described a new heuristic algorithm which iteratively selects potential over-the-cell nets with an ultimate objective of minimizing channel width. Experimental evidence shows considerable improvement over earlier results [CL88] in the number of tracks used both inside the channel, and over the cell.

1.2.1.2 Via minimization

In channel routing, each signal net is formed by the wire segments interconnecting a set of electrically connected terminals. If multiple conducting layers are available for interconnection, the wires of the same net lying in different layers are connected through vias (contact holes). In reserved layer model using two layers, all horizontal wire segments are placed on one layer, and all vertical wire segments are assigned to the other layer. However, this method tends to generate a large number of vias in the final routing solution. The presence of too many via holes may cause certain disadvantages, e.g.,

(i) besides increasing the manufacturing cost and complexity, vias in a circuit degrade its performance and reliability, and
(ii) vias require more layout area. In a compactible channel routing, the nMOS design rule requires 3λ spacing for path width, 3λ for feature separation, whereas, (4λ x 4λ) square area for a via hole in metal layers.

Thus, minimization of via holes leads to considerable improvement in VLSI layout design in terms of chip area, circuit performance and cost [H83]. This is even true if a short run of metal is replaced by polysilicon because the capacitance and resistance of the contact are likely to be greater than those of added polysilicon [CD88].

The existing methods for via minimization are classified into two general categories:

(i) unconstrained via minimization (UVM);
(ii) constrained via minimization (CVM).

Although in most cases, UVM produces a solution with fewer vias, the layout area

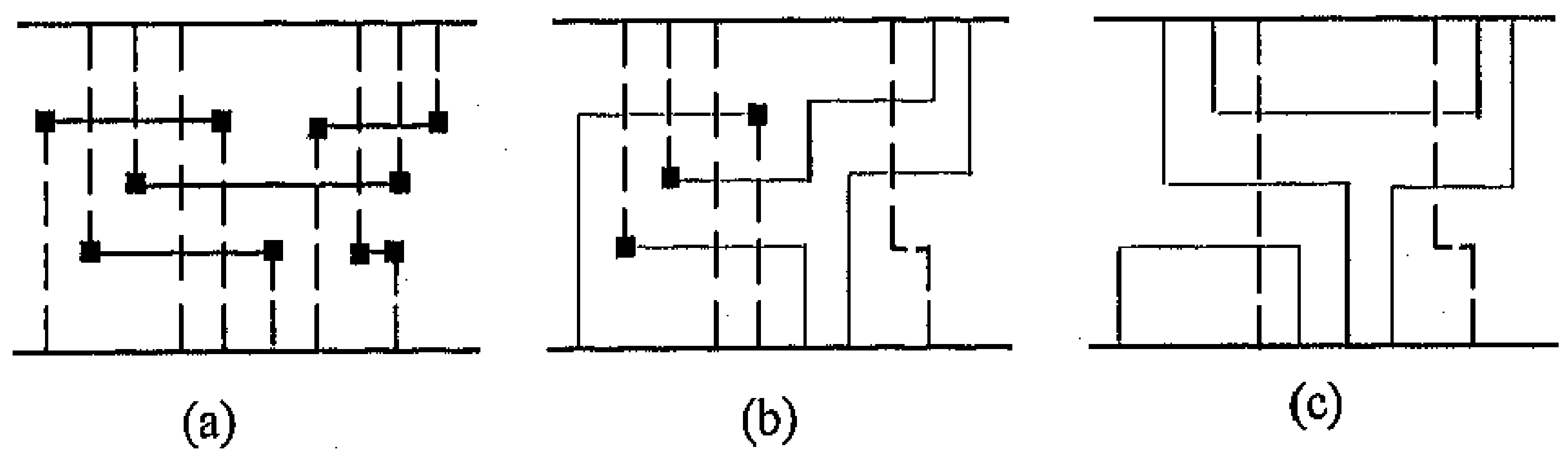


Fig. 1.2 : Routing solution of a channel
(a) before via minimization (# via = 10),
(b) after via minimization using [TWC91] (# via = 3), and
(c) another solution with no via

is not compact. An example of a channel, routed without via-minimization, is shown in Fig. 1.2(a). The layout obtained by via minimization algorithm of [TWC91] is shown in Fig. 1.2(b). Further via minimization may be possible with some clever heuristic as demonstrated in Fig. 1.2(c).

An approach to the via minimization technique was first proposed in [L61] based on layout modification followed by maze routing. In the unreserved layer model, via minimization problem is solved in [TWC91]. In this thesis, we have reported a new approach for via minimization [DB93] using the advantages of both CVM and layout modification. Our algorithm outperforms the result (in terms of the number of vias) of [TWC91] for all standard benchmarks without increasing the routing area. For Deutsch's difficult example, we obtained only 190 vias with 19 tracks, which is, to the best of our knowledge, the fewest ever reported. The total wire length also decreases in almost all the cases compared to the solutions of [TWC91].

1.2.2 Channel routing in Manhattan-diagonal model

The conventional channel routing algorithms, studied for the last two decades assume horizontal and vertical wire segments for interconnection. Recently, efforts have been made to develop algorithms using non-manhattan geometries. A *diagonal model* with $\pm 45°$ wires was reported in [LLP89a]. Routing in *time square model* using $\pm 60°$ wires was considered in [LLP90]. A channel router using both rectilinear and diagonal ($\pm 45°$) wire segments was proposed in [CR91, CHS94]. The diagonal model is well supported by the fabrication process and provides better utilization of routing space. It also tends to reduce wire length and hence area and delay. However, the existing algorithms based on this model are either too elementary or inefficient, and therefore, have limited applicability to real life routing problems.

In this thesis, we propose a new approach to channel routing that combines the advantage of both diagonal model and manhattan geometry. We consider two routing layers; the terminals reside along the channel boundary are fixed, and the routing wires are assumed to be of manhattan-diagonal ($\pm 45°$) model (MD-model). First, we consider the reserved layer model (i.e., horizontal and diagonal wire segments are in one layer and vertical wire segments are in other layer), and present a simple $O(l\ .d)$ time algorithm that routes an arbitrary channel with no cyclic vertical constraints in w tracks, where l is the length of the channel, d is the channel density, and $d \leq w \leq (d+1)$. Next, we describe an output-sensitive algorithm using unreserved layer manhattan-diagonal model, that can route two-layer channels with cyclic vertical constraints using ω tracks, in $O(l.w)$ time (i.e., linear in area of the channel). An illustration of our model is given in Fig. 1.3.

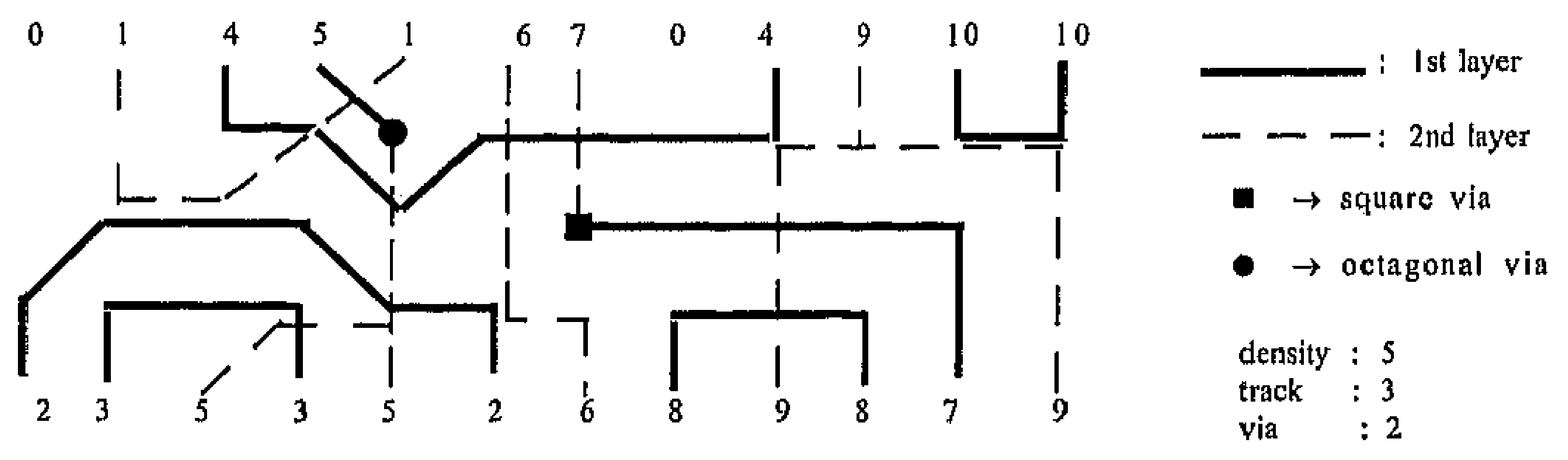


Fig. 1.3: Channel routing in unreserved layer MD model

Our proposed algorithm outperforms the methods in [YK82, HIZ91, CR91] both in execution time and in quality of solutions. Experimental results on benchmark examples show very encouraging results. It is also observed that the number of vias used and the wire length is significantly less compared to the via minimizer of [TWC91]. To the best of our knowledge, this work is the first reporting of routing in the MD-model.

1.2.3 L-shaped channels, staircases and switchboxes

Routing of *L-shaped channels* also arises in many instances. In some routing model, the global routing problem can be viewed as a sequence of L-shaped channel routing problem [C87, CG96]. Here, a net-list along the boundary of an L-shaped channel is given, and the goal is to complete the interconnections minimizing (i) channel area, (ii) the number of vias, and (iii) wire length.

In this thesis, we propose routing algorithms for L-shaped channels, switchboxes and staircases in 2-layer Manhattan-Diagonal model with tracks in horizontal, vertical and $\pm 45^{\circ}$ directions. The core of our proposed method lies on routing a right triangular region which requires $O(n^2)$ time, where n is the number of nets which are routed through the triangle. We show that if the L-shaped channel has no cyclic vertical constraint, our algorithm runs in $O(ld)$ time, where l is the length of the L-shaped channel, and d depends on the density of horizontal and vertical parts of the L-shaped channel, and the density of the triangular regions joining those channels. The exact expression for d is given in Chapter 6. Next, a greedy heuristic method is proposed for routing an L-shaped channel with cyclic vertical constraint, which guarantees a routing solution with at most d+1 tracks. The time complexity of the latter method is also $O(ld)$. An example of an L-shaped channel and its routing is shown in Fig. 1.4.

The *switchbox routing problem* in MD-model is also solved elegantly. These techniques are easily extendible to the routing of staircase channels, and it yields

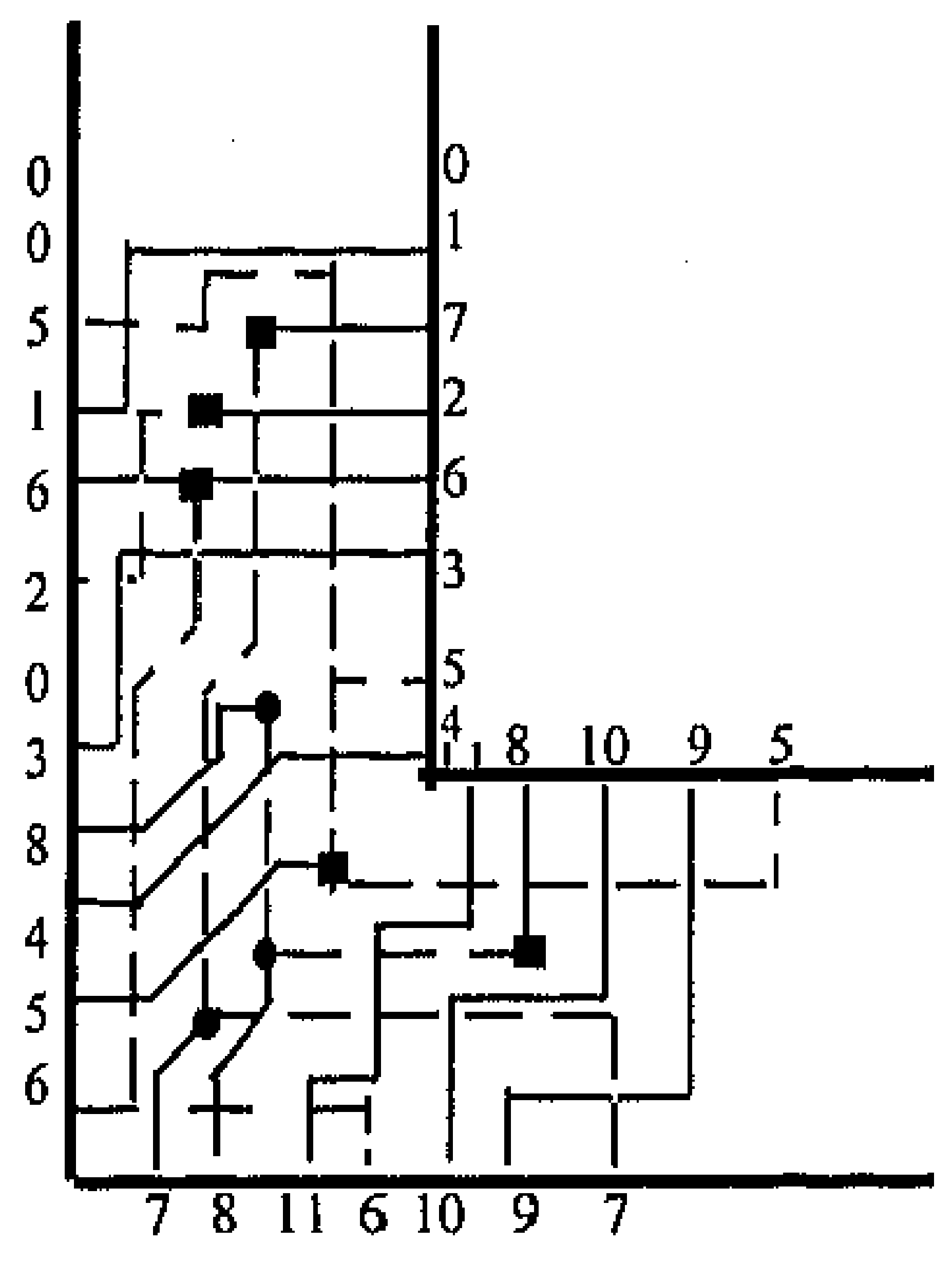


Fig. 1.4 : Examples of L-shaped channel routing using MD-model

efficient solution to detailed routing in general floorplans [DSB98]. Experimental results show significantly low via count and reduced wire length, thus establishing the superiority of MD-routing over classical strategies used in [CH88]. An example of a switchbox and its routing is shown in Fig. 1.5.

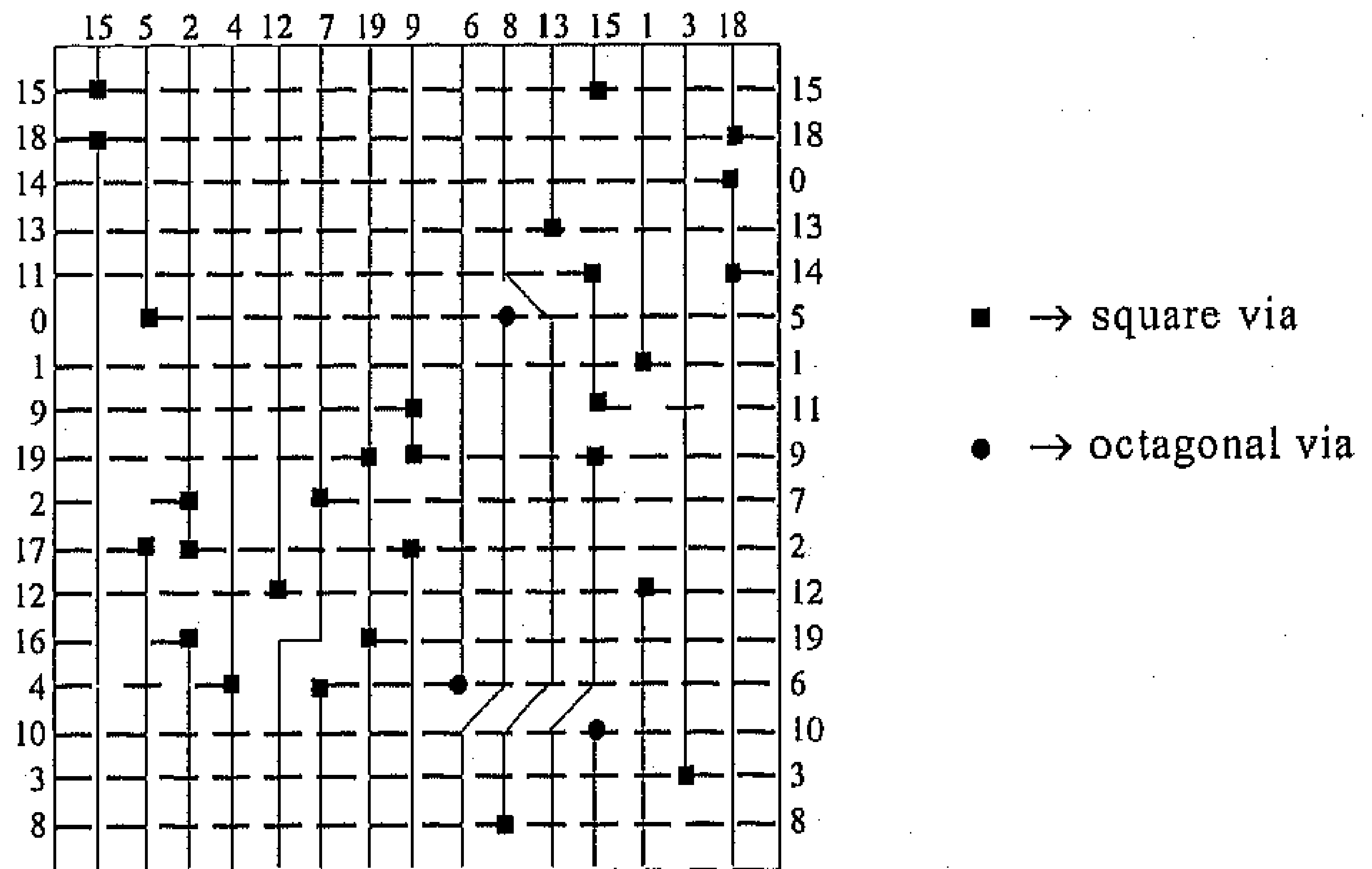


Fig. 1.5 : Switchbox routing using the proposed MD-model

1.2.4 Routing in multi-chip modules (MCM)

With rapid decrease of feature size, efficiently packing multiple circuit modules in a single chip has become a promising area of research. Using MCM technology a large number of functional blocks can be mounted and interconnected on a single substrate. The numbers of distinct nets and terminals are usually very high. A typical MCM may consist of 7000 nets on a grid size of over 2000 x 2000 [KC92]. With the emergence of deep sub-micron technology, the numbers of chip, nets, and terminals per net, in an MCM are likely to increase in the future.

Inside an MCM, the circuit components are mounted on the top layer; other layers are used for pin redistribution and routing. As many layers are usually available for MCM routing, the main objective is to improve performance, i.e., to achieve tight packaging with lower fabrication cost, satisfying the constraints on net length, net separation, via size, via separation, etc. Thus, the primary goal is to minimize the number of vias, wire length, cross talks, delay in signal propagation, with the secondary objective of reducing the number of routing layers in the MCM.

Several approaches to MCM routing have been proposed recently [CLRS94, SBP95]. A pair of routing layers is considered at a time, and the yet unrouted terminals are projected on them through stack vias. Khoo and Cong developed SLICE [KC92] and V4R [KC95]; in the latter method, at most four vias are used for routing each two-terminal net in a non-monotone fashion. Multi-terminal nets are transformed into two-terminal nets. Another algorithm, called SEGRA, was proposed in [CRN97]. It uses a plane sweep technique and splits each multi-terminal net into two-terminal nets considering a minimum spanning tree among its terminals.

In [DNB99], we introduce an improved method of MCM routing. For high performance routing, one has to minimize the number of vias and wire length. To minimize the number of vias, we first route the nets having more terminals, through the layers that are closer to the top, such that a large number of nets is routed with short pieces of wires. To take care of the delay factor, we also route the farthest terminals of a net through the layers closer to the top, and explore subsequent layers if they are unavailable. The proposed algorithm consists of three phases. First, we preprocess the nets to determine a routing order. We estimate the cost of a feasible wiring for each net by constructing a minimum-bend rectilinear steiner tree among its terminals, avoiding the obstacles on the concerned layer. The wire length and the number of terminals of the net are used to define a metric called *average path length.* The ordering of nets for actual routing in the next phase depends on the average path length. In order to reduce delay for some important nets, one may need to change the order. In the next phase, actual routing is done assuming a reserved layer model. Nets are routed following the order as obtained in the first phase. After the completion of routing in the current layer-pair, the unrouted terminals are projected in the next layer-pair and the same routing technique is adopted keeping the order of nets invariant. This process continues until all the terminals are routed. Our technique uses significantly less number of vias, which can be further reduced in the third phase, where we improve the routing by flipping and other techniques.

1.3 Conclusion

In this thesis, we have reported various new results on channel routing in Manhattan as well as in Manhattan-diagonal model. Novel algorithms for switchbox and L-shaped channel routing problems have also been formulated and studied in this thesis.

We have explored many new combinational and geometric properties of routing problems and found interesting results based on which routing algorithms for channels, switchboxes and L-shaped channels have been proposed. Experimental results on benchmark examples demonstrate that the proposed algorithms outperform the existing ones both in execution time and in quality of solutions. A routing technique for multi-chip-modules has also been developed based on computational geometry. This algorithm achieves significant reduction in via count and wire length for MCM benchmarks.

Chapter 2

Review

2.1 Introduction

Two important factors responsible for minimizing layout area of a VLSI circuit are:

(i) circuit design which attempts to reduce the number of transistors keeping the functional behavior unaltered, and

(ii) physical design of the circuit targeted to improve the geometric layout.

To reduce the complexity, the physical design phase of a VLSI circuit is split into five major steps, e.g., partitioning, floorplanning, placement, routing and compaction. A brief description of these steps is given below.

1. **Partitioning :** A chip may contain several millions of transistors. Layout design of such a circuit cannot be handled if the entire circuit is considered as a whole. This problem is nicely tackled by partitioning the circuit into a set of functional blocks (modules). Partitioning of large circuits is a hierarchical process. In each level, the blocks of its previous level are further partitioned into smaller sub-blocks. Partitioning of a block considers a set of important factors such as, the size and number of the sub-blocks, number of interconnections between the sub-blocks, etc. The set of interconnections among the sub-blocks are referred to as a *netlist*. The occurrence of the same signal net in different blocks is referred to as *terminals*. In the routing phase, the objective is to devise an efficient way of connecting the terminals of the same net using a set of available layers, satisfying a set of predefined constraints.

2. **Floorplanning :** After the partitioning phase, the size of each block is determined by the number and the type of components in that block.

Usually the blocks are assumed to be rectangular. In the floorplanning phase, the topology of the layout (the relative positions of the functional blocks on the chip) is determined based on the interconnection information among the circuit modules, so as to minimize the total area of the chip, and to reduce routing congestion.

3. **Placement** : The arrangement of components inside a block may be customized considering the interconnection pattern among its neighboring blocks. So, a set of possible aspect ratios is specified with each functional block before the placement phase. The floorplanning phase suggests the geometric adjacency among the functional blocks. The goal of the placement phase is to position the functional blocks on a floor of minimum area such that, the 100% interconnection between the blocks is feasible. Initially, a feasible placement of the blocks is determined considering an estimate of required routing space of different routing regions. It may actually results an unroutable layout. So, iterative improvements may be necessary during the placement process.

4. **Routing** : The objective routing phase is to complete the interconnection among blocks according to the specified netlist. The goal of this phase is to complete the connections among the pins of each net satisfying different constraints and optimizing different performance criteria.

5. **Compaction** : After the routing phase, an attempt is taken to reduce the layout area by compressing the layout from all directions parallel to the four boundaries of the rectangular floor defining the circuit. During the compaction process, the size of each circuit modules should remain unaltered. A substantial compaction reduces the wire length, which in turn reduces the signal delay between the components of the circuit. The compaction process must ensure that no design and fabrication rule are violated during the process.

The thesis is mainly concerned with the design and development of various routing techniques. In the rest of this chapter, we review the past works reported in this area.

2.2 Fabrication Factors

Continuing progress in the scale of integration has resulted in incorporation of a large number of functions in a single chip that increases the chip area, and corresponding deterioration of performance. An immediate remedy is the minimization of interconnection area which reduces the die size. A router should also consider the following factors to achieve high performance goals.

Delay : Reduction of line width of the interconnection wires due to the development of high-resolution lithographic techniques generates high resistance through interconnection wires, and as a consequence it results in higher delay in a longer path in comparison to that in a shorter one. In a synchronous circuit, limitations on circuit speed are determined, in general, by the delay of the longest path through the functional blocks, and the presence of clock skew among the synchronizing circuit components. Thus to enhance the performance of the circuit, it is necessary to minimize the maximum wire length, and an efficient clock routing that achieves minimum clock skew.

Yield : The *yield* is the percentage of IC's fabricated on an wafer, which are free from faults and operate at or above the desired frequency and reliability. The yield can be enhanced by fabricating a larger number of dies on a single wafer by reducing die size. This also minimizes the waste area on the wafer surface. From the physical design point of view, the potential area for yield enhancement includes minimization of total wire length, routing with less number of jogs, net separation to reduce cross talks, via minimization, to name a few.

Cost : Minimization of die size and increase in the yield, reduce the cost of a chip. Thus enormous effort is needed to produce dies of minimum size with simultaneously improved performance, yield and cost and which are free from manufacturing defects.

A VLSI chip may contain several millions of transistors, and there may be several hundreds of possible routing paths for each net. This makes the routing problem computationally hard. To reduce the complexity of the routing phase, it is traditionally divided into two major phases, namely *global routing* and *detailed routing*.

2.3 Global Routing

Global routing, or in other words the loose routing, is the preliminary step of the complete routing process. Here the entire routing region is split into a set of sub regions (e.g., channels of different shapes, switchboxes, etc.), depending on the layout style. A sequence of routing regions is assigned to each net without specifying the exact geometric layout of wires. The goal is not only 100% routing of all the nets, but also to minimize the total wire length / routing area. A detailed survey of various global routing techniques is given below.

2.3.1 Different design styles

The objective functions for global routing depend on design styles. In this subsection, we discuss the standard design styles.

Full-custom : This is the most general design style where the functional blocks (of arbitrary sizes) may be placed at any location on the chip floor in a non-overlapping fashion. This allows very compact design; however, the process of automating a full-custom design style is more complex than other restricted models. The problem of routing is also very difficult since the routing regions are not symmetric. However, the routing space may be allowed to be redefined by slightly perturbing the placement of blocks. Here, the role of compaction is significant for area minimization of the entire circuit after routing is completed.

Standard cell : The standard cell design style is somewhat simpler than a full-custom design style. A circuit is partitioned into a set of predefined sub-circuits, called *cell*s, which are available in the standard cell library. These cells are assumed to be rectangular and of same height. They are placed in rows, and the space between two adjacent rows is called a channel. The channels are used for routing the signal nets that appear among the cells lying on the same row or in adjacent rows. If a net appears in two cells appearing along the boundary of different channels, the interconnecting wire passes through the feedthroughs. A *feedthrough* is an empty space (cell) between a pair of topologically adjacent cells in a row, and it may be created wherever it is necessary during the routing process.

Gate Arrays : In gate array design, the entire wafer is prefabricated with an array of identical gates or cells. These cells are separated by both vertical and horizontal spaces called channels. Each functional block is mapped onto a prefabricated cell on the wafer, during the partitioning/ placement phase. Now, the task of the router is to establish the connections between the prefabricated cells through the horizontal and vertical channels. Note that, here the number of tracks allowed for routing in each channel is fixed. As a result, the purpose of routing is simply to complete the connections rather than minimize the area.

FPGA : It is a new approach to ASIC design that can significantly reduce manufacturing turn around time and cost. In general, an FPGA consists of a regular array of programmable logic blocks interconnected by a programmable routing network. A programmable logic block can be programmed by the user to act as a small logic module. Given a circuit, user can program the programmable logic module using an FPGA programming tool. One of the advantage of FPGAs is reprogrammability. In FPGA, a data channel is provided, which allows easy transfer of the new logic function and reprogramming the FPGA. Partitioning, placement and routing are three main steps for the physical design automation of FPGA. The partitioning problem mainly depends on the architecture in which the circuit has to be implemented. Placement problem in FPGAs is very similar to the gate array placement problem. The routing in FPGAs is to find a connection path and program the appropriate interconnection points.

2.3.2 Existing methods for global routing

The common problem of global routing for all the three design styles is to find the shortest path for connecting the terminals of a net placed in different blocks. The first work in this direction is the *maze routing* algorithm due to Lee [L61] and Moore [M59]. The algorithm is simple and it guarantees an optimal solution, if at least one such path exists. But as this algorithm is grid based, it requires a large amount of storage and its performance degrades rapidly as the number of grid points increases. Thus, this algorithm is very inefficient for a large full custom design problem. Numerous attempts have been made to modify Lee's algorithm to reduce its memory requirements [A67]. Soukup [S78] modified Lee's algorithm by proceeding towards the target through the grid points in a depth-first manner until an obstacle is encountered; a breath-first search method is then used to get around the obstacle. The depth-first search then resumes its motion towards target. The problem specification may however have some special requirements which do not permit fully automated routing. An interactive *maze router* [AS88] is available for this purpose, where restricted human interactions are allowed.

A heuristic algorithm using A* search technique was proposed by Hadlock

[H77] to improve upon the speed. A widely applicable line-probe algorithm was developed by Makami and Tubachi [MT68], and essentially the same algorithm was independently proposed by Hightower [H69]. The basic idea of a line probe algorithm is to reduce the size of memory requirement by using *isothetic line segments* during the search process, instead of grids. The time and space complexities of these algorithms are $O(L)$, where L is the number of line segments produced by the algorithms. In [ZLI96], an efficient algorithm for obstacle avoiding rectilinear shortest path among multi-terminal nets is proposed. This algorithm combines *implicit strong connection graphs* among the terminals of nets, and applies A^* search method to get an excellent routing solution. In all these methods, congested channels are assumed to be expanded as required. If the channel congestions are predefined, one may adopt the "rip-up and reroute" or "shove aside" techniques, as proposed in [B79, DK82].

All the algorithms we have considered so far, are related to routing of two terminal nets. These approaches can be extended to multi-terminal nets, but they involve large time and space complexities. Routing of multi-terminal nets is based on constructing a *rectilinear steiner tree.* The problem of finding a minimum-cost rectilinear steiner tree is NP-hard [GJ77]. The rectilinear steiner tree problem can be solved in polynomial time if the points appear on a set of axis-parallel lines, or on the boundary of the rectangular floor [AGH77]. Hwang [H76] has shown that the ratio of the cost of a minimum spanning tree to that of an optimal rectilinear stainer tree is no greater than 3/2. Good heuristic algorithm for finding the rectilinear steiner trees are available in [H78, HVW85]. Burman, Chen and Sherwani [BCS91] studied the global routing problem in a generalized geometry, called δ-geometry, where the edges may be oriented in any of $i\pi/\delta$ angles, where i can be any integer and δ is also an integer ≥ 2. In [CSW89], a restricted case of Steiner tree, called Steiner min-max tree is used for the global routing problem. They propose an efficient algorithm for obtaining a Steiner min-max tree in a weighted coarse grid graph, where the vertices are labeled *actual* and *steiner* depending on whether the vertex corresponds to the terminal of a net or a steiner vertex. The weight of an edge depends on the density of the corresponding channel through which the pair of vertices may be connected, and the capacity of the same channel.

The general global routing problem can be mapped to a 0/1 integer programming problem. Heisterman and Lengaur [HL91] presented a hierarchical algorithm for the global routing of multi-terminal nets using integer linear programming.

In [V91], an interior point method is proposed for global routing. It uses the famous Karmakar's polynomial time algorithm [AKR+89] in the linear programming formulation of routing problem.

Global routing algorithm can be mapped to different other optimization problems based on its varying optimization functions. A multi-commodity flow-based algorithm is proposed in [SK87, CLC96]. Vecchi and Krikpatrick used simulated annealing technique [VK83] to solve the same problem.

A practical global router for row based layout, such as sea-of-gates, gate array and standard cell was developed by Lee and Sechen [LS88]. For gate array, Aoshima and Kuh [AK83] proposed a global routing algorithm based on multi-channel optimization. Karp et al. [KLR+87] discussed the problem of global routing in a two-dimensional gate array. It uses linear programming followed by randomized rounding-off technique [RT91].

Simultaneous partitioning and routing-driven correlated approaches for placement of mixed macro blocks and standard cells are available in [DK85, LD86, C87, KPS89] which recursively apply mincut algorithm of [FM82]. In [YW96], a novel edge cost model is proposed to capture the congestion of routing region using an incremental assignment strategy [S85]. Then a resistive network analogy [DK69] is used to characterize the routability of nets. The combination of the edge cost model and the resistive network model provides a very effective guidance for partitioning.

2.4 Detailed Routing

The detailed router places the actual wire segments within the region indicated by the global router, thus completing the required connections between the terminals. The detailed routing problem is usually solved by considering a region at a time. Regions generally are of the form of ordinary channels and switchboxes. Depending on the placement of blocks we can mark the routing regions and their ordering in which the routing will be done.

A routing region may have terminals only on two opposite sides of the region. This type of routing region is called channel. If the routing region is of rectangular shape, then it is called straight channel. An L-shaped channel consists of a region which is of L-shaped form. Switchbox is a rectangular routing region where the terminals are on all four sides of the region.

Characteristics of routing problem largely depend on the topology of the routing region. Regions may contain one or more layers. An electrical connection between the segments on adjacent conducting layers is made by etching a hole in the insulator before manufacturing the second conducting layer, and such a connection is called via. Multilayer routing problem is NP-hard [S85a] even when number of layer is one [R84]. One obvious constraint in routing problem is that no two wires from different nets are allowed to cross each other on the same layer. Various detailed routing algorithms have been developed depending on the nature of the problem and design goals.

2.4.1 Routing considerations

There are many parameters in a routing problem which are usually determined by design rules and routing strategy.

Two- and multi-terminal nets : Though simple algorithm may exist for two-terminal nets, practical algorithms must be designed for handling multi-terminal nets.

Net width : The width of a net depends on the layer it is assigned to, and its current carrying capacity. Usually routers must allow power and ground nets for such width variation.

Via restriction : Vias are necessary to connect the wires of the same net on different layers. Vias still remain a concern in routing problem as they increase manufacturing cost, decrease yield and performance. The physical dimension of a via is usually larger than the width of connecting wires in today's manufacturing technology. Thus minimizing vias increases chip space usage.

Number of layers : Generally fabrication processes allow at least two metal layers for routing. Increase of layers increases the manufacturing cost and design complexity, but at the same time reduces chip area. In MCM routing, a large number of layers are available.

Boundary type : A boundary is the border of the routing region which contains the terminals. Most detailed router assume that the boundaries are rectangular. If the terminals are on two opposite sides of the boundary, then they are said to form a channel. Terminals appearing on four sides form a switchbox. In three-sided switchbox, the terminals lie on three sides of the boundary. Boundaries may be irregular in shape. Some recent routers [C86, CK86, VCW89] have the capability of routing within irregular boundaries.

Net types : Power, ground and clock nets are considered critical nets. Since power and ground nets are normally wider than signal nets, they need special consideration. Clock nets require very careful routing preference, as the delay of the entire chip may depend on clock routing. Critical nets are routed before signal nets using specialized routers.

2.4.2 Routing models

The most common model used in routing is the grid-based Manhattan model, where a rectilinear grid is superimposed on the routing region, and wires are restricted to follow path along the grid lines. A horizontal grid line is called a track, and a vertical grid line is called a column. If in addition, 45^0 diagonal lines, are present in the grid, the model is known as Manhattan diagonal (MD) model [DB96]. A routing model that does not use an underlying grid, is known as a gridless model.

In the grid-based approach, terminals, wires and vias are required to conform to the grid geometry. For grid-based router, different grid size can cause a via-alignment problem between different layers. The gridless approach allows arbitrary location of terminals, nets; vias and nets are allowed to have arbitrary widths. Due to these advantages, the gridless approach has gained more popularity than the grid-based approach [CK86].

Routing problems can also be modeled based on the layer assignment of horizontal and vertical segments of nets. In *unreserved layer* model, any net segment along horizontal, vertical, or 45^0 diagonal tracks can be placed in any layer. Any such segment is allowed in a given layer, as long as two different net segments do not overlap. However, two net segments can overlap if they are in different layers. In this model, routing space can be used more efficiently, and the number of vias is reduced significantly. We have used this model in the thesis extensively. In Manhattan model with reserved layer wiring, only horizontal or vertical segments are allowed to appear in a particular layer. In other words, each layer contains either horizontal or vertical wire segments but not both. In Manhattan-diagonal reserved-layer model proposed in this thesis, all diagonal net segments are assumed to appear in the layer where the horizontal segments are assigned. This restriction simplifies routing algorithms considerably and provides excellent solutions since vertical segments do not block horizontal segments and vice versa, as they are on different layers. Layers assigned to horizontally oriented segments are called horizontal (H) layers and those reserveded for vertically oriented segments are called vertical (V) layers. In the two layer routing H-V or V-H wiring model is used. When three layers are available, HVH, VHV models are used in general.

With the advent of two and a half layer technology used in the layout design of standard cells, several routers now-a-days use the extra routing area inside the cell area for interconnection. These models are referred to as over-the-cell routing which renders considerable savings in channel width [CL88, CPL90, HSS91, LPHL91].

2.4.3 Channel routing

A channel routing problem is specified by channel length, net list of the channel, and the number of layers. The goal of channel routing is to complete inter-connections of all the nets in the channel satisfying design rule so that the channel width is minimized. Optimization of other secondary objectives, e.g., minimizing the number of vias, wire length, length of the longest wire are often required for higher performance and good yield.

We describe here few basic definitions and terminologies related to channel routing.

Channel area: The space between two adjacent parallel rows of cells.

Top (bottom) terminal list: $T = (t_1, t_2, ..., t_m)$ $(B = (b_1, b_2, ..., b_m))$, where t_i (b_i) is a positive integer identifying the nets that enters through the i-th column in the channel at the top (bottom). If t_i $(b_i) = 0$, no net has to be connected to the ith terminal.

Left (right) connection set: The set of nets that enters the channel from the left (right) end of the channel.

Local density and density [YK82] : The local density d_j of column j, is the number of nets crossing that column. The density d of a given channel is defined by $d = \max_{i \leq j \leq m} \{ d_j \}$

Doglegging: When a net is broken into two or more horizontal segments occupying different rows, a vertical segment is used to connect those horizontal segments. This is called *doglegging,* which may be necessary to complete routing, and to reduce channel width. Doglegging however, tends to increase vias.

Vertical constraint graph: A directed graph G(V, E), whose nodes V represent horizontal segments. A directed edge $(i,j) \in E$ implies that horizontal segment *i* must be placed above the horizontal segment *j*, because of a vertical constraint. If G is cyclic, routing requirement cannot be satisfied without doglegging in two-layer reserved model.

2.4.3.1 Single row routing problem

In this scenario, a set of evenly spaced terminals of two or multi-terminal nets appear on a real line, which is called the net axis. The single row routing problem is to realize the interconnection of the nets by means of non-crossing paths which are basically the horizontal and vertical line segments. All paths are placed on a single layer so that no two paths cross each other and each path cannot occupy two tracks at any column. The area above and below the node axis is called the upper street and lower street respectively. The number of tracks available for routing in these areas is called their capacity. For a given realization, the number of horizontal tracks needed in these area is called upper and lower street congestion. The most common objective function is to minimize the maximum of upper and lower street congestions. The single row routing problem was introduced by So [S74], for the layout design of multilayer circuit board. Later on, extensive studies on this problem were made in [HS84, KKF79, RS83, RS84, TKS76, TKS82, TSK84]. In [SD89], Sherwani and Deogun developed a new graph-theoretic approach for this problem.

2.4.3.2 Single layer routing algorithm

Given a set of blocks with terminal of different nets, placed on a routing region, the objective of the single layer routing problem is to establish

(i) connections (routing) among the terminals of each net in the same layer,

(ii) the routing of different nets are non-overlapping and does not violate the specified design rule.

Here the problem is to determine whether all the nets can be routed in the same layer. Although the general single layer routing problem is conceptually easier than the multi-layer routing problem, it is still computationally hard. Different cases of this problem are handled in polynomial time [BP83, DKS$^+$87, LP83, M90, SD81, T81].

General river routing problem

River routing is a special case of single layer routing, where all terminals lie on the boundary of the region and every net consists of exactly two terminals and there are no blocks in the region. A solution exists if terminals are located in such a way that nets do not cross over. In [H83a, JP89, LP83, ST83, TH90], the formulation of river routing problem was considered. Hsu [H83a] proposed an algorithm for general river routing problem which is capable of routing arbitrary shaped rectilinear regions, and guarantees a solution if one exist. The algorithm is gridless and allows arbitrary net widths and wire separations.

2.4.3.3 Two-layer channel routing algorithm

The main objective of routing is to reduce the routing area by minimizing the channel height. Let h_{max} and v_{max} represent the maximum clique in horizontal constraint graph (HCG) and the longest path in vertical constraint graph (VCG) respectively, for a routing instance. Then lower bound on the number of tracks of a two-layer dogleg free routing problem is $max(h_{max}, v_{max})$. The algorithms are mainly based on left-edge, graph based, greedy or hierarchical routing. Left-edge algorithms (LEA) [HS71] start with sorting the trunks from left to right and assign the segments to a track such that no two segments overlap. Some algorithms use various graph modeling, such as Interval graphs, permutation graphs etc., to assign nets to tracks. The algorithm in [RF82] uses a greedy strategy to route nets in the channel. It starts with the leftmost column and works towards the right end of the channel by routing the nets one column at a time. The hierarchical router generates the routing in the channel by repeatedly bisecting the routing region and then routing each net within the smaller routing regions to generate the complete routing.

The first algorithm developed for channel routing was the left-edge algorithm, proposed by Hashimoto and Stevens [HS71] which uses a reserved layer model and does not allow doglegs and any vertical constraints. Consequently, it does not allow cyclic vertical constraints. Deutsch [D76] proposed an algorithm known as dogleg router by observing that the use of doglegs can reduce the channel height. Like LEA [HS71], the dogleg router uses a reserved layer model. Experimentally dogleg routers achieve far superior results as compared to LEA, often requiring very few tracks beyond the channel density.

Sangiovanni-Vincentelli and Santomauro [VS82] proposed YACR (yet another channel router) which operates under the assumption that a user may add vertical tracks when they are needed anywhere within a channel. Clearly one can achieve the channel width equal to density under these assumptions, since a vertical column may be added every time there is a vertical conflict, after the horizontal segments are packed by left edge routine. YACR does not use dogleg and attempts to minimize heuristically the number of added columns. Experimental results demonstrate improvement in overall routing area with fewer vias over routers operating within classical model. YACR-2 is the follow-up version of YACR, where it is not allowed to add columns but it is allowed to perform horizontal jogs on the vertical layer which may result in wire overlap. The algorithm works in two phases : horizontal track assignment and maze running. YACR-2 is very fast and practical in most cases, since most of the technologies allow limited wire overlap.

Yoshimura and Kuh [YK82] presented a new channel routing algorithm based on net merging. This work is the first attempt to analyze the graph theoretic structure of the channel routing problem. It considers horizontal and vertical constraint graphs and assigns tracks to nets so as to minimize the effect of vertical constraint chains in the vertical constraint graph. It does not allow dogleg and cannot handle vertical constraint cycles. The main steps in this algorithm are zone representation, net merging to minimize vertical constraint chains, and track assignment.

Chen and Kuh [CK86] first proposed a gridless variable-width channel router called Glitter. Glitter can utilize multiple layer technology and design rules. Terminals can be located at arbitrary positions and can be located on off-grid points. No columns or tracks are used in routing. Only the wire width, spacing, and via size are the factors in the routing algorithm. It is a reserved-layer routing model and is considered similar to net merge algorithm which uses a weighted constraint graph formed by using vertical and horizontal constraint graph.

Assigning the complete trunk of a multiterminal net severely restricts LEA and dogleg routers. Optimal channel routing can be obtained if it can be guaranteed that for each column, there is only one horizontal track per net. Based on this observation, one approach to reduce channel height could be to route nets column by column trying to join split horizontal tracks (if any) that belong to the same net as much as possible. Based on the above observation and approach, Rivest and Fiduccia [RF82] developed greedy channel router. The algorithm starts from the leftmost column and places all the net segments of a column before proceeding to the next right column. In each column, the router assigns net segments to tracks in a greedy manner. The greedy router allows the doglegs in any column of the channel, not necessarily where the terminal of the doglegged net occurs.

In [BP83], Burstein and Pelvin presented a two-layer channel router based on the reduction of the routing problem in (m x n) grid to the routing in (2 x n) grid and consistent utilization of divide and conquer approach.

Kowamoto and Kajitani [KK79] proposed an algorithm that guarantees routing with an upper bound of number of tracks equals to the density plus one, but additional columns are needed to complete the routing. Ho, Iyenger and Zhenq [HIZ91] developed a simple but efficient heuristic channel routing algorithm, which is greedy in nature.

All algorithms are not equally good in nature. Many benchmark examples can certify the quality of channel router. In most famous deutsch difficult example, dogleg router uses 21 tracks and 346 vias. In the same example YK-router takes 20 tracks and 403 vias and greedy router takes 20 tracks and 329 vias. Most popular hierarchical and YACR2 take 19 tracks each, and 326 and 287 number of vias respectively.

2.4.3.4 Multi-layer channel routing algorithm

With the advance in VLSI technology, utilization of more than two layers for signal routing has become feasible. Most of the current gate-array technologies use three layers for routing. For example the Motorola 2900ETL macrocell array is a bipolar gate array which uses three metal layers for routing [S93].

Both the reserved layer and unreserved layer model are considered for three-layer routing algorithm. The reserved layer model can be classified into the VHV model and the HVH model. In the three layer VHV model the lower bound on the number of tracks for a routing problem is h_{max}, whereas in HVH model, the lower bound on the number of tracks for a routing problem is $max(v_{max}, h_{max}/2)$.

In [CL84] Chen and Liu presented a three layer channel router based on the net merging method and the left edge algorithm introduced by Yoshimura and Kuh [YK82] for their routing algorithm. As no vertical constraint exists in VHV, in this context, the left edge algorithm is sufficient. But in the HVH routing, vertical constraint exist, so the concept of (net) merging algorithm was extended to the HVH problem. In [CWL88], Cong, Wong, and Liu presented a general technique that systematically transforms a two-layer routing solution into a three-layer routing solution which is very similar to the Yoshimura and Kuh algorithm. Chen [C86] proposed a three-layer gridless router, called

Trigger, which is an extension of *Glitter*, the gridless router appeared in [CK86].

In [PZ87] Pitchumani and Zhang developed a three-layer channel router that combines both HVH and VHV models, based on the idea of partitioning the channel. In this approach the channel can be thought of as two separate channels, not necessarily of the same size. One portion (upper or lower) is routed using the VHV and the other portion is routed using the HVH model. A transition track is usually needed between the two portions. The algorithm does not allow any dogleg. This algorithm is a generalization of pure HVH and VHV approaches. This algorithm partitions the given netlist into two netlists, such that each netlist is best suited for either VHV or HVH style of routing. It then routes them separately thus obtaining two sub-solutions. The algorithm then insert transition tracks to complete the connections between the two routed sub-solutions.

In [H85], Hambrusch presented an algorithm for a n-layer channel router. The number of layers, the channel width, the amount of overlap and the number of contact points are four important factors for routing multi-terminal nets in multi-layer channels. An insight into the relationship between these four factors is also presented in [H85]. In [BBD+86], Braun developed a multi-layer channel router called Chameleon, based on YACR2. The main feature of Chameleon is that it uses a general approach for multilayer channel routing. Stacked vias can be included or excluded, and separate design rules for each layer can be specified.

In [ED86], Enbody and Du presented two algorithms for n-layer channel routing that guarantee successful routing of the channel for $n > 3$.

2.4.3.5 Via minimization and over-the-cell channel 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. Different via minimization approach is followed for reducing vias in a layout design. In constrained via minimization problem (CVM) we are given a set of wire segments, and k layers for routing, where the placement of wire segments has already been determined by some router. 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 unconstrained via minimization problem (UVM), the placement and the layer assignment of segments are not given. The problem is both the placement of the segments and assignment of layers so as to minimize the total number of vias. UVM is also known as topological via minimization problem (TVM).

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]. Chang and Du [CD87] developed a heuristic algorithm by splitting vertices in a graph. Kajitani [K80] 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. Pinter [P82] proposed an optimal algorithm for two-layer CVM problem when the maximum junction degree is limited to three.

The TVM problem was first introduced by Hsu in [H83]. Marek-Sadowska [S84] proved that the TVM problem is NP-complete. Sarrafzadeh and Lee [SL89] and Cong and Liu [CL91] considered the crossing-channel TVM problem, and shown that the Crossing-channel K-layer TVM problem is solvable in polynomial time.

2.4.4 L-shaped channel and switchbox routing

L-shaped channel and switch box routing problems are the generalization of the channel routing problem. In L-shaped channel routing problem, terminals are located in the boundary of the L-shaped region. Kajitani [K83] first introduced the concept and term of safe routing order and studied the problem analytically. In order to overcome cyclic dependencies in routing order for nonslicible topologies, L-shaped routing regions [DAK85, SB91] were introduced. Switchbox is a four sided enclosed region where terminals are located on all four sides and routing must be completed within this region.

Algorithms by Maddila et al. [MZ89] for general routing regions are based on the restrictive knock-knee model [S93]. Tsai et al. [TCC$^+$92] formulated general area routing as a planning problem in which the routing process is decomposed into a conjunction of subgoals; each subgoal consisting of selection of net segments and assignment of track resources. However, it may need to backtrack and may not generate any solution as it uses the Manhattan model even if Manhattan-diagonal routing exists. The time complexity of this method is high. Among non-Manhattan geometries, a diagonal model with $\pm 45^0$ wires has been adopted [LLS91, C87]. Song [S92] presents an optimal knock-knee diagonal routing for L-channel with two-terminal nets only. In

Manhattan-diagonal (MD) model which combines diagonal and Manhattan geometries, routing method for straight channels based on sorting was proposed in [CR91]. Another MD channel router was presented by Wang [W91] but the track count and via count for denser channels seem to be higher and too many jogs are required. Recently, Das et al. gave a simpler MD channel router providing reduced wire length and vias [DB96]. Nevertheless, applicability of these methods to realistic routing in general regions are to be studied.

In general, switchbox routing is more difficult than standard channel routing. Theoretically, it falls into the category of NP-hard problems [S93]. Several heuristics are developed for the problem. Hamachi and Ousterhout [HO84] proposed a good switchbox router. The idea of Greedy channel router can be naturally extended to do the switchbox routing [L85]. Hierarchical channel router can be modified by introducing alternate partition in both vertical and horizontal directions instead of partitioning along the length of the channel [BP83]. One of the best practical switchbox router was proposed by Joobani [J86]. His algorithm WEAVER, was a successful knowledge-based expert-system router. Cohoon and Heck developed a computational-geometry-based tool BEAVER for switchbox routing.

2.5 MCM Routing

To get the advantage of advance fabrication technology, it is reasonable to consider new technique for packaging which can increase the packing density. Packaging has an important role in determining interconnection delay, cost, speed, reliability of high-speed system and the simplification of interconnection in the PCB. Recently, the multi-chip-module technology (MCM) has been developed that provides a common substrate where several bare chips can be mounted on the top layer. The substrate consists of multiple routing layers used for chip-to-chip interconnections. MCM routing problem is much more difficult in comparison with the conventional IC or PCB routing problem.

In MCM, chips are flip mounted on the top layer via solder bumps. The next few layers are called redistribution layers which are required to connect the chip bonding pads through the surface pads to the signal wires. Next, there are several interconnection layers. The MCM routing problem is a three-dimensional area routing problem where routing can be carried out almost everywhere in the entire multilayer substrate.

There are several new and interesting design automation problems associated with this technology [D90, HSV$^+$90, PPC89]. Cho and Sarrafzadeh [CS91] investigated the pin redistribution problem on a uniform grid that arises in the design of MCM. There exist several methods for MCM routing, after pin redistribution step [CLR$^+$94, D90, HSV$^+$90, HYY90, KC92, KC95, MBG$^+$91, SK92]. The approaches in [CLR$^+$94, HYY90, MBG$^+$91] extend the traditional maze routing technique [L61]. The methods in [HSV$^+$90, SK92] uses layer assignment in order to overcome the size and net ordering. Here a global routing instance is obtained and then each net is assigned to layers so that wires of different nets do not intersect each other. Recently Khoo and Cong developed two powerful routers, namely SLICE [KC92] and V4R [KC95], which iteratively process the nets. Cha et al. [CRN97] propose a simple router SERGA. An improved technique for via minimization in MCM has been reported in this thesis.

Chapter 3
Over-the-cell Channel Routing

3.1 Introduction

The two-layer channel routing is the most widely used technique for interconnecting nets in VLSI design. With the advent of two and half layer technology used in the layout design of standard cells, several channel routers now-a-days use the extra routing area available outside the channel for interconnections. These routers are called *over-the-cell* channel routers which render considerable savings in channel width [DG80, SS87, GN87, CL88]. Over-the-cell channel routing problem is known to be NP-hard [GN87]. The best known heuristic algorithm for over-the-cell channel routing appeared in [CL88], which consists of three major phases in the following sequence: (i) routing over the cell, (ii) choosing net segments within the channel and (iii) routing within the channel.

It is easy to observe that the first two steps are highly interrelated. Routing a net over the cell may force some other nets to be routed within the cell through the heavily congested area. Thus maximizing the number of over-the-cell candidates as suggested in [CL88] may not lead to a good solution. In this thesis we describe a new heuristic algorithm which iteratively selects over-the-cell nets with an ultimate objective of minimizing channel width. Experimental evidence shows considerable improvement over earlier results [CL88] in the number of tracks used both inside the channel and over the cell.

3.2 Formulation of the Problem

Consider a rectangular channel with n columns and two rows of terminals along its top and bottom sides. A number between 0 and m is assigned to each terminal. Terminals with the same number i $(1 \leq i \leq m)$ are to be connected by net i, while those with number 0 designate unconnected terminals.

We assume two routing layers within the channel and a single layer over the cell. Routing consists of only rectilinear wire segments. Inside the channel, all horizontal segments lie on one layer and all vertical segments on the other. Interconnection between horizontal and vertical layers is accomplished through via holes. We also assume that the terminals are accessible only at the edges of the cell. Given a netlist, our objective is to generate a routing with possible over-the-cell interconnections which minimizes the channel width i.e., the number of horizontal tracks inside the channel. Fig. 3.1 displays an example of conventional versus over-the-cell routing.

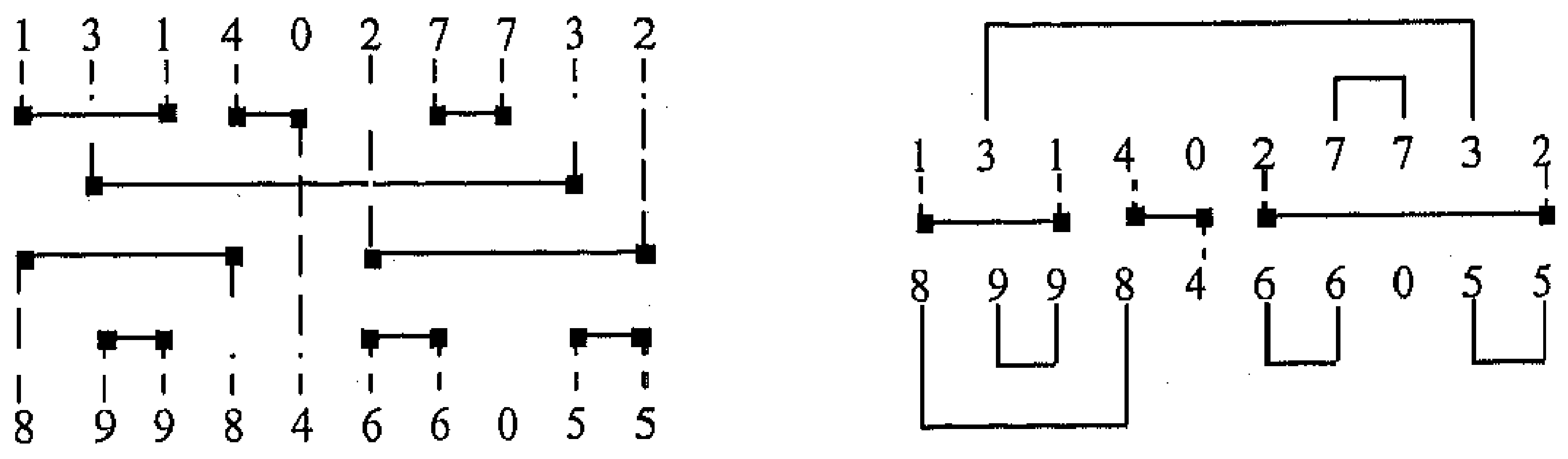


Fig. 3.1 : Conventional vs Over-the-cell channel routing.

3.2.1 Basic steps

The proposed method for over-the-cell channel routing consists of three phases which are performed in a sequence different from that of [CL88] : (i) initial terminal selection, (ii) over-the-cell routing by a new heuristic, (iii) routing within the channel.

3.2.1.1 Initial terminal selection

Nets having terminals on both sides of the channel should have at least one connection across the channel. Thus, if a net having multiple terminals on one side routed over the cell, it leaves a problem of selecting a net segment with two terminals of this net one from the top and other from the bottom of the channel. Appropriate choice of these net segments which minimize the channel density is NP-hard, and was accomplished earlier with a heuristic procedure [CL88]. In our work, we first select initial terminals by suitable heuristic which attempts to minimize the channel width and then go for deciding the over-the-cell candidates.

Definition : The *congestion* of a column *r* is the number of nets passing through that column and is denoted by c_r. The *density* of a channel is the maximum of c_r's over all columns.

Given a net v, let there be p terminals on the top side of the channel occupying column positions v_i^t for i=1,...,p (t stands for top) and q terminals on the bottom side of the channel occupying the column positions v_j^b for j=1,...,q (b stands for bottom). In over-the-cell channel routing, as the terminals appearing in the top (bottom) side are connected over the cell, here the problem is to select a suitable terminal from the top and another from bottom for connecting the two sets.

For each pair of terminals (v_i^t , v_j^b) we now define the cost function w_{ij} as :
$w_{ij} = \sum_{r=1}^{n} \delta_r.(b_r^2 + c_r)$ where *r* stands for column number,
$\delta_r = 1$, if *r* lies within the interval $[v_i^t , v_j^b]$ and 0 otherwise;
b_r = number of occupied tracks in the *r*-th column by net segments which are selected so far for routing through the channel.
c_r = the congestion of the *r*-th column, as mentioned earlier. Note that, c_r is computed prior to over the cell routing.

Given a net v, we select a terminal pair (v_x^t , v_y^b) on the upper and lower sides of the channel respectively, for connecting inside the channel if

$$w_{xy} = \min_{i=1}^{p} \ \min_{j=1}^{q} \ w_{ij} \qquad \text{..........(1)}$$

The rationale behind choosing such an weight function is that we encourage the connection of a net among the top and bottom sides of the channel through a track within the interval $[v_i^t , v_j^b]$ if that portion of the channel is less occupied till now (i.e., b_r's are less), and are also less congested (i.e., c_r's are less). Also our weight function implies that the first criteria is more important than the second one.

First, nets having exactly one terminal on the top and the bottom sides are considered. They contribute towards the initial values of b_r's. For connecting the nets having terminals on both side of the channel, we select the nets in non-decreasing order of their spans. For each net, an appropriate interval is chosen for connecting inside the channel as described above. If more than one pair of terminals (x,y) for a net have the same minimum weight, then conflict may be resolved in favour of one, whose length of the longest path in the vertical constraint graph (VCG) is minimum. Finally, after interconnecting a net ν across the channel, the b_r values are updated for the columns through which the net ν passes. This process is repeated till all nets are considered.

3.2.1.2 Over-the-cell routing

Nets having two or more terminals on one side of a channel are candidates for over-the-cell routing. In [CL88], a graph theoretic criteria for choosing candidates (nets) for over the cell routing is formulated using circle graph G_C, where the terminals of both the sides are placed along the boundary of the circle. Pins of same net are joined by chords. In the graph, we put an edge between two nodes if the chords corresponding to those two nets intersect. In that paper [CL88], it is suggested that the maximum independent set of the above mentioned circle graph indicates the set of net segments for over the cell routing. However, the following example (Fig. 3.2) shows that choosing of maximum independent set may often yield a worse solution. Experimentation with many benchmark examples suggests the following empirical observations :

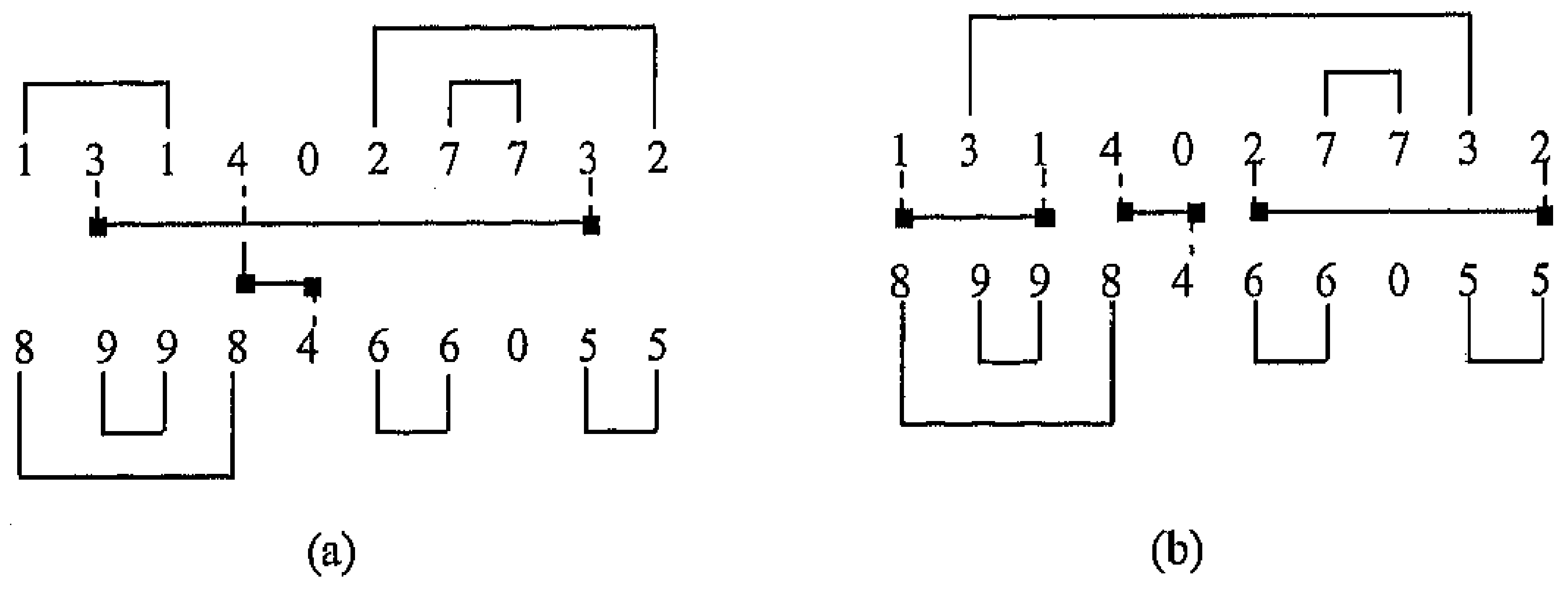


Fig. 3.2 : Over-the-cell routing : (a) [CL88], (b) proposed method

Observation 1 : For minimizing channel width, nets passing through the columns having maximum congestion should preferably be selected for over-the-cell routing.

Usually, the congestion of tracks is described by an interval graph G_I[YK82], where nodes correspond to nets. Two nodes are connected by an edge if the corresponding horizontal spans determined by the leftmost and the rightmost terminals of the two nets overlap. Note that the maximum clique in G_I defines the density of the channel. In order to reduce the channel width, we choose those net segments for over the cell routing which immediately reduce the density of the channel.

It is often better to route a portion of the chosen net over-the-cell instead of routing the whole (see the routing of net 3 in Fig. 3.3), because this may subsequently allow another net (net 4 in the same figure) to be routed over the cell causing a further reduction of channel width.

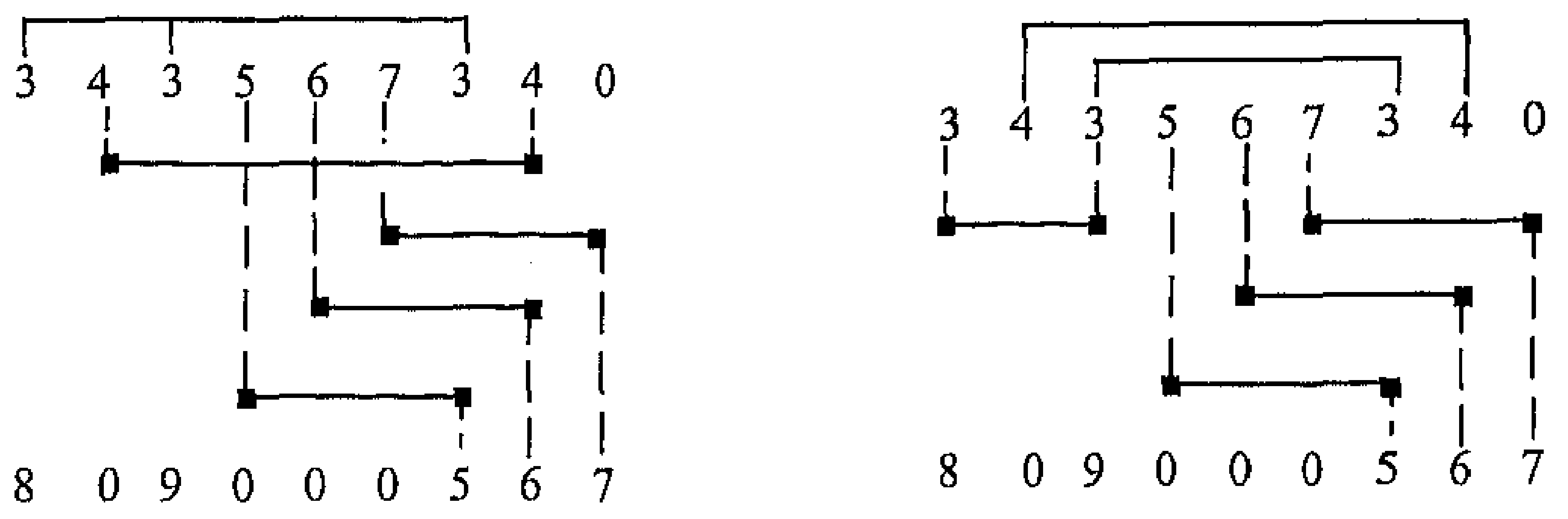


Fig. 3.3 : Routing a net segment over-the-cell

We now define a *net interval* and an *overlap graph* which capture all useful information regarding the formulation of the problem.

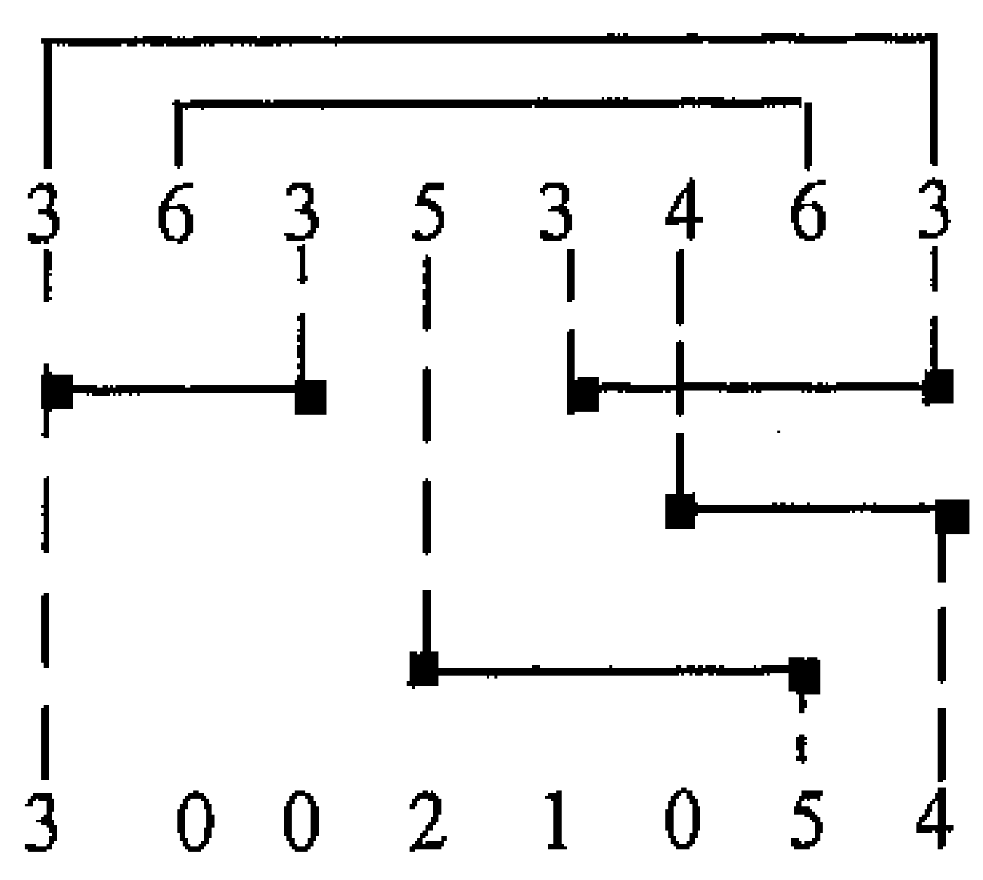
Definition : An interval [a, b] is a *net interval* if a and b are two terminals of the same net and belong to the same side of a channel. Thus a net having p terminals on one side creates $\binom{p}{2}$ net intervals on that side. Each net interval is a possible candidate for over-the-cell routing at an instant of time.

Definition : An *overlap graph* G_o is a graph whose nodes correspond to net intervals. Here, two such intervals $[a_1, a_2]$ and $[a_3, a_4]$ are said to overlap if $a_1 < a_3 < a_2 < a_4$ or $a_3 < a_1 < a_4 < a_2$. An edge between two nodes implies that the corresponding net intervals (i) overlap, (ii) belong to the same side of the channel, and (iii) are portions of two different nets. In other words, these two net intervals cannot be routed over-the-cell simultaneously.

A net interval corresponding to a node α of the overlap graph G_o chosen for over-the-cell routing forces some net intervals of the other nets to be routed inside the channel. These correspond to nodes adjacent to net α in G_o.

Observation 2 : Routing of a net interval over-the-cell which does not pass through the columns contributing to the maximum congestion, may reduce the density as shown in Fig. 3.4.

Observation 3 : For over-the-cell side one x-x interval routing in over the cell mames y-y interval impossible to be routed in over the cell where terminals are along (..x...y...x...y...). Hence, a priority should be given to such an x-x interval



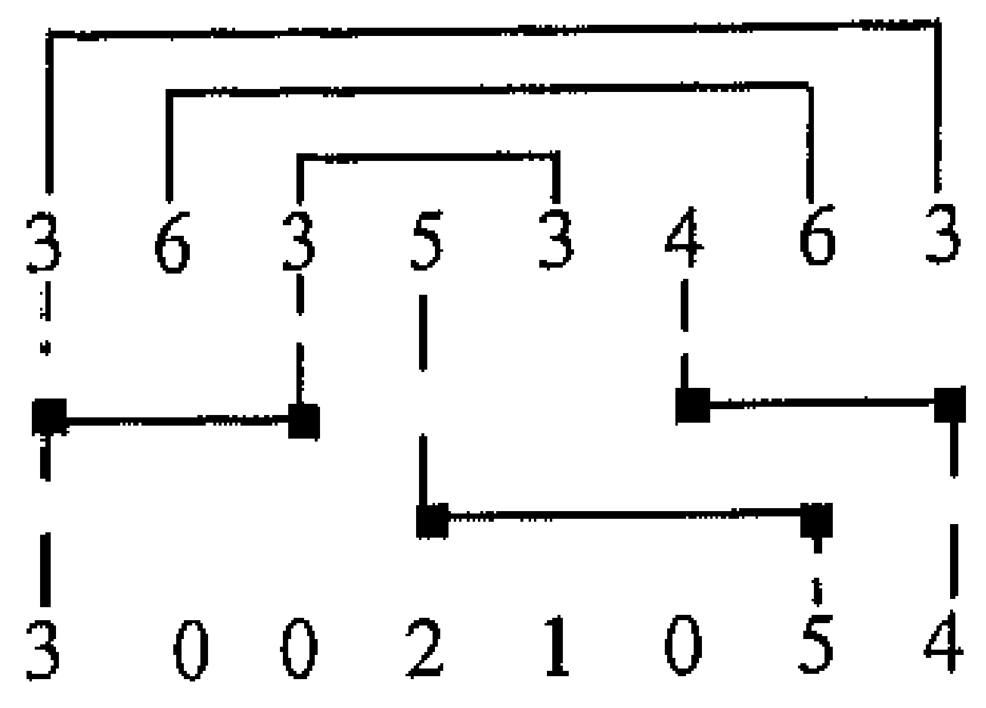


Fig. 3.4 : Illustrating Observation 2

that minimize the number of intervals the number of intervals that cross x-x and among the same count a longer span contributing towards the maximum congestion is a good choice for selection. In Fig. 3.5, notice that either net 4 or net 5 at the bottom of the channel qualifies for over-the-cell routing, but net 5 is a better candidate than net 4, because the former one affects a longer span contributing towards the maximum congestion.

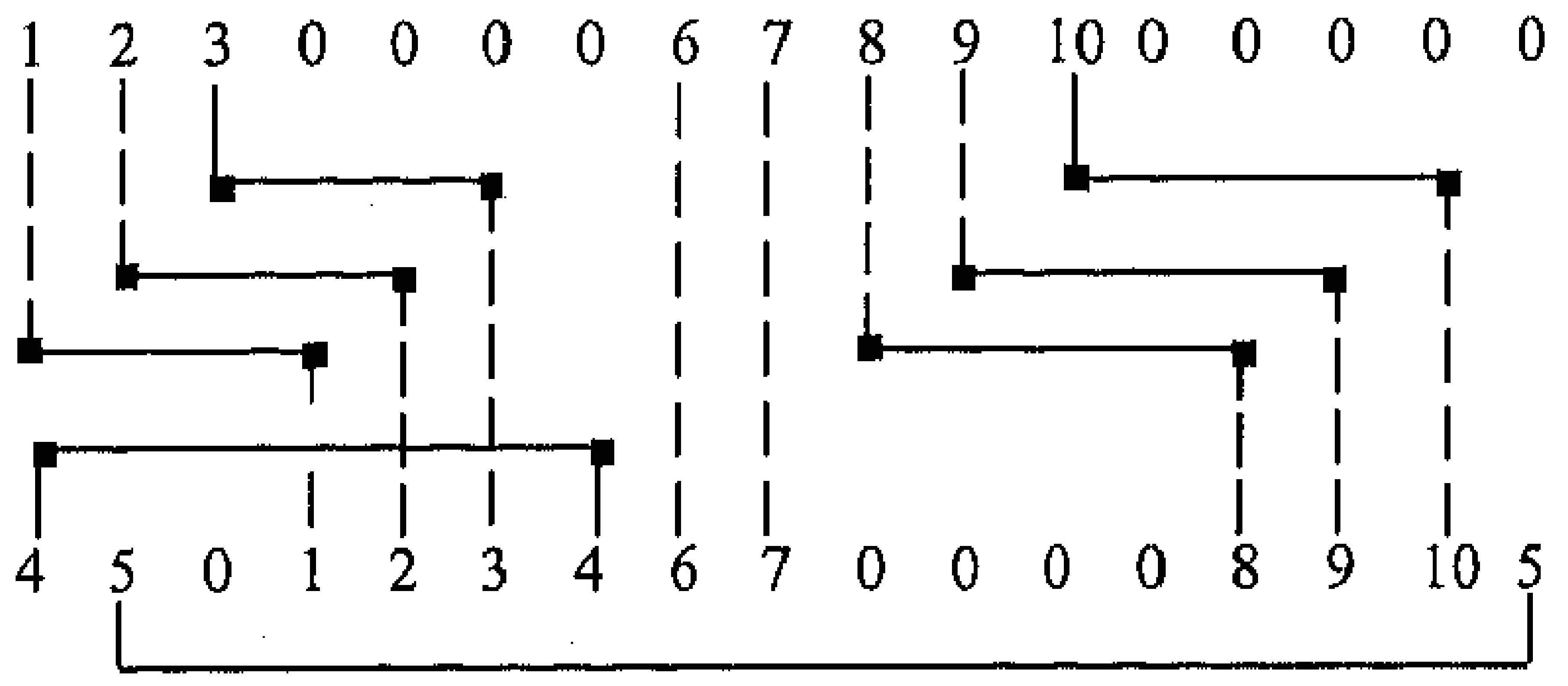


Fig. 3.5 : Choice of over-the-cell candidates based on span

Thus our choice of net intervals for over-the-cell routing is dictated by the following facts :

(i) it should preferably dilute the portion of the channel having the maximum congestion and span, and

(ii) it should subsequently force the smallest set of other nets to be routed inside the channel.

Observation 4 : It may be necessary to route two net intervals over-the-cell one at the top and the other at the bottom simultaneously in order to reduce the density by one. In Fig. 3.6, net 3 should be routed over-the-cell both at the top and bottom. The proposed algorithm in subsection 3.3.1 takes care of this.

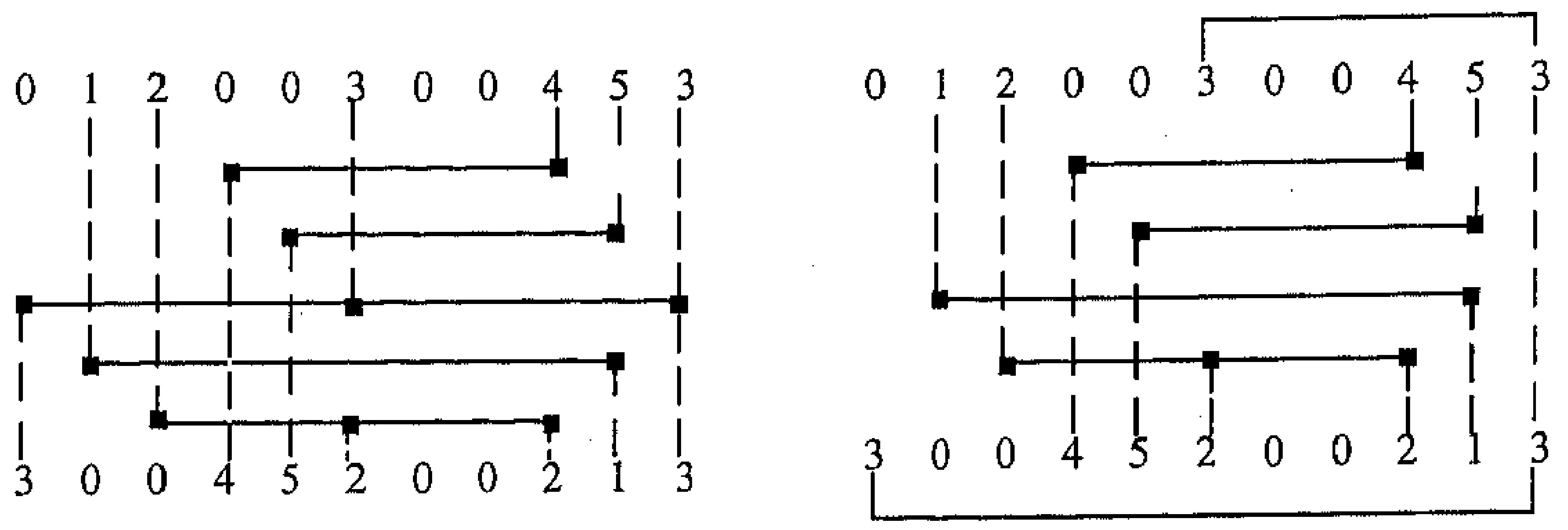


Fig. 3.6 : Top-bottom simultancity of over-the-cell routing

3.2.2 The proposed cost function

Our algorithm selects over-the-cell candidates iteratively. Initially, we start with no net interval passing over-the-cell and then on the basis of the following weighing criterion, a seemingly best net interval is chosen. This process is repeated until no more candidate for over the cell routing is found.

Let d_i denote the current congestion of i-th column, f_i is the number of tracks already occupied by some nets which are forced to be routed inside the channel among the set nets considered up to the current instant of time. Initially, $d_i = c_i$.

The *blocking factor* of net interval β at column i, denoted by θ_i^β, is the number of nets forced to be routed within the channel through column i for routing β over-the-cell. Clearly, $0 \le \theta_i^\beta \le d_i - f_i - 1$. Thus blocking factor of a net interval β may depend on other net intervals which have already been routed over-the-cell, as those nets cannot be chosen for routing through the channel. These are the net intervals corresponding to the nodes which are adjacent to β in the overlap graph.

So, we formulate the weight of a net interval β as follows :

$$w(\beta) = \sum_{i=1}^{n} \theta_i^\beta / (\alpha(f_i^2 + d_i)), \quad \text{where } \alpha = \text{span of } \beta \quad ....(2)$$

It needs to mention that, the motivation of choosing such an weight function is same as that of (1); the only difference is that it encourages a net interval to be routed over the cell which passes through (i) the congested columns, (ii) the numbers of already occupied tracks is more, (iii) forces less number other nets to be routed inside the channel, and (iv) having longer span.

3.3 Heuristic Algorithm

In each iteration of the algorithm, a net interval β *is selected for* over-the-cell routing whose $W(\beta)$ is minimum. This weighing criterion has been empirically chosen such that it reduces the maximum congestion over the longest span and forces minimum number of nets to be routed through the channel with the ultimate goal of minimizing the channel density.

Let a net interval β be routed over-the-cell, such that nodes $\gamma_1, \gamma_2, \ldots, \gamma_k$ of some net γ are adjacent to β, in the current overlap graph. Then it is enough to route only one out of $\gamma_1, \gamma_2, \ldots, \gamma_k$ inside the channel and the rest need not be considered. Net interval with the largest $W(\gamma_j)$ is selected for routing through the channel. Nodes $\gamma_1, \gamma_2, \ldots, \gamma_k$ and β are deleted from the overlap graph. The d_i and f_i values are updated for all columns which are affected in the current iteration. The procedure terminates when the overlap graph becomes empty. However, in the actual algorithm, no overlap graph is explicitly created; its effect is simulated with an alternative data structure.

3.3.1 Method

```
Step 1 : select initial terminals iteratively using expression - (1);
         obtain $b_i$'s for $1 \le i \le n$;

Step 2 : /* Over-the-cell routing */
          Let S be the set of net intervals corresponding to all the nets;
         for all i = 1 to n do   { $d_i = c_i$; $f_i = b_i$; }
         repeat
            for each member $\beta$ of S do
                for all i = 1 to n do calculate $\theta_i^\beta$;
                find weight $W(\beta)$ using expression - (2);
               find the net interval $\beta^*$ having minimum weight;
                route $\beta^*$ over-the-cell;
                for all nets $\gamma$ whose net interval(s) overlap with $\beta^*$ do
                 begin  /* Let $\gamma_1, \gamma_2, \ldots, \gamma_k$ be the net intervals of $\gamma$ */
                    find $\gamma_j \in \{ \gamma_1, \gamma_2, \ldots, \gamma_k \}$ whose $w(\gamma_j)$ is maximum;
                    select $\gamma_j$ for routing through the channel;
                    update $f_i$ (increase by one in all columns spanned by $\gamma_j$);
                    remove $(\gamma_1, \gamma_2, \ldots, \gamma_j, \ldots, \gamma_k)$ from S
                 end;
                 remove $\beta^*$ from S;
                update $d_i$ (reduce by one in all columns spanned by $\beta$);
         until < the set S is empty >;

Step 3 : call greedy channel router [RF82] for routing within the channel.
```

3.3.2 Complexity of the algorithm

The given channel consists of n columns and m nets with density ρ. Let λ and σ denote the maximum number of terminals and span among all nets respectively. $|S|$ denotes the total number of net intervals and let s be the size of the maximum independent set of the initially created overlap graph.

Step 1 can be performed in $O(n^2.\rho.\sigma)$ time. In *step 2*, initialization requires $O(n.\rho)$ time. The number of net intervals in S is $O(n^2)$. Computation of weights for all nodes of the overlap graph i.e., for all net intervals needs at most $O(\rho.\sigma.|S|)$ time. The update time for d_i's and f_i's is $O(\sigma+n.\rho)$. Thus in each pass, the time complexity becomes $O(\rho.\sigma.n^2)$. Since the iteration can be repeated at most s times, the worst-case time complexity of the algorithm is $O(s.\rho.\sigma.n^2)$.

3.4 Experimental Results

We implemented our algorithm together with a greedy channel router [RF82]. Table 3.1 compares the performance of our method with the result obtained in [CL88] on standard benchmarks (in the table below, YK_i denotes the i-th example) taken from [YK82]. The famous Deutsch's difficult example (De) is also studied (Fig. 3.7). Experimental results show that in all these examples our solution consistently outperforms that in [CL88] with respect to the number of horizontal tracks both within the channel and over-the-cell. Since our algorithm is an iterative one, it can readily be adopted in a situation where the number of tracks available over-the-cell is fixed priori [DG80].

Table 3.1 : The performance of our router on standard benchmarks

Example	Original channel density	Number of tracks required			
		Proposed method		[CL88]	
		over-the -cell	inside the channel	over-the -cell	inside the channel
YK_1	12	7	9	7	10
YK_{3a}	15	6	11	9	12
YK_{3b}	17	7	13	7	13
YK_{3c}	18	6	14	7	15
YK_{4b}	17	9	13	9	16
De	19	11	16	16	17

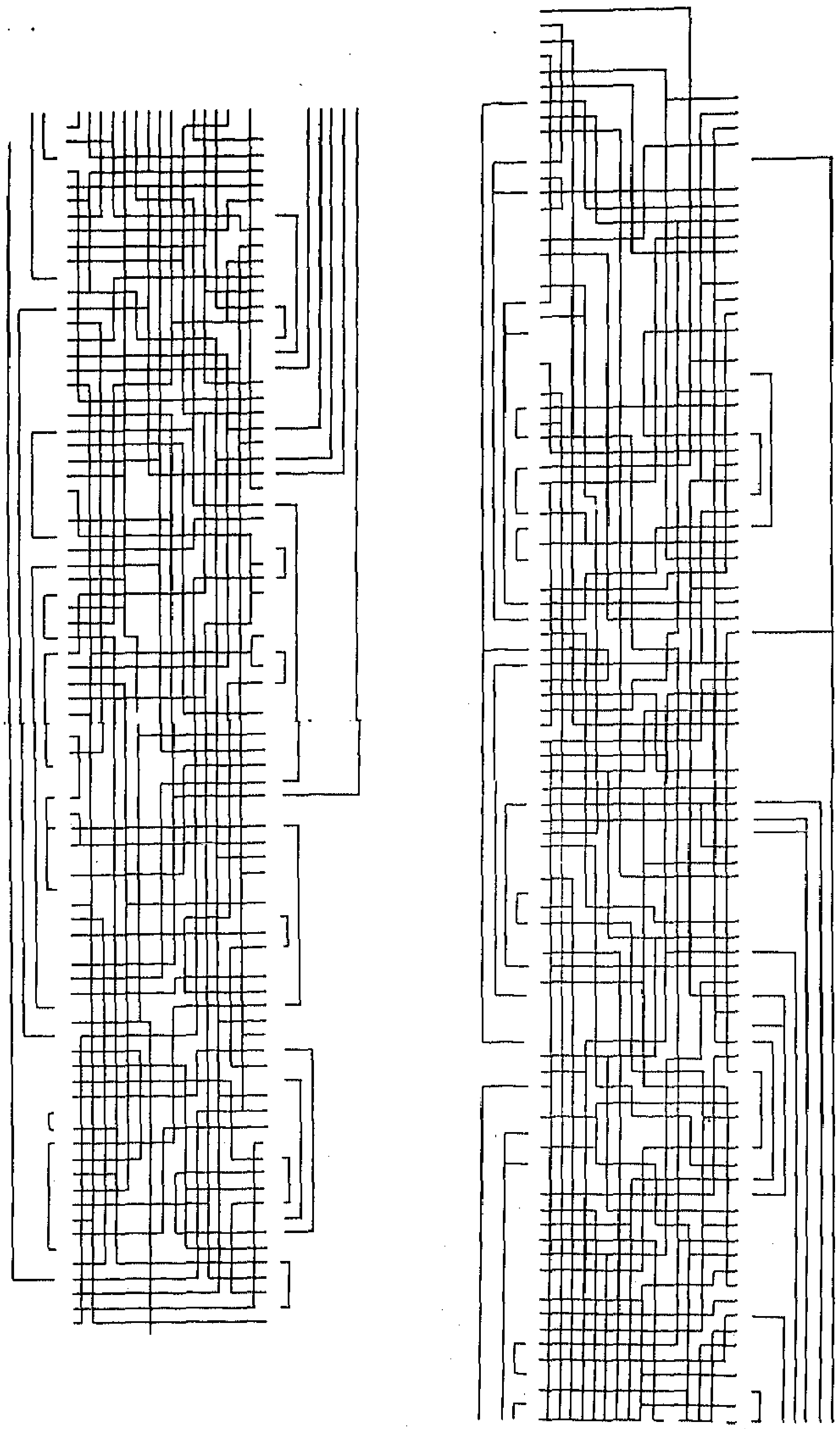


Fig. 3.7 : Deutsch's difficult example

Chapter 4
Via Minimization in Channel Routing

4.1 Introduction

In channel routing of VLSI layout, each signal net is formed by the wire segments interconnecting a set of electrically connected terminals. If multiple conducting layers are available for interconnection, the wires of the same net lying in different layers are connected through vias (contact holes). In two-layer channel routing with reserved layer model, all horizontal wire segments are placed on one layer, and all vertical wire segments are assigned to the other layer. However, this method generates a large number of vias in the final routing solution. The presence of these via holes introduces some problems e.g. (i) besides increasing the manufacturing cost and complexity, vias in a circuit degrade its performance and reliability, and (ii) in a compactible channel routing, the design rule requires one unit spacing for path width, one unit for feature separation, whereas (2 x 2) unit square for a via hole. Thus, minimization of via holes renders a considerable improvement in VLSI layout design in terms of chip area, circuit performance and cost. This is even true if a short run of metal is replaced by polysilicon because the capacitance and resistance of the contact are likely to be greater than those of added polysilicon [CD88].

The existing methods for via minimization are classified in two general categories: unconstrained via minimization (UVM) and constrained via minimization (CVM). Although in most cases UVM [H83, S84] can produce a solution with fewer vias, the layout area is not compact. An example is shown in Fig. 4.1(a) and the layout after geometric mapping is shown in Fig. 4.1(b), the solution in Fig. 4.1(c) requires only three track using just two vias [TWC91].

On the other hand, the CVM approach first starts with a layout that has a small area and then assigns layers to the wire segments such that the number of vias is minimized. Given a collection of paths representing the interconnection wires without specifying the layer assignment, CVM method find a layer assignment to all the path segments that will minimize the total number of vias used. 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]. Chang and Du [CD87] developed a heuristic algorithm by splitting vertices in a graph. Kajitani [K80] 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. Later Pinter [P82] proposed an optimal algorithm for two-layer CVM problem when the maximum junction degree is limited to three. The constraint is that the layout cannot be changed; only the layers to which wire segments are assigned and the positions of vias can be altered in the via minimization process. But the major disadvantage is that the result obtained by the CVM method is minimized only with respect to the given layout. In fact, there may exist many other layouts with the same area and fewer number of vias.

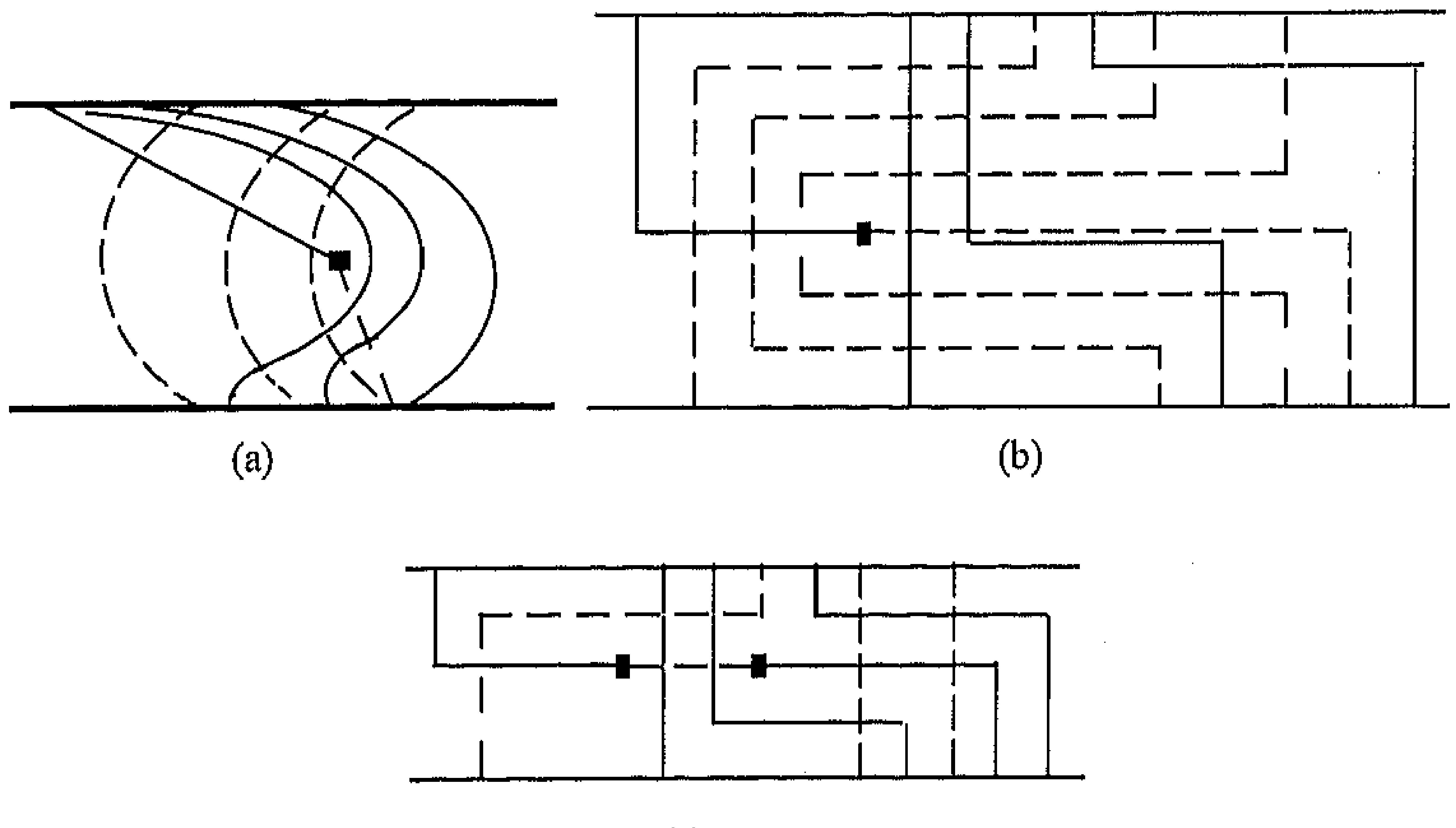


Fig. 4.1 : UVM. (a) Topology of nets with one via. (b) Geometrical mapping
(c) A compact layout with two vias [TWC91]

A better approach to the via minimization problem based on layout modification and assisted by maze router, is introduced in [L61]. Notice that the layout shown in Fig. 4.2(a), can be routed with five vias (Fig. 4.2(b)) when optimized using the CVM method. The layout modification approach [TWC91] yields a solution with three vias as shown in Fig. 4.2(c).

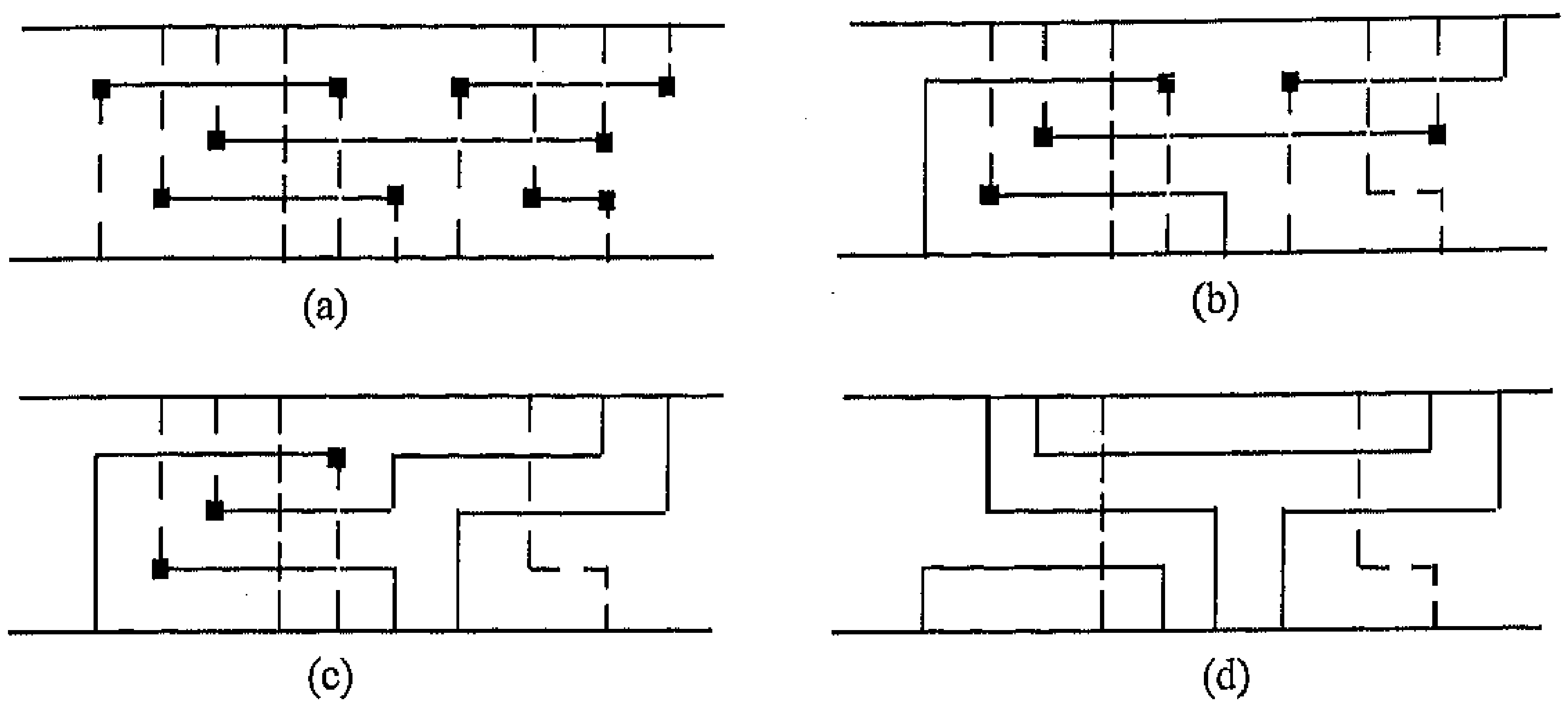


Fig. 4.2 : CVM. (a) Before via minimization. (b) After via minimization.
(c) via minimization by [TWC91] (d) Another layout with fewer vias.

This chapter outlines a new approach to the via minimization problem, which combines the advantage of both CVM optimization method and layout modification. For better utilization of routing space, here we consider the unreserved layer model where both horizontal and vertical net segments may be present in any layer. The model is grid base. The vias can be place in any grid points and they are not necessarily restricted on bends. Here two different net segments can occupy the same track in different layers. The given layout is gradually transformed to a routing instance in this model by a sequence of track interchange; the transformation is guided by the visibility relations existing among net segments. Our algorithm produces better results than those of [TWC91], and at the same time does not increase the routing area (see Fig. 4.2(d)). For all benchmark routing problems we have studied, our solution have significantly fewer number of vias compared to the result of [TWC91], and that of the optimal CVM algorithm [NMN87]. For Deutsch's difficult example, we obtained only 190 vias with 19 tracks, which to the best of our knowledge, is the fewest ever reported. The total wire length which is our secondary objective, also decreases almost in all cases compared to the solutions of [TWC91].

4.2 Layout Modification

Our layout modification approach is based on a visibility property of net intervals, which will be introduced subsequently. Before describing our technique, we first list few observations which will aid our search.
We assume that the net terminals are accessible from both the layers. If the vertical connection of one net does not cross any horizontal connection of other nets, we

can use the same layer for the horizontal connection of that net, thereby reducing vias. Below we state few observations which will aid our search process.

Observation 1 : if the horizontal connection is placed in the first track, then we can avoid vias by putting the vertical connection on the same layer (see Fig. 4.3).

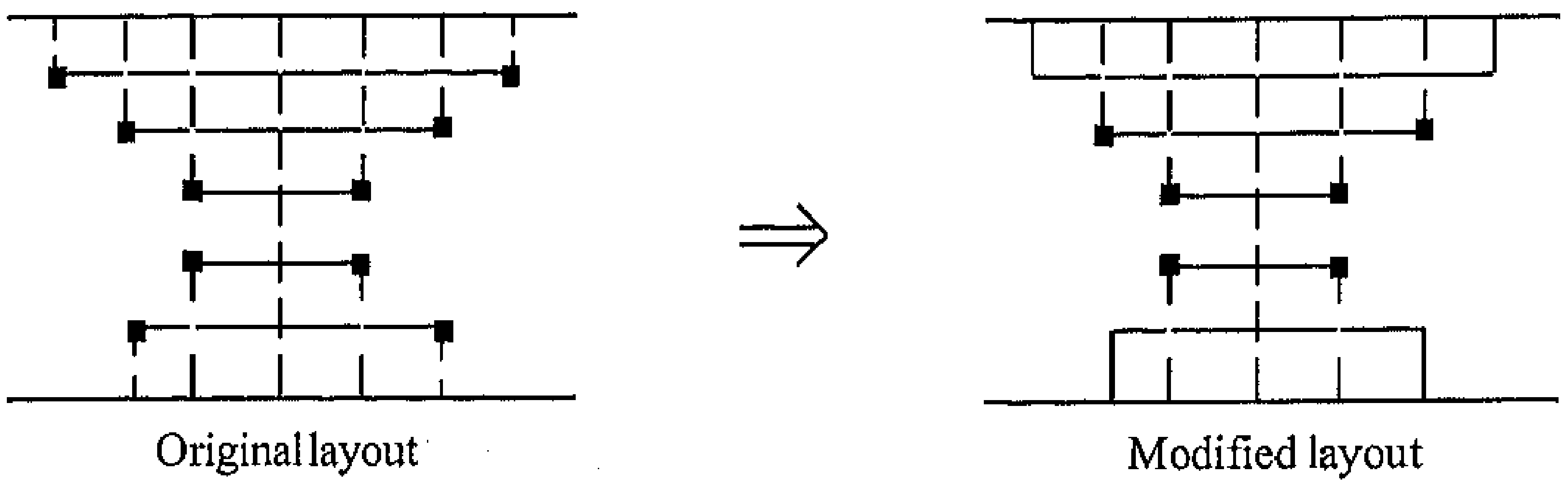


Fig. 4.3 : Example illustrating observation 1

Observation 2 : Via minimization can often be achieved by interchanging track (horizontal) assignments (see Fig. 4.4).

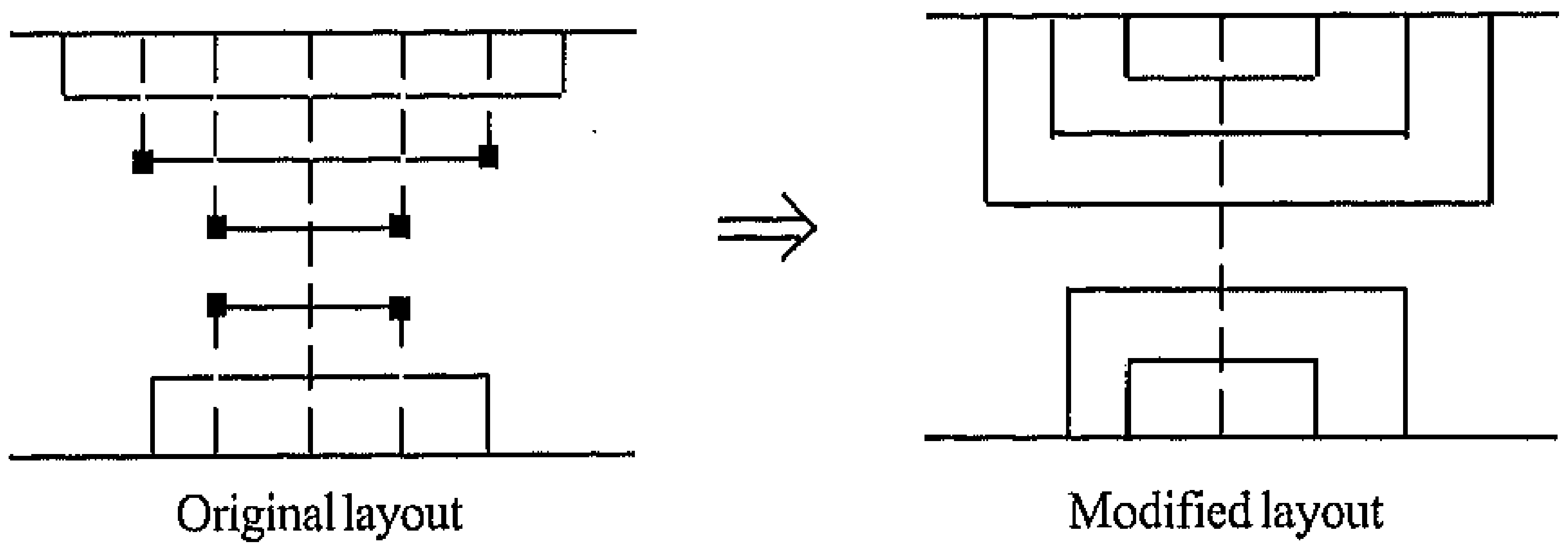


Fig. 4.4 : Example illustrating track-interchange

Observation 3 : It is often better to interchange the track for a portion of the chosen net instead of interchanging the whole. This may subsequently allow another net to be routed on the same layer (see Fig.4.5).

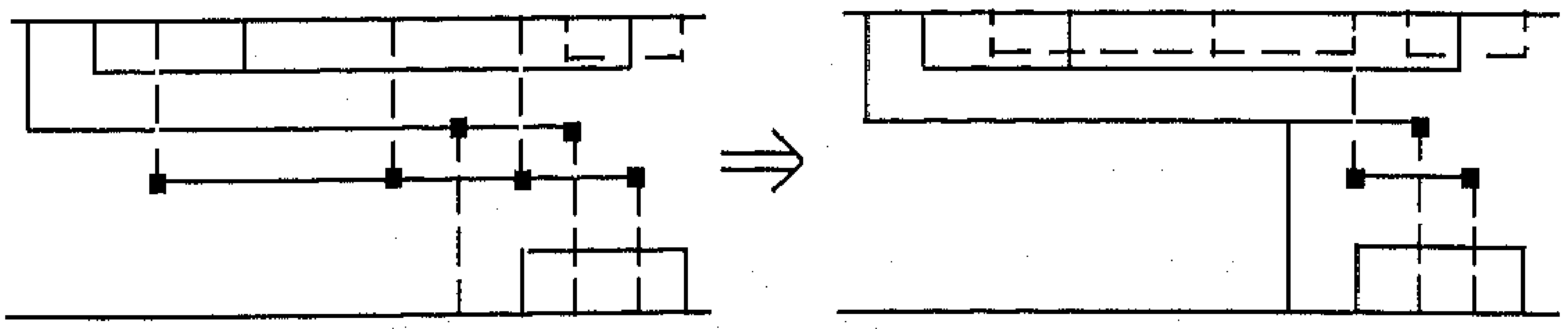


Fig. 4.5 : Illustration of Observation 3

Observation 4 : Interchanging the track assignment for nets which remove vias, might in turn require doglegging on the same layer (see Fig. 4.6).

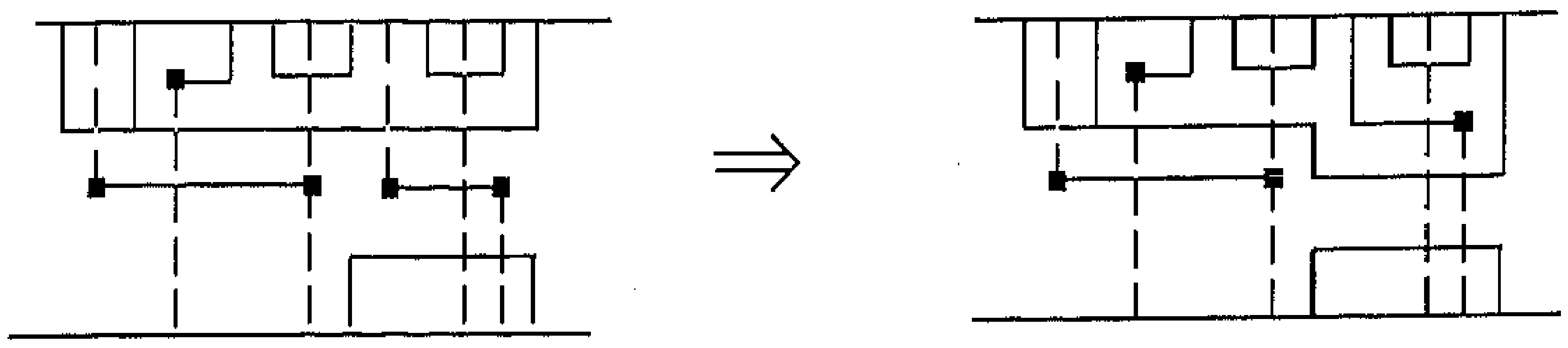


Fig. 4.6 : Illustration of Observation 4

Observation 5 : Unused portions of horizontal tracks sometimes provide handles for possible interchange of tracks for some other nets (see Fig. 4.7).

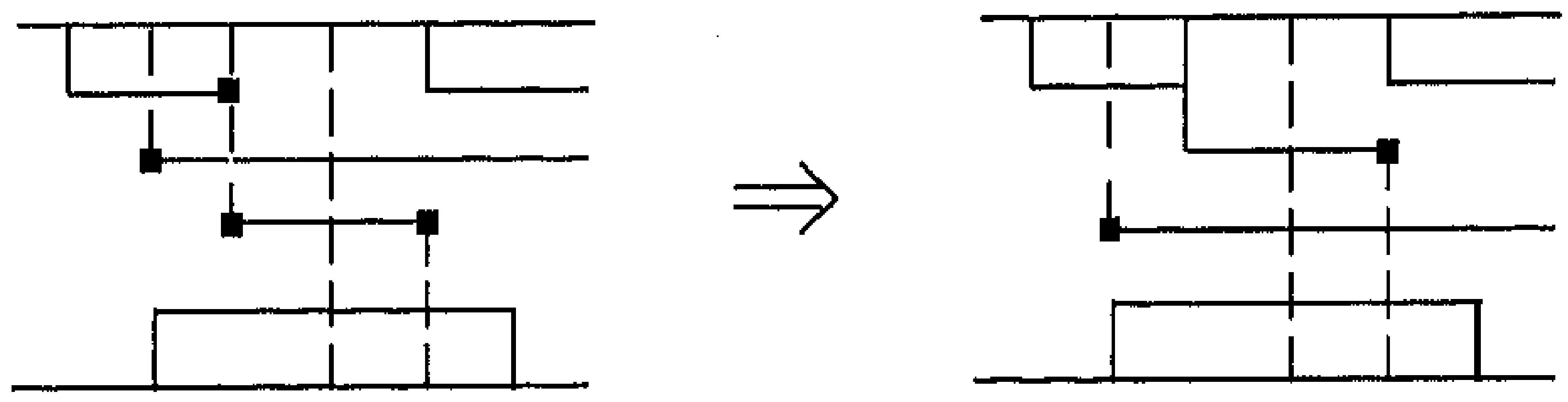


Fig. 4.7 : Illustration of Observation 5

Observation 6 : We may sometimes need shifting of vias for interchanging two tracks (see Fig. 4.8).

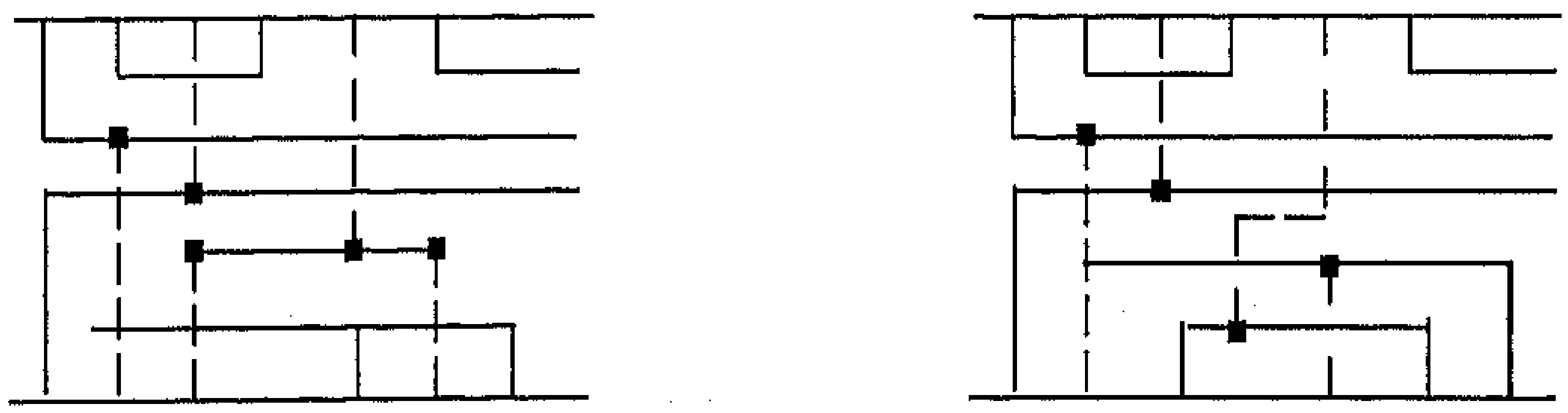


Fig. 4.8 : Shifting of vias

Observation 7 : Apart from interchanging horizontal tracks one can also interchange two vertical tracks which may contribute to via minimization (see Fig. 4.9). However, this is very hard to incorporate in the algorithm.

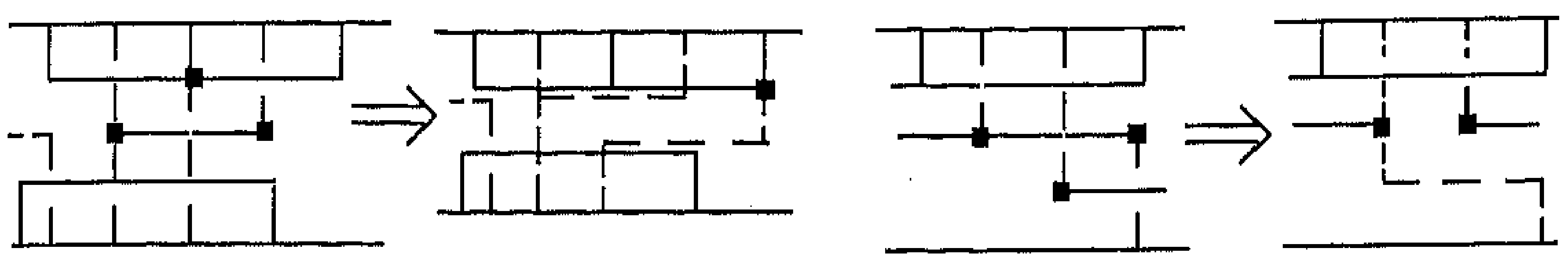


Fig. 4.9 : Interchange of vertical tracks

In the proposed layout modification approach, the algorithm scans the channel from left to right for searching possibilities of layout modification (i.e., to find the locations where via reduction is feasible by interchanging tracks). It sets a left pointer and then moves a lookahead pointer to the right across the channel where it can take a decision for modifying the layout over the portion of the channel so far scanned. The new position of the left pointer depends upon the nature of the decision and the current position of lookahead pointer. The process then continues towards right.

4.3 Scope of Modification

We identify different instances of layout structure where reduction of vias is possible by interchanging horizontal tracks. This is based on the underlying overlap graph formed by the overlap relationship of net intervals. Below we define the overlap graph and some related concepts which capture all useful information in this regard.

Definition : An interval $[a_1, a_2]$ is a *net interval* if two terminals of the same net occupy columns a_1 and a_2 on the same or different sides of the channel.

Definition : An *overlap graph* G_0 is a graph whose nodes correspond to net intervals. Two net-intervals $[a_1, a_2]$ and $[a_3, a_4]$ are said to overlap if $a_1 < a_3 < a_2 < a_4$ or $a_3 < a_1 < a_4 < a_2$. An edge between two nodes implies that the net intervals corresponding to these two nodes (i) overlap and (ii) are portions of two different nets.

Definition : Two net intervals $I_1 = [a_1, a_2]$ and $I_2 = [a_3, a_4]$ belonging to two different nets are said to be in *containment relation* if

$$I_1 \supset I_2 : a_1 < a_3 < a_4 < a_2, \quad \text{or}$$
$$I_1 \subset I_2 : a_3 < a_1 < a_2 < a_4.$$

We assume that a solution of the channel routing instance is available using the reserved layer model containing two layers. Consider the horizontal net segments, assigned to tracks in the channel, as a family of net-intervals. The y-coordinates associated with an interval is the horizontal track-id to which it is assigned. We now introduce a *visibility relation* among the net intervals.

Two net intervals α and β are said to be mutually visible if a non-degenerate rectangle R can be drawn whose two opposite sides consist of portions of α and β, such that, the horizontal span of R neither intersects nor contains any other interval [W85].

A *visibility graph* is a graph G_V whose nodes are the net-intervals. In addition, two extra nodes u, v are present in G_V which represent the top and bottom rows of the channel. An edge between nodes i and j indicates that the corresponding net intervals satisfy the visibility relation. Fig. 4.10 shows a layout and its visibility graph.

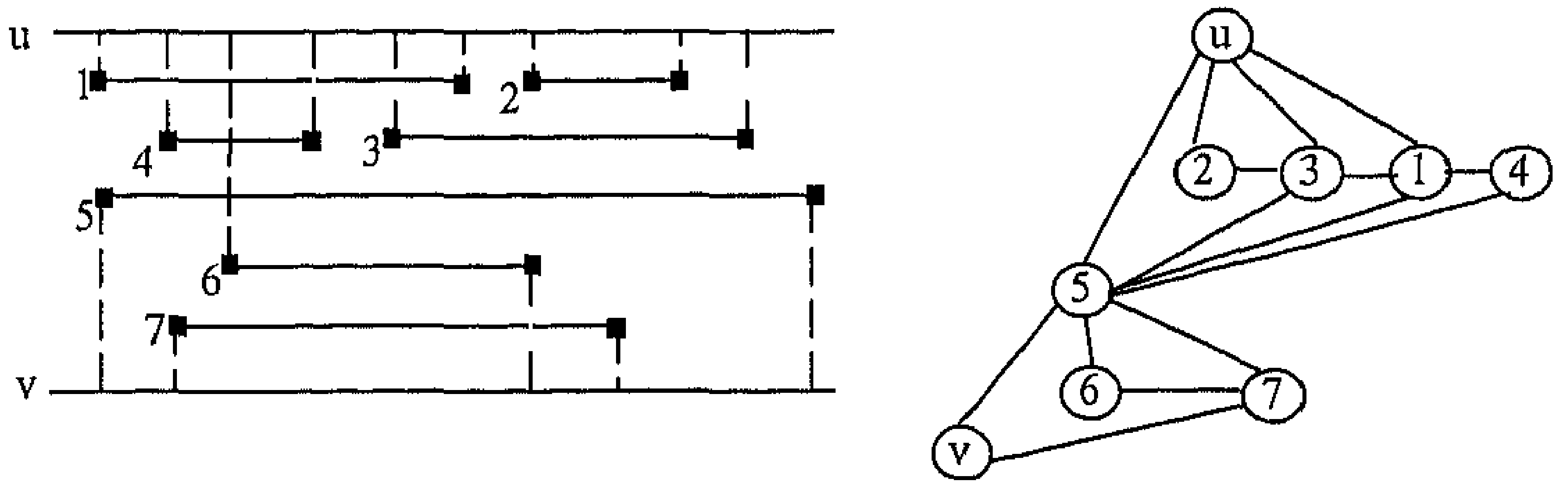


Fig. 4.10 : Channel layout and associated visibility graph

Our layout modification approach aims at reassigning net segments to the horizontal tracks such that the degrees of nodes u and v (denoting the top and the bottom of the channel respectively) in the visibility graph are maximized. It is easy to observe that such a transformation minimizes vias.

The following cases illustrate where one can minimize vias by increasing the degree of u and v in the visibility graph by rearranging the track assignments of the net segments.

Let us number the horizontal tracks of the given channel as 1 to k from the top of the channel towards its bottom side. Our ultimate objective is to reduce the number of vias without increasing the number of tracks. Suppose $I_1^t, I_2^t, \ldots, I_\alpha^t$ are the net intervals all of whose terminals belong to the top boundary of the channel and $I_\alpha^t \supset I_{\alpha-1}^t \supset \ldots \supset I_2^t \supset I_1^t$. In such a situation, our strategy would be to place I_i^t in the i-th track. If similar containment relation holds among net-intervals $I_1^b, I_2^b, \ldots, I_\beta^b$ belonging to the bottom boundary, i.e., $I_\beta^b \supset I_{\beta-1}^b \supset \ldots \supset I_2^b \supset I_1^b$, then the net-interval I_i^b will be placed in the (k-i)-th track. Note that $\alpha+\beta<k$. Now we attempt to reroute the net segments (using dogleg if necessary) according to the new assignment of net intervals using the tracks where $I_1^t, I_2^t, \ldots, I_\alpha^t, I_1^b, I_2^b, \ldots, I_\beta^b$ were originally present. A successful attempt will reduce the number of vias according to the rules listed below.

(i) If the terminals of a net-interval belong to the same side of the channel, and if there is no other terminal of that net, then we can reduce two vias by routing them in the same layer. If a net-interval I_x corresponding to a net x has more terminals to the left and/or right sides of I_x (in either top or bottom of the channel), then we cannot eliminate any via by interchanging that particular net segment without using any extra track (see Fig. 4.11a). Furthermore, if the net x has other terminals to the left and/or to the right of I_x, but the terminals of I_x belong to two different sides of the channel, then via reduction is not possible by track interchange (see Fig. 4.11b).

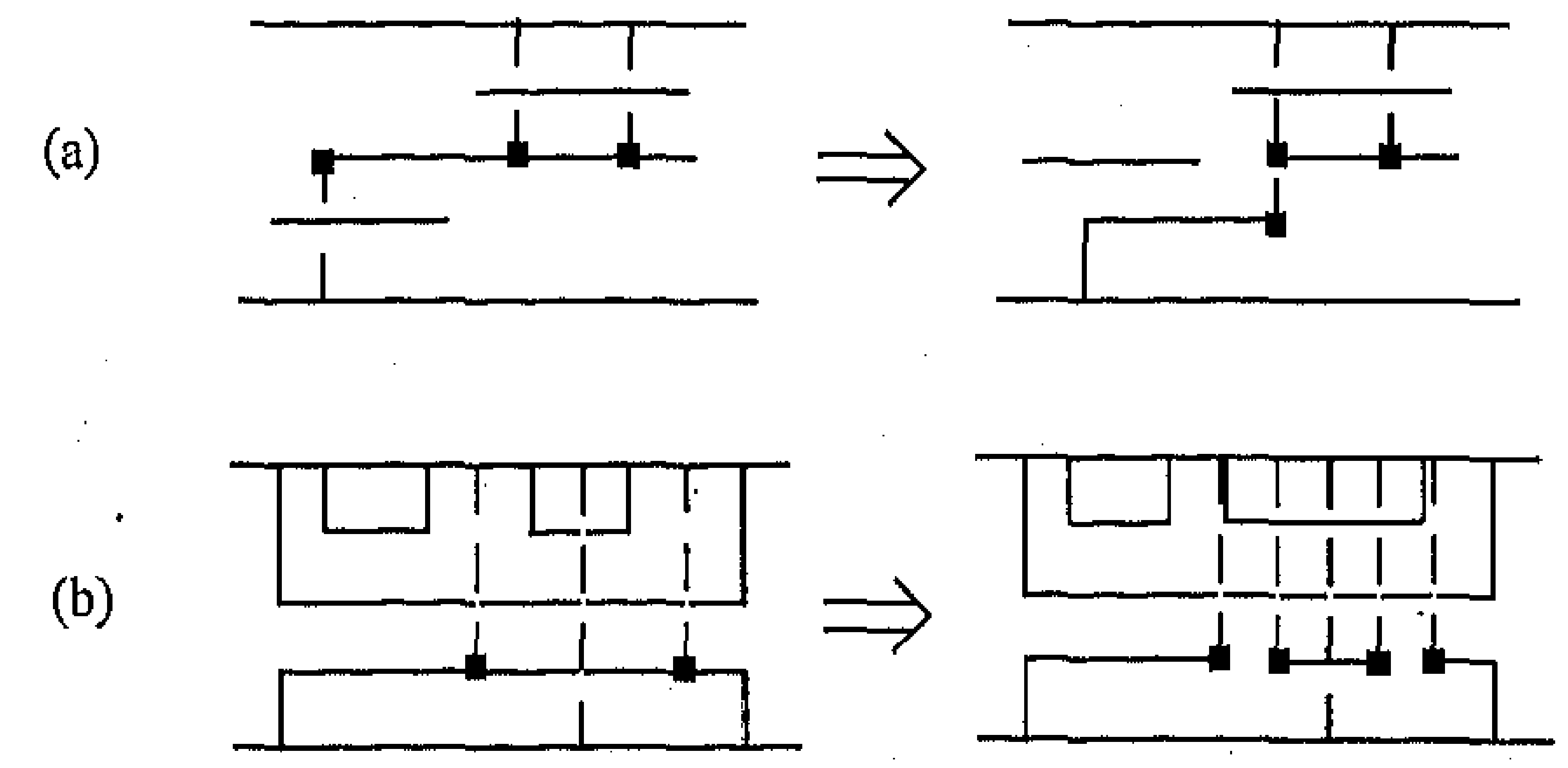


Fig. 4.11 : Situations where via reduction is not possible by track interchange

(ii) If the two terminals of the net-interval I_x lie on the same boundary of the channel and the remaining terminals (if they exist) of the same net are either to the left or to the right of I_x, then we can reduce one via as shown in Fig. 4.12.

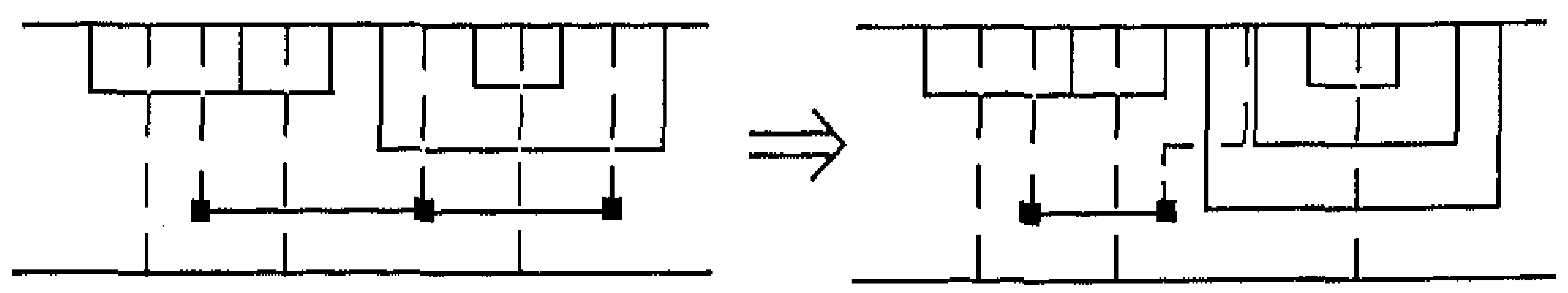


Fig. 4.12: Illustration of via reduction by rule (ii)

(iii) If the terminals of the net-interval I_x belong to the two different boundaries of the channel, but there exists no other terminal of the net x, then one via can easily be reduced (see Fig. 4.13).

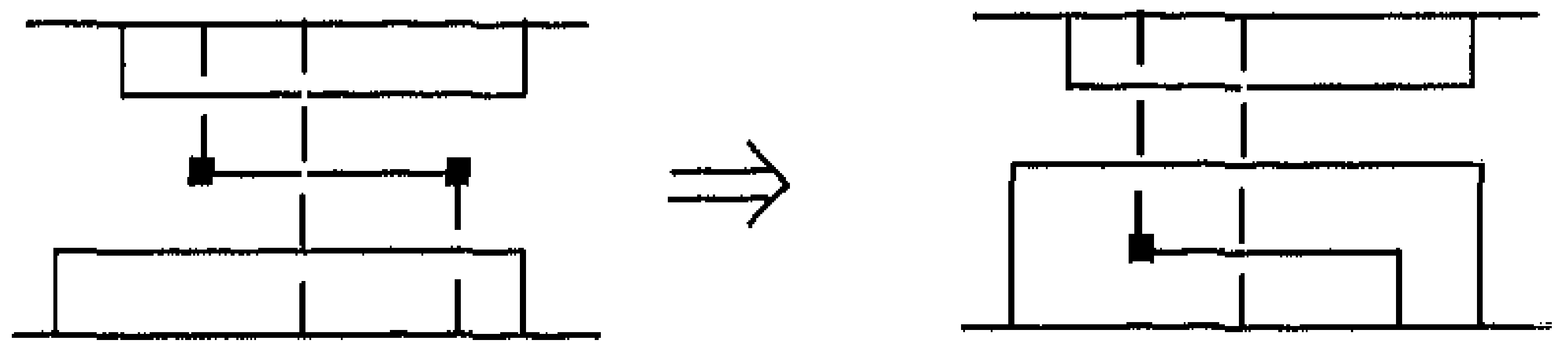


Fig. 4.13 : Illustration of rule (iii)

To capture all these instances systematically, consider the overlap graph of net-intervals. Let $I_1, I_2, \ldots, I_n$ be a set of net intervals forming a clique in the overlap graph (of net intervals) and the left boundaries of all of them appear on the top (bottom) side of the channel. Also assume that the nets attached to $I_1, I_2, \ldots, I_n$ are all two terminal nets. Now if $a_1 < a_2 < a_3 \ldots < a_n$ then we may eliminate atleast one via from each net by placing the net interval I_i in the i-th ((k-i)th) track. Such a transformation tends to increase the degree of u and v in the visibility graph, leading to via minimization.

We now present a simple algorithm for reducing vias incorporating above observations. We sweep a vertical line from left to right. Initially, the sweep line stands at the leftmost column of the channel. Let $I_i\ [a_i, b_i]$, $i=1,2,\ldots,q$ be a set of net intervals to the right of the current position of the sweep line such that all the a_i's appear on the same side (top or bottom) of the channel and for all i, $a_i < \min(b_1, b_2, \ldots, b_q)$. For these net segments we apply the above rules for rerouting. Suppose, *this causes r extra vias and eliminates s existing vias*. Then we define the net savings (gain) = s - r. If the gain is positive, the above rerouting is worthwhile. Next we shift the sweep line to the column position $\min_{i=1}^{q} (b_i)$, and the process continues.

4.4 Via Minimization Algorithm

The algorithm explores the possibility of eliminating vias in the given layout. If the search is successful, then the layout is modified according to the suggested rerouting. The algorithm also checks whether the existing vias can be eliminated using an alternative route for the nets. The search for an alternate path to connect two net segments, or terminals of the same net, is accomplished by a MAZE router [L61].

Algorithm : Via minimization

```
Input :   A channel layout in two layer Manhattan model.
Output :  Modified layout in the unreserved layer model with reduction in vias
          and without increasing the number of tracks.

Step 1 : Set the left-pointer and the look-ahead pointer on the first column.

Step 2 : While left-pointer <> n do
     2.1  Initialize lookahead pointer to left pointer.
     2.2  Move the lookahead pointer to select a set of net intervals
          $I_i\ [a_i, b_i]$, $i = 1, 2, ..., q$ such that (i) both $a_i$ and $b_i$ are on the
          same side of the channel, (ii) $a_i > a_{i-1}$, (iii) $a_i < \min(b_1, b_2, ..., b_q)$,
          and (iv) $a_{q+1} > \min(b_1, b_2, ..., b_q)$.
          /* Note that all the nets corresponding to the net intervals $I_1, I_2, ...I_q$
          must be different. */
     2.3  Apply observation 2 for possible track interchange for reducing the
          number of vias.
     2.4  If a net interval is defined by terminals on both top and bottom side
          of a channel, then we may apply observation 3 for choosing a portion
          of a net interval which satisfies the condition stated in step 2.2.
     2.5  Advance the left-pointer to $p = \min_{i=1}^{q} (b_i)$ where q is the number
          of intervals selected by the look-ahead pointer.
     2.6  Set the look-ahead pointer at the left-pointer.

Step 3 :  /* Layout modification */
```

```
        /*   We assume that for each track the assigned net segments is available.
             They may overlap since for some nets the tracks are assigned in
             Step 2, and for those nets which are not considered in Step 2,
             will assume their initial track assignments in the original layout. */

        For each track do

    3.1 If there is no overlap among the assigned net segments in this track then
        route those net segments.

    3.2 If an overlap is observed, then apply observation 4 and 5 to for track
        reassignment for the affected net segments. This may cause dogleg in
        the routing layout.

Step 4: For each via v in the layout do

        /* Assume that via v is belongs to net i. */
    4.1 eliminate via v.

    4.2 Find components of net i separated by via v.

    4.3 /* Find an alternate path reconnecting the two components of net i
        appearing in two different layers */

        4.3.1 Use maze router to find alternate paths to reconnect the two
              components in layer 1 and layer 2 separately.

        4.3.2 Compare the two paths found in step 4.3.1 and the original path
              involving the via v, in terms of the number of vias eliminated
              and wire length.

        4.3.3 Based on the decision in step 4.3.2, select the better one.

        4.3.4 If any of the alternate paths found in step 4.3.1 is chosen in
              step 4.3.3, then the original path is eliminated from the layout.
```

Complexity analysis

The time complexity for the above algorithm in the worst case is $O(vg + nc)$ where v is the number of vias, g is the number of grid points, n is the number of columns and c is the number of net terminals. The complexity of the algorithm follows from the fact that in each pass of Step 2, one requires $O(c)$ time for searching the possible arrangement of net intervals. Step 3 requires $O(g)$ times. In step 4, the maze routing algorithm requires $O(g)$ time in total to check whether each via quantifies for favourable elimination.

4.5 Experimental Results

We have implemented the algorithm along with a maze router [L61]. Table-4.1 compares the performance of our method with the results obtained in [TWC91] and [NMN87] on standard benchmarks (YK_i denotes the i-th example taken from [YK82]). The famous Deutsch's difficult example (De) is also studied. Experimental results show that in all these examples, our solutions consistently outperform the previous results with respect to the number of vias used. For example, in the 19-track solution of the Deutsch's difficult example [D85], our result shows 36% reduction in the number of vias, with respect to the solution of [D85], 24% improvement over the optimal CVM algorithm [NMN87], and 15% improvement compared to the result given in [TWC91]. As far as our knowledge goes, the proposed method produces the fewest number of vias with 19-tracks for the Deutsch's difficult problem ever reported in the literature.

Furthermore, this technique does not introduce unnecessary long wire segments. In most cases, the total length of wires used in the layout is reduced compared to the result in [TWC91]. Results for the selected benchmarks are shown in Table-4.2.

Table 4.1: Performance analysis with respect to via minimization

Example	# of horizontal tracks	Number of vias			
		Original solution [source]	Optimal CVM algorithm [NMN87]	algorithm in [TWC91]	our method
YK_{3a}	15	91[YK82]	72	66	44 (Fig. 4.14)
YK_{3b}	17	107[YK82]	91	78	54 (Fig. 4.15)
YK_{3c}	18	125[YK82]	109	103	66 (Fig. 4.16)
YK_5	20	150[YK82]	114	105	73 (Fig. 4.17)
De	19	301[D85]	252	226	190 (Fig. 4.18)

Table 4.2 : Wire length comparison on standard benchmarks

Example	Total wire-length		
	Original solution [source]	algorithm in [TWC91]	our method
YK_{3a}	1365 [YK82]	1362	1281
YK_{3b}	1651 [YK82]	1653	1543
YK_{3c}	2100 [YK82]	2100	2055
YK_5	2785 [YK82]	2801	2665
De	4989 [D85]	5058	5093

4.6 Conclusion

We have presented a new algorithm for via minimization in a two-layer channel routing environment using layout modification technique. The method is based on interchange of horizontal tracks and does not incorporate additional horizontal or vertical tracks. A significant reduction of vias for standard benchmark examples is observed. The total wire-lengths are also found to be very competitive.

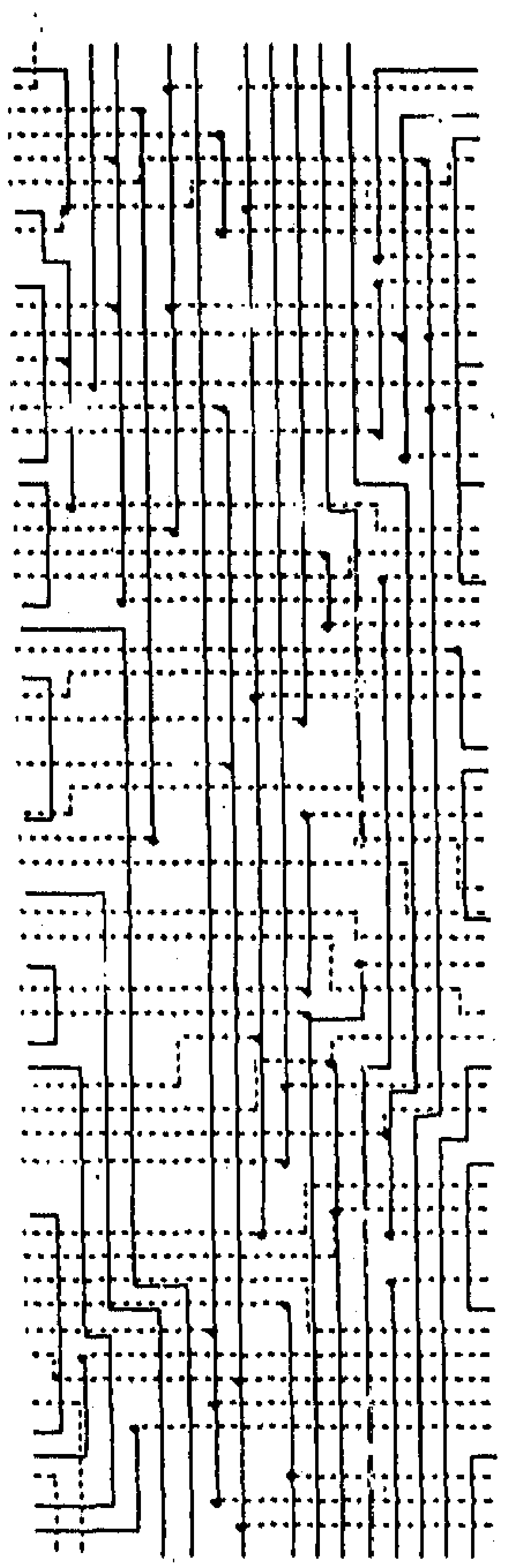


Fig. 4.15 : Example YK_{3b}

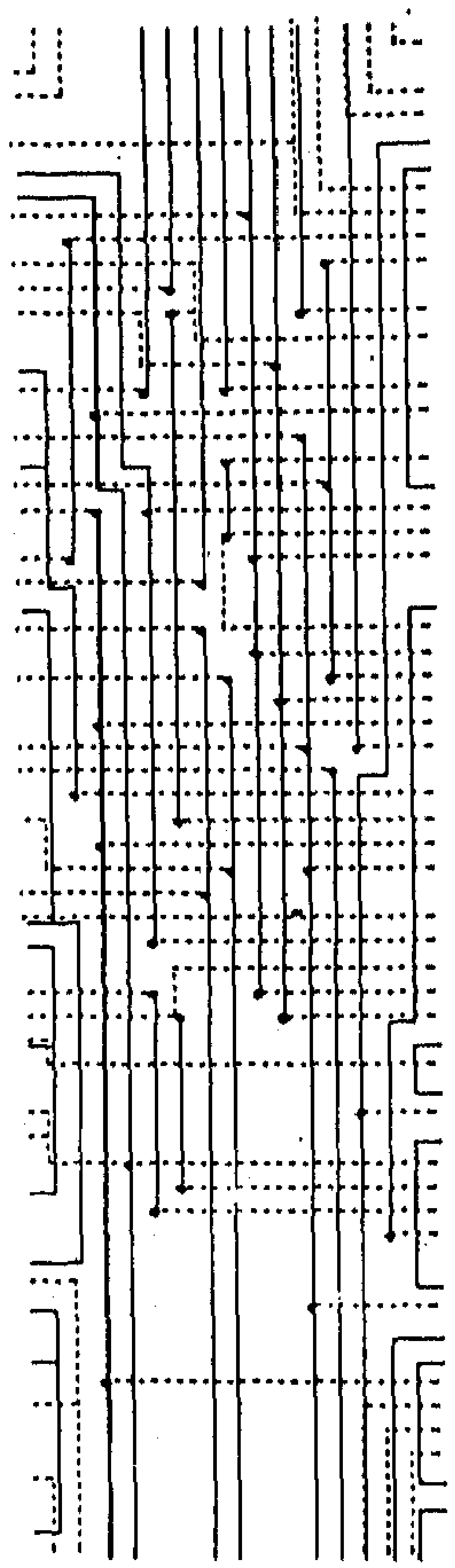


Fig. 4.14 : Example YK_{3a}

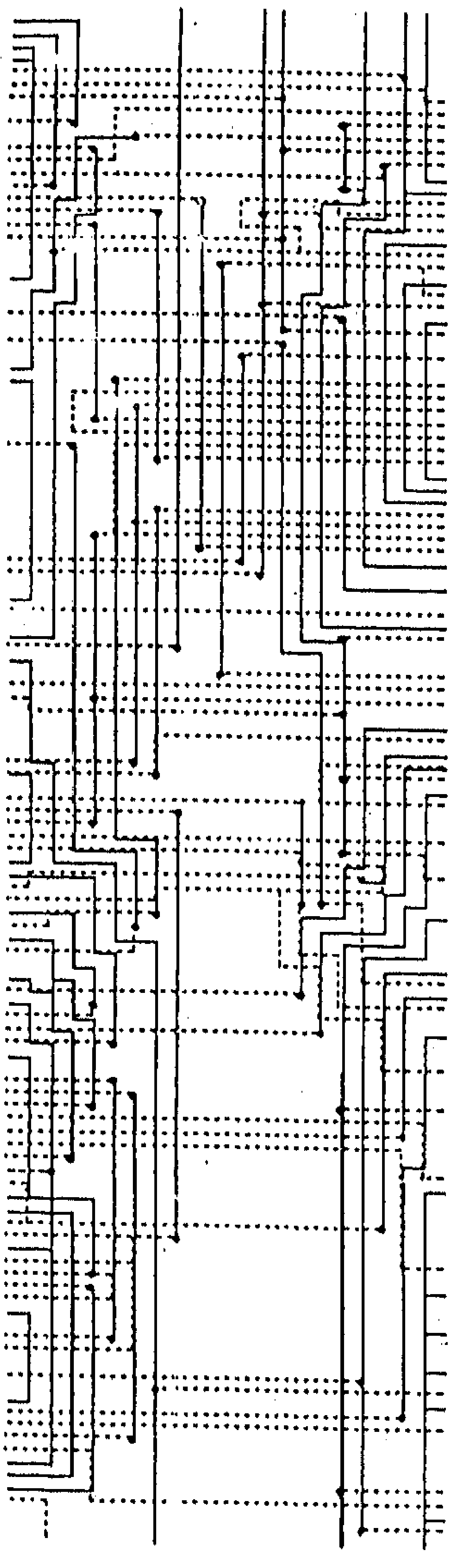


Fig. 4.16 : Example YK_{3c}

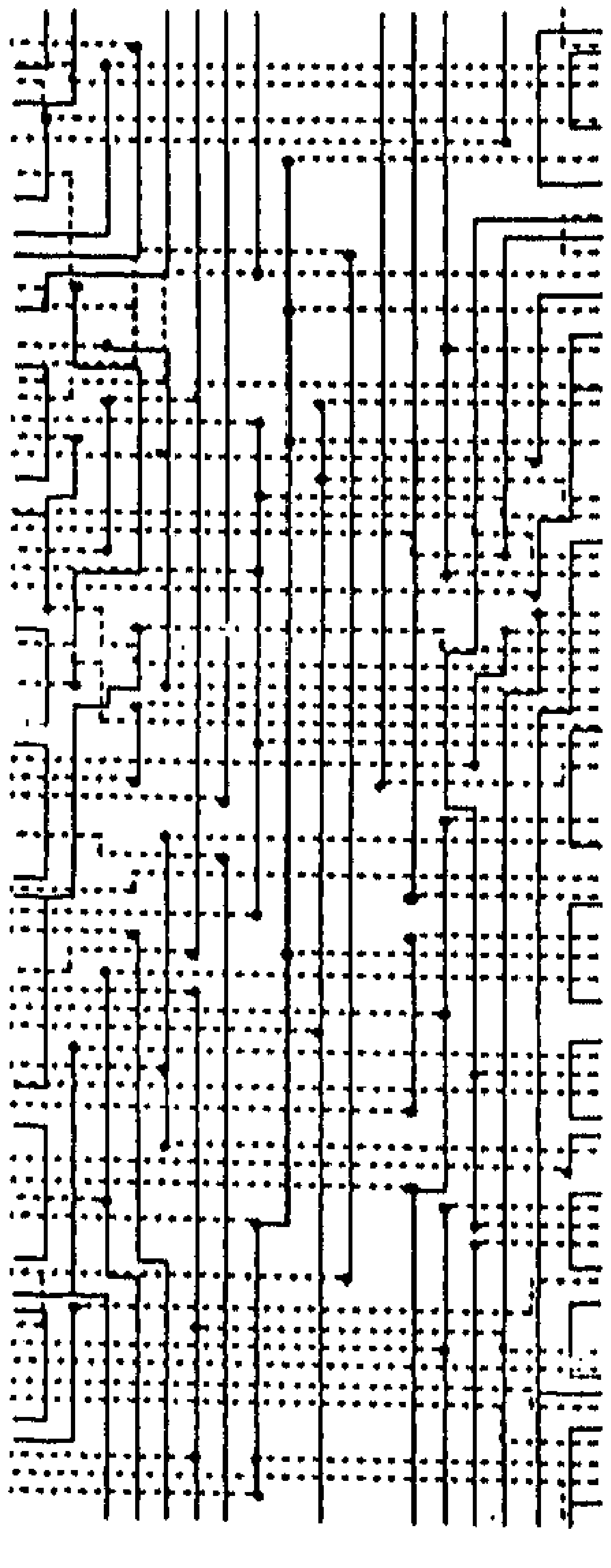


Fig. 4.17 : Example YK_5

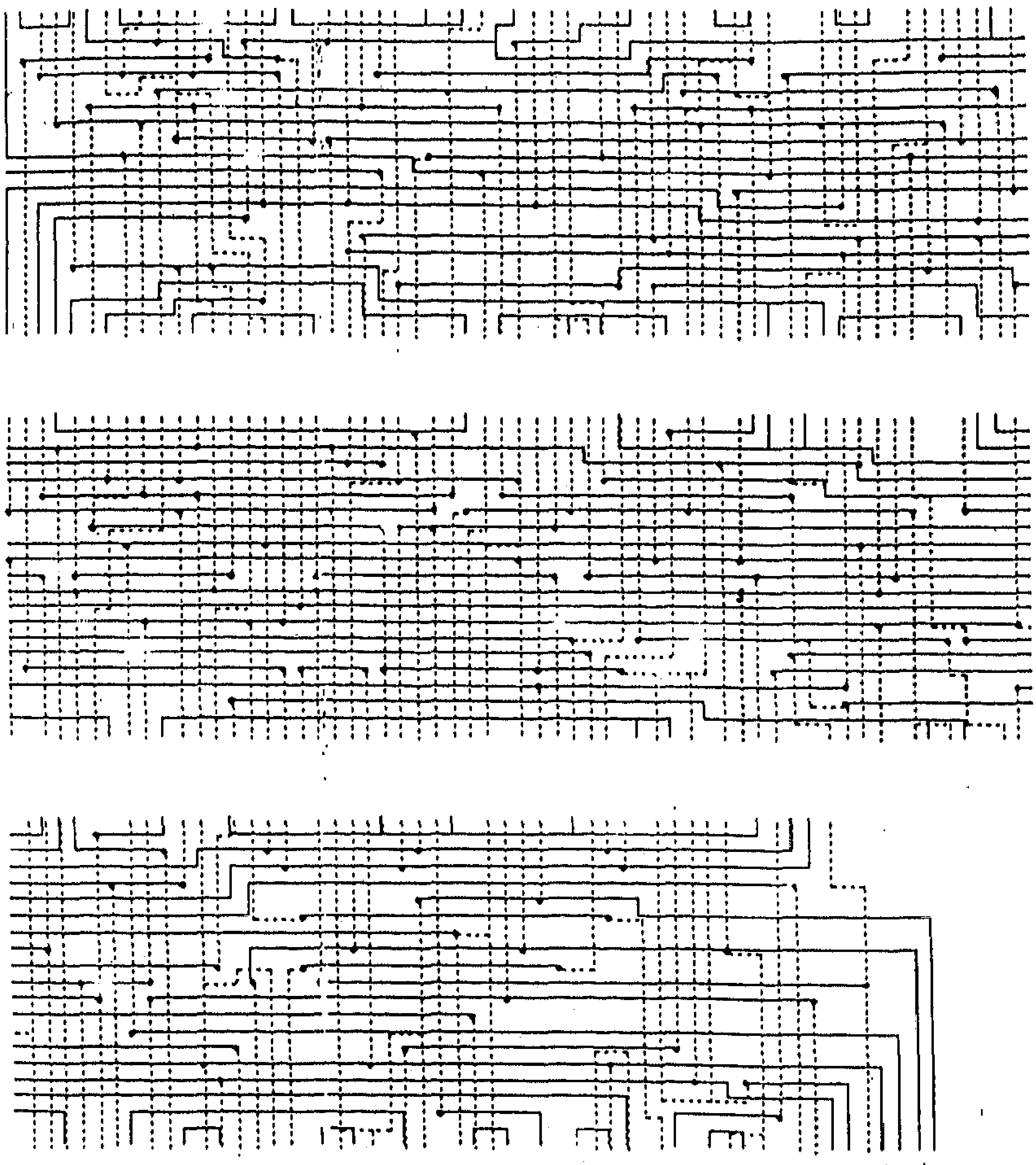


Fig. 4.18 : Deutsch's difficult example

Chapter 5

Channel Routing using Manhattan-Diagonal Model

5.1 Introduction

Channel routing is a complex and indispensable part of VLSI design cycle. Given a net list in a channel, the primary goal of a router is to complete the interconnection, and minimize channel area, number of vias, or wire length [S93]. As the problem is NP-complete [L80, S85a, SB80, PPDP95], a significant amount of computation time is spent on optimizing channel routing. Various models of channel routing exist depending on the number of available layers, routing mode (Manhattan/ non-Manhattan), layer assignments (reserved/ unreserved), and terminal behavior (fixed, movable, or interchangeable) [S93]. The channel routing problem has been studied extensively in the past two decades based on the Manhattan [YK82, L80, S85a, SB80, PPD$^+$95, PDP$^+$95, BBD$^+$86, CWL88, BP83, HIZ91, RVS85, RF82] and the knock-knee [MPS86] model. Most of the models assume either horizontal or vertical wire segments for interconnection. Recently, some effort has been made to develop algorithms in non-Manhattan geometries. A diagonal model with $\pm 45^0$ wires was used in [LLP89a, LLP89b, LLS91]. Routing in time square mode using $\pm 60^0$ wires was also considered [LLP90]. A channel router based on sorting using both rectilinear and diagonal wire segments ($\pm 45^0$) was proposed in [CR91]. Recently different modified version of the non-Manhattan channel routing algorithm based on bubble-sorting technique were proposed in [CHS94, Y99]. It has also been observed that diagonal model can be supported by the fabrication process; it also provides better utilization of routing space, tends to reduce wire length and hence area and delay. However, the existing algorithms based on this model are either too elementary or inefficient, and therefore, have limited applicability to real-life routing problems. Design rules for Manhattan-diagonal model can be supported by CMOS fabrication process as shown in the 5.1. The problem of satisfying minimum separation between a via and an adjacent 45^0 wire can be solved by using octagonal vias wherever necessary. Henceforth, a square (octagonal) via will be represented by a black square (dot) in all subsequent figures.

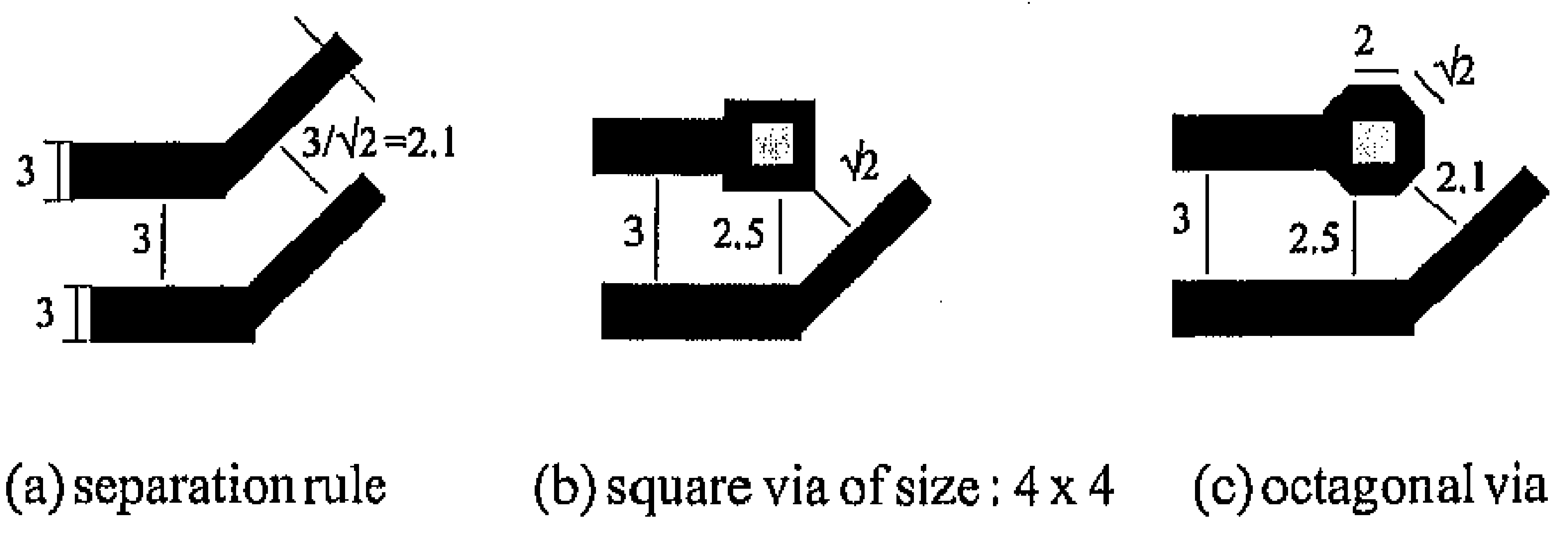


Fig. 5.1 Design rules (in λ) for Manhattan-diagonal geometry

5.2 Main Results

This chapter outlines a new approach to channel routing, that combines the advantage of both diagonal model and Manhattan geometry. We consider *two* routing layers with *fixed terminals,* and the *Manhattan-diagonal ($\pm 45^o$)* model. Thus, wire segments may either be rectilinear or at $\pm 45^0$ directions. First, we consider the *reserved layer* model (i.e., horizontal and diagonal wire segments in one layer and vertical wire segments in other layer), and present a simple $O(l.d)$ time algorithm that routes an arbitrary channel if *no cyclic vertical constraints* in w tracks, where l is the length of the channel i.e., the number of equidistant column in the channel; d is the channel density, and $d \leq w \leq (d+1)$. Next, we describe an output-sensitive algorithm that can route general channels with *cyclic vertical constraints* using w tracks, in $O(l.w)$ time (i.e., linear in area of the channel), assuming *unreserved layer MD model* where horizontal, vertical and diagonal wire segments are in any layer that allows overlapping of wiring segments in two layers. The proposed algorithms outperform those in [YK82, PPD+95, HIZ91, CR91] both in time complexity and quality of solutions. Experiments with benchmark examples show very encouraging results. The router outputs an 18-track solution for the Deutsch's difficult example [D85], a 2-track solution for Burstein's difficult channel [BP83], a 15-track solution for *cycle.tough* [HI89] without inserting any extra row or column. The router also produces solutions with low via count and reduced wire length compared to the via minimizer of [TWC91]. To the best of our knowledge, routing of these benchmarks in the MD model is being reported for the first time. Further applications of such routing strategy to L-shaped and staircase channels, as well as switchboxes, are reported in the next chapter.

5.3 Preliminaries

A rectangular channel consists of l equidistant columns (length), and two rows of terminals along its top and bottom sides. A number i , $0 \leq i \leq n$ is assigned to each terminal. Terminals with the same number i, $i \geq 1$ are to be connected by a net i, while those labeled with 0 represent unconnected terminals. The objective of the classical channel routing problem is to connect all the terminals specified by a given net list with minimum number of tracks w (channel width), thus minimizing channel area. Reducing via count and/or wire length is also important for high-performance layout design.

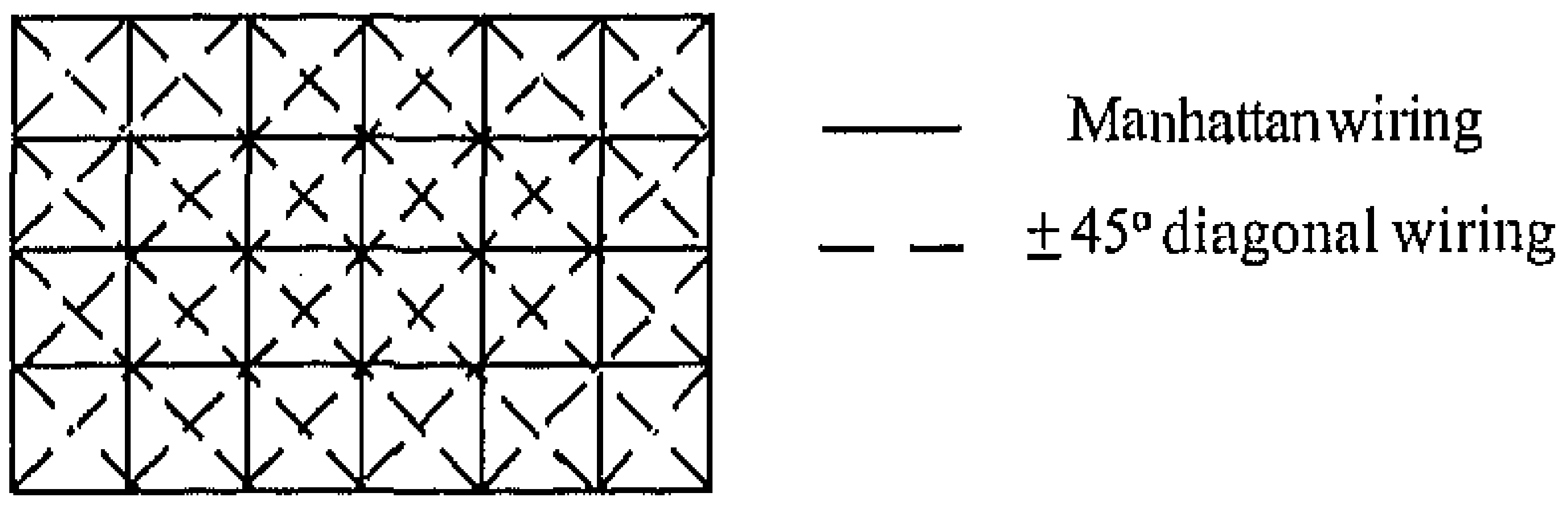


Fig. 5.2 : Grid structure in the Manhattan-diagonal (MD) model

In the MD-model, each layer of a channel C of length l and width w is assumed to consist of (l x w) grid points (Fig. 5.2) formed by isothetic equidistant grid lines separated at unit distance determined by the design rule. In addition, the diagonal tracks connect all pairs of grid points separated by a distance of $\sqrt{2}$. Thus, the degree of an interior node of the grid structure becomes 8. Terminals appear on the first and the last row and vias must be placed on grid points.

Example

Fig. 5.3 shows a typical example [YK82] of channel routing in unreserved layer MD model that uses only 3 tracks, whereas d = 5.

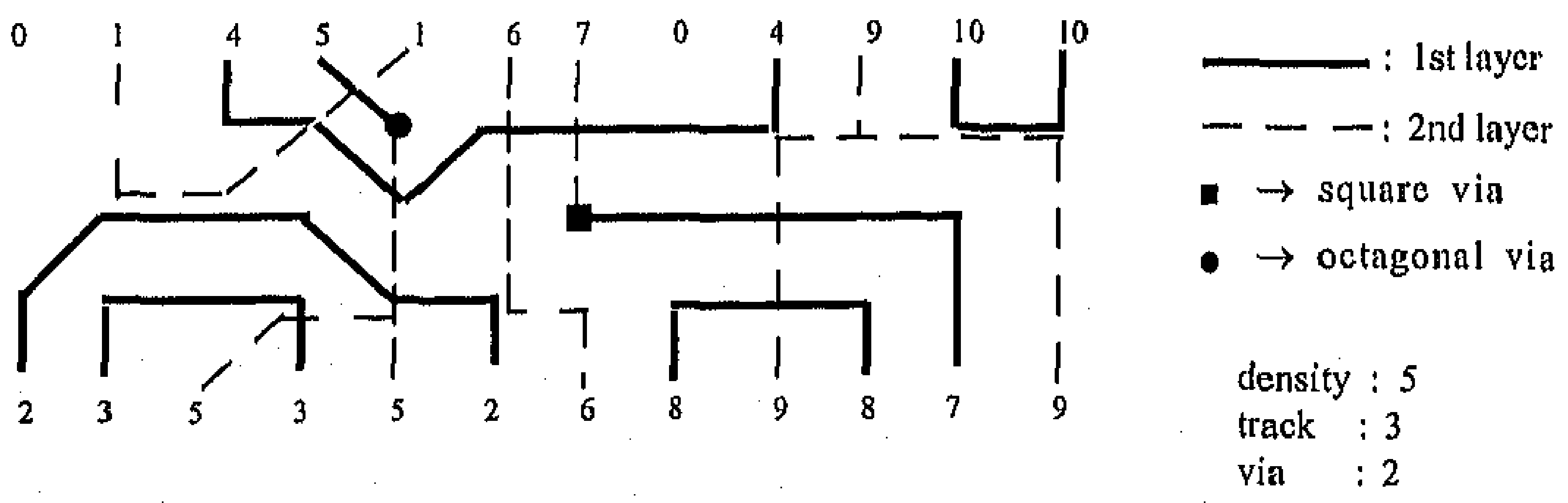
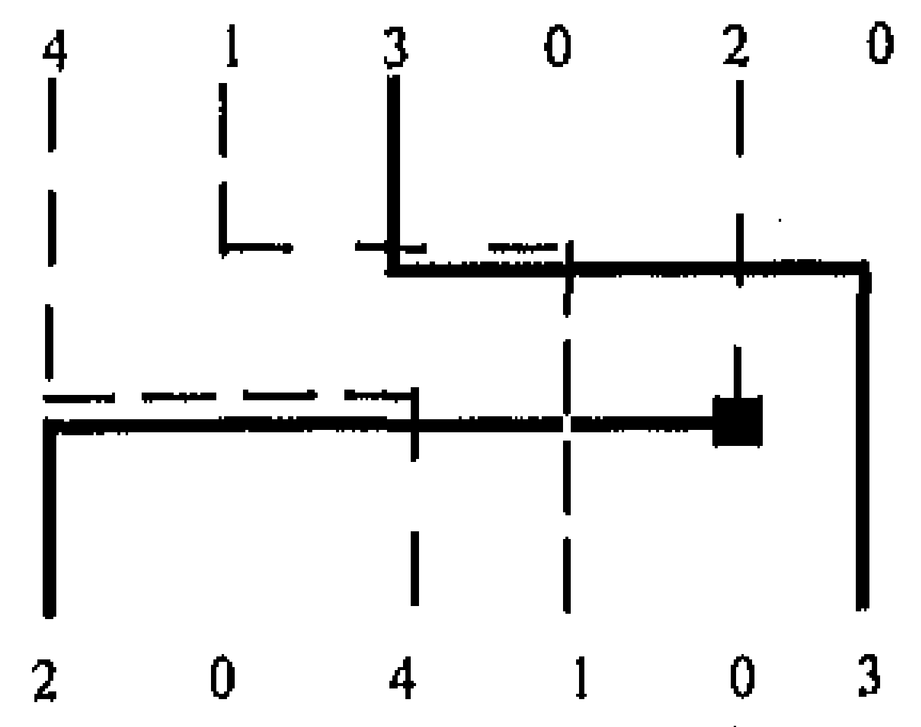


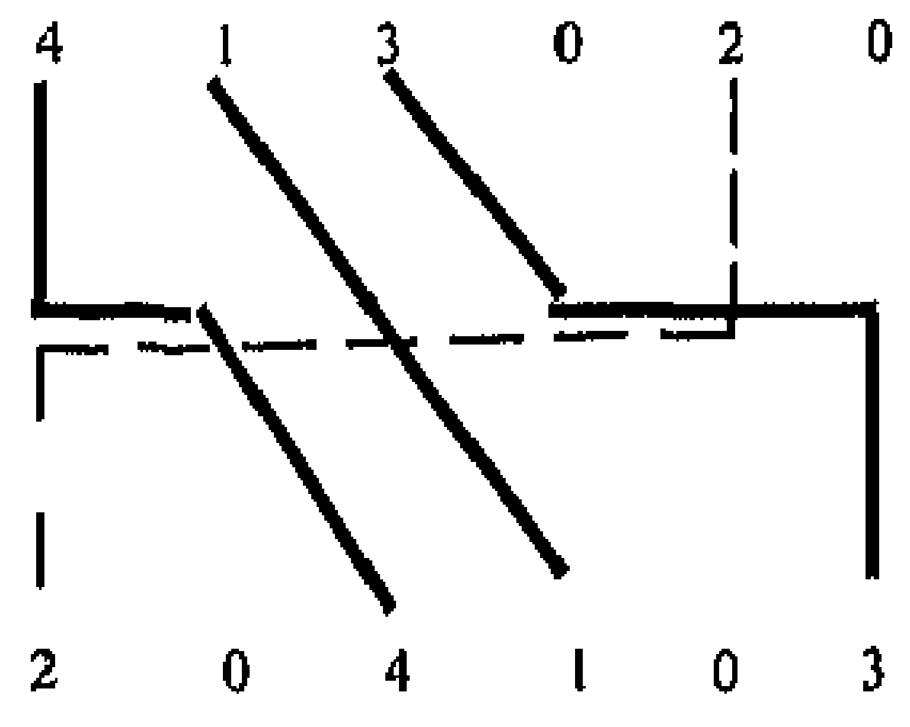
Fig. 5.3 : Channel routing in unreserved layer MD model

Lower bounds on the number of tracks

In reserved layer Manhattan model, the minimum number of tracks $w \geq d$, which trivially follows from the [YK82]. For unreserved layer Manhattan model, $w \geq \lceil d/2 \rceil$ [S93]. The bound further improves in the unreserved layer MD model, as observed below. Fig. 5.4 shows comparative routing of an example.



(a) unreserved layer mamhattan channel routing model; $w = 2$; $d = 4$



(b) MD-model ; $w = 1$, $d = 4$
$\lceil (d - 2)/2 \rceil$ is the new lower bound on w

Fig. 5.4 : Illustration of lower bounds

Observation : In unreserved layer MD-model, $w \geq \lceil (d\text{-}2)/2 \rceil$.

Proof : As two different nets cannot pass through the same grid point of a given layer, at most $2w$ nets can be accommodated in a column in the Manhattan model. In MD model, two extra nets may be routed through the same column (Fig. 5.4b) using the diagonal edges from the top and bottom terminals. So at most $2w+2$ nets can pass through the column. Hence $w \geq \lceil (d\text{-}2)/2 \rceil$. ∎

5.4 Channel Routing in Reserved Layer MD Model

5.4.1 Formulation

If the VCG is acyclic, then it is possible to route any arbitrary channel using a simpler MD model, with reserved layer strategy. We assume that all horizontal and ±45 wiring segments lie in one layer, and all vertical segments lie in the other layer. Thus no two wire segments in two different layers overlap.

Theorem : Any channel with an acyclic VCG is routable in w tracks in reserved layer MD model where, $d \le w \le d+1$.

Proof : Since all horizontal tracks lie in one layer, $w \ge d$. The rest of the proof follows from the following facts.

We sort the acyclic VCG topologically, and construct an ordered sequence of nets $S = \{\alpha_1, \alpha_2, \ldots\ldots, \alpha_n\}$ such that if the net α_i precedes net a_j in the VCG, then $j > i$. We now show that routing can be accomplished in at most $(d+1)$ tracks such that the set of nets passing through in each column ordered from the topmost track to the bottom, is a subsequence of S.

Suppose routing is being done in the channel from left-to-right in a column-by-column fashion. Since $w \ge d$, we assume, at least d tracks are already laid out in the channel. Consider a column c_i where t tracks are already occupied (Fig. 5.5). The ordering of nets occupying the tracks from top to bottom conforms with the set S. Tracks that are not assigned to any net in a column are called *free* in that column. Now, need to consider the following two cases.

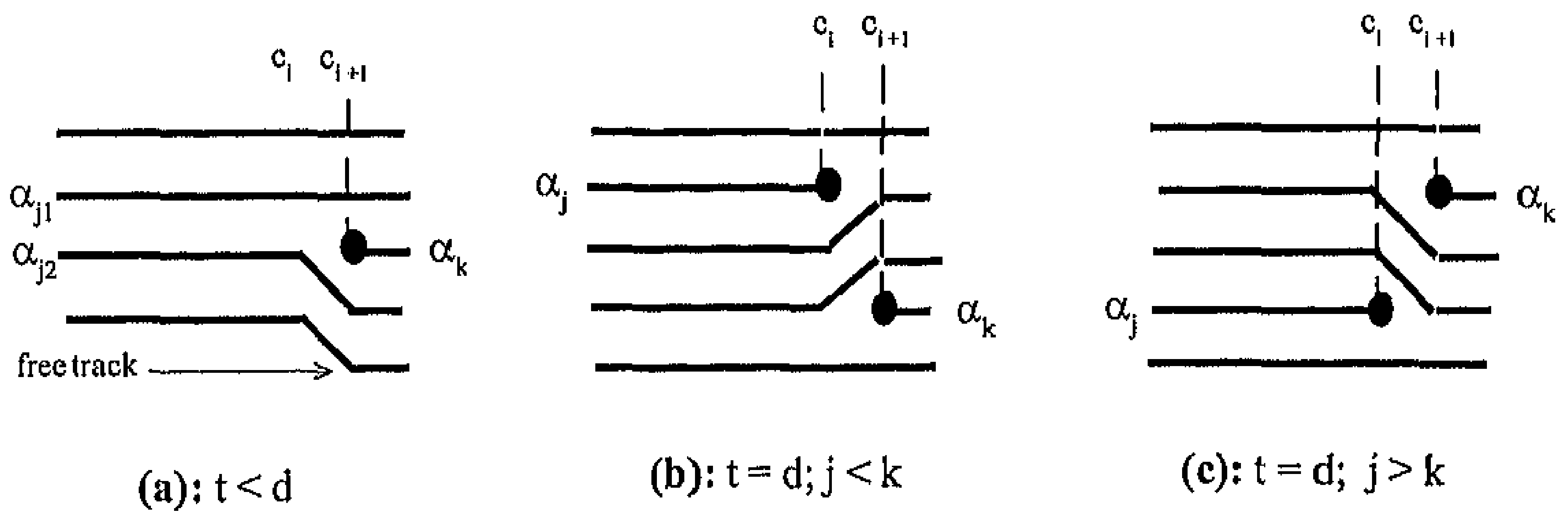


Fig. 5.5 : Transfer of nets onto adjacent tracks via diagonal edges

Case 1 : Only one new net α_k begins (i.e., its leftmost terminal appears) at the next column c_{i+1}.

Case 1.1: $t < d$

Assume that α_k appears in between α_{j1} and α_{j2} ($j1 < j2$) in topological order. Then α_k can be accommodated using diagonal wiring within the interval (c_i, c_{i+1}) and a *free* track at c_{i+1} as shown in Fig. 5.5a. The sign of the diagonal (i.e., + 45° or -45°) depends on the relative position of the free track and the net α_k. Thus, column c_{i+1} needs at most d tracks.

Case 1.2 : $t = d$

Since t cannot exceed d in any column, at least one net, say α_j (occupying track t_j) terminates at column c_i to make room for the incoming net. If k > j in net ordering (Fig. 5.5b), then in column c_{i+1}, the net α_p is assigned to track t_{p-1} for all p, $j < p \leq k$. For other nets, track assignments are kept unchanged. Horizontal segments of the same net occupying adjacent tracks are then connected using +45° diagonal wiring. If k < j, we need to use - 45° diagonals (Fig. 5.5c). Thus, routing can be completed up to column c_{i+1} in d tracks preserving the net ordering in S.

Case 2 : *Two* new nets begin at column c_{i+1}.

Let the two nets that begin from column c_{i+1} be α_{k1} and α_{k2}, where k1 < k2 with respect to the topological order of nets.

Case 2.1 : Assume that at most d -1 tracks are occupied in the previous column c_i, i.e., at least one track is free; in addition, at least one net, say α_{j1} must terminate in column c_i. Using diagonal wiring from appropriate tracks in columns c_i and c_{i+1}, one can accommodate two incoming nets using at most d tracks as shown in Fig. 5.6.

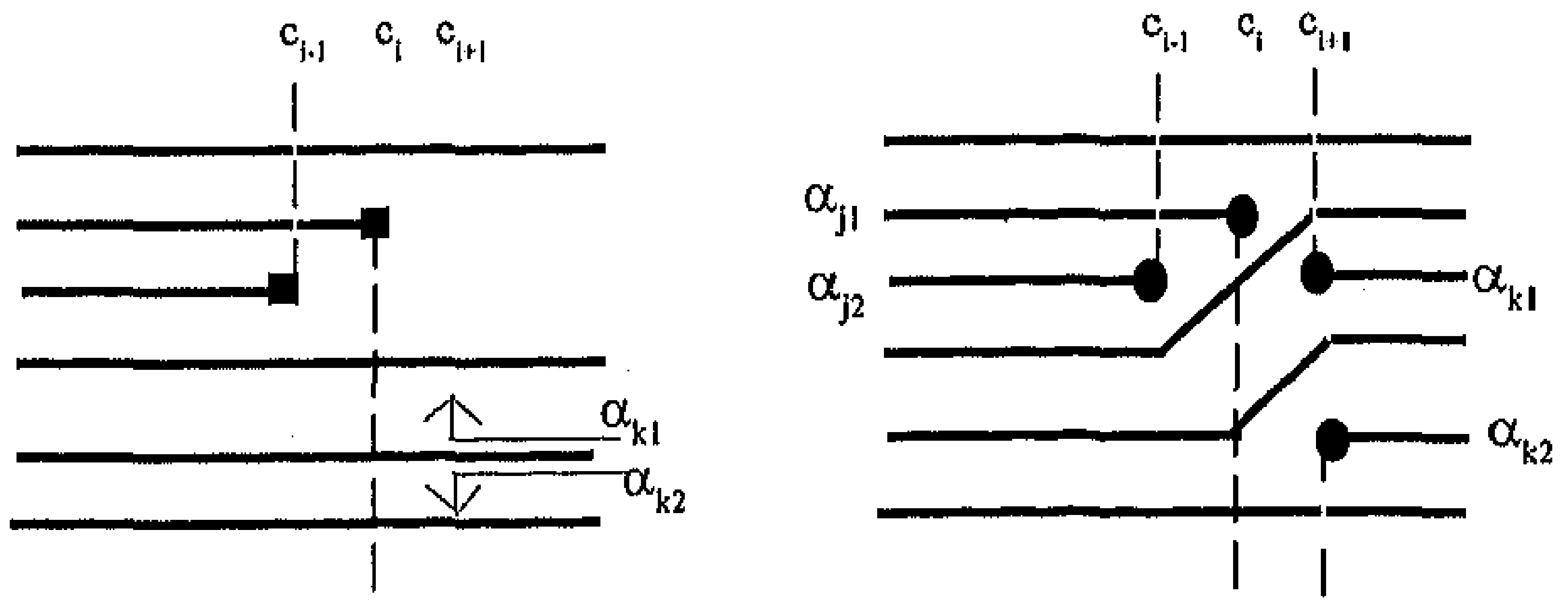


Fig. 5.6 : Accommodation of two incoming nets without inserting a track

Case 2.2: Assume that d tracks are occupied in the previous column c_i ; two nets, say α_{j1} and α_{j2} (j1 < j2) must therefore terminate in column c_i . Without loss of generality assume that, j1 < k1.

Case 2.2.1 : (i) j1 < k1 < j2 < k2 (see Fig. 5.6a);
(ii) j1 < k1 < k2 < j2 (see Fig. 5.6b);

It is easy to see that the concerned portion of the channel can be partitioned horizontally into two parts each of which encounters only one new net. From the results observed in Case 1, it follows that routing can progress from column c_i to c_{i+1} using only d tracks (Fig. 5.7). Case 2.2.1(ii) is similar.

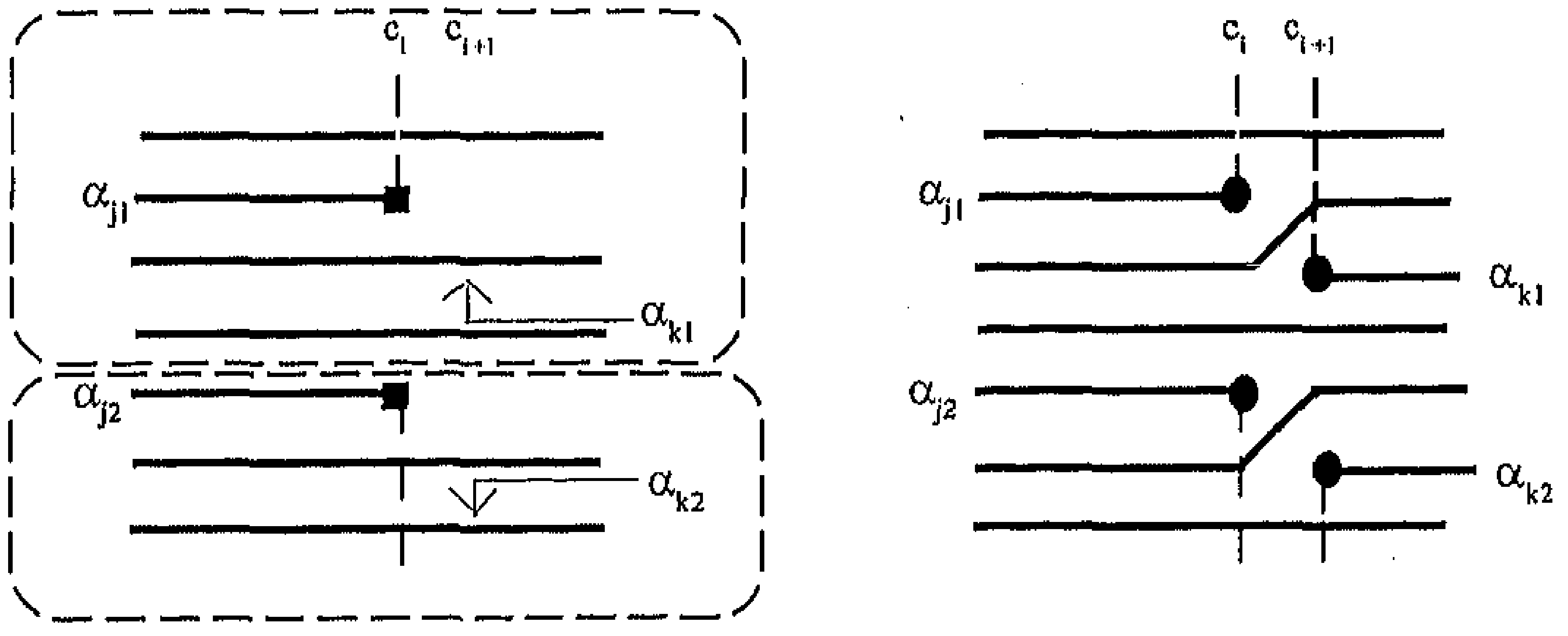


Fig. 5.7 : Accommodation of two incoming nets in d tracks

Case 2.2.2 : j1 < j2 < k1 < k2 ;

In Fig. 5.8, it is demonstrated that only one extra track is required to continue routing. Thus (d+1) tracks are necessary and sufficient in column c_{i+1}. Since the congestion of a column cannot exceed d, one track must be free in column c_{i+1}. Hence, by virtue of Case 1 and Case 2.1, for all subsequent columns, no more than (d+1) tracks are required.

Since the VCG is acyclic, the order of nets remains invariant throughout the length of the channel. Thus, the progressive routing strategy, discussed above, will be successful using w tracks where, $d \leq w \leq (d+1)$. ■

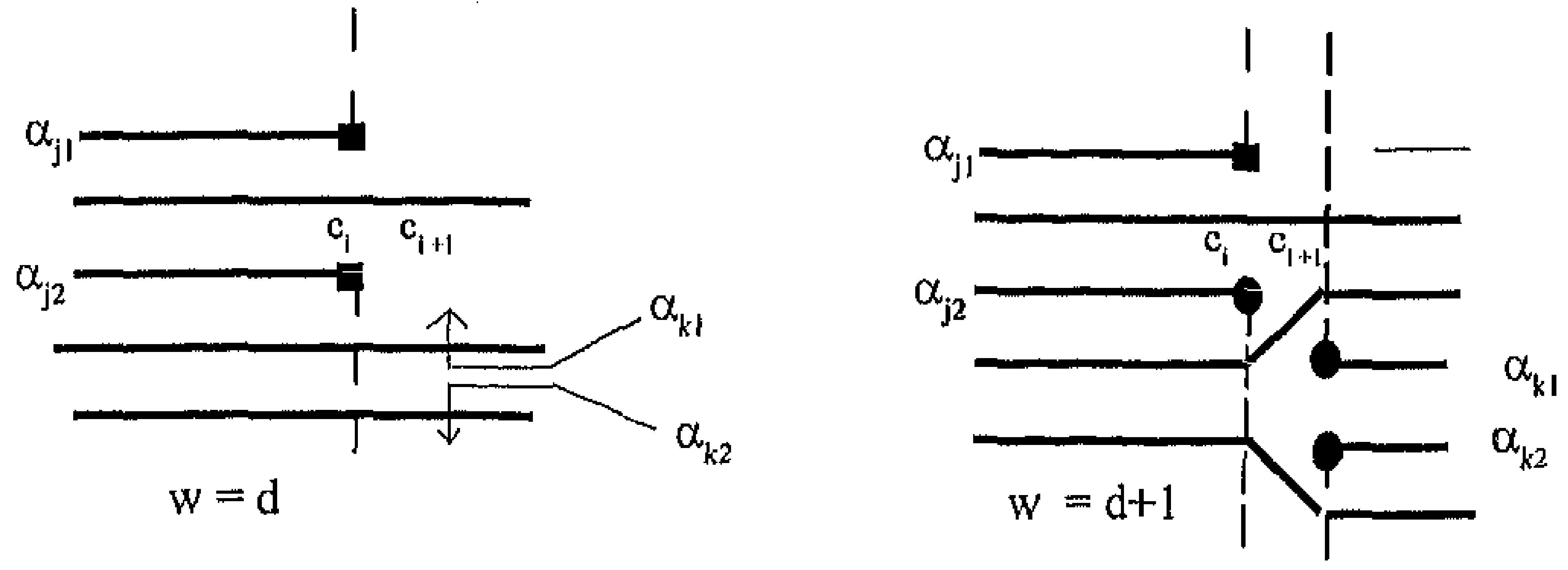


Fig. 5.8 : An extra track is required to continue routing

Example : Fig. 5.9 shows routing of an interesting example in reserved layer MD model.

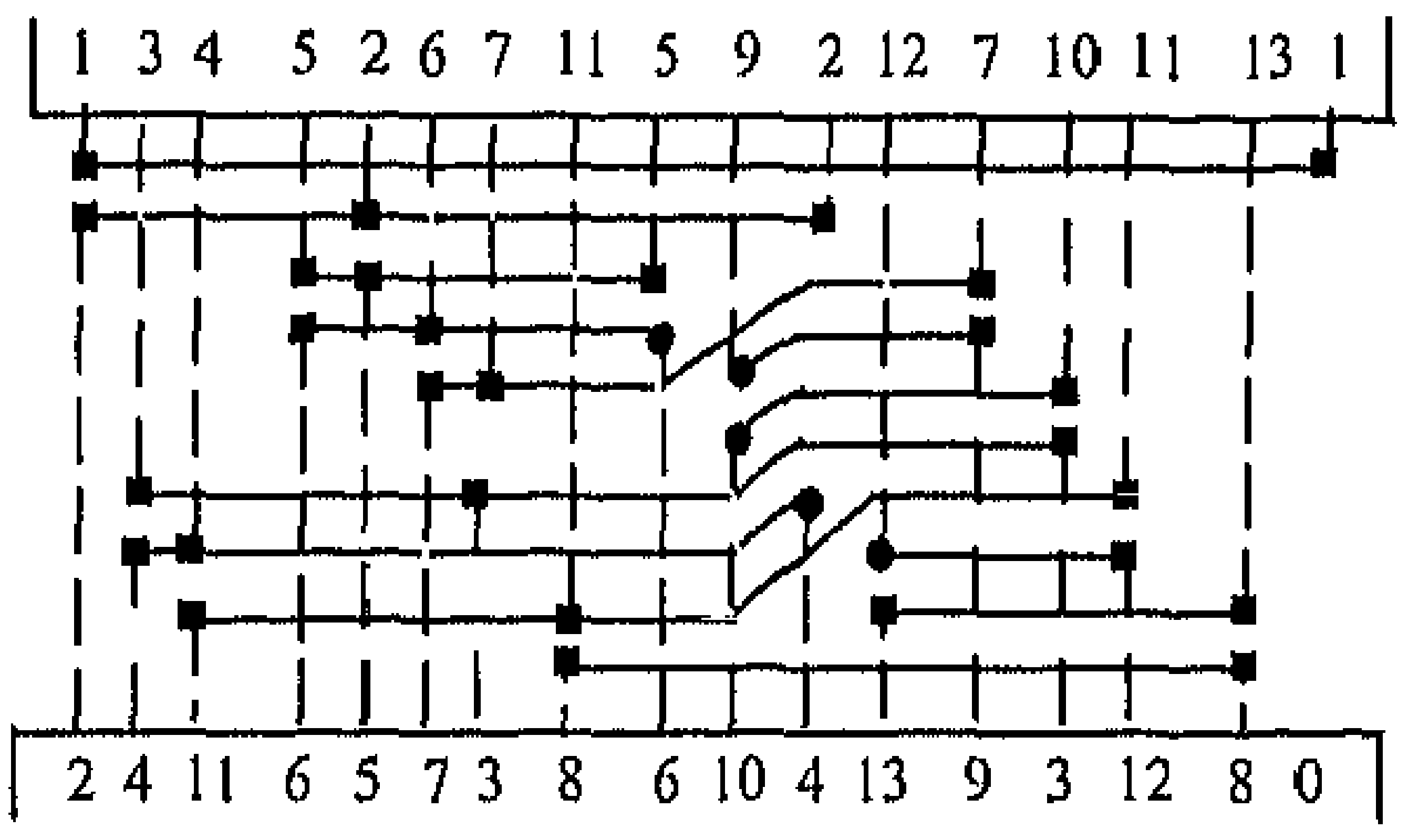


Fig. 5.9 : An example where, density $d = 9$, length of the longest chain in VCG = 13, *and the number of tracks* w = 10

Remark : The width of the channel increases from d to d+1 only in the situation described in Case 2.2.2. However, if the diagonal wire segments are allowed to occupy both the layers, then it may be possible to route in d tracks only as shown in Fig. 5.10.

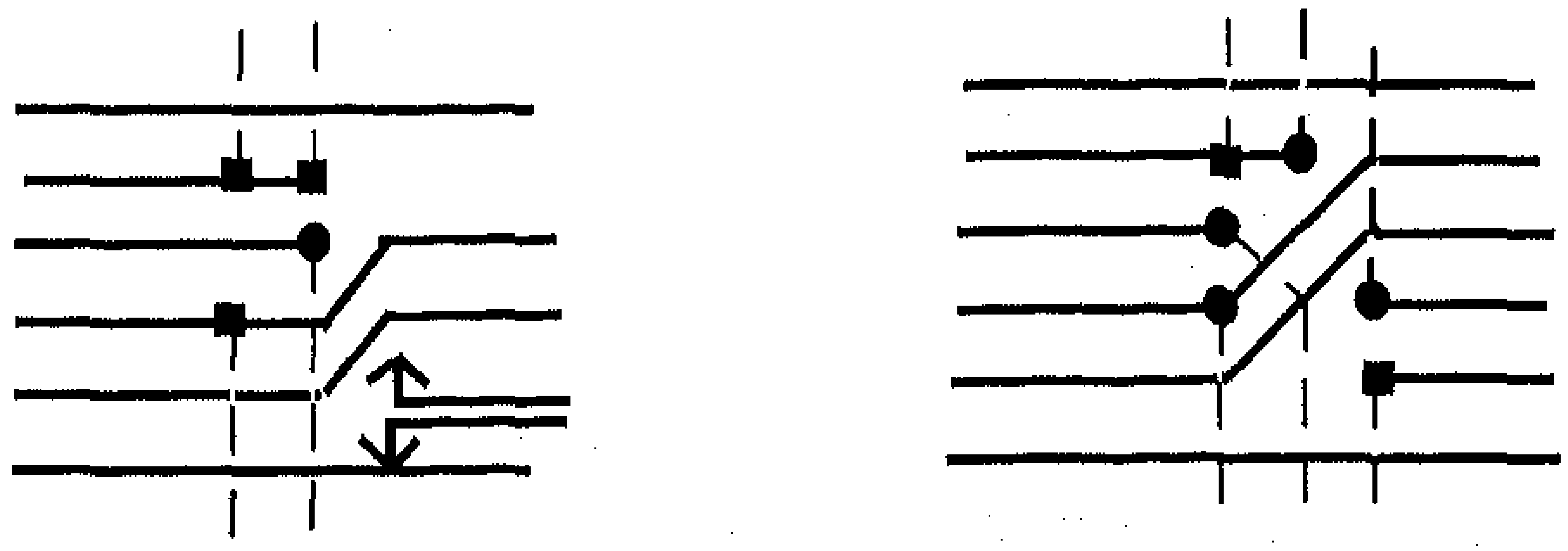


Fig. 5.10 : (a) An example of Case 2.2.2 where d+1 tracks are necessary
(b) Observed improvements if layer reservation for diagonals is relaxed

5.4.2 Algorithm: 1 (Reserved layer MD router)

Input : A channel C of length l, a netlist of n nets (with acyclic VCG), and two routing layers;

Output : Reserved layer MD routing (horizontal and $\pm 45^0$ connections in one layer, and vertical connections in the other layer), in at most $(d+1)$ tracks.

```
Step 1  (a): Topologically sort the nodes of the acyclic VCG and construct an
             ordered sequence of nets $\{\alpha_1, \alpha_2, \ldots\ldots, \alpha_n\}$. Determine the density
             $d$ of C ;
               /* Let $c_1, c_2, \ldots\ldots, c_l$ denote the columns ordered from left to right */

        (b): freetrack $(c_0) = d$;                  /*Make $d$ free tracks in C */

  Repeat Steps 2 & 3 until all the columns are scanned from left to right;

Step 2: (a)  if left- or rightmost terminal of any net does not appear in column
               $c_i$, then continue track assignment as in column $c_{i-1}$;

        (b)  if r nets (r =1 or 2) terminate in column $c_i$ then mark the free tracks;
               freetrack$(c_i)$ := freetrack$(c_{i-1})$ + r;

        (c)  if # new-net$(c_i)$ = 1, then          /* Here freetrack$(c_{i-1}) \geq 1$ */
               locate the track (t) that fits the incoming net satisfying the topo-
                  logical order;
               make the track free by using diagonal connections from the grid
                  points in column $c_{i-1}$ as in the Case 1, and
               assign the track for the new net in column $c_i$;
               freetrack$(c_i)$ := freetrack$(c_{i-1})$ -1;

        (d)  if # new-net$(c_i)$ = 2, then
               if freetrack$(c_{i-2}) \geq 1$, then
                  assign tracks for new nets and put the diagonals as in Case
                     2.1;
                  freetrack$(c_i)$ := freetrack$(c_{i-1})$ -2;

               else if freetrack$(c_{i-2})$ = 0,
                                    /* two nets say, $\alpha_{j1}$ and $\alpha_{j2}$ (j1 < j2) must
                                  terminate in column $c_{i-1}$, and let nets $\alpha_{k1}$ and
                                          $\alpha_{k2}$ (k1 < k2) begin in column $c_i$*/
```

```
if (j1 < k1 < j2) or (k1 < j1 < k2), then
    assign tracks and diagonals as in Case 2.2.1;
    freetrack($c_i$) := freetrack($c_{i-1}$) -2;

else insert an extra track. Make width $d$+1 and follow
    Case 2.2.2 for assigning tracks and diagonals for the
    nets appearing in the current column $c_i$;
    freetrack($c_i$) := freetrack($c_{i-1}$) -1;
    /* w = $d$+1 is necessary in this circumstance */
```

Step 3. Complete vertical connections and place vias appropriately.

Complexity : Step 1 can be performed in $O(l)$ time. Step 2 requires $O(l.d)$ time over l iterations. Vertical connections and placement of vias in Step 3 take $O(l)$ time. Thus the overall complexity of the algorithm is $O(l.d)$, i.e., *linear in the area* of the channel. The router needs at least d tracks, and it outputs no more than d+1 tracks. The space complexity is also $O(l.d)$.

5.4.3 Experimental results

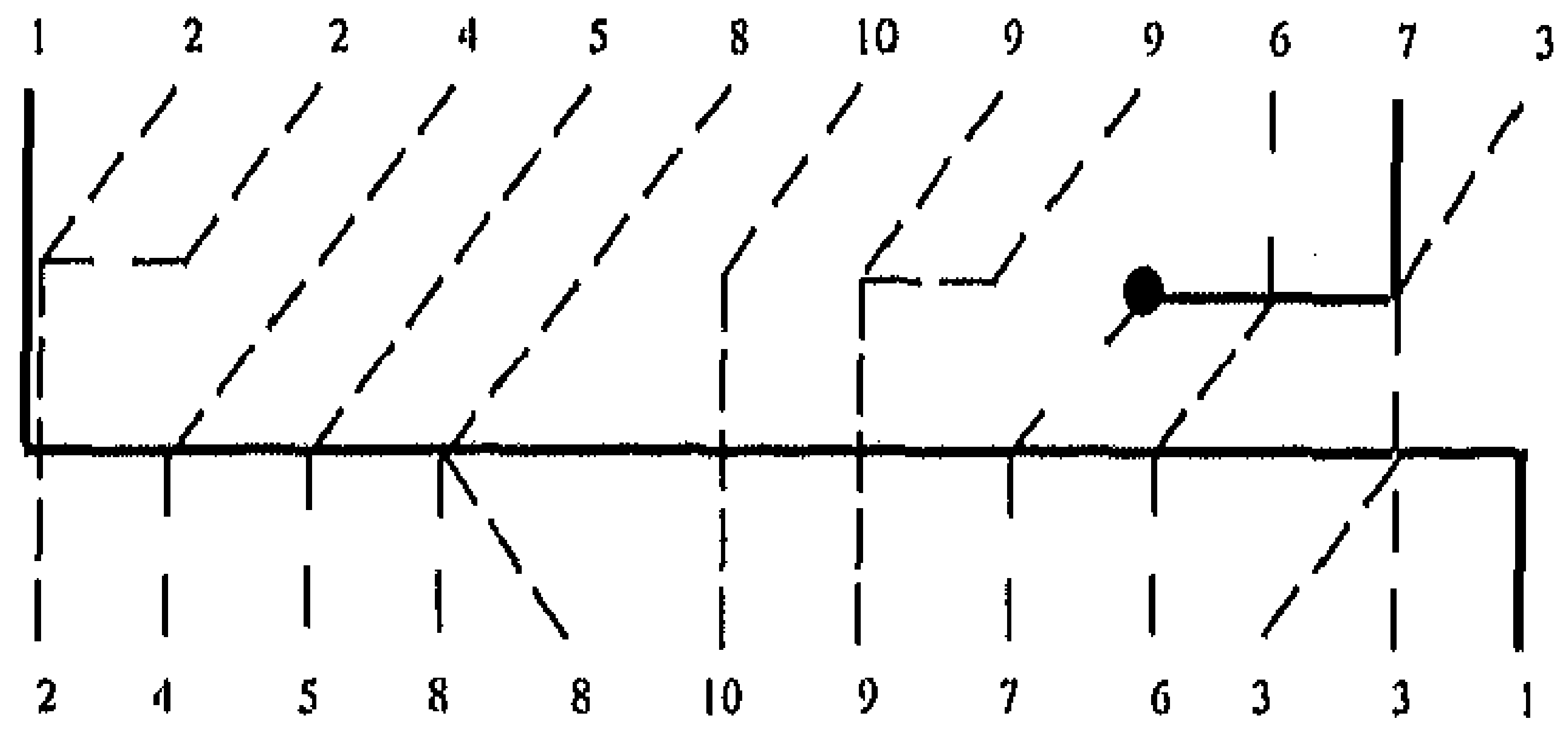
We have run our algorithm on some benchmarks (where YK_i denotes the i-th example of [YK82]) and Deutsch difficult example (DDE) [D85], whose VCG's are acyclic. The algorithm terminates very fast and outputs routing with a channel width (w) equal to density (d) in each of these cases. Fig. 5.9 shows a contrived example (Ex) where the number of tracks required is d +1. Experimental results are given in Table 5.1.

Table 5.1 : Results on benchmarks with acyclic VCG in reserved layer MD model

Example	density (d)	number of tracks (w)
YK_{3a}	15	15
YK_{3b}	17	17
YK_{3c}	18	18
DDE	19	19
Ex (Fig. 5.9)	9	10

5.5 Routing in Unreserved Layer MD Model

If the VCG contains one or more directed cycles, no topological ordering of nets is possible. In other words, ordering of track assignment cannot remain invariant in all columns. In reserved 2 -layer Manhattan model a channel with a cyclic VCG may not be routable unless a suitable column (or a blank column) is found to accommodate a dogleg. In unreserved or knock-knee model, this problem is relatively easy to tackle. In unreserved MD model, cycles in VCG can be handled by using overlapping of nets in different layers, or by X-routing (swap routing) [CR91]. In Fig. 5.11, we have studied Burstein's difficult example (BDE).



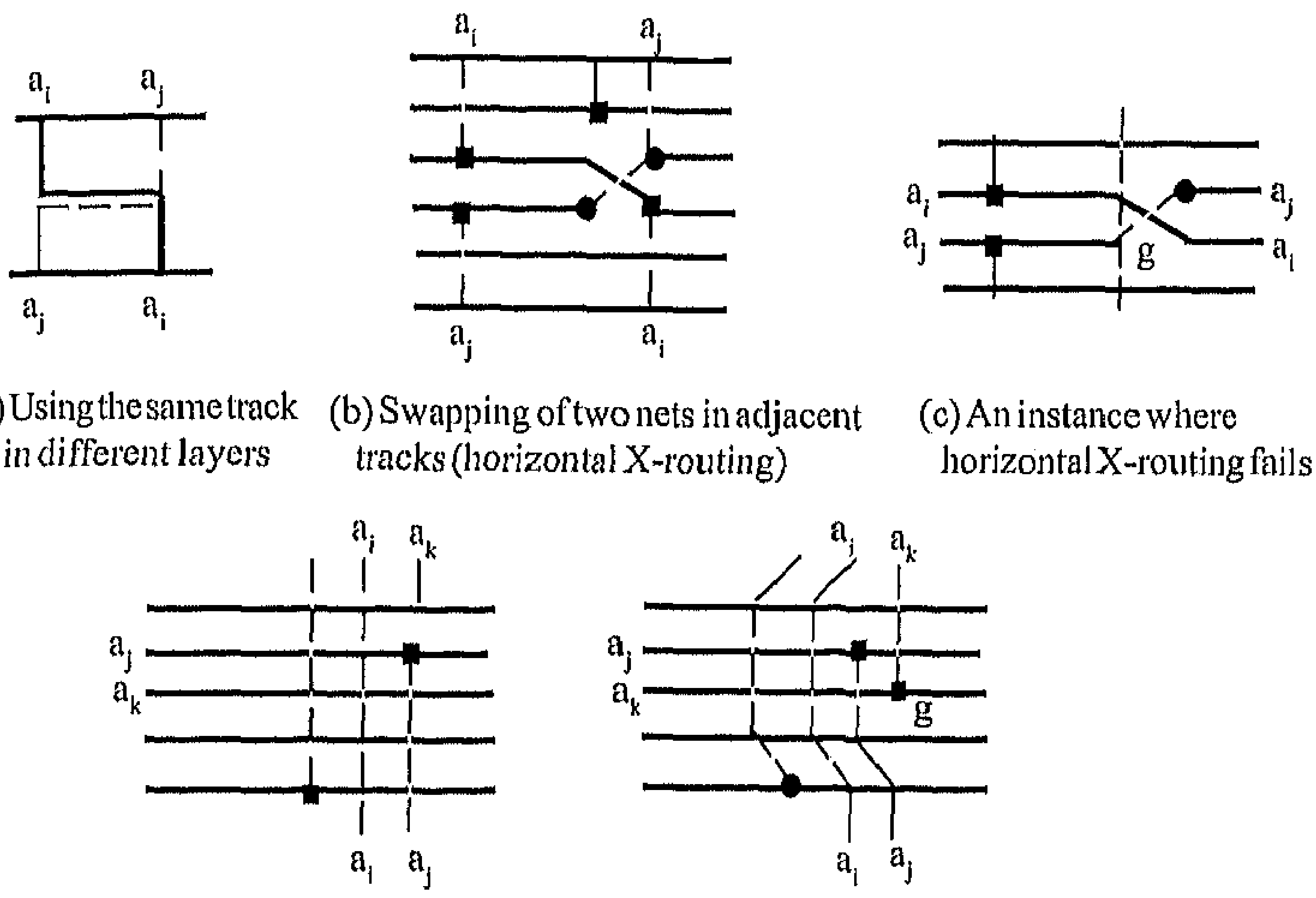
density : 3
# tracks : 2
# via : 1

Fig. 5.11 : Routing of Burstein's difficult channel in unreserved layer MD model

5.5.1 Resolving cyclic conflict in vertical constraints

(A) If the VCG of a channel routing instance has a cycle, two nets will appear in some column c_i in reverse order. In reserved layer model, to continue routing one needs to swap the track assignments. In the unreserved layer model if these two nets are routed in different layers (Fig. 5.12a), the conflict can easily be avoided. If they are in the same layer in column c_{i-1}, we can use horizontal X-routing provided they occupy adjacent tracks (Fig. 5.12b). However, there are cases (Fig. 5.12c) where, this cannot be done due to non-availability of via space at the grid point g. Cyclic conflict can also be eliminated by using vertical X-routing and column shifting (Fig. 5.12d).

(B) The number cyclic conflicts can be considerably reduced if we pick up minimum number of edges whose removal renders the remaining VCG acyclic. This calls for solving the **minimum feedback arc set** (MFAS) problem in a directed graph which is known to be NP-complete for general graphs [GJ79]. We use a linear time greedy heuristic for solving MFAS based on depth-first search.



(a) Using the same track *in different layers* (b) Swapping of two nets in adjacent tracks (*horizontal* X-*routing*) (c) An instance where horizontal X-routing fails

(d) column shifting (vertical X-routing)

Fig. 5.12 : Resolving cyclic conflict in unreserved layer MD model

5.5.2 Theme of the algorithm

We now present a simple greedy algorithm for routing a general channel in unreserved MD model. First, the VCG is made acyclic by removing minimum number of edges. This identifies the columns where nets are to be swapped. Topologically sort the remaining VCG to determine a linear ordering of nets. We assume $\lceil d/2 \rceil$ tracks are available initially.

The algorithm routes the nets in column-by-column fashion as before. Since horizontal segments in two layers may overlap, effectively we have d tracks to start with. Tracks are assigned to nets following the order (topological), and diagonals are inserted if necessary, when new nets appear. Cyclic conflicts are resolved using overlaps or X-routing. New tracks are inserted whenever necessary. Vertical connections are implemented using straight wire segments and minimum vias, and in the case, no such path is found, we check the possibility of re-routing it through its neighborhood (Fig. 5.13), or call a maze router [S93] on the grid structure of Fig. 5.2. If the maze router fails, we need to insert an extra track (or a blank column) to complete the vertical connection. The algorithm is formally described below.

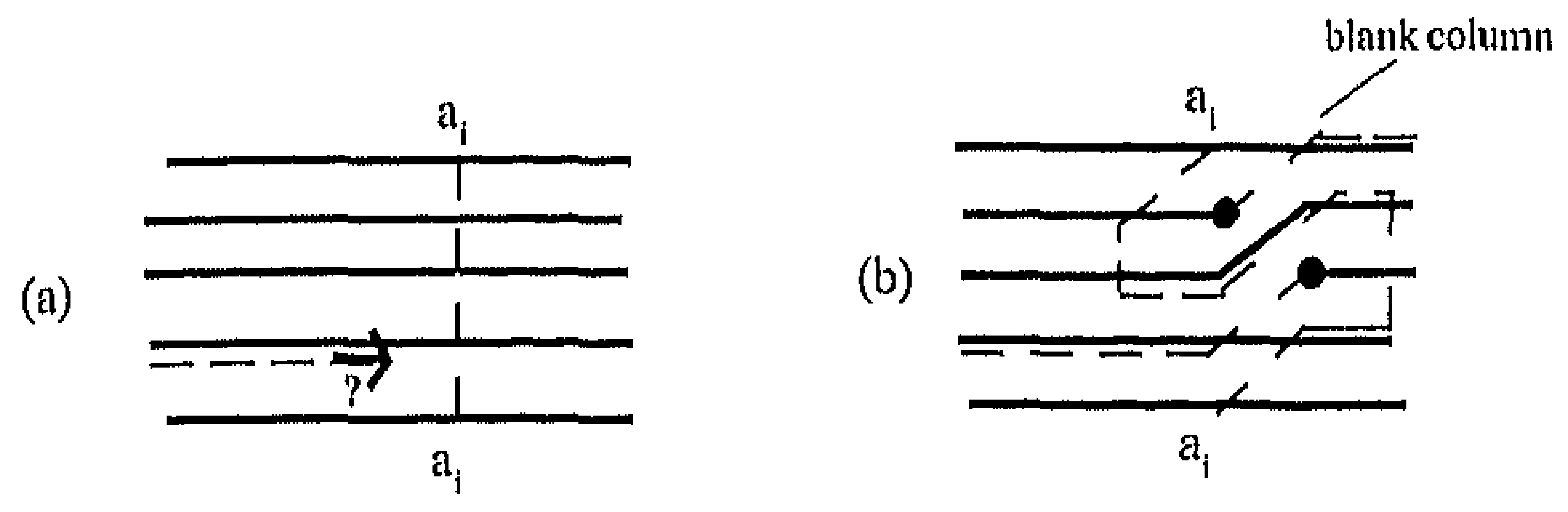


Fig. 5.13 : Implementing vertical connection in unreserved layer model using blank portion of the next column

5.5.3 Algorithm: 2 (Unreserved layer MD router)

Input : A channel C of length l, a netlist of n nets, and two routing layers;
Output : Unserved MD routing.

```
Step 1(a): Construct the VCG and make it acyclic by removing minimum number
           of edges using a simple greedy heuristic. Topologically sort the nodes
           of the acyclic VCG and construct a linearly ordered sequence of nets
           {$\alpha_1, \alpha_2, \ldots\ldots, \alpha_n$}. Determine the density $d$ of C ;
           /* Let $c_1, c_2, \ldots\ldots, c_l$ denote the columns ordered from left to right */

    (b): freetrack-layer-1 ($c_0$) = $\lceil d/2 \rceil$;  freetrack-layer-2 ($c_0$) = $\lceil d/2 \rceil$;
                              /* initially, $d$ tracks are free in both the layer of C */

  *Repeat steps 2 to 4 until* all the columns are scanned from left to right;

Step 2(a): *if* left- or rightmost terminal of any net does not appear in column $c_i$,
             *then* continue track assignment as in column $c_{i-1}$;

    (b): *if* r nets (r = 1 or 2) terminate in column $c_i$ *then* mark the free tracks;
             increment the free track count of the two layers accordingly;

    (c): *if* # new-net($c_i$) = 1, *then*
             *if* freetrack-layer-1 ($c_{i-1}$) > 0, *then* set q :=1; call TRACK(q)
             *else if* freetrack-layer-2 ($c_{i-1}$) > 0, *then* set q :=2; call TRACK(q)
                 *else* insert a free track in a position satisfying the topological
                     order, and assign new net in layer 1;
                     freetrack-layer-2 ($c_i$) := freetrack-layer-2 ($c_{i-1}$) +1;
```

```
(d): if  # new-net($c_i$) = 2,  then
        if freetrack-layer-1 ($c_{i-1}$) > 1, then execute step 2(d) of Algorithm 1
        else if freetrack-layer-1 ($c_{i-1}$) = 1, then
                if freetrack-layer-2 ($c_{i-1}$) > 0, then
                    set q :=1; call TRACK(q), set q :=2; call TRACK(q)
                else insert a free track in a position satisfying the topological
                        order and execute step 2(d) of Algorithm 1;
                freetrack-layer-2 ($c_i$) :=  freetrack-layer-2 ($c_{i-1}$) +1;
        else insert a free track in a position satisfying the topological order;
        set q :=1; call TRACK(q), set q :=2; call TRACK(q);

Step 3 :  /* let $a_i$ ($a_j$) be the net that is to be connected to the top (bottom)
             terminal in column $c_i$. */

      if the tracks $t_i$ and $t_j$ assigned to $a_i$ and $a_j$ are in the same layer and occupy
          adjacent tracks in column $c_{i-1}$, then
       if the edge ($a_i$ ,$a_j$) was removed for making of VCG acyclic, then
          interchange tracks assigned to $a_i$, $a_j$  by X-routing, else skip this step;

Step 4 :  Vertical-Connect (($g(t_i,c_i)$), ($top(c_i)$));
          Vertical Connect (($g(t_j,c_i)$), ($bottom(c_i)$));
          Place vias appropriately.
```

Procedure TRACK(q)

```
begin
   locate the track (t) in layer q, that fits the incoming net satisfying the topological
       order;
   make the track free if otherwise, by using diagonal connections from the grid
       points in column $c_{i-1}$ and assign the tracks in column $c_i$ accordingly;
   freetrack-layer-q($c_i$) := freetrack-layer-q($c_{i-1}$) -1;
end;
```

Procedure Vertical-Connect (($g(t,c_i)$), ($top/bottom(c_i)$))

/* g is a grid point formed by intersection of a horizontal track t and column c_i, and top (bottom(c_i)) denotes the top (bottom) terminal in column c_i. The goal is to interconnect these two points indicated by the parameters of the function. */

```
begin
    1. Connect them by a vertical line if possible, through either layer or using
       both layers and vias; else check whether re-routing is possible as in Fig.
       5.12 or Fig. 5.13;

    2. Failing 1, call a maze router with the given source-destination pair within
       the subgrid bounded by $c_k$ and $c_i$;
                  /* $c_k$ denotes the column where the maze router was called last;
                                                              initially, $k = 1$. */

    3. Failing 2, add a new horizontal track in the required position or a blank
       column (between $c_i$ and $c_{i+1}$ ); readjust tracks using diagonals; complete
       the interconnection using a vertical line.
end
```

Complexity : The VCG is made acyclic by removing minimal number of edges using a greedy heuristic based on depth-first search. Thus, Step 1 can be performed in $O(l)$ time. Step 2 requires $O(w.l)$ time over l iterations as before, where w is the channel width. Step 3 takes $O(l)$ time. Vertical connections and placement of vias in Step 4 takes $O(w.l)$ time over l iterations, because the search space in the (l x w) grid over which the maze router is called is distinct in each call. Thus the overall complexity of the algorithm is $O(w.l)$, i.e., *linear in area* of the channel. The space complexity is also $O(w.l)$. The time complexity of the router is much better than those reported in [YK82, PPDP95, HIZ91, CR91].

5.5.4 Experimental results

We have implemented the algorithm along with a heuristic for the MFAS problem and a simple maze router. Table-5.2 compares its performance with the results obtained in Manhattan model [YK82, HIZ91, RVS85, RF82, D85, HI89], and diagonal model [CR91]. The example CR is taken from [CR91]. Experimental evidence reveals that our solutions compare favorably with the best-known existing results with respect to channel width and consequently, channel area. Using two layers, the proposed method outputs 18 tracks for DDE, and 2 for BDE (Fig. 5.11).
The algorithms are simple, easy-to-implement and produce outputs in linear time. As seen from the table, only a few calls are made to the maze router. The number of vias and wire length also improve compared to those in [YK82, TWC91] (Table-5.3). Efficient routing solutions of staircase channels and switchboxes using the

MD-model have been reported in the next chapter. The technique can be adapted for multi-layer routing as well.

Table 5.2 : Number of tracks for benchmark examples

Example	Density	Number of tracks by earlier routers (2 layers)	Proposed router in MD-model		
			# tracks	#extra blank columns	#calls to maze router
YK_{3a} (Fig. 5.14)	15	15 [YK82]	13	0	3
YK_{3b} (Fig. 5.14)	17	17 [YK82]	15	0	2
YK_{3c} (Fig. 5.14)	18	18 [YK82]	17 16	0 1	1 1
DDE* (Fig. 5.15)	19	19 [BP83, HIZ91, RVS85]	18	0	0
CR	19	17 [CR91]	17	0	0
BDE** (Fig. 5.11)	4	3 [HIZ91]	2	0	0
Cycle.tough	16	16 [HI89]	15 14	0 1	0 3

* Deutsch's difficult example ; ** Burstein's difficult example ; entries within square brackets indicate references.

Table 5.3 : Comparison number of vias and wire length

Example	Number of vias			Wire-length		
	Router output	Via minimizer [TWC91]	Proposed method	Router output	Algorithm in [TWC91]	Proposed method
YK_{3a}	91 [YK82]	66	57	1365	1362	1228.37
YK_{3b}	107 [YK82]	78	66	1651	1653	1540.40
YK_{3c}	125 [YK82]	103	99	2100	2100	2061.30
YK_{3c}	125 [YK82]	103	97	2100	2100	2027.54
DDE	301 [D85]	226	225	4989	5058	4909.64
BDE	3 [HIZ91]	-	1	71	--	56.04

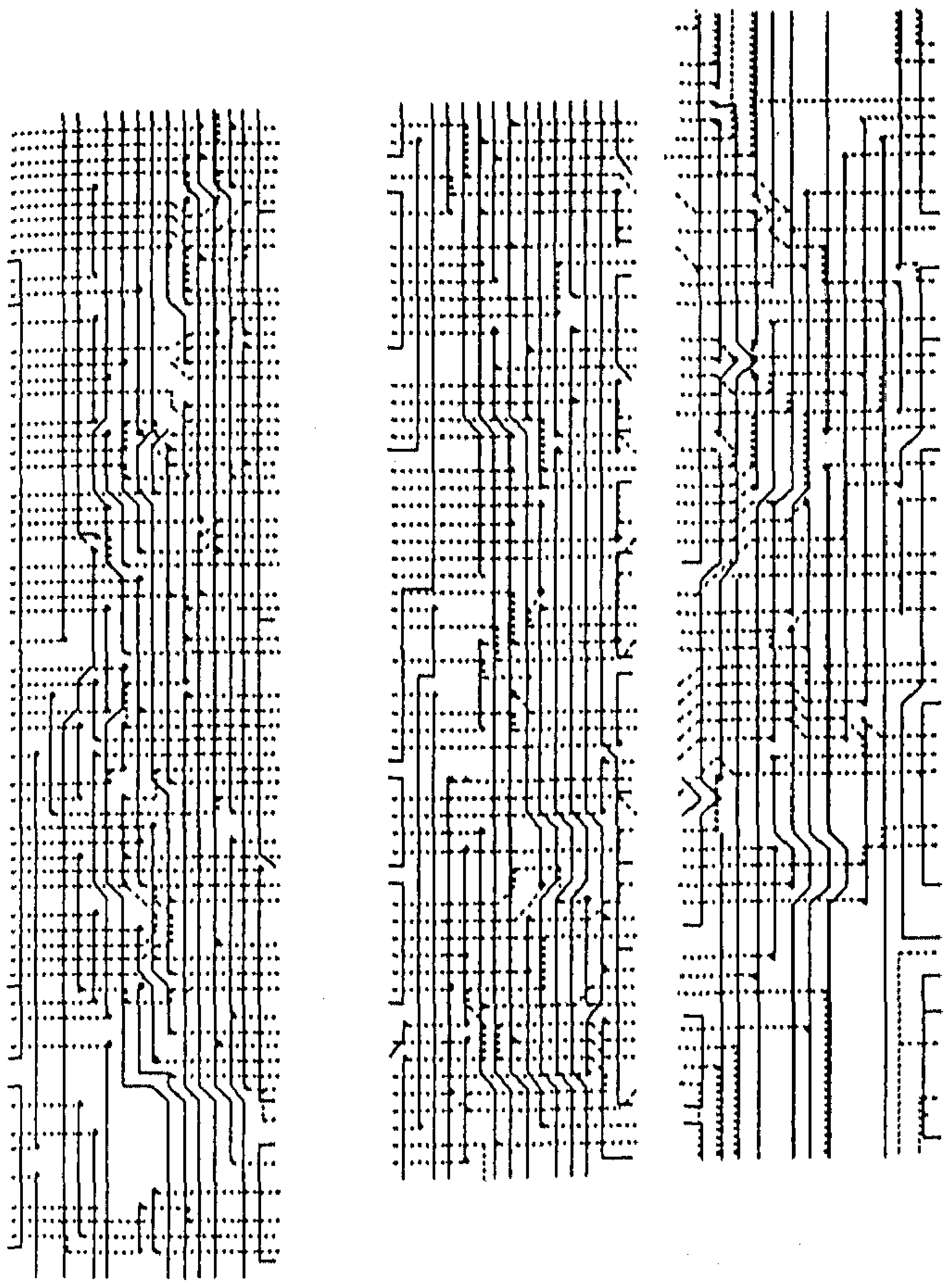


Fig. 5.14 : Routing of YK_{3a}, YK_{3b}, YK_{3c} in unreserved layer MD model.

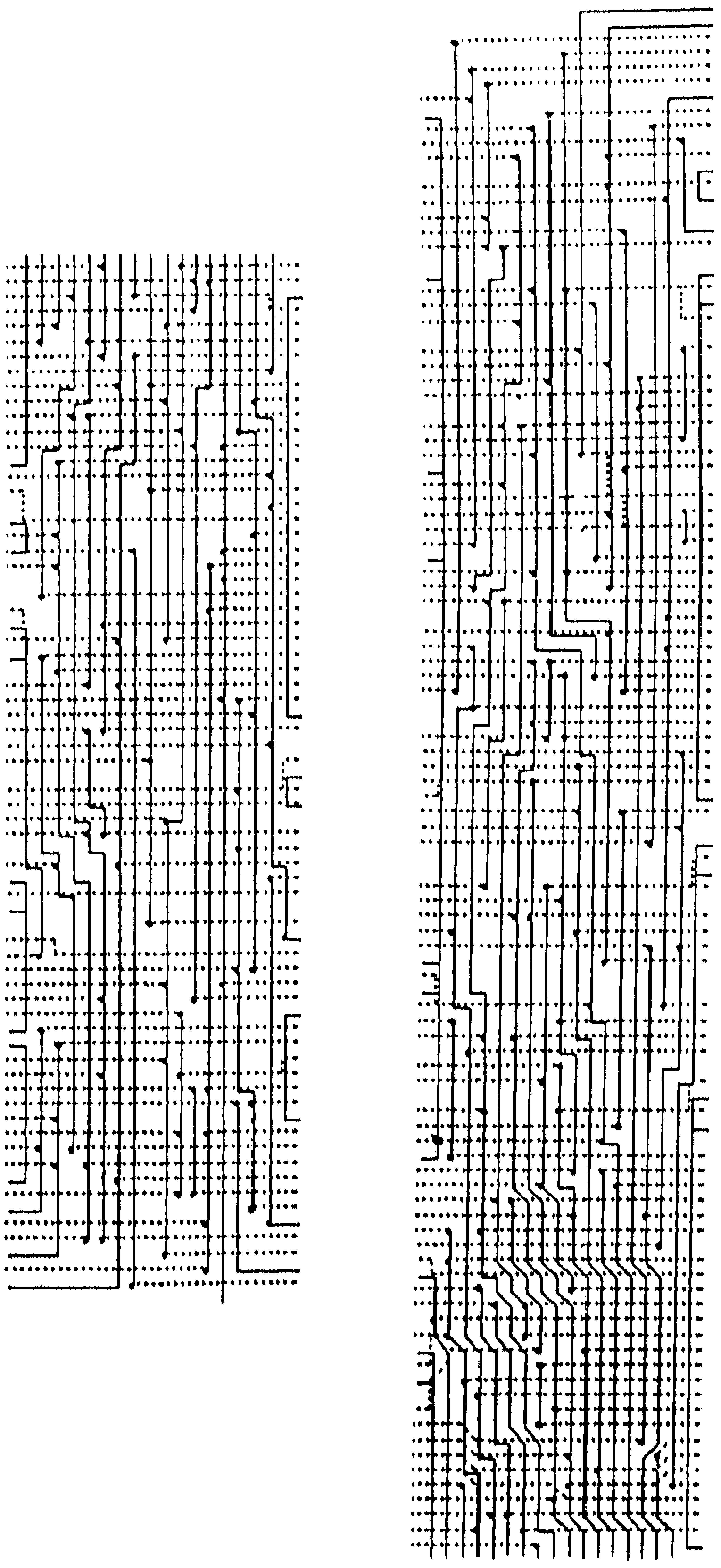


Fig. 5.15 : Routing of Deutsch's difficult example (DDE) in unreserved layer MD model.

Chapter 6

Routing in L-Shaped Channels, Switchboxes, and Staircases

6.1 Introduction

In the area routing problem, the routing area is first decomposed into a set of routing regions (channels), then global routing of the nets through the channels and finally detailed routing in each channel are carried out [YK82, HIZ91]. Given a netlist in a routing region, the objectives of a detailed router are to complete interconnection, minimize channel area, wire length, number of vias, crosstalk, and maximize yield. Layout topology imposes an order in which detailed routing of the channels is to be completed, as the location of some of the terminals may depend on the routing in an adjacent channel. Assuming rectangular chip, blocks and channels, slicible topologies can be represented by a binary tree and post-order traversal of the internal nodes of this tree provides the safe routing order [K83, DAK85], e.g., <2, 3, 4, 1> in Fig. 6.1a. Kajitani [K83] first introduced the concept and term of safe routing order and studied the problem analytically. In order to overcome cyclic dependencies in routing order for nonslicible topologies (Fig. 6.1b), L-shaped routing regions [DAK85, SB91] were introduced; thus in Fig. 6.1c, the safe order is <1, 2, 3, 4'>, where 4' is an L-shaped channel. L-shaped channels were further generalized to monotone staircase channels in [GW88, SB91, MNB98] which guarantee safe routing order. The problem of safe ordering with minimum number of switchboxes has also been solved efficiently [YH96]. Routing of a 4-sided switch box is a difficult problem in general; it also arises in tower routing of MCMs [S93, SBP95]. These issues lead us to solving the problem of routing L-shaped channels, staircases and switchboxes.

Algorithms by Maddila et al. [MZ89] for general routing regions are based on the restrictive knock-knee model [S93]. Tsai et al [TCCH92] formulated general area routing as a planning problem in which the routing process is decomposed into a conjunction of subgoals; each subgoal consisting of selection of net segments and assignment of track resources. However, it may need to backtrack and may not report any solution as it uses the Manhattan model even if Manhattan-diagonal routing exists. The time complexity of this method is high. Among non-Manhattan geometries, a diagonal model with $\pm 45^0$ wires has been adopted [LLS91, C87]. Song [S92] presents an optimal knock-knee diagonal routing for L-channel with two-terminal nets only. In Manhattan-diagonal (MD) model which combines diagonal and Manhattan geometries, routing method for straight channels based on sorting was proposed in [CR91]. Another MD channel router was presented by Wang [W91] but the track count and via count for denser channels seem to be higher and too many jogs are required. In the previous chapter we discussed a simpler MD channel router providing reduced wire length and vias. Nevertheless, applicability of these methods to realistic routing in general regions are to be studied.

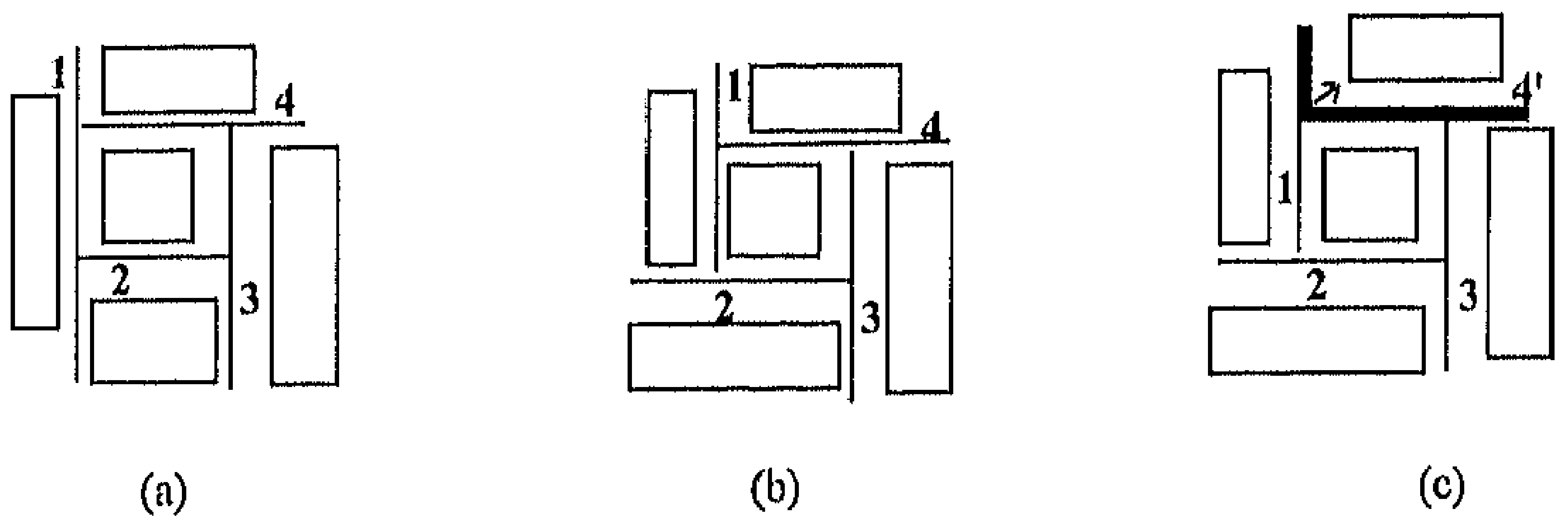


Fig. 6.1 : Safe routing order of channel

In this chapter, we propose new algorithms for detailed routing of L-shaped channels, staircases and switchboxes in the 2-layer MD model. Section 6.2 deals with L-shaped channel routing; the core of the proposed method lies in routing a right triangular region. This method is greedy, uses minimum number of tracks, can handle cyclic constraints and yields encouraging results. Next in Section 6.3, a switchbox router in MD-model is proposed, which outperforms existing routers. No extra row or column is required to route Burstein's and other difficult benchmark switchboxes. Finally, a scheme for staircase channel routing is given. The complexity of all algorithms are linear in the area of the region. These techniques can provide a unified and efficient routing paradigm to solve a general class of routing problems.

6.2 Routing in L-shaped Channels

An L-shaped channel (L-channel) has equidistant terminals of nets along the boundary of an L-shaped region with no obstacles in its interior. The objective is to connect the nets using minimum width. If extra tracks are required for completion of routing, the L-channel is expanded in the direction marked out in Fig. 6.1c. Among the previous attempts, the router in [C87] divides an L-channel into a vertical channel with terminals on opposite sides, routed by one of the standard Manhattan channel routers, and a horizontal channel with terminals on three sides — its routability cannot be ensured always in Manhattan unreserved layer model without expansion (see Fig. 6.2, where top terminal of net 3 is not possible to connect with its horizontal segment).

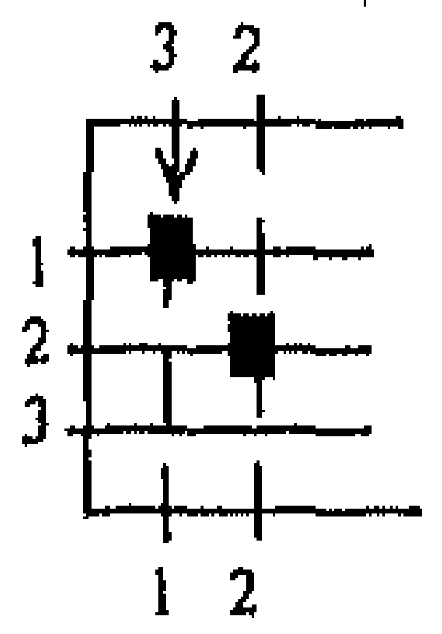


Fig. 6.2 : Three-sided channel

With 45^0 wires, Chen's method maps an L-channel entirely into a straight channel, routes it by standard reserved layer Manhattan router and then replaces all the vertical wires by diagonal wires. However, there are two triangular regions at both ends of the L-shaped region, which have to be routed later on; this was not taken care of explicitly in [C87]. If the two sub-channels are of same density, then this easy method gives a space efficient solution. Since channel boundary can be moved in 45^0 direction only, the less dense sub-channel may have empty tracks unless a refinement pass is carried out. The second major disadvantage of Chen's approach is that it uses many long diagonal wires that may not only increase the wire length but also the probability of manufacturing defects even under the modified set of design rules in Fig. 5.1 of previous chapter. In [MZ89], an L-channel is divided into three parts, namely two straight channels and a special switchbox junction with fixed terminals on two adjacent sides. The junction is routed first and then the two three-sided straight channels are routed using the knock-knee model. An efficient routing algorithm to overcome the above-mentioned shortcomings is presented in this section. This algorithm relies on solving the triangular routing problem.

6.2.1 Scheme for L-channel routing

In our approach for routing L-channels, the L-channel is at first split into two subregions A and B (Fig. 6.3a), each of which is further decomposed into a two-sided channel and a triangular region (Fig. 6.3b). It may be pointed out that the triangular region in our scheme is near the L-junction as opposed to that at the open end of the L-channel in Chen's algorithm [C87]. The vertical subregion A is rotated by 90^0, placed to the left of subregion B (Fig. 6.3c) and the density of the entire channel is computed. Ignoring the terminals within the two triangular regions A'' and B'', the routing is completed by standard Manhattan channel router. Thus track assignment for the nets is accomplished and the sequence is identical along the two hypotenuses of the two triangular regions. Finally, algorithm MD-T described below completes the triangular routing in A'' and B''. If extra columns are required by MD-T for A'' (B''), then the routing in sub-region A' (B') is modified accordingly by a scan and using a few diagonal wire segments. The novelty of this algorithm is in handling the bend by triangular routing in MD model; most of the channel has Manhattan wiring hence reducing wire-length. The complexity remains proportional to the area of the L-shaped routing region.

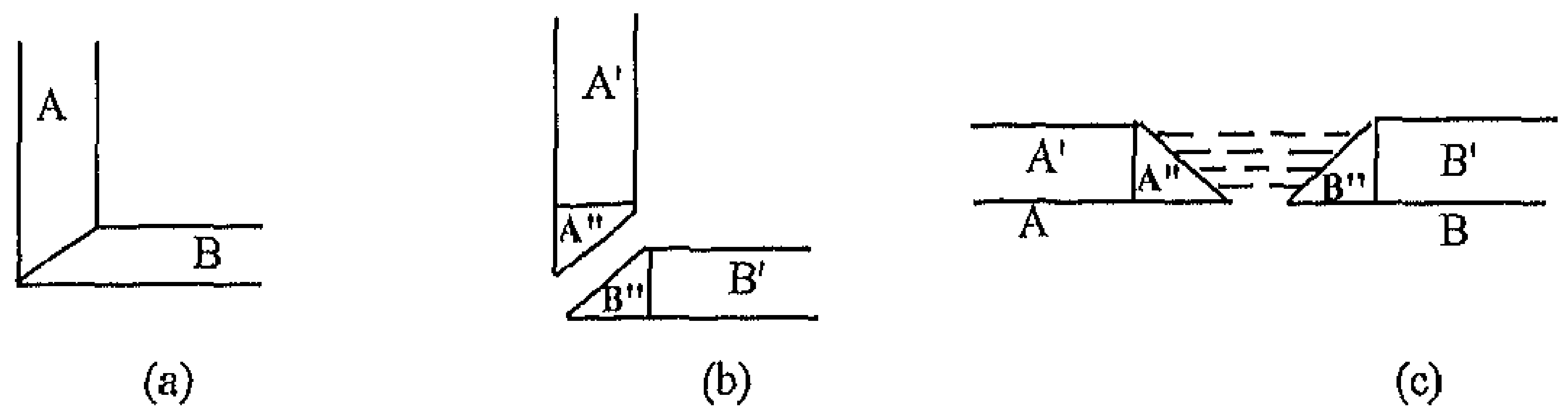


Fig. 6.3 : Key steps of MD-L

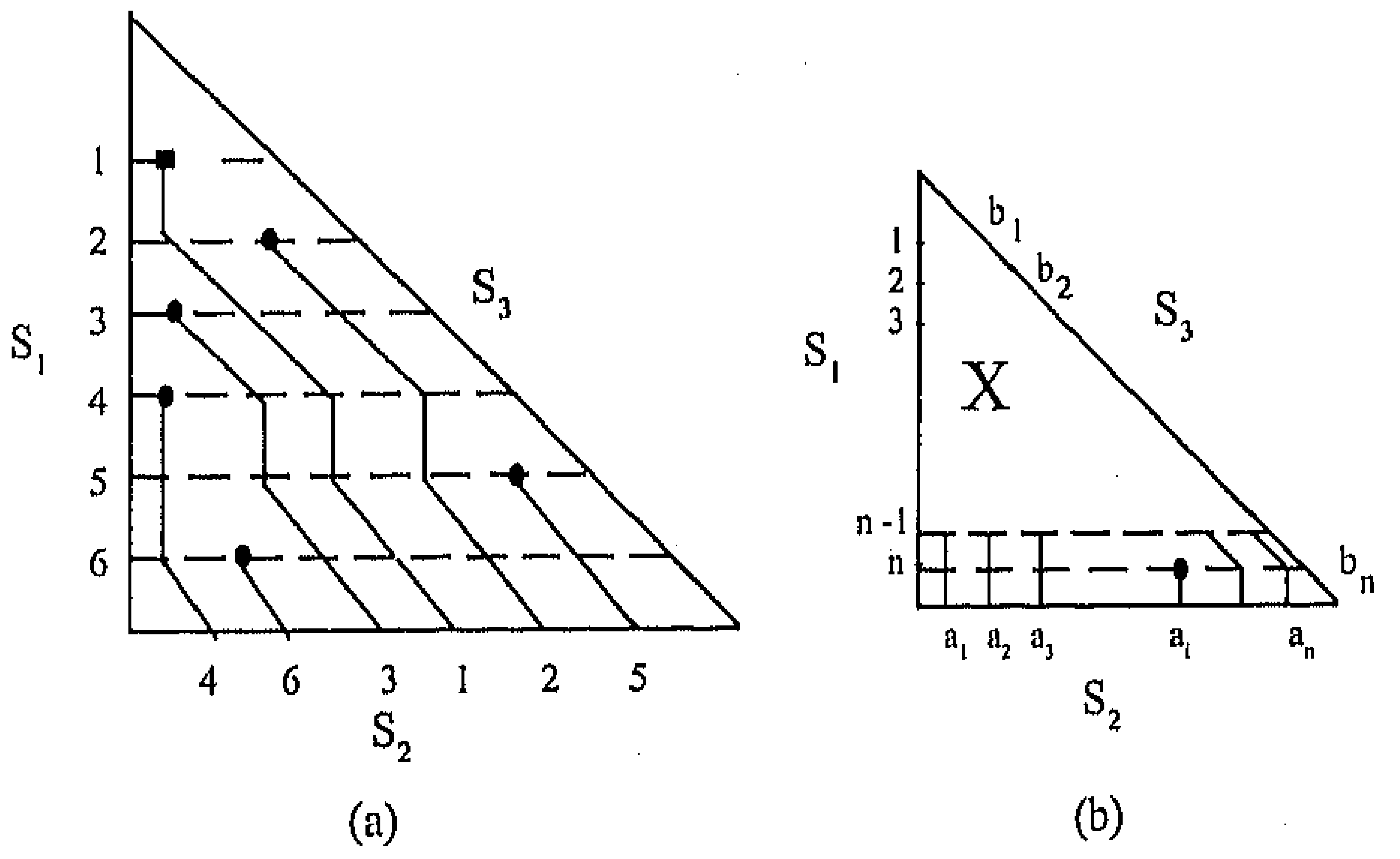
6.2.2 Triangular channel routing

Consider an isosceles right angle triangle (Fig. 6.4a) with its two equal sides S_1 and S_2, and S_3 being its hypotenuse. Without loss of generality, let the terminals on S_1 be distinct and labeled <1, 2,...., n> sequentially. Any terminal on the other two sides has a label i, $0 < i < n$, where 0 represents a null terminal. The objective is to connect all the nets within the triangular area; reducing via count and/or wire length may be secondary. Now, several cases may arise.

Case 1 : The sequence of terminals on S_3 is same as that of S_1, and that on S_2 a permutation of <1, 2,..., n >.

Lemma 6.1 : *An instance of Case 1 is always routable in the MD model within the triangular area.*

Proof : For $n = 2$, routing in the triangular area is possible. Now, suppose the lemma holds for n-1 nets. For the induction step, consider the triangular routing problem with n nets having the sequence of terminals on S_1 and S_3 as $< 1, 2, \ldots\ldots, n >$, and the terminal sequence for on S_2 is $< a_1, a_2, \ldots\ldots, a_n >$ which is a permutation of $< 1, 2, \ldots\ldots, n >$. Thus the terminals on S_1 and S_3 are routed using horizontal tracks. Let $a_i = n$, then it can be connected to the n-th horizontal track using a vertical segment. Vertical wire segments from all terminals on the left of a_i and diagonal segments from all terminals on the right of a_i to the n-th horizontal track are inserted (not necessarily connected). This yields a triangular region X (Fig. 6.4b) having n-1 nets which by induction hypothesis is routable within that area and hence the lemma. ■



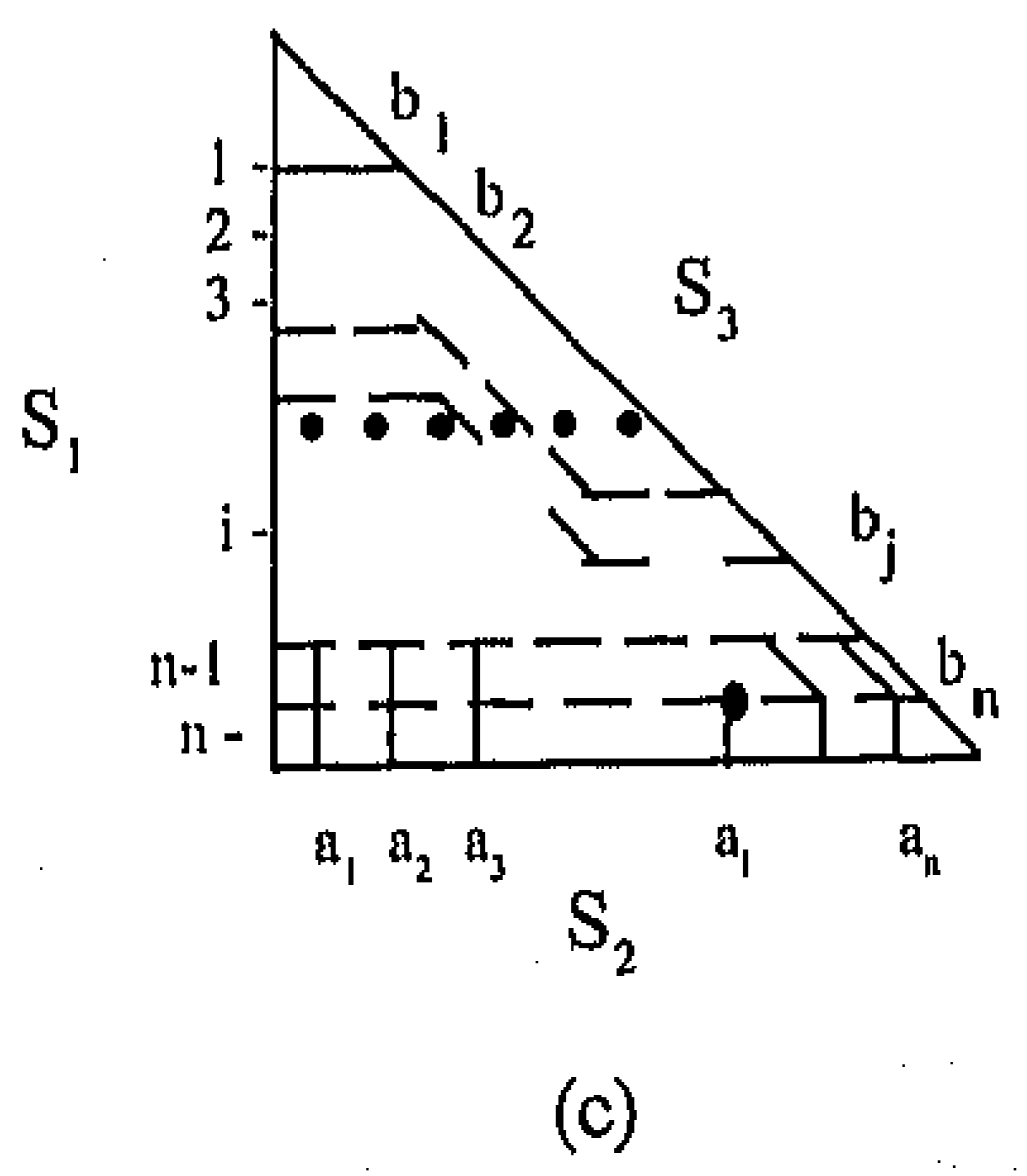


Fig. 6.4 : Triangular routing

Case 2 : Some of the nets may have more than one terminals on S_2, while some terminals may be 0. Each net appears at least on one of S_3 and S_1.

We need to study the following two sub-cases.

Case 2.1 : Sequences of terminals on S_3 and S_1 are identical.

Lemma 6.2 : *An instance of Case 2.1 is routable in MD model within the triangular area if for any net i on S_1, the number of terminals on S_2 having labels either 0 or in the range {i+1, ..., n} is at least (n-i).*

Proof : The identical sequences on S_1 and S_3 are connected by horizontal tracks. For any net j, its terminals on S_2, if any, have to be connected to the j-th horizontal track using vertical and diagonal wire segments. By reasoning similar to that for Lemma 1, for any net i, the triangular region above the i-th horizontal track is routable if the number of nets whose terminals on S_2 yet to be routed, is less than i. ■

Case 2.2 : Sequences on S_1 and S_3 are not identical. Here let $<b_1, b_2, \ldots, b_n>$ be the sequence of terminals on S_3. For any i, the label b_i belongs to $\{1, 2, \ldots, n+k\}$, and any terminal on S_2 belongs to a net which also has a terminal on either S_1 or S_3 or both. Now we study the following sub-cases.

Case 2.2.1 : If terminals of any net appear on both S_1 and S_3, then their corresponding indices in S_1 and S_3 are equal, i.e., $i = b_i$. This is the case where no dogleg is necessary.

Lemma 6.3 : *An instance of Case 2.2.1 is routable in MD model within the triangular area, if for any $i \neq b_i$ on S_2, (i) all the terminals of net i is to the left of all terminals of net b_i, and (ii) for any i on S_1, the number of terminals of the nets in {1, 2,...,i-1} and the nets $\{b_1, b_2, \ldots, b_{i-1}\}$ on S_2 is less than i.*

Proof : The terminals of nets which appear on both S_1 and S_3 are connected by horizontal tracks as $i = b_i$. For $i \neq b_i$, the corresponding horizontal track is shared by the two nets with left segment of the track occupied by net $i \in S_1$ and right segment of the track by net $b_i \in S_3$. For any net j, its terminals on S_2, if any, are connected to the j-th horizontal track using vertical and diagonal wire segments. This can be achieved if on S_2, all terminals of net i appear to the left of all terminals of net b_i. Here some of the nets starting on S_1 will end on S_2 and then to the right of these terminals on S_2, some new nets {n+1, n+2,....., n+k} start which end on S_3.

Now, as in Lemma 1, the triangular region above the i-th horizontal track for any i, is routable if the number of terminals on S_2 yet to be routed is less than i. These terminals must belong to nets $j < i$ or to any of $\{b_1, b_2, \ldots, b_{i-1}\}$. ■

Case 2.2.2 : If terminals of any net appear on both S_1 and S_3, then their corresponding indices in S_1 and S_3 are not necessarily equal, i.e., there exists $i \neq j$ such that $b_i \in S_3$ is a terminal of net j.

This is a case of doglegs (see Fig. 6.4c) and may require extra horizontal track and/or column if $b_j = i$ for $j \neq i$. While this may arise in general, in the context of the L-shaped channel routing algorithm MD-L which spawns off an instance of triangular routing this case of doglegs will never arise.

We now present an algorithm for routing a triangular region generated by the algorithm MD-L.

Algorithm MD-T for triangular routing

Input : Sequence of terminals $<a_1, a_2, a_3, \ldots, a_n>$ on S_2 and sequence $<b_1, b_2, b_3, \ldots, b_n>$ on S_3.
Output : MD-routing within the triangular area.

```
begin
    $l$ := 0;                    /* $l$ is the leftmost column used and is initially the
                                        left boundary of the triangle */
    for i := n downto 1 do;
        begin
            $c$ := 1;            /* $c$ denotes the rightmost column with null ter-
                                    minal and is initialized to 0 the leftmost column
                                        along the left boundary of the triangle */
            if null terminal exists in row i then $c \leftarrow$ rightmost null terminal
                else $l$ := $l$ - 1;  $c$ := 1;
            insert diagonal wire segments between rows i-1 and i for nets of
              $S_2$ in layer 1 from right to left upto column $c$;
            insert vertical wire segments between rows i-1 and i in layer 1 for
              all the terminals or endpoints of wire segments to the left of $c$;
            if i = $b_l$ then
                begin
                    insert horizontal wire segments connecting i and $b_l$ in
                        layer 2; insert via(s) to connect diagonal/vertical
                        wire segment(s) of net i in layer 1;
                    /* Sequence $S_2$ is updated such that nets connected
                        to vias in row i are replaced by 0 and then the
                        $c$-th element is deleted */
                    for all $a_j \in S_2$ if $a_j$ = i then $a_j$ = 0;
                    $S_2 := < a_1, a_2, \ldots, a_{c-1}, a_{c+1}, \ldots, a_n >$;
                end
            if $b_l$ is not in $S_l$ then
                begin
                    if rightmost terminal of net i is left of the leftmost terminal
                        of net $b_l$ in $S_2$ then net i and $b_l$ share the same track
                      else insert a track between row i and i-1;
                    insert horizontal wire segments for i and $b_l$ in layer 2;
                    insert via(s) to connect diagonal/vertical wire segments
                      of net i and $b_l$ in layer 1;
                    for all $a_j \in S_2$ if $a_j$ = i or $b_l$ then $a_j$ = 0;
                    $S_2 := < a_1, a_2, \ldots, a_{c-1}, a_{c+1}, \ldots, a_n >$;
                end
        end for
end.
```

Theorem 6.1 *Algorithm MD-T produces a valid routing.*

Proof : If the conditions in Lemmas 1 to 3 hold, then the theorem follows. For the case 2.2.2, algorithm MD-T adds extra column(s) and/or horizontal track(s) and always completes the routing. ■

Complexity : The complexity of the above algorithm MD-T is $O(n^2)$, i.e., linear in the area of the routing region. It may be noted that horizontal and vertical wires are on reserved layers thus non-overlap, but diagonal wires may appear in either layer and can overlap.

6.2.3 L-channel router MD-L

In this subsection, we present our L-channel routing algorithm MD-L based in the scheme discussed in Sec 6.2.1. As pin or terminal alignment across a routing region influences routing as well as placement, significant feature of MD-L is virtual pin alignment for further reduction in channel density. Shifting the boundary of a channel can change the alignment of pins and hence the density. It involves inherent trade-offs between minimizing area and wire length. The density of a channel varies with the pin alignment. It may be pointed out that the boundary of the straight channel (obtained by transforming an L-channel) cannot be shifted by more than the density of the L-channel initially. The pin alignment for which the density is minimum, is obtained as follows. Suppose the vertical segment A of an L-channel is bounded by lines ab and pq as in Fig. 6.5. If b is aligned with q by shifting the top boundary towards the right over the lower boundary, then the density may change. Let d and d' be the original densities of the given sub-channels A and B after alignment. Suppose nets $< a_1, a_2, \ldots, a_k >$ occur in both A and B. An operation *Move(x)* is defined which shifts the bottom boundary of A towards right up to x terminals. Then the corresponding density of A is denoted by *den(x)* and the set of x rightmost terminals on the lower boundary of A by $T_s(x)$ (Fig. 6.6). Similar definitions hold for the horizontal sub-channel B of the L-channel. Suppose the densities in sub-channels A and B are minimum for movement of their respective boundaries over x_A and x_B terminal positions. The minimum possible density of the entire L-channel under pin alignment scheme is :

$$d_{min} = \min_{xA < d,\, xB < d'} (\max(\text{den}(x_A), \text{den}(x_B), |P|))$$
$$\text{where } P = T_s(x_A) \cup T_s(x_B) \cup \{a_1, a_2, \ldots, a_K\}.$$

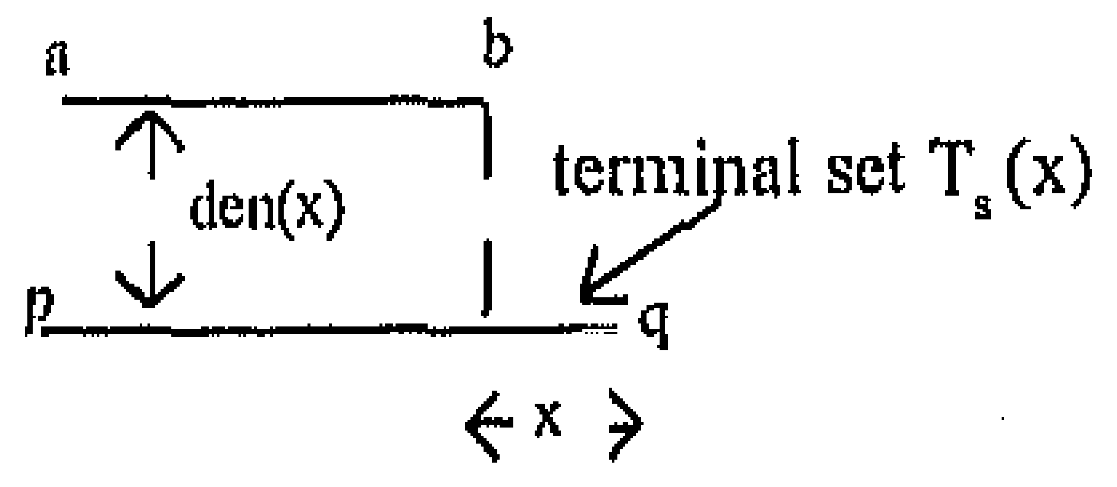


Fig. 6.6 : Pin alignment in MD-L

Algorithm MD-L for L-channel routing

Input : Sequence of terminals along the border of L-shaped channel.
Output : MD-routing within the L-shaped area.

Step 1. Compute d_{min} for L-channel
Compute x_A and x_B.

Step 2. Rotate subchannel A by 90^0, place it to the left of subchannel B.

Shift the terminals on the bottom boundaries of A and B by x_A right and x_B left respectively to obtain a straight channel C.

Step 3. Route the straight channel C by a greedy router [HIZ91].
$w \leftarrow$ the width of the channel.

Step 4. Form triangular region for subchannel A by topmost x_A tracks and rightmost x_A columns.
Route the triangular region by MD-T.
Route the counterpart for subchannel B by MD-T.
Move terminals on the bottom boundaries of subchannel A and B by shifting $x = w - x_A$ position right and $x' = w - x_B$ position left respectively.
Shift x and x' bottom most tracks in A and B accordingly and replace vertical segments by appropriate diagonal connections.

Step 5. Place routing output from *step 4* in the form of L-shaped by joining the corresponding ends of A and B.
Scan from bottom to top of A and left to right of B for removal of diagonal segments in unreserved layer model whenever room is available (Fig. 6.6).

In Step 3, this algorithm scans the terminals in channel C from left to right and the greedy router [S93, HIZ91] completes routing for one column before proceeding to the next. The justification for employing a greedy router is that due to shifting of terminals there is a likelihood of introducing cyclic constraints in the VCG. If no cyclic constraints are introduced after pin alignment corresponding to d_{min}, on MD reserved layer channel router algorithm (proposed in the previous chapter) is used as it guarantees a solution with at most $d_{min} + 1$ tracks.

Complexity : The complexity of MD-L is $O(l.d_{min})$ for a channel of length l.

6.2.3.1 Experimental results

The total diagonal wire length, width and number of vias in the routing solutions produced by algorithm MD-L for some difficult examples (Figs. 6.6-6.8) are summarized in Table 6.1. Comparison of its performance with that of the algorithm in [C87] is also made. In the routing solutions of Figs. 6.7-6.9, the via count is found to be significantly small; the circular vias in the figures depict the recommended octagonal vias and these account for less than 10% of the total via count.

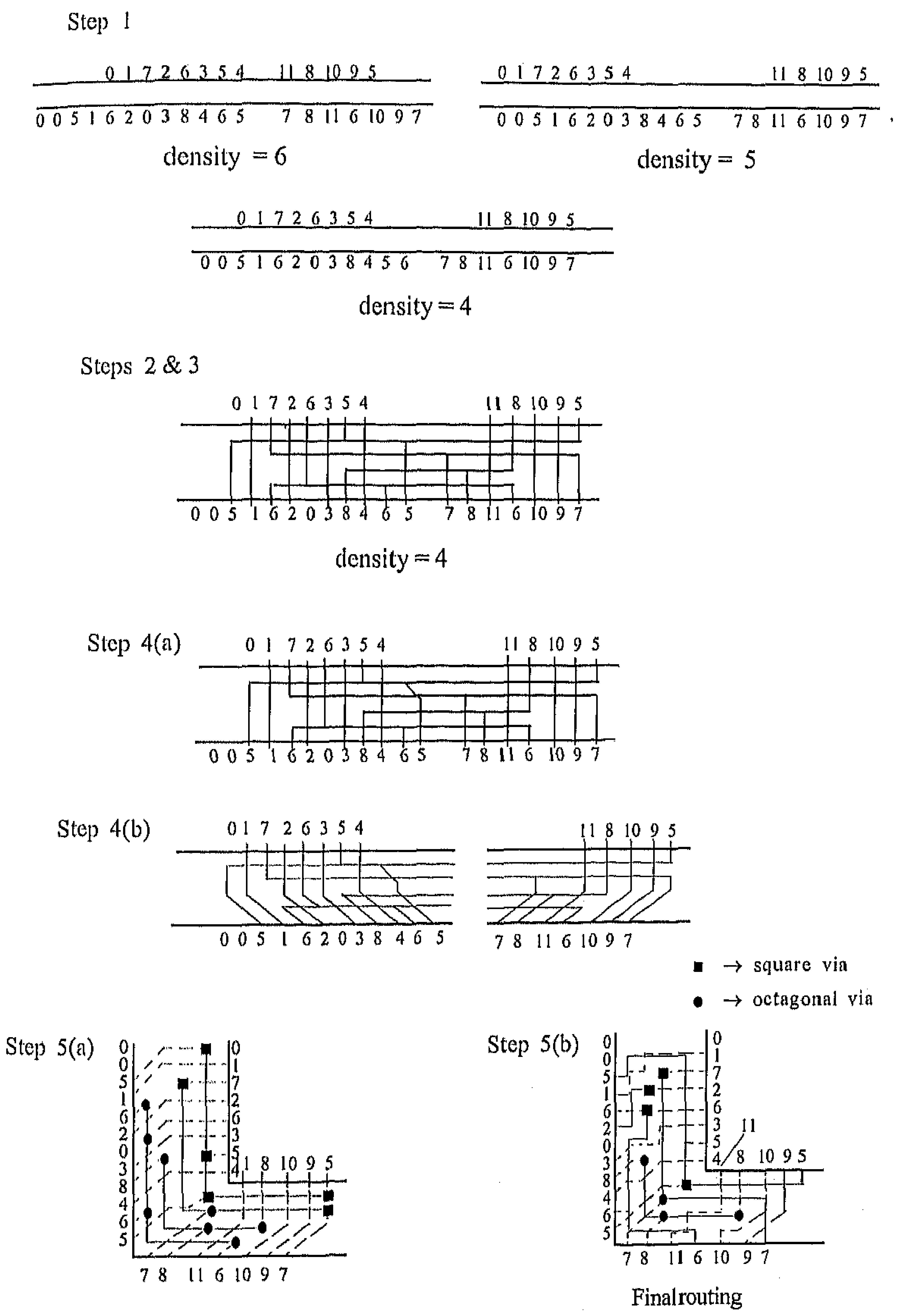
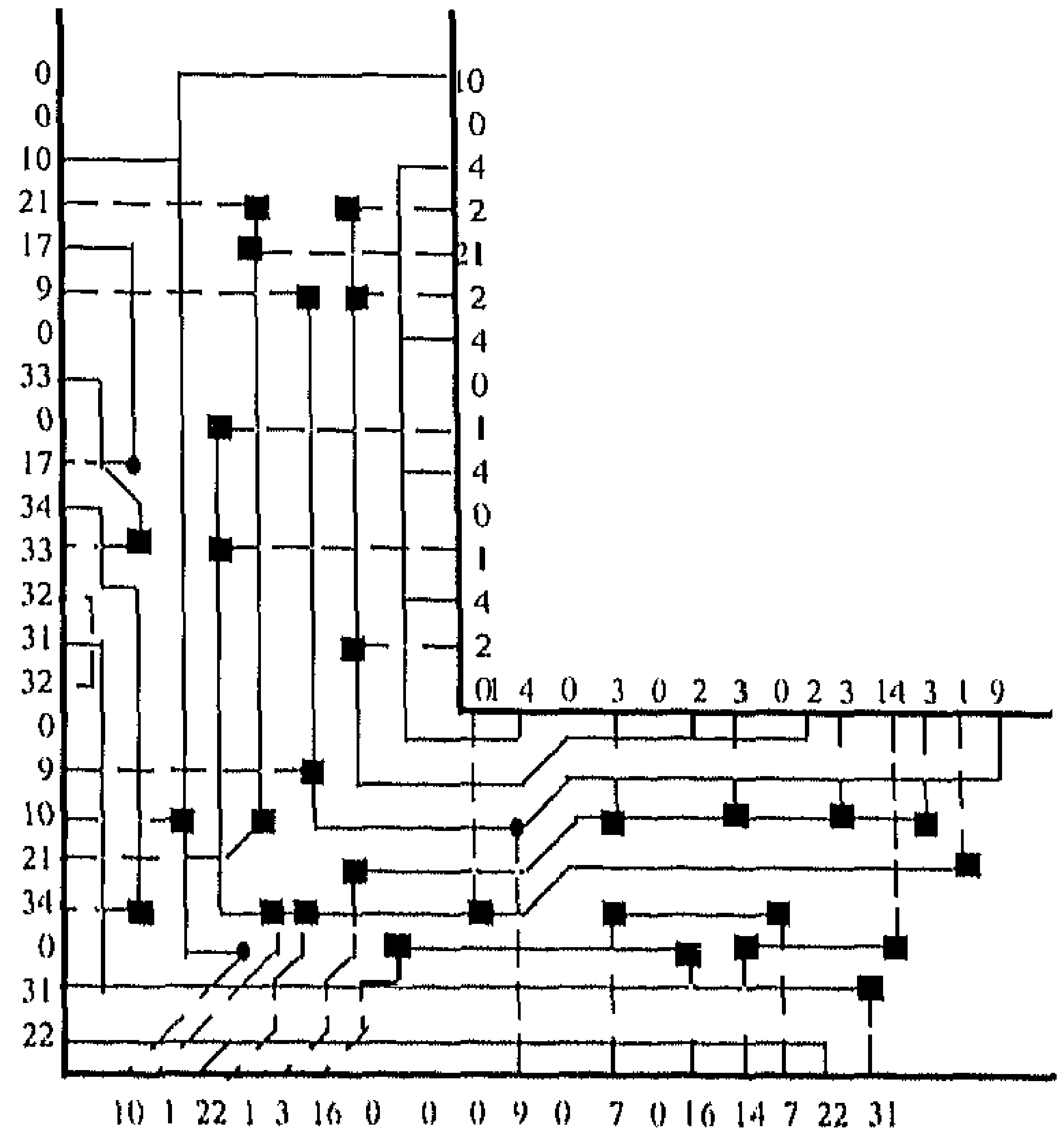


Fig. 6.6 : Routing steps in MD-L for Example 1

Algorithm MD-L is easy to implement and terminates very quickly. In addition to routing examples of L-channels, an instance of a nonslicible floorplan (Example 6, Fig. 6.9) has been created which has one L-channel and three straight channels. Of these three, one is an instance of Burstein's difficult channel [J86] and another of YK1 [YK82]. The L-channel has been routed with the proposed MD-L, Burstein's channel with MD unreserved layer channel router, and the remaining two straight channels including YK1 by a standard Manhattan router [YK82]. This demonstrates an efficient routing with minimum area for an entire floorplan.

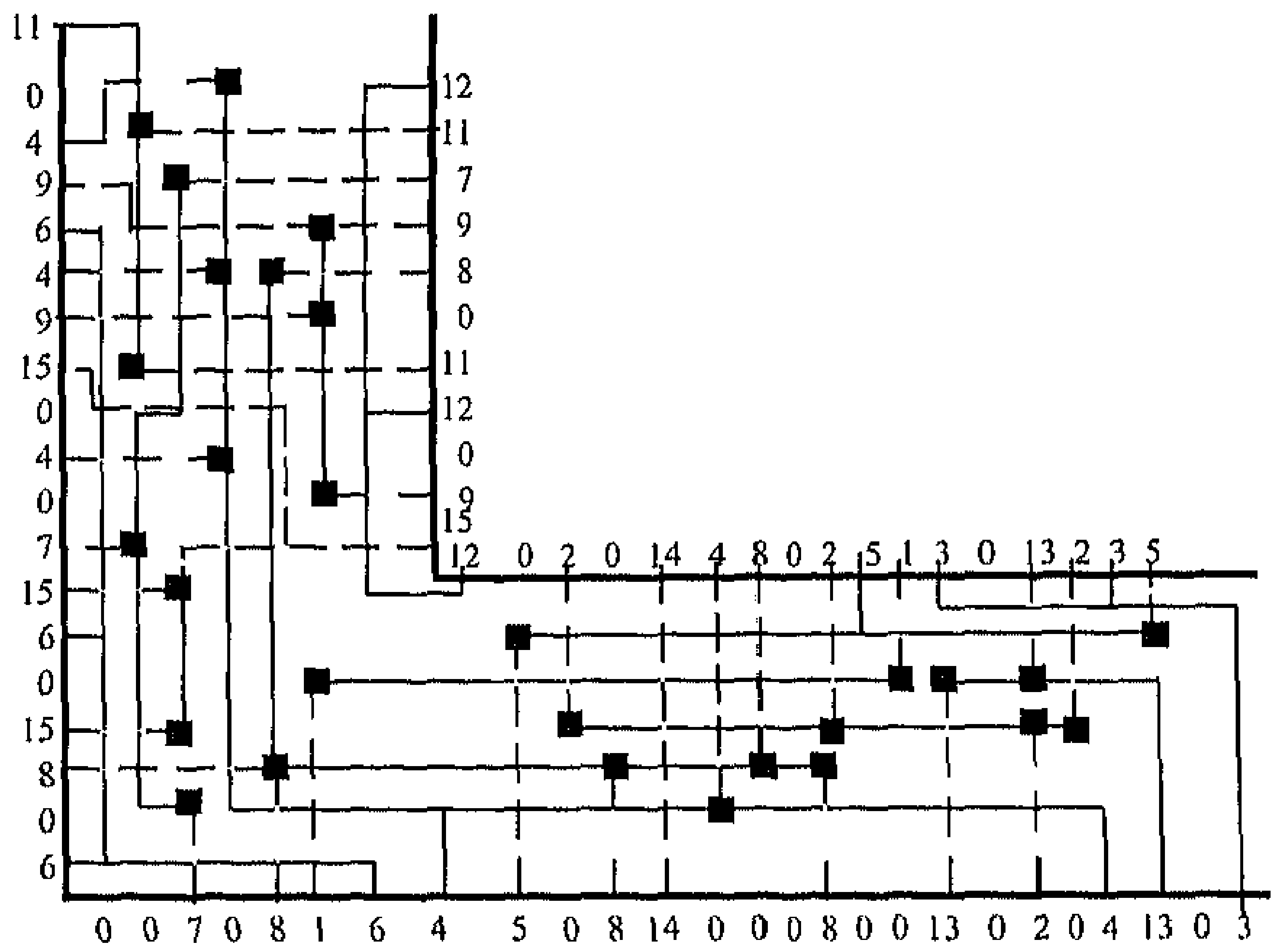
(a)



Example 2

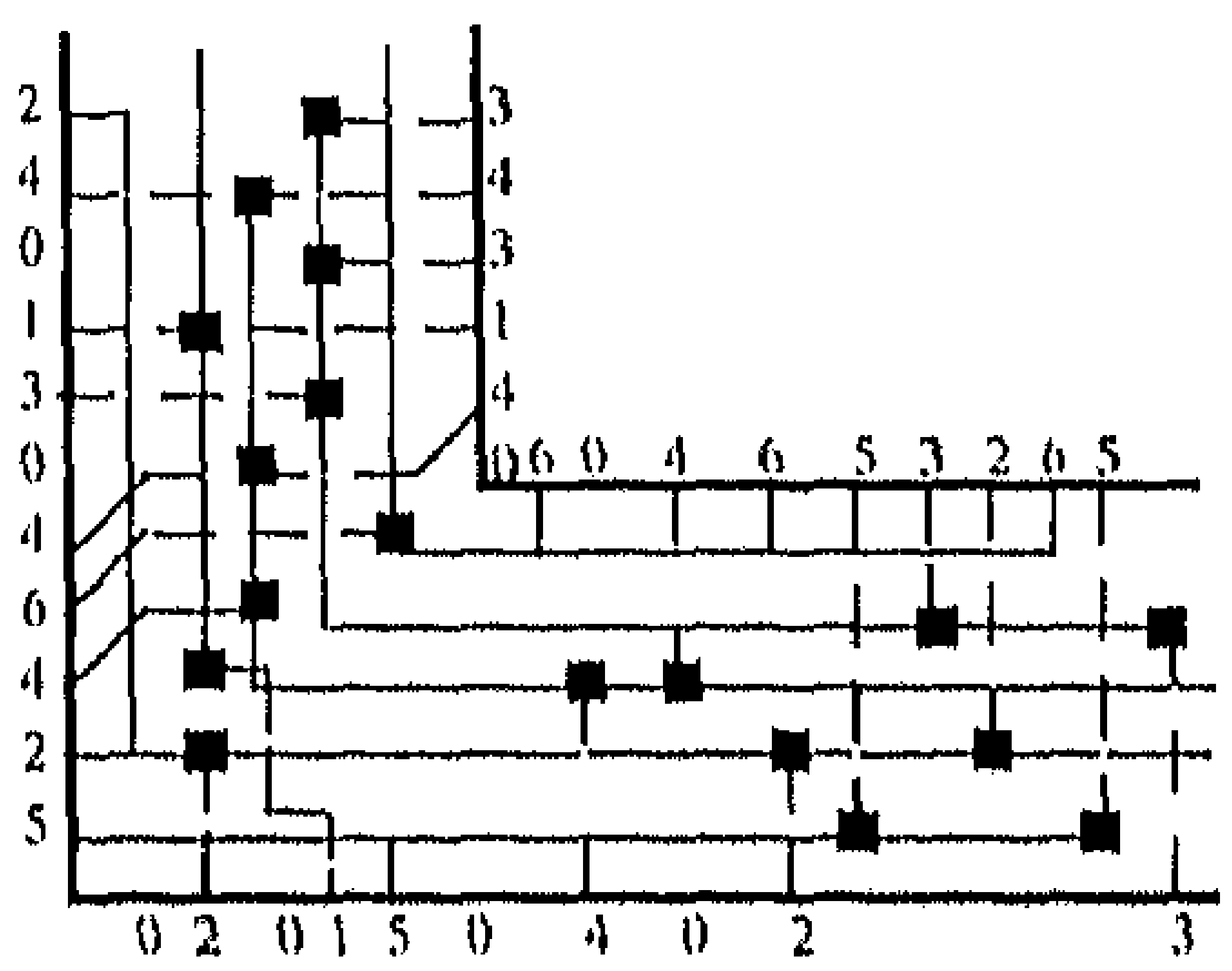
Fig. 6.7 : Routing produced by MD-L for four problem instances (contd.)

(b)



Example 3

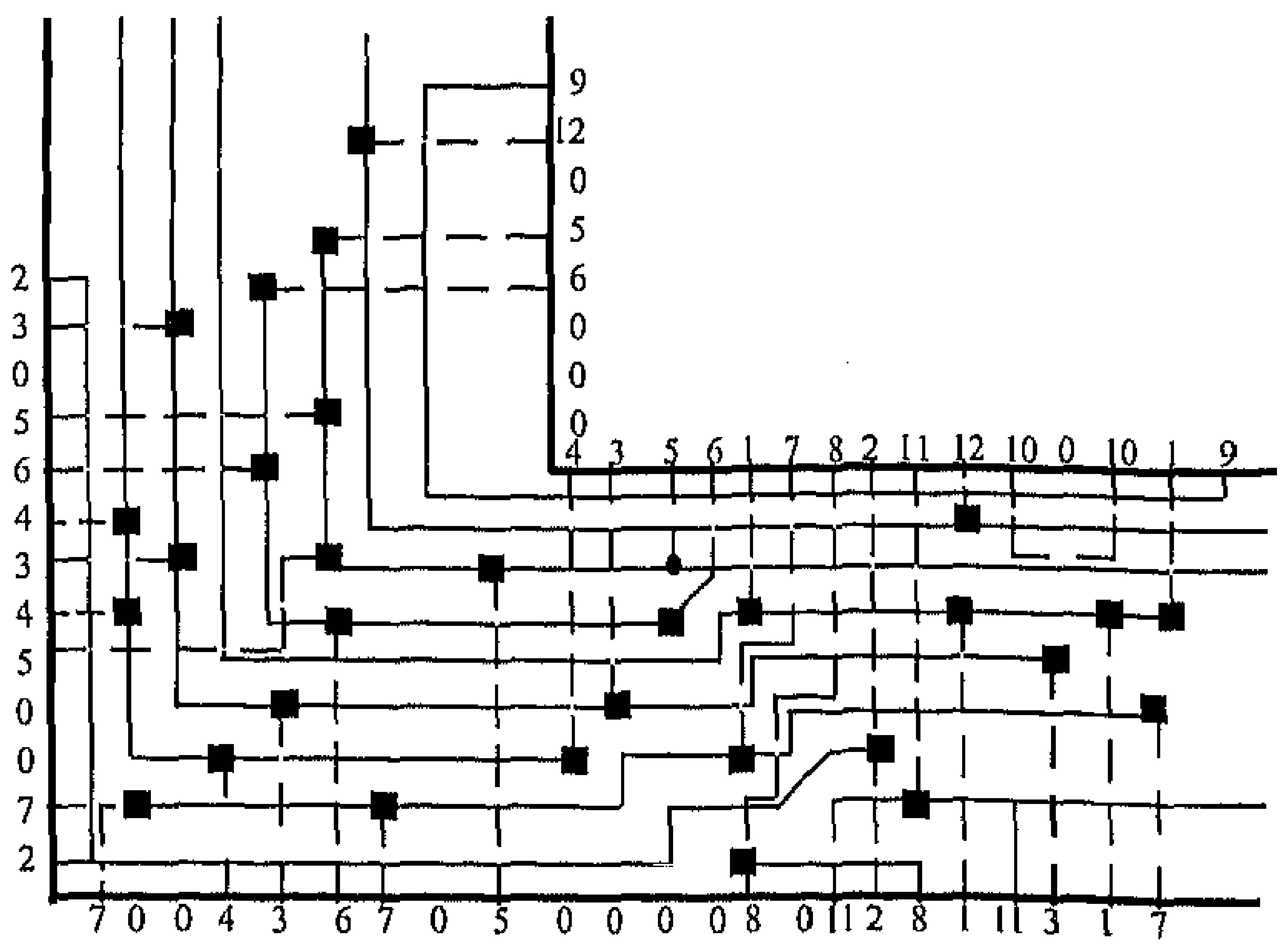
(c)



Example 4

Fig. 6.7 : Routing produced by MD-L for four problem instances (contd.)

(d)



Example 5

Fig. 6.7 : Routing produced by MD-L for four problem instances

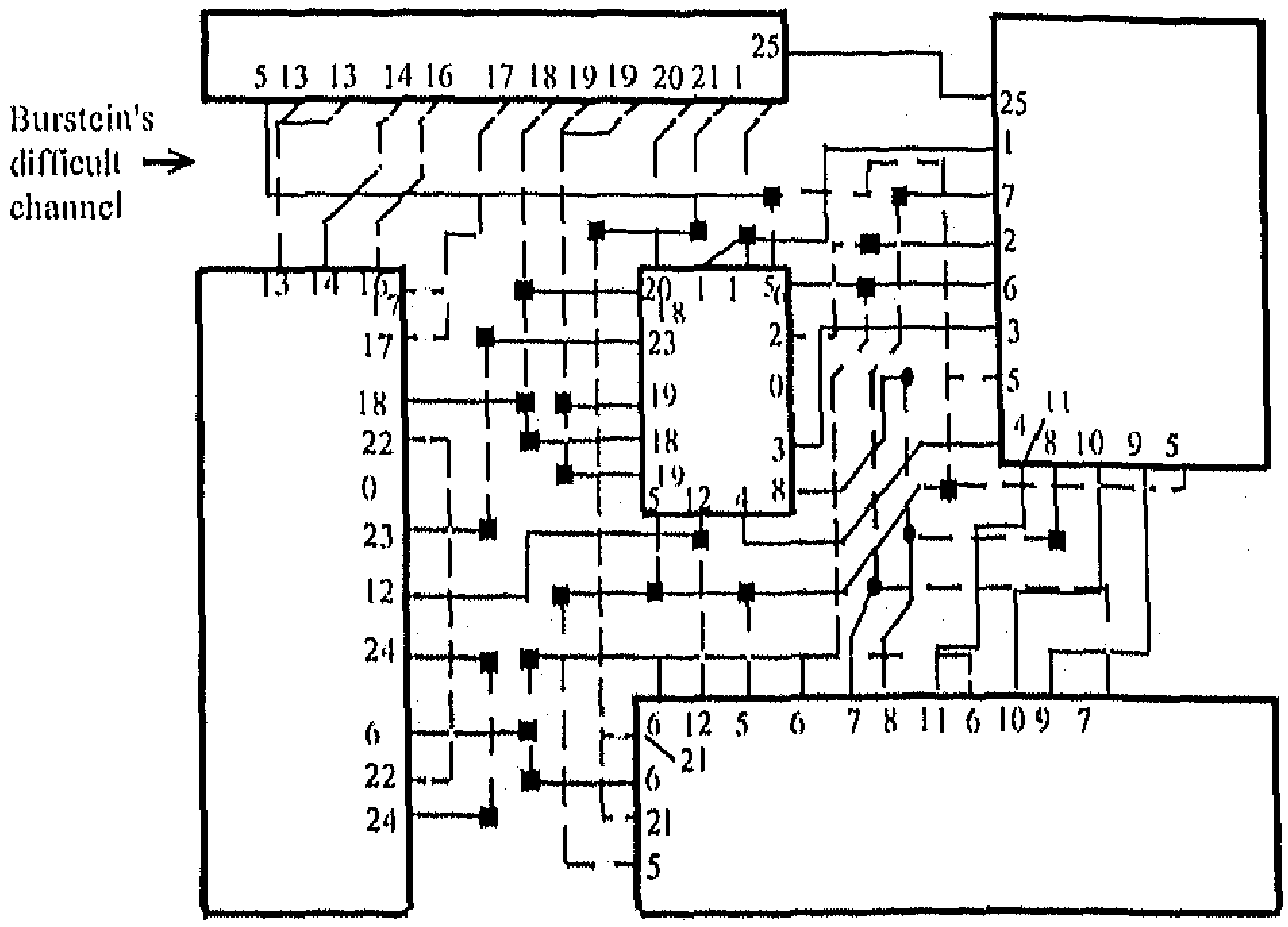


Fig. 6.8 : Example 6 routed by MD-L

Table 6.1 : Performance of our MD-L router

Example	diagonal wirelength		# tracks		# vias	
	in [C87]	MD-L	in [C87]	MD-L	in [C87]	MD-L
Ex. 1 (Fig. 6.7)	*	$9\sqrt{2}$	*	4	*	7
Ex. 2 (Fig. 6.8a)	*	$19\sqrt{2}$	*	8	*	32
Ex. 3 (Fig. 6.8b)	$220\sqrt{2}$	0	10	7	62	29
Ex. 4 (Fig. 6.8c)	$11\sqrt{2}$	$4\sqrt{2}$	8	5	36	18
Ex. 5 (Fig. 6.8d)	$130\sqrt{2}$	$2\sqrt{2}$	11	9	51	31
Ex. 6 (Fig. 6.9)	*	23	*	♣	*	28

* – not available; ♣ - not applicable

6.3 Routing in Switchboxes and Staircases

Some floorplans may have cross-junctions (Fig. 6.9a) where four channels meet. The routing in this junction region can be carried out by redefining the cross-junction as one L-channel and two straight channels. However, there may still be the need for handling switchboxes by themselves.

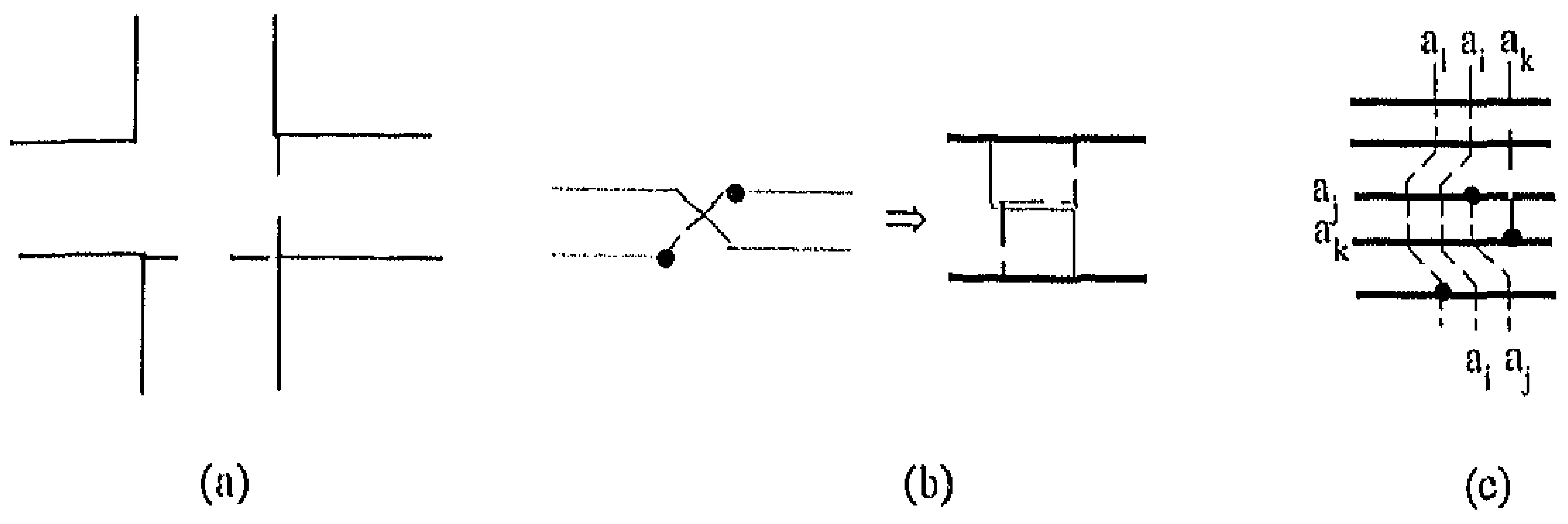


Fig. 6.9 : (a) Cross junction; (b) Swap routing, (c) Re-routing

6.3.1 Switchbox routing

In a switchbox problem, terminals are located on three or four sides of the rectangular channel. Given a problem instance, let $<l_1, l_2,...., l_k>$ and $<r_1, r_2,...., r_k>$ be the sequence of nets from top to bottom on the left and right sides respectively, and $<t_1, t_2, \ldots, t_m>$ and $<b_1, b_2, \ldots, b_m>$ the sequence of nets from left to right on the top and bottom sides respectively. The switchbox constraint graph, SCG is defined as a supergraph of the VCG [S93]. First, the VCG is formed using the terminals on top and bottom sides and then for each of the two sequences on the left and right sides a chain of directed arcs is added to obtain the SCG. An ordered sequence of nets for assigning tracks is determined whenever the SCG is acyclic. For cyclic SCG, two nets or terminals with conflicting order can be handled by swap (Fig. 6.9b) or X routing [CR91]. Algorithm for switchbox routing in two models are presented below.

Algorithm MD-SN (for acyclic SCG)

Input : A switchbox S of size $(k \times m)$, a netlist of n nets, and two routing layers;

Output: Reserved layer MD routing (horizontal and $\pm 45^0$ wires in one layer and vertical with $\pm 45^0$ wires in other layer).

Step 1. Construct a straight channel C as in the form

$$TOP = <l_1, l_2, \ldots, l_{k-1}, t_1, t_2, \ldots, t_m, r_1, r_2, \ldots, r_{k-1}> \text{ and}$$
$$BOT = <l_2, l_3, \ldots, l_k, b_1, b_2, \ldots, b_m, r_2, r_3, \ldots, r_k>.$$

Step 2. Route C by reserved layer MD algorithm.

Step 3. Extract the portion of the channel bounded within column i and column $(i+m-1)$ to obtain routing of the switchbox.

Remark : If the width w of C is greater than k, then the above router needs $(w-k)$ extra tracks for routing the switchbox. This may be improved by using a channel router in the unreserved layer MD model described in the previous chapter.

Algorithm MD-SO (for general SCG)

Input : A switchbox S of size $(k \times m)$, a netlist of n nets and two routing layers;

Output: Unreserved layer MD routing.

Step 1. Construct the SCG and remove the minimum number of edges to make it acyclic (Minimum feedback edge set problem) using a simple greedy heuristic. Topologically sort the nodes of the acyclic SCG and construct a linearly ordered sequence of nets $L = <a_1, a_2, \ldots, a_n>$.

Step 2. Start from the left side of the switchbox, and assign nets on horizontal tracks in layer 1, according to *L*. Update the count of unused horizontal tracks.

Step 3. Repeat steps (a) to (d) until all the columns are scanned from left to right.

(a) If two nets are in conflicting order, then readjust the net assignment on tracks by swapping them in adjacent tracks (Fig. 6.9b) [CR91] whenever possible.

(b) If a net terminates in column c_i then mark the respective horizontal track free, and update the count of unused tracks accordingly.

(c) If new nets appear at column c_i then free the track according to *L* by shifting nets onto adjacent tracks by 45^0 routing as in MD router inserting an extra track if needed.

(d) (i) Top and bottom terminals are routed in vertical columns on layer 2, failing which both layers and vias are used, else check whether rerouting is possible (Fig. 6.9c).

(ii) Failing (i) call a maze router with the given source-destination pair within the sub-grid bounded by c_u and c_i (using horizontal, vertical, and diagonal moves on both layers). c_u is the column where the maze router was called last; initially $u = 1$.

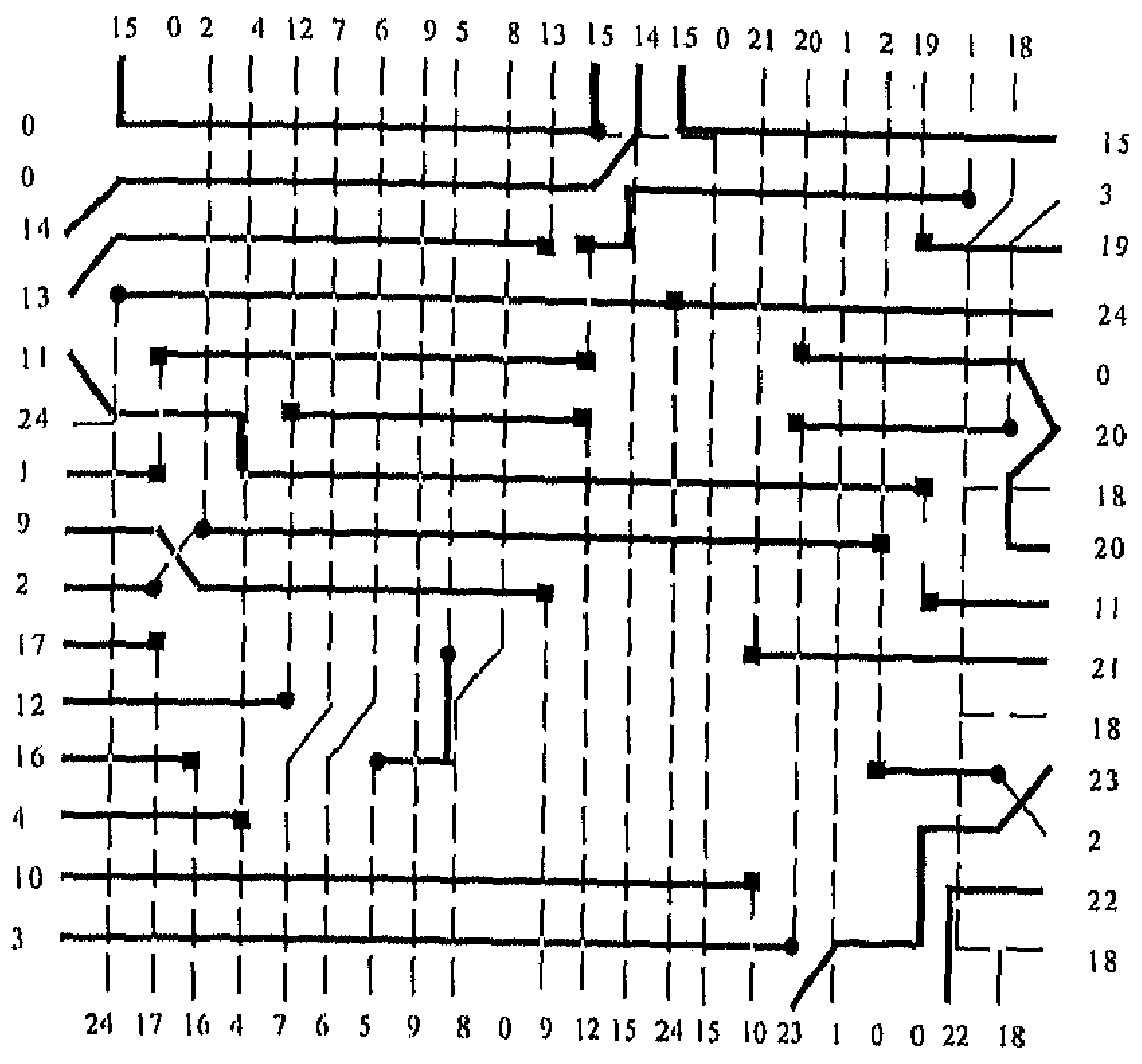
(iii) Failing (ii), add an extra track or column as required.

Step 4. If more than one terminal of a net, say *a*, appear on the left (right) boundary of the switchbox then use diagonal moves to reroute and push the existing vertical connections towards right (left) till the leftmost (rightmost) column is available for routing *a*. If the strategy fails, insert an extra column at the leftmost (rightmost) position.

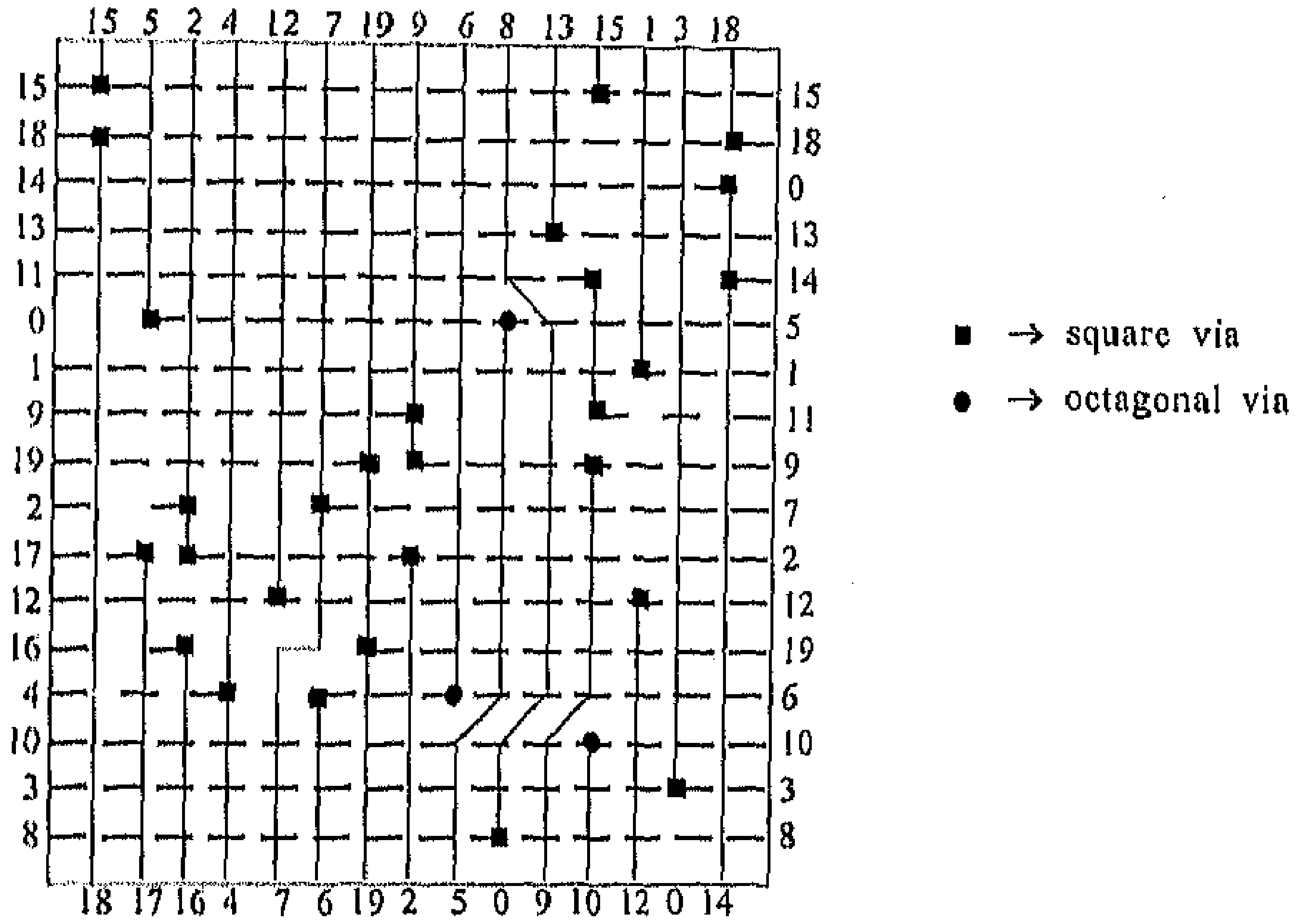
Complexity : The complexity of switchbox routing is $O(k \times m)$. It is worth pointing out that although the maze router may be called, each call is on a disjoint sub-grid.

6.3.1.1 Experimental results

Our algorithm routes Burstein's difficult switchbox and Dense switchbox [J86] without inserting any blank row or column (Figs. 6.10a, 6.10b). As the complexity



(a)



(b)

Fig. 6.10: (a) Routing of Burstein's difficult switchbox by MD-SO, (b) routing of Dense switchbox by MD-SN [J86].

is linear in the area, the algorithm terminates very fast. In both the benchmark examples, no call to the maze router (as in *Step 3(d)-(iii)* of Algorithm MD-SO) was necessary. The results are summarized in Table 6.2.

Table 6.2 : Performance of our MD-SO router

Switchbox Example	Number of		in [J86], # extra		in MD-SO, # extra	
	rows	columns	rows	columns	rows	columns
Burstein's (Fig. 6.10a)	15	22	1	0	0	0
Dense (Fig. 6.10b)	17	15	0	1	0	0

6.3.2 Staircase routing

A staircase channel is an isothetic rectilinear region monotonically increasing from one corner of the floorplan to its diagonally opposite corner (Fig. 6.11), with terminals along the two rectilinear boundaries except at the top and bottom ends. It was proven [DAK85] that for any floorplan there exists a channel definition which induces acyclic ordering of the staircase channels, thereby enabling addition of extra tracks if required without any rerouting within previously routed channels. A monotone staircase routing region is a generalization of L-channels, i.e., a series of L-channels. The algorithm to route such a staircase channel is similar to MD-L where the entire staircase is straightened and pin alignment for each sub-channel done to obtain minimum density over entire channel. However, if the density varies widely along the staircase, it should be subdivided into a series of smaller staircases where for each smaller staircase, the densities of the sub-channels are more or less uniform. There is scope for minor refinement of routing at a bend where the two sub-channels have unequal densities. At the bend (junction) where two consecutive smaller staircases with different densities meet, an instance of switchbox routing is created. This switchbox is then routed by one of the routers proposed in Section 6.3.1.

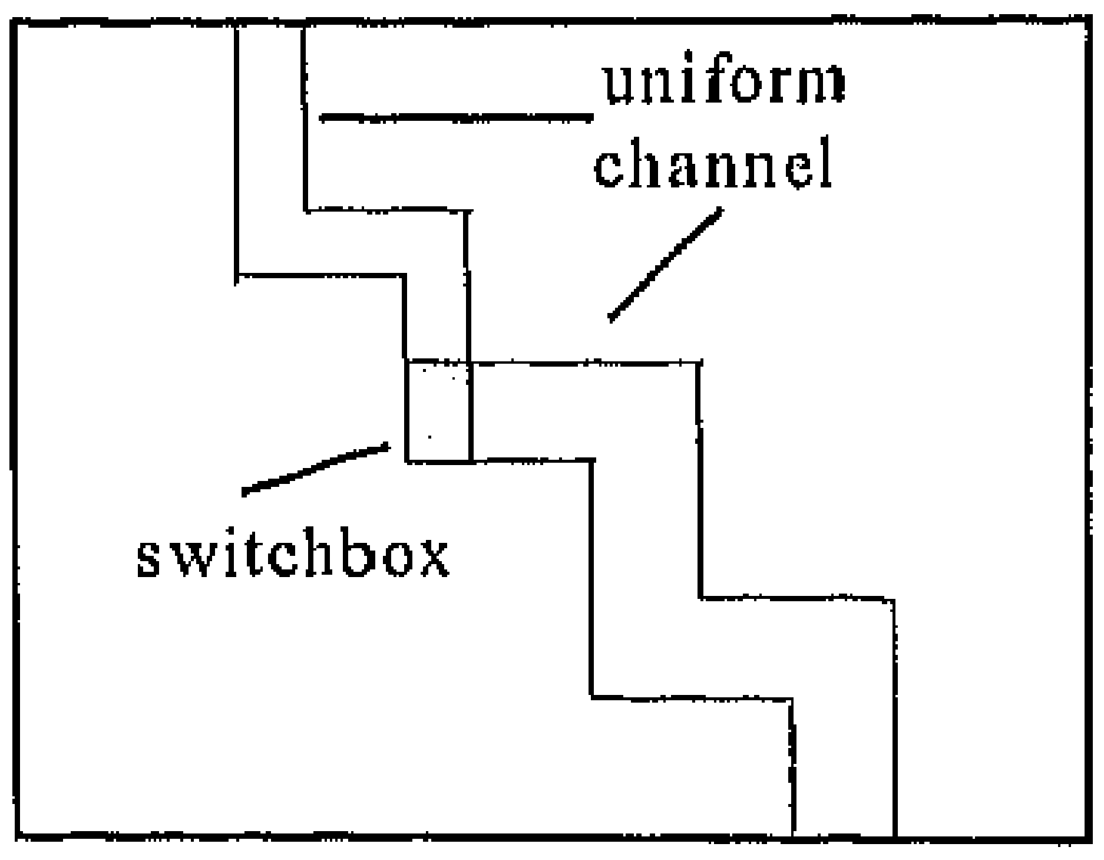


Fig. 6.11 : Routing a staircase channel.

6.4 Conclusion

We have proposed a routing scheme based on Manhattan-diagonal model for L-channels, switchboxes, and staircases. These general channel definitions always guarantee a safe routing order even in nonslicing floorplans, and make the router useful for MCMs. Our algorithms are simple, terminate in time linear in the routing area and provide elegant solutions to complex routing problem in a general floorplan.

Chapter 7
Via Minimization in MCM Routing

7.1 Introduction

With rapid decrease of feature size, efficiently packing multiple circuit modules in a single chip has become a promising area of research. Using MCM technology a large number of functional blocks can be mounted and interconnected on a single substrate. The numbers of distinct nets and terminals are usually very high. A typical MCM may consist of 7000 nets on a grid size of over 2000 x 2000 [KC92]. With the emergence of deep sub-micron technology, the numbers of chips, nets, and terminals per net in an MCM, are likely to increase in the future.

Inside an MCM, the circuit components are mounted on the top layer; other layers are used for pin redistribution and routing. As many layers are usually available for MCM routing, the main objective is to improve performance, i.e., to achieve tight packaging with lower fabrication cost, satisfying the constraints on net length, net separation, via size, via separation, etc. Thus, the primary goal is to minimize the number of vias, wire length, cross talks, delay in signal propagation, with the secondary objective of reducing the number of routing layers in the MCM.

In this chapter, we report an efficient method of MCM routing. For high performance routing, one has to minimize the number of vias and wire length. To minimize the number of vias, we first route the nets having more terminals, through the layers that are closer to the top, such that a large number of nets is routed with short pieces of wires. To take care of the delay factor, we also route the farthest terminals of a net through the layers closer to the top, and explore subsequent layers if they are unavailable. Our algorithm consists of three phases. First, we preprocess the nets to determine a routing order. We estimate the cost of a feasible wiring for each net by constructing a minimum-bend rectilinear steiner tree among its terminals, avoiding the obstacles on the concerned layer. The wire length and the number of terminals of the net are used to define a metric called *average path length*. The ordering of nets for actual routing in the next phase depends on the average path length. In order to reduce delay for some important nets, one may need to change the order. In the next phase, actual routing is done assuming a reserved layer model. Nets are routed following the order as obtained in the first phase. After the completion of routing in the current layer-pair, the unrouted terminals are projected in the next layer-pair and the same routing technique is adopted keeping the order of nets invariant. This process continues until all the terminals are routed. Our technique uses significantly less number of vias, which can be further reduced in the third phase, where we improve the routing by flipping and other techniques.

Our preprocessing phase efficiently guides the selection of nets so that the number of layers is reduced. As the splitting of multi-terminal nets into a set of two-terminal nets usually increases the problem size [KC92, KC95], we consider all the terminals of a net at a time. The restriction of a fixed number of vias as proposed in [CL91, KC95], reduces the flexibility of connecting the terminals of a net, which in turn, may lead to increased number of layers and stack vias. The proposed algorithm is run on several benchmarks whose test results suggest that it outperforms the solutions obtained in [CRN97, KC92, KC95] with respect to via count. Total wire length also compares well with those in [CRN97, KC95].

7.2 Preliminaries

The MCM routing problem consists of a rectangular multi-layer routing substrate, a set of rectangular circuit modules (dies), and a set of nets. Dies are mounted on the top layer of the routing substrate, and the other layers are used for pin redistribution and routing. The die terminals are attached to the substrate using some bonding technologies. The objective is to interconnect all the terminals of same signal net such that there is no overlap of wiring with that of the other nets in the same layer.

Definition : *3-D routing space* is a three-dimensional space divided into a number of layers parallel to the XY-plane. The layers are numbered from top to bottom, and the top layer is marked as *layer-0*. Each layer consists of a finite number of grid points generated by a set of horizontal and vertical tracks. A grid point is identified by (x, y, z) in the 3D space, where x, y coordinates have their usual meaning, and the z coordinate indicates the layer identification.

A line parallel to the X-axis (Y-axis) is referred to as horizontal (vertical) line.

A *net* consists of a set of terminals t_1, t_2, ..., t_k attached to the boundaries of circuit modules, which are to be interconnected.

The rectangular circuit modules are arranged on the substrate in the form of a K x L mesh of rectangles on *layer-0* of the 3D routing space. A pin redistribution layer is often needed when the spacing between the pins does not satisfy the design rule for routing. The grid lines are assumed to be equi-spaced and the spacing is determined by the routing pitch for the given technology. In our discussion we don't consider pin redistribution layer, as we assume that the pad spacing (spacing between the die-terminals) are satisfying the design rule for routing. A scheme of our model is shown in Fig. 7.1. We assume that the circuit modules are identical in shape and size, and terminals are equi-spaced, and no wire is allowed to pass through them. The terminals marked with '0' represent blank terminals.

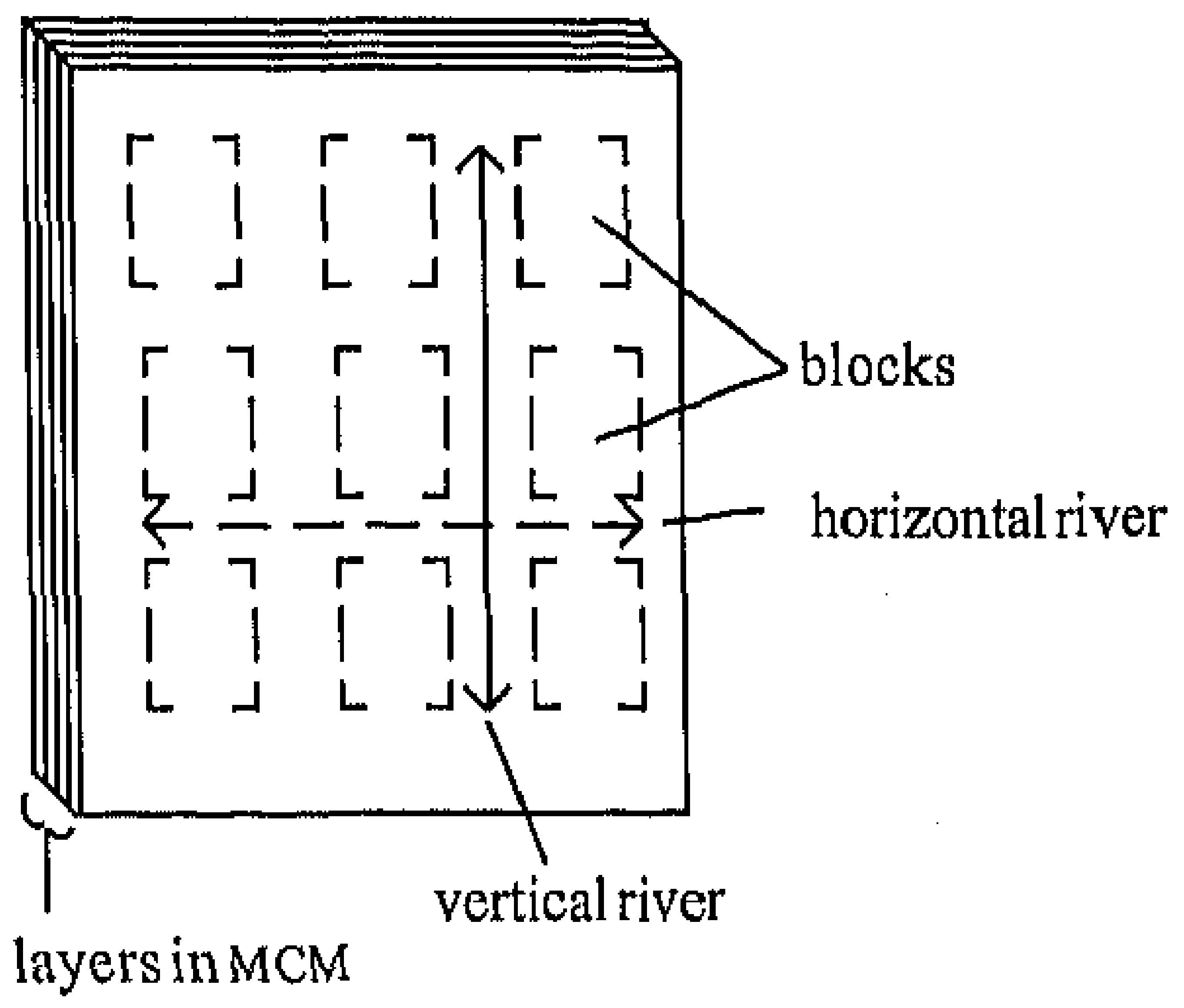


Fig. 7.1 : Schematic description of an MCM

The router routes two adjacent layers at a time. Usually, the vertical and the horizontal connections are made in two different layers (reserved layer model). But, in our model we allow some exceptions for layer assignment to reduce the number of vias. Fig. 7.2 demonstrates the classification of different types of vias. An *interconnection via* or simply *via* is a hole through which a connection is made to the adjacent layer. Vias may be stacked on top of each other to form a *stack via* for connecting wires in non-adjacent layers. A stack via contributes k to *via* *count* if it penetrates k layers. In our model, the stack vias are placed at the original terminal positions of the nets that in *layer-0*. The vias used for propagating the die terminals to the first routing layer are called *distribution via*. These vias do not contribute to via-count.

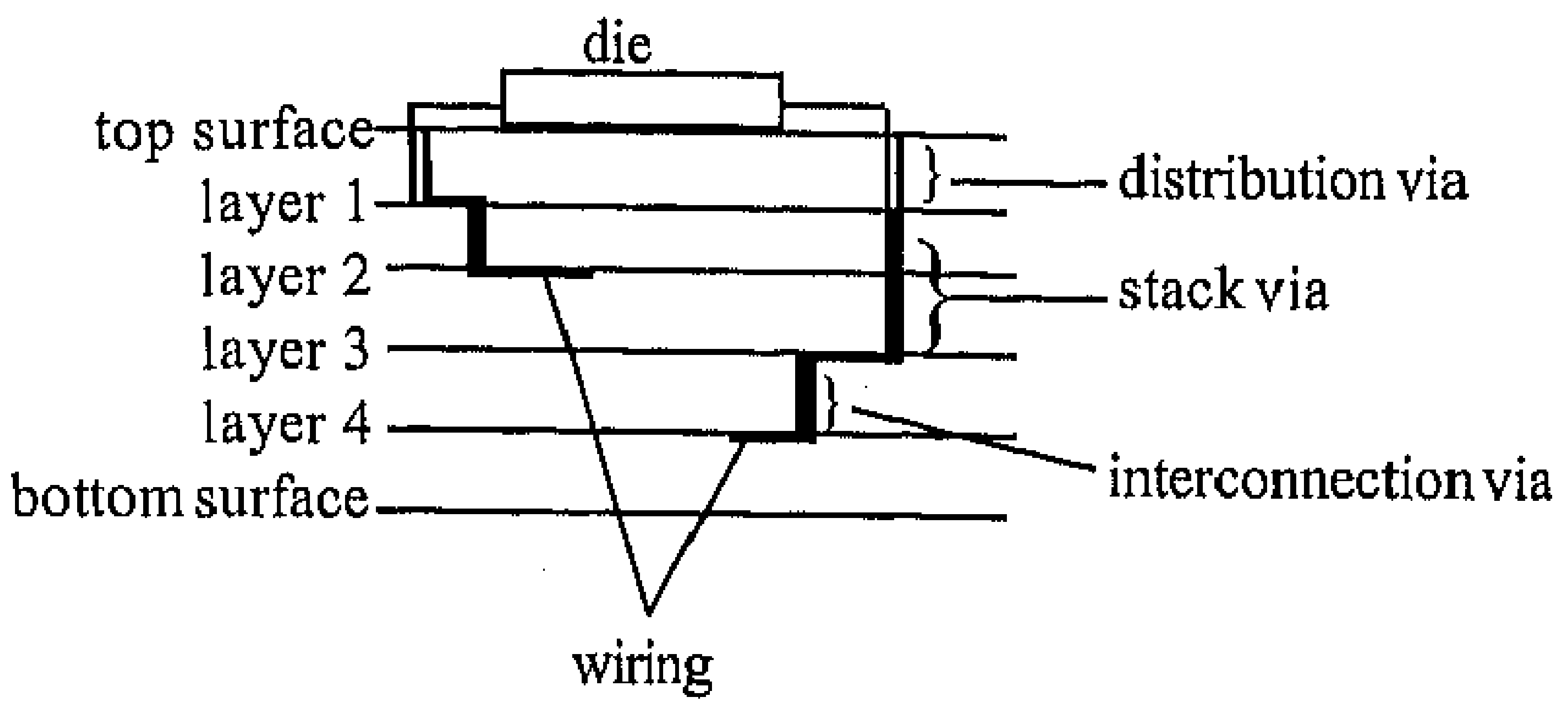


Fig. 7.2: Classification of vias

Definition : The horizontal (vertical) routing channel between each pair of consecutive rows (columns) of blocks, is termed as *X-river (Y-river)*. In a K x L mesh of blocks we denote the X-rivers as R_1, R_2, ..., R_{K+1}, and the Y-rivers as C_1, C_2,...., C_{L+1}.

If the four sides of a block are projected on a routing layer, the rectangular space bounded by these lines is termed as *space inside the block*. All X- and Y-rivers as well as the space inside the blocks are used for routing. A routing wire may move from the space inside any block to its adjacent river through a blank terminal. Needless to say, no wiring can pass through occupied terminals.

A trivial lower bound on the number of layers can be obtained by considering *2L* vertical lines, each of which is aligned to a boundary of a Y-river. Let κ_i be the number of nets whose terminals appear in both the sides of the *i*-th line, and η_i be the number of pin positions on this line, who are occupied by some net(s) appearing in either side of this line. Then the number of nets whose wiring pass through this line, considering all the layers, is $\kappa_i + \eta_i$. Let $\chi = \mathrm{Max}_{i=1}^{L} (\kappa_i + \eta_i)$, i.e., the maximum of this quantity considering all such vertical lines. Thus, the lower bound on the number of layer-pairs is χ/H, where H is the number of horizontal tracks available in the given substrate. A similar lower bound on the number of layer-pairs may be obtained considering the *2K* horizontal lines aligned with the boundaries of the X-rivers. The maximum of these two quantities would give a lower bound on the number of layer-pairs required for routing the given instance.

In Fig. 7.3, we show an instance of MCM routing with a given area of cross-section. The problem is shown to be unroutable even when the number of available layers is infinite. Thus, we observe that an increase in the number of layers is not always sufficient for the completion of MCM routing. Layer area may sometimes need to be increased to achieve routability.

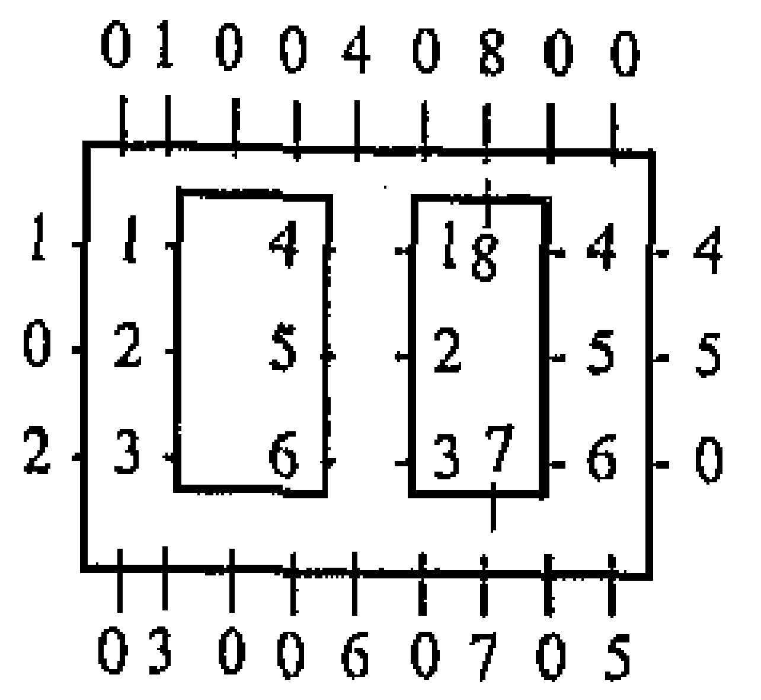


Fig. 7.3 : Example of an unroutable MCM

7.3 Proposed Routing Technique

Let $\alpha_1, \alpha_2, \ldots, \alpha_m$ be m distinct nets whose terminals are attached along the boundaries of different blocks as well as the floor. The number of terminals for net α_i is n_i, where $n_i \geq 2$ for all i = 1,..., m. Initially, the terminals of all the nets are available at *layer-0* along the boundaries of the blocks. These terminals are propagated to their appropriate routing layers using stack vias. In the following subsections we describe our three-phase algorithm.

7.3.1 Ordering of nets

In this phase, we find a feasible path for connecting terminals of each net considering the blocks as obstacles. For each net α_i, we construct RST_i a rectilinear steiner tree connecting all the terminals of α_i is considered. The edges of RST_i are directed through the rivers and indicate a feasible path for routing. In reserved layer model any junction (bend) of a horizontal and a vertical edge of RST_i causes a via. Thus, to minimize via count in a layer-pair, we need to construct a RST with minimum number of bends. We compute the total path length by summing over the lengths of all the edges of RST_i. Next, *average path length* for α_i is calculated by dividing the total path length by n_i. Although many algorithms for finding RST are available in the literature [CSW92, CG96], to meet our special constraints, we use a simple heuristic scheme for obtaining a RST with minimum number of bends through the rivers.

Let $b_1, b_2, \ldots, b_k$ be a set of blocks where the net α_i appears. For each block, position(s) of the terminals of α_i is known. For each X-river R_j, its adjacent blocks containing the net α_i are considered. These terminals of α_i may be connected through R_j using one via or three vias. In Fig. 7.4 we find that a terminal of net α_i in block B_{11} is connected with river R_j using three vias, whereas that of the block B_{21} is connected with river R_j using a single via. We consider only the terminals of α_i adjacent to R_j, such that they may be connected to the wire passing through R_j using exactly one via. The terminals attached to the vertical boundaries of the blocks will be considered similarly. The segment of R_j in which all the terminals of α_i are spanning, is denoted as the *horizontal span* of R_j; its weight is determined by the number of terminals of α_i adjacent to R_j. The same technique is adopted to calculate the *vertical spans* for all the Y-rivers and their weights with respect to each net α_i. We use a plane sweep technique to interconnect these spans to obtain a RST.

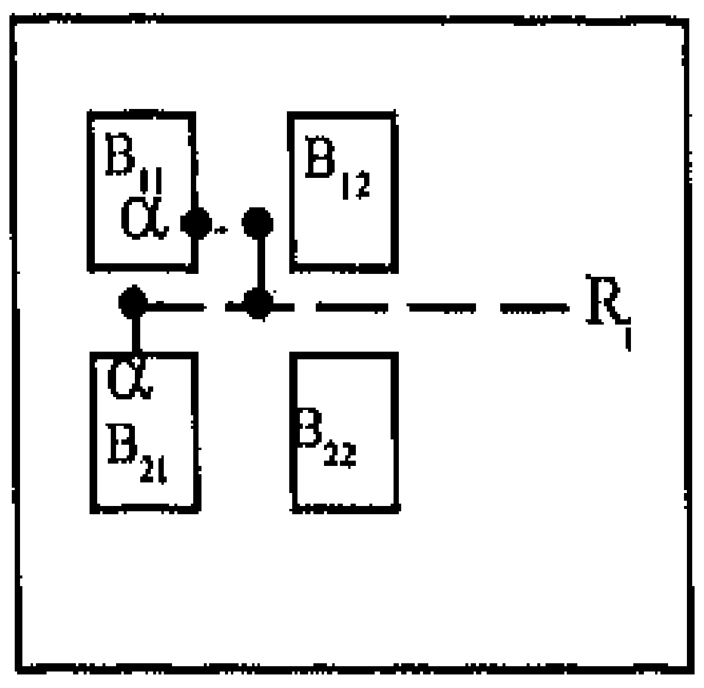


Fig. 7.4 : Connecting a terminal to its nearest horizontal river in reserved layer model

The algorithm Find_Path generates a RST among the terminals of each net considering the blocks as obstacles, using minimum number of bends (crossing point of horizontal and vertical wires).

It first finds the set of all horizontal and vertical spans in $O(n_i \log n_i) + \theta_i$ time, where n_i and θ_i are respectively the number of terminals of α_i and the total number of horizontal and vertical spans for α_i. It then finds RST_i in $O(\theta_i)$ time. As $\theta_i \leq n_i+1$ for all i, the overall complexity is $O(n \log n)$, where $n = \Sigma_{i=1}^{m} n_i$.

```
Algorithm Find_Path($\alpha_i$)

Data Structure :Set of horizontal spans for $\alpha$, whose each member is associated
                    with a *mark* field,
                A set of vertical spans for $\alpha_i$,
                *LIST1* : a temporary list containing the horizontal spans encountered
                    during line sweep, and which are not yet connected with a
                    vertical span.
                *LIST2* : a similar list of vertical spans which are not yet connected
                    with a horizontal span.

Step 1 :  Compute all the horizontal and vertical spans and their weights for net $\alpha_i$;
Step 2 :  Set the *mark* field of all the horizontal spans to 0;
Step 3 :  Sort vertical spans and the left end points of the horizontal spans in
              increasing order of their $x$-coordinates;
Step 4 :  Initialize *LIST1* and *LIST2* to NULL;
Step 5 :  Process the elements of the sorted list from left to right;
        /* For each element, any of the following three distinct situations may arise *\
          if a left end point of a horizontal span is encountered, then
              the current horizontal span is entered in a *LIST1*;
              if *LIST2* is non-empty, then
                  the current horizontal span is extended towards left so that
                      all the members of *LIST2* can be projected on it;
                  all the members of *LIST2* are removed;
          if a vertical span is encountered, following two situations may take place.
            • if *LIST1* is non-empty, then
                the right end points of all the horizontal spans in *LIST1* are
                    extended (if necessary) upto the $x$-coordinate of the vertical
                    span;
                the current vertical span is extended in both sides (if necessary)
                    to cross all the horizontal spans present in *LIST1*;
                all the members in *LIST1* are removed keeping a single one arbi-
                    trarily by setting its mark field to 1.
            • if *LIST1* is empty, then
                      /* No horizontal span is available on which current
                          vertical span can be projected */
                  enter the current vertical span in *LIST2*;
```

Step 6 : finally, after processing all the events,

- if the number of elements *LIST1* is more than one, then left coordinates of all unmarked members in *LIST1* are projected to the nearest vertical span, if at least one such exists; else, a vertical line is drawn through an Y-river to cut the members in *LIST1*;
- if the number of elements *LIST2* is more than one, then a vertical line through an X-river is drawn to cut all the members in *LIST2*;

Theorem 7.1: The algorithm Find_Path generates a rectilinear steiner tree among the terminals of each net considering the blocks as obstacles, using minimum number of junctions, where the junctions are the crossing point of horizontal and vertical wires.

Proof : The rectilinear steiner tree among the terminals of a net is drawn through rivers. Note that, for the projection of each terminal on the river, one junction is always required. In addition, junctions are created at the steiner vertices. We now show that the number of steiner vertices created by this algorithm is minimum. We have got a minimum number of segments, each of them is lying in different river and spanning all the terminals appearing on the boundary of the river. In our algorithm, if there exists θ segments of which at least one is a horizontal segment and at least one is a vertical segment, we create θ - 1 steiner vertices, which is optimum since in a tree of θ nodes, there is θ - 1 edges to connect them. In case, if all the θ segments are parallel, we need exactly θ steiner vertices. This is also optimal since in order to connect them, a line passing through any river which is orthogonal to these segments are required, which creates $\theta + 1$ segments in total.

After finding a feasible path for connecting the terminals of a net α_i (corresponding to the RST among its terminals), its length Δ_i, is obtained. The average path length for the net α_i is obtained as Δ_i / n_i. As an output of this phase, a sorted list of the average path length of all the nets is generated. This guides the selection of nets in the routing phase.

7.3.2 Routing of nets

In the actual routing phase, nets are selected in increasing order of their average path length. We may accommodate more nets with smaller average path length compare to the larger one on topmost lair pairs and consequently it is possible to reduce the number of stack vias. Once a net is selected all its terminals are connected before the selection of next net for routing. If a net appears more than once in a particular block, its terminals are routed inside the block as described in the subsection 7.3.2.1. If a net appears in more than one block which belongs to same row or same column, an attempt is made to connect them using a path through the blocks and avoiding the rivers. The objective is to reduce the congestion of the river, and effectively use the space in the routing layers inside the block boundaries. The method is illustrated in subsection 7.3.2.2. The case where a net has more than two terminals, each of them is present in different blocks, and no pair of these blocks appears in the same row or in same column, is discussed in subsection 7.3.2.3.

In the reserved two-layer routing model, vertical and horizontal routing wires pass through different layers of a layer-pair, say *top layer* and *bottom layer* respectively. The lines passing through the top (bottom) layer of this layer-pair are denoted by solid (dashed) lines. In the unreserved layer model, both the horizontal and the vertical lines may be present in any layer. In our MCM routing model, an even number of routing layers are available. Always a layer-pair is considered for routing the unrouted nets, and their terminals are propagated to the appropriate routing layers using stack vias. We follow the convention of representing the wires of top and bottom layers of a layer-pair by using solid and dashed lines. But we allow restricted deviation from reserved layer assignment in our routing strategy.

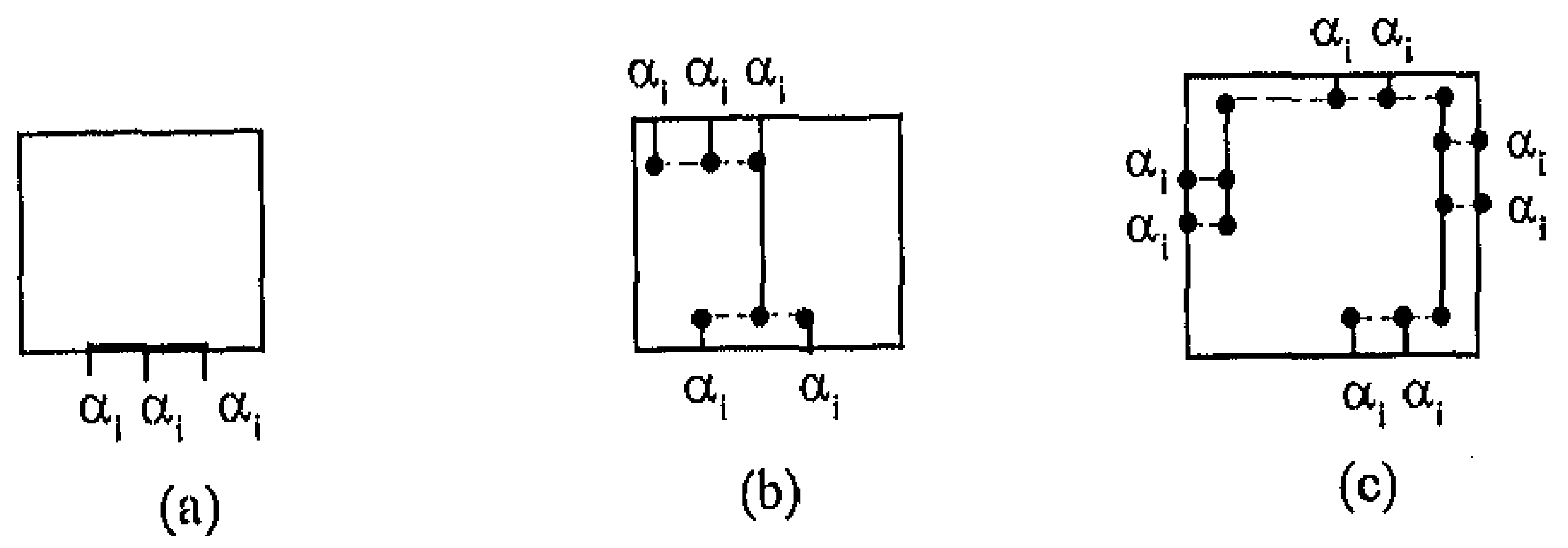


Fig. 7.5: Routing inside a block in reserved layer model

7.3.2.1 Routing inside a block

Consider the set of terminals of a net α_i appearing on the boundary of a block. While routing them in a particular layer pair, the following situations may arise.

Case 1: If the terminals of α_i appear in the consecutive pin positions, routing may be done inside the block by drawing solid line on the top layer over the span of terminals through the track adjacent to the boundary of the block. The terminals are then projected on this routing line using solid lines in the top layer without using any via (see Fig. 7.5a).

Case 2 : If terminals of α_i are attached to a particular side of the block, a line is drawn over the span of terminals through an available track inside the block. This line will be referred as base line; it may be solid or dashed depending on whether this line is a vertical or a horizontal. But unlike Case 1, here the terminals are connected with the base line in different layer and by using vias.

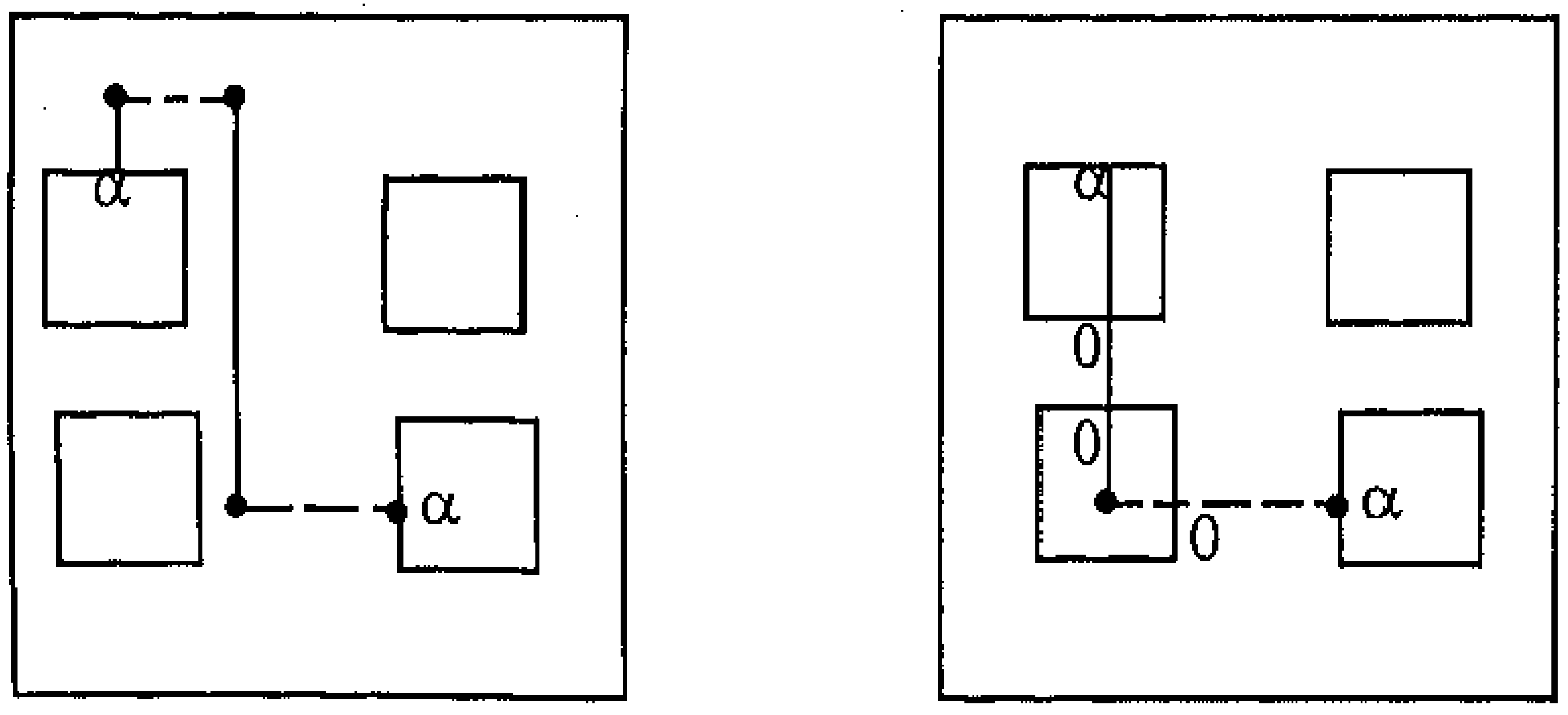
Case 3 : If terminals of α_i are attached to adjacent sides of the block, then those attached to each side are connected independently using the method described earlier. Finally, the base lines are connected by extending them to appropriate sides.

Case 4 : If the terminals are attached to the opposite sides of the block (Fig. 7.5b), those attached to each side are connected independently. Now the base lines of the opposite sides are connected through a track orthogonal to those. The choice of such a track should be such that it requires minimum number of vias. So, first of all, a track is searched such that its both ends coincide with the terminals of α_i in opposite sides. In this case, no extra via is required. The next option is to choose a track whose one end coincides with a terminal of α_i. Such a connection consumes an extra via. If both these attempts fail, we choose an arbitrary track for joining the base lines using two extra vias, provided such a track exists. The next option is to connect them through rivers, otherwise, they will be connected using the next layer-pair.

Case 5: If terminals of α_i are attached to all the four sides of the block, actions taken in earlier cases are judiciously used to handle this situation (Fig. 7.5c).

7.3.2.2 Routing of two-terminal nets

In order to route terminals of a net belonging to different blocks, the traditional way is to connect them through rivers as discussed in section 7.3.1. But Fig. 7.6a demonstrates that such connections need more vias. Thus an attempt is made to locate the routing paths for the two terminal nets through a series of blocks, crossing the rivers as shown in Fig. 7.6b. The routing wires use blank terminals to enter the interior of a block from its adjacent river, or vice versa. In order to obtain such an wiring, we need to store the blank terminals in an appropriate data structure as discussed below. A heuristic scheme is then suggested for locating a path through the blocks.



(a) (b)

Fig. 7.6: Routing a two terminal net (a) through a river; (b) through a block in reserved layer model

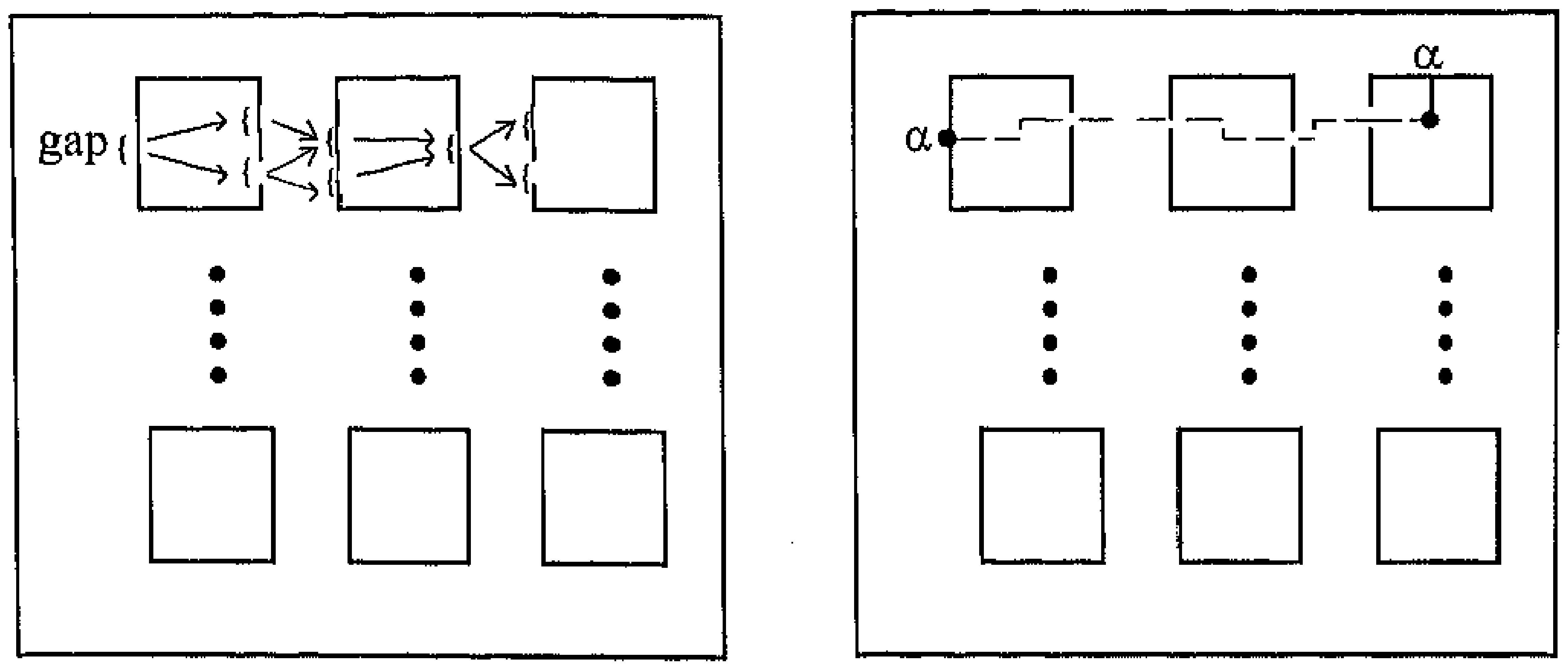
A successful routing utilizes the routing space inside blocks and keeps room for routing other nets along the river. If such an attempt fails, we try to route them through rivers by following the path guided by the RST, failing which, they are projected on the next layer-pair for routing.

Data structure

Consider the projections of unrouted terminals on the current layer-pair. A set of terminals (corresponding to different nets) appearing consecutively along a vertical/ horizontal boundary of a block may be considered as an obstacle for routing, and a set of consecutive blank terminals may be termed as *gap*. The blocks arranged in the K x L mesh are named as $\{B_{i,j}, i = 1,..., K \text{ and } j = 1, ..., L\}$. This configuration generates 2K *block boundary rows* (BBR), indexed as $BBR_1, BBR_2, ..., BBR_{2K}$, and 2L *block boundary columns* (BBC), indexed as $BBC_1, BBC_2, ..., BBC_{2L}$. We use the following data structure for storing the vertical gaps. The horizontal gaps are also stored similarly.

For each BBC_i, the vertical gaps are stored in a height-balanced tree. A vertical gap g ($\in BBC_i$) is represented as an interval $[y_{g1}, y_{g2}]$ ($y_{g1} > y_{g2}$), corresponding to y-coordinates of the two ends of the gap, and points to two linked lists of gaps visible to g, as explained below.

Let $g[y_{g1}, y_{g2}] \in BBC_i$. Its right pointer points to an ordered linked list containing (i) the set of gaps in BBC_{i+1} which are horizontally visible to g, and (ii) two more gaps g′ and g″ of BBC_{i+1} which are vertically just above and just below g respectively, i.e., $y_{g2} > y_{g'1}$ and $y_{g''2} > y_{g1}$. Note that g′ and g cannot belong to different sides of a horizontal river. Thus, we may have g′ = NULL. Similarly, g″ should also have the same property. Fig. 7.7 illustrates the elements of the linked list pointed by the right pointer of the gap g. The left pointer of g points to such a link list containing a set of gaps in BBC_{i-1}.



(a) (b)

Fig. 7.7: Demonstration of the data structure :
(a) gaps; (b) path through blocks

Path location algorithm

Consider two terminals $a(x_a, y_a)$ and $b(x_b, y_b)$ of a net α_i appearing in two different blocks. In order to connect them through blocks we consider the following situations separately.

Case 1: a, b $\in BBC_j$. Here a and b appear on the same horizontal side (top or bottom) of two blocks $B_{\lceil i/2\rceil j}$ and $B_{\lceil i/2\rceil ,l}$ respectively. A feasible routing between these two terminals is done by drawing a horizontal line through the gaps nearer to BBC_j of all blocks $B_{\theta,\lceil j/2\rceil}$, θ = i, ...,k. Finally, a and b is to be connected to this horizontal line (see Fig. 7.8a).

Case 2: a, b $\in BBR_l$. Here a and b appear on the same vertical side (left or right) of two blocks $B_{i,\lceil j/2\rceil}$ and $B_{k,\lceil j/2\rceil}$ respectively. A similar technique as in Case 1 is followed to get a feasible routing path between these two terminals.

Case 3: a $\in BBC_j$ and b $\in BBC_l$, $j \neq l$. Here a appears on a vertical side of block $B_{i,\lceil j/2 \rceil}$ and b appears on a vertical side of block $B_{k,\lceil l/2 \rceil}$ (see Fig. 7.8b). For the sake of simplicity, let us assume $x_a < x_b$. Now the following steps are executed to get a path for connecting these two terminals through the blocks.

Step 1 : A binary search on the list of gaps of BBC_{j+1} locates a gap g which is horizontally visible to a.

Step 2 : Follow the right pointers to the list of gaps visible to g to reach a gap on BBC_{j+2}.

Step 3 : Apply the same process to reach inside a block $B_{i,\theta}$, $\lceil j/2 \rceil \leq \theta \leq \lceil l/2 \rceil$.

Step 4 : A similar procedure is performed to find a horizontal path from b to the block $B_{k,\theta}$.

Step 5 : The vertical connection between these two horizontal paths may be done through blocks by considering the gaps on the horizontal boundaries of the blocks appearing on the θ-th column of the mesh.

Step 6 : If the search in Step 5 fails, the connection among these two horizontal paths is made through a vertical river.

Case 4: a $\in BBR_i$ and b $\in BBR_k$, $i \neq k$. Here a appears on a horizontal side of block $B_{\lceil i/2 \rceil, j}$ and b appears on a horizontal side of block $B_{\lceil k/2 \rceil, l}$. A similar procedure as stated in Case 3 may be followed to get a feasible routing in this case.

Case 5: a $\in BBR_i$ and b $\in BBC_l$. Here a appears on a horizontal side of block $B_{\lceil i/2 \rceil j}$ and b appears on a vertical side of block $B_{k,\lceil l/2 \rceil}$. In this configuration, an L-paths through the blocks is feasible as shown in Fig. 7.8c. A horizontal path from the terminal b up to the inside of block $B_{k,j}$ is obtained using the procedure as stated in Case 3. The vertical path from the terminal a up to the inside of the block $B_{k,j}$ is obtained as in Case 4. Inside $B_{k,j}$, these two wires are connected using exactly one via.

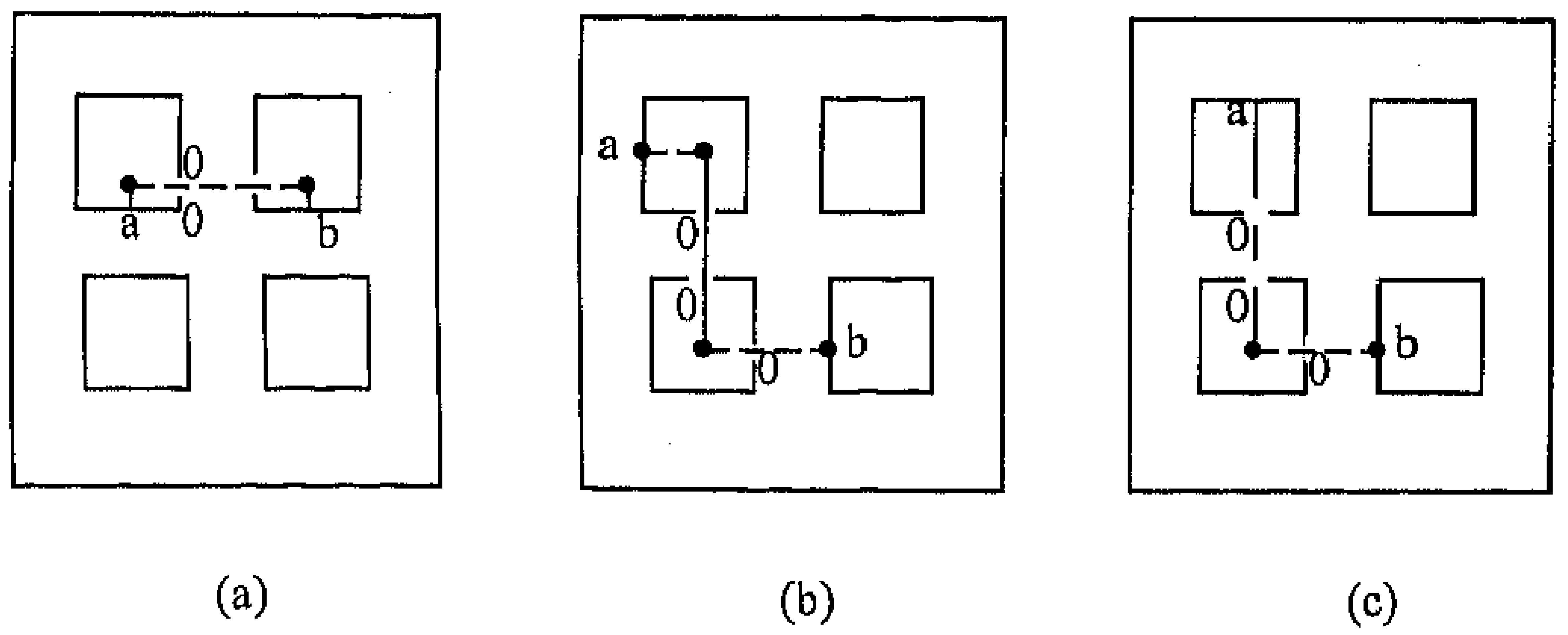


Fig. 7.8: Routing using space inside the blocks

Our path location algorithm efficiently finds a feasible routing path between any pair of terminals a and b, lying on the boundary of $B_{i,j}$ and $B_{k,l}$ respectively, in $O(\log G) + (|i-k| + |j-l|)$ time, where G is the maximum of the number of gaps on a block boundary row (BBR), and those on a block boundary column (BBC). This algorithm will be used for connecting two-terminal nets through the blocks, i.e., without using any track in the river(s).

7.3.2.3 Routing of multi-terminal nets

Consider a net α_i whose number of terminals (n_i) is more than two. The rectilinear steiner tree RST_i, obtained in Section 7.3.1, will guide the routing of the terminals of α_i. Let t_1, t_2, ..., t_q be the set of edges of RST_i, which were initially drawn through the rivers. We shall consider each edge of RST_i individually and try to move it on a path through the blocks, such that it releases a track from its assigned river. In case of failure, it will remain in its original position, i.e., through a track in the river. If a track is not available in the assigned river, the terminals on the selected edge are extended to the next layer-pair.

First we search for all vertical lines having unit weight, i.e., which contain a single terminal of α_i. Let t_j be such a line on which the position of the net α_i is at *a*. In order to reduce vias, an attempt is made to project *a* to a neighboring vertical line of t_j, say t_k, having weight more than one, by drawing a horizontal line from *a* to t_k through the vertical gaps. A new terminal is conceptually introduced on t_k at the point where *a* is projected on it. The weight of t_k is also incremented by one. If such a line is not feasible, the connection of *a* with RST_i using line t_j is retained. A similar procedure is then executed to consider all horizontal lines of RST_i each having unit weight.

We now have the vertical lines of RST_i with weights more than one. Let us consider those lines one be one in increasing order of their x-coordinates. Let t_j be the current vertical line under consideration which meets a horizontal line of RST_i, say t_l, at a point δ. Note that t_j is initially present on a vertical river. Now we try to replace t_j by a vertical path through the horizontal gaps of a set of blocks arranged in a column such that the path spans all the terminals on t_j. If such a vertical path is not found, but a vertical path spanning a subset of these terminals exists, it will not be selected since it may cause more vias and increase the complexity of the search. In this case, t_j is retained in its original position, i.e., through the assigned river, provided the congestion of the river satisfies one more wire to pass through the portion where the terminals of t_j spans. In either case, a new terminal is introduced on the horizontal line t_l at the point δ.

If the routing of t_j is not possible in the current layer-pair, all the terminals on t_j are projected to the next layer-pair. A similar procedure is applied to connect the terminals on the horizontal paths of RST_i.

7.4 Via Minimization

After completion of routing in the second phase, re-routing of wires through the rivers to relax the reserved-layer strategy may reduce the via count significantly. We adopt the following three techniques.

7.4.1 Flipping of layers

We assumed that all the vertical (horizontal) wire segments pass through layer-1 (layer-2) in a layer-pair. Consider a pair of terminals connected using a Z-path as shown in Fig. 7.9. It is easy to notice that if the terminals are connected to two vertical wires of the Z-path (Fig. 7.9a), only two vias are required. On the other hand, if the terminals are connected to the horizontal wires of the Z-path (Fig. 7.9b) , four vias are required, as two vias are necessary at the terminal positions. In such a scenario, an interchange in the layer assignment of the vertical/ horizontal wires causes that four vias are required in the situations shown in Fig. 7.9c, while two vias are required in the situations shown in Fig. 7.9d. Thus, if the number of terminals connected to vertical wire segments is more than the number of terminals connected to horizontal wire segments, we assign layer-1 and layer-2 to vertical and horizontal wire segments respectively. Otherwise, we assign vertical (horizontal) wire segments to layer-2 (layer-1).

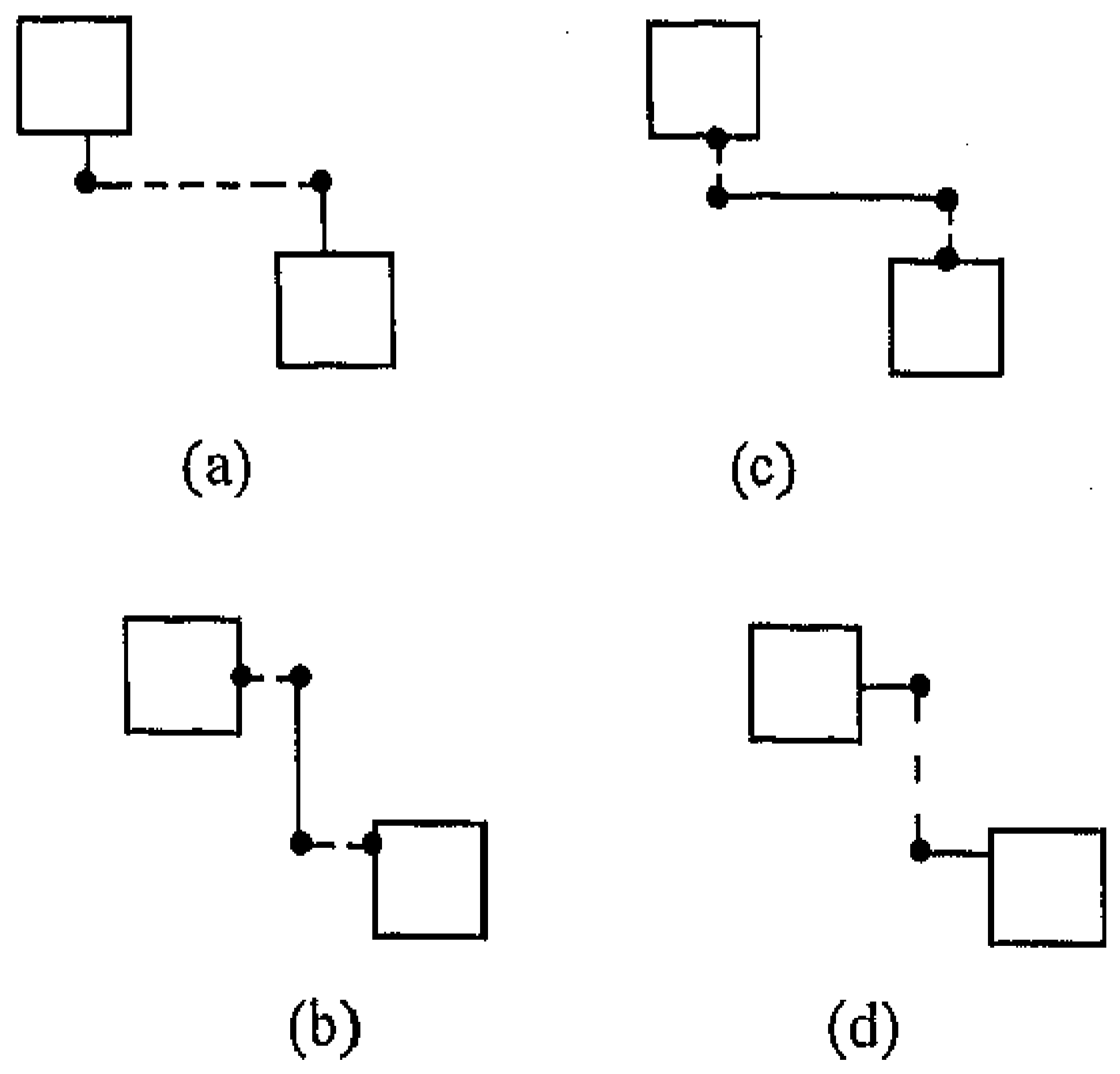


Fig. 7.9: Via minimization by flipping layer assignments

We may achieve further reduction of vias if we assume the unreserved layer routing model where both the horizontal and vertical tracks can pass through same layer. Below we explain few techniques which successfully identify the appropriate situations and reduce vias by assigning horizontal and vertical wire segments in the same layer avoiding overlapping of two different net segments and crossover in the same layer.

7.4.2 Permutation of tracks

Without loss of generality, assume that the vertical connections appear on the top layer of a layer-pair. Now consider the wire segments through a vertical river. These may be partitioned into three groups : (i) whose one end point is attached to a block adjacent to the left side of the river, (ii) whose one end point is attached to a block adjacent to the right side of the river, (iii) whose no end point is connected to a block adjacent to the river. Note that, the set of wire segments in group (i) and group (ii) may not be disjoint.

Let us first consider the set of vertical wire segments in group (i), and split all of them into disjoint classes, say C_1, C_2, ..., such that the topmost boundary of the elements of C_i is less than bottommost boundary of elements of C_{i+1} i.e., any two intervals of two different classes do not overlap. (Note that, if interval graph corresponding to this interval is connected then a single class is formed). Each class is considered separately for via minimization.

Consider a class C_i. Let C_{i1} and C_{i2} be two subsets of C_i, such that the members of C_{i1} (C_{i2}) are connected to some block along the left boundary of the river, at their top (bottom) end points. Note that $C_{i1} \cup C_{i2} \neq \emptyset$. If $|C_{i1}| \geq |C_{i2}|$, we

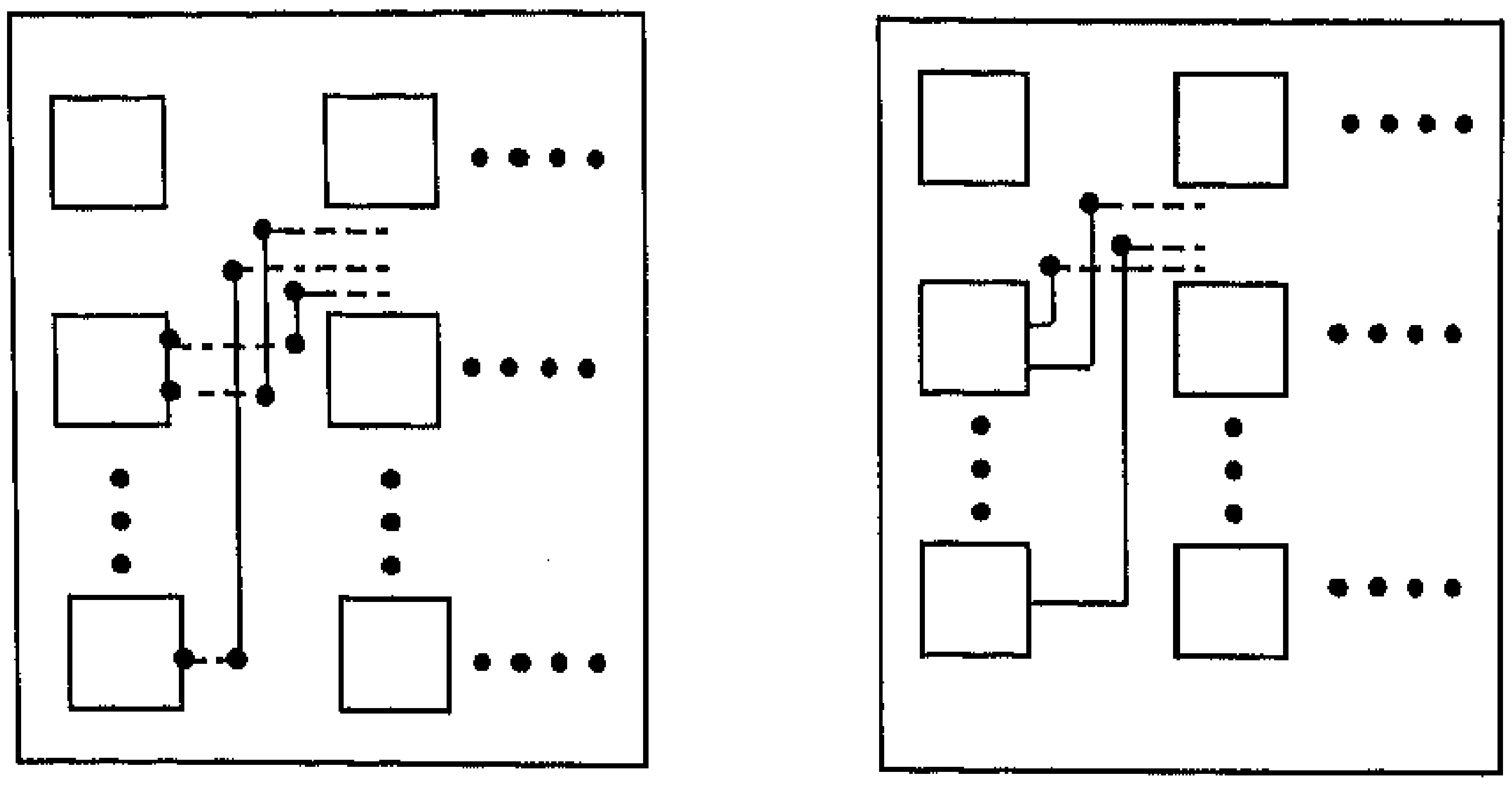


Fig. 7.10: Via minimization by permuting track assignments

assign tracks to the members of C_{i1} such that their top end point may be connected to the respective terminals along to the block boundaries without using any via. These tracks will be closer to the left boundary of the river. The situation is illustrated in Fig. 7.10. On the contrary, if $|C_{i1}| \leq |C_{i2}$, we assign the tracks to the segments in C_{i2} such that their bottom end points may be connected to the respective block boundaries without using any via. Needless to mention, these tracks will also be closer to the left boundary of the river.
For the elements in group (ii), the track allocation is done using the same technique as discussed above, but it is based on the connections of these segments with the blocks adjacent to the right side of the river. If an element belongs to both group (i) and group (ii), and its track allocation is already done while processing the elements of group (i), it need not be considered at the time of processing the elements in group (ii).

The segments of group (i) and group (ii), for which the track assignment is not done in the previous two steps, are assigned to tracks arbitrarily to the middle of the river. The end points of these wire segments are connected to terminals of the blocks using wires in the other layer through via holes. The track assignment for the elements in group (iii) is also done similarly.

7.4.3 Removal of redundant vias

In this method, we consider each dashed wire segment in layer-2 between two consecutive vias. We inspect whether any wire in layer-1 passes through that interval. If no such wire in layer-1 is found, we replace the dashed line by a solid line in layer-1, and reduce two vias. The same technique is then adopted for locating the possible solid lines in layer-1 which can be moved to layer-2 for reduction of vias.

7.5 Experimental Results

In Table 7.1 we give a list of standard benchmarks, where the first one is a randomly generated example [SBP95], and the last two examples, labeled *mcc1* and *mcc2-75*, are actual industrial problems. In Table 7.2, we present the performance of our algorithm using reserved layer routing model. We have implemented our algorithm in SUN ULTRA10 (333 MHz) workstation with 256 MB RAM. Our algorithm is simple, easy to implement, and produces output very fast.

Table 7.1 : Benchmark examples

Examples	# chips	# nets	# pins	grid size
test1	4	500	1,000	300 x 300
mcc1	6	802	2,495	599 x 599
mcc2-75	37	7,118	14,659	2,032 x 2,032

Table 7.2 : Performance of reserved layer MCM routing model

Examples	Number of layers	Number of vias	Total wire length	Time in sec.
test1	4	1852	102,832	0.22
mcc1	4	6383	380,938	10.28
mcc2-75	6	33882	5,540,616	17.38

We have also studied the performance of MCM routing under unreserved layer model. The solutions are easily obtainable from the results obtained for the reserved layer assignment model by incorporating the techniques discussed in Subsection 7.4.2 and 7.4.3. Table 7.3 presents the output of MCM routing using unreserved layer assignment model. This shows a significant improvement with the results obtained by V4R [KC95] and SEGRA [CRN97] in terms of number of vias. The number of layers remain same as those used in V4R and SEGRA, and with respect to the wire length, our solutions also compares well. It is also observed that if the grid size is increased slightly, the number of layers reduces significantly. For example, if we make the grid size (420 x 420) instead of (300 x 300), the number of routing layers reduces from 4 to 2 for the example *test1*.

Table 7.3 : Experimental results

Examples	Number of layers			Number of vias			Total wire length			
	V4R	SEGRA	our work	V4R	SEGRA	our work	V4R	SEGRA	our work	Lower bound
test1	4	4	4	2,250	2,080	1,520	104,128	103,682	103,072	102,238
mcc1	4	4	4	6,993	6,773	5,682	394,272	433,202	384,064	343,767
mcc2-75	6	6	6	36,438	36,362	30,136	5,559,479	5,517,133	5,541,806	5,362,181

7.6 Conclusion

In this chapter, we have proposed an efficient heuristic algorithm for via minimization in MCM routing. The overall time complexity of the algorithm is O(NlogN), where N is the total number of terminals present in the MCM. The algorithm produces around 18% fewer vias than the methods in [KC95, CRN97], with competitive values of wire length.

Chapter 8
Conclusion

Detailed routing is one of the most fundamental steps in VLSI physical design cycle. In this thesis, detailed routing problem is solved by routing the channels, L-shaped channels and switchboxes. Channel routing is one of the most important and probably one of the most popular phases of physical design for VLSI chips as well as PC-boards.

Routing results may differ based on the selection of routing models. A routing model can be grid-based where wires follow paths along the grid lines, or gridless where wires can be placed at any place as long as the design rules are not violated. In reserved-layer grid-based model, horizontal and vertical wire segments are allowed only on the pre-assigned layer(s). A model is unreserved if any segment is allowed to be placed in any layer. The most widely considered objective function for routing a channel is the minimization of channel width. A routing algorithm must also take into consideration the following objectives : total wire length, length of the longest wire segment, number of vias, to name a few.

In MCM routing, a number of functional blocks need to be connected using multiple routing layers. The objective is to achieve tight packaging and improve the performance of the circuit. Thus, to avoid faults in the circuit, we need to route using minimum number of vias, and to reduce the signal propagation delay, interference etc., the length of the interconnection wire segments among the functional modules need to be shortened.

In this thesis, we studied different channel routing techniques. The performance analysis is done in comparison to the existing works. Finally, we considered MCM routing and proposed an efficient algorithm to achieve high performance.

Over-the-cell area for channel routing has been successfully used to achieve dramatic reduction in the channel width. We have described a new heuristic algorithm which iteratively selects over-the-cell nets with an ultimate objective of minimizing channel width. Experimental evidence shows considerable improvement over earlier results [CL88] in the number of tracks used both inside the channel and over the cell.

We have presented a new algorithm for via minimization by layout modification in a two-layer channel routing environment. The method is based on interchange of horizontal tracks, and does not incorporate additional horizontal or vertical tracks. A significant reduction of vias for standard benchmark examples is observed. The total wire-lengths are also found to be very competitive.

In this thesis we have proposed a new approach to channel routing, that combines the advantage of both diagonal model and Manhattan geometry. We consider *two* routing layers with *fixed terminals,* and the *Manhattan-diagonal ($\pm 45^o$)* model. Thus, wire segments may either be rectilinear or at $\pm 45^0$ directions. First, we consider the *reserved layer* model (i.e., wire segments in different layers do not overlap), and present a simple $O(l.d)$ time algorithm that routes an arbitrary channel with *no cyclic vertical constraints* in w tracks, where l is the length of the channel i.e., the number of equidistant column in the channel; d is the channel density, and $d \leq w \leq (d+1)$. Next, we describe an output-sensitive algorithm that can route general channels with *cyclic vertical constraints* using w tracks, in $O(l.w)$ time (i.e., linear in area of the channel), assuming *unreserved layer MD model* that allows overlapping of wiring segments in two layers. The proposed algorithms outperform those in [YK82, PPD+95, HIZ91, CR91] both in time complexity and quality of solutions. Experiments with benchmark examples show very encouraging results.

The Manhattan-diagonal model is used for routing L-channels, switchboxes, and staircases. The core of the proposed L-shaped channel routing method lies in routing a right triangular region. This method is greedy, uses minimum number of tracks, can handle cyclic constraints and yields encouraging results. Next we described a switchbox router in MD-model, which outperforms existing routers. No extra row or column is required to route Burstein's and other difficult benchmark switchboxes. Finally, a scheme for staircase channel routing is proposed. The complexity of all algorithms are linear in the area of the region. These general channel definitions always guarantees a safe routing order even in nonslicing floorplans, and makes the router useful for MCMs. Our algorithms are simple, terminate in time linear in the routing area and provide elegant solutions to complex routing problem in a general floorplan.

Lastly, we have proposed an efficient heuristic algorithm for via minimization in MCM routing. The overall time complexity of the algorithm is O(NlogN), where N is the total number of terminals present in the MCM. The algorithm produces around 18% fewer vias than the methods in [KC95, CRN97], with competitive values of wire length.

Bibliography

[A67] S. B. Akers, "A modification of Lee's path connection algorithm," *IEEE Trans. on Computers*, EC-16, pp. 97-98, 1967.

[AK83] K. Aoshima and E. S. Kuh, "Multi-channel optimization in gate array layout," in Proc. *IEEE ISCAS*, pp. 1005-1008, 1983.

[AKR⁺89] I. Adler, N. Karmarkar, M. G. C. Resende and G. Veiga, "An implementation of Karmarkars algorithm for linear programming," *Math. Program.*, 44, pp. 297-335, 1989.

[AS88] M. H. Arnold and W. S. Scott, "An interactive maze router with hints," *25th. Design Automation Conference*, pp. 672-676, 1988.

[B79] H. Bollinger, "A mature DA system for PC layout," *Proc. First International Printed Circuit Conference*, 1979.

[BBD⁺86] D. Braun, J. Burns, S. Devadas, H. K. Ma, K.Mayaram, F. Romeo, and A. Sangiovanni-Vincentelli, "Chameleon: A new multi-layer channel router," in Proc. *23rd Design Automation Conference*, pp. 495-502, 1986.

[BCS91] S. Burman, H. Chen and N. Sherwani, "Improved global routing using λ-Geometry," *The proceedings of 29th Annual Allerton Conference on Communications, Computing and Controls*, October 1991.

[BP83] M. Burstein, and R. Pelavin, "Hierarchical channel router," *in Proc. 20-th Design Automation Conference*, pp. 591-597, 1983.

[C86] H. H. Chen, "Trigger: A three layer gridless channel router," *IEEE Int. Conf. Computer-Aided Design*, pp. 196-199, 1986.

[C87a] M. C. Chi, "An automatic rectilinear partitioning procedure for standard cells," *Proc. 24th. Design Automation Conference*, pp. 50-53, 1987.

[C87b] H.H. Chen, "Routing L-Shaped channels in nonslicing-structure placement," *Proc. 24th Design Automation Conference*, pp. 152-158, 1987.

[CD87] K. C. Chang and D. H. Du, "Efficient algorithms for layer assignment problem," *IEEE Trans. on Computer-Aided Design*, vol. 6(1), pp. 67-78, 1987.

[CD88] C. K. Cheng and D. N. Deutch, "Improved channel routing by via minimization and shifting," *in Proc. 25th Design Automation Conference*, pp. 677-680, 1988.

[CG96] J. P. Cohoon and J. L. Ganley, "Rectilinear interconnection in the presence of obstacle," in Y. T. Wong and M.Peht, Ed., *Advanced routing of electronic Modules*. CRC Press, Bocaraton, Florida, 1996.

[CH88] J. P. Cohoon and P. L. Heck, "BEAVER : A computational geometry based tool for switchbox routing," *IEEE Trans. on Computer-Aided Design*, vol. 7, pp. 684-697, 1988.

[CHS94] C. Y. R. Chen, C. Y. Hou, and U. Singh, "Optimal algorithms for bubble sort based non_Manhattan channel Routing," *IEEE Trans. on Computer-Aided Design*, vol. 13, pp 603-609, 1994.

[CK81] M. J. Ciesielski and E. Kinnen, "An optimal layer assignment for routing in IC's and PCB's," *Proc. 18th. Design Automation Conf.*, pp. 733-737, 1981.

[CK86] H. H. Chen and E. Kuh, "Glitter : A gridless variable-width channel router," *IEEE Trans. on Computer-Aided Design*, vol. 5(4), pp. 459-465, 1986.

[CKC83] R. W. Chen, Y. Kajitani and S. P. Chain, "A graph-theoretic via minimization algorithm for two-layer printed circuit boards," *IEEE Trans. on Circuits syst.*, Vol. 30, pp. 284-299, 1983.

[CL84] Y. Chen and M. Liu, "Three-layer channel routing," *IEEE Trans. on Computer-Aided Design*, vol. 3, pp. 156-163, 1984.

[CL88] J. Cong and C. L. Liu, "Over-the-cell channel routing,," *in Proc. International Conf. on Computer-Aided Design.*, pp. 80-83, 1988.

[CL91] J. Cong and C. L. Liu, "On the k-layer planar subset and via minimization problems," *IEEE Trans. on Computer-Aided Design*, vol. 10, pp. 972-981, 1991.

[CLC96] R. C. Carden, J. Li and C. -K. Cheng, "A global router with a theoretical bound on the optimal solution," *IEEE Trans. on Computer-Aided Design*, vol. 15(2), pp. 208-216, 1996.

[CLR+94] J. D. Cho, K. Liao, S. Raje and M. Sarrafzadeh, "M^{2}R : Multilayer routing algorithm for high performance MCM's," *IEEE Trans. on Circuits and Systems*, vol. 1, pp. 253-265, 1994.

[CPL90] J. Cong, B. T. Preas and C. L. Liu, "General models and algorithms for over-the-cell routing in standard cell design," *in Proc. 27th Design Automation Conference*, pp. 709-715, 1990.

[CR91] K. Chaudhary and P. Robinson, "Channel routing by sorting," *IEEE Trans. on CAD*, vol. 10 (6), pp. 754-760, 1991.

[CRN97] Y. -J. Cha, C. S. Rim and K. Nakajima, "A simple and effective greedy multi-layer router for MCMs," *Proc. Int. Symp. on Physical Design*, pp. 67-72, 1997.

[CS91] J. D. Cho and M. Sarrafzadeh, "The pin redistribution problem in multichip modules," *Proc. IEEE Application Specific IC Conference*, pp. 9-21, 1991.

[CSW89] C. Chiang, M. Sarrafzadeh and C. K. Wong, “A powerful global router : based on steiner min-max trees,” *IEEE Int. Conf. Computer-Aided Design*, pp. 2-5, 1989.

[CSW92] C. Chiang, M. Sarrafzadeh and C. K. Wong, “An algorithm for exact rectilinear steiner tree for switchbox with obstacles,” *IEEE Trans. on Circuits and Syst..*, vol. 39, pp. 446-455, 1992.

[CWL88] J. Cong, D. F. Wong, and C. L. Liu, “A new approach to three or four-layer channel routing,” *IEEE Trans. on CAD*, vol. 7, pp. 1094-1104, 1988.

[D76] D. N. Deutsch, “A dogleg channel router,” *in Proc. 13th Design Automation Conference*, pp. 425-433, 1976.

[D85] D. N. Deutsch, “Compacted channel routing,” in *Proc. ICCAD*, pp. 223 -225, 1985.

[D90] W. M. Dai, “Performance driven layout of thin flim substrates for multichip modules,” *Proc. of multichip module workshop*, pp. 114-121, 1990.

[DB93] S. Das and B.B. Bhattacharya, “Via minimization in channel routing by layout modification,” *Proc. 6th Intl. Conf. VLSI Design*, p. 109 , 1991.

[DB96] S. Das and B.B. Bhattacharya, “Channel routing in manhattan-diagonal model,” *Proc. 9th Intl. Conf. VLSI Design*, pp. 43 - 48 , 1996.

[DAK85] W. M. Dai, T. Asano and E.S. Kuh, “Routing region definition and ordering scheme for building-block layout,” *IEEE Trans. on CAD*, vol. 4(3), pp. 189-197, 1985.

[DG80] D. N. Deutsch and P. Glick, “An over-the-cell router,” *in Proc. 17th Design Automation Conference*, pp. 80-83, 1980.

[DK69] C. A. Desoer and E. S. Kuh, *Basic Circuit Theory*. McGraw Hill, New York, 1969.

[DK82] W. A. Dees and P. G. Karger, "Automated rip-up and reroute techniques," *Proc. 19th. Design and Automation Conference*, pp. 432-439, 1982.

[DK85] A. E. Dunlop and B. W. Kernighan, "A procedure for placement of standard-cell VLSI circuits," *IEEE Trans. Computer Aided Design*,, vol. 4, pp. 92-98, 1985.

[DKS+87] D. Dolev, K. Karplus, A. Siege, A. Strong and J. D. Ullman, "Optimal wiring between rectangle," *13 Annual ACM Symposium on Theory of Computation*, pp. 312-317, 1987.

[DNB91] S. Das, S. C. Nandy and B. B. Bhattacharya, "An Improved heuristic algorithm for over-the-cell channel routing," *Proc. International Symposium on Circuits and Systems (ISCAS)*, pp. 3106- 3109, 1991.

[DNB99] S. Das, S. C. Nandy and B. B. Bhattacharya, "High performance MCM routing : A new approach," *Proc. 12th Intl.Conf. VLSI Design*, pp. 564-569, 1999.

[DNS+96] P.S. Dasgupta, S. C. Nandy, A. K. Sen and B. B. Bhattacharya, "Geometric bipartitioning problem and its applications to VLSI," *Proc. 9th Int. Conf. VLSI Design*, pp. 400- 405 , 1996.

[DSB98] S. Das, S. Sur-Kolay and B.B. Bhattacharya, "Routing of L-shaped channels, switchboxs and staircases in Manhattan-diagonal model," *Proc. 11th Intl.Conf. VLSI Design*, pp. 65-70, 1998.

[ED86] R. J. Enbody and H. C. Du, "Near-optimal n-layer channel routing," *Proc. 23rd. Design Automation Conf.*, pp. 708-714, 1986.

[FM82] C. M. Fiduccia and R. M. Mattheyses, "A linear time heuristic for improving network partitions," *Proc. 19th. Design Automation Conf.*, pp. 175-181, 1982.

[G80] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1980.

[GJ77] M. R. Garey and D. S. Johnson, "The rectilinear steiner tree problem is NP-complete," *SIAM Journal on Applied Mathematics*, 32, pp. 826-834, 1977.

[GJ79] M. R. Garey and D. S. Johnson, *Computers and Intractability : A Guide to the Theory of NP-completeness.* W. H. Freeman & Co., San Francisco, 1979.

[GN87] G. Gudmundsson and S. Ntafas, "Channel routing with superterminals," *in Proc. 25th Allerton Conference on Computing Control and Communication,* pp. 665-670, 1987.

[GW88] M. Guruswamy and D.F. Wong, "Channel routing order for building-block layout with rectilinear modules," *Proc. ICCAD*, pp. 184-187, 1988.

[H69] D. W. Hightower, "A solution to the line routing problem on a continuous plane," *Proc. Design Automation Workshop*, pp. 1-24, 1969.

[H76] F. K. Hwang, "On steiner minimal trees with rectilinear distances," *SIAM Journal on Applied Mathematics*, vol. 30(1), pp. 104-114, 1976.

[H77] F. O. Hadlock, "A shortest path algorithm for grid graphs," *Networks*, 1977.

[H78] F. K. Hwang, "An O(n logn) algorithm for suboptimal rectilinear steiner trees," *IEEE Transactions on Circuits and Systems*, vol. 26(1), pp. 75-77, 1978.

[H83] C. P. Hsu, "Minimum-via topological routing," *IEEE Trans. Computer-Aided Design,* vol. 2, pp. 235-246, Oct. 1983.

[H83a] C. P. Hsu, "General river routing algorithm," *in Proc. 20th Design Automation Conference,* pp. 578-583, 1983.

[H85] S. E. Hambrusch, "Channel routing algorithms for overlap models," *IEEE Trans. on Computer-Aided Design,* vol. 4(1), pp. 23-30, 1985.

[HI89] T. T. Ho and S. S. Iyenger, "Density based general greedy channel routing," *Manuscript,* 1989.

[HIZ91] T. T. Ho, S. S. Iyengar, and S. Q. Zheng, "A greedy channel routing algorithm," *IEEE Trans. on CAD,* vol.10, pp. 204-211, Feb. 1991.

[HL91] J. Heisterman and T. Lenguer, "The efficient solution of integer programs for Hierarchical Global Routing," *IEEE Trans. on Computer-Aided Design*, CAD-10, pp. 748-753, 1991.

[HO84] G. Hamachi and A. Ousterhout, "A switchbox router with obstacle avoidance," Proc. *21-st Design Automation Conference*, pp. 173-179, 1984.

[HS71] A. Hashimoto and J. Stevens, "Wire routing by optimization channel assignment within large apertures," *in Proc. 8th Design Automation Workshop,* pp. 155-163, 1971.

[HS84] S. Han and S. Sahni, "A fast algorithm for single row routing," Technical Report 84-5, Depertment of Computer Science, University of Minesota, Minneapolis, 1984.

[HSS91] N. Holmes, N. A. Sherwani and M. Sarrafzadeh, "New algorithm for over-the-cell channel routing using vacant terminals," *in Proc. 28th Design Automation Conference,* pp. 126-131, 1991.

[HSV$^+$90] J. M. Ho, M. Sarrafzadeh, G. Vijayan and C. K. Wong, "Layer assignment for multichip modules," *IEEE Trans. on Computer-Aided Design*, vol. 9, pp. 1272-1277, 1990.

[HVW85] J. M. Ho, G. Vijayan and C. K. Wong, "A new approach to the rectilinear steiner tree problem," *IEEE Transactions on Computer-Aided Design*, vol. 9(2), pp. 185-193, 1985.

[HYY90] A. Hanafusa, Y. Yamashita and M. Yasuda, Three-dimentional routing for multilayer ceramic printed circuit boards," *Proc. ICCAD,* pp. 386-389, 1990.

[J86] R. Joobani, *An Artificial Intelligence Approach to VLSI Routing.* Kluwer Academic Publishers, Boston, MA., 1986.

[JP89] A. Joseph and R. Y. Pinter, "Feed-through river routing," *Integration, the VLSI Journal,* vol. 8, pp. 41-50, 1989.

[K80] Y. Kajitani, "On via hole minimization of routing in a 2-layer board," *in Proc. IEEE International Conference on Circuits and Computers*, pp. 295-298, 1980.

[K80] Y. Kajitani, "Order of channels for safe routing and optimal compaction of routing area," *IEEE Trans. on Computer-Aided Design*, vol. 2, pp. 293-300, Apr. 1983.

[KC92] K. Y. Khoo and J. Cong, "A fast multilayer general area router for MCM designs," *IEEE Trans. on Circuits and Systems II*, pp. 841-851, 1992.

[KC95] K. Y. Khoo and J. Cong, "An efficient multilayer MCM router based on four-via routing," *IEEE Trans. on Computer-Aided Design*, vol. 14, pp. 1277-1290, Oct. 1995.

[KCS88] Y. S. Kuo, T. C. Chern and W. K. Shih, "Fast algorithm for optimal layer assignment," *in Proc. 25th Design Automation Conference,* pp. 554-559, 1988.

[KK79] T. Kawamoto and Y. Kajitani, "The minimum width routing of a 2-row 2-layer polycell layout," *in Proc. 16th Design Automation Conf.*, pp. 290-296, 1979.

[KKF79] E. S. Kuh, T. Kashiwabara and T. Fujisawa, "On optimum single row routing," *IEEE Trans. on Circuits and Systems*, vol. 26, pp. 361-368, 1979.

[KLR+87] R. M. Karp, F. T. Leighton, R. L. Rivest, C. D. Thompson, U. V. Vazirani and V. V. Vazirani, "Global wire routing in two dimensional arrays," *Algorithmica*, 1987.

[KPS89] B. Korte, H. J. Promen and A. Streger, "Combining partitioning and global routing in sea-of-cells design," *Proc. IEEE Int. Conf. Computer-Aided Design*, pp. 98-101, 1989.

[L61] C. Lee, "An algorithm for path connection and its applications," *IRE Trans. Electronic Computers*, EC-10, pp. 346-365, 1961.

[L80] A. S. LaPaugh, *Algorithms for Integrated Circuit Layout : An Analytic Approach.* Ph. D. thesis, MIT Lab, Comp. Sc., 1980.

[L85] W. K. Luk, "A greedy switchbox router," *Integration, the VLSI Journal,* vol. 3, pp. 129-149, 1985.

[LD86] D. LaPotin and S. W. Director, "Mason : a global floorplanning approach for VLSI design," *IEEE Trans. on Computer-Aided Design*, vol. 5, pp. 477-489, 1986.

[LLP89a] E. Lodi, F. Luccio, and L. Pagli, "A preliminary study of a diagonal channel-routing model," *Algorithmica,* vol. 4, pp. 585-597, 1989.

[LLP89b] E. Lodi, F. Luccio, and L. Pagli, "Channel routing for strictly multiterminal nets," *Integration, the VLSI Journal* , vol. 8(2), pp. 143-153, 1989.

[LLP90] E. Lodi, F. Luccio, and L. Pagli, "Routing in time square mode," *Information Processing Letters,* vol. 35, pp. 41-48, 1990.

[LLS91] E. Lodi, F. Luccio, X. Song, "A 2D channel router for the diagonal model," *Integration, the VLSI Journal* , vol. 1(2), pp. 111-125, 1991.

[LP83] C. E. Leiserson and R. Y. Pinter, "Optimal placement for river routing," *SIAM Journal of Computing,* vol. 12(3), pp. 447-462, 1983.

[LPH+91] M. S. Lin, H. W. Perng, C. Y. Hwang and Y. L. Lin, "Channel density reduction by routing over the cells," *in Proc. 28th Design Automation Conference,* pp. 120-125, 1991.

[LS88] K. W. Lee and C. Sechen, "A new global router for row-based layout," *IEEE Int. Conf. Computer-Aided Design*, 1988.

[LSL90] R.D. Lou, M. Sarrafzadeh and D. T. Lee, "An optimal algorithm for the maximum two-chain problem," *in Proc. of first SIAM-ACM Conference on Discrete Algorithms*, 1990.

[M59] E. F. Moore, "Shortest path through a Maze," *Proc. Int. Symp. Theory of Switching Circuits*, in Annals of the Harvard Computation Laboratory, vol. 30, pt. II, Cambridge, Mass, Harvard University Press, pp. 285-292, 1959.

[M90] F. M. Maley, *Single-layer wire routing and compaction.* The MIT Press, 1990.

[MBG+91] R. Miracky, T. Bishop, C. Galanakis, H. Hashemi, T. Hirsch, S. Madere, H. Muller, T. Ruswick, L. Smith, S. Sommerfeldt and B. Weighler, "Technology for rapid prototyping of multi-chip modules," *Proc. IEEE Int'l Conf. on Computer Design*, pp. 588-591, 1991.

[MNB98] S. Majumder, S. C. Nandy and B. B. Bhattacharya, "Routing driven floorplan partitioning using staircase channels," *Proc. 12th Intl.Conf. on VLSI Design*, pp. 59-64, 1998.

[MPS86] K. Mehlhorn, F. P. Preparata, and M. Sarrafzadeh, "Channel routing in knock-knee mode: simplified algorithms and proofs," *Algorithmica*, vol. 1(2), pp. 213-221, 1986.

[MT68] K. Mikami and K. Tabuchi, "A computer program for optimal routing of printed circuit connectors," *IFIPS Proc.*, H47, pp. 1475-1478, 1968.

[MZ89] S. Rao Maddila and D. Zhou, "Routing in general junctions," *IEEE Trans. on Computer-Aided Design*, vol. 8(11), pp. 1174-1184, 1989.

[NMN87] N. J. Naclerio, S. Masuda and K. Nakajima, "Via minimization for gridless layouts," *Proc. 24th Design Automation Conference*, pp. 159 -165, 1987.

[NMN89] N. J. Naclerio, S. Masuda and K. Nakajima, "Via minimization problem is NP-complete," *IEEE Trans. on Computers*, vol. 38(11), pp. 1604-1608, 1989.

[P82] R. Y. Pinter, "Optimal layer assignment for interconnect," *in Proc. International Conf. Circuit Comput.*, pp. 398-401, 1982.

[PDP+95] R. K. Pal, A. K. Datta, S. P. Pal, M. M. Das, and A. Pal, "A general graph theoretic framework for multi-layer channel routing," in *Proc. 8th Int. Conf. VLSI Design*, pp. 202-207, 1995.

[PE96] D. A. Pucknell and K. Eshraghian, *Basic VLSI Design*. Prentice Hall, 1996.

[PPC89] B. Preas, M. Pedram and D. Curry, "Automatic layout of silicon-on-silicon hybrid packages," in Proc. *Design Automation Conference*, pp 394-399, 1989.

[PPD+95] R. K. Pal, S. P. Pal, M. M. Das, and A. Pal, "Computing area and wire length efficient routes for channels," in *Proc. 8th Int. Conf. VLSI Design*, pp. 196-201, 1995.

[PZ87] V. Pitchumani and Q. Zhang, "A mixed HVH-VHV algorithm for three layer channel routing," *IEEE Trans. on Computer-Aided Design*, vol. 6(4), 1987.

[R74] F. Rubin, "The Lee path connection algorithm," *IEEE Trans. on Computers*, vol. 23, pp. 907-914, 1974.

[R84] D.Richards, "Complexity of single-layer routing," *IEEE Trans. on Computer-Aided Design*, vol. 3, pp. 286-288, 1984.

[RF82] R. L. Rivest and C. M. Fiduccia, "A greedy channel router," in Proc. *19th Design Automation Conference*, pp.418-424, 1982.

[RKN89] C. S. Rim, T. Kashiwabara and K. Nakajima, "Exact algorithms for multilayer topological via minimization," *IEEE Trans. on Computer-Aided Design*, vol. 8, pp. 1165-1184, 1989.

[RS83] R. Raghavan and S. Sahni, "Single row routing," *IEEE Trans. on Computers*, vol. 32, pp. 209-220, 1983.

[RS84] R. Raghavan and S. Sahni, "Complexity of single row routing problems," *IEEE Trans. on Circuits and Systems*, vol. 31(5) pp. 462-471, 1984.

[RVS85] J. Reed, A. Sangiovanni-Vincentelli, and M. Santomauro, "A new symbolic channel router: YACR2," *IEEE Trans. on Computer-Aided Design,* vol 4, pp. 208 - 219, 1985.

[RT91] R. Raghavan and C. D. Thompson, "Multiterminal global routing : a deterministic approximation scheme," *Algorithmica*, 6, pp. 73-82, 1991.

[RZ89] S. Rao Maddila and D. Zhou, "Routing in general junctions," *IEEE Trans. on CAD,* vol. 8, no. 11, pp. 1174-1184, November 1989.

[S78] J. Soukup, "Fast maze router," *Proc. 15th Design Automation Conference*, pp. 100-102, 1978.

[S74] H. C. So, "Some theoretical results on the routing of multilayer printed-wiring boards," *Proc. of IEEE International Symposium on Circuits and Systems*, pp. 296-303, 1974.

[S84] M. Marek-Sadowska, "An unconstrained topological via minimization problem for two-layer routing," *IEEE Trans. Computer-Aided Design,* vol. 3(3), pp. 184-190, 1984.

[S85a] T. G. Szymanski, "Dogleg channel routing is NP-Complete," *IEEE Trans. on Computer-Aided Design*, vol. 4, pp. 31-41, 1985.

[S85b] Y. Sheffi, *Urban Transportation Network.* Englewood Cliffs, Prentice Hall, NJ, 1985.

[S92] X. Song, "An algorithm for L-shaped channel routing in diagonal model," *IEEE Trans. on Computer-Aided Design*, vol. 11, pp. 267-270, 1992.

[S93] N. A. Sherwani, *Algorithms for VLSI Physical Design Automation.* Kluwer Academic Publishers, 1993.

[S93a] X. Song, "A Heuristic DM switchbox router," *Manuscript,* 1993.

[SB80] S. Sahni and A. Bhatt, "The complexity of design automation problems," in Proc. *17th DAC*, pp. 402-411, 1980.

[SB91] S. Sur-Kolay and B.B. Bhattacharya, "The cycle structure of channel graphs in nonslicible floorplans and a unified algorithm for feasible routing order," *Proc. ICCD*, pp. 524-527, 1991.

[SBP95] N. Sherwani, S. Bhingarde, A. Panyam, *Routing in the Third Dimension : From VLSI Chips to MCMs*, IEEE Press, Piscataway, NJ, 1995.

[SD81] A. Siegel and D. Dolev, "The separation for general single-layer wiring barriers," *Carnegie-Mellon Conference on VLSI Systems and Computations*, pp. 143-152, 1981.

[SD89] N. Sherwani and J. Deogun, "A new heuristic for single row routing problems," *Proc. ACM Design Automation Conf.*, pp. 167-172, 1989.

[SHL90] M. Stallmann, T. Hughes and W. Liu, "Unconstrained via minimization for topological multilayer routing," *IEEE Trans. on Computer-Aided Design*, vol. 10, pp. 970-980, 1990.

[SK87] E. Shragowitz and J. Keel, "A global router based on multicommodity flow model," *INTEGRATION : The VLSI Journal*, 1987.

[SK92] M. Sriram and S. M. Kang, "Detailed layer assignment for MCM routing," *Proc. IEEE Int'l Conf. on Computer-Aided Design*, pp. 386-389, 1992.

[SL89] M. Sarrafzadeh and D. T. Lee, "A new approach to topological via minimization," *IEEE Trans. on Computer-Aided Design*, vol. 8, pp. 890-900, 1989.

[SS87] Y. Shiraishi and Y. Sakemi, "A permeation router," *IEEE Trans. Computer-Aided Design*, vol. 6, pp. 462-471, 1987.

[ST83] M. Marek-Sadowska and T. T. Trang, "Single-layer routing for VLSI : Analysis and algorithms," *IEEE Trans. on Computer-Aided Design*, pp. 246-259, 1983.

[ST93] X. Song and X. Tan, "An optimal channel routing algorithm on a hexagonal grid," *Manuscript* 1993.

[SV79] K. R. Stevens and W. M. VanCleemput, "Global via elimination in generalized routing environment," *Proc. International Symposium on Circuits and Systems*, pp. 689-692, 1979.

[T81] M. Tompa, "An optimal solution to a wire routing problem," *Journal of Computer and System Science*, vol. 23(2), pp. 127-150, 1981.

[TCC+92] C. Tsai, S. Chen, Y. Chen and Y. Hu, "Planning strategies for Area Routing" *in Proc. European Design Automation Conference*, pp. 338-342, 1992.

[TH90] T. Tuan and S. L. Hakimi, "River routing with small number of jogs," *SIAM Journal of Discrete Mathematics*, vol. 3(4), pp. 585-597, 1990.

[TKS76] B. S. Ting, E. S. Kuh and I. Shirakawa, "The multilayer routing problem: Algorithms and necessary and sufficient conditions for single row, single layer case," *IEEE Trans. on Circuits and Systems*, pp. 768-778, 1976.

[TKS82] S. Tsukijama, E. S. Kuh and I. Shirakawa, "An algorithm for single row routing with prescribed street congestion," *IEEE Trans. on Circuits and Systems*, pp. 765-772, 1982.

[TSK84] T. T. K. Trang, M. Marek-Sadowska and E. S. Kuh, "An efficient single row routing algorithm," *IEEE Trans. on Computer-Aided Design*, pp. 178-183, 1984.

[TTN+91] M. Terai, K. Takahashi, K. Nakajima and K. Sato, "A new model for over-the-cell channel routing with three layers," *in Proc. of IEEE International Conference on Computer-Aided Design*, pp. 432-435, 1991.

[TWC91] K-S. The, D. F. Wong, and J. Cong, "A layout modification approach to via minimization," *IEEE Trans. on Computer-Aided Design*, vol. 10(4), pp. 536-541, 1991.

[V91] A. Vannelli, "An adaptation of the interior point method for solving the global routing problem," *IEEE Trans. on Computer-Aided Design*, vol. 10, 1991.

[VCW89] G. Vijayan, H. H. Chen and C. K. Wong,"On VHV-routing in channels with irregular boundaries," *IEEE Trans. on Computer-Aided Design*, vol. 8(2), 1989.

[VK83] M. P. Vecchi and S. Kirkpatrick, "Global wiring by simulated annealing," *IEEE Trans. on Computer-Aided Design*, vol. 2, 1983.

[VS82] A. Sangiovanni-Vincentelli, and M. Santomauro, "YACR: Yet another channel router," *Proc. Custom Integr. Circuit Conf.*, Rochester, NY, pp. 460 - 466, 1982.

[W85] S. K. Wismath, "Characterizing bar line-of-sight graphs," *Proc. 1st ACM Symposium on Computational Geometry*, pp. 147 -152, 1985.

[W91] D. Wang, "Novel routing schemes for IC layout, Part I : Two-layer channel routingr," *in Proc. 28th Design Automation Conference*, pp. 49-53, 1991.

[WHS+92] B. Wu, N. Holmes, N. Sherwani and M. Sarrafzadeh, "Over-the-cell routers for new cell models," *in Proc. 29th Design Automation Conference*, pp. 604-607, 1992.

[Y99] J. T. Yan, "An improved optimal algorithm for bubble-sorting-based Non-Manhattan channel routing," *IEEE Trans. on Computer-Aided Design*, vol. 18(2), pp 163-171, 1999.

[YH96] J. T. Yan, and P. Y. Hsiao, "Minimizing the number of switchboxes for region definition and ordering assignment," *IEEE Trans. on Computer-Aided Design*, vol. 15, pp 336-347, 1996.

[YK82] T. Yoshimura and E. S. Kuh, "Efficient algorithms for channel routing," *IEEE Trans. on Computer-Aided Design*, vol. 1, pp. 25-35, 1982.

[YW96] C. Yeh and C. S. Wang, "On the integration of partitioning and global routing for rectilinear placement problems," *IEEE Trans. on Computer-Aided Design*, vol. 15, pp. 83-91, 1996.

List of Publications of the Author related to this Thesis

1. S. Das, S. C. Nandy and B. B. Bhattacharya "An improved heuristic algorithm for over-the-cell channel routing," *Proceedings, International Symposium on Circuits and Systems (ISCAS)*, Singapore, pp. 3106-3109, IEEE CS Press, June 1991.

2. S. Das and B. B. Bhattacharya "Via minimization in channel routing by layout modification," *Proceedings, 6th International Conference on VLSI Design*, p. 109, IEEE CS Press, January 1993.

3. S. Das and B. B. Bhattacharya "Channel routing in Manhattan-Diagonal model," *Proceedings, 9th International Conference on VLSI Design*, pp. 43-48, IEEE CS Press, January 1996.

4. S. Das, S. Sur-Kolay and B. B. Bhattacharya "Routing of L-shaped channels, switchboxes, and staircases in Manhattan-Diagonal model," *Proceedings, 11th International Conference on VLSI Design*, pp. 65-70, IEEE CS Press, January 1998.

5. S. Das, S. C. Nandy and B. B. Bhattacharya, "High performance MCM routing: A new approach," *Proc. 12th International Conference on VLSI Design*, pp. 564-569, IEEE CS Press, January 1999.