4-1993

Topology Based Routing Algorithms for Multilayer IC Layout Design

Moazzem Hossain
Western Michigan University

Follow this and additional works at: http://scholarworks.wmich.edu/masters_theses
Part of the Computer Sciences Commons

Recommended Citation
http://scholarworks.wmich.edu/masters_theses/774

This Masters Thesis-Open Access is brought to you for free and open access by the Graduate College at ScholarWorks at WMU. It has been accepted for inclusion in Master's Theses by an authorized administrator of ScholarWorks at WMU. For more information, please contact maira.bundza@wmich.edu.
TOPOLOGY BASED ROUTING ALGORITHMS FOR MULTILAYER IC LAYOUT DESIGN

by

Moazzem Hossain

A Thesis
Submitted to the
Faculty of The Graduate College
in partial fulfillment of the
requirements for the
Degree of Master of Science
Department of Computer Science

Western Michigan University
Kalamazoo, Michigan
April 1993
TOPOLOGY BASED ROUTING ALGORITHMS FOR MULTILAYER IC LAYOUT DESIGN

Moazzem Hossain, M.S.
Western Michigan University, 1993

In this thesis, we consider multilayer topological routing problem and the geometric routing problem based on topological routing solution. We present a provably-good approximation algorithm for the multilayer topological planar routing problem for different routing regions. Our algorithm can always find a solution whose weight is at least $1 - \frac{1}{e} \approx 63.2\%$ of the weight of an optimal solution. When the number of routing layers is fixed, we have even tighter performance bounds.

We also propose a graph theoretic algorithm to find 2-layer topological solutions that are guaranteed to be transformable to geometric routing solution. The basic idea is to develop topological routing such that the congestions over the entire routing surface are uniform. An iterative algorithm is then applied to this uniform topological solution to obtain a geometric routing. Our experimental results show that a significant reduction of vias is achieved by the router.
ACKNOWLEDGMENTS

First and foremost, I would like to thank Professor Naveed A. Sherwani, my advisor and thesis committee chairman, for his counsel, his encouragement, his trust, his unlimited patience and generosity, and for the countless hours he spent with me. Undoubtedly, it is a privilege to have been a student of such a distinguished and experienced computer scientist who possesses a brilliant style. I have seldom seen, in my life, individuals with that capacity of forgiveness, respect, and faith in each human being as Professor Sherwani possesses. For the appreciation of computer science knowledge that he has given to me, I am grateful. It has been a great honor to work with him and to know him as a friend.

Due to the outstanding activity that Professor Sherwani does to promote, create, and conduct research groups within the 'nitegroup', I have received enormous benefits by having the opportunity to work together with several computer scientists and mathematicians and to participate in many conferences.

The Financial assistantship and the research facilities provided by the Department of Computer Science has been a great help in finishing this thesis work and conducting other research. I would like to thank Professor Donald Nelson, the Department Chair, for giving me the maximum help he could.

I am grateful to the members of my thesis committee, Professor Alfred Boals and Professor Ajay Gupta, who accepted this additional task with high enthusiasm.

Special thanks to Professor Gary Chartrand and Professor Joseph Mc-Canna, Department of Mathematics and Statistics, for giving me a unique opportunity to work with them.
I would like to thank the 'nitegroup' members, Wasim Khan, Aizaz Manzar, Arun Shanbhag, Qiong Yu, Anand Panyam, and Chandra Shekar, for their constant help, cooperation, and friendship. It has been an unique opportunity to be a member of the 'nitegroup'. Special thanks to Rikio Ichishima for sharing, in many occasions, his mathematical knowledge with me. I would also like to thank all my friends at Western, who, one in way or the other, made my study here an enjoyable one.

Special thanks to Department Secretaries, Susan Moorian and Phyllis Wolf, who, with humor and friendship, helped me in many situations.

Finally, I thank my parents for showing me the secret of accomplishing goals - hard work.

Moazzem Hossain
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI
University Microfilms International
A Bell & Howell Information Company
300 North Zeeb Road. Ann Arbor, MI 48106-1346 USA
313:761-4700 800:521-0500

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Topology based routing algorithms for multilayer IC layout design

Hossain, Moazzem, M.S.

Western Michigan University, 1993
TABLE OF CONTENTS

ACKNOWLEDGMENTS ................................................. ii
LIST OF TABLES ................................................... v
LIST OF FIGURES ................................................... vi
I. INTRODUCTION ................................................. 1
II. APPROXIMATION ALGORITHM FOR $k$-TPR ............. 6
   2.1 Approximation Result for $k$-TVM Problem .......... 10
III. ROUTING ALGORITHM BASED ON TOPOLOGICAL SOLUTION 12
   3.1 Overview of the Algorithm TGR2_ROUTE ............ 13
      3.1.1 Phase 1: Solution for 2-RTVM Problem ........ 13
      3.1.2 Phase 2: Topology to Geometric Routing ...... 19
   3.2 Operations .................................................. 20
   3.3 Algorithm ................................................... 21
   3.4 Extensions .................................................. 24
      3.4.1 Multilayer Topology Based Routing .......... 24
      3.4.2 Avoiding Vias in Critical Nets .......... 25
IV. EXPERIMENTAL RESULTS ..................................... 27
   4.1 Iterative-Peeling Algorithm ............................ 27
   4.2 TRG2_ROUTE Algorithm ................................ 28
V. CONCLUSION ................................................... 31
REFERENCES ..................................................... 32
LIST OF TABLES

1  Performance Ratio of the Iterative-peeling Algorithm .................. 10

2  Percentage of Nets Completed by Planar Routing Using Iterative-
    peeling Algorithm for Different Number of Routing Layers ........ 28

3  Comparison Between BEAVER and TRG2 ROUTE Algorithm on Some
    Switchbox Routing Examples ........................................... 29
LIST OF FIGURES

1. Example of Panels and Regions. ........................................... 13
2. Example of Region Boundary Net Segments and Capacities. ....... 15
3. A Net Segment Assigned to a Region Without Via. .................... 16
4. A Net Segment Assigned to a Region Using Via. ....................... 16
5. Routing of a Net by Assigning Net Segment to Regions. ............. 17
6. Example of Regions and Region Adjacency Graph. .................... 18
7. Assigning Grids to Net Segments in a Region. .......................... 26
CHAPTER I

INTRODUCTION

Advances in VLSI fabrication technology have made it possible to use more than two routing layers for interconnections. Many VLSI chips have been designed using three or more metal layers for routing in recent years. For example, Motorola 2900ETL macrocell array is a bipolar gate array with three metal layers for routing. Several algorithms have been proposed for multilayer routing problems (for example, [5, 12]). The primary goal of these approaches was to reduce the total routing area. In this paper, we shall study the multilayer topological planar routing (k-TPR) problem. The objective is to choose a maximum (weighted) set of nets so that each net in the set can be topologically routed entirely on one of the given layers without crossing other nets. Our research on the k-TPR problem is motivated by the following applications:

1. In the layout design of multilayer IC circuits, we usually want to compute a planar routing solution for each of the critical nets (such as the power and ground nets and the clock nets) so that these nets can be routed on their reserved layers.

2. A good topological planar routing solution reduces the number of vias used in the final layout. For high-performance circuits, it is known that vias not only increase the manufacturing cost but also degrade that system performance since they form inductive and capacitive discontinuities and cause reflections when the wires have to be modeled as transmission lines [2].

3. If we can topologically route the majority of the nets on a single layer, the detailed routing problem in multilayer IC designs is greatly simplified. We can carry out planar routing for each layer independently and several effective methods (such as rubber-band routing [13, 19], have been developed for the planar routing problem.
Another important application of multilayer planar routing arises in the design of multichip modules (MCMs) and high density printed circuit boards (PCBs), where many more routing layers are used for interconnections. For example, the MCM developed for the IBM 3081 mainframe has 33 layers of molybdenum conductors (including 1 bonding layer, 5 distribution layers, 16 interconnection layers, 8 voltage reference layers, and 3 power distribution layers. Fujitsu's latest supercomputer, the VP-2000, uses ceramic PCBs with over 50 interconnection layers.

All these applications require efficient solutions to the $k$-TPR problem. The classical 2-TPR problem has been studied extensively by CAD researchers because of its practical and theoretical importance. Unfortunately, solving the $k$-TPR problem is NP-complete [8]. The problem remains NP-complete even when the routing region is restricted to a two-layer switchbox [26]. Polynomial time optimal solutions to the $k$-TPR problem were developed for a special type of channels, called crossing channels. The unweighted case for crossing channels were solved by Rim, Kashiwabara, and Nakajima [25]. The general weighted case for crossing channels were solved by Cong and Liu [8] and by Sarrafzadeh and Lou [28] independently. However, there is no effective solution to the $k$-TPR for general routing regions.

Due to the NP-completeness of the general $k$-TPR problem, many heuristic algorithms have been proposed [8, 24]. All these algorithms are restricted to some specific routing regions; for example, in [24], topological routing for switchboxes was considered; similarly in cite [25], topological channel routing problem was considered. Algorithms for a specific region routing problems are usually not effective for other routing regions. Furthermore, to the best of our knowledge, there is no known solution for multilayer planar routing problem in arbitrary routing region.
Another important problem, namely via minimization problem, is directly related to the topological planar routing problem. Via minimization problem has been studied from two different points of views [8, 16, 20, 24, 29]. Constrained via minimization (CVM) problem considers a pre-routed topology, i.e., given the location of all wire segments and each possible via, the problem is to find a layer assignment for each wire segment without changing the topology so that the final routing uses minimum number of vias [20]. Topological via minimization (TVM) problem (also known as unconstrained via minimization problem) concerns with finding a topological routing and layer assignment of wires so that the total number of vias are minimized [1, 8, 16, 22, 24, 25, 26, 27, 28, 29, 31]. The physical dimensions of the wires, pins and vias are not considered in topological routing. The general TVM problem in $k$ layers ($k$-TVM) may be stated as follows: Given a set of nets, number of layers $k$ and terminal locations, find a $k$-layer topological routing solution that completes the interconnections of all nets using the minimum number of vias. In weighted version of $k$-TVM problem ($k$-WTVM), each net $n_i \in N$ is assigned a positive weight $w(n_i)$ which is a measure of the priority of the net. The weight of a via represents the weight of the corresponding net. The problem is to minimize the total weight of vias used in the routing.

The TVM problem was introduced by Hsu in [16], and it was conjectured that TVM problem is NP-hard. Recently Hsu's conjecture has been proven to be true [26]. Due to the computational difficulty of the general TVM problem, several special classes of the problem have been considered. One of the special classes of TVM problem is known as two-shore channel TVM problem [26]. In two-shore channel TVM problem, the routing region is a two-sided channel, represented by two horizontal lines, called upper and lower shores, respectively, in the $XY$-plane. A two-terminal net $n_t = (a_t, b_t)$ has two terminals, one on the upper shore at $X = a_t$ and the other on the lower shore at $Y = b_t$. Two-shore channel $k$-TVM and $k$-WTVM problems are solvable in polynomial time [8, 22, 25, 26, 28]. In [26]
an $O(n^2)$ time complexity algorithm is presented for the two-shore channel 2-WTVM problem, where $n$ is the total number of nets. An optimal $\Theta(n \log n)$ time complexity algorithm for two-shore channel 2-TVM problem is presented in [22]. Two-shore channel $k$-WTVM has been solved in $O(kn^3)$ time [8]. This algorithm was improved to $O(kn^2)$ in [28]. Multi-layer TVM problem was considered in [29] and it was shown that if the terminals are preassigned to layers, then the problem can be solved in $O(kn^2)$ time, where $k$ is the maximum number of terminals of a net in a single layer and $n$ is the total number of terminals. In [23], the problem of transforming single layer topological routing into geometric routing was considered.

In view of NP-Hardness of $k$-TVM, it is important to develop approximation algorithms for $k$-TVM problem with tight performance bounds. Since it appears that CVM problem does not offer enough flexibility for via minimization, we believe that topological routing offers a good starting point as vias are already minimized. However, we have observed that minimum-via topological routing often uses very long wires for many nets and causes high congestions in the routing region. Since the geometric routing problem has fixed area routing region, it may not be possible to transform a high congestion topological routing solution to geometric routing solution. Therefore, we need a topological routing solution that is guaranteed to be transformable into an actual geometric routing solution. This can be achieved by allowing some extra via's to keep the topology as close to the actual geometric solution as possible so that the final topological routing solution can be easily transformed into actual geometric routing solution. We denote this problem as routable topological via minimization problem in $k$ layers ($k$-RTVM). The major difference between the solutions of TVM and RTVM problems is that the solution of RTVM problem can be transformable into actual geometric routing solution.
In this thesis, we consider both $k$-TPR problem and geometric routing problem based on topological routing solution. We first present an approximation algorithm for the multilayer topological planar routing problem which is applicable to switchboxes (or arbitrary rectilinear polygons), channels (including L-shaped and staircase channels), and general routing regions [10, 11]. Our algorithm can always find a solution whose weight is at least $1 - \frac{1}{6} \approx 63.2\%$ of the weight of an optimal solution. The algorithm works for multi-terminal nets and arbitrary number of routing layers. When the number of routing layers is fixed, we have even tighter performance bounds. In particular, the performance-ratio of the algorithm is at least 75\% if there are two routing layers, and is at least 70.4\% if there are three routing layers. According to Lemma 1 in [24], these results also lead to provably good solutions to the multilayer topological via minimization problem. Then we propose a graph-theoretic algorithm for geometric routing problem based on topological solution [17]. The algorithm consists of two different phases. The first phase of the algorithm finds a solution to 2-RTVM problem. In the second phase, the solution to 2-RTVM problem is transformed into actual geometric routing. The proposed algorithm has been implemented in C on SUN SPARC station 1+ and has been tested on many routing problems. The experimental results show that a significant reduction of vias is achieved by the router.
CHAPTER II

APPROXIMATION ALGORITHM FOR K-TPR

Given a number of routing layers, the multilayer topological planar routing problem is to choose a maximum (weighted) set of nets so that each net can be topologically routed entirely on one of the given layers without crossing other nets.

In this section, we use maximum weighted planar subset to present our unified algorithm to compute to approximate maximum k-planar subset of nets in a routing region. The algorithm is conceptually simple. Let $N$ be the set of nets to be routed. First we choose a maximum weighted planar subset $N_1$ from $N$ and assign to layer 1. Then, we choose a maximum weighted planar subset $N_2$ from remaining nets $N - N_1$ and assign $N_2$ to layer 2, and so on. At the i-th step, we choose a maximum weighted planar subset $N_i$ from $N - N_1 \cup N_2 \cup \ldots \cup N_{i-1}$. We repeat this process $k$ time. Clearly, at the end $N_1 \cup N_2 \cup \ldots \cup N_k$ forms a $k$-planar subset, because each $N_i$ is a planar subset by construction, and the union of $k$ such subsets forms a $k$-planar subset by definition. Since at each iteration we 'peel' off a maximum weighted planar subset from the remaining nets, we call this algorithm the iterative-peeling algorithm. Assuming that we have a procedure $\text{max\_planar\_subset}(N)$ to compute a maximum weighted planar subset from a set of nets $N$, we can describe the iterative-peeling algorithm formally as follows:

Algorithm iterative-peeling;
1. $N' = N$;
2. for $i = 1$ to $k$ do
   $N_i = \text{max\_planar\_subset}(N')$;
   $N' = N' - N_i$;
3. return $N_1 \cup N_2 \cup \ldots \cup N_k$;
end.
Although the iterative-peeling algorithm is greedy in nature, we are able to show that it has a good performance ratio. In fact, for any arbitrary $k$, the algorithm has a performance ratio of at least 63.2%. The lower bound on the performance ratio is established based on the following results:

**Lemma 1** Let $\Psi^*_k$ be the weight of the optimal solution of maximum $k$-TPR problem. Let $\psi_i$ be the weight of the subset $N_i$ ($1 \leq i \leq k$) chosen by the iterative-peeling algorithm at the $i$-th iteration. Then we have

\[
\psi_1 \geq \frac{\Psi^*_k}{k} \\
\psi_2 \geq \frac{\Psi^*_k - \psi_1}{k} \\
\psi_3 \geq \frac{\Psi^*_k - (\psi_1 + \psi_2)}{k} \\
\vdots \\
\psi_k \geq \frac{\Psi^*_k - (\psi_1 + \psi_2 + \ldots + \psi_{k-1})}{k}
\]

**Proof:** Let $N$ be the entire set of nets in the problem. Let $N^* = N_1^* \cup N_2^* \cup \ldots \cup N_k^*$ be an optimal solution to the $k$-TPR problem, where each $N_i^*$ is a planar subset and $N_i^*$'s are pairwise disjoint. At the end of $(i - 1)$-th iteration of the iterative-peeling algorithm, the set of remaining nets is

\[
N' = N - \bigcup_{j=1}^{i-1} N_j
\]

Note that

\[
|N_1^* \cap N'| + |N_2^* \cap N'| + \ldots + |N_k^* \cap N'| = |N^* \cap N'|
\]

\[
= |N^* \cap (N - (N_1 \cup N_2 \cup \ldots \cup N_{i-1}))|
\]

\[
\geq |N^* - (N^* \cap (N_1 \cup N_2 \cup \ldots \cup N_{i-1}))|
\]

\[
\geq |N^*| - |(N_1 \cup N_2 \cup \ldots \cup N_{i-1})|
\]

\[
= \Psi^*_k - (\psi_1 + \psi_2 + \ldots + \psi_{i-1})
\]
By pegeonhole's principle, there exist at least a $j$ \((1 \leq j \leq k)\) such that
\[
|N_j \cap N'| \geq \frac{\Psi_k - (\psi_1 + \psi_2 + \ldots + \psi_{i-1})}{k}
\]
Since, $N_j \cap N' \subseteq N'$ is a planar subset and $N_i \subseteq N'$ is a maximum planar subset of $N'$ according to the iterative-peeling algorithm, we have
\[
\psi_i = |N_i| \geq |N_j \cap N'| \geq \frac{\Psi_k - (\psi_1 + \psi_2 + \ldots + \psi_{i-1})}{k}
\]
\[
\square
\]

**Lemma 2** Let $\Psi_k = \psi_1 + \psi_2 + \ldots + \psi_{i-1}$, where $\psi_i$ is the weight of the subset $N_i$ produced by the iterative-peeling algorithm. Then, we have
\[
\Psi_k \geq [1 - (1 - \frac{1}{k})^k]\Psi_k^*
\]

**Proof:** Let
\[
x_1 = \frac{\Psi_k^*}{k}
\]
\[
x_2 = \frac{\Psi_k^* - x_1}{k}
\]
\[
x_3 = \frac{\Psi_k^* - (x_1 + x_2)}{k}
\]
\[
\vdots
\]
\[
x_k = \frac{\Psi_k^* - (x_1 + x_2 + \ldots + x_{k-1})}{k}
\]
Then, it is easy to show that
\[
x_i = \frac{k - 1}{k} x_{i-1} = \frac{(k - 1)^2}{k^2} x_{i-2} = \ldots = \frac{(k - 1)^{i-1}}{k^{i-1}} x_1 = \frac{(k - 1)^{i-1} \Psi_k^*}{k^{i-1}}
\]
Therefore, we have
\[
\sum_{i=1}^{k} x_i = \sum_{i=1}^{k} \frac{(k - 1)^{i-1} \Psi_k^*}{k^{i-1}} = [1 - (1 - \frac{1}{k})^k]\Psi_k^*
\]
Now we show by induction that
\[
\sum_{i=1}^{l} \psi_i \geq \sum_{i=1}^{l} x_i \quad (2.1)
\]
holds for every $1 \leq l \leq k$. When $l = 1$, the inequality (2.1) holds since $\psi_1 = x_1$.

Assume that inequality (2.1) holds for $l$. Then for $l + 1$, we have

$$
\begin{align*}
\sum_{i=1}^{l+1} \psi_i \\
\geq \sum_{i=1}^l \psi_i + \frac{\psi_k^* - \sum_{i=1}^l \psi_i}{k} \\
= \frac{\psi_k^* + (k - 1) \sum_{i=1}^l \psi_i}{k} \\
\geq \frac{\psi_k^* + (k - 1) \sum_{i=1}^l x_i}{k} \\
= \sum_{i=1}^l x_i + \frac{\psi_k^* - (k - 1) \sum_{i=1}^l x_i}{k} \\
= \sum_{i=1}^{l+1} x_i
\end{align*}
$$

Therefore, we have

$$
\psi_k = \sum_{i=1}^k \psi_i \geq \sum_{i=1}^k x_i = [1 - (1 - \frac{1}{k})^k] \psi_k^*
$$

\hspace{1cm} \square

From the above two lemmas, we can conclude the following:

**Theorem 1** Let $\rho_k$ be the performance ratio of the iterative-peeling algorithm for $k$-planar subset problem. Then,

$$
\rho_k \geq 1 - (1 - \frac{1}{k})^k
$$

\hspace{1cm} \square

It is easy to see that the function $f(x) = 1 - (1 - \frac{1}{x})^x$ is a decreasing function. Moreover,

$$
\lim_{x \to \infty} [1 - (1 - \frac{1}{x})^x] = 1 - e^{-1}
$$

where $e \approx 2.718$. Therefore, we have
Table 1
Performance Ratio of the Iterative-peeling Algorithm

<table>
<thead>
<tr>
<th>no. of layers</th>
<th>performance ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>75%</td>
</tr>
<tr>
<td>3</td>
<td>70.4%</td>
</tr>
<tr>
<td>4</td>
<td>68.4%</td>
</tr>
<tr>
<td>5</td>
<td>67.3%</td>
</tr>
<tr>
<td>∞</td>
<td>63.2%</td>
</tr>
</tbody>
</table>

**Corollary 1** For any integer $k$, the performance ratio of the iterative-peeling algorithm for $k$-planar subset problem is at least

$$\rho_k \geq 1 - e^{-1} \approx 63.2\%$$

When the number of layers is known, we can use the formula in Lemma 4 to obtain a more precise performance ratio for the iterative-peeling algorithm. In particular, the performance ratio of the iterative-peeling algorithm is at least 75% for the 2-TPR problem and 70.4% for the 3-TPR problem. Table 1 shows the performance ratio of the iterative-peeling algorithm for the $k$-TPR problem for some small values of $k$.

2.1 Approximation Result for $k$-TVM Problem

In solving a routing problem, vias are undesirable for many reasons. Vias lead to increased routing area, decrease yield, reliability and performance of the circuit, and cause problems in fabrication process [10, 8, 24]. For these reasons, via minimization is generally considered as one of the important optimization criteria in solving routing problems.
In this Section, we consider $k$-TVM problem. In TVM problem, the main objective is to maximize the total number of nets that can be routed in planar fashion (without using via). A planar subset is a set of nets which are topologically routable in a single layer. A $k$-planar subset is a set of nets that can be partitioned into at most $k$ planar subsets. Clearly, given a $k$-layer routing region, we can always route a $k$-planar subset without using any vias. Marek-Sadowska [24], Cong-Liu [8] and Rim-Kashiwabara-Nakajima [25] proved the following lemma for two-terminal nets (in this paper, we consider all nets to be two terminal nets):

**Lemma 3** There exists a solution to an arbitrary instance of topological via minimization problem such that each net uses at most one via.

This lemma establishes an upper bound on the number of vias. In order to find the minimum-via solution, we need to maximize the number of nets that can be routed without any vias. Consequently, for $k$-layer routing region, $k$-TVM problem is equivalent to finding maximum $k$-planar subset of nets from the given set. However, it is known that finding maximum $k$-planar subset of nets, even for $k = 2$, in a bounded region is NP-hard [28].

Based on Lemma 3 and the Theorem 1, we conclude,

**Theorem 2** Given a set of nets $N$ in a $k$-layer bounded region, the $k$-TVM problem can be approximated using iterative-peeling algorithm with with at most $(1 - \frac{1}{k})^k \times \Psi^*_k$ more vias than the minimum number of vias, where where $\Psi^*_k$ the weight of a maximum $k$-planar subset of nets in $N$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
CHAPTER III

ROUTING ALGORITHM BASED ON TOPOLOGICAL SOLUTION

In this Chapter, we develop a new 2-layer topology-based geometric routing algorithm. The algorithm works in two phases. In the first phase, the algorithm finds a solution for 2-RTVM problem, i.e., finds a topological routing solution that is guaranteed to be transformable into geometric routing. In the second phase, the algorithm iteratively transforms the topological solution to geometric routing by imposing a grid onto the topological solution.

Before presenting the algorithm, we first define some of the terminologies and notations needed to develop the algorithm. A net is called solid net if it is completely routed in a single layer. If a net is routed in two layers using a via then the net is called a broken net. Two non-crossing solid nets or a solid net and the boundary of the routing region in a layer form a panel. A panel $P$, formed by two non-crossing nets $n_1$ and $n_2$, is defined by two ordered tuples: $P = ((a_1, a_2), (b_1, b_2))$, where $a_1, b_1$ and $a_2, b_2$ are the terminals of nets $n_1$ and $n_2$ respectively. All the terminals between $a_1$ and $a_2$ lie on the boundary (of the region) of the panel. Similarly, all the terminals between $b_1$ and $b_2$ lie on the boundary of the panel. If a panel $P$ is formed by one net $n_1$ and the boundary, then it is defined by $P = ((a_1, -), (b_1, -))$, where $a_1$ and $b_1$ are the two terminals of $n_1$.

It is clear that if $n$ non-crossing solid nets are routed in a layer, then $n + 1$ panels are formed in that layer. We denote the panels in layer 1 as a set $P^1 = \{P_0^1, P_1^1, \ldots\}$ and the panels in layer 2 as a set $P^2 = \{P_0^2, P_1^2, \ldots\}$. If two panels $P_i^1$ and $P_j^2$, for $0 \leq i \leq p$ and $0 \leq j \leq q$ intersect, they form a region, denoted by $R_{ij}$. That is, a region is an intersection of two panels. Note that in this approach, two panels may intersect more than once in the re-routing.
step. However, we are only interested in one particular intersection (region), as explained in the next section. In general, a region $R_{ij}$ is bounded by four nets, two nets of panel $P_i$ ($n_i$ and $n_{i+1}$) and two nets of panel $P_j$ ($n_j$ and $n_{j+1}$). Figure 1 shows an example of panels in layer 1 and different panel intersections.

\begin{figure}[ht]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{Example of Panels and Regions.}
\end{figure}

3.1 Overview of the Algorithm TGR2\_ROUTE

The algorithm consists of two phases. In the first phase, the algorithm finds a solution to 2-RTVM problem. Graph model is used to find the topological solution. The second phase converts the topological solution into actual geometric routing. Before presenting the algorithm, we first describe in details of the two phases of the algorithm.

3.1.1 Phase 1: Solution for 2-RTVM Problem

In this phase, the algorithm TGR2\_ROUTE finds a solution to 2-RTVM problem. The algorithm first finds two planar subsets, $N_1$ and $N_2$, of nets from
the given set of nets $\mathcal{N}$ using iterative-peeling algorithm and then forms panels $\mathcal{P}^1 = \{P_0^1, P_1^1, \ldots, P_p^1\}$ and $\mathcal{P}^2 = \{P_0^2, P_1^2, \ldots, P_p^2\}$ by routing $\mathcal{N}_1$ in layer 1 and $\mathcal{N}_2$ in layer 2, respectively. By Theorem 1, $|\mathcal{N}_1 \cup \mathcal{N}_2|$ is at least 0.75 of the size of an optimal 2-planar subset solution. Moreover, since $\mathcal{N}_1$ is a maximum planar subset, every net in $\mathcal{N}_1$ must intersect with some nets in $\mathcal{N}_2$ (otherwise, $\mathcal{N}_1$ can be augmented). If we project the planar routing solutions in layers 1 and 2 onto a single layer, the intersections of panels form regions which loosely describe a mesh. A unique feature of our algorithm is that we carry out routing on a mesh naturally formed by the 2-planar subset, instead of on the imposed routing grid. The set of regions formed by the intersection of all panels is defined as:

$$R = \{R_{ij} | \mathcal{P}^i_1 \text{ and } \mathcal{P}^j_2 \text{ intersect, for all } 0 \leq i \leq p, \text{ and } 0 \leq j \leq q\}$$

The boundary nets of $R_{ij}$ are $n_i, n_{i+1}, n_j,$ and $n_{j+1},$ where $n_i, n_{i+1} \in \mathcal{N}_1$ and $n_j, n_{j+1} \in \mathcal{N}_2$ (if $i(j) = 0,$ $n_i(n_j)$ is not defined). A net $n_i$ is the boundary of all regions $R_{ij},$ for $0 \leq j \leq q$ such that $n_i$ and $n_j$ intersect. Let $n_i \in \mathcal{N}_1$ intersects with panels $\mathcal{P}^2_0, \mathcal{P}^2_1, \ldots, \mathcal{P}^2_j,$ then $n_i$ can be divided into $j + 1$ segments $n_i0, n_i1, \ldots, n_ij.$ Similarly, if net $n_j \in \mathcal{N}_2$ intersects with panels $\mathcal{P}^1_0, \mathcal{P}^1_1, \ldots, \mathcal{P}^1_i,$ then $n_j$ can be divided into $i + 1$ segments, $n_j0, n_j1, \ldots, n_ji.$ Now, region $R_{ij}$ can be defined by a four tuple as: $R_{ij} = (n_{ij}, n_{i(j+1)}j, n_{ij}, n_{i(j+1)})$. At most three entries in the 4-tuple may be null.

The capacity of a net segment $n_{ij}$, denoted by $c_{ij},$ is the maximum number of nets that can cross this segment. The capacity of a region $R_{ij}$, denoted by $C_{ij},$ is defined by a four tuple as: $C_{ij} = (c_{ij}, c_{i(j-1)}j, c_{ij}, c_{(j+1)i})$. Similarly, we define the load of a net segment $n_{ij}$, denoted by $l_{ij},$ be the number of nets passed this segment at certain time of routing, and the load of the region $R_{ij}$, denoted by $L_{ij}$ is defined as $L_{ij} = (l_{ij}, l_{i(j-1)}j, l_{ij}, l_{(j+1)i})$.

As can be seen that the proper distribution of nets over the entire routing region mostly depends on the initial capacities of each net segments in each region. There can be more than one ways to assign the initial capacities to the regions.
One straightforward assignment could be to assign a capacity of \( n - |N| \) to each net segment in each region. This ensures that each unrouted net can pass through any region. Another assignment of capacities depends on the local prediction of nets. Let \( n_j, n_{j+1}, \ldots, n_{j+m} \) pass through panel \( P_i^1 \), i.e., these nets cut \( n_{i-1} \) and \( n_i \). Let the offset of two nets \( n_{i-1} \) and \( n_i \) on two sides of the boundary are \( x \) and \( y \). Without loss of generality, we assume that \( x > y \). Then we define \textit{capacity difference} (\( cd \)) of two adjacent nets \( n_j \) and \( n_{j+1} \) as: \( cd = \frac{x - y}{m+1} \). Now we compute the capacities of net segments \( n_{ji}, n_{(j+1)i}, \ldots, n_{(j+m)i} \) as \( x, x - cd, x - 2cd, \ldots, x - (m+1)cd \). See Figure 2 for illustration. Similarly, the capacities of all the segments can be computed.

Each time a net \( n_t \) is topologically routed, the capacities of all the regions to that the net is assigned are adjusted. Let a segment of a net \( n_t \) has been assigned to a region \( R_{ij} = (n_{ij}, n_{(i+1)j}, n_{ji}, n_{(j+1)i}) \). Without loss of generality, assume that \( n_{ij}, n_{(i+1)j}, n_{ji}, n_{(j+1)i} \) are top, bottom, left and right boundaries respectively. In case the net \( n_t \) does not use any via in region \( R_{ij} \), then it will only cross either \( n_{ij} \) and \( n_{(i+1)j} \) or \( n_{ji} \) and \( n_{(j+1)i} \) (see Figure 3 (a) and (b)). If the net crosses \( n_{ij} \), and \( n_{(i+1)j} \) then \( c_{ij} \) and \( c_{(i+1)j} \) will be decreased by one and the \( c_{ji} \) and \( c_{(j+1)i} \) will be unchanged. Similarly, if the net crosses \( n_{ji} \) and \( n_{(j+1)i} \), \( c_{ji} \) and \( c_{(j+1)i} \) will be unchanged.
decreased by one and $c_{ij}$ and $c_{(i+1)j}$ will be unchanged. If a via is used in the region $R_{ij}$, as shown in Figure 4, the capacity of each boundary will be decreased by one. This is due to the fact that a via occupies both a horizontal and a vertical track.

Topological routing of a net corresponds to assigning segments of that net to regions such that the segments combined represent a continuous interconnection of its two terminals. For example, let the segments of a net $n_t = (a_t, b_t)$ be assigned to a subset of regions $R' = \{R_{i1}, R_{i2}, \ldots, R_{ir}\}$, where $R_{im} \in R$, for $1 \leq m \leq r$. Then the valid routing of $n_t$ implies that there exists a permutation of regions in

Figure 3. A Net Segment Assigned to a Region Without Via.

Figure 4. A Net Segment Assigned to a Region Using Via.
R', say $R_{ij}, R_{i'j}, \ldots, R_{i'r}$, such that $a_t$ and $b_t$ lie in $R_{ij}$ and $R_{i'}$, respectively, and $R_{ij}$ is adjacent to $R_{ij+1}$, for $1 \leq j < r$. It is clear to see that segments of a particular net to be routed cannot be assigned to a region more than once in its minimum via topological routing. As shown in Figure 5 (a), assignment of segments in a region more than once causes a "loop" in the routing and the loop can be removed to get a shorter routing as shown in Figure 5 (b).

We define a vertex-weighted graph, denoted by region adjacency graph, $G = (V, E)$, from the regions $R_{ij}$, where each vertex $v_{ij} \in V$ represents a region $R_{ij}$ and the edge set $E$ consists of two types of edges, $E_1$ and $E_2$, where

$E_1 = \{(u_{ij}, v_{(i+1)j}) | R_{ij}$ and $R_{(i+1)j}$ share the net segment $n_{(i+1)j}\}$ and

$E_2 = \{(v_{ij}, v_{(i+1)j}) | R_{ij}$ and $R_{i(j+1)}$ share the net segment $n_{i(j+1)}\}$.

We refer to edges in $E_1$ ($E_2$) as type 0 (1) edges. Figure 6 gives an example showing regions and the corresponding region adjacency graph. The labels in Figure 6 (b) indicate the types of edges. The weight of a vertex $v_{ij}$, denoted by $w(v_{ij})$, is the sum of the loads of the boundaries of region $R_{ij}$, i.e.,

$$w(v_{ij}) = l_{ij} + l_{(i+1)j} + l_{ji} + l_{(j+1)i}.$$
The initial weight of each vertex is zero, since no net has been assigned to regions. Clearly, a path in the region adjacency graph corresponds to a topological route. A path is called trivial if it contains a single vertex. Note that a path may contain both 0 and 1 type edges. We call an internal vertex in a path in the region adjacency graph a switch vertex if it incidents to two edges of different types.

Assume that a net $n_t = (a_t, b_t)$ has terminal $a_t$ and $b_t$ in regions $R_{mn}$ and $R_{pq}$ respectively. Clearly, the topological routing of $n_t$, i.e., the wire connecting $a_t$ and $b_t$ represents a non-trivial path $P$ in the corresponding region adjacency graph $G$ from the vertex $v_{mn}$ to $v_{pq}$.

Let the length of a path $P$ from vertex $v_{mn}$ to $v_{pq}$ be $L$. Then the weight of the path $P$ is computed by the following function:

$$W(P) = L + \sum_{v \in P} w(v) + Cx$$

where $x$ is the number of switch vertices in $P$. It is easy to see that a switch vertex in the region adjacency graph corresponds to a region in which a via is needed to route the net. Thus the total number of switch vertices in a routing

Figure 6. Example of Regions and Region Adjacency Graph.
path of a net is equal to the total number of vias used to route the net and $C$ is a constant determining the weight of a via. Since our objective is to minimize the number of vias, we assign a relatively large value to $C$. The vertices along the path $P$ represent the regions in which the net $n_i$ has to be passed in routing. There may be more than one paths for any given pair of vertices and we are to find the one that minimizes the number of vias as well as wire length. It is easy to see that minimizing the number of vias in routing is equivalent to finding a weighted shortest path in the region adjacency graph from a source vertex to a destination vertex, where source and destination vertices corresponds to the regions in which two terminals of that net are located. Therefore, to route a net in $N - (N_1 \cup N_2)$, the algorithm first forms region adjacency graph and routes one net at a time by finding weighted shortest path in the region adjacency graph.

3.1.2 Phase 2: Topology to Geometric Routing

After finding the topological routing, the algorithm transforms the topological routing solution into a geometric routing solution by mapping the topological solution onto a routing grid. The algorithm iteratively assigns each net segment in each region to grid lines. The regions are considered in row major order. Note that in this phase, the load of each boundary net segment of a region is known, i.e., it is known that how many nets pass through each boundary net segment. A net segment may or may not use a via in a region. Assume that the region $R_{ij} = (n_{ij}, n_{i(j+1)}, n_{ji}, n_{(j+1)i})$ is considered to assign grid (Figure 7 (a)). Let the boundary net segment $n_{ij}$ is assigned to $i_1$-th column (we refer vertical grid lines as columns and horizontal grid lines as tracks) and $n_{ji}$ is assigned to $j_1$-th track. Consider the net segments passing boundary segment $n_{ij}$. Let $n_{ji}'$ be the $l_1$-th net segment from the left in the list of all net segments passing $n_{ij}$. There are two cases to consider:
Case 1 \((n_{j'} \text{ uses a via in region } R_{ij})\): In this case \(n_{j'}\) may pass through either one of the boundary net segments \(n_{ij}\) or \(n_{(i+1)j}\). In either case, consider the ordering of \(n_{j'}\) in the list of nets passing \(n_{ij}\) or \(n_{(i+1)j}\). Let \(n_{j'}\) be the \(l_2\)-th net in that list. Then assign \(l_1\)-th column and \(l_2\)-th track to complete the connection of \(n_{j'}\) in region \(R_{ij}\) (see Figure 7 (b) for illustration). The via is placed in the grid point of \(l_1\)-th column and \(l_2\)-th track.

Case 2 \((n_{j'} \text{ does not use via in region } R_{ij})\): In this case, let \(n_{j'}\) be the \(l_2\)-th net in the list of net segments of boundary \(n_{(j+1)i}\) (from the left). Note that all the net segments left to \(n_{j'}\) in \(n_{(j+1)i}\) must use vias in region \(R_{ij}\). Thus each net segment needs a track and a column in \(R_{ij}\). Therefore, we need to use \(l_2\)-th column to connect the net segment to the grid point of \(l_2\)-th column and \(l_2\)-th track. Then use a jog in \(l_2\)-th track to complete the routing of \(n_{j'}\) (see Figure 7 (c) for illustration).

We first consider all net segments passing \(n_{ji}\), then we consider all net segments in boundary \(n_{(j+1)i}\) that have not been considered in \(n_{ji}\). At last we consider all nets that pass through \(n_{ij}\) and \(n_{(i+1)j}\) that have not been considered previously. Note that a via uses a track as well as a column and in phase 1 that is already considered by adjusting the capacities and loads of each region boundary.

### 3.2 Operations

Now we explain some procedures used in the algorithm.

**partition** \((N, N_1, N_2)\) : This procedure partitions a 2-planar subset of nets \(N\) into two planar sets \(N_1\) and \(N_2\).

**route** \((N, L)\) : This procedure routes the planar set of nets \(N\) in given layer \(L\). Since \(N\) is a planar set, nets in \(N\) are routed without using vias.
form_region($P_1^j, P_2^j$) : This procedure takes two panels $P_1^j \in P^1$ and $P_2^j \in P^2$ and outputs the intersection region as $R_{ij}$. $R_{ij}$ is null if the panels do not intersect.

initialize_capacities($R_{ij}$) : This procedure initializes the capacities of each net segments of the region $R_{ij}$. It computes the capacities according to the function explained previously.

form_region_adjacency_graph() : This procedure forms region adjacency graph from the regions obtained by the procedure form_region. It also initializes the weight of each vertex in the graph.

shortest_path($G, v_{ij}, v_{mn}$) : This procedure finds the weighted shortest path between two vertices $v_{ij}$ and $v_{mn}$ in the region adjacency graph $G$.

assign_net_to_region($n_j, \mathcal{P}$) : This function assigns the net $n_j$ to the regions corresponding to the vertices along the path $\mathcal{P}$ in the topology. It also creates a via in a region if necessary.

3.3 Algorithm

Now we formally describe the algorithm. For ease of reference, we denote the iterative-peeling algorithm to find a 2-planar subset as iterative-peel(N).

Algorithm TGR2_ROUTE()

*Input* : $N = \{n_1, n_2, \ldots, n_n\}$.

*Output* : A geometric routing.

begin

/* PHASE 1: find a routable topological solution */

1. $N' = \text{iterative-peel}(N)$;
   $N = N - N'$;

2. /* partition the 2-planar subset obtained by iterative-peel(N) into two planar sets */

*/
partition($N', N_1, N_2$);
Let $N_1 = \{n_1, n_2, \ldots, n_p\}$, $N_2 = \{n_1, n_2, \ldots, n_q\}$.

3. /* route nets in $N_1$ and $N_2$ in layers 1 and 2 without vias */
   route($N_1, L_1$);
   route($N_2, L_2$);

4. /* find all the regions formed by panels in two layers */
   for ($i = 1$ to $p$) do
      for ($j = 1$ to $q$) do
         if (intersect($P_i, P_j$)) then
            $R_{ij} = \text{form}_\text{region}($$P_i, P_j$);
         else
            $R_{ij} = \text{null}$;
   
5. /* initialize the loads and the capacities of each region net segment */
   for ($i = 1$ to $p$) do
      for ($j = 1$ to $q$) do begin
         ($c_{ij}, c_{(i+1)j}, c_{ij}, c_{(j+1)i}$) = initialize_capacities($R_{ij}$);
         ($l_{ij}, l_{(i+1)j}, l_{ij}, l_{(j+1)i}$) = (0,0,0,0);
      end;

6. $G = \text{form}_\text{region}_\text{adjacency}_\text{graph}()$;

7. /* topologically route the remaining nets using region adjacency graph */
   for each net $n_t \in N$ do begin
      Let $n_t = (a_t, b_t)$;
      $R_{ij} = \text{get}_\text{region}(a_t)$;
      $R_{mn} = \text{get}_\text{region}(b_t)$;
      $\mathcal{P} = \text{shortest}_\text{path}(G, v_{ij}, v_{mn})$;
      assign_region_to_net($n_t, \mathcal{P}$);
   end;

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
/* PHASE 2: This phase transforms the topological routing solution obtained in phase 1 to geometric routing by assigning grids to each net segment */

8. for \((i = 1\) to \(p)\) do
   
   for \((j = 1\) to \(q)\) do
      
      for each net segment \(n_{ij'}\) passing \(n_{ji}\) do
         
         let \(n_{ij'}\) uses \(l_{1}\)-th column in \(n_{ji}\);
         
         if \(n_{ij'}\) uses via in \(R_{ij}\) then
            
            find boundary net segment \(n_{rj'}\) \(\in\) \(\{n_{ij}, n_{(i+1)j}\}\) such that \(n_{ij'}\)
            
            passes through \(n_{rj'}\);
            
            let \(n_{ij'}\) be the \(l_{2}\)-th net segment in \(n_{rj'}\);
            
            assign \(l_{2}\)-th track to \(n_{ij'}\) in layer 2 and \(l_{1}\)-th column to \(n_{ij'}\) in layer 1;
            
            assign via at grid point \((l_{1}, l_{2})\);
         
         else
            
            let \(n_{ij'}\) be the \(l_{2}\)-th net segment in \(n_{(i+1)j}\);
            
            assign \(l_{2}\)-th column and \(l_{1}\)-th column to \(n_{ij'}\) and use a jog in \(l_{2}\)-th track bottom from \(n_{(i+1)j}\);
            
            for each net segment \(n_{ij'}\) passing \(n_{(i+1)j}\) and not assigned to the grid do
               
               let \(n_{ij'}\) use \(l_{1}\)-th column in \(n_{(i+1)j}\);
               
               find boundary net segment \(n_{rj'}\) \(\in\) \(\{n_{ij}, n_{(i+1)j}\}\) such that \(n_{ij'}\)
               
               passes through \(n_{rj'}\);
               
               let \(n_{ij'}\) be the \(l_{2}\)-th net segment in \(n_{rj'}\);
               
               assign \(l_{2}\)-th track to \(n_{ij'}\) in layer 2 and \(l_{1}\)-th column to \(n_{ij'}\) in layer 1;
               
               assign via at grid point \((l_{1}, l_{2})\);
            
         for each net segment \(n_{ij'}\) passing \(n_{ij}\) and not assigned grid do
            
            let \(n_{ij'}\) use \(l_{1}\)-th column in \(n_{ij}\) and \(l_{2}\)-th column in \(n_{(i+1)j}\);
            
            assign \(l_{2}\)-th and \(l_{1}\)-th track to \(n_{ij'}\) in layer 2 and use a jog in \(l_{2}\)-th column top from \(n_{ji}\);
      
      end. 
   
end.
In step 1, the time complexity of iterative-peeling algorithm, iterativipeel(N), is $O(n^2)$. Step 3 of the algorithm takes $O(n)$ time. Steps 2, 4, 5, and 6 takes $O(n^2)$ time. The complexity of finding the shortest path is $O(n^2)$, where $n$ is the total number of vertices in the graph, and we need to find the shortest path for each un-routed net, which is of $O(n)$. Thus, the total time complexity of step 7 of the algorithm is $O(n^3)$. It is easy to see that the time complexity of steps 8 and 9 in phase 2 of the algorithm is $O(n^2)$. Hence, the total time complexity of the algorithm is $O(n^3)$. Therefore, we conclude,

**Theorem 3** Given a set of nets $N = \{n_1, n_2, \ldots, n_n\}$ in a bounded region, the algorithm TGR2.ROUTE finds a geometric routing in time $O(n^3)$.

### 3.4 Extensions

In this section we present two extensions of our proposed topology based routing algorithm. First we show that the algorithm can be easily modified to handle routing problems with a large number of routing layers. Then we show that our algorithm is effective to handle the weighted topology based routing, where the net is assigned some positive weight according to the priority of that net.

#### 3.4.1 Multilayer Topology Based Routing

Advances in VLSI fabrication technology have made it possible to use more than two routing layers for interconnections. Many VLSI chips have been designed using three or four metal layers for routing. Multichip modules (MCMs) and high density printed circuit boards (PCBs) may use even more layers for interconnections. For example, the MCM developed for the IBM 3081 mainframe has 33
layers of molybdenum conductors (including 1 bonding layer, 5 distribution layers, 16 interconnection layers, 8 voltage reference layers, and 3 power distribution layers [3, 4]).

These recent developments in VLSI fabrication and packaging technology calls for an effective routing algorithm for multilayer routing problems. Our proposed algorithm can easily be extended for routing problems in more than two layers. If \( k \) layers are available then iterative-peeling algorithm can be used to find a \( k \)-planar subset of nets and can be routed them in \( k \) layers in planar fashion. Adjacent layers can be grouped into pairs and the remaining nets are routed in pairs of layers using via minimization as an optimization criteria.

3.4.2 Avoiding Vias in Critical Nets

In the design of VLSI circuits, often one net, called critical net (such as power and ground nets, clock nets), is preferred to be routed entirely in one layer over another net. The proposed algorithm TGR2 ROUTE can easily be modified to handle these critical nets. Associated to each net \( n_i \), we may assign a positive weight \( w(n_i) \) which is the measure of the priority (or degree of criticality) of \( n_i \). The weight of a subset of nets \( n' \) is defined to be \( W(n') = \sum_{n_j \in n'} w(n_j) \). The iterative-peeling algorithm can be used to find maximum weighted planar subsets in this case.
Figure 7. Assigning Grids to Net Segments in a Region.
CHAPTER IV

EXPERIMENTAL RESULTS

We have experimentally evaluated both iterative-peeling algorithm and TGR2 ROUTE. In this Chapter, we report some of the experimental results on different examples.

4.1 Iterative-Peeling Algorithm

We implemented the iterative-peeling algorithm in C language on Sun SPARC workstations and tested it on a number of switchbox and channel routing benchmark examples, including Burstein's difficult switchbox routing example [6] and Deutsch's difficult channel routing example [14]. Table 2 reports the results of the iterative-peeling algorithm on these examples. The Burstein example is labeled as 'burs' and the Deutsch example is labeled as 'deut'. The remaining test cases are channel routing examples from [32]. For all examples, we simply assign the weight of each net to be one, i.e., we maximize the cardinality of the $k$-planar subsets to be computed. The next five columns of the table show the percentages of the nets completed using planar routing using iterative-peeling algorithm for one to five routing layers. (Note that these values are not the performance ratio of the iterative-peeling algorithm; Recall that the performance ratio of our algorithm to an optimal algorithm is proven to be at least 63.2%.) The last column shows the number of layers needed for the iterative-peeling algorithm to produce planar routings for all the nets. The computation time for each example is less than 25 seconds.
Table 2

Percentage of Nets Completed by Planar Routing Using Iterative-peeling Algorithm for Different Number of Routing Layers

<table>
<thead>
<tr>
<th>Ex</th>
<th>1L %</th>
<th>2L %</th>
<th>3L %</th>
<th>4L %</th>
<th>5L %</th>
<th>Total layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>burs</td>
<td>41</td>
<td>58</td>
<td>70</td>
<td>79</td>
<td>83</td>
<td>9</td>
</tr>
<tr>
<td>ex1</td>
<td>38</td>
<td>52</td>
<td>66</td>
<td>76</td>
<td>80</td>
<td>9</td>
</tr>
<tr>
<td>ex3a</td>
<td>45</td>
<td>59</td>
<td>68</td>
<td>75</td>
<td>79</td>
<td>11</td>
</tr>
<tr>
<td>ex3b</td>
<td>31</td>
<td>53</td>
<td>68</td>
<td>78</td>
<td>82</td>
<td>10</td>
</tr>
<tr>
<td>ex3c</td>
<td>37</td>
<td>55</td>
<td>62</td>
<td>70</td>
<td>75</td>
<td>13</td>
</tr>
<tr>
<td>ex4b</td>
<td>40</td>
<td>57</td>
<td>68</td>
<td>75</td>
<td>81</td>
<td>13</td>
</tr>
<tr>
<td>ex5</td>
<td>31</td>
<td>48</td>
<td>59</td>
<td>70</td>
<td>78</td>
<td>9</td>
</tr>
<tr>
<td>ex5b</td>
<td>31</td>
<td>48</td>
<td>60</td>
<td>71</td>
<td>79</td>
<td>11</td>
</tr>
<tr>
<td>deut</td>
<td>23</td>
<td>38</td>
<td>50</td>
<td>58</td>
<td>63</td>
<td>18</td>
</tr>
</tbody>
</table>

4.2 TRG2ROUTE Algorithm

The TRG2ROUTE algorithm has been implemented in C on SUN SPARC 1+ workstation and has been tested on many switchbox and channel routing examples. We compare our router with BEAVER [7], and the experimental results show that a significant reduction of vias is achieved by the router.

We have also run the algorithm for well-known Burstein difficult switchbox routing problem by breaking the multi-terminal nets into two terminal nets. Initially, we consider all possible interconnections of terminals of each net. After routing all the wires, the algorithm eliminates the connections that have more vias than others one at a time as long as the connectivity is preserved. Our algorithm
obtained routing with 23 vias. We would like to make a note that this is not the least number of vias reported in literature for burstein difficult switchbox routing problem, in [29], 20-via routing has been reported for this problem. Table 3 reports results of comparison of our router with BEAVER on several switchbox routing problems. The switchbox routing problems labeled as ‘ex1 - ex10’ are randomly generated examples. The Burstein’s difficult switchbox routing problem is referred to as ‘burs’.

Table 3
Comparison Between BEAVER and TRG2_ROUTE Algorithm on Some Switchbox Routing Examples

<table>
<thead>
<tr>
<th>Ex</th>
<th>Total nets</th>
<th>Total vias</th>
<th>BEAVER</th>
<th>our algorithm</th>
<th>number of nets with vias</th>
<th>% improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Total</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>vias</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BEAVER</td>
<td>our algorithm</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>&gt; 2</td>
</tr>
<tr>
<td>ex1</td>
<td>15</td>
<td>6</td>
<td>3</td>
<td>12</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>ex2</td>
<td>18</td>
<td>11</td>
<td>6</td>
<td>13</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>ex3</td>
<td>22</td>
<td>12</td>
<td>8</td>
<td>16</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>ex4</td>
<td>26</td>
<td>14</td>
<td>8</td>
<td>19</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>ex5</td>
<td>27</td>
<td>16</td>
<td>11</td>
<td>19</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>ex6</td>
<td>29</td>
<td>16</td>
<td>8</td>
<td>22</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>ex7</td>
<td>32</td>
<td>17</td>
<td>13</td>
<td>23</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>ex8</td>
<td>35</td>
<td>20</td>
<td>13</td>
<td>25</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>ex9</td>
<td>33</td>
<td>18</td>
<td>13</td>
<td>21</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>ex10</td>
<td>42</td>
<td>27</td>
<td>16</td>
<td>29</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>burs</td>
<td>24</td>
<td>35</td>
<td>23</td>
<td>12</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>
From the results shown in Table 3, we have few interesting observations:

1. Majority of the nets are planar routed. As a result significant reduction on vias is achieved. This is due to the fact that the algorithm first finds maximum planar subset of nets and routes them without vias. Based on this result, we predict that if the algorithm is extended for more layers, further reduction on vias can be achieved.

2. Very few number of nets needed more than two vias to complete the routing. However, this is not the case for multiterminal nets. For example, in burstein's example two nets used three vias and one net used four vias. Thus for multiterminal net routing problems, our algorithm is not as efficient as two terminal nets.

3. The algorithm is inherently for unreserved layer model and as a result a significant reduction on tracks are achieved in channel routing examples.
CHAPTER V

CONCLUSION

In this thesis, we have considered topological planar routing and geometric routing based on topological routing solution. We have presented a provably good multilayer topological planar routing algorithm based on the idea of iterative peeling. Our algorithm is easy to implement and it works for multi-terminal nets and arbitrary number of routing layers with performance ratio at least $1 - \frac{1}{e} \approx 63.2\%$. Experimental results show that our algorithm can generate planar topological routing for most of the nets using small number of routing layers. Such an algorithm is important on the layout design of multilayer IC technology. It can also be used for generating planar routing sketches for rubber-band based routing algorithms [13, 19] to construct detailed planar routing solutions.

We have also presented a new graph-theoretic 2-layer routing algorithm based on topological topological routing solution. We also have shown that the algorithm can easily be extended for routing problems with more than two-layer routing region and to avoid vias on critical nets. Experimental results show that a significant reduction in vias are achieved by our algorithm. Since the general $k$-TVM problem is NP-complete, we have presented an upper bound on the number of extra vias required to solve the $k$-TVM problem.
REFERENCES


