國 立 交 通 大 學

電 信 工 程 學 系

碩 士 論 文

考慮晶片上互感及電阻、電感、電容雜訊之漏話導向電路佈局

Crosstalk-Driven Placement with Considering On-Chip Mutual Inductance and RLC Noise

研 究 生：邱震軒

指導教授：李育民　教授

中 華 民 國 九 十 四 年 十 月

考慮晶片上互感與電阻、電感、電容雜訊之漏話導向電路佈局
Crosstalk-Driven Placement with Considering On-Chip Mutual Inductance and RLC Noise

研 究 生：邱震軒　　Student：Chen-Hsuan Chiu

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

考慮晶片上互感及電阻、電感、電容雜訊之漏話導向電路佈局

學生：邱震軒　　指導教授：李育民 博士

國立交通大學電信工程學系碩士班

摘　　要

當深次微米技術演進至0.18 微米之下，雜訊效應成為電路設計者所無法忽視的一個重要問題。本論文提供一新穎的漏話導向之電路佈局演算法，用於消減晶片上因互感及電阻、電感、電容所產生之雜訊。我們將證明在佈局時僅考慮因電阻與電容所引起之雜訊，將過分樂觀化實際電路所產生之雜訊效應。實驗結果說明，我們提供之演算法相較於面積導向之佈局法，僅平均多增加8.4%的面積，但卻減低了44.1%的機率雜訊值、並縮短了30.1%的總估計線長。而相較於壅塞導向與電阻、電容雜訊導向之佈局法，我們也分別平均改善了15.9%、8.9%的機率雜訊值，以及縮短14.9%與6.8%的總估計線長。

Crosstalk-Driven Placement with Considering On-Chip Mutual Inductance and RLC Noise

Student：Chen-Hsuan Chiu　　Advisor：Dr.Yu-Min Lee

Department of Communication Engineering
National Chiao Tung University

ABSTRACT

As the deep-submicron technologies scale down to 0.18 μm, the crosstalk noise has become a critical issue which designer cannot neglect. In the thesis, a novel crosstalk-driven placement algorithm for on-chip mutual inductance and RLC noise consideration will be proposed. We also demonstrate that only take account of the RC noise during placement will be excessively optimistic in the noise effects produced by designed circuits. Results show that our approach can reduce 44.1% probabilistic RLC noise and improve 30.1% total estimated wirelength on average than the area-driven placement only at the cost of 8.4% increase of total area. For the congestion-driven and RC-driven placement, our algorithm also achieves 15.9% and 8.9% improvement on average in probabilistic RLC noise, and averagely minimizes 14.9% as well as 6.8% total estimated wirelength, respectively.

誌　　謝

在這篇論文能順利完成的當下，首先最需感謝的是我的指導教授 李育民博士，老師專業的背景總能適時點出我在研究上的思考盲點，引導我深入思索問題的根源、進而解決問題。也因老師的帶領下，讓我在碩士的二年裡，學習到做研究應有的正確態度與方法，我相信這對我的影響絕非僅止於研究所生涯，對我未來的成長更是受益無窮。

除此之外，並謝謝台灣大學 張耀文教授實驗室所提供的 B*-tree 原始碼，使得本研究能在此良好架構上繼續發揮，最後並得以順利付梓。

再者，需感謝我實驗室的夥伴 — 至鴻學長、義琅、逸宏及培育，有你們的陪伴及寶貴的意見，才能使我在做研究中更快發現自己所忽略的問題，觸發靈感來解決它。

最後我要深深感激的，是一直在旁支持我的家人及我摯愛的女友宜驊，你們總是在我情緒最低落時默默地陪著我、無悔地伴我一起走過，也因為有了你們，這一切的辛苦終於得到回報，我的喜悅也才得以分享!!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter gives basic concepts of the signal integrity problems. Since many researches have detected that the on-chip noise becomes very seriously in the deep-submicron era, we first construct the noise fundamental concepts in Chapter 1.1. Then, various of the conventional solutions for signal integrity problems are illustrated in Chapter 1.2. Finally, the motivations of this research, our contributions, and the organization of this thesis are stated in the Chapter 1.3, 1.4, and 1.5, respectively.

1.1 Noise Fundamentals

In today's VLSI (Very-Large Scale Integrated circuit) design, increase of the circuit complexity and wire congestion make coupling effects between interconnects to be more severely than before. Especially, when the technology scales down to $0.18\mu m$ [3] or the duty frequency up to GHz, the signal integrity and coupling noise have become a critical issue that designers cannot ignore anymore. Generally, the kinds of on-chip noise includes [1, 2, 4]:

- **Interconnect coupling noise:** Coupling noise, or crosstalk, is primarily due to capacitive and inductive coupled between metal wires. In Fig. 1.1, two parallel wires are modeled as the RLC model with an active voltage $V_s(t)$ and lumped capacitive loads, and the coupled noise interferes with the victim line by means of the coupling capacitance (C_x) and mutual inductance (L_x). We can see that the voltage of

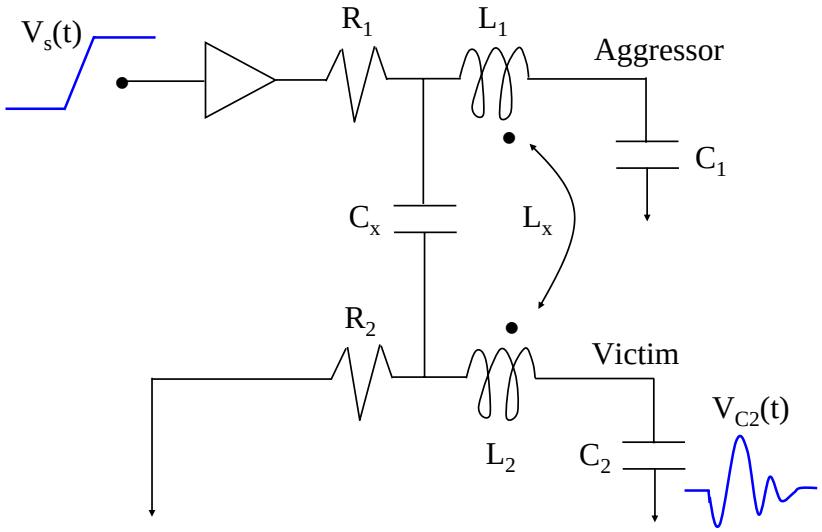


Fig. 1.1: The configuration of two coupled wires.

the victim's far-end is not still quiet but with a voltage fluctuation. Actually, when the clock rate of a circuit speeds up, the inductive coupling noise will dominate the noise effects of a circuit.

- **Supply noise:** The voltage noise on the supply is due to other wires switch nearby or the local IR-drop. IR-drop is a voltage fluctuation because of the resistance of the on-chip power delivery network. In Fig. 1.2, each nodal voltage of the power/ground(P/G) network should be ideally equal to V_{DD}. However, with considering the resistance of the interconnect, each nodal voltage of the P/G network will be less than V_{DD}. This kind of noise may malfunction circuits, and detriment the signal integrity of the designs.

- **Charge-sharing noise:** Charge injection is due to a new circuit path to a diffusion capacitance at different voltage, and it can induce a small pulse noise in the circuit to cause malfunction. Fig. 1.3 shows two NMOS M1 and M2 with their grounded diffusion capacitance, C_1, C_2, and C_3. At time 2, since M1 turns off and M2 turns

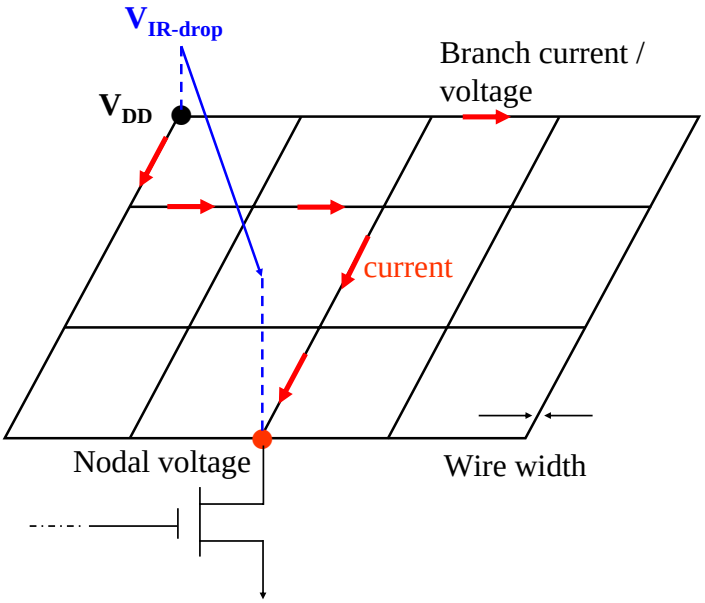


Fig. 1.2: Supply noise and IR-drop on a P/G network.

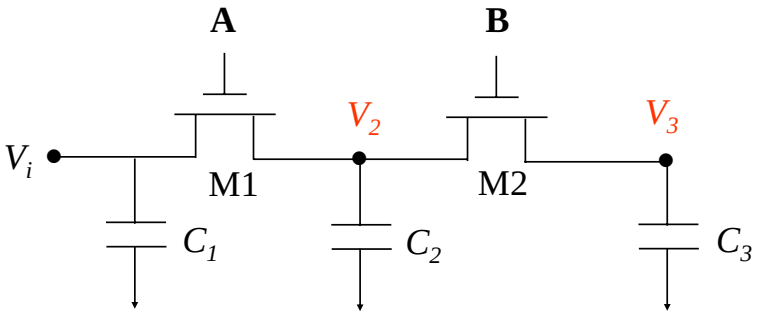
on, the voltage of V_3 makes a redistribution called *charge-sharing*. Therefore, the logic degree of V_2 and V_3 may be out of designer's anticipation.

- **Source-drain leakage:** In deep-submicron techniques, the threshold voltage of device becomes much lower than before (about 0.15V), and MOS cannot turn off completely in the cut-off region. It induces a few source-drain leakage current when devices turn off, and the noise is occured.

1.2 Conventional Solutions for Signal Integrity

Those on-chip noise introduced in Section 1.1 can injure the signal integrity of circuits seriously, even fail our design. In order to conquer those noise, several useful solutions are proposed as follows[1, 5].

- **Minimize the coupling range:** It is an instinctively solution to mitigate the noise between interconnects. Increasing the space of wires can decline the coupling capacitance and mutual inductance between wires, and the crosstalk noise can be



time	V_i	V_A	V_2	V_B	V_3
0	1	1	1	1	1
1	0	1	0	0	1
2	0	0	1/2	1	1/2

Charge sharing !!

Fig. 1.3: Charge-sharing between device M1 and M2.

minimized. Nevertheless, with considering the effects of mutual inductance, many researches [5, 6, 7, 9, 24] showed that to increase the space is not a good solution to deal with the inductive coupling noise.

- **Differential signals:** If designers know the switching patterns of the bus or signal wires, the signals tending to switch in the same direction should be interleaved by others switching in the opposite direction. If a wire has stronger inductive coupling, designers should minimize the amounts of wires switching in the same direction by interleaving a wire with its logic inversion (signal-bar). On the contrary, if the capacitive coupling noise of a wire is stronger, we should avoid inserting a wire with its logic inversion right next to it. Because to do so will be slow down the signal and increase the noise if the wire with stronger inductive coupling effects. The method can provide enough nearby current return paths for fast signals, and usually suit the clock wires.

- **Buffer insertion:** It is another effective technique to mitigate the on-chip noise and

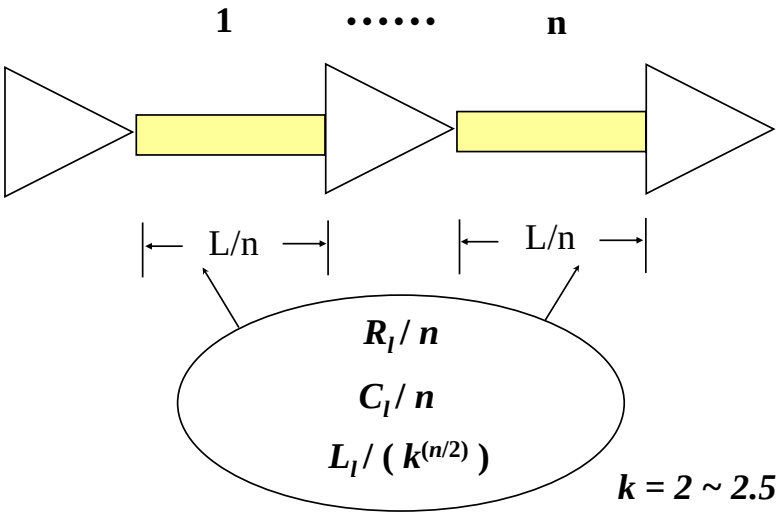


Fig. 1.4: Buffer insertion in an RLC wire for mitigation of its noise and propagation delay.

improve the timing. The original long wires can be shortened by inserting buffers to minimize the parasitics of wires. Fig. 1.4 illustrates n buffers uniformly inserted into an RLC wire. Shortening the wirelength makes the resistance, capacitance, and inductance of the wire scale down. Therefore, the RC-delay and coupling noise between parallel interconnects can be controlled.

- **Shielding wires insertion:** This is the most common and efficient solution to overcome the crosstalk noise. Adding a P/G wire (or called *shielding wire*) near a critical line canl provide a current return path for it, and make the coupling effects between wires decline quickly. However, the width of the P/G wire is much larger than signal wires, and it costs more routing resource to improve the signal integrity of circuits.

1.3 Motivation

With recent advances of deep-submicron technology, crosstalk noise has become the major problem to affect the behavior of VLSI design. Neglecting the on-chip noise will usually make our design failed. Traditional works mainly adopt post-extraction after routing, then perform noise analysis to verify the signal integrity of the design. If the result unfits the noise constraint,designers will rectify the circuit topology repeatedly. This design flow is not efficient and unsuitable for today's VLSI design.

In order to overcome the problems of signal integrity, researchers have believed that considering the coupling effects of circuits during placement stage is a more effective flow, and several crosstalk-driven placement topics [12, 14, 15] have been investigated.

[12] employed a congestion map based on the probabilistic model[13] to control the routing congestion during placement, used a quick global router by skipping the layer assignment phase to estimate routing topology, and calculated the average coupling capacitance for each wire segment. Eventually, a coupling capacitance map was generated to guide the placement.

[14] contended that the coupling capacitance map cannot completely indicate the noise behavior during placement. First, a global router was used to produce a global congestion map, then the coupling capacitance of each wire segment was extracted . Finally, these extracted capacitances were used to produce a *noise map* based on the RC model [10] to guide the placement.

[15] proposed a GA-based (Genetic Algorithm) crosstalk-driven placement that had two-level hierarchical structure, outline and detailed levels, to improve the coupling capacitance noise, RC-delay, and power consumption. In the crosstalk estimation, the coupling capacitance was determined between the aggressor and victim according to the states of signals switching.

However, all of the previous researches in crosstalk-driven placement only take account of the *RC* coupling noise, the effects of the *mutual inductance* have not been con-

	Peak noise without L_x	**Peak noise with L_x**
space 1	0.2352V	0.3276V
space 2	0.1087V	0.3109V
space 3	0.0617V	0.3007V
space 4	0.0395V	0.2954V
space 5	0.0273V	0.2920V
space 6	0.0200V	0.2897V
space 7	0.0153V	0.2879V

Table 1.1: Peak coupling noise with and without considering mutual inductance.

sidered when estimating crosstalk noise. In order to demonstrate the criticalation of the mutual inductance, we implement the RLC noise model proposed in [11] to estimate the peak coupling noise between two identical wires with $0.13\mu m$ / 1.2V technology. The wire width, thickness, length, unit of space between wires, aggressor resistance, victim resistance, and the input rising time are $0.16\mu m$, $0.28\mu m$, $3000\mu m$, $0.18\mu m$, 75Ω, 50Ω, and 100ps, respectively. Table 1.1 shows the *peak coupling noise* with and without considering the mutual inductance (L_x).

We can see that the estimated coupling noise without considering mutual inductance drops quickly, and as wires are separated more than 5 space units, the coupling noise only remains 10% of the 1 unit of space. However, with considering the mutual inductance, the coupling noise declines very slowly. This phenomenon illustrates that the mutual inductance dominates the coupling noise in deep-submicron technologies, and without considering the mutual inductance during the crosstalk-driven placement will extremely underestimate the crosstalk noise.

1.4 Our Contribution

In this thesis, a placement which is capable of considering the on-chip *RLC* noise is proposed. Our main contribution include:

- During placement, the proposed placement method introduces a novel technique that can deal with the crosstalk noise due to the on-chip mutual inductance. First, a

probabilistic model is utilized for congestion estimation. Then a transmission-line based RLC model [11] is employed to estimate the peak coupling noise between wires. With the information of congestion and peak noise, an effective algorithm to compute the worst-case average probabilistic noise is developed to guide the placement.

- Our placer utilizes the probabilistic model proposed in [13] to estimate the routing congestion during placement. If the routing demand is larger than the grid capacity, we deem that the probability of two nets (net A and net B) coexisting in this grid is not still equal to $P_A \times P_B$. Hence, we propose a set of equations to calculate the *upper bound* of the coexisting probability of two nets during estimating their RLC crosstalk noise.

1.5 Organization of this Thesis

The rest of this thesis is organized as follows. Chapter 2 introduces a probabilistic model [13] for congestion estimation, an RLC analytical model [11], and B*-tree representation [16]. Chapter 3 describes the proposed algorithm flow. In Chapter 4, we compare the experimental results of area-driven, congestion-driven, RC-driven, and RLC-driven placement. Finally, the conclusions are given in Chapter 5.

Chapter 2

Preliminaries

This chapter introduces several background knowledge that will be used in our crosstalk-driven placement. We first introduce the traditional VLSI design flow and the basic concept of placement. Next, the probabilistic model for our congestion estimation is stated. In Chapter 2.4, we will introduce a novel transmission line based RLC model for the on-chip RLC noise estimation during our placement. Finally, the B*-tree representation is illustrated in Chapter 2.5.

2.1 Traditional VLSI Design Flow

The traditional VLSI design flow shown in Fig. 2.1 [22]. First, designers synthesize their circuit by several synthesizers, Verilog or VHDL. Then the synthesized circuits should be partitioned into several blocks according to the circuit functions or achieve the minimum number of cut between blocks. In the floorplan and placement stage, each function block is placed on the *proper* location where to achieve the minimum total area, wirelength, congestion, crosstalk, or power consumption, etc.

After placement stage, the routing stage is performed. In general, this stage emphasizes the routability, wiring congestion, and timing improvement. When the routing is complete, the compaction, extraction, and circuit verification is performed to minimize the total area and verify the performance as well as signal integrity, respectively. Finally, taping out and finishing the design.

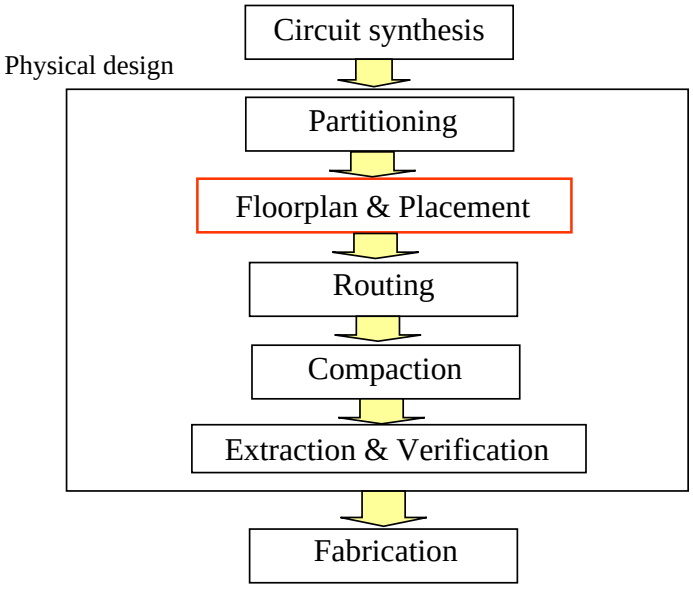


Fig. 2.1: Traditional VLSI design flow.

In addition, the design stage 2 $\sim$ 6 (partition stage $\sim$ extraction & verification stage) belong to the physical design region.

2.2 Basic Concept of Placement

In the physical design flow, placement is a crucial stage that affects the performance of the design. A good placement topology can achieve the best performance, and the minimum area, propagation delay, wirelength as well as crosstalk noise, etc. For example, Fig. 2.2 illustrates two placement solutions with different total wirelength. The placement topology shown in Fig. 2.2(a) with the longer total wirelength, but if we change its topology to Fig. 2.2(b), we can obtain the better wirelength solution.

However, to balance the above constraint, even obtain the optimum solution of placement is a NP-complete problem. In order to solve the problem and determine the proper location of each blocks, many algorithms of placement have been proposed, such as force-directed, simulated-evolution, and simulated-annealing (SA) algorithm, etc [22, 23]. Because of the popularity and usefulness of the SA algorithm, and it is also capable to obtain

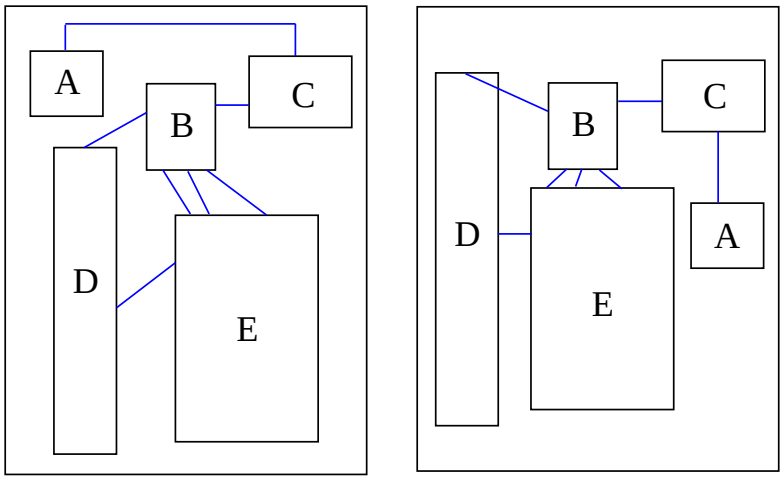


Fig. 2.2: Comparison of the total wirelength. (a) A placement with longer total wirelength. (b) A placement with shorter total wirelength.

the optimum solution of placement, we choose this algorithm for our placement.

2.3 Probabilistic Model for Congestion Estimation

Although the global router can be used to accurately estimate the wire congestion, it is too costly. Therefore, an efficient *probabilistic model* [13] is adopted to estimate the congestion. This model is correct and more efficient to help us predict the routing congestion during placement.

We first divide a design into some uniform rectangular grids proportional to its core area, then analyze the congestion in every grid. Before understanding how the model works, there are several definitions should be known as follows.

- **Definition 1:** The *capacity* of a grid is defined as the number of allowable routing tracks within a grid, and it includes the horizontal and vertical capacity shown in the following equations, respectively. Here, the number of horizontal layers is N^h, the number of vertical layers is N^v, and the minimum pitches for the ith horizontal

and vertical layer are L_i^h and L_i^v, respectively. Also, it assumes that the width and height of each grid are $Width$ and $Height$.

$$horizontal\_capacity = Height \times \sum_{i=1}^{N^h}(\frac{1}{L_i^h}) \quad (2.1)$$

$$vertical\_capacity = Width \times \sum_{i=1}^{N^v}(\frac{1}{L_i^v}) \quad (2.2)$$

- **Definition 2:** The *usage* of a grid is defined as the number of used routing tracks within a grid. Similar to definition 1, the usage of each grid also includes the horizontal and vertical one, respectively.

- **Definition 3:** If the total horizontal or vertical usage of each net i within grid(m,n) is lager than its capacity, this grid is *congested*, that is

$$\frac{\sum_{i\in(m,n)} H\_usage_i}{capacity_{H}\_of\_grid(m,n)} > 1 \quad (2.3)$$

$$\frac{\sum_{i\in(m,n)} V\_usage_i}{capacity_{V}\_of\_grid(m,n)} > 1 \quad (2.4)$$

where the H and V denote the horizontal and vertical direction, respectively.

- **Definition 4:** *F(m,n)* is the total number of possible ways to optimally route a two-pin net covering an $m \times n$ mesh that is the minimum routing region of the net.

$$F(m,n) = C_{m-1}^{m+n-2} = C_{n-1}^{m+n-2} \quad (2.5)$$

- **Definition 5:** We define 3 types of shapes for the two-pin nets employed in the model, that is *short net, flat net,* and *3rd type net*. Fig. 2.3 illustrates the net topologies of these types. Short net is a two-pin net whose source and sink are within the same grid. If a two-pin net whose source and sink are within the same row or column of grid, it is called the flat net. Otherwise, if the two-pin net covers more than 1 row and 1 column of grid, we call it the *3rd type net*.

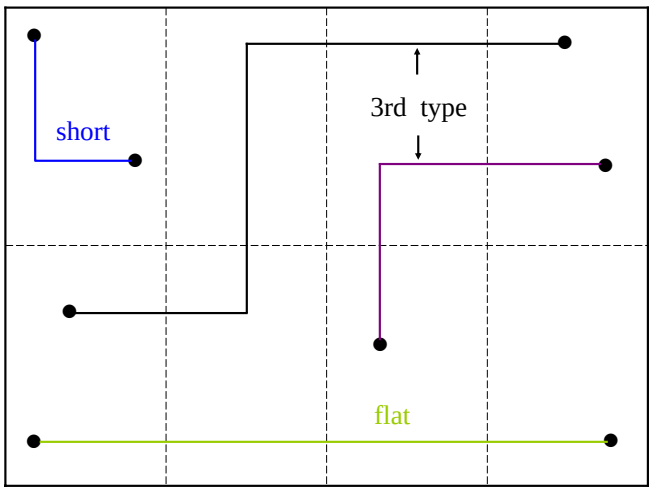


Fig. 2.3: Various shapes of the two-pin nets.

After the definition, the horizontal ($P_H(i,j)$) and vertical usage ($P_V(i,j)$) of each shape of the two-pin net within its grid can be calculated by the following equations, respectively. Further, the $Width$, $Height$, d_{x1}, d_{x2}, d_{y1}, and d_{y2} are illustrated in Fig. 2.4.

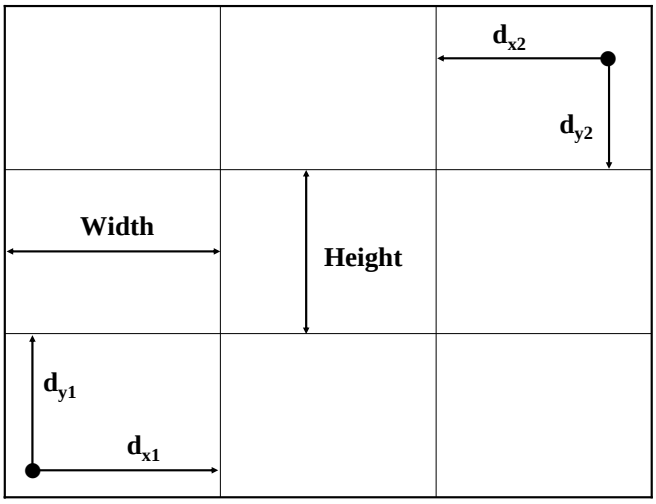


Fig. 2.4: Offgrid pins of a two-pin net.

- **Short net:**

$$P_H(i,j) = \frac{|d_{x1} + d_{x2} - Width|}{Width} \tag{2.6}$$

$$P_V(i,j) = \frac{|d_{y1} + d_{y2} - Height|}{Height} \tag{2.7}$$

- **Flat net:** If the source and sink of the net are in the same column, and they also covers m rows of grids, then its horizontal and vertical usage within grid(i,j) can be written as:

$$P_H(i,j) = \begin{cases} \frac{|d_{x1}+d_{x2}-Width|}{2\times Width}, & \text{i = 1, m} \\ 0, & otherwise \end{cases} \tag{2.8}$$

$$P_V(i,1) = \begin{cases} \frac{d_{y1}}{Height}, & \text{i = 1} \\ \frac{d_{y2}}{Height}, & \text{i = m} \\ 1, & otherwise \end{cases} \tag{2.9}$$

Similarly, if the pins of the two-pin net are in the same row, and they also covers n columns of grids, then its probabilistic usage with grid(i,j) can be computed by:

$$P_H(1,j) = \begin{cases} \frac{d_{x1}}{Width}, & \text{j = 1} \\ \frac{d_{x2}}{Width}, & \text{j = n} \\ 1, & otherwise \end{cases} \tag{2.10}$$

$$P_V(1,j) = \begin{cases} \frac{|d_{y1}+d_{y2}-Height|}{2\times Height}, & \text{j = 1, n} \\ 0, & otherwise \end{cases} \tag{2.11}$$

- **3rd type net:**

$$P_H(i,j) = \frac{1}{F(m,n)} \times \begin{cases} F(m,n-1)\times\frac{d_{x1}}{Width}, & \text{i = 1, j = 1} \\ F(m,n-1)\times\frac{d_{x2}}{Width}, & \text{i = m, j = n} \\ \frac{d_{x2}}{Width}, & \text{i = 1, j = n} \\ \frac{d_{x1}}{Width}, & \text{i = m, j = 1} \\ F(m-i+1,n-1)\times\frac{d_{x1}}{Width}, & \text{1 < i < m, j = 1} \\ F(i,n-1)\times\frac{d_{x2}}{Width}, & \text{1 < i < m, j = n} \\ \frac{F(m,n-j+1)+F(m,n-j)}{2}, & \text{i = 1, 1 < j < n} \\ \frac{F(m,j)+F(m,j-1)}{2}, & \text{i = m, 1 < j < n} \\ \frac{F(i,j)F(m-i+1,n-j)+F(i,j-1)F(m-i+1,n-j+1)}{2}, & otherwise \end{cases}$$

$$P_V(i,j) = \frac{1}{F(m,n)} \times \begin{cases} F(m-1,n) \times \frac{d_{y1}}{Height}, & \text{i = 1, j = 1} \\ F(m-1,n) \times \frac{d_{y2}}{Height}, & \text{i = m, j = n} \\ \frac{d_{y1}}{Height}, & \text{i = 1, j = n} \\ \frac{d_{y1}}{Height}, & \text{i = m, j = 1} \\ \frac{F(m-i+1,n)+F(m-i,n)}{2}, & 1 < \text{i} < \text{m, j = 1} \\ \frac{F(i,n)+F(i-1,n)}{2}, & 1 < \text{i} < \text{m, j = n} \\ F(m-1,n-j+1) \times \frac{d_{y1}}{Height}, & \text{i = 1, } 1 < \text{j} < \text{n} \\ F(m-1,j) \times \frac{d_{y2}}{Height}, & \text{i = m, } 1 < \text{j} < \text{n} \\ \frac{F(i,j)F(m-i,n-j+1)+F(i-1,j)F(m-i+1,n-j+1)}{2}, & otherwise \end{cases}$$

Finally, the probabilistic model can be implemented by the algorithm illustrated in Table 2.1:

Algorithm of Probabilistic Model	
1	**Begin**
2	Compute the capacity of each grid
3	Compute the F(m,n) matrix
4	**For** each net in the design
5	MST(net)
6	**For** each segment of the MST
7	Determine the size of mesh
8	Compute the horizontal and vertical usages within its grids
9	**EndFor**
10	**EndFor**
11	**For** each grid in the design
12	Compute the congestion of the grid
13	**EndFor**
14	**End.**

Table 2.1: Algorithm of the probabilistic model.

For the algorithm shown in Table 2.1, the preparations for the model are the stage 1∼3. For each net in the design, if it is a multi-pin net, we make a decomposition by the Minimum Spanning Tree technique (MST) [8]. Then for each two-pin net within its routing grids, we calculate its horizontal and vertical usages by the above equations. This process can be computed in constant time with precomputed F(m,n) matrix. Finally, for each grid in the design, its congestion is computed by Equation (2.4).

Assume that the number of nets in a design is n, the size of the grids is $m \times m$, and

the maximum number of pins for any net is p. The overall runtime complexity of this model is $O(np^2 + m^2np)$. Suppose if the grid size and the maximum number of pins are constants, the runtime is linear with respect to the amounts of nets in the design.

2.4 Analytical RLC Model for Noise Estimation

Given two parallel interconnects, they can be modeled as a transmission-line based model [11] illustrated in Figure 2.5, where the R, L, C, C_x, L_x are the unit resistance, inductance, capacitance, coupling capacitance, and mutual inductance of the wires, respectively. This figure shows two coupled interconnects which one line is active and the other is quiet. The active line is denoted as "aggressor", and the quiet line is the "victim". The driver of aggressor is modeled as a ramp voltage V_s with an equivalent resistance R_s. The driver of victim is represented as an equivalent resistance R_v connected to ground. The sink at the far-end of each wire is modeled as a lumped capacitive load.

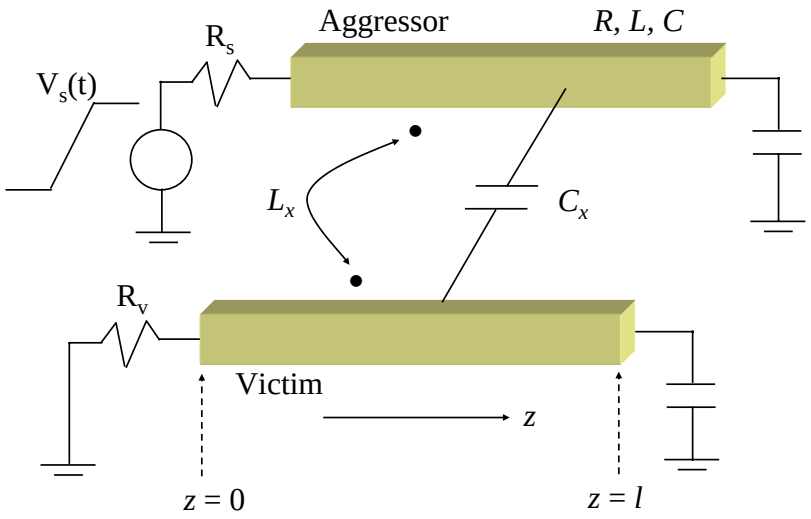


Fig. 2.5: The configuration of two coupled wires.

For two *non-identical* wires, that means, they have different line parasitics. The unit line parasitics for the aggressor are R_1, L_1, and C_1 and those for the victim are R_2, L_2,

and C_2. At any point z along the wire, the voltage and current waveforms on the aggressor (line 1) and victim (line 2) satisfy the following set of differential equations:

$$\begin{aligned} -\frac{\partial V_1}{\partial z} &= (R_1 + sL_1)I_1 + sL_xI_2 \\ -\frac{\partial V_2}{\partial z} &= (R_2 + sL_2)I_2 + sL_xI_1 \\ -\frac{\partial I_1}{\partial z} &= (C_1 + C_x)V_1 + sC_xV_2 \\ -\frac{\partial I_2}{\partial z} &= (C_2 + C_x)V_2 + sC_xV_1 \end{aligned} \tag{2.12}$$

Due to the far-end reflection coefficient is around +1 [24], the generic solution for the above set of differential equations is given by:

$$\begin{aligned} V_1 &= A_1(e^{-\gamma_e z} + e^{\gamma_e z}) + A_3(e^{-\gamma_o z} + e^{\gamma_o z}) \\ V_2 &= A_2(e^{-\gamma_e z} + e^{\gamma_e z}) + A_4(e^{-\gamma_o z} + e^{\gamma_o z}) \\ I_1 &= \frac{A_1}{Z_{0e1}}(e^{-\gamma_e z} - e^{\gamma_e z}) + \frac{A_3}{Z_{0o1}}(e^{-\gamma_o z} - e^{\gamma_o z}) \\ I_2 &= \frac{A_2}{Z_{0e2}}(e^{-\gamma_e z} - e^{\gamma_e z}) + \frac{A_4}{Z_{0o2}}(e^{-\gamma_o z} - e^{\gamma_o z}) \end{aligned} \tag{2.13}$$

For simplicity, we first consider the case of *lossless* lines, that is, the wire resistance $R_1 = R_2 = 0$. The even and odd mode propagation constant γ_e and γ_o are

$$\begin{aligned} \gamma_e &= s\sqrt{\frac{(a_1 + a_2) + \sqrt{(a_1 - a_2)^2 + 4b_1b_2}}{2}} \\ \gamma_o &= s\sqrt{\frac{(a_1 + a_2) - \sqrt{(a_1 - a_2)^2 + 4b_1b_2}}{2}} \end{aligned} \tag{2.14}$$

where

$$\begin{aligned} a_1 &= L_1(C_1 + C_x) - L_xC_x \\ a_2 &= L_1(C_2 + C_x) - L_xC_x \\ b_1 &= -L_1C_x + L_x(C_2 + C_x) \\ b_1 &= -L_2C_x + L_x(C_1 + C_x) \end{aligned} \tag{2.15}$$

In the solution of Equation (2.13), the coefficients are related as:

$$\begin{aligned} \frac{A_1}{A_2} &= \frac{(a_1-a_2)+\sqrt{(a_1-a_2)^2+4b_1b_2}}{2b_2} \\ \frac{A_3}{A_4} &= \frac{(a_1-a_2)-\sqrt{(a_1-a_2)^2+4b_1b_2}}{2b_2} \end{aligned} \tag{2.16}$$

The characteristic impedences of the aggressor and victim line can be written as:

$$\begin{aligned} Z_{0e1} &= \frac{s(L_1L_2-L_x^2)}{\gamma_e(L_2-\frac{A_2}{A_1}L_x)} \\ Z_{0e2} &= \frac{s(L_1L_2-L_x^2)}{\gamma_e(L_2-\frac{A_1}{A_2}L_x)} \\ Z_{0o1} &= \frac{s(L_1L_2-L_x^2)}{\gamma_o(L_2-\frac{A_4}{A_3}L_x)} \\ Z_{0o2} &= \frac{s(L_1L_2-L_x^2)}{\gamma_o(L_2-\frac{A_3}{A_4}L_x)} \end{aligned} \tag{2.17}$$

Then the boundary conditions are given by

$$\begin{aligned} \frac{V_s-V_1(z=0)}{I_1(z=0)} &= \frac{V_s-(A_2+A_4)}{\frac{A_2}{Z_{0e2}}+\frac{A_3}{Z_{0o2}}} = R_s \\ \frac{-V_2(z=0)}{I_2(z=0)} &= \frac{-(A_2-A_4)}{\frac{A_2}{Z_{0e2}}-\frac{A_3}{Z_{0o2}}} = R_v \end{aligned} \tag{2.18}$$

Applying the boundary conditions to solve Equation (2.13), we can obtain the voltage steps traveling on the victim line:

$$\begin{aligned} A_2 &= \frac{V_s(t)}{(\frac{A_1}{A_2})(\frac{Z_{0e1}+R_s}{Z_{0e1}})-(\frac{A_3}{A_4})(\frac{Z_{0e2}+R_v}{Z_{0o2}+R_v})(\frac{Z_{0o2}}{Z_{0e2}})(\frac{Z_{0o1}+R_s}{Z_{0o1}})} \\ A_4 &= \frac{V_s(t)}{(\frac{A_3}{A_4})(\frac{Z_{0o1}+R_s}{Z_{0o1}})-(\frac{A_1}{A_2})(\frac{Z_{0o2}+R_v}{Z_{0e2}+R_v})(\frac{Z_{0e2}}{Z_{0o2}})(\frac{Z_{0e1}+R_s}{Z_{0e1}})} \end{aligned} \tag{2.19}$$

For line length l, the step propagating with the even mode constant arrives at the far-end after an even time of flight t_{fe}, and the step propagating with the odd mode constant arrives at the far-end after an odd time of flight t_{fo}.

$$\begin{aligned} t_{fe} &= l\sqrt{\frac{(a_1+a_2)+\sqrt{(a_1-a_2)^2+4b_1b_2}}{2}} \\ t_{fo} &= l\sqrt{\frac{(a_1+a_2)-\sqrt{(a_1-a_2)^2+4b_1b_2}}{2}} \end{aligned} \tag{2.20}$$

The coupling noise in the victim's far-end is composed of two ramp waves, $2A_2(t - t_{fe})$ and $2A_4(t - t_{fo})$, which is illustrated in Fig. 2.6.

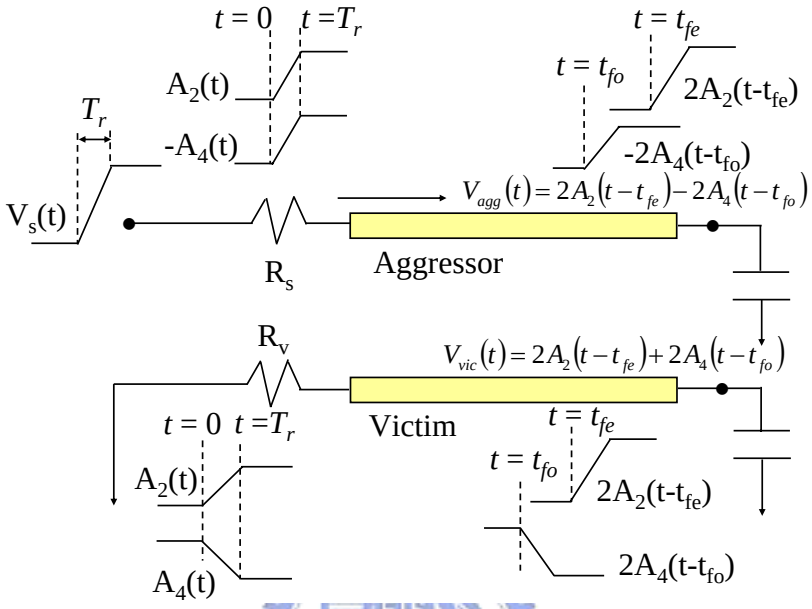


Fig. 2.6: Reflection waves in the two coupled transmission line.

This figure shows the *fist* reflection behavior of the wires. Based on the above equations, the far-end waveforms of the aggressor and victim can be computed by the following steps:

- Given an input ramp $V_s(t)$, the even and odd mode ramps, $A_2(t)$ and $A_4(t)$, which can be calculated by Equation (2.19), respectively.
- The ramp $A_2(t)$ reaches the far-end with a time of flight delay t_{fe}, and $A_4(t)$ reaches the far-end with a time of flight delay t_{fo}.
- Due to the reflection coefficient of the far-end is +1, the ramp voltages are double to the far-end of the aggressor and victim line, respectively.
- Superposition of the even and odd mode ramps occuring in the far-end of line to obtain the waveform for the aggressor and victim line.

- Reverse traveling waves will be reflected at the near-end and add to the far-end waveforms after three time of flight delays.

Performing the above steps in Fig. 2.6, it shows that ramp voltages $A_2(t)$ and $A_4(t)$ are produced in the near-end of lines. These ramps travel with different velocities and reach the far-end after different time delays. After performing the superposition, the output waveforms then be computed by:

$$\begin{aligned} V_{agg}(t) &= 2A_2(t - t_{fe}) - 2A_4(t - t_{fo}) \\ V_{vic}(t) &= 2A_2(t - t_{fe}) + 2A_4(t - t_{fo}) \end{aligned} \tag{2.21}$$

where $V_{agg}(t)$ and $V_{vic}(t)$ are the waveforms at the output of the aggressor and victim, respectively.

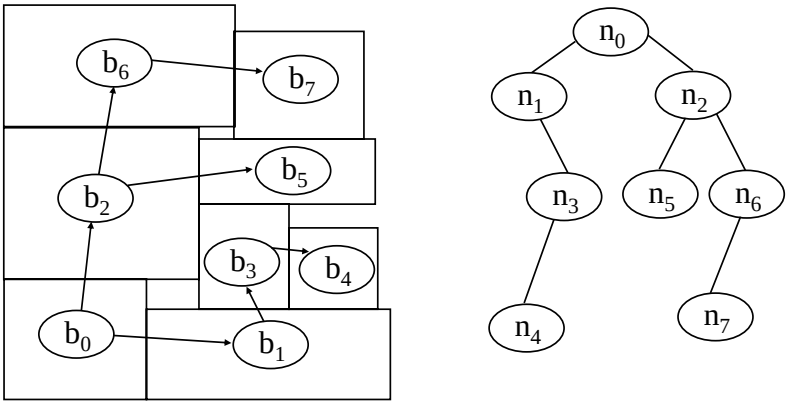
Furthermore, for a *lossy* transmission line, that means, the wire resistance is not equal to zero. The *peak coupling noise* of the far-end of a wire can be calculated by Equation (2.22). Here, R, V^+, V^- are the unit wire resistance, positive and the negative peak, respectively.

$$\begin{aligned} V^+_{loosy} &= V^-_{lossless} \times e^{-\frac{R}{2Z_{0o}}} + (V^+_{lossless} - V^-_{lossless}) \times e^{-\frac{R}{2Z_{0e}}} \\ V^-_{loosy} &= V^-_{lossless} \times e^{-\frac{R}{2Z_{0o}}} \end{aligned} \tag{2.22}$$

2.5 B*-Tree for Placement Representation

In this thesis, B*-tree [16] is used for our placement representation. A B*-tree is an ordered tree for modeling a slicing or a non-slicing placement. Given an admissible placement [17] (that means, no blocks can be moved left or down), an unique B*-tree can be constructed in linear time to model the placement.

Fig. 2.7 indicates an admissible placement and its corresponding B*-tree. The root of the B*-tree corresponds to the block on the bottom-left corner. Similar to DFS (Deep-First Search), it constructs a B*-tree T for an admissible placement in a recursive procedure:



(a) (b)

Fig. 2.7: (a) An admissible placement. (b) The corresponding B*-tree representation for the placement.

Beginning from the root, then it recursively construct the left subtree and then the right one.

For example, in order to construct a corresponding B*-tree shown in Fig. 2.7(a), we first pick n_0, the root of T, and place b_0 on the bottom-left corner. Then traversing the left child of n_0, which is n_1. b_1 is placed on the right of its parent, b_0. Because n_1 does not any left children, then the next one for chosen is the n_3, and place it on the top of b_1. Recursively repeating the process in the DFS procedure, then the corresponding admissible placement can be obtained.

B*-tree is an efficient representation for placement. It achieves a smaller area due to the admissible placement structure. Moreover, it improves the run time complexity more than the O-tree representation [17]. That is why we adopt B*-tree representation for our placement.

Chapter 3

Crosstalk-Driven Placement

In this chapter, we will introduce the algorithm flow of our proposed placement. We first state the problem formulation of the research, then point out the main difference between ours and the general placement. In Chapter 3.2, a novel technique for on-chip RLC noise estimation is proposed, we utilize the algorithm to estimate the probabilistic noise during placement. Then we also develop a set of equations to compute the *upper bound of the coexisting probability* between two nets within a routing grid if it is likely overflowing. Finally, the whole algorithm of our crosstalk-driven placement is integrated in Chapter 3.5.

3.1 Problem Formulation

The problem formulation of the crosstalk-driven placement can be formulated as follows.

- ***Input:*** *Given a fixed chip boundary area A, a set of blocks $B = \{b_1, b_2, \cdots, b_n\}$, their pins' locations $P = \{p_1^1, p_1^2, \cdots, p_2^1, p_2^2, \cdots, p_n^1, p_n^2, \cdots, p_n^n\}$, and a netlist N to represent the connection relations of each pin in P.*

- ***Object:*** *Determining an optimal location for each b_i in A, where $b_i \in B$, make the average probabilistic RLC noise between each net in N is the minimum.*

- ***Output:*** *Output the locations of each b_i ($b_i \in B$) and p_i ($p_i \in P$). Then it also ouputs the results of the area, estimated total wirelength, congestion, and proba-*

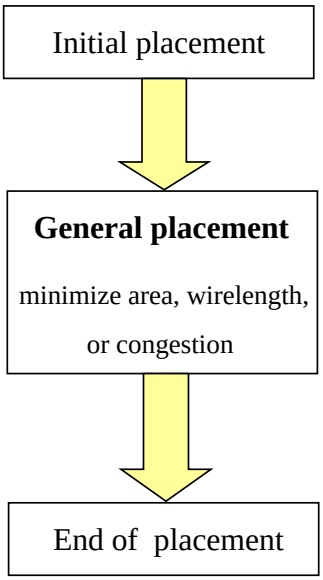


Fig. 3.1: Algorithm flowchart of the general placement.

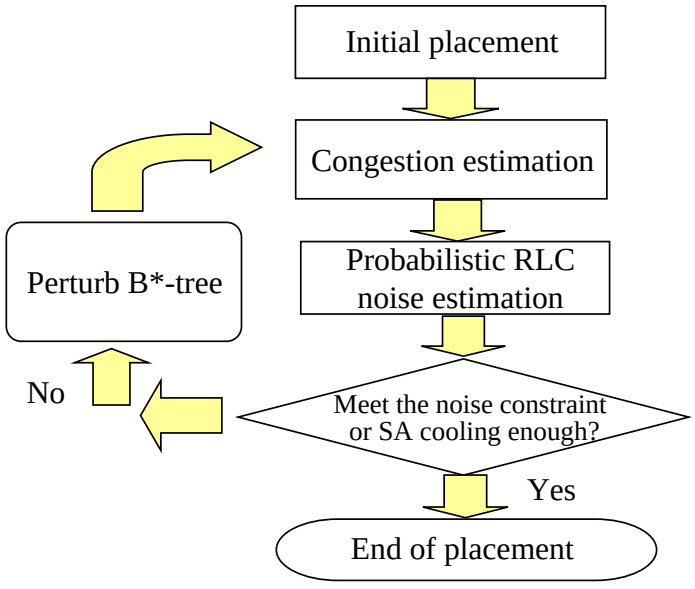


Fig. 3.2: Algorithm flowchart of our crosstalk-driven placement.

bilistic RLC noise, respectively.

Fig. 3.1 illustrates the algorithm flow of the general placement. In the general place-

ment, it minimizes the area, total wirelength or the congestion by ordinary, and does not take account of crosstalk noise issues. The general placement flow merely can improve the area, timing or the routability. However, the procedure is not enough for today's deep-submicron techniques. Without considering the on-chip noise during placement can make the circuit be malfunctioned.

The proposed crosstalk-driven placement is based on *Simulated-Annealing* (SA) algorithm, and B*-tree [16] representation. The algorithm flow is shown in Fig. 3.2. At the initial placement stage, we utilize *linear ordering* technique [18] to obtain a better initial solution for the placement.

After initial placement, the congestion estimation is proceeded. Since the probabilistic model [13] is a two-pin net based structure, each multi-pin net is firstly decomposed into several two-pin nets by the Minimum Spanning Tree (MST) technique. Then for each two-pin net in its shortest path routing region, its horizontally and vertically probabilistic usages are computed. The detail congestion estimation procedure can be referenced to the Chapter 2.3. The probabilistic information is useful to control the overall routing density as well as estimate the probabilistic noise for each two-pin net.

While the congestion estimation stage is completed, we perform the probabilistic RLC noise estimation for each two-pin net. The statement of this stage will be illustrated in the next sub-section. After performing the probabilistic RLC noise estimation for each two-pin net, if the overall average probabilistic noise meets the noise constraint or the temperature of SA is cool enough, the placement flow is finished. Otherwise, the placer will iteratively perturb B*-tree to seek a better solution.

3.2 Probabilistic RLC Noise Estimation

When the congestion estimation stage of our placement is complete, it carries on executing the probabilistic RLC noise estimation. In this section, the contents of the probabilistic noise estimation and how it works are stated. We will interpret the detail work of

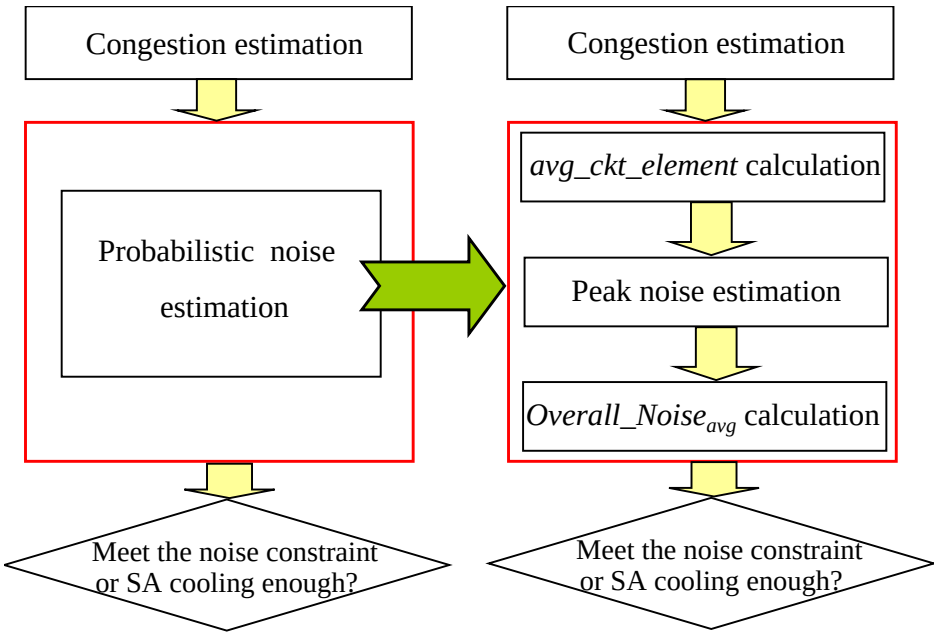


Fig. 3.3: Procedure flow of probabilistic noise estimation. (a) Summarized flow. (b) Detailed flow.

each stage in the probabilistic noise estimation in Chapter 3.2.1 $\sim$ 3.2.4, and integrate the procedures of the probabilistic noise estimation in Chapter 3.2.4.

3.2.1 Procedure Flow of Probabilistic Noise Estimation

After the congestion estimation stage, the probabilistic usages of each two-pin net within its routing grids are obtained, we then go to the next stage: probabilistic noise estimation stage. Fig. 3.3 illustrates the summarized and detailed flow of the probabilistic noise estimation. We can see that the probabilistic stage of our crosstalk-driven placement can be subdivided into the *avg_ckt_element calculation*, *peak noise estimation*, and the $Overall\_Noise_{avg}$ *calculation* stage.

In the *avg_ckt_element calculation* stage of the probabilistic estimation, the unit R, L, C, C_x, L_x of each two-pin net is calculated for the later peak noise estimation. In the peak noise stage, the peak RLC coupling noise is computed by a novel transmission line

based RLC model [11]. We propose an effective approach to pick out the maximum lossy coupling noise from several turning points instead of exhaustively searching. Eventually, the overall average probabilistic noise is computed at the last stage of the probabilistic noise estimation, and our proposed algorithm is integrated.

3.2.2 Average Circuit Elements Calculation

Since we estimate the net topology without performing layer assignment, the unit *average circuit element* for each two-pin net n is first calculated for the later RLC peak noise estimation. Equation (3.1) presents the calculation of unit average circuit element for two-pin net n. Here, the $ckt\_element(n)_i$ denotes the unit R, L, C, C_x, L_x of the two-pin net n in layer i, respectively, and they can be computed by [19, 20]. Further, w_i is the weight for each layer. It models the probability that two-pin net n will go through the layer i, and we can obtain the values of w_i by a trial route or derive them empirically. In general, we set $w_i = 1/(\text{total number of layers})$.

$$avg\_ckt\_element(n) = \sum_{\forall layers} w_i \times ckt\_element(n)_i \tag{3.1}$$

Assume that the length of each pair of coupled wire is equal. The unit R, L, C, C_x, L_x of each two-pin net can be computed by the following equations.

$$\begin{aligned} \frac{R}{l} &= \frac{(3.3\mu\Omega - cm)}{WT} \\ \frac{L_{ii}}{l} &= 0.002[ln(\frac{2l}{W+T}) + 0.5 - 0.2235 \times \frac{(W+T)}{l}] \\ \frac{L_x}{l} &= 0.002[ln(\frac{l}{d} + \sqrt{1+\frac{l^2}{d^2}}) - \sqrt{1+\frac{d^2}{l^2}} + \frac{d}{l}] \end{aligned} \tag{3.2}$$

where the l, W, T denote the wirelength (μm), width (μm), and thickness (μm) of two-pin net i, respectively. Also, d is the distance (μm) between the victim and aggressor. Further, the unit C and C_x between two nets in the *middle layers* can be written as:

$$\begin{aligned} \frac{C}{\epsilon_{ox}} &= (\frac{W}{H_1} + \frac{W}{H_2}) + 2.04(\frac{T}{T+4.5311H_1})^{0.071}(\frac{d}{d+0.5355H_1})^{1.773} \\ &+ 2.04(\frac{T}{T+4.5311H_2})^{0.071}(\frac{d}{d+0.5355H_2})^{1.773} \end{aligned}$$

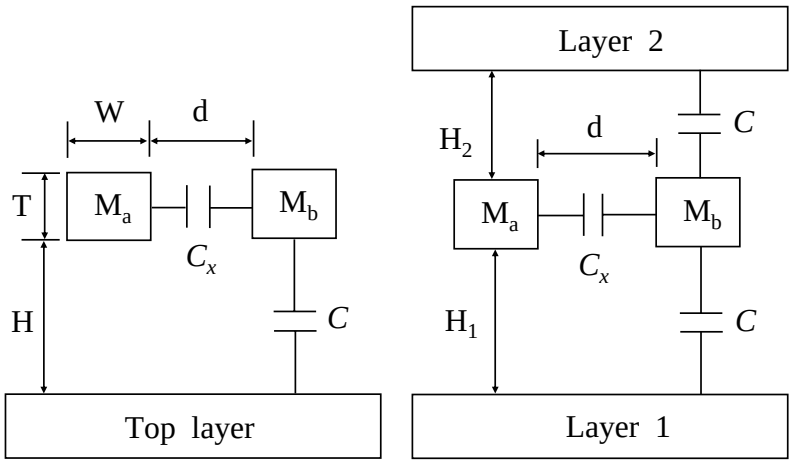


Fig. 3.4: Cross-view of parallel wires. (a) Cross-section diagram of parallel wires on the top layer. (b) Cross-section diagram of parallel wires in the middle layers

$$\begin{aligned}\frac{C_x}{\epsilon_{ox}} &= 1.4116\frac{T}{d}exp(-\frac{2d}{d+8.014H_1}-\frac{2d}{d+8.014H_12}) \\ &+ 1.1852(\frac{W}{W+0.3078d})^{0.25724} \\ &\cdot\{ (\frac{H_1}{H_1+8.961d})^{0.7571}+(\frac{H_2}{H_2+8.961d})^{0.7571}\} \\ &\times exp(-\frac{2d}{d+3(H_1+H_2)}) \end{aligned} \tag{3.3}$$

where $\epsilon_{ox} = 3.9 \times 8.85 \times 10^{-14}$ F/cm, and the configurations of H, H_1, H_2 are indicated in Fig. 3.4.

Similarly, for the case of *top layer* shown in Fig. 3.4(a), its unit C and C_x can be calculated as following.

$$\begin{aligned}\frac{C}{\epsilon_{ox}} &= \frac{W}{H}+2.217(\frac{d}{d+0.702H})^{3.193}+1.171(\frac{d}{d+1.51H})^{0.7642} \\ &\cdot (\frac{T}{T+4.532H})^{0.1204}\end{aligned}$$

$$\frac{C_x}{\epsilon_{ox}} = 1.144\frac{T}{d}(\frac{H}{H+2.059d})^{0.0944} + 0.7428(\frac{W}{W+1.592d})^{1.144} + 1.158(\frac{W}{W+1.874d})^{0.1612} \cdot (\frac{H}{H+0.9801d})^{1.179} \quad (3.4)$$

3.2.3 Peak Noise Estimation

After the $avg\_ckt\_element$ of every two-pin net is determined, its peak RLC noise estimation is beginning. Assume that there are h nets and t available routing tracks in a grid. For a victim net A and aggressor net B shown in Fig. 3.5, since the real routing topology and the length of a net are not known before routing, *we assume that each aggressor's length is equal to the victim's*, and that is the worst case for a victim in its minimum routing region (called mesh(A)). Then the probabilistic noise between net A and B in grid(m,n) can be written as

$$Noise_{AB}(m,n) = P^L_{AB}(m,n) \times \sum_{s=1}^{t-1} peak\_noise_{AB}(s) \times P_{AB}(s) \quad (3.5)$$

$$P_{AB}(s) = \frac{C_1^{t-s}}{C_2^t} \quad (3.6)$$

where s is the unit of space between net A and B, and $P_{AB}(s)$ represents the probability of the separated space being s space units between net A and net B. For example, if there are 10 available routing tracks in a grid where net A and B may pass through. Then the probability of 1 routing track separating of net A and B is equal to 0.2 that can be computed by Equation (3.6). Table 3.1 enumerates the separating probability between two nets from 1 to 9 tracks, when there are 10 available routing tracks in a grid. If there are not any shielding wires in grid(m,n), we have to consider the crosstalk noise ranging over the space of $t-1$.

$P^L_{AB}(m,n)$ is the *legal probability* of net A and B on grid(m,n). If the grid is not overflowed, the legal probability of net A and B on grid(m,n) is equal to the probability of net A going through grid(m,n) times the probability of net B going through grid(m,n), that

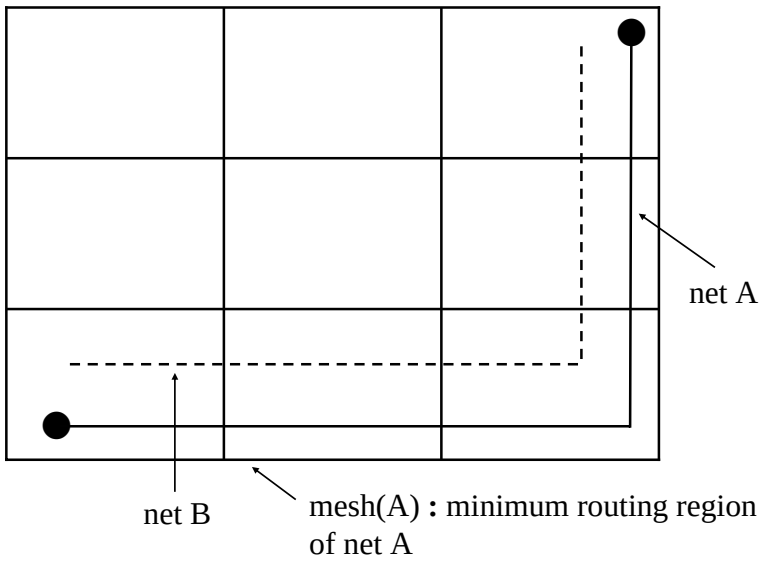


Fig. 3.5: The probabilistic noise estimation between victim net A and aggressor net B.

Separating tracks	Separating probability
1	0.2000
2	0.1778
3	0.1556
4	0.1333
5	0.1111
6	0.0889
7	0.0667
8	0.0444
9	0.0222

Table 3.1: Separating probability of two nets in a grid with 10 available routing tracks.

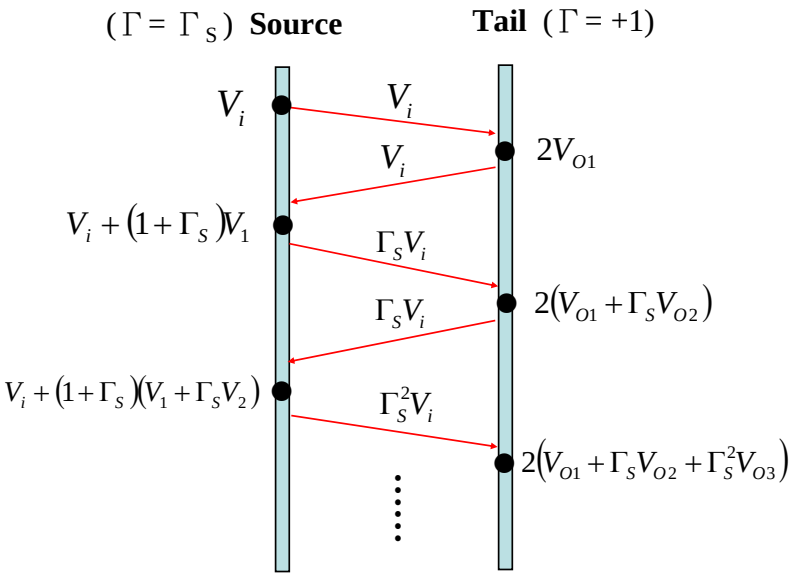


Fig. 3.6: Reflection behavior in the near-end (source) and far-end (tail) of the victim wire.

is, $P_A(m,n) \times P_B(m,n)$. On the other hand, the legal probability of net A and B should be re-computed by subtracting the redundant probabilistic usages due to the overflow. However, finding the exact legal probability is exhaustive, it suggests us to calculate its *upper bound of the coexisting probability* which is stated in Chapter 3.3.

An other important term in Equation (3.5) is the $peak\_noise_{AB}(s)$, which denotes the RLC peak noise between the victim (net A) and aggressor (net B) with separated by s unit space, and can be calculated by Equation (2.22) which is stated in Chapter 2.4. Fig. 2.6 exhibits the *first* reflection behavior in a transmission line. Actually, the reflective waves in transmission lines are reflected *repeatedly*. Fig. 3.6 illustrates the whole reflection behavior in the near-end and far-end of a victim wire.

The reflective wave of the far-end in a transmission line is [24]

$$\begin{aligned} V_{far-end} &= V_{original} + V_{inc} + V_{refc} \\ &= V_{original} + V_{inc} + \Gamma V_{inc} \end{aligned} \tag{3.7}$$

where $V_{original}$, V_{inc}, and V_{refc} indicate the original wave, incident wave, and the reflective

Index	Corresponding ramp waves
V_i	$A_2(t) + A_4(t)$
V_1	$A_2(t - 2t_{fe}) + A_4(t - 2t_{fo})$
V_2	$A_2(t - 4t_{fe}) + A_4(t - 4t_{fo})$
V_{O1}	$A_2(t - t_{fe}) + A_4(t - t_{fo})$
V_{O2}	$A_2(t - 3t_{fe}) + A_4(t - 3t_{fo})$
V_{O3}	$A_2(t - 5t_{fe}) + A_4(t - 5t_{fo})$

Table 3.2: Ramp waves in the victim wire shown in Fig. 3.6.

wave of the terminations in a transmission line, respectively. Also, Γ denotes the reflection coefficient of the wire ($-1 \leq \Gamma \leq 1$).

Table 3.2 shows each ramp wave exhibiting in Fig. 3.6. From Fig. 3.6, the original incident wave V_i reflects repeatedly between the near-end and far-end of the victim wire. Since the reflection coefficient of the victim's far-end is equal to +1, the ramp waves shown in Fig. 3.6 can be determined by using Equation (3.7). In view of the wave velocities of the even mode and odd model reflective wave are different, the reflection coefficient of the victim's near-end should be divided into the even and odd one, respectively. Therefore, the lossless coupling noise of the victim's far-end can be written as

$$V_{vic}(t) = 2\sum_{k=0}^{\infty}\{\Gamma_{even}^{k}\, A_2(t - (2k+1)t_{fe}) + \Gamma_{odd}^{k}\, A_4(t - (2k+1)t_{fo})\} \tag{3.8}$$

$$\begin{aligned} \Gamma_{odd} &= \frac{R_v - Z_{0o2}}{R_v + Z_{0o2}} \\ \Gamma_{even} &= \frac{R_v - Z_{0e2}}{R_v + Z_{0e2}} \end{aligned} \tag{3.9}$$

where Γ_{odd} and Γ_{even} are the victim's near-end odd mode and even mode reflection coefficient, respectively.

In order to obtain the waveform in the victim's far-end, we sum up the turning points of each reflective waves. Each ramp wave with 2 turning points illustrated in Fig. 3.7, where

$$\begin{aligned} t_0 &= t_{fo} \\ t_1 &= t_{fe} \end{aligned}$$

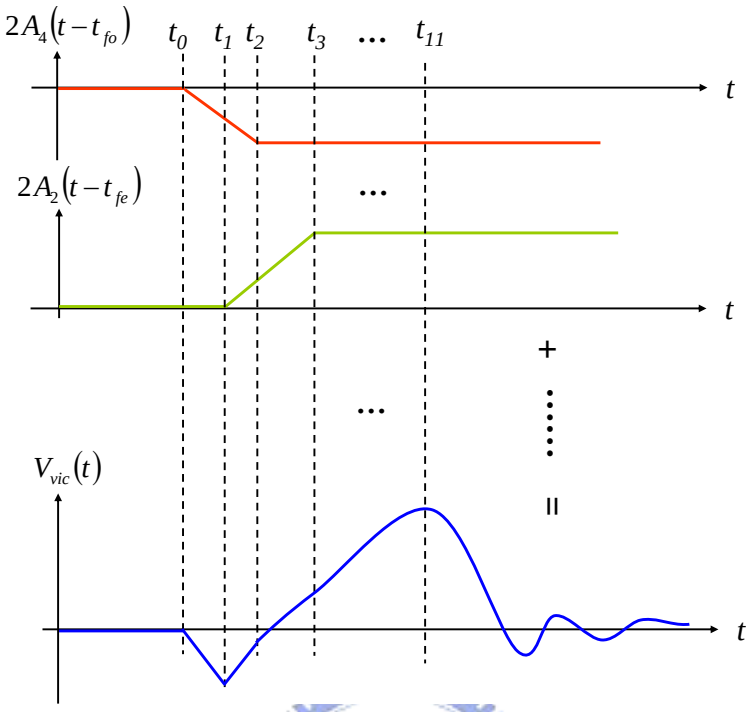


Fig. 3.7: Sum up turning points of each reflective wave to pick out the peak noise.

$$\begin{aligned} t_2 &= t_{fo} + T_r \\ t_3 &= t_{fe} + T_r \\ t_{11} &= 5t_{fe} + T_r \end{aligned} \tag{3.10}$$

Here, T_r denotes the transition time of the input signal. Be noticeable, the above figure shows the worst-case to find the peak noise, that means, the peak occurs in $t = t_{11}$. In general, most peak noise of a net occurs within t_1 to t_{11}.

Interestingly, we discover that the peak noise of the all nets always occur within *200ps* in our experiment environment (0.13μm / 1.2V, $T_r = 100ps$, and two metal layers for routing). Generally, it should sample 11 turning points to determine the peak value within the noise window, that is

$$\begin{aligned} t &= \{nt_{fo} \mid n = 3, 5\} \\ t &= \{nt_{fe} \mid n = 1, 3, 5\} \end{aligned}$$

$$
\begin{aligned}
t &= \{nt_{fo} + T_r \mid n = 1, 3, 5\} \\
t &= \{nt_{fe} + T_r \mid n = 1, 3, 5\}
\end{aligned} \tag{3.11}
$$

The sampling points shown in Equation (3.11) that neglect $t = t_{fo}$, since this node is a starting point of $V_{vic}(t)$, and always be equal to $0V$. However, sampling 11 points to pick out the peak noise in the victim's far-end is too costly and will degrade the speed of placement. For the specific wirelength of our experiment environment, we discover that sampling less than 11 points is enough to determine the peak noise. For example, if the length of a net is longer than 4000μm, only sampling the first 3 turning points ($t_{fo} + T_r$, t_{fe}, and $t_{fe} + Tr$) can pick out the peak noise. However, if the experiment environment or the design technology is changed, we should construct a table to record the relationship between the wirelength and the number of points for sampling, then obtain correct peak values during RLC estimation procedure.

Developing the length property during placement can extremely reduce the computation complexity of probabilistic noise estimation, and speed up the overall placement procedure. Generally, for a shorter net, whose length is shorter than 4000 μm, it must sample more turning points to pick out the peak noise. This is due to the shorter wirelength, the smaller t_{fe} and t_{fo}, and there are more ramps will be occured within the noise window. Therefore, we should sample more turning points for a shorter net more than the longer ones.

While the lossless peak noise of a net is determined by Equation (3.8), it can be transferred to the *lossy peak noise* by Equation (2.22) stated in Chapter 2.4, and the total probabilistic noise of net A in grid(m,n) can be written as:

$$
Noise_A(m, n) = \sum_{K \in \Omega(m,n)} Noise_{AK}(m, n) \tag{3.12}
$$

where $\Omega(m, n)$ is a set that nets may pass through grid(m,n). Finally, the average probabilistic noise of net A in its mesh can be computed as following:

$$
Avg\_Noise_A = \sum_{(m,n) \in mesh(A)} \frac{Noise_A(m, n)}{\text{\# grids in mesh(A)}} \tag{3.13}
$$

3.2.4 Overall Average Probabilistic Noise Estimation

When the average probabilistic noise of each two-pin net is acquired, the next stage is to compute the overall average probabilistic noise of a placement, and it can be calculated by the following equation:

$$Overall\_Noise_{avg} = \frac{\sum_{\forall two-pin\ nets} Avg\_Noise_K}{\text{\# two-pin nets}} \tag{3.14}$$

Table 3.3 exhibits the whole algorithm of probabilistic RLC noise estimation. For each two-pin net i, its average circuit element is computed by the referenced equations stated in Chapter 3.2.2, and we initialize the sum of probabilistic noise for net i to be zero. Next, for each two-pin net j ($j \neq i$) may pass through the grids of mesh(i), its peak noise interferences with net i is computed by using Equation (3.5). After all of the probabilistic noise within mesh(i) is calculated, we estimate the Avg_Noise_i by Equation (3.13). At last, when the probabilistic noise of each two-pin net is obtained, the $Overall\_Noise_{avg}$ is computed by means of Equation (3.14), and the stage of the probabilistic noise estimation is finished.

In practice, when the length of a two-pin net is too short, that means, $\frac{l}{w} < 10$ [21], the Grover formulae used for our on-chip inductance estimation will make a large error. Therefore, we let the probabilistic RLC noise of the victim be equal to *zero* if the length of the aggressor or victim is shorter than $5\mu m$. This assumption is reasonable due to if the length of a net is too short, it will suffer tiny coupling noise and interfere other nets restrictedly when it is an aggressor. Further, the probabilistic usages of a two-pin net in its routing grids that divides into the horizontal and vertical usages, its coupling noise for the horizontal and vertical direction can be computed by Equation (3.13), respectively.

3.3 Upper Bound of the Coexisting Probability

Assume that there are h nets, t routing tracks in grid(m,n), and $h > t$. Since the arithmetic mean of a set of values is larger or equal to its geometric mean. Utilize the property, the

Algorithm of Probabilistic RLC Noise Estimation	
Input: *Probabilistic usages of each two-pin net in its grids*	
Output: *Average probabilistic RLC noise of each two-pin net*	
1	**Begin**
2	**For** each net i in the design
3	Compute $avg\_ckt\_element(i)$
4	set $Noise_i = 0$
5	**For** each grid(m,n) within mesh(i)
6	**For** each net j within grid(m,n) /*$j \neq i$*/
7	**If** $length_i$ or $length_j \leq 5\mu m$ **Then** $Noise_{ij}(m,n) \leftarrow 0$
8	$Noise_i += Noise_{ij}(m,n)$
9	**EndFor**
10	**EndFor**
11	$Avg\_Noise_i = Noise_i/$(# of grids within mesh(i))
12	**EndFor**
13	Compute the $Overall\_Noise_{avg}$
14	**End.**

Table 3.3: Algorithm of probabilistic RLC noise estimation.

lower bound probability of the total illegal terms of net A and B, $P^I_{AB}(m,n)$, which means more than t nets coexist in this grid, can be written as

$$\begin{aligned} \sum_{j=1}^{x} P^I_{AB_j}(m,n) &\geq x \cdot \sqrt[x]{\prod_{j=1}^{x} P^I_{AB_j}(m,n)} \\ &= x \cdot \sqrt[x]{\prod_{\substack{K\in\Omega(m,n)\\K\neq A,B}} (P^r_K \times \bar{P^s_K})} \end{aligned} \tag{3.15}$$

where $\Omega(m,n)$ indicates the set that nets may pass through grid(m,n). $P^I_{AB_j}(m,n)$ represents the jth illegal term of net A and B in grid(m,n), and x denotes the number of terms of $P^I_{AB}(m,n)$, which can be computed as following

$$\begin{aligned} \text{\# terms of } P^L_{AB}(m,n) &= \sum_{i=2}^{t-2} C_i^{t-2} \\ \text{\# terms of } P^I_{AB}(m,n) &= x = 2^{t-2} \\ &- (\text{\# terms of } P^L_{AB}(m,n)) \end{aligned} \tag{3.16}$$

similarly, $P^L_{AB}(m,n)$ denotes the number of terms of the *legal probability* of net A and B in grid(m,n). Also, the P_K and $\bar{P_K}$ indicate the probability that net K goes and does

not go through this grid, and r as well as s denote the number of terms that net K goes and does not go through grid(m,n), respectively. They can be computed by the following equation:

$$\begin{aligned} r &= \frac{1}{h-2}\sum_{p=-1}^{h-t-2}[(t+p)C_{t+p}^{h-2}] \\ s &= \sum_{p=-1}^{h-t-2}[(1-\frac{t+p}{h-2})C_{t+p}^{h-2}] \end{aligned} \tag{3.17}$$

Consequently, the upper bound of the legal probability of net A and B in grid(m,n) can be written as:

$$P_{AB}^{L}(m,n) = P_A(m,n) \times P_B(m,n) \times [1 - \sum_{j=1}^{x} P_{AB_j}^{I}(m,n)] \tag{3.18}$$

In order to further clarify the equations of the upper bound of the coexisting probability, we give an example as follows. Assume that there are 3 available routing tracks in grid(m,n), and 5 nets (net A, B, C, D, and E) may go through the grid. Then the $overflow = 5 - 3 = 2$, $P_{AB}(m,n)$ is consisted of

$$\begin{aligned} P_{AB}(m,n) &= P[AB\bar{C}\bar{D}\bar{E}] + P[AB\bar{C}\bar{D}E] + P[AB\bar{C}D\bar{E}] + P[ABC\bar{D}\bar{E}] \\ &+ P[AB\bar{C}DE] + P[ABC\bar{D}E] + P[ABCD\bar{E}] \\ &+ P[ABCDE] \end{aligned} \tag{3.19}$$

where the legal terms are the first four ones, and the amounts of them can be computed by using Equation (3.16)

$$\begin{aligned} \text{\# terms of } P_{AB}^{L}(m,n) &= C_2^3 + C_3^3 = 4 \\ \text{\# terms of } P_{AB}^{I}(m,n) &= 2^{\#tracks-2} - (\text{\# terms of } P_{AB}^{L}(m,n)) \\ &= 2^3 - 4 = 4 \end{aligned} \tag{3.20}$$

In Equation (3.19), we can see that the number of illegal terms of C (or D, E) are 3, and number of illegal term of $\bar{C}$ (or $\bar{D}$, $\bar{E}$) is 1. Their computations are corresponding to the coefficient r and s stated in Equation (3.17).

While the coefficient r and s are obtained, the illegal probability of net A and B can be calculated by Equation (3.15):

$$\begin{aligned}\sum_{j=1}^{4} P^I_{AB_j}(m,n) &\geq 4 \cdot \sqrt[4]{\prod_{j=1}^{4} P^I_{AB_j}(m,n)} = 4 \cdot \sqrt[4]{\prod_{\substack{K\in\Omega(m,n)\\K\neq A,B}} (P^r_K \times \bar{P^s_K})} \\ &= 4 \cdot \sqrt[4]{(P(C))^3(P(D))^3(P(E))^3 \cdot P(\bar{C})P(\bar{D})P(\bar{E})} \quad (3.21)\end{aligned}$$

Finally, we add the the result acquired from Equation (3.21) into the Equation (3.18), and the upper bound of the coexisting probability of net A and B can be obtained.

3.4 Partial Estimation

In general, utilizing the algorithm stated in Fig. 3.2 and Table 3.3 is capable to accomplish the crosstalk-driven placement successfully. However, since each B*-tree perturbation in SA algorithm only selects 1 or 2 blocks to move, flip, or rotate. The above procedure costs too much runtime to analyze many redundant nets. That is, we should only analyze the nets whose belonging blocks are moved after performing last B*-tree perturbation. Also, we should re-analyze the nets whose belonging blocks are shifted due to their neighboring blocks are moved. We call this procedure to be "*partial analysis(estimation)*", and the flow stated in Table 2.1 and Table 3.3 are called "*complete analysis(estimation)*".

In the following sub-sections, we will introduce the partial estimation of the congestion and probabilistic noise, respectively. Then finally, the algorithm and several properties will be given to show how these procedures work.

3.4.1 Partial Congestion Estimation

After perturbing a B*-tree, the block selected in this perturbation are recorded. Since we not only re-analyze the nets belonging to the selected blocks but also need to care about the blocks which is shifted due to their neighbors, we record the placement order of each block after a perturbation. Fig. 3.8 illustrates the procedure: Assume that there are two nodes, n_4 and n_5 selected for perturbing, and resulting in a B*-tree shown in Fig.

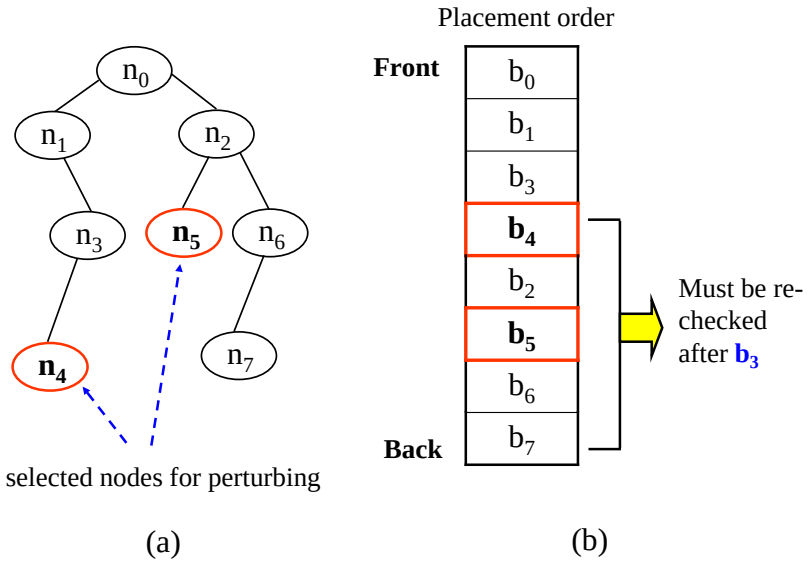


Fig. 3.8: Re-check the blocks whose placement orders after that of the selected nodes. (a) Node n_4 and n_5 are selected during a B*-tree perturbation. (b) The corresponding placement order.

3.8(a). Then we record its corresponding placement order exhibiting in Fig. 3.8(b). If the placement order of a block is after b_3, its new placement location should be compared to its previous. If it is moved, the probabilistic noise and usages of the nets that belongs to the block have to be re-analyzed. This is because the blocks whose placement orders before b_4 are placed earlier, and their new locations will be the same as their previous.

Making the record of placement orders to determine which blocks should be re-analyzed is more efficiently than the exhaustive location comparison, and can save much more runtime in the partial estimation procedure.

After determining which blocks' locations are shifted, the nets belonging to these blocks begin to re-analyze. Fig. 3.9 illustrates the procedure of the partial congestion estimation. Assume that the original net A (called *old_net A*) exists in the *old_mesh(A)* where is the yellow region. If it is moved to the top-left corner of the chip after a perturbation, we should compute its new usage and add it into every grid within the *new_mesh(A)*, and update the congestion. Also, the old existence of the *old_net A* within its original mesh has

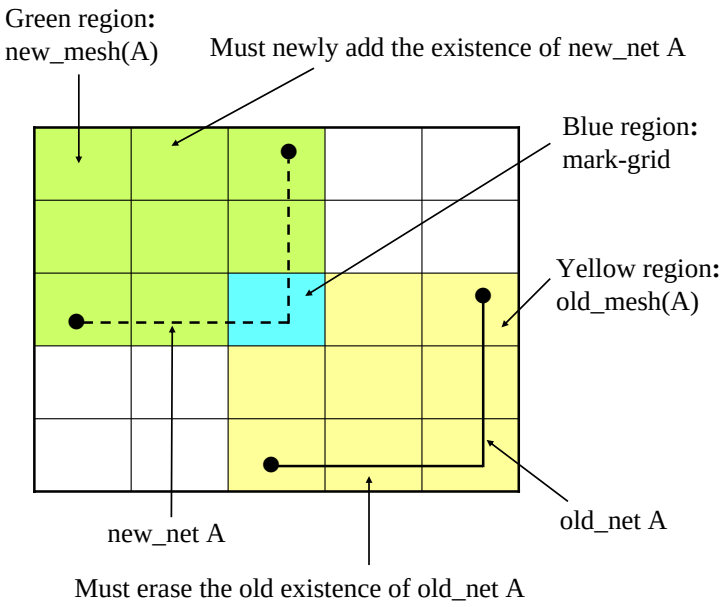


Fig. 3.9: Illustration of the partial congestion estimation procedure.

to be erased and updated. Furthermore, if there are *mark-grids* between the *old_mesh(A)* and *new_mesh(A)* ($mark\_grid(A) \in \{old\_mesh(A) \cap new\_mesh(A)\}$), we only need to re-compute the congestion within these grids and without updating the existence of net A.

Table 3.4 shows the algorithm of the partial congestion estimation. First, the approach illustrated in Fig. 3.8 is employed to determine which nets should be re-analyzed, we call them the *changed-nets* herein. For each changed-net, in addition to perform the congestion estimation similar to before, we update its existence and probabilistic usages within each grid of its *old_mesh* as well as *new_mesh*. Finally, we compute the congestion for each grid and finish the partial congestion estimation.

3.4.2 Partial Probabilistic Noise Estimation

The concept of the partial noise estimation is similar to that of the partial congestion estimation but more complicated. We use the configuration shown in Fig. 3.9 again to interpret the procedure of the partial noise estimation: When net A is moved to the

Algorithm of Partial Congestion Estimation	
Input: *Netlist of the placement*	
Output: *Probabilistic usages of each two-pin net; Congestion of each grid*	
1	**Begin**
2	**For** each changed-net
3	MST(net)
4	**For** each segment of the MST
5	Determine the size of *new_mesh*
6	Erase the *old_usage* within its *old_mesh*
7	Erase the *old_existence* within its *old_mesh* excluding *mark_grids*
8	Compute the horizontal and vertical usages within its *new_mesh*
9	Update the new_existence within its *new_mesh*
10	**EndFor**
11	**EndFor**
12	**For** each grid in the design
13	Compute the congestion of the grid
14	**EndFor**
15	**End.**

Table 3.4: Algorithm of the partial congestion estimation.

new_mesh(A) where is the green region, the probabilistic noise of the nets which relate to net A within *old_mesh(A)* should be updated firstly. That means, if there is a net (called net B) where in the *old_mesh(A)* is not shifted, its probabilistic noise has to subtract the interference due to net A. Also, the probabilistic noise of net A should be updated by subtracting the interference resulting from net B.

After the update of *old_mesh(A)* is complete, we carry on the noise update in *new_mesh(A)*. Similar to complete noise estimation stated in Table 3.3, if there is a net (called net C) within the *new_mesh(A)*, and it does not exist in the *old_mesh(A)*, it needs to add the interference due to net A. Similarly, net A also adds the interference resulting from net C.

In brief, we can summarize the properties of the noise update as follows:

- The net which is a changed-net (called net A): It needs to subtract the interference due to the related nets where in the *old_mesh(A)*, and re-compute the new proba-

bilistic noise due to the nets where in the *new_mesh(A)*.

- The net exists in *old_mesh(A)* "*originally*" (means that it is not a changed-net), and does not belong to *new_mesh(A)*: Only subtracting the interference due to net A.

- The net whose minimum routing region covers the *old_mesh(A)* and *new_mesh(A)*: Subtracting the old interference due to net A, then adding the new interference results from the net A.

Modifying the step $2 \sim 4$ exhibiting in Table 3.3 by the above properties can determine the manner of the noise update for each net, and accomplish the partial noise estimation. Because the coupling noise results from the mutual inductance costs considerable efforts to be handled, performing the partial noise estimation is necessary and can improve much runtime during placement.

3.5 Algorithm Flow of Crosstalk-Driven Placement

In this section, the whole procedures are integrated and the overall algorithm of our crosstalk-driven placement will be given. Table 3.5 illustrates the full algorithm of our crosstalk-driven placement.

In the first place, we set perturb_flag to be false and all the nets to belong to the changed-net, that means, it performs once complete estimation for the congestion and probabilistic RLC noise, which is stated in Table 2.1 and Table 3.3, respectively.

After executing the complete estimation for each two-pin net, the outcome performance of the placement is judged in the line 8. If it meets the noise constraint or the temperature of SA is cool enough, then exit the placement procedure. Otherwise, the B*-tree perturbation is performed to seek a better solution for placement, and set the perturb_flag to be true. Also, the members of the changed-net are updated by the manner stated in Chapter 3.4.1.

While the perturbation is complete, we execute the partial congestion estimation and

Algorithm of Crosstalk-Driven Placement

Input: *Netlist of the design*

Output: *Area, total estimated wirelength, and probabilistic RLC noise of the placement*

```
1   Begin
2     Set perturb_flag ← false
3     Set {changed-net} ← {all the nets}
4     For each net of the {changed-net}
5       If perturb_flag = false
            Perform the complete congestion estimation ;
            Perform the complete probabilistic RLC noise estimation ;
6       Else
            Perform the partial congestion estimation ;
            Perform the partial probabilistic RLC noise estimation ;
7     EndFor
8     If meet the noise constraint or SA cooling enough
        Exit placement ;
9     Else
        Perturb B*-tree ;
        Set perturb_flag ← true ;
        Update the {changed-net} ;
        Goto line 4 ;
10  End.
```

Table 3.5: Algorithm of the proposed crosstalk-driven placement.

partial probabilistic RLC noise estimation for each changed-net, which are illustrated in Table 3.4 and Chapter 3.4.2, respectively. The above procedures are iteratively proceed until one of the conditions is satisfying, then exit the placement.

Chapter 4

Experimental Results

In order to check the validity of our proposed placement, we test our method on MCNC benchmarks and two additional cases, ckt529 and ckt1681. The number of cells and nets of each test case is shown in Table 4.1. The proposed placement is implemented by C++ language, and run on a Pentium IV 3.2 GHz with 2GB memory.

We compare the results with the area-driven, congestion-driven, RC-driven, and RLC-driven placement. In the area-driven placement, it only minimizes the placement area. In the congestion-driven placement, we minimize the area, total wirelength, number of overflowing grids, and overall routing density. In the others, they minimize the area, total wirelength, the overall routing density, and penalize the overflow to prevent from congestion, then use the proposed algorithm to minimize the crosstalk noise. The difference of RC-driven and RLC-driven placement is, the former only takes R, L, C, C_x of wires into account, but the latter one extra considers the mutual inductance. After each placement

Benchmark	# cells	# nets
apte	9	97
hp	11	83
xerox	10	203
ami33	33	123
ami49	49	408
ckt529	529	613
ckt1681	1681	1991

Table 4.1: Number of cells and nets of MCNC and our benchmarks.

Placement flow of area-driven placement	
1	Parsering the input netlist
2	Minimize the total area /*$cost = \alpha(Area)$*/
3	Probabilistic RLC noise verification
4	Maximum routing density analysis

Table 4.2: Placement flow of the area-driven placement.

is complete, its overall average probabilistic RLC noise is verified by eq.(3.14). Here, we calculate the *true* legal coexisting probability instead of the upper bound of legal coexisting probability of two nets to verify the real probabilistic noise of a design.

All of the testcases utilize $0.13\mu m$ / 1.2V technology and two metal layers for congestion estimation. Furthermore, we set the input signal rising time $T_r = 100ps$ and the resistance of each pin is around 30 to 120Ω which is proportional to its cell area, and a shielding wire is inserted between each 10 wires. Hence, the crosstalk noise is considered ranging over 10 units of space. The experimental results are shown in Table 4.6 $\sim$ 4.8, where *WL*, *max_H*, and *max_V*, indicate the total wirelength estimated by the half-perimeter wirelength technique, the maximum estimated horizontal and vertical routing density of the overall global routing grids, respectively. In addition, the *P_RC*, *P_RLC*, and *Peak_RLC noise* denote the overall average probabilistic RC, RLC noise, and the peak probabilistic RLC noise, respectively.

We set up the environment for each placer to compare their experimental results as following, and the sum of cost coefficient for each cost function is equal to 1.

- **Area-driven placement:**

  The placement flow of area-driven placement is exhibited in Table 4.2. For the results of area-driven placement shown in Table 4.6, it only minimizes the total area (cost coefficient $\alpha = 1$), then it obtains the minimum placement area but sacrifices the total wirelength, routability, and crosstalk immunity. It even may be unroutable in several benchmarks if the maximum estimated density is larger than 1.

Placement flow of congestion-driven placement	
1	Parsering the input netlist
2	Minimize the cost fuction /*$cost = \alpha(Area) + \beta(WL) + \gamma(avg\_congestion) + \delta$(number of overflowing grids)*/
3	Probabilistic RLC noise verification
4	Maximum routing density analysis

Table 4.3: Placement flow of the congestion-driven placement.

Placement flow of RC-driven placement	
1	Parsering the input netlist
2	Minimize the cost fuction /*$cost = \alpha(Area) + \beta(WL) + \gamma(avg\_congestion) + \delta(P\_RC)$*/
3	Probabilistic RLC noise verification
4	Maximum routing density analysis

Table 4.4: Placement flow of the RC-driven placement.

- **Congestion-driven placement:**

  The placement flow of congestion-driven placement is indicated in Table 4.3. In the congestion-driven placement, it simultaneously minimizes the total area, wirelength, average congestion, and the number of overflowing grids, respectively. That means, it sets $\alpha + \beta + \gamma + \delta = 1$.

  We instinctively figure that minimizing the congestion is equivalent to mitigate the coupling effects between interconnects, and the overall crosstalk noise can be controlled. However, it is negated in the experimental results of congestion-driven placement shown in Table 4.7. It is because the coupling noise is not only dominated by C_x and L_x but also by the pin resistances. If the net in a higher coupling region but with a stronger driver, it will have stronger noise immunity and suffer smaller noise. That is the reason that only minimizing the congestion cannot achieve the best noise immunity.

- **RC-driven placement:**

Placement flow of RLC-driven placement	
1	Parsering the input netlist
2	Minimize the cost fuction /*$cost = \alpha(Area) + \beta(WL) + \gamma(avg\_congestion) + \delta(P\_RLC)$*/
3	Maximum routing density analysis

Table 4.5: Placement flow of the RLC-driven placement.

The placement flow of RC-driven placement is exhibited in Table 4.4. For the RC-driven placement, it simultaneously minimizes the total area, wirelength, average congestion, and the probabilistic RC noise (P_RC), respectively. That means, it sets $\alpha + \beta + \delta + \gamma = 1$. After placement is complete, we verify its probabilistic RLC noise again to check the signal integrity.

The results shown in Table 4.8 reveals that the RC-driven placement has smaller area, higher congestion but worse noise immunity on average than the RLC-driven placement. We speculate that it is due to without the mutual inductance consideration, the probabilistic RC noise may be still slight enough in the higher congested region. Moreover, for the comparison of P_RC and P_RLC indicated in Table 4.8, the RC model indeed underestimates the crosstalk noise about *3X* against that of RLC model in our testcases.

- **RLC-driven placement:**

  The placement flow of area-driven placement is exhibited in Table 4.5. Similar to RC-driven placement, the difference between the RC and RLC-driven placement is: RLC-driven considers the probabilistic RLC noise (P_RLC) instead of P_RC.

  In RLC-driven placement, it achieves the best performance in the total wirelength and crosstalk noise against the other placers. Because considering the effect of the mutual inductance, its placement area is a little larger than that of the others. From the experimental results shown in Table 4.9, RLC-driven placement averagely improves 8.9%, and 15.9% in the probabilistic RLC noise than that of the RC-

Benchmark	**Area-driven**			
	Area (mm^2)	WL (mm)	max_H / max_V	P_RLC noise ($\times 10^{-3}$)
apte	49.801	781.102	0.512 / 0.624	2.977
hp	10.158	436.200	0.844 / 0.931	1.898
xerox	20.774	710.513	1.534 / 1.816	3.601
ami33	1.320	211.006	0.913 / 1.044	2.219
ami49	38.369	1420.180	2.209 / 1.955	6.107
ckt529	46.996	4012.700	2.969 / 2.084	26.512
ckt1681	101.710	16058.4	3.018 / 2.966	133.518
Comparison	0.916	1.301	2.312 / 2.458	1.441

Table 4.6: Results of area-driven placement.

Benchmark	**Congestion-driven**			
	Area (mm^2)	WL (mm)	max_H / max_V	P_RLC noise ($\times 10^{-3}$)
apte	50.174	521.280	0.205 / 0.326	2.469
hp	10.742	339.157	0.485 / 0.437	1.211
xerox	21.168	679.828	0.544 / 0.681	1.633
ami33	1.411	149.247	0.537 / 0.610	1.485
ami49	40.012	1251.050	0.725 / 0.688	4.016
ckt529	50.098	3019.761	0.802 / 0.691	15.749
ckt1681	109.424	14903.5	0.908 / 0.811	115.680
Comparison	0.963	1.149	0.810 / 0.914	1.159

Table 4.7: Results of congestion-driven placement.

driven and congestion-driven placement, respectively. And it also improves 6.8% and 14.9% in the total wirelength on average than that of the above two placers, and merely sacrifices 8.4% in the area averagely compared to the area-driven placement, but it may be unroutable in the most benchmarks.

In addition, Table 4.10 shows the comparison of the peak probabilistic RLC noise. We can see that the peak noise of the area-driven placement is still the largest, and our algorithm can mitigate more RLC peak noise than that of the other placers.

Further showing our experimental results, the results of RLC-driven placement configuration of all the benchmarks are exhibited in Fig. 4.1 $\sim$ 4.7.

Benchmark	RC-driven				
	Area (mm^2)	WL (mm)	max_H / max_V	P_RC noise ($\times 10^{-3}$)	P_RLC noise ($\times 10^{-3}$)
apte	50.002	651.330	0.425 / 0.468	0.608	2.191
hp	10.721	304.021	0.796 / 0.531	0.398	1.358
xerox	22.084	655.243	0.788 / 0.841	0.526	1.542
ami33	1.439	120.771	0.743 / 0.759	0.411	1.364
ami49	41.008	1206.030	0.835 /0.687	0.796	3.852
ckt529	51.712	2877.640	0.896 / 0.782	1.327	13.658
ckt1681	111.238	13759.700	0.929 / 0.878	13.923	109.65
Comparison	0.980	1.068	1.043 / 1.065	-	1.089

Table 4.8: Results of RC-driven placement.

Benchmark	RLC-driven			
	Area (mm^2)	WL (mm)	max_H / max_V	P_RLC noise ($\times 10^{-3}$)
apte	50.019	516.074	0.368 / 0.433	1.799
hp	11.005	281.698	0.731 / 0.535	1.162
xerox	22.301	640.531	0.796 / 0.822	1.447
ami33	1.470	102.305	0.711 / 0.638	1.196
ami49	41.522	1126.352	0.799 / 0.657	3.529
ckt529	53.883	2635.692	0.873 / 0.717	10.064
ckt1681	113.767	12862.05	0.912 / 0.841	103.550
Comparison	1.0	1.0	1.0 / 1.0	1.0

Table 4.9: Results of RLC-driven placement.

Benchmark	**Area-driven**	**Congestion-driven**	**RC-driven**	**RLC-driven**
	Peak_RLC noise ($\times 10^{-3}$)	Peak_RLC noise ($\times 10^{-3}$)	Peak_RLC noise ($\times 10^{-3}$)	Peak_RLC noise ($\times 10^{-3}$)
apte	31.279	33.517	20.031	15.714
hp	15.891	17.729	11.507	9.916
xerox	94.112	48.025	46.331	45.092
ami33	105.741	98.552	95.997	89.120
ami49	77.510	58.124	43.716	42.005
ckt529	172.013	153.440	134.199	117.245
ckt1681	482.600	421.192	393.352	376.264
Comparison	1.408	1.194	1.072	1.0

Table 4.10: Peak probabilistic RLC noise of each benchmark.

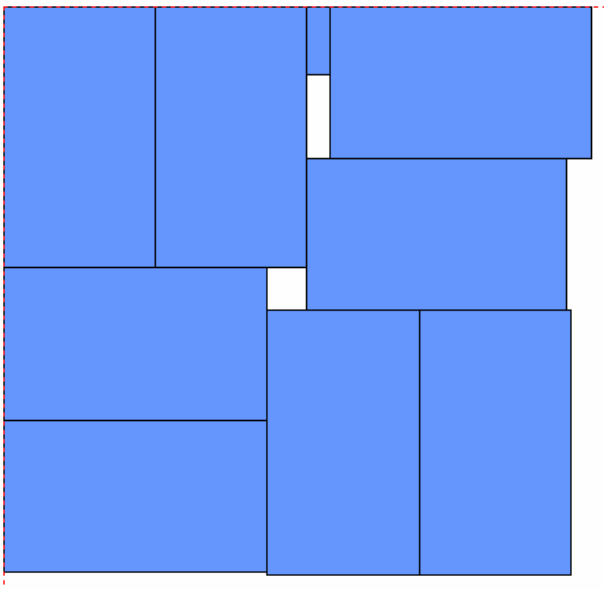


Fig. 4.1: The RLC-driven placement configuration of apte.

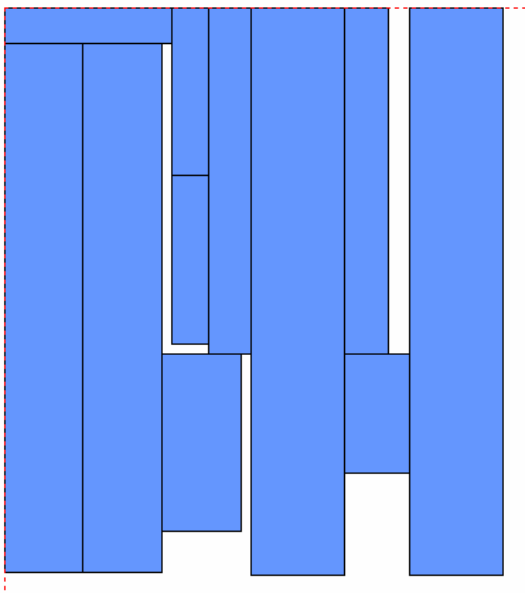


Fig. 4.2: The RLC-driven placement configuration of hp.

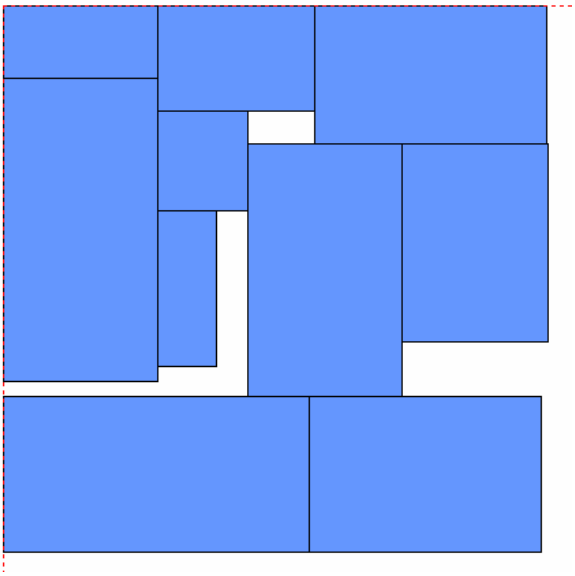


Fig. 4.3: The RLC-driven placement configuration of xerox.

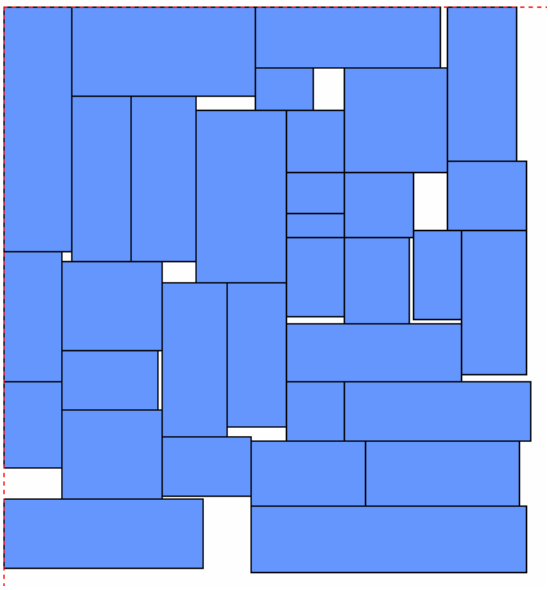


Fig. 4.4: The RLC-driven placement configuration of ami33.

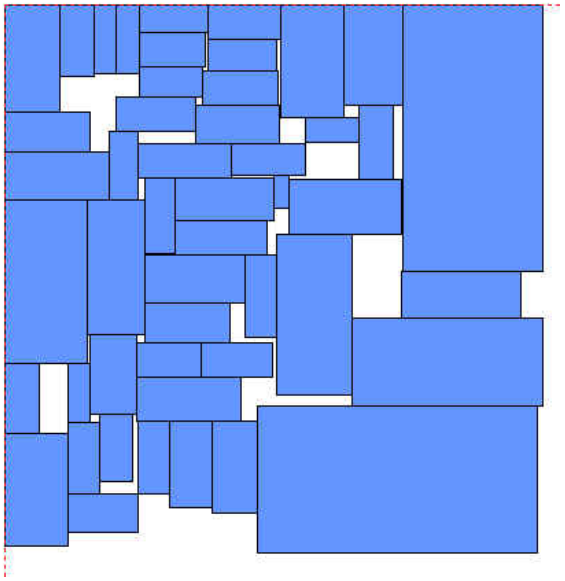


Fig. 4.5: The RLC-driven placement configuration of ami49.

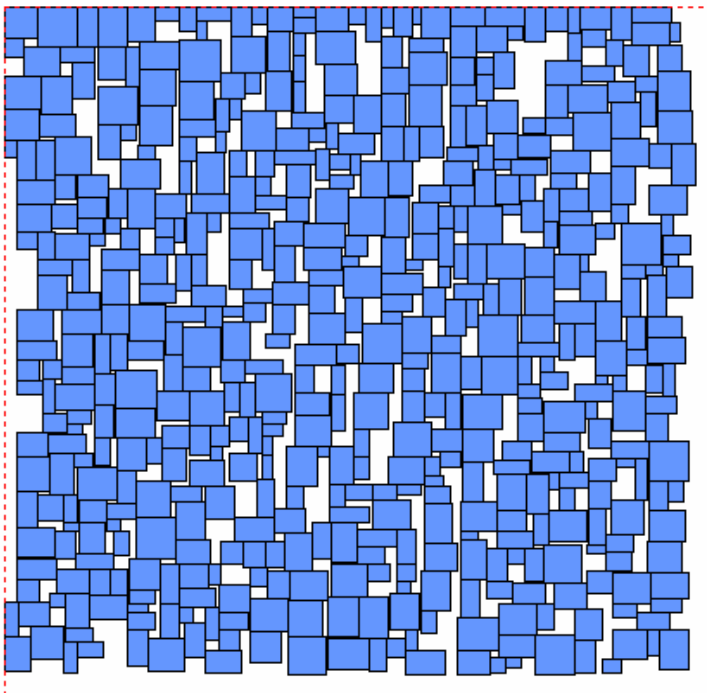


Fig. 4.6: The RLC-driven placement configuration of ckt529.

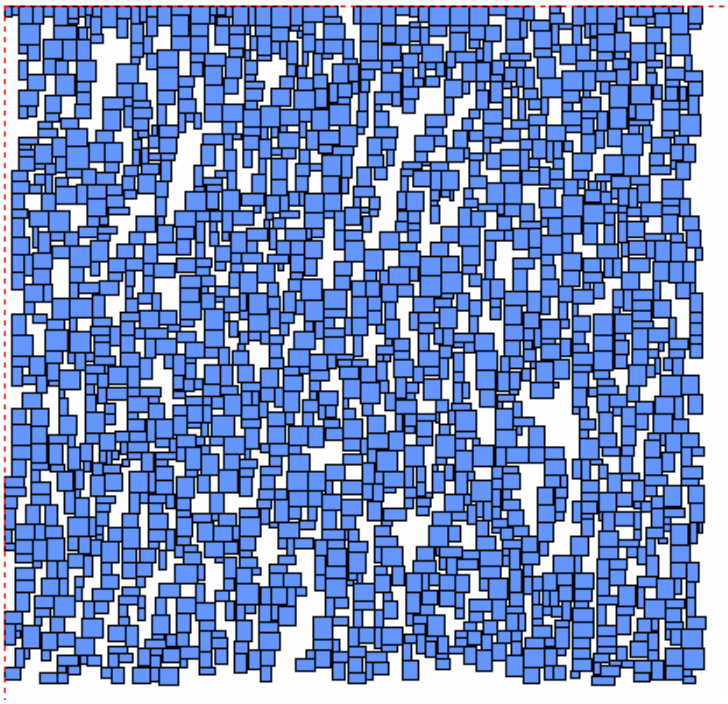


Fig. 4.7: The RLC-driven placement configuration of ckt1681.

Chapter 5

Conclusion

This thesis proposes a novel algorithm to handle the on-chip RLC noise during placement. Results show that our algorithm can indeed estimate the RLC noise and deal with the mutual inductance effectively. In the research, there are several conclusions given as follows:

- Ignoring the interferences of the mutual inductance will significantly underestimate the noise effects during placement. Due to advance of the technology, the circuit complexity of our design is much more than before. More and more interconnects parallel to each other makes the crosstalk noise become seriously. Experimental results show that only considering the RC noise during placement will averagely underestimate *3X* probabilistic noise than using RLC model. Therefore, only performing the RC-driven placement is not enough for today's VLSI physical design. Be excessively optimistic in the on-chip noise of the circuit will malfunction our design, even make it fail.

- Minimizing the congestion cannot truly mitigate the crosstalk noise. The results exhibit that the congestion-driven placement obtains an inferior performance in the probabilistic RLC noise than that of the RC and RLC-driven placement. This is due to the coupling noise is not only dominated by L_x and C_x, but also by the pin resistances. Consequently, only minimize the routing congestion cannot entirely solve the noise problems during placement.

To our best knowledge, we are the first one to consider the on-chip noise due to the mutual inductance during placement, and results show the proposed algorithm achieve excellent performance in the placement stage. In order to perfect the research, our future works will focus on the following objects:

- **Runtime reduction:** Since the computation complexity in crosstalk-driven placement is much more than other objective placement (ex: congestion-driven, low power driven placement, etc.), especially in considering the effects of mutual inductance. *Multilevel* is a good structure for large scale placement, it efficiently reduces data size to be handled at a time more than that of the flat placement. Therefore, it is a good choice for our future going.

- **Algorithm modification:** Our proposed algorithm for probabilistic RLC noise estimation can work well and achieve great results in the placement stage. However, it still limits to the long coupling range due to the mutual inductance, and have to handle a huge computation complexity. Our another future work is to keep on modifying the algorithm of RLC noise estimation, then speed up the overall placement procedure.

- **Extend to crosstalk-driven routing:** In the thesis, the transmission line based RLC model has worked well in RLC noise estimation, and it is also capable to handle the case of non-identical wires. Since the noise effects of a circuit is more obvious in the route stage, the model can be applied to the procedure of the crosstalk-driven routing, and will achieve a good performance.

Bibliography

[1] D. A. Kirkpatrick, "The Deep-Submicron Signal Integrity Challenge," *International Symposium on Physical Design*, 1999.

[2] K. L. Shepard, V. Narayanan, and R. Rose, "Harmony: Static Noise Analysis of Deep Submicron Digital Integrated Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 18, August. 1999.

[3] D. A. Kirkpatrick, "Modeling and Analysis of Differential Signaling for Minimizing Inductive Cross-Talk," *Design Automation Conference*, pp. 804-809, June, 2001.

[4] Howard H. Chen and L. K. Wang, "Design for Signal Integrity: The New Paradigm for Deep-Submicron VLSI Design," *International Symposium on Physical Design*, pp. 329-333, 1997.

[5] C. K. Cheng, J. Lillis, S. Lin, and N. H. Chang, "Interconnect Analysis and Synthesis," *John-Wiley*, 2000.

[6] G. Servel, F. Huret, E. Paleczny, J. F. Legier, and D. Deschacht, "Effective On-Chip Inductance Modeling for Multiple Signal Lines and Application on Repeater Insertion," *International Symposium on Quality Electronic Design*, pp. 185-190, 2001.

[7] Jun Chen and Lei He, "Determination of Worst-Case crosstalk Noise for Non-Switching Victims in GHz+ Interconnects," *Asia and South Pacific Design Automation Conference*, 2003.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, "Introduction to Algorithms," *Cambridge, MA: MIT Press*, 1992.

[9] Yu Cao, X. D. Yang, X. Huang, and D. Sylvester, "Switch-Factor Based Loop RLC Modeling for Efficient Timing Analysisw," *International Conference on Computer-Aided Design*, Sep. 2003.

[10] J. Cong, D. Z. Pan, and P. V. Srinivas, "Improved Crosstalk Modeling for Noise Constrained Interconnect Optimization," *Asia and South Pacific Design Automation Conference*, Jan. 2001.

[11] K. Agarwal, D. Sylvester, and D. Blaauw, "A Simplified Transmission-Line Based Crosstalk Noise Model for On-Chip RLC Wiring," *Asia and South Pacific Design Automation Conference*, pp. 859-865, 2004.

[12] J. Lou and W. Chen, "Crosstalk-Aware Placement," *IEEE Design and Test of Computers*, pp. 24-32, Jan. 2004.

[13] J. Lou, S. Thakur, S. Krishnamoorthy, and H. S. Sheng "Estimating Routing Congestion Using Probabilistic Analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 21, Jan. 2002.

[14] H. Ren, D. Z. Pan, and P. G. Villarrubia, " True Crosstalk Aware Incremental Placement with Noise Map," *International Conference on Computer-Aided Design*, Sep. 2004.

[15] M. Yoshikawa and H. Terai, "Crosstalk-Driven Placement Based on Genetic Algorithms," *IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, July. 2004.

[16] G. M. Wu, Y. C. Chang, and Y. W. Chang, "Rectilinear Block Placement using B*-Trees," *TODAES*, Vol. 8, pp. 188-202, 2003.

[17] P. N. Guo, C. K. Cheng, abd T. Yoshimura, "An O-Tree Representation of Non-Slicing Flooplans and Its Applications," *Design Automation Conference*, pp. 268-273, 1999.

[18] S. Kang, "Linear Ordering and Application to Placement," *Design Automation Conference*, pp. 457-464, 1983.

[19] S. C. Wang, G. Y. Lee, and D. J. Ma, "Modeling of Interconnect Capacitance, Delay, and Crosstalk in VLSI," *IEEE Transactions on Semiconductor Manufacturing*, Vol. 13, Feb. 2000.

[20] F. W. Grover, "Inductance Calculations: Working Formulas and Tables," *Dover*, New York 1962.

[21] H. Kim and C. C.-P. Chen, "Be Careful of Self and Mutual Inductance Formulae," *Reports*, 2001.

[22] Naveed A. Sherwani, "Algorithms for VLSI Physical Design Automation, 3rd Edition," *Kluwer Academic Publishers*, 1999.

[23] S. M. Sait and H. Youssef, "VLSI Physical Design Automation: Theory and Practice," *World Scientific Publishing*, 1999.

[24] B. Young, "Digital Signal Integrity," *Prentice Hall*, 2001.