A New Efficient Approach to Multilayer Channel Routing Problem

Sung-Chuan Fang, Wu-Shiung Feng and Shian-Lang Lee

Department of Electrical Engineering

National Taiwan University

Taipei, Taiwan, R.O.C.

Abstract

Among many successfully published papers for solving multilayer channel routing problem, they usually allowed doglegs in their solutions. As a consequence, they correspondingly will introduce more vias (or contacts) and longer wire length in their results than the ones using dogleg-free routing model. From the view of circuit performance, larger via count and longer wire length will result in longer timing delay and lower yield of the design. In this paper, a very efficient multilayer channel router, the M^3CR , will be well-described. The goal of M^3CR is to use optimum number of tracks for any multilayer routing instance in dogleg-free routing model. As well M^3CR has simple time complexity because it is not a maze-running-based router. From the experiments, M^3CR has produced optimum numbers of tracks for all the test examples from 4- through 8- layer in very short running time. Moreover, the variable-wire-width consideration according to the design rules is also discussed and resolved in this paper.

1 Introduction

The channel routing problem is one of the key phases in VLSI automatic layout design and implementation. The traditional channel routing problem assumes that there are two or three layers of materials available for routing. With the advance of manufacturing technology, more layers for routing can be expected in the near future. Thus it is getting more and more important to extend channel routing to handle multilayer regions.

A few multilayer channel-routing algorithms have been developed. *Enbody and Du* [1] proposed a multilayer router using leading column heuristics and limited backtracking. They determine the routing by using a *Five-Column-Window* that moves from left to right through the channel. *Hambrusch* [2] developed an n -layer channel router which puts severe constraints on the arrangement of the wiring and deals with only two terminal nets. Such constraints make the algorithm inapplicable. *Cong, Wong and Liu* [3] proposed a general technique which transforms a two-layer routing solution systematically into a three-layer or four-layer routing solution. But the transformation considered for only three and four layers. *Braun et al.* [4] implemented a multilayer channel router which divides the problem into two-layer and three-layer subproblems and completes each subproblem by the extended

two-layer router *YACR2* [5] and a three-dimensional maze router.

All previous works introduced many doglegs in their routing solutions. Based on electrical standpoint, each dogleg adds one or two additional vias, increases the wire length, and decreases the performance of the circuit. The idea of introducing dogleg is to remove the heavy constraint of longest vertical constrained paths in vertical constraints graph [6]. However, if more than two layers are available for vertical wire segments, the longest vertical constrained paths are no longer related to a lower bound of channel width, which is mainly decided by channel density now. Thus a *dogleg-free* (or *restrictive*) multilayer channel router is valuable to be developed.

This paper presents a new approach to *multilayer channel routing problem*. Only the dogleg-free solutions are allowed in our router. From some theorems and the experiments, we can show that the dogleg is no longer need to improve the minimum channel width in multilayer routing environments. The router divides layers into some partitions. Each partition contains either two or three layers. Layers dedicated to horizontally oriented wire segments are called horizontal layers (H-layers); layers dedicated to vertically oriented wire segments are called vertical layers (V-layers). The VH (or HV) pattern is used by two-layer partition. The HVH is used by three-layer partition.

2 Preliminaries

2.1 Channel Description

A rectangle channel is described by two ordered list of terminals along its top and bottom boundaries. A nonnegative integer is assigned to each terminal. terminals are ordered in top and bottom lists: $TOP = (T_1, T_2, \dots, T_L)$ and $BOT = (B_1, B_2, \dots, B_L)$, where L is the length of the channel. T_i (or B_i) is a terminal at the top (or bottom) boundary in the i -th column. Terminals with the same positive label have to be connected. Zeros in two lists mean that there is no connection should be made in those points. A simple example is shown in Figure 1. The length of the channel is determined by the positions of the leftmost and rightmost terminals on the top and bottom boundaries. The channel width, the number of horizontal tracks used, is determined by the router. If a router wants to minimize the channel area, it must minimize the number of horizontal tracks.

29th ACM/IEEE Design Automation Conference®

Figure 1: A simple channel routing problem.

2.2 Some Basic Definitions

To describe our approach more conveniently, we need the following definitions.

Definition 2.1 : Let L_i and R_i denote a column number of the leftmost and rightmost terminals for a net i , respectively. The x_region of a net i is defined by an interval (L_i, R_i) , and the x_length of a net is the number of columns in its x_region .

Definition 2.2 : Let $S(j)$ be the set of unassigned nets in which their horizontal segments intersect column j . The unassigned nets are those nets which haven't been physically located in the channel.

Definition 2.3 : The local density of a channel at a given column j , d_j , is the number of different nets in $S(j)$. The maximum local density, denoted by $dmaz$, is the well-known channel density classically. And let D_{dmaz} is dynamic channel density, it is updated during the routing process.

Definition 2.4 : The d_region is an interval (x_1, x_2) in the channel, where

$$\begin{aligned} d_j &= D_{dmaz}, x_1 \leq j \leq x_2; \\ x_1 &= L_i \text{ of some net } i; \\ x_2 &= \min_{\text{net } k \in S(x_1)} R_k. \end{aligned}$$

That is, the dynamically local density of each column in a d_region is D_{dmaz} . For example, in Figure 1, there are two $d_regions$, (5, 5) and (6, 7), and $D_{dmaz} = 5$.

Definition 2.5 : Let N_H (N_V) be the number of layers for horizontal (vertical) wires.

2.3 Graph Models of Channel Routing Problem

Two wiring conflicts, *vertical constraint* and *horizontal constraint*, are necessary to be avoided in the routing process. As the vertical wire of a net must be placed below the vertical wire of another net, there is a vertical constraint between these two nets. If the x_region of a net overlaps the x_region of another net, then there is a horizontal constraint between these two nets. Both graphs which are *vertical constraints graph* (VCG) and *horizontal constraints graph* (HCG) are used to describe the vertical and horizontal constraints among nets, respectively. These two graphs are described as follows.

(1) VCG :

Each node in VCG corresponds to a net. The directed edge from node A to node B means that net A must be placed below net B .

(2) HCG :

HCG and VCG have the same set of nodes. The edge with end nodes A and B means that two $x_regions$ of net A and net B are not overlapping each other. Each clique in HCG forms a group of nets which can be routed in the same track.

Definition 2.6 : Let $vmaz$ be the number of nodes on the longest directed path in VCG. And let D_{vmaz} is the dynamic $vmaz$, it is updated during the routing process.

Definition 2.7 : Let $depth_i$ be the longest directed path length from node i to some vertices which has no descendant in VCG.

Definition 2.8 : In a channel routing problem, the routing is said to be dogleg-free if the number of tracks on which any net's horizontal segments can be positioned is limited to one.

2.4 Lower Bound of Channel Width

The lower bound of channel width in HV alternate model can be expressed in terms of two parameters " $dmaz$ " and " $vmaz$ ". The lower bounds of the routing instances can be classified into the following two types.

(1) The number of vertical layers is only one:

Let the longest directed path in VCG is $n_1 \rightarrow n_2 \rightarrow \dots \rightarrow n_k$ ($k = vmaz$), no two nets among $n_1, n_2, \dots, n_k$ can be assigned on the same track. Therefore, the channel routing needs at least $vmaz$ horizontal tracks. In addition, as the horizontal segments are placed on N_H layers, in an ideal case the maximum density at each horizontal layer can be reduced to $\lceil dmaz/N_H \rceil$, so the number of tracks must also be equal to or larger than $\lceil dmaz/N_H \rceil$. We can conclude that the lower bound of channel width is equal to $\max(\lceil dmaz/N_H \rceil, vmaz)$.

(2) The number of vertical layers is more than one:

In this type there are two or more vertical layers, we can eliminate the major part of vertical constraints. The vertical constraints can be ignored theoretically, therefore, the channel width will have the lower bound equal to $\lceil dmaz/N_H \rceil$.

3 Mixed Mode Multilayer Channel Routing

As in Section 1 stated, the available layers will be divided into some partitions. In each partition, there are two or three layers (VH or HVH pattern) for routing the nets. In the simplest case, there is only one partition divided for routing. Of course, our router M^3CR can easily handle this basic case: the two- or three-layer channel routing.

3.1 Routing Algorithm for Two- and Three-Layer Channel Routing

As any vertical constraint exists, the ordering of nets to be routed can be derived from VCG if we route every net from the bottom track to the top track in the channel. The net A will be routed preceding the net B if there is a directed path from node A to node

B in VCG. Hence the nodes which have no ancestors in VCG (or all its ancestors have been routed) can be routed close to the bottom track (i.e., above the horizontal wires of assigned nets). And these nodes will construct a subgraph of HCG, denoted as H_{sub} . To avoid the horizontal conflicting in the routing, only nets with corresponding nodes that form a clique in the H_{sub} can be routed in the same track. The nodes in H_{sub} can form some cliques. In two-layer routing N_H is equal to one, therefore, the router selects a maximal weighted cost clique in H_{sub} to assign appropriate track for the nets corresponding to the nodes of the maximal weighted cost clique. The cost function will be defined and described in Subsection 3.3. In three-layer routing N_H is equal to two, proportionally, the router selects two maximal weighted cost cliques in H_{sub} . The nets corresponding to the nodes in first maximal weighted cost clique in H_{sub} will be assigned to the first horizontal layer and the others for second horizontal layer on the same track.

3.2 Routing Algorithm for Multilayer Channel Routing

In Chameleon [4], it divides the layers into some sets, where each set contains either two or three layers, and also partitions the nets into these sets. Each set of layers and nets are then solved by a detailed router YACR2. Our routing algorithm doesn't need a preprocessor to partition the nets into sets. And the dogleg-free routing model is employed by our router, the problem of channel routing is to determine a horizontal track number for each net.

3.2.1 Layer Partitioning

Our router divides layers by three cases. Let P be the number of partitions.

- Case 1 ($N \bmod 3 = 0$ and $N \geq 4$):
The case has $N/3$ HVH partitions as below.
HVH, HVH, ..., HVH ($P = N/3$)
- Case 2 ($N \bmod 3 = 1$ and $N \geq 4$):
The case has $\lfloor N/3 \rfloor - 1$ HVH partitions and two VH partitions as below.
HVH, HVH, ..., HVH, VH, VH ($P = \lfloor N/3 \rfloor + 1$)
- Case 3 ($N \bmod 3 = 2$ and $N \geq 4$):
The case has $\lfloor N/3 \rfloor$ HVH partitions and one VH partition as below.
HVH, HVH, ..., HVH, VH ($P = \lfloor N/3 \rfloor + 1$)

3.2.2 Free Horizontal Layer

In Chameleon [4], the routable layers are divided into some partitions, where each partition is independent, i.e., if net A is assigned in partition B , all vertical and horizontal segments of net A are entirely routed in partition B . In fact, vias are allowed between adjacent layers in the different partitions. Therefore, the vertical segments of net A can be routed in different vertical layers so long as these vertical layers are adjacent to the horizontal layer, where the horizontal segment of net A are routed. For example, the four-layer channel router arranges the four layers as $V_1 H_1 V_2 H_2$.

Obviously, if net A is assigned in partition 1, vertical segments of net A can be routed in V_1 (belongs to partition 1) and V_2 (belongs to partition 2).

A multilayer channel router which allows nets' vertical segments to be routed in different partitions is called a *mixed mode multilayer channel router*. The mixed mode multilayer channel router can utilize the vertical layers more effectively.

Definition 3.1 : A horizontal layer which is adjacent to two vertical layers is called a *free horizontal layer*.

Doglegs are allowed in Chameleon. Although doglegs can reduce the vertical constraints, but doglegs have the penalty of increasing the number of vias and the wire length. If VCG has directed cycles, the usage of doglegs may also break the directed cycles. But no published algorithm can guarantee that their introduced doglegs can break all directed cycles. If there are two or more vertical layers available, we can eliminate the major part of vertical constraints by using these vertical layers. And any directed cycle can be broken if any two nets in the directed cycle are routed in the different partitions. We can have some theorems described as follows.

Theorem 3.1 : If the horizontal segments of the nets are routed in the free horizontal layer, they have no vertical constraints among them.

Theorem 3.2 : If there is any free horizontal layer, all the directed cycles in VCG can be broken successfully, i.e., no cyclic constraint exists.

The proofs are omitted here. There are free horizontal layers in Case 2 and Case 3 (see Subsection 3.2.1). Therefore, doglegs are no longer used in our routing algorithm when two or more vertical layers are available. The lower bound of channel width will only depend on the channel density (dm_{max}) when there are two or more vertical layers.

3.2.3 The Algorithm

Let MC be the maximal weighted cost clique in HCG. Every time our router selects one MC to be assigned to an appropriate track, we hope that maximum density of the channel can be reduced by one after removing the corresponding nets from both the VCG and HCG. It is very clear that if our router can satisfy such condition in each processing, we can get the optimum solution consequently.

Our router divides layers into some partitions. Each partition contains either two or three layers. The VH pattern is used by two-layer partition. The HVH pattern is used by three-layer partition. Then our router treats all nodes in VCG as candidates to find the corresponding MC for each partition. In two-layer partition it selects one MC. In three-layer partition it selects two MCs. Two types of constraints are introduced after router assigns appropriate track for the nets corresponding to the nodes of the MC. The first constraint is called *father_constraint*.

Definition 3.2 : If MC_i is assigned in the j -th partition, the nodes which are the fathers of the nodes of MC_i in VCG cannot be routed in the j -th partition. We define that such nodes have *father_constraints*.

During the routing process, the node with father_constraint in j -th partition are therefore assigned an j -color for avoiding the conflicts of such constraints. Next time when router finds MC(s) for the j -th partition, it treats all nodes as candidates in VCG except those nodes with j -color. Note that a node in VCG may have more than one father_constraints if there are two or more partitions divided.

The second constraint is called *partition_constraint*. **Definition 3.3** : If an unassigned net has no partition for track assignment after we assign an j -color to itself, we define that this node has the partition_constraint.

If any node has partition_constraint in j -th partition, router must remove such node's children in VCG from candidates and find new MC(s) for enforcing no partition_constraint occurred. We summarise the algorithm as follows.

ALGORITHM #3.1

```

Step 1: input the TOP and BOT lists and construct
        the corresponding HCG and VCG.
Step 2: divide layers into some partitions.
Step 3: ROUTE_NORMAL_H(LBN).
Step 4: ROUTE_FREE_H(LBN).
Step 5: while HCG and VCG are not null do
Step 6:     ROUTE_NORMAL_H(1),
Step 7:     ROUTE_FREE_H(1).
Step 8: end
END ALGORITHM #3.1

```

PROCEDURE ROUTE_NORMAL_H(num_of_tracks)

```

Step 1: for j=1 to num_of_tracks do
Step 2:     for i=1 to P do
Step 3:         treat all nodes as candidates in VCG except
                those nodes with  $i$ -color.
Step 4:         if any unassigned net has partition_constraint
                in the current partition
                then remove the corresponding node's
                children from candidates.
Step 5:         select the corresponding MC(s) for horizontal
                layer(s) except free horizontal layer(s).
Step 6:         assign the nodes which have father_constraints
                an  $i$ -color.
Step 7:         assign appropriate track for the nets
                corresponding to the nodes of the MC(s).
Step 8:         remove the nodes of the MC(s) from HCG
                and VCG.
Step 9:         update cost function.
Step 10:        if HCG and VCG are both null then exit.
Step 11:    end
Step 12: end
Step 13: return.
END PROCEDURE ROUTE_NORMAL_H

```

PROCEDURE ROUTE_FREE_H(num_of_tracks)

```

Step 1: for j=1 to num_of_tracks do
Step 2:     for all free horizontal layers do
Step 3:         treat all nodes in VCG as candidates.
Step 4:         select the corresponding MC for the free
                horizontal layer
Step 5:         assign appropriate track for the nets
                corresponding to the nodes of the MC.
Step 6:         remove the nodes of the MC from HCG and
                VCG.
Step 7:         update cost functions.
Step 8:         if HCG and VCG are both null then exit.

```

```

Step 9:     end
Step 10: end
Step 11: return.
END PROCEDURE ROUTE_FREE_H

```

3.3 Cost Function

The cost function of a net is discussed in two conditions.

(1) Condition 1 ($N_V=1$):

The cost estimated on each nodes in VCG does not only involve the x -length of the corresponding net but also consider its longest path length in VCG. The component, x -length, of cost function aims for maximising the occupied space of horizontal track. The v_{maz} is a lower bound of channel width in this case. Therefore, for any net i , the $depth_i$ is an important component of cost function. The d -region is an crucial interval. If all x -regions of nets of an MC can cover all d -regions, the D_{dmas} can be reduced by one. Let nd_i be the number of d -region covered by x -region of a net i . And let $cost_d_i$ be equal to nd_i multiplied by 500. The cost function for a net i can be summarized as follows.

- (a) if ($D_{vmas} * N_H \geq D_{dmas}$ and $depth_i = D_{vmas}$) then
 $cost(i) = 15000 + w_1 * depth_i + w_2 * (R_i - L_i)$;
- (b) if ($D_{vmas} * N_H \geq D_{dmas}$ and $depth_i < D_{vmas}$) then
 $cost(i) = w_1 * depth_i + w_2 * (R_i - L_i)$;
- (c) if ($D_{vmas} * N_H < D_{dmas}$) then
 $cost(i) = w_1 * depth_i + w_2 * (R_i - L_i) + w_3 * cost_d_i$;

where w_1, w_2 and w_3 are weighted parameters which can be adjusted for each experiment. By default, $w_1 = 20$, $w_2 = 1$ and $w_3 = 1$. And the numbers, 500 and 15000, are experienced constants to enforce the influences of the corresponding situations.

(2) Condition 2 ($N_V \geq 2$):

In this case v_{maz} is no longer a lower bound of channel width. The $depth_i$ of any net i can be removed from cost function in Condition 1. The lower bound of channel width only depends on d_{maz} now. Every time the router selects an MC, it is important to reduced D_{dmas} by one. Therefore, the component, $cost_d_i$ of any net i , of cost function is very essential. In routing process we must keep the number of nodes which have father_constraints as small as possible. Hence, we give a penalty in cost function to reduce father_constraints. Let $indegree_i$ be the number of fathers of a net i in VCG. The cost function for a net i can be described as follows.
 $cost(i) = a_1 * (R_i - L_i) + cost_d_i - a_2 * indegree_i$;
 where a_1 and a_2 are weighted parameters which can be adjusted for each experiment. By default, $a_1 = 1$ and $a_2 = 50$.

3.4 Maximal Weighted Cost Clique Algorithm

The maximal weighted cost clique problem in the general graph is NP-complete. However, the problem

becomes tractable when the graph is a comparability graph [7]. We define the comparability graph and prove that HCG is a comparability graph as follows.

Definition 3.4 : A directed edge $\langle i, j \rangle$ in directed graph is an edge connecting from node i to node j .

Definition 3.5 : In undirected graph an edge connecting node i and node j is denoted by (i, j) or (j, i) .

Definition 3.6 : An undirected graph $G = (V, E)$ is a comparability graph if there exists an orientation (V, F) of G satisfying the following transitive condition :

$\langle a, b \rangle \in F$ and $\langle b, c \rangle \in F$ imply $\langle a, c \rangle \in F$,
for every $(a, b), (b, c), (a, c) \in E$.

Theorem 3.3 : The horizontal constraints graph (HCG) is a comparability graph. (The proof is omitted.)

Theorem 3.4 : The time complexity of the algorithm for finding a maximal weighted cost clique in $HCG=(V, E)$ is linear in the size of HCG. That is, the time complexity is $O(|V|)$.

Proof : The time complexity of finding a maximal weighted cost clique in a comparability graph is linear in size of itself [7]. According to *Theorem 3.3*, the proof is completed, therefore. #

Theorem 3.5 : The time complexity of ALGORITHM #3.1 is $O(L * dmaz)$, where L is the channel's length.

Proof : The crucial factor in the analysis of our algorithm is the time required to search MCs and update cost functions. The time for searching an MC is $O(n)$ known from *Theorem 3.4*, where n is the number of nodes in VCG and also is the number of nets. The time for updating cost functions include the reconstructions of d_region and VCG. The reconstruction of d_region takes $O(L)$ time. The reconstruction of $VCG=(V, E)$ takes $O(|E|)$ time. There are L vertical constraints in the worst case. One edge in VCG represents one vertical constraint so that $O(|E|) \leq O(L)$. We conclude that $O(L)$ time is required to update cost functions. The algorithm may perform searching MCs and updating cost functions for $O(dmaz + c)$ times, where c is a small constant. If the router produces any optimal solution, c is equal to zero. Therefore, the time complexity of ALGORITHM #3.1 is $O((n + L) * (dmaz + c)) \doteq O((n + L) * dmaz) \doteq O(L * dmaz)$. #

3.5 Variable-Wire-Width Consideration According to Design Rules

As more available routing layers are used, the minimum wire width and the size of the vias are increased as in top layers due to the design rules; thus, even if a channel can be routed with fewest tracks, the actual routing area is still large providing that the worst case design rules in topmost layer are employed. We suggest that the wire width and the size of the vias are invariant for the layers in a certain partition, but they are variant in the different partitions. The variable-wire-width approach can take the advantages of saving routing area and considering the performance of the circuits, e.g., critical nets were routed on the top lay-

Figure 2: An example to illustrate the advantage of variable-wire-width.

ers, where the wire width is wider so that the wires have lower resistivity.

Figure 2 is used to illustrate how the approach saves the routing area. We suppose that a simple channel requires 4 tracks to complete the routing problem in two-layer channel routing as shown in Figure 2(a). In four-layer channel routing, there are two partitions, where partition 2 has the minimum track width which is equal to 3 times of the track width of partition 1. If the router distributes 4 tracks equally to two partitions, i.e., each partition contains 2 tracks as shown in Figure 2(b), the channel width is larger than the one as shown in Figure 2(c). From the discussions above, our router can be easily modified to handle such consideration. We define that the minimum *track-to-track* spacing of j -th partition is $unit_j$, and the *height_j* of j th partition is equal to $unit_j$ multiplied by the number of tracks used in j -th partition. Our router tries to uniformly distribute the nets to all partitions, and makes each partition with the same *height_j* as possible.

4 Experimental Results

The Mixed Mode Multilayer Channel Router (M^3CR) was programmed in C language and implemented on the SUN4 workstation. We use the same examples as in [8]. The router generates optimum results in all test examples from 4- through 8-layer. Also the running time for each test example is very short (from 0.03 to 1.27 seconds) because no maze-running algorithm is invoked. Table 1 - Table 5 show the routing results from 4- through 8-layer, respectively. The well-known *Deutsch's difficult example* is denoted as *diff*. Table 6 and Table 7 show some results produced by the variable-wire-width approach.

Table 8 shows the comparison with Chameleon [4]. For all the examples in Table 8, our router shows equal results of Chameleon. The authors of Chameleon declared that they can use minimum number of tracks. However, the lower bound of channel width derived by them is not proper since wire overlaps are allowed in their solutions. From the theoretical point of view, the results of Chameleon are not very surprising. Moreover, the doglegs are also introduced in their solutions, and more vias and longer wire length are used, therefore. From the view of circuit performance, our router

is expectably superior to Chameleon. Further M^3CR with time complexity $O(L \cdot dmaz)$ will be much faster than other maze-running-based routers.

5 Conclusions

A new efficient approaches to multilayer channel routing problem was developed in this paper. Due to the dogleg-free wiring model employed by M^3CR , the router not only minimizes the number of tracks but also uses less number of vias as well as shorter wire length. The router doesn't need a preprocessor to partition the nets and the maze router as done in Chameleon [4]. In short words, if router invokes a maze-running algorithm in the routing algorithm, the routing algorithm is becoming more complex and requires more running time. The time complexity of M^3CR is $O(L \cdot dmaz)$, where L is the channel's length and $dmaz$ is the channel density. The router generated optimum results in all test examples from 4- through 8-layer. Additionally the variable-wire-width consideration according to the design rules was discussed and resolved for more compacted channels under the practical multilayer design environments.

Acknowledgements

Here we want to express our gratitude to National Science Council, Taipei, Taiwan, R.O.C., for the supports of this work.

References

- [1] R. Enbody and H. Du, "Near-optimal n-layer channel routing," *Proc. 23rd Design Automation Conference*, Las Vegas, NV, pp. 708-714, June 1986.
- [2] S. Hambrush, "Channel routing algorithms for overlap models," *IEEE Trans. on Computer-Aided Design*, vol. CAD-4, no.1, pp. 23-30, January 1985.
- [3] J. S. Cong, D. F. Wong and C. L. Liu, "A new approach to three- or four-Layer channel routing," *IEEE Trans. on Computer-Aided Design*, vol. CAD-7, no.10, pp. 1094-1103, October 1988.
- [4] D. Braun, J. L. Burns, S. Devadas, H. K. Ma, K. Mayaram, F. Romeo, and A. Sangiovanni-Vincentelli, "Chameleon: A new multi-layer channel router," *Proc. 23rd Design Automation Conference*, Las Vegas, NV, pp. 495-502, June 1986.
- [5] J. Reed, A. Sangiovanni-Vincentelli, and M. Santomauro, "A new symbolic channel router: YACR2," *IEEE Trans. on Computer-Aided Design*, vol. CAD-4, no.3, pp. 208-219, July 1985.
- [6] D. N. Deutsch, "A dogleg channel router," *Proc. 13rd Design Automation Conference*, pp. 425-433, 1976.
- [7] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, U.S.A., 1980, pp. 133-134.
- [8] Y. K. Chen and M. L. Liu, "Three-Layer Channel Routing," *IEEE Trans. on Computer-Aided Design*, vol. CAD-3, no.4, pp. 156-163, April 1984.

Table 1: Four-layer channel routing results in mixed mode

Example	dmaz	LB ₄	#Track	#Via	Wire length	Time(sec)
ex1	12	5	5	57	513	0.07
ex3a	15	5	5	68	858	0.15
ex3b	17	5	5	107	1363	0.35
ex3c	18	5	5	125	1727	0.48
ex4b	17	5	5	179	2205	0.47
ex5	20	10	10	150	2200	0.45
diff	19	10	10	290	4100	0.78

Table 2: Five-layer channel routing results in mixed mode

Example	dmaz	LB ₅	#Track	#Via	Wire length	Time(sec)
ex1	12	4	4	57	459	0.07
ex3a	15	5	5	68	782	0.18
ex3b	17	5	5	107	1192	0.37
ex3c	18	5	5	125	1558	0.45
ex4b	17	5	5	179	1902	0.43
ex5	20	7	7	150	2013	0.47
diff	19	7	7	290	3597	1.00

Table 3: Six-layer channel routing results in mixed mode

Example	dmaz	LB ₆	#Track	#Via	Wire length	Time(sec)
ex1	12	3	3	57	407	0.07
ex3a	15	4	4	68	707	0.27
ex3b	17	5	5	107	1127	0.32
ex3c	18	5	5	125	1494	0.37
ex4b	17	5	5	179	1859	0.38
ex5	20	5	5	150	1870	0.87
diff	19	5	5	290	3360	1.27

Table 4: Seven-layer channel routing results in mixed mode

Example	dmaz	LB ₇	#Track	#Via	Wire length	Time(sec)
ex1	12	3	3	57	429	0.03
ex3a	15	4	4	68	723	0.20
ex3b	17	5	5	107	1127	0.37
ex3c	18	5	5	125	1474	0.45
ex4b	17	5	5	179	1882	0.42
ex5	20	5	5	150	1666	0.45
diff	19	5	5	290	3451	0.90

Table 5: Eight-layer channel routing results in mixed mode

Example	dmaz	LB ₈	#Track	#Via	Wire length	Time(sec)
ex1	12	3	3	57	415	0.07
ex3a	15	3	3	68	679	0.20
ex3b	17	4	4	107	1059	0.40
ex3c	18	4	4	125	1418	0.53
ex4b	17	4	4	179	1732	0.50
ex5	20	4	4	150	1774	0.53
diff	19	4	4	290	3255	1.03

Table 6: Four-layer channel routing results by variable-wire-width approach

Example	dmaz	height ₁ /height ₂	height ₁ ⁽¹⁾	R ⁽²⁾
ex3a	15	10u/10u	18u	38%
ex3b	17	12u/12u	18u	33%
diff	19	14u/14u	20u	30%

- (1) The height₁ is the minimum channel width used by uniform approach.
(2) The R is the reduced area ratio equal to

$$(height_1 - \max_{i=1}^P height_i) / height_1$$

Table 7: Seven-layer channel routing results by variable-wire-width approach

Example	dmaz	height ₁ /height ₂ /height ₃	height ₁	R
ex3a	15	6u/4u/4u	16u	63%
ex3b	17	7u/6u/4u	20u	65%
diff	19	10u/10u/8u	20u	50%

Table 8: Comparison with Chameleon

Example	dmaz	Comparison with Chameleon for some benchmarks			
		Four-Layer		Five-Layer	
		#Track	#Via	#Track	#Via
ex3a	15	5	5	5	4
ex3b	17	5	5	6	5
ex3c	18	5	5	6	5
diff	19	10	10	7	5