國 立 交 通 大 學

電 信 工 程 學 系

碩　士　論　文

以性能和擁擠為導向的多階層繞線方法

Performance- and Congestion- Driven Multilevel Router

研 究 生：林義琅

指導教授：李育民　教授

中 華 民 國 九 十 四 年 十 月

以性能和擁擠為導向的多階層繞線方法
Performance- and Congestion- Driven Multilevel Router

研　究　生：林義琅　　　Student：Yih-Lang Lin

指導教授：李育民　　　Advisor：Yu-Min Lee

國　立　交　通　大　學
電　信　工　程　系
碩　士　論　文

A Thesis

Submitted to Department of Communication Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Communication Engineering

October 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年十月

以性能和擁擠為導向的多階層繞線方法

學生：林義琅　　　　　　　　　　　　　　　　　　　指導教授：李育民

國立交通大學電信工程學系碩士班

摘　　　要

本論文研製之多階層繞線器可降低擁擠區域、得到較好的性能以及繞線完成度。藉由最小距離生成樹演算法以及最短改善方法可以對所有的網路建構出一個滿足時序限制以及較少線段長度的繞線拓撲結構。之後將網路分成 critical 以及 non-critical 部分並分別對其做繞線，而且結合了機率及實際的繞線擁擠程度以獲得更精確的擁擠程度估算值。在實驗結果中，可以看出我們提出的多階層繞線器比先前的繞線器得到更令人注目的結果。對於所有的測試電路，我們可以百分之百繞線完成，然而以前提出的多階層繞線器對任一個測試電路都沒有辦法完全的成功繞出結果。

Performance- and Congestion-Driven Multilevel Router

student：Yih-Lang Lin Advisor：Dr. Yu-Min Lee

Department of Communication Engineering
National Chiao Tung University

ABSTRACT

In this thesis, we present a novel framework of multilevel routing to decrease the congestion and achieve better performance as well as routability. By performing the minimum distance spanning tree (MDST) algorithm and the *shortest modification heuristic* to construct the performance-driven topology of all nets, we can obtain a better routing topology which satisfies the timing constraint and has less total wire length. After constructing the routing topology, we classify the nets into critical and non-critical nets, route them at different stages, and integrate the probabilistic congestion model with the current routing congestion to improve the accuracy of congestion estimation. The experimental results show that our proposed method achieves significantly better solution than the existing methods. Our approach can achieve 100% routing completion rate for all benchmarks and none of them can be completely routed by prior multilevel routers.

誌　　　謝

本篇論文得以順利完成，首先要感謝指導教授李育民博士，這兩年來在研究以及課業上的指導，使學生獲益良多，並且在我遇到瓶頸之時，給予我適時的幫助與建議，讓學生的研究得以更加完善。

另外要感謝一起在實驗室打拼奮鬥的飛鴻、逸宏、震軒、小宇，還要感謝培育、哲宇、獻哲、阿康等各位學弟在這些日子裡陪伴。最後要感謝我的父母以及家人對我的支持與鼓勵，使我在這段期間能夠平順的度過。最後由衷的感謝每位幫助我以及關懷我的人，願你們能一起分享這份喜悅。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Routing Stage

The physical routing problem is quite complex in Very Large Scaled Integration (VLSI) designs. Generally, it is solved by two steps, global routing followed by the detail routing. The global router first partitions the area of the IC into the $n \times n$ tiles. Then, the global router finds the tile-to-tile path for each two-pin net. Then, the detail router assigns the actual tracks for each two-pin net according to its tile-to-tile path. The traditional methods applied the flat framework [3, 4, 6] to solve the routing problems, but they were unable to handle the large designs. Therefore, the two- or three-level hierarchical frameworks were developed to deal with the large designs by [12, 14, 15]. However, with the technology advanced, they were also not good enough to deal with this problem either. Thus the multilevel frameworks [16, 1, 2] were proposed to efficiently cope with the present and future VLSI designs.

1.2 Multilevel Routing

Many researchers have taken the multilevel structures to overcome the different problems in VLSI designs. The multilevel structure has been verified to be suitable for many different VLSI design tasks, such as multilevel partitions *hMETIS* [7], *ML* [8], and *HPM* [9], and multilevel placements *mPL* [10], and *MR*-tree* [11].

The first multilevel routability-driven router for the large-scale design was proposed by Cong *et al.* in [16]. They calculated the estimation of touring resources at the coarsening stage, and used the multicommodity flow algorithm to generate initial routing at the coarsest stage, and refined the routing solution at the uncoarsening stage. They developed a multilevel method to construct the tile-to-tile paths of all nets, and utilized a non-multilevel gridless detailed router [17] to find the exact connection path of each net. Lin and Chang [1] proposed a multilevel router to consider the routability and performance. They integrated grid-based global routing, detailed routing, and resource estimation together at coarsening stage, and refined the unroutable nets at uncoarsening stage. Ho *et al.* [2] presented a multilevel framework for the full-chip grid-based routing to consider the crosstalk and performance optimization. They used the congestion-driven global routing at coarsening stage, utilized the layer/track assignment at the bottom of coarsening stage, and performed detailed routing to refine the routing solution at uncoarsening stage.

1.3 Congestion Estimation

The congestion of routing is another important issue of modern VLSI physical designs. The congestion not only causes the problem of routability but also degrades the performance of system. [18] proposed a multilevel full-chip routing method based on balancing the routing congestion to optimize the multiple-fault probability, chemical mechanic polishing (CMP), and optical proximity correction (OPC) induced manufacturability and crosstalk effects.

However, the existing multilevel routing methods [1, 2, 18] only consider the current routing congestion. Therefore, the routed nets may pass through the area being heavily congested such that the other nets are unroutable. Hence, in order to increase the routability, and accurately balance the routing congestion, we develop a performance-driven multilevel full-chip router with an accurate congestion estimating engine which integrates the probabilistic routing resource demand (static demand) of not yet routed nets with the

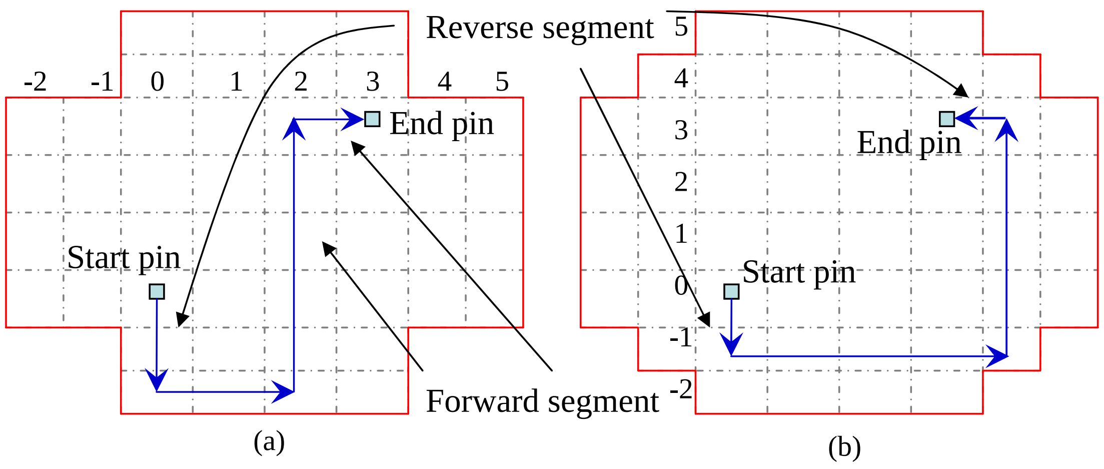


Fig. 1.1: (a) The detour routing in [20]. The allowable reverse segment is one. (b) The proposed detour routing model. The allowable reverse segment is two.

actual demand (dynamic demand) of the routed nets [19].

[20] proposed a new static congestion estimation method which allowed each two-pin net to have bounded-length detours with at most one reverse segment to bypass the congested area, as illustrated in Fig. 1.1. They assumed that the pins are located at the lower left and upper right tiles. The lower left pin was the *start pin* and the upper right pin was the *end pin*. A *forward segment* was a route segment which went continuously up or right. A *reverse segment* was a route segment which went continuously down or left. They assigned the coordinate of the tile containing the start pin to $(0, 0)$ and the coordinate of the tile containing the end pin to (m, n), as illustrated in Fig. 1.1. The coordinate of the end pin in Fig. 1.1 is $(3, 3)$. The experimental results in [20] showed that their congestion prediction is close to a real global router. In this thesis, we will modify their method to get a more accurate congestion estimation by allowing the maximum number of reverse segments to be two and the length of detour to be l.

1.4 Our Contribution

Our multilevel router is different from the previous works [1, 2] with the following significant features.

- **Performance-driven routing topology:** Before routing, we construct a minimum spanning tree (MST) with better performance for each net by our minimum distance spanning tree (MDST) algorithm as shown in Fig. 3.2. Then, the initial topology is modified by an efficient method called the "*shortest modification heuristic*" to meet the timing constraint. The conception of this heuristic is to change the topology to decrease the delay of the pin violating timing constraint with least wire length increased. The total wire length increased by using our *shortest modification heuristic* is less than the recalling modification method developed by [1]. Therefore, our multilevel router needs less routing resource, and extremely improves the routability, as well as satisfies the timing constraint.

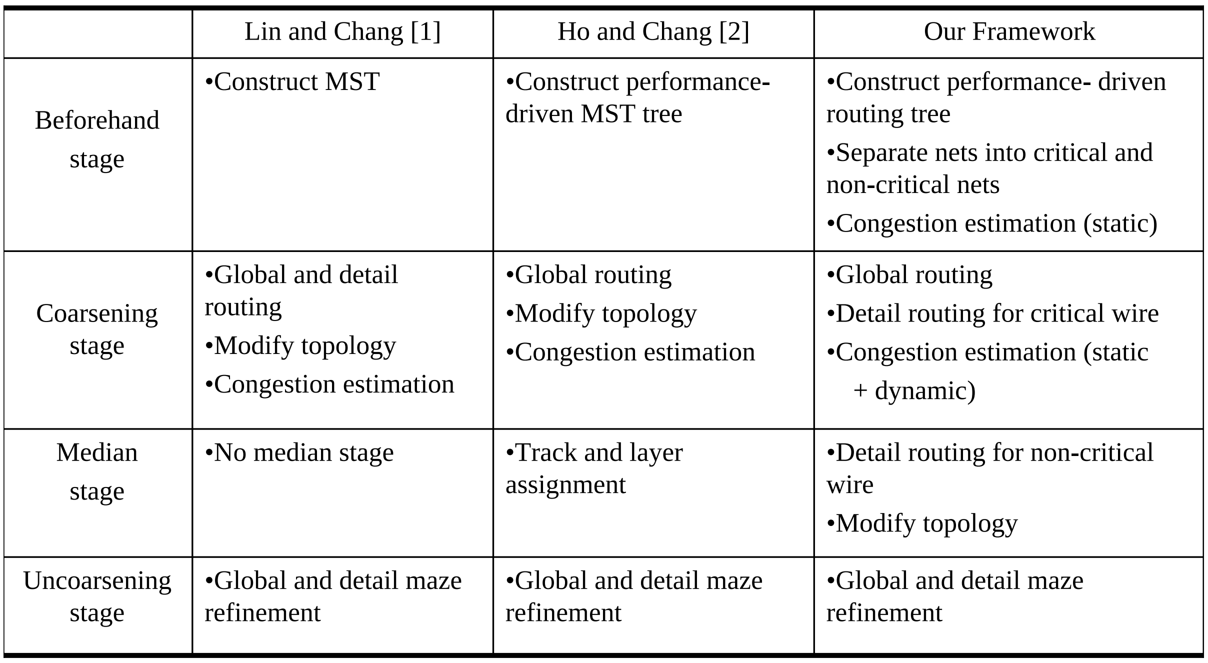
- **Separation of critical and non-critical nets:** Different from the existing methods [1, 2], we classify the nets into critical and non-critical nets after constructing the performance-driven routing topology. A critical net is a net which its max delay is near its timing constraint and may violate the timing constraint after detour. Then, we first route each critical net at the coarsening stage to decrease its delay, and prevent the routing topology of critical nets from perturbing too much; moreover, based on the actually routing shape of critical and non-critical net, we construct an accurately probabilistic congestion estimation model.

- **Probabilistic congestion model:** We propose a more precise detoured congestion estimation model for the non-critical nets, and integrate this congestion model with the L-shaped congestion estimation model for the critical nets to construct a congestion estimation table before routing. Hence, the initial routing connection can avoid the area which will be congested to improve the routability and balance congestion.

- **Congestion-driven L-shaped global routing:** Because the most nets are routed by the L-shaped global routing, we present a modified method to determine what kind of L-shaped routing path (upper-L or lower-L) should be chosen to reduce the total wire length like the work [29] and balance congestion. The detail of this method is presented in Chapter 3.4.

Fig. 3.1 and Table 3.1 show the flow and algorithm of the proposed multilevel router, Fig. 1.2 compares each routing stage between our multilevel router and the existing methods [1, 2]. At the beforehand stage, [1, 2] only build the minimum spanning tree (MST). However, we construct the performance-driven topology to satisfy the timing constraint of each net in order to stabilize the topology and estimate the exactly congestion before routing.

At the coarsening stage, [1, 2] only modify the topology to meet the timing constraint by using their *recalling modification heuristic*. They perform the global router according to the current congestion. Therefore, the primary routing net may pass through the congested area to cause the later nets unroutable. However, our congestion table integrates the probabilistic and actual congestion for routed and not yet routed nets to get the accurate congestion estimation. By this precise congestion estimation mechanism, we can improve the routability and balance the routing congestion. They deal the net correspondingly at the coarsening stage to increase the delay of critical net to violate the timing constraint, and the connection of the net may change again. In order to reduce the delay of the critical net to make the topology unchanging, we separate the nets into critical and non-critical nets, and furthermore the critical nets have priority to be routed by detail router. In the median stage, [1] has no initial routing, and [2] performs the layer/track assignment. However, we perform the detail router on non-critical nets because non-critical nets allow more increasable delay. After that the shortest modification heuristic is used to modify the topology for the net violating its timing constraint again. In the uncoarsening stage, [1, 2] and our router refine in the same way the routing solution.



	Lin and Chang [1]	Ho and Chang [2]	Our Framework
Beforehand stage	•Construct MST	•Construct performance-driven MST tree	•Construct performance- driven routing tree •Separate nets into critical and non-critical nets •Congestion estimation (static)
Coarsening stage	•Global and detail routing •Modify topology •Congestion estimation	•Global routing •Modify topology •Congestion estimation	•Global routing •Detail routing for critical wire •Congestion estimation (static + dynamic)
Median stage	•No median stage	•Track and layer assignment	•Detail routing for non-critical wire •Modify topology
Uncoarsening stage	•Global and detail maze refinement	•Global and detail maze refinement	•Global and detail maze refinement

Fig. 1.2: Comparison of each routing stage between our multilevel router and the existing method [1, 2].

1.5 Organization of This Thesis

The remainder of this thesis is organized as follow. First, the routing model and the multilevel routing framework are presented in Chapter 2. Then, the detailed flow of our multilevel routing system and the main features of our method are particularly described in Chapter 3. Finally, the experimental results and conclusion are shown in Chapter 4 and 5, respectively. The Appendix A shows our probabilistic congestion model with detour length being l, and the Appendix B displays the formulations in [20].

Chapter 2

Preliminary

In this chapter, the implement method for routing problem is presented. First, the routing model for solving the complicated routing problem is introduced in the Chapter 2.1. Due to the technology growths, the traditional approaches can't efficiently handle such involved problem so the multilevel model is presented in the Chapter 2.2. In the Chapter 2.3, we introduce the Elmore delay model for analyseing the delay of each pin. Finally, the motivation and the problem formulation are stated in the Chapter 2.4,2.5, respectively.

2.1 Routing Model

In order to efficiently solve the complex problem of routing in the current and future VLSI designs, the routing regions are represented to the graph based on the actual layout. Our multilevel router uses the graph search algorithm to route, and the congested information integrating with probabilistic and factual demand guide the algorithm. In order to balance the routing congestion, the routing cost of heavily congested areas should be significantly more than the uncongested areas. Furthermore, the routing cost should not be increased in lightly congested areas to prevent from introducing unnecessary detours.

Our probabilistic congestion estimation model and global router work on a model similar to the model in Fig. 2.1. The chip is divided into the $n \times n$ rectangular tiles. In the routing model, a tile represent a node, and an edge expresses the boundary between two adjacent tiles. Each edge has a capacity according to the number of available routing

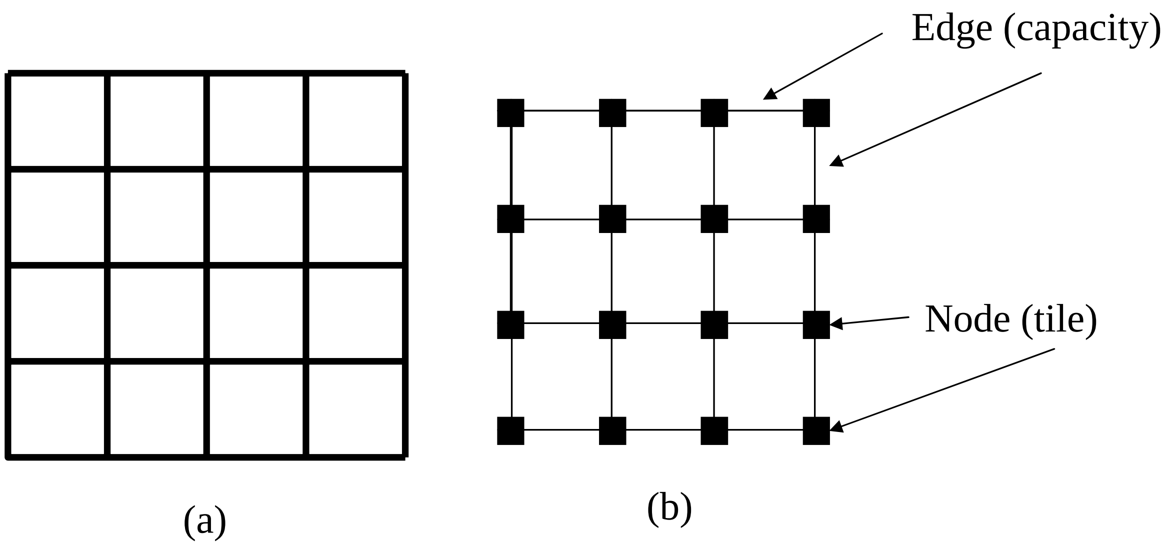


Fig. 2.1: The divided area in (a), and the routing graph .

tracks. According to this routing graph, we can compute the probabilistic congestion for each edge, and our global router finds the tile-to-tile path which has the minimum total cost for each net to lead the detail router in general.

In our routing model, routing in the same layer is only allowed to route the vertical or horizontal direction.

2.2 Multilevel Routing Model

As represented in Fig. 3.1, the routing graph of the level 0 of the multilevel coarsening stage is the G_0. We route the local nets (or local two-pin connections) that the nets (connections) which are fully placed in a tile at each level. After dealing with all local nets, we merge the 2×2 tiles of G_0 into a large grid. The coarsening continues until the number of grids is less than a threshold at the level (k level). At the end of coarsening stage, the detailed routing is performed on not yet routed nets. The unroutable connections are refined by the global and detailed maze routing methods, and the techniques of rip-up and reroute starting at the k level. Then we continue to process the next level $k - 1$ by extending each grid to four finer grids. It continues until reach level 0.

2.3 Elmore Delay Model

We adopt the Elmore delay model [25] to perform the timing analysis for each net to calculate the delay of each sink in a net. The Elmore delay model is described as follow. The Fig. 2.2 shows that a wire segment is modeled as π-model, and a via is also modeled as a π-model, which the value of resistance (*R*) and capacitance (*C*) are twice as more as the wire segment. All nodes in a net are dealt with drivers modeled by a resistance connected to the downstream and a capacitance connected to the upstream. The Elmore delay is calculated by summing the time constant ($resistance \times downstream\ capacitance$).

Let T be an RC tree, with vertices $V = \{v_1, v_2, \cdots, v_n\}$, c_k is the capacitance of a vertex v_k, r_k is the resistance of the edge between v_k and its immediate predecessor. The subtree capacitance at node k is given as

$$C_k = c_k + \sum_{i \in S_k} C_i \tag{2.1}$$

where S_k is the set of all the immediate successor of v_k. Let $\delta_{m,n}$ be the path between v_m and v_n, excluding v_m and including v_n. Then the delay between two nodes m and n is

$$t_{mn} = \sum_{n \in \delta_{m,n}} r_n C_n \tag{2.2}$$

Fig. 2.3 shows a tree, we can calculate the delay from Source to Sink$_b$.

$$\begin{aligned} t_{sink\_b} &= R_s(c_1 + c_2 + c_3 + c_4 + c_{L\_a} + c_{L\_b} + c_{L\_c}) \\ &\quad + R_2(\frac{c_2}{2} + c_3 + c_4 + c_{L\_b} + c_{L\_c}) + R_4(\frac{c_2}{2} + c_{L\_b}) \end{aligned} \tag{2.3}$$

2.4 Motivation

In the present and future Very Large Scale Integration (VLSI) design, the traditional methods aren't good enough to deal with the complex problem of the physical routing. Hence, the multilevel structure is presented to efficiently solve such problem. In the routing problem, the routability is quite important. However, the completely routing circuit considering just the routability may invite the critical path that has heavily loading, and therefore

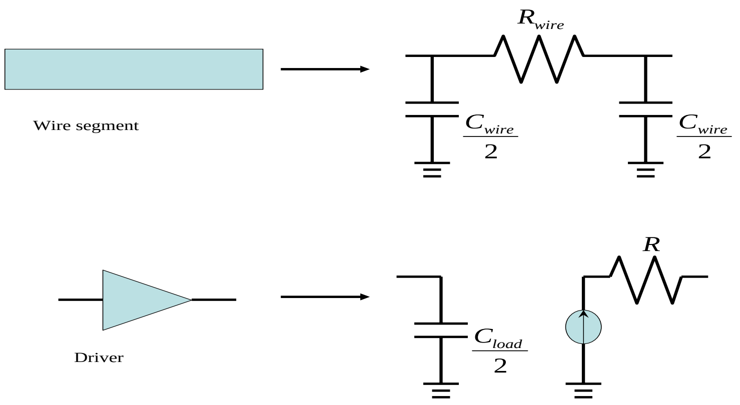


Fig. 2.2: The model of the wire segment and the driver.

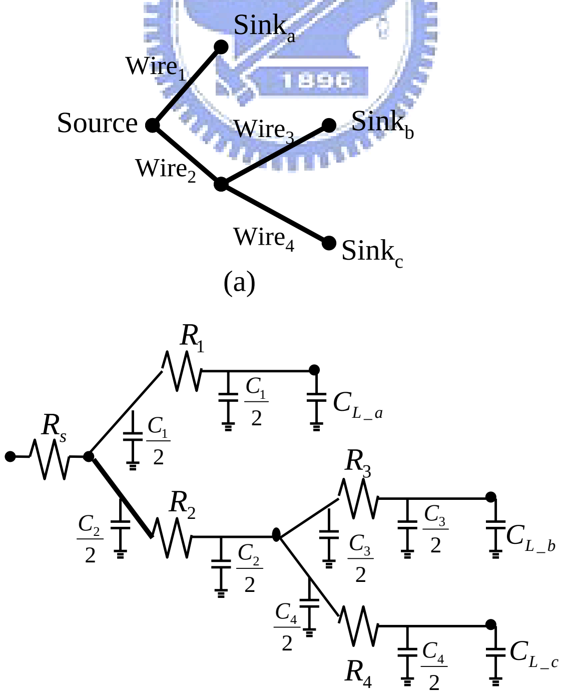


Fig. 2.3: A example calculating the Elmore delay. (a)the tree. (b)the RC tree.

the sinks of the critical path violate the timing constraint to cause the function of the circuit fail. Hence, improving the routing completion and satisfying the timing constraint is significant issue.

The congestion of routing is another serious issue in the current VLSI design. The routing congestion not only causes the unroutable, but also induces the Signal Integrate (SI) and hot spot etc. Without considering the routing congestion before routing, the early routed net has no any information of congested area to pass through the area which may be heavily congested area such that the late net which must pass through the area are untroutable. Hence, a method of precisely predicted congestion is outstanding to improve greatly the routability.

2.5 Problem Formulation

In this thesis, our goal is to route as many nets as possible under the given timing constraint and to balance the routing congestion.

- **Input:**

  The benchmark circuit and the timing constraint.

- **Output:**

  The actual routing result is shown to the screen by the LEDA (Library of Efficient Data Types and Algorithms) [27].

- **Goal:**

  Under the given timing constraint, we construct the performance-driven topology and then separate all nets into critical and non-critical nets to estimate the routing congestion and to route respectively to route completely for the given circuit.

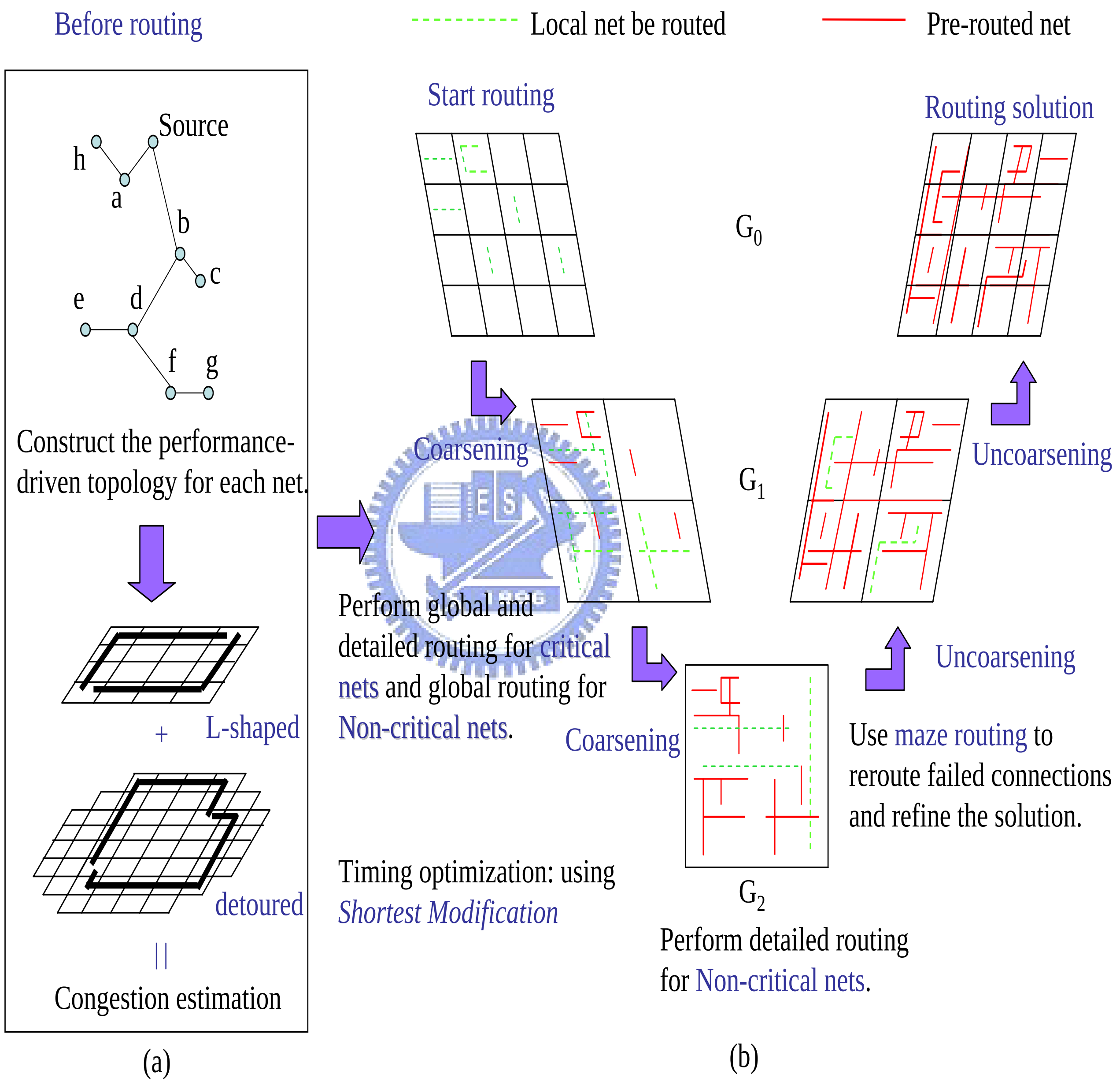
Chapter 3

Multilevel Routing Framework

In this chapter, we specifically describe the flow and the main features of our multilevel routing system. First, we explicitly and gradually characterize the flow of our multilevel router in the Chapter 3.1. Secondly, the principle of the performance-driven topology building by the minimum distance spanning tree (MDST) and the shortest modification heuristic is shown in the Chapter 3.2. Then, the Chapter 3.3 presents the detoured routing congesiton estimation used for the non-critical nets. Finally, the modified method for L-shaped global routing is displayed in the Chapter 3.4.

3.1 Multilevel Routing Flow

The flow and algorithm of the proposed multilevel router are shown in Fig. 3.1 and Table 3.1, respectively. The preliminary work of this router is shown in Fig. 3.1.(a). First, we construct the performance-driven routing topology for each net to meet its timing constraint by our minimum distance spanning tree (MDST) and the *shortest modification heuristic*, and decompose nets into two-pin connections. Then, the timing analysis based on the Elomre delay model is used to estimate the delay of each net by using the Manhattan distance. After that, the nets are separated into critical and non-critical nets. A critical net is a net satisfying the condition that the difference between its maximum delay and its timing constraint is less than the delay of a wire with two-tiled length. A net which is not a critical net is called a non-critical net. Afterward, all of nets are dealt individually



Before routing
Local net be routed
Pre-routed net
Source
h
a
b
c
e
d
f
g
Construct the performance-driven topology for each net.
+ L-shaped
detoured
||
Congestion estimation
(a)
Start routing
Routing solution
G0
G1
G2
Coarsening
Uncoarsening
Perform global and detailed routing for critical nets and global routing for Non-critical nets.
Coarsening
Uncoarsening
Use maze routing to reroute failed connections and refine the solution.
Timing optimization: using Shortest Modification
Perform detailed routing for Non-critical nets.
(b)

Fig. 3.1: The multilevel routing flow.

```
Algorithm : Multilevel Routing($G$, $N$, $C$)
Input : $G$ - partitioned layout;
        $N$ - multi-terminal nets;
        $C$ -timing constraints.
Output : Routing solution for $N$.
Begin
1. Construct performance-driven topology
     a. Build Minimum Distance Spanning Tree (MDST) for $N$.
     b. Using Shortest Modification Heuristic to modify the topology for violated nets.
2. Analyze timing for $N$, and classify $N$ into critical and non-critical nets.
3. Perform congestion estimation for critical (L-shaped) and non-critical (detour) net.
     //Coarsening stage
4. While (coarsening stage)
5.       Choose local connection $n$;
6.       if (the connection $n$ is critical)
7.             L-shaped global routing and detail routing
8.             if (the connection $n$ cannot be routed)
9.                   Z-shaped global routing and detail routing
10.      else
11.            Global routing (L-shaped $\rightarrow$ Z-shaped $\rightarrow$ Detour);
     //Median
12. Detailed maze routing for non-critical nets.
13. Analyze timing for $N$, perform Shortest Modification for violated nets, and
      reroute the change.
     //Uncoarsening stage
14. While (uncoarsening stage)
15.      Choose a local connection $n$ which violates the timing constraint or is untouted.
16.            Global maze routing
17.            Detail maze routing
End
```

Table 3.1: The multilevel routing algorithm.

by using our probabilistic congestion model to construct the initial congestion estimation table. Due to the most critical nets are routed by the L-shaped global routing, we use the L-shaped congestion estimation model for the critical nets and add their probabilistic demands to the congestion estimation table. The detail routing of non-critical nets are not performed until all the critical nets are routed, so they might be routed by the detoured routing. Therefore, we use the accurately detoured congestion estimation model for the non-critical nets and add their probabilistic demands to the congestion estimation table. Our detoured congestion estimation model consisting of the weighted sum of the non-detour mazing routing demand, the detour routing demand with the detoured length being one or two, is used to compute the probabilistic routing demand.

After the preliminary work, our multilevel routing framework starts to route local nets (or local two-pin connections) which are fully placed in a tile at each level. At the coarsening stage, the critical nets are routed by the global and detailed routers. In order to meet timing constraint and reduce the total wire length, a modified L-shaped global router and a detailed maze routing is used to route each critical net. If it is failed to be routed by the detailed routing, it is rerouted by the the Z-shaped global routing method, and a detailed maze routing. However, if both them are failed, this critical net rerouted at the uncoarsening stage. On the other hand, each non-critical net is only routed by the global router at the coarsening stage, and we also precedentially adopt our modified L-shaped global router. If both L-shaped paths of a connection pass through any congested area, we switch to the Z-shaped global routing method. Eventually, if all Z-shaped paths are also through the congested area, the routing method is switched to the global maze routing with allowing detours.

Our global router is based on the shortest path algorithm guided by the congestion table which is constructed by the demand of each connection at each tile. The demand consists of the static and dynamic parts. Before routing, the congestion table only contains the static part constructed by the probabilistic congestion model. When a connection is routed by the global router, the congestion table is updated by subtracting its probabilistic demand and adding its actual routing demand. Moreover, when a connection's global routing type is changed, the congestion table is also updated by subtracting its past actual routing demand and adding its currently actual routing demand. Hence, this updated congestion table is useful to guide the global router to improve the routability and balance congestion.

The routing cost of each routing edge i is proportional to its routing resource demand in the congestion table. To avoid worsening the routing congestion, the routing cost of heavily congested areas should be significantly more than the uncongested areas. Furthermore, in order to prevent from introducing unnecessary detours, the routing cost should not be increased in the lightly congested areas. Therefore, if the routing edge i has demand over 80% of its capacity, the proportional factor, α_i, is set to 1.2. If it has demand under 30% of its capacity, α_i is set to 0. Otherwise, α_i is set to 1. The cost of each routing edge i is equal to $\alpha \times ($*demand of routing edge i / capacity of routing edge i*$)$.

At the end of coarsening stage, the detailed routing is performed on non-critical nets. If there exists routed nets which violate the timing constraints, the *shortest modification heuristic* is adopted to adjust its topology, and these changed connections are rerouted. During the uncoarsening stage, the unroutable connections are refined by the global and detailed maze routing methods, and the techniques of rip-up and reroute. Finally, the timing analysis is performed to check that each net meets its timing constraint. For the violated net, the *shortest modification heuristic* is used to adjust its topology, and the changed connections are rerouted.

3.2 Performance-Driven Routing Topology

Because the interconnect delay has become more serious and dominated the performance of the IC, it is important to construct a performance-driven routing topology. The shortest path tree (SPT) can achieve the best performance but the large amount of total wire length

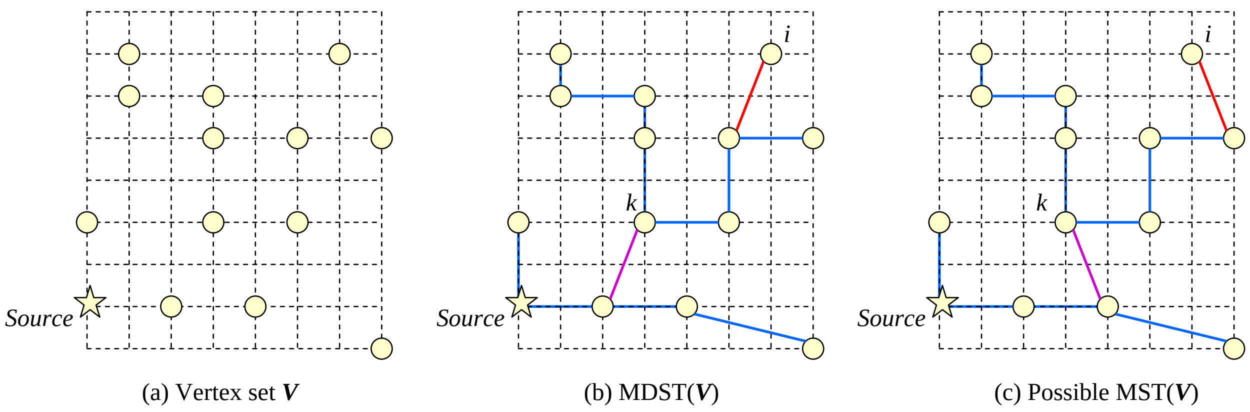


Fig. 3.2: The example of building MDST. (a) A given vertex set ***V***. (b) The topology constructed by MDST. (c) The possible MST.

may result in the nets being unroutable. On the contrary, the minimum spanning tree (MST) produces the minimum total wire length for each net but it might induce the heavily loading of critical paths to degrade the performance. It is not an easy task to construct a better routing topology which has both shorter total wire length and better performance.

The procedure of constructing performance-driven routing topology is divided into two steps. First, we build the initial topology by using a minimum distance spanning tree (MDST) algorithm for each net. The MDST is a MST which has better performance. Then, the *shortest modification heuristic* is employed to modify each initial topology to ensure that each pin doesn't violate the timing constraint.

The MDST algorithm is illustrated in Table 3.2. The idea of our MDST is that there are many short edges having the same cost and so we choose the edge closest to the source from the edges having the same cost to improve the performance. First, we assign the edges which have the same Manhattan distance into the same class, and sort the edges of each class by the less distance between the source of tree and the midpoint of the edge. Then, the classes are ordered by the less Manhattan distance. The rest of steps are the same as the traditional MST algorithm. Fig. 3.2(b) shows an example for the MDST algorithm, and Fig. 3.2(c) shows a possible MST. Compare MDST with traditional MST, the purple line in MST causes that the distance and delay from the $source$ to sink k increase. The red line in MST also make the sink i to have more delay.

```
Algorithm : Minimum Distance Spanning Tree (G)
Input : A connection graph G = (V, E) and source S;
Output : A MDST
Begin
1. Assign the edges having the same cost into a class, and
   sort classes by length from small cost to large ($\varepsilon_1..\varepsilon_i$);
2. For each class($\varepsilon_i$)
   Dis = distance from source to midpoint of edge
   Sort edges by Dis
3. While (exist more than two trees) do
4. For $\varepsilon_1$ to $\varepsilon_{max}$
5.     For each edge$(m, n) \in \varepsilon_i$
6.         If ( $T_m$ != $T_n$)
7. $T_m = T_m \cup T_n$
End
```

Table 3.2: The MDST algorithm.

After the initially directed routing topology is constructed by the MDST algorithm, the *shortest modification heuristic* is applied to modify those MDSTs violating the timing constraints to meet the timing constraints with minimum added wire length. For each net's MDST violating the timing constraint, we utilize the breadth first search (BFS) algorithm to find its first sink ($sink_i$) which doesn't meet the timing constraint, and choose its child ($sink_c$) that is closest to the parent of $sink_i$. Then, we trace back to the source from $sink_c$, and calculate the added wire length if we change an arbitrary sink's grandparent to be its parent. After that, we pick the sink with the minimum added wire length, and change its grandparent to be its parent. Finally, we repeat the above procedure until the routing topology of this net satisfies its timing constant. A simple example by using our minimum distance spanning tree (MDST) algorithm and *shortest modification heuristic* is illustrated in Fig. 3.3, and two real cases shown in Fig. 3.4 and 3.5 demonstrate that our method can get a better routing topology than the recalling modification [1].

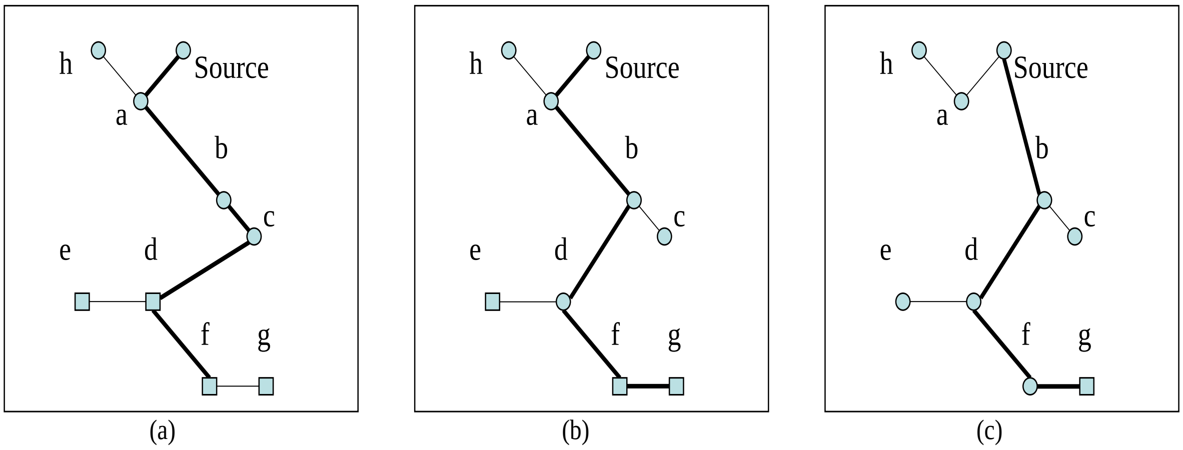


Fig. 3.3: An example of shortest modification. The rectangular sinks violate the timing constraint. (a) Find the minimum added wire length starting from sink f, and the sink d is chosen. (b) Find the minimum added wire length starting from sink g, and b is selected. (c) Continue this approach until all sinks satisfy the timing constraint.

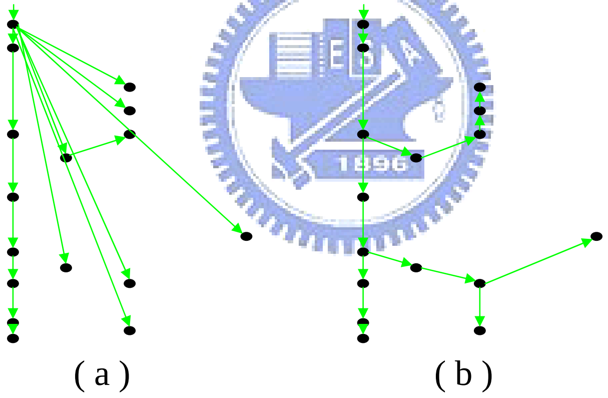


Fig. 3.4: The final routing topology of net 625 in benchmark "S9234" with k=5.5. (a) The result of the recalling modification [1]; (b) The result of our shortest modification heuristic.

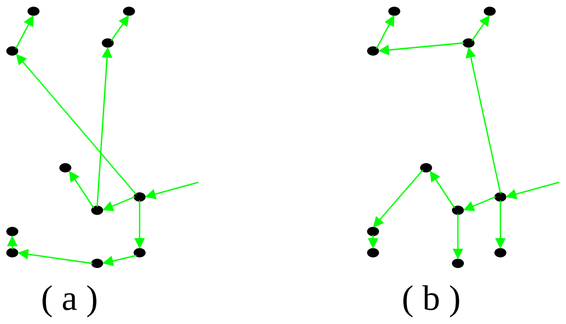


Fig. 3.5: The final routing topology of net 466 in benchmark "S9234" with k=5.5. (a) The result of the recalling modification [1]; (b) The result of our shortest modification heuristic.

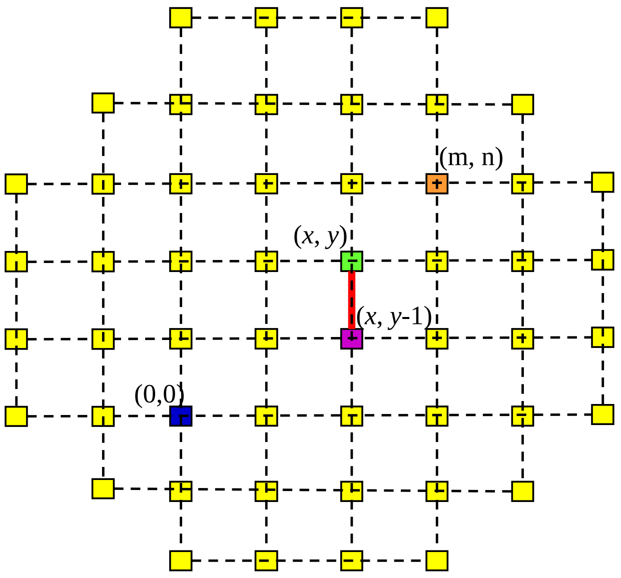


Fig. 3.6: The example of calculating the number $n\_v\_2(x, y, m, n)$.

3.3 Detoured Congestion Estimation

In order to balance the congestion of each routing area, and increase the routability, we derive a more accurate congestion estimation method which allows the maximum number of reverse segments to be two and the maximum length of each detour to be n for our multilevel router. Given a non-critical connection, we first calculate its total number of possible routes with maximum number of reverse segments being two and the maximum detour length being n, and the number of its possible routes which horizontally/vertically pass through each specific tile. The congestion of each horizontal/vertical tile contributed by this non-critical connection is the number of routes passing through it divided by the total number of possible routes. The formulas are listed in the following. For simplicity, we only present the formulas with the maximum detour length being 2, and the general formulas of the maximum detour length being n is presented in Appendix A. The Fig. 3.6 is the example that the start pin of the wire is at $(0, 0)$ node(tile) and the end pin of the wire is at (m, n) node(tile) and the wire may pass through all node(tile).

3.3.1 The Total Number of Routes

Notation 1 $t\_n\_2(m,n)$ is the total number of routes from $(0,0)$ the blue node in Fig. 3.6 to (m,n) the orange node in Fig. 3.6 with the length of detour being two and the number of reverse segment being two.

Theorem 1

$$t\_n\_2(m,n) \quad = \quad \left\{ \begin{array}{l} 2C_{m-1}^{m+n+1} + 2C_{n-1}^{m+n+1} + 2(C_{m+1}^{m+n+1} + C_{m}^{m+n}) + (m+n+4) \\ (C_{m-2}^{m+n} + C_{n-2}^{m+n}) + 3(m+n-2)(C_{m-3}^{m+n-1} + C_{n-3}^{m+n-1}) \\ +(m+n)(m+n-1)(C_{m-4}^{m+n-2} + C_{n-4}^{m+n-2}) \\ +(m+n-1)[(m+n+2)C_{m-1}^{m+n-2} + 2C_{m-2}^{m+n-2} + 2C_{n-2}^{m+n-2})] \end{array} \right. \tag{3.1}$$

3.3.2 The Number of Routes Crossing a Tile

Notation 2 $n\_v\_2(x,y,m,n)$ is the total number of routes from $(0,0)$ to (m,n) with passing through the vertical edge being red edge in Fig. 3.6 $(x,y) \rightarrow (x,y-1)$ with the length of detour being two and the number of reverse segment being two. (x,y) is the purple node $(x,y-1)$ is the green node in Fig. 3.6.

Notation 3 $r\_v\_1(m,n)$ is the total number of routes from $(0,0)$ to (m,n) with the vertical reverse segment not connecting to $(0,0)$ or (m,n) and the number of reverse segment being one.

Notation 4 $r\_h\_1(m,n) = r\_v\_1(n,m)$ is the total number of routes from $(0,0)$ to (m,n) with the horizontal reverse segment not connecting to $(0,0)$ or (m,n) and the number of reverse segment being one.

Lemma 1

$$r\_v\_1(m,n) = mC_{m-1}^{m+n} + (n+1)C_{n-1}^{m+n} \tag{3.2}$$

Notation 5 $r\_v\_2(m,n)$ is the total number of routes from $(0,0)$ to (m,n) with the vertical reverse segment not connecting to $(0,0)$ or (m,n) and the number of reverse segment being two.

Lemma 2

$$r\_v\_2(m,n) = \begin{cases} C_{m-1}^{m+n+1} + (m+n+2)C_{m-2}^{m+n} + 2(m+n-2)C_{m-3}^{m+n-1} \\ +(m+n-3)(m+n-4)C_{m-4}^{m+n-2} + 2C_{n-1}^{m+n+1} \\ +(m+n+4)C_{n-2}^{m+n} + 3(m+n-2)C_{n-3}^{m+n-1} \\ +(m+n-3)(m+n-4)C_{n-4}^{m+n-2} + C_{m+1}^{m+n+1} + 2C_{n+1}^{m+n} \\ +(m+n-1)\left((m+n+1)C_{m-1}^{m+n-2} + 2C_{m-2}^{m+n-2} + C_{n-2}^{m+n-2}\right) \end{cases} \tag{3.3}$$

Theorem 2

$$n\_v\_2(x,y,m,n) = \begin{cases} r\_v\_2(x,y-1)\cdot C_{m-x}^{m-x+n-y} + C_{x}^{x+y-1}\cdot r\_v\_2(m-x,n-y) \\ +r\_v\_1(x,y-1)\cdot r\_v\_1(m-x,n-y) + r\_h\_1(x-1,y) \\ \cdot C_{m-x-1}^{m-x+n-y} + C_{x-1}^{x+y-1}\cdot r\_h\_1(m-x-1,n-y+1) \\ + \sum\limits_{r=max(o,y)}^{min(n+2,y+l)} C_{x-1}^{x-1+r}\cdot C_{m-x-1}^{m-x+n-r+l} \end{cases} \tag{3.4}$$

Notation 6 $n\_h\_2(x,y,m,n) = n\_v\_2(y,x,n,m)$ is the total number of routes from $(0,0)$ *to* (m,n) with passing through the vertical tile $(x,y) \rightarrow (x-1,y)$ *with* the length of detour being two and the number of reverse segment being two.

After computing the number of possible routes, the congestion of each tile contributed by a specific non-critical connection can be estimated by using the same formula shown in [20]. The congestion of vertical tile $(x,y) \rightarrow (x,y-1)$ is equal to

$$cong\_v(x,y,m,n) = \frac{\sum\limits_{i=0}^{2} w_i(m,n) \times n\_v\_i(x,y,m,n)}{\sum\limits_{i=0}^{2} w_i(m,n) \times t\_n\_i(m,n)}, \tag{3.5}$$

where each $w_i(m,n)$ is a weight factor, and

$$n\_v\_1(x,y,m,n) \equiv n\_v(x,y,m,n,1), \tag{3.6}$$

$$n\_v\_0(x,y,m,n) \equiv n\_v(x,y,m,n,0), \tag{3.7}$$

$$t\_n\_1(m,n) \equiv t\_n(m,n,1), \tag{3.8}$$

$$t\_n\_0(m,n) \equiv t\_n(m,n,0). \tag{3.9}$$

The definitions of right hand sides of Equation (3.6)–(3.9) can be found in [20], and their formulas are listed in Appendix B. The congestion of the horizontal tile $(x,y) \rightarrow$

$(x-1, y)$ is similar to Equation (3.5). Finally, we can calculate the congestion of each tile induced by each non-critical connection i, and add them together to estimate the congestion of each tile.

Because the non-critical nets are allowed to detour bypass the heavily congested areas, so their congestions are estimated by the detour routing. The critical wires are almost routed by pattern routing to avoid violating timing constraint. Therefore, their congestions are estimated by the pattern routing. The probabilistic congestion estimation is computed before routing. For accurately estimating congestion, we combine the probabilistic demands which wires have not been yet routed with the real demand which wires are routed. When a net is routed, its probabilistic demand is subtracted from the congestion table and its real routing demand is added to the congestion table. The routing cost of this framework is a linear function of congestion. By this accurate congestion estimation, we can reduce the number of congested areas and improve the routability.

3.4 Congestion-Driven L-shaped Global Routing

Because most connections are routed by the L-shaped global routing [21], we develop a modified L-shaped global routing method to reduce the total wire length like the work [29] and consider the congestion problem. The main idea is to determine what kind of L-shaped routing path (upper-L or lower-L) should be chosen. We use an example illustrated in Fig. 3.7 to explain our approach. Fig. 3.7 shows that the choice of the different kind of L-shaped routing path for connection i. First, we find the connection of the parent of the connection i to calculate the overlapping wire length of the upper-L or lower-L global routing. Then, we also find all the connection of the children of the connection i to calculate the overlapping wire length of the upper-L or lower-L global routing. Finally, we choose the type of L-shaped routing path with maximum overlapping wire length. Fig. 3.7 shows that the lower-L should be chosen for connection i since the saved wire length L_{low} is larger than L_{top}. However, if the chosen path passes through any heavily

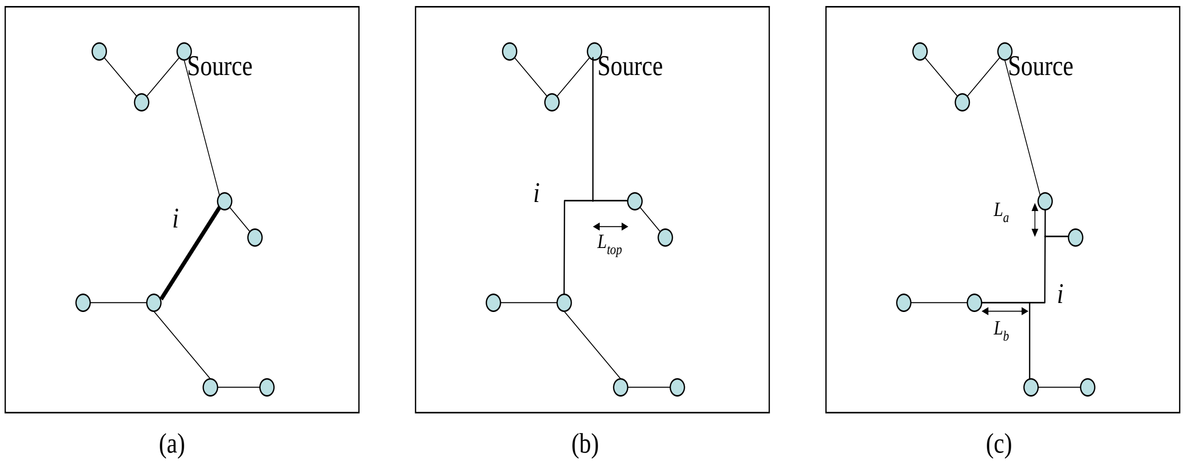


Fig. 3.7: An example of choosing upper-L or lower-L. (a) Pattern route is used for edge i. (b) If edge i is upper-L routing, the saved wire length is L_{top}. (c) If edge i is lower-L routing, the saved wire length is $L_{low}(= L_a + L_b)$.

congested area, the other path should be selected. If both paths pass through any heavily congested area, the path which has the minimum total routing cost is selected.selected.

Chapter 4

Experimental Results

We implement our multilevel routing system in C++ language, and test it on a Pentium IV 3.2G Hz machine with 2GB memory. The results are compared with [1] and [2] based on six benchmark circuits. In Table 4.1, "Size" represents the layout dimension, "#Layers" is the number of routing layers, and "#Nets" represents the number of two-pin nets. The parameters of resistance and capacitance, and the model of via are the same as [1] and [2]. Like [1] and [2], we construct a shortest path tree for each net by connecting all sinks directly to their net source to obtain their timing constraints. We then assign the timing bound of each sink as the multiplication of the constant k and the shortest path delay of the net.

The ratio of timing constraint, *k*, defined in [1], is also set equivalent for comparison.

First, we set the timing constraint ratio k used in [1, 2] to be 5.5 to obtain the comparison of routability. Table 4.2 demonstrates that our multilevel routing approach obtains

Circuits	Size(*um*)	#Layers	#Nets	#Pins
S5378	4330*2370	3	3124	4734
S9234	4020*2230	3	2774	4185
S13207	6590*3640	3	6995	10562
S15850	7040*3880	3	8321	12566
S38417	11430*6180	3	21035	32210
S38584	12940*6710	3	28177	42589

Table 4.1: The benchmark circuits.

Circuits	Result of [1]			Result of [2]			Our Result		
	Time (s)	Wire length	Cmp. Rate	Time (s)	Wire length	Cmp. Rate	Time (s)	Wire length	Cmp. Rate
S5378	35	9.5e7	99.7%	10.6	8.4e7	99.8%	7.75	9.0e7	100%
S9234	26.2	7.7e7	99.7%	8.1	6.0e7	99.9%	5.67	7.2e7	100%
S13207	106.7	2.4e8	99.8%	22.6	2.3e8	99.8%	23.1	2.2e8	100%
S15850	538.8	2.9e8	99.3%	62.6	2.9e8	99.7%	54.1	2.8e8	100%
S38417	899.9	7.2e8	99.5%	71.3	8.0e8	99.8%	183.9	6.9e8	100%
S38584	1953.7	9.3e8	99.6%	255.6	1.1e9	99.8%	369.4	9.7e8	100%
Comp.		1.04	99.6%		1.02	99.8%		1	100%

Table 4.2: Comparison of run-time and routability with timing constraint ratio *k*=5.5 between [1], [2], and our router. Note: [1] and [2] were run on a 1 GHz Sun Blade 2000 with 1 GB memory; our approach was run on a P4 3.2GHz with 2GB memory.

Circuits	Result of [1]			Result of [2]			Our Result		
	D_{max} (ps)	D_{avg} (ps)	Cmp. Rate	D_{max} (ps)	D_{avg} (ps)	Cmp. Rate	D_{max} (ps)	D_{avg} (ps)	Cmp. Rate
S5378	13651	798	94.6%	12854	751	95.2%	13497	773	100%
S9234	11426	659	94.3%	10019	599	94.2%	10130	646	100%
S13207	20149	749	93.1%	18769	693	94.4%	18879	729	100%
S15850	28049	859	93.1%	25221	743	93.6%	27294	845	99.9%
S38417	40500	702	93.4%	38957	670	93.7%	39721	688	99.9%
S38584	129267	739	93.7%	129267	655	94.0%	53869	716	99.9%
Comp.	1.28	1.03	-6.2%	1.21	0.94	-5.7%	1	1	1

Table 4.3: Comparison of run-time and routability with timing constraint ratio *k*=2 between [1], [2], and our multilevel routing.

Circuits	Result of [1]		Our Result	
	#Cong. areas	Max. cong.	#Cong. areas	Max. cong.
S5378	11	0.611	1	0.500
S9234	10	0.780	1	0.504
S13207	8	0.624	1	0.506
S15850	12	0.594	1	0.523
S38417	28	0.635	5	0.532
S38584	25	0.608	34	0.566
Comp.	7.89	1.23	1	1

Table 4.4: Comparison of routing solutions with number of congested area and maximum congestion. The timing constraint ratio *k* is 5.5.

significantly better routing solutions than the results of [1] and [2]. Our method can achieve 100% routing completion rate for each benchmark, and can reduce the total wire length for most of the benchmarks. However, none of them can be completely routed by [1, 2]. With the stringent timing constraint ratio (k=2), Table 4.3 shows that our scheme also produces outstanding completion rates, and reduces the average and maximum delay compared with [1].

Finally, our results are compared with [1] in Table 4.4 for the routing congestion. Therefore, the original net might pass through heavily congested area, lead to more congested area, and induce the not yet routed net unroutable. In Table 4.4, "#Cong. areas" is the number of tiles which congestion is over 0.5. The results show that the proposed router can remarkably reduce the number of congested tiles in most benchmark, and decrease the maximum congestion to balance the congestion for all benchmarks. In benchmark "s38584", although our congested areas are more than [1], we efficiently reduce the maximum congestion $(0.608 \rightarrow 0.566)$ by allotting the wire to the uncongested area to balance the routing congestion and also route completely this benchmark. Fig. 4.1, 4.2, 4.3, 4.4, 4.5, and 4.6 (Upper) and (lower) display the routing solution of all benchmark from [1] and our approach under the timing constraint ratio $(k = 5.5)$, respectively. The blue line is the first layer routing horizontal direction, the red line is the second layer routing vertical direction, and the green line is the third layer routing horizontal direction. Fig. 4.1 illustrates our second and third layer are lighter , and furthermore our routability is 100%. Fig. 4.2 obviously shows that our routing solution have uniform routing to balance the congestion. The Fig. 4.3, 4.4, and 4.5 also show that we have better routing solutions to balance the congestion. In Fig. 4.6, our second layer is lighter, and we utilize more interconnection on the third layer in order to completely route all nets.

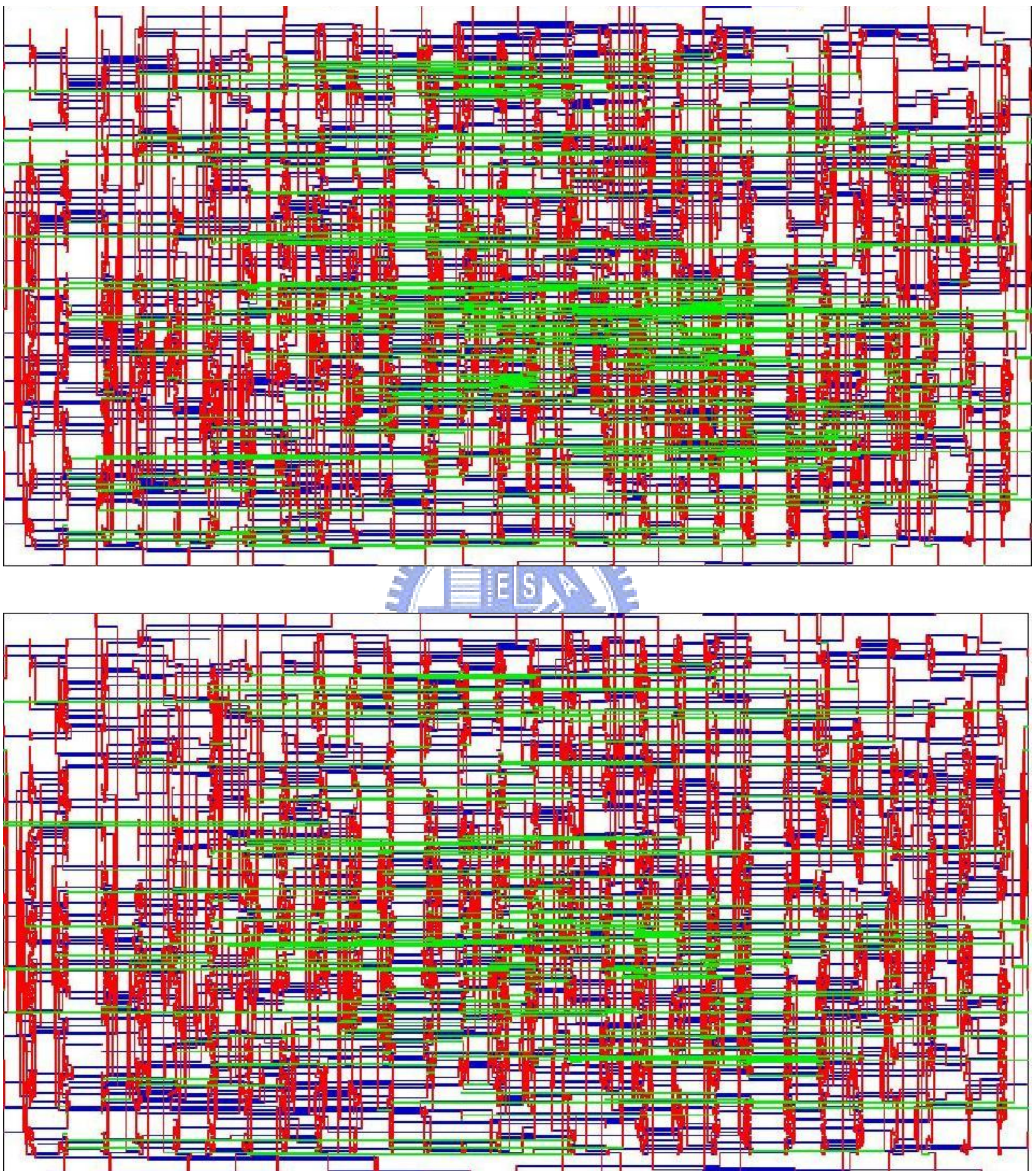


Fig. 4.1: The routing solution of benchmark “S5378” ($k = 5.5$). (Upper) The solution routed by [1] and the completion rate = 99.7%; (Lower) The solution routed by our method and the completion rates = 100%.

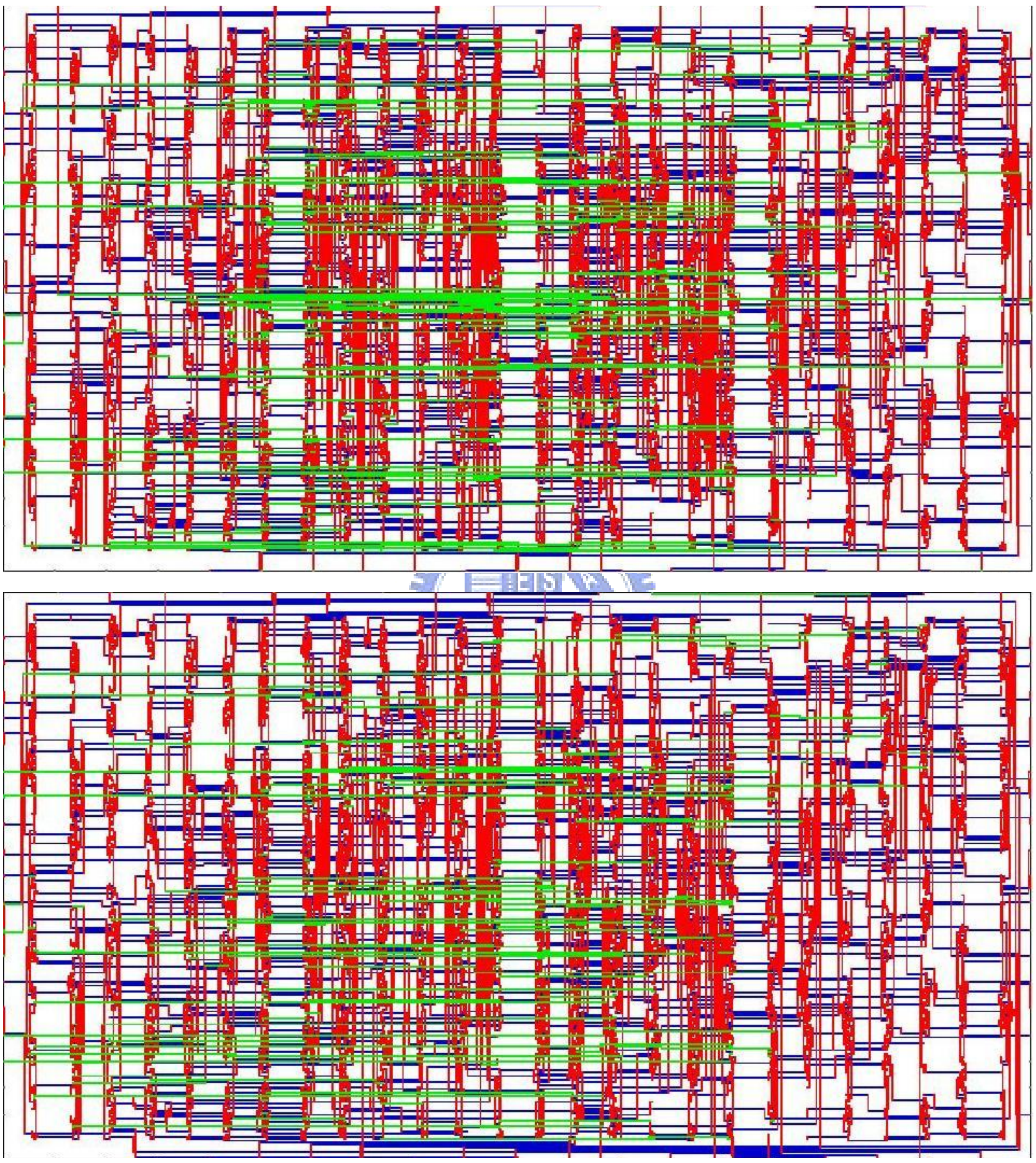


Fig. 4.2: The routing solution of benchmark “S9234” ($k = 5.5$). (Upper) The solution routed by [1] and the completion rate = 99.7%; (Lower) The solution routed by our method and the completion rates = 100%.

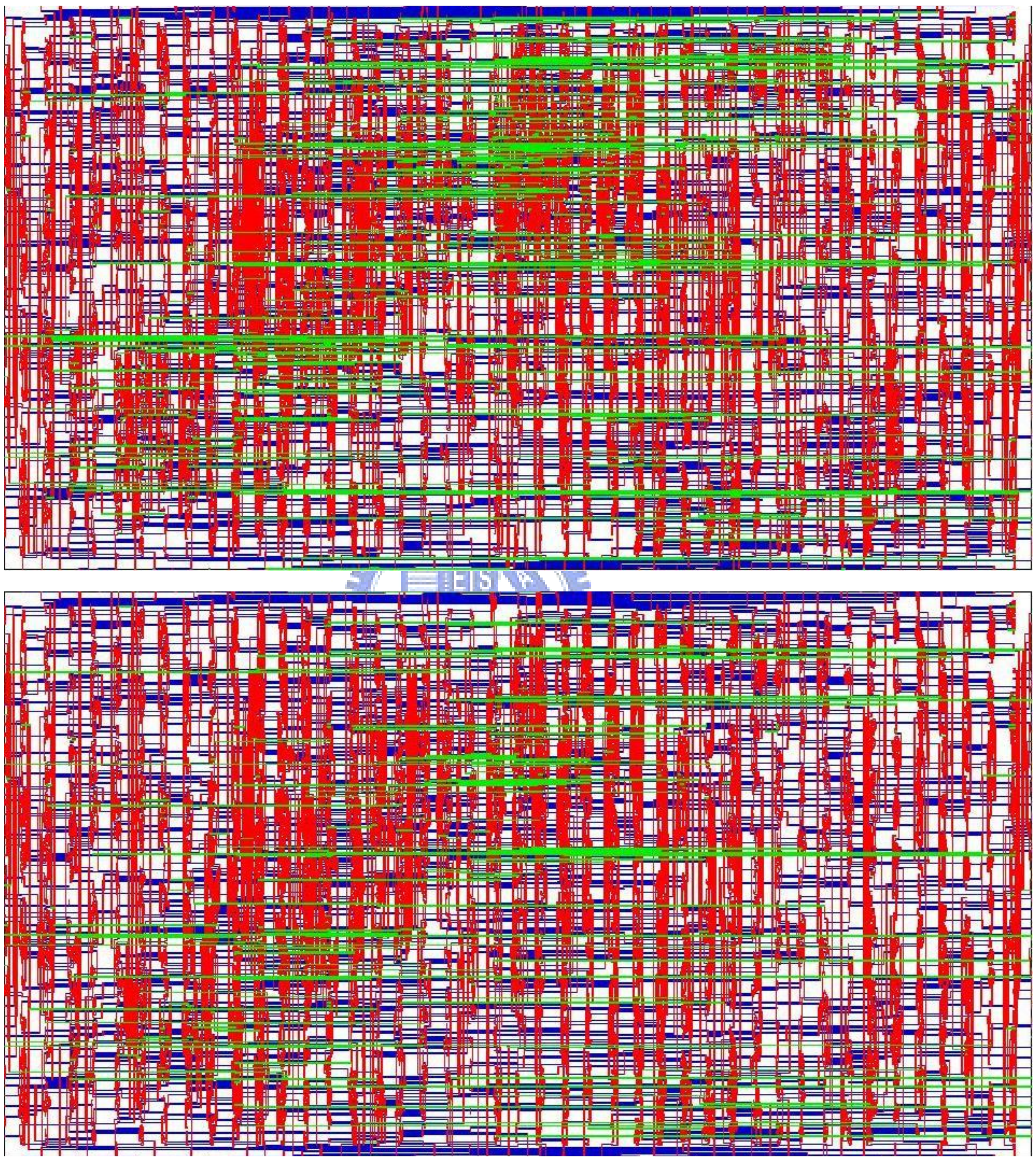


Fig. 4.3: The routing solution of benchmark "s13207" ($k = 5.5$). (Upper) The solution routed by [1] and the completion rate = 99.8%; (Lower) The solution routed by our method and the completion rates = 100%.

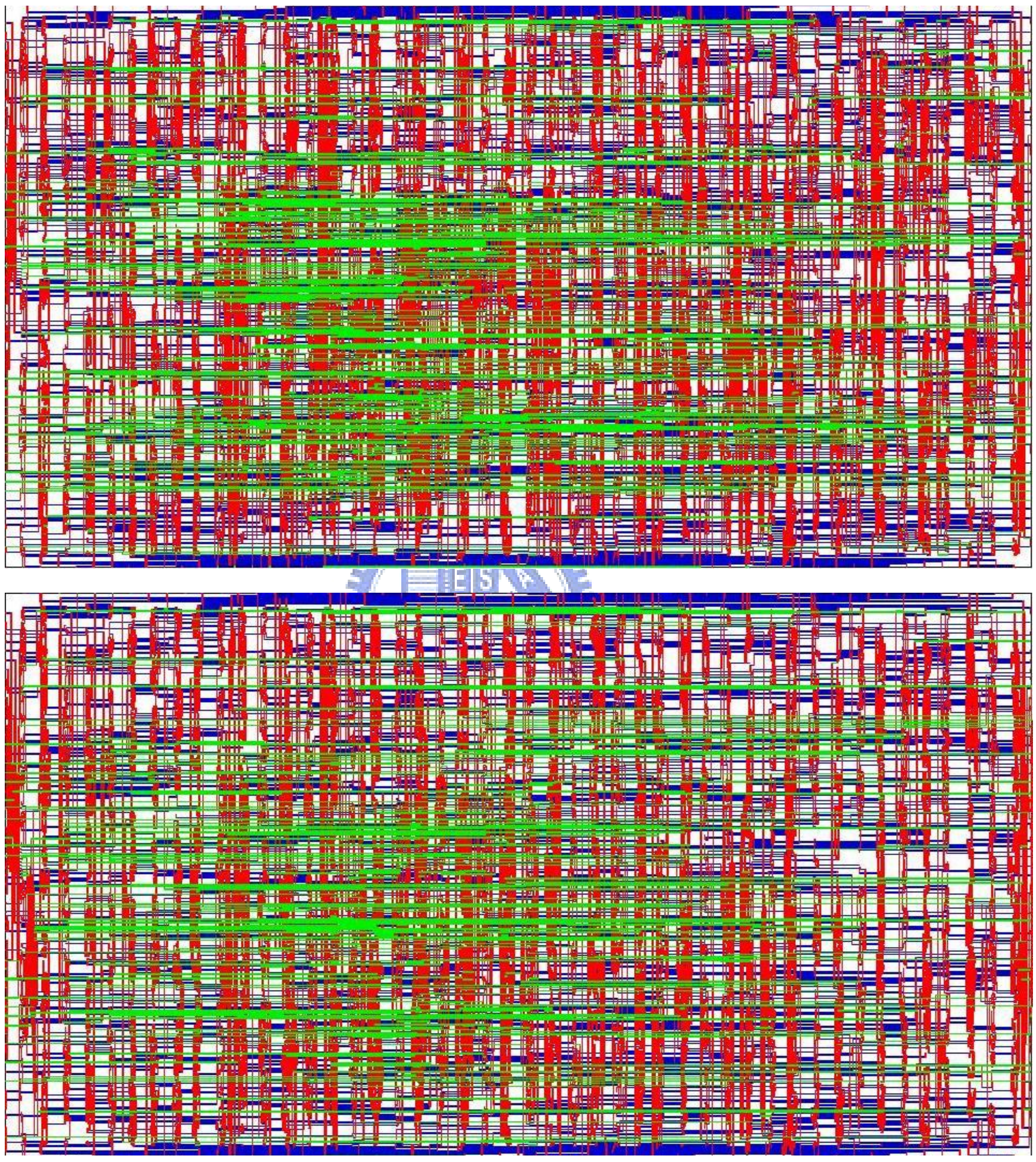


Fig. 4.4: The routing solution of benchmark "s15850" ($k = 5.5$). (Upper) The solution routed by [1] and the completion rate = 99.3%; (Lower) The solution routed by our method and the completion rates = 100%.

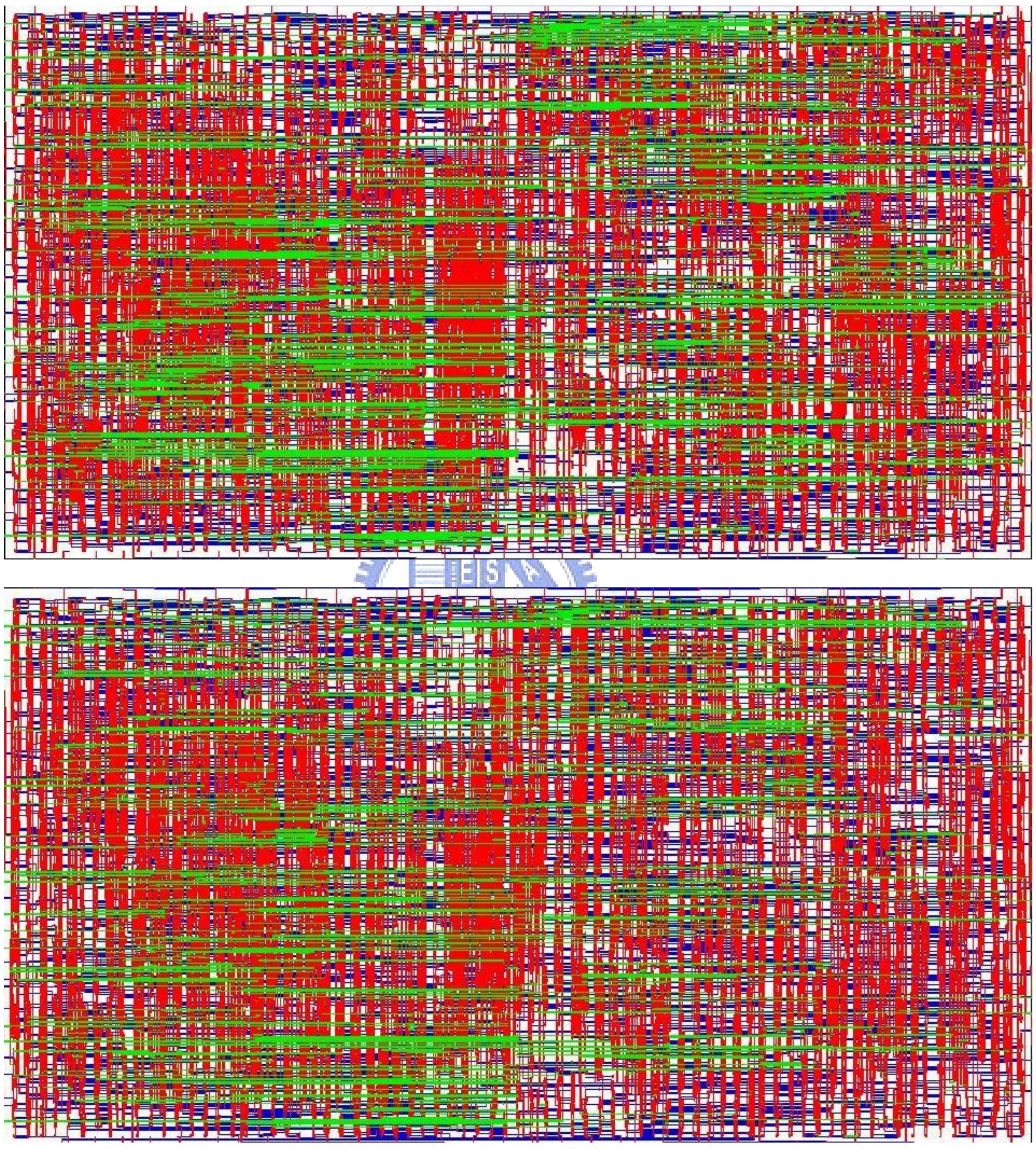


Fig. 4.5: The routing solution of benchmark “s38417” ($k = 5.5$). (Upper) The solution routed by [1] and the completion rate = 99.5%; (Lower) The solution routed by our method and the completion rates = 100%.

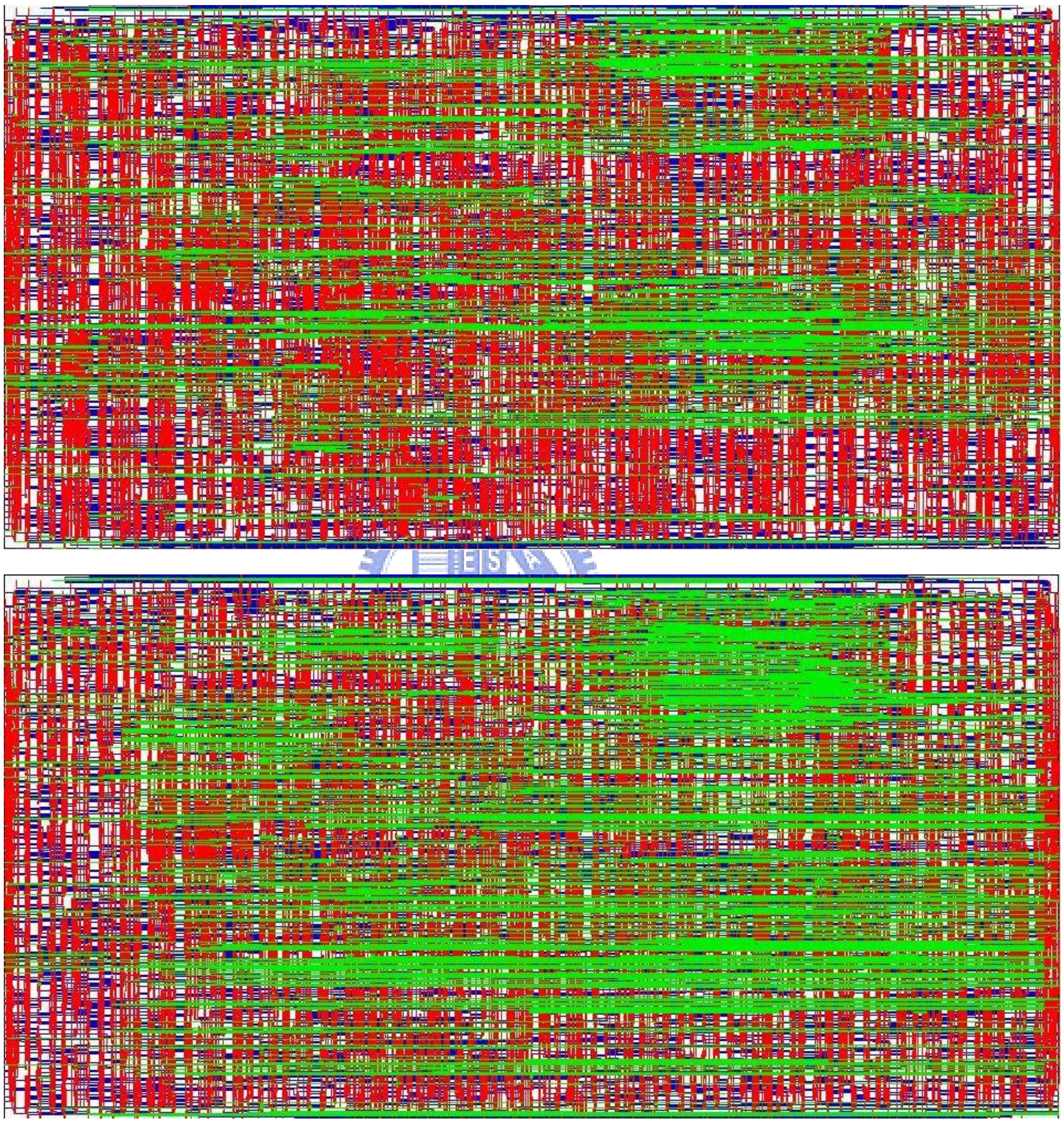


Fig. 4.6: The routing solution of benchmark “s38584” ($k = 5.5$). (Upper) The solution routed by [1] and the completion rate = 99.6%; (Lower) The solution routed by our method and the completion rates = 100%.

Chapter 5

Conclusion

We have already presented a novel performance- and congestion-driven multilevel routing framework to decrease the congestion and achieve better performance as well as routability. Our framework is different from the works [1, 2]. The exact congestion table is built by the probabilistic congestion estimation at the preliminary stage, and is built by the probabilistic and actual congestion estimation at the coarsening. The congestion estimation in cite [1, 2] is built by the actual congestion at coarsening stage. We separate all nets into the critical and non-critical nets, and the detail router is used only for critical net for decreasing the added delay to fix the topology. The stable topology is useful for accurately estimating congestion.

By performing the minimum distance spanning tree (**MDST**) algorithm and the **shortest modification heuristic** to construct the performance-driven topology of all nets, we have obtained a better routing topology which satisfies the timing constraint and has consumed less total wire length. Our proposed minimum distance spanning tree (**MDST**) can build a minimum spanning tree (*MST*) with better performance to be the initial topology. The initial topology violating the timing constraint is modified by the *shortest modification heuristic* to meet the timing constraint. The Fig. 3.4 and 3.5 demonstrated that our topology is better than [1].

We proposed the detoured congestion estimation model which is accurate than the work [20]. Combining the detoured congestion estimation model used for the non-critical nets with L-shaped congestion estimation model used for the critical net estimate the probabilistic congestion. The congestion table associated the statically probabilistic congestion for not yet routed nets with the dynamically actual congestion for routed nets to guide the net to avoid congested areas. The Table 4.4 showed that our router can reduce and balance routing congestion.

The modified L-shaped global routing could reduce the total wire length, and the Fig. 3.7 showed that our approach found the upper-L or lower-L to save the maximum routing resource and couldn't pass any congested area.

The experimental results have shown that our scheme can obtain dramatic routability under any timing constraint ratios, and balance the congestion.

Appendix A

Probabilistic Estimation for Detour

Chapter 3.3 has listed the formulation with the maximum detour length being 2 for simplification. This appendix demonstrates the formulas with the maximum detour length being l (≥ 2). First, we develop the formula of the total number of routes with the detour length being l and the number of reverse segment being two. Then, we show the number of routes crossing the horizontal/vertical tile with the detoured length being l and the number of reverse segment being two.

A.1 Total Number of Routes

$t\_n\_2(m, n, l)$ is the total number of routes from $(0, 0)$ to (m, n), with the length of detour being l and the number of reverse segment being two. The forward segments are right (R) and upper (U) direction, and the reverse segments are left (L) and down (D) direction. The $t\_n\_2(m, n, l)$ is the sum of the number having only down reverse segments, the number having only left reverse segments, and the number having simultaneously one down and one left reverse segment. This approach is the permutation combination that permute some R, U, L, and D segments.

A.1.1 Only Down Reverse Segment

First, we develop the number of routes having only down reverse segments. We separate the sum into six parts as follow.

The first part is that the number of down segment is only one in the boundaries $(DR....or....RD)$. The number of upper segments is $n + l$, and the number of right segments is $m - 1$. The number of route is

$$2C_{m-1}^{m+n+l-1} \tag{A.1}$$

The second part is that the number of down segment is only one in the medium$(....RDR....)$. The number of upper segments is $n + l$, and the number of right segments is $m - 2$ part. The number of route is

$$C_{1}^{m+n+l-1}C_{m-2}^{m+n+l-2} \tag{A.2}$$

The third part is that one of two down segments has 1 $l-1$ length in the boundary and other has $l - 1$ 1 length in the medium $(DR....RDR....or....RDR....RD)$. The number of upper segments is $n + l$, and the number of right segments is $m - 3$. The number of route is

$$2(l-1)C_{1}^{m+n+l-2}C_{m-3}^{m+n+l-3} \tag{A.3}$$

The fourth part is that two down segments both are in the medium$(....RDR....RDR....)$ and the length of one of two down segments is 1 $l - 1$ and the length of other down segment is $l - 1$ 1. The number of upper segments is $n + l$ and the number of right segment is $m - 4$ part. The number of route is

$$\left\lceil \frac{l-1}{2} \right\rceil C_{1}^{m+n+l-2}C_{1}^{m+n+l-3}C_{m-4}^{m+n+l-4} \tag{A.4}$$

The fifth part is that two down segments are both in the boundary ($DR....RD$ or $DRDR....$ or $....RDRD$). The number of upper segment is $n+l$ and the number of right segments is $m - 2$. The number of route is

$$3(l-1)C_{m-2}^{m+n+l-2} \tag{A.5}$$

Th sixth part is that two down segments are connected by a right part ($....RDRDR....$). The number of upper segment is $n + l$ and the number of right segment is $m - 3$. The number of route is

$$(l-1)C_1^{m+n+l-2}C_{m-3}^{m+n+l-3} \tag{A.6}$$

The number of routes having only down reverse segments is the sum of A.1, A.2, A.3, A.4, A.5, and A.6:

$$t\_n\_D(m,n,l) \;=\; \begin{cases} 2C_{m-1}^{m+n+l-1} + (m+n+4l-4)C_{m-2}^{m+n+l-2} \\ +3(l-1)(m+n+l-2)C_{m-3}^{m+n+l-3} \\ +\left\lceil \frac{l-1}{2} \right\rceil (m+n+l-2)(m+n+l-3)C_{m-4}^{m+n+l-4} \end{cases} \tag{A.7}$$

A.1.2 Only Left Reverse Segment

Secondly, we develop the number of routes having only left reverse segments. We also separate the sum into six parts as follow.

The first part is that the number of left segment is only one in the boundaries $(LU....$ or $....UL)$. The number of upper segments is $n-1$, and the number of right segments is $m+l$. The number of route is

$$2C_{n-1}^{m+n+l-1} \tag{A.8}$$

The second part is that the number of lest segment is only one in the medium$(....ULU....)$. The number of upper segments is $n-2$, and the number of right segments is $m+l$ part. The number of route is

$$C_1^{m+n+l-1}C_{n-2}^{m+n+l-2} \tag{A.9}$$

The third part is that one of two left segments has 1 $l-1$ length in the boundary and other has $l-1$ 1 length in the medium $(LU....ULU....or....ULU....UL)$. The number of upper segments is $n-3$, and the number of right segments is $m+l$. The number of route is

$$2(l-1)C_1^{m+n+l-2}C_{n-3}^{m+n+l-3} \tag{A.10}$$

The fourth part is that two left segments both are in the medium$(....ULU....ULU....)$ and the length of one of two left segments is 1 $l-1$ and the length of other left segment is $l-1$ 1. The number of upper segments is $n-4$ and the number of right segment is $m+l$

part. The number of route is

$$\left\lceil \frac{l-1}{2} \right\rceil C_{1}^{m+n+l-2} C_{1}^{m+n+l-3} C_{n-4}^{m+n+l-4} \tag{A.11}$$

The fifth part is that two left segments are both in the boundary ($UL....LU$ or $LULU....$ or $....ULUL$). The number of upper segment is $n-2$ and the number of right segments is $m+l$. The number of route is

$$3(l-1)C_{n-2}^{m+n+l-2} \tag{A.12}$$

Th sixth part is that two left segments are connected by a upper part ($....ULULU....$). The number of upper segment is $n-3$ and the number of right segment is $m+l$. The number of route is

$$(l-1)C_{1}^{m+n+l-2} C_{n-3}^{m+n+l-3} \tag{A.13}$$

The number of routes having only left reverse segments is the sum of A.8, A.9, A.10, A.11, A.12, and A.13:

$$t\_n\_L(m,n,l) = \left\{ \begin{array}{l} 2C_{n-1}^{m+n+l-1} + (m+n+4l-4)C_{n-2}^{m+n+l-2} \\ +3(l-1)(m+n+l-2)C_{n-3}^{m+n+l-3} \\ +\left\lceil \frac{l-1}{2} \right\rceil (m+n+l-2)(m+n+l-3)C_{n-4}^{m+n+l-4} \end{array} \right. \tag{A.14}$$

A.1.3 Down and Left Reverse Segments at the Same Time

Finally, we develop the number of routes having simultaneously down and left reverse segments. We also separate the sum into six parts as follow.

The first part is that the reverse segments are both in the boundary ($DLU....or....ULD$ or $LDR....or....RDL$), and the (a, b) in the Fig. A.1 illustrate to avoid calculating the impossible situation. The number of route is

$$2\sum_{a=1}^{l-1} \left[\sum_{k=0}^{a-1} \left(C_{k}^{l+k-a-1} C_{m+a-k}^{m+n+a-k-1} + C_{k}^{a+k-1} C_{n+l-a-k}^{m+n+l-a-k-1} \right) \right] \tag{A.15}$$

The second part is that the reverse segments are in the different boundary ($DR....UL$ or $LU....RD$). The number of route is

$$2\sum_{a=1}^{l-1} C_{m+a-1}^{m+n+l-2} \tag{A.16}$$

The third part is that the down segment is in the boundary and the left segment is in the medium $(DR....ULU....or...ULU....RD)$. The (c, d) in the Fig. A.1 show to avoid calculating the impossible situation. The number of route is

$$2\sum_{a=1}^{l-1}\left(\begin{array}{l}\sum\limits_{k=0}^{l-a-1} C_k^{a+k-1} C_1^{m+n+l-a-k-2} C_{m-1}^{m+n+l-a-k-3} \\ +\sum\limits_{k=0}^{a-1} C_k^{l+k-a} C_1^{m+n+a-k-2} C_{n-2}^{m+n+l-a-k-3}\end{array}\right) \tag{A.17}$$

The fourth part is that the left segment is in the boundary and the down segment is in the medium $(LU....RDR....or....RDR....UL)$. The (e, f) in the Fig. A.1 show to avoid calculating the impossible situation. The number of route is

$$2\sum_{a=1}^{l-1}\left(\begin{array}{l}\sum\limits_{k=0}^{a-1} C_k^{l+k-a-1} C_1^{m+n+a-k-2} C_{n-1}^{m+n+a-k-3} \\ +\sum\limits_{k=0}^{l-a-1} C_k^{a+k} C_1^{m+n+l-a-k-2} C_{m-2}^{m+n+l-a-k-3}\end{array}\right) \tag{A.18}$$

The fifth part is that the left and down segments are in the medium, and the down segment is in the front of the left segment $(....RDR....ULU....)$. The (g) in the Fig. A.1 shows to avoid calculating the impossible situation. The number of route is

$$\sum_{a=1}^{l-1}\sum_{c=0}^{m-2}\sum_{d=0}^{n-2} C_c^{c+d}\left(\begin{array}{l}\sum\limits_{k=0}^{a} C_k^{a+k-1} C_1^{m+n+l-a-c-d-k-3} C_{m-c-2}^{m+n+l-a-c-d-k-4} \\ +\sum\limits_{k=0}^{a-1} C_k^{l+k-a} C_1^{m+n+a-c-d-k-3} C_{n-d-2}^{m+n+a-c-d-k-4}\end{array}\right) \tag{A.19}$$

The sixth part is that the left and down segments are in the medium, and the left segment is in the front of the down segment $(....ULU....RDR....)$. The number of route is

$$\sum_{a=1}^{l-1}\sum_{c=0}^{m-2}\sum_{d=0}^{n-2} C_c^{c+d}\left(\begin{array}{l}\sum\limits_{k=0}^{a} C_k^{l+k-a-1} C_1^{m+n+a-c-d-k-3} C_{n-d-1}^{m+n+a-c-d-k-4} \\ +\sum\limits_{k=0}^{a-1} C_k^{a+k} C_1^{m+n+l-a-c-d-k-3} C_{m-c-3}^{m+n+l-a-c-d-k-4}\end{array}\right) \tag{A.20}$$

The number of routes having simultaneously down and left reverse segments is the sum of A.15, A.16, A.17, A.18, A.19, and A.20.

$$t\_n\_DL(m,n,l) = A.15 + A.16 + A.17 + A.18 + A.19 + A.20. \tag{A.21}$$

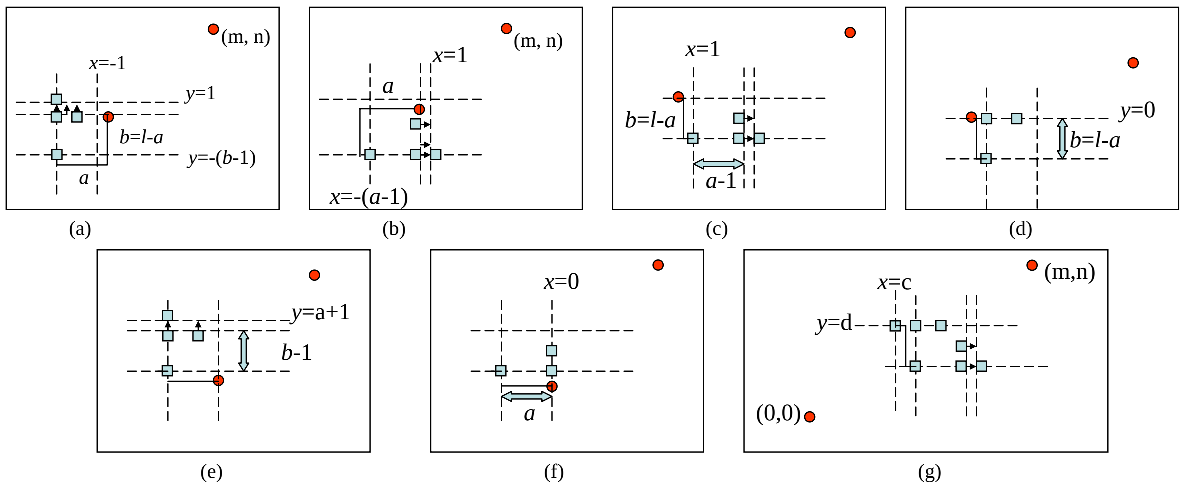


Fig. A.1:

The total number of routes from $(0, 0)$ to (m, n) $t\_n\_2(m, n, l)$, with the length of detour being l and the number of reverse segments being two, is the sum of A.7, A.8, and A.21.

$$t\_n\_2(m, n, l) = t\_n\_D(m, n, l) + t\_n\_L(m, n, l) + t\_n\_DL(m, n, l) \tag{A.22}$$

A.2 The Number of Routes Crossing a Tile

Notation 7 $n\_v\_2(x, y, m, n, l)$ is the total number of routes from $(0, 0)$ to (m, n) with passing through the vertical tile $(x, y) \to (x, y - 1)$, the length of detour being $l(\geq 2)$ and the number of reverse segment being two.

Notation 8 $r\_v\_1(m, n, l)$ is the total number of routes from $(0, 0)$ to (m, n), and the down(vertical) reverse segment can't connect to $(0, 0)$ or (m, n), and the number of reverse segment is one, and the length of detour is l.

Notation 9 $r\_h\_l(m, n, l)$ is the total number of routes from $(0, 0)$ to (m, n), and the left(horizontal) reverse segment can't connect to $(0, 0)$ or (m, n), and the number of reverse segment is one, and the length of detour is l. $r\_h\_l(m, n, l) = r\_v\_l(n, m, l)$

Lemma 3

$$r\_v\_l(m, n, l) = mC_{m-1}^{m+n+l-1} + (n + 1)C_{n-1}^{m+n+l-1} \tag{A.23}$$

Notation 10 $r\_v\_2(m,n,l)$ is the total number of routes from $(0,0)$ to (m,n), and the down(vertical) reverse segment can't connect to $(0,0)$ or (m,n) and the number of reverse segment is two, and the length of detour being l.

Lemma 4

$$r\_v\_2(m,n,l) = \begin{cases} C_{m-1}^{m+n+l-1} + (m+n+2l-2)C_{m-2}^{m+n+l-2} + 2(l-1)(m+n+l-2) \\ C_{m-3}^{m+n+l-3} + \lceil \frac{l-1}{2} \rceil (m+n+l-2)(m+n+l-3) \\ C_{m-4}^{m+n+l-4} + 2C_{n-1}^{m+n+l-1} + (m+n+4l-4)C_{n-2}^{m+n+l-2} \\ +3(l-1)(m+n+l-2)C_{n-3}^{m+n+l-3} + \left\lceil \frac{l-1}{2} \right\rceil (m+n+l-2) \\ (m+n+l-3)C_{n-4}^{m+n+l-4} + t\_n\_UL(m,n,l) - \frac{A.15}{2} - \frac{A.17}{2} - \frac{A.18}{2} \end{cases} \tag{A.24}$$

Theorem 3

$$n\_v\_2(x,y,m,n,l) = \begin{cases} r\_v\_2(x,y-1,l) \cdot C_{m-x}^{m-x+n-y} + C_{x}^{x+y-1} \cdot r\_v\_2(m-x,n-y,l) \\ +\sum_{a=1}^{l-1} r\_v\_1(x,y-1,a) \cdot r\_v\_1(m-x,n-y,l-a) \\ +\sum_{a=1}^{l-1} \sum_{r=max(o,y)}^{min(n+a,y-a+l)} (r\_h\_1(x-1,r,l-a) \cdot C_{m-x-1}^{m-x-1+n-y+a} \\ +C_{x-1}^{x+r-1} \cdot r\_h\_1(m-x-1,n-r+a,l-a)) \\ + \sum_{r=max(o,y)}^{min(n+l,y-1+l)} C_{x-1}^{x-1+r} \cdot C_{m-x-1}^{m-x+n-r-l+l} \end{cases} \tag{A.25}$$

Notation 11 $n\_h\_2(x,y,m,n,l)$ is the total number of routes from $(0,0)$ to (m,n) with passing through the horizontal tile $(x,y) \rightarrow (x-1,y)$, the length of detour being $l(\geq 2)$ and the number of reverse segment being two. $n\_h\_2(x,y,m,n,l) = n\_v\_2(y,x,n,m,l)$

Appendix B

The Formulations in [20]

The Appendix A showed the congestion estimation with the maximum detour length being $l(\geq 2)$. In this appendix, we show the formulations with the detour length being 1 and 0, and the formulations have proposed in the [20].

The total number of routes from $(0,0)$ to (m,n) without detouring is $t\_n\_0(m,n) \equiv t\_n(m,n,0)$ which $t\_n(m,n,0)$ can be found in the [20].

$$t\_n\_0(m,n) = C_m^{m+n} \tag{B.1}$$

The total number of routes from $(0,0)$ to (m,n) with the detour length being 1 is $t\_n\_1(m,n) \equiv t\_n(m,n,1)$ which $t\_n(m,n,1)$ can be found in the [20].

$$t\_n\_1(m,n) = mC_{m-1}^{m+n} + (n+1)C_{n-1}^{m+n} \tag{B.2}$$

The total number of routes from $(0,0)$ to (m,n) with passing through the vertical tile $(x,y) \rightarrow (x,y-1)$ without detouring is $n\_v\_0(x,y,m,n) = n\_v(x,y,m,n,0)$.

$$n\_v\_0(x,y,m,n) = C_x^{x+y} \cdot C_{m-x}^{m-x+n-y} \tag{B.3}$$

The total number of routes routes from $(0,0)$ to (m,n) with passing through the vertical tile $(x,y) \rightarrow (x,y-1)$ is $n\_v\_1(x,y,m,n) = n\_v(x,y,m,n,1)$ which the detour length is 1.

$$n\_v\_1(x,y,m,n) = \begin{cases} r\_v(x,y-1,l) \cdot C_{m-x}^{m-x+n-y} + \\ C_x^{x+y-1} \cdot r\_v(m-x,n-y,l) + \\ \sum\limits_{r=max(o,y)}^{min(y+l-1,n+l)} C^{x-1,r} C_{m-x-1}^{m+x+l-x-r-1} \end{cases} \tag{B.4}$$

Bibliography

[1] S.-P. Lin and Y.-W. Chang, “A novel framework for multilevel routing considering routability and performance”, in *Proc. of Int. Conf. Computer-Aided Design*, pp. 44-50, November 2002.

[2] T.-Y. Ho, Y.-W. Chang, S.-J. Chen, and D. T. Lee, “A fast crosstalk- and performance-driven multilevel routing system”, in *Proc. of Int. Conf. Computer-Aided Design*, pp. 382-387, November 2003.

[3] C. Y. Lee, “An algorithm for path connection and its application”, *IRE Trans. on Electronic Computers*, EC-10, 1961.

[4] S. B. Akers, “A modification of Lee’s path connection algorithm”, *IEEE Trans. on Electronic Computers*, pp. 97-98, February 1967.

[5] D. Hightower, ”A solution to line routing problems on the continuous plane”, in*Proc. IEEE 6th Design AutomationWorkshop*, 1969, pp. 1V24.

[6] G. Meixner and U. Lauther, “A new global router based on a flow model and linear assignment”, in *Proc. of Int. Conf. Computer-Aided Design*, pp. 44-47, November 1990.

[7] G. Karypis, R. Aggarwal, V. Kumar, and S. shekhar, ”Multilevel hypergraph partitioning: Application in VLSI domain”, *IEEE Trans. VLSI Systems*, vol. 7, pp. 69V79, March 1999.

[8] C. J. Alpert, J.-H. Huang, and A. B. Kahng, ”Multilevel circuit partitioning”, *IEEE Trans. Computer-Aided Design*, vol. 17, no. 8, pp. 655V667, August 1998.

[9] J. Cong, S. Lim, and C. Wu, ”Performance driven multilevel and multiway partitioning with retiming”, *Proc. of Design Automation Conf.*, pp. 274V279, June 2000.

[10] T. Chan, J. Cong, T. Kong, and J. Shinnerl, ”Multilevel optimization for large-scale circuit placement”, *Proc. of Int. Conf. Computer-Aided Design*, pp. 171V176, Nov. 2000.

[11] H.-C. Lee, Y.-W. Chang, J.-M. Hsu, and H. Yang, ”Multilevel floorplanning/ placement for large-scale modules using B*-trees”, *Proc. of Design Automation Conf.*, pp. 812V817, June 2003.

[12] J. Heisterman and T. Lengauer, “The efficient solution of integer programs for hierarchical global routing”, *IEEE Trans. on CAD*, vol. 10, no. 6, pp. 748-753, June 1991.

[13] M. Marek-Sadowska, ”Router planner for custom chip design”, *Proc. of Int. Conf. Computer-Aided Design*, Nov. 1986.

[14] D. Wang and E. Kuh, ”A new timing-driven multilayer MCM/IC routing algorithm”, in *Proc. Multi-chip Module Conference*, pp. 89-94, February 1997.

[15] Y.-W. Chang, K. Zhu, and D.-F. Wong, “Timing-driven routing for symmetrical-array- based FPGAs”, *ACM Trans. Design Automation of Electronic Systems*, vol. 5, no. 3, pp. 433-450, July 2000.

[16] J. Cong, J. Fang, and Y. Zhang, “Multilevel approach to full-chip gridless routing”, in *Proc. of Int. Conf. Computer-Aided Design*, pp. 396-403, Nov. 2001.

[17] J. Cong, J. Fang, and K. Khoo, “DUNE: a multi-layer gridless routing system with wire planning”, in *Proc. International Symposium on Physical Design*, pp. 12-18, April 2000.

[18] S.-M. Li, C.-qL. Lee, Y.-W. Chang, C. Su, J.-E Chen, “Multilevel full-chip routing with testability and yield enhancement”, in *Proc. System Level Interconnect Prediction*, pp. 29-36, April 2005.

[19] R. Hadsell and P. Madden, “Improved global routing through congestion estimation”, in *Proc. of Design Automation Conf.*, pp. 28-31, June 2003.

[20] L. Cheng, X. Song, G. Yang, Z. Tang, “A fast congestion estimation for routing with bounded detours”, in *Proc. Asia and South Pacific Design Automation Conf.*, pp. 66-70, January 2004.

[21] R. Kastner, E. Bozorgzadeh and M. Sarrafzadeh, “Predictable routing”, in *Proc. of Int. Conf. Computer-Aided Design*, pp. 110-114, November 2000.

[22] J. Westra, P. Groeneveld, “Is probabilistic congestion estimation worthwhile?” *in Proc. System Level Interconnect Prediction*, pp. 99-106, April 2005.

[23] T.-Y. Ho, Y.-W. Chang, and S.-J. Chen, “Multilevel routing with antenna avoidance”, *Proc. of Int. Symp. on Physical Design*, pp. 34V40, 2004.

[24] C.-W. Sham, E. F. Y. Young, “Congestion Prediction in Early Stages”, *in Proc. System Level Interconnect Prediction*, pp. 91-98, April 2005.

[25] W. C. Elmore, The transient response of damped linear networks with particular regard to wide band amplifiers, *Journal of Applied Physics*, vol. 19, pp 55-63, January 1948.

[26] T.-C. Chen and Y.-W. Chang, “Multilevel Full-Chip Gridless Routing Considering Optical Proximity Correction”, *Proc. Asia and South Pacific Design Automation Conf.*, pp. 29-36, January 2005.

[27] K. mehlhorn, s. Naher, *LEDA A Platform for Combinatorial and Geometric Computing*, Cambridge University Press, 1999.

[28] R. Kastner, E. Bozorgzadeh, M. Sarrafzadeh, “Pattern Routing: Use and Theory for Increasing Predictability and Avoiding Coupling” ,*IEEE Trans. on CAD*, vol. 21, NO. 7, July 2002.

[29] J. Ho, G. Vijayan, and C. K. Wong, “A new approach to the rectilinear steiner tree problem”, in *Proc. ACM/IEEE Design Automation Conf.*, June 1989.