Chip-Level Thermal Analysis, Modeling, and Optimization Using Multilayer Green's Function

by

Baohua Wang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2008

to my family

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor, Prof. Pinaki Mazumder, for his continued support and wise guidance throughout my graduate study. He allowed me to explore different research topics and consistently encouraged me to make progress. Having worked with him for more than five years, I am sure that his creativity and insight will continue to influence my future work greatly.

I want to express my sincere gratitude to Professor Michel M. Maharbiz, Professor Scott Mahlke, and Professor Pei-cheng Ku for serving on the committee and giving me invaluable suggestions during my proposal. I also thank Professor Igor Markov for providing me with the Capo placement tool and advising me on its use.

I would like to thank the members of our research group — Li Ding, Qinwei Xu, and Hui Zhang, whom I first met here, as well as Taeli Jung, Byungsoo Kim, Woo Hyung Lee, Sing-Rong Li, Yu-Wei Lin, Manoj Rajagopalan, Dan Shi, Kyungjun Song, and Dr. Jianping Sun.

I want to thank all of the friends I met at the University of Michigan, particularly Amit Jain, Chongzheng Na, Chang-Hao Tsai, Jianhui Wu, Jun Yang, Liming Zhang, Jimin Zhao, Xin Zhao, and Ran Zhuo.

Finally, I want to thank my family for their encouragement and support throughout this long journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

Introduction

1.1 Motivation

As predicted by Moore's law, the number of transistors integrated into an ultra-large scale integrated (ULSI) circuit continues to grow exponentially. The current complementary metal-oxide-semiconductor (CMOS) process technology doubles the transistor count about every two years, by progressively scaling MOS devices in gate oxide thickness, channel length, doping level, etc. Meanwhile, the dimensions of the on-chip interconnects shrink and the number of metalization levels increases. The amount of heat dissipation in devices and interconnects causes thermal management and chip cooling issues. A state-of-the-art microprocessor chip, designed with sub-100-nanometer CMOS process technology, often integrates hundreds of millions of transistors. Precisely modeling circuit power dissipation and temperature gradients within the chip becomes necessary to ensure circuit performance and reliability because the circuit operates at the level of several gigahertz (GHz) frequency.

The enormous number of on-chip transistors switch in a pseudo-random fashion. Traditionally, the transistor-switching-induced circuit dynamic power consumption, denoted by P_{dyn}, dominated the total power dissipation of the chip. Circuit dynamic power consumption has often been estimated by

$$P_{dyn} = afC_{total}V_{DD}^2$$

where a denotes the switching factor, f denotes the circuit operating frequency, C_{total} denotes the total capacitance, and V_{DD} denotes the circuit operating voltage. Nowadays, circuit leakage power becomes prominent because miniaturizing the chip dimensions increases the gate tunneling and sub-threshold currents. Leakage power is predicted to be a large portion of the total power consumption of the chip in the upcoming years. Notably, circuit temperature super-linearly affects MOSFET sub-threshold leakage current, as depicted by

$$I_{leakage} \propto T^2 e^{\frac{q(V_{gs}-v_t)}{nkT}} \left(1 - e^{-\frac{qV_{ds}}{kT}}\right)$$

where T denotes the circuit temperature, v_t denotes the transistor threshold voltage, q denotes the electron charge, and k denotes the Boltzmann's constant [79]. For a low-power circuit that mainly operates in the sub-threshold region [41, 68, 16, 74], accurately analyzing the circuit temperature can be critical in calculating the leakage current. Today, a typical microprocessor chip dissipates more than 100 Watts peak-power in a 350-500 mm^2 die. Extremely high power density and high temperature increase the cooling cost for the chip and decrease circuit performance because the circuit operating frequency needs to be temporarily slowed down to reduce heat dissipation in the chip.

Circuit temperature also contributes to many types of subtle timing failures in the circuit. According to the alpha-power law MOSFET model [57], gate delay G_{delay} relates to the MOSFET drain-source current I_{ds} by

$$G_{delay} \propto \frac{CV_{DD}}{I_{ds}} \propto \frac{CV_{DD}}{\mu\left(V_{DD} - v_t\right)^{\alpha}}$$

where α is the MOSFET velocity-saturation factor. The threshold voltage v_t and the carrier mobility μ relate to the circuit temperature by

$$v_t(T) \propto v_t(T_0) - \kappa(T - T_0)$$

$$\mu(T) \propto \mu(T_0)(T/T_0)^{-m}$$

where T_0 is the room temperature, κ is the threshold voltage temperature coefficient, and m is the mobility temperature exponent [39]. Both the carrier mobility and threshold voltage reduce at a higher circuit temperature. Thus the gate delay decreases if the gate supply voltage is larger than the zero-temperature coefficient (ZTC) point (e.g., 1.2 V) and increases if the gate supply voltage is below the ZTC point [39].

Besides influencing the gate timing, temperature distribution inside a chip also prominently affects the delay in propagating signals through the on-chip interconnects. Using low dielectric constant (low-k) materials in interconnects has considerably reduced the capacitive coupling between adjacent interconnects. However, using low-k materials simultaneously increases the thermal impedance from the interconnect wires to the chip heat sink, thereby causing more heat to accumulate at the interconnect wires and aggravating the interconnect self-Joule heating issue [7]. Rising interconnect temperature increases interconnect delay because interconnect resistivity becomes larger [19]. Thermal effects on the gate and interconnect timing make it very difficult to precisely tune the clock distribution network across the chip. In designing microprocessor circuits, temperature gradients within the chip in lieu of a set of worst- or best-case chip temperatures must be considered to control the clock skews and to avoid synchronization failures at the data-storage circuit elements [17, 71, 72].

On the other hand, increasing circuit power density and elevating chip temperature reduce the mean time to fail (MTTF) of a metal wire because the transport of metal ions significantly accelerates, as described by Black's equation:

$$MTTF\,(T) = AJ^{-2}e^{\phi/kT}$$

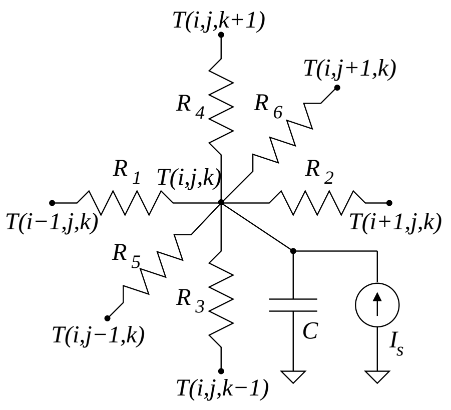
where J is the current density in the metal wire, and ϕ and A are technology-dependent parameters [13, 8, 18]. To meet the stringent reliability requirements concurrently with the demand for high performance, thermal-aware or thermally optimized chip design has now become a trend. Therefore, to analyze and optimize full-chip temperature distribution, an integrated circuit (IC) design automation tool should be able to handle millions of transistors and interconnects [35] and run in a repetitive fashion with short turnaround time.

1.2 Approaches for Thermal Analysis and Optimization

1.2.1 Grid-Based Approaches

The finite-difference (FD) method has traditionally been used in IC thermal analysis and optimization. To solve the steady-state or time-dependent heat conduction equation, the FD method discretizes the Laplacian operator with the second order central finite-difference scheme [22, 65]. The discretization forms an RC network that consists of a matrix of hexagonal junctions, with resistors to model the heat conductivities, capacitors to model the thermal diffusivities, and current sources to model the power densities. Fig.1.1 illustrates a hexagonal junction for discretizing the heat conduction equation, where ΔL denotes the grid size. To simulate the obtained RC network, one acceleration approach is to apply the Krylov-subspace-based model order reduction technique [49, 27].

To solve the time-dependent heat conduction equation, the alternating direction



$$\alpha\frac{\partial T(x,y,z)}{\partial t} = k\nabla^2 T(x,y,z) + f(x,y,z)$$

$$\nabla^2 T \approx \frac{T(i+1,j,k)+T(i-1,j,k)-2T(i,j,k)}{\Delta L^2} + \frac{T(i,j+1,k)+T(i,j-1,k)-2T(i,j,k)}{\Delta L^2} + \frac{T(i,j,k+1)+T(i,j,k-1)-2T(i,j,k)}{\Delta L^2}$$

$$R_1,\ldots,R_6 = \Delta L^2/k$$

$$C = \alpha, I_s = f(x,y,z)$$

Figure 1.1: Hexagonal junction for discretizing heat conduction equation

implicit (ADI) method exploits the smooth temperature variation in the temporal domain [76]. To discretize the Laplacian operator, the ADI method adopts the three-step Douglas-Gunn scheme, with a one-dimensional (1-D) FD scheme applied at each step. The FD method instead uses a 3-D FD scheme. The ADI method produces a tridiagonal system that can be solved by the Thomas algorithm in linear time [64]. Compared to the FD method, at every time step, the ADI method actually solves the same thermal network in three stages by the implicit backward-Euler FD scheme. At each solution stage, the voltage gradients along two of the coordinates are computed explicitly from the node voltages at the previous solution stages, while the voltage gradients along the remaining coordinate become unknowns to be solved from the system of equations.

The finite-element (FE) method has also been used in IC thermal analysis and optimization. The FE method uses a set of shaping functions, called basis functions, to interpolate the interior temperatures from the temperatures at the neighboring nodes [32]. Then the method represents the Laplacian of the interior temperatures by the Laplacian of the shaping functions. Finally, the method solves the node temperatures from the resultant system of equations. The aforementioned FD and FE methods are grid-based methods, which are advantageous in modeling detailed chip geometries such as inter-layer vias, bonding wires, buses, etc. However, grid-based methods need to dispense significant amounts of nodes to large-volume structures such as the bulk, substrate, and heat sink for the given chip [22, 65, 27, 76, 32]. The large problem size causes grid-based methods to have long run-time and limits their applications in ULSI physical design flow, especially when thermal simulation needs to be conducted for a large number of iterations [21]. Grid-based methods also have numerical stability issues [76]. When the grid-based methods are integrated with other numerical algorithms, numerical stability analysis becomes complicated because it is insufficient to analyze the stability of each numerical algorithm separately, as individually stable components may not ensure the entire system is constructed stably. In this case, the classical *Von Neumann Analysis* may be somewhat inadequate, and stricter stability criteria such as the passivity of the applied numerical algorithms may be used [11, 66, 70].

1.2.2 Green's Function-Based Approaches

Spectral methods based on the Fourier transform technique have been used in IC thermal analysis and optimization [43, 5, 29]. The spectral methods assume that the heat sources considered are in a 2-D rectangular region, so the power-density spectrum can be computed by the fast Fourier transform (FFT). To apply the spectral methods, heat sources must be on the top surface of the chip, and heat transfer is forbidden there. A chip with the wire-bonding packaging normally satisfies these requirements; however, a chip with the popular flip-chip packaging does not, because heat dissipates via both the bottom and top sides of the chip: the side with the cooling devices (heat sink, fan, etc.) and the side with the solder balls. The spectral methods are suitable for computing the temperature distribution incurred by a planar heat source distribution; however, they cannot calculate the temperature distribution incurred by heat sources of arbitrary shapes. Thermal analysis and optimization methods based on the multilayer heat conduction Green's function, and more flexible than the spectral methods, are introduced in this dissertation. The heat conduction Green's function gives the temperature distribution incurred by a Dirac delta heat source. Therefore, it can be used to solve the temperature distribution due to arbitrarily shaped heat sources by spatially convoluting with the power density distribution of the heat sources. The heat conduction Green's function can also be used to compute the thermal transfer impedance between any two locations in the chip.

Compared to the grid-based methods, the thermal analysis and optimization methods based on the heat conduction Green's function, named the Green's function-based methods, are advantageous at the earlier stages of ULSI physical design flow, such as floorplanning and cell placement [21, 67, 78]. The Green's function-based methods do not discretize the chip regions of no heat sources and of no monitored temperatures. Therefore, the Green's function-based methods improve the thermal simulation speed by a few orders of magnitude by not modeling otherwise costly chip regions such as the bulk. Based on a single-layer thermal (SLT) model, the heat conduction Green's function under various boundary conditions has been investigated and applied in the thermal analysis and optimization of ULSI chips [38, 21, 78]. However, considering only one type of heat conduction material in the simulated environment, the SLT model, is overly simplistic and inaccurate because the heat conduction path in a realistic chip involves multilayer heterogeneous heat conduction materials. Particularly, the SLT model is inadequate to analyze the cutting-edge 3-D ICs that vertically integrate multiple active layers [9].

The heat conduction Green's function is derived from the Poisson's equation that is widely studied in the context of extracting parasitic elements within an IC. However, the heat conduction Green's function differs from the Green's function for parasitic extraction in several aspects. In parasitic extraction, charge sources are presumed to be on the surfaces of the conducting geometries; therefore, a charge-sheet model is frequently used in parasitic extraction, such as in the FE and boundary-element (BE) methods [47, 24, 51]. In thermal analysis, heat sources instead span the 3-D volume space of the chip. Furthermore, the horizontal dimensions of the chip are usually approximated as infinite in parasitic extraction. By the approximations, either the free-space Green's function is directly used or the radial symmetric property is exploited to simplify the free-space Green's function with the Hankel transform [47, 80]. To analyze the temperature distribution of the chip, the real horizontal dimensions of the chip must be taken into consideration. Therefore, boundary conditions must be properly imposed on the four sidewalls of the chip to model heat insulation or heat transfer between the sidewalls and the ambient environment [78]. For the purpose of substrate coupling analysis, the Green's function for the Poisson's equation has been derived for heterogeneous dielectric materials; however, zero potential and zero potential gradient are assumed for the top and bottom surfaces of the chip. Furthermore, numerical stability problem may occur in calculating the Green's function [48, 77]. In thermal analysis, general heat-convection boundary conditions must be imposed on the top and bottom surfaces of the chip.

1.3 Thesis Organization

This dissertation includes mainly three chapters. Chapter II is concentrated on the derivation of fully analytical temperature solutions to the steady-state heat conduction equation, particularly the derivation of the multilayer heat conduction Green's function. Chapter III is focused on the computation of the temperature solutions by $\mathcal{O}(n \lg n)$ algorithms and mainly introduces the fast thermal analysis method called LOTAGre. Chapter V is focused on the optimization of the chip temperature distribution in the cell placement stage and introduces the optimal power budget model to augment the Capo placement tool with thermal optimization capability. In addition to the three major chapters, this dissertation briefly introduces the modeling of interconnect temperature distribution in chapter IV.

1.3.1 Derivation of Homogeneous and Inhomogeneous Temperature Solutions

Chapter II derives fully analytical temperature solutions to the steady-state heat conduction equation. In the derivation, the temperature distribution of the chip is separated into two parts: the homogeneous temperature distribution attributed to the ambient temperatures only, and the inhomogeneous temperature distribution attributed to the heat sources inside the chip only. To solve the inhomogeneous temperature distribution, the chapter applies the multilayer heat conduction Green's function. Various boundary conditions are considered based on a general multilayer chip structure that consists of heterogeneous heat conduction materials. The chapter also derives a fully analytical solution to the homogeneous temperature distribution, which was traditionally neglected because the chip was assumed to be surrounded by a uniform ambient temperature [21, 78]. The assumption is inaccurate because temperature gradients at different boundaries of the chip are dissimilar and heat flow from different surfaces of the chip to the outer environment is unbalanced. In addition, the chapter introduces the s-domain Green's function for the time-dependent heat conduction equation. By the s-domain heat conduction Green's function, the thermal transfer impedance between any two interested regions of the chip can be computed so that compact thermal models can be established for the chip.

1.3.2 Computation of Homogeneous and Inhomogeneous Solutions

Chapter III presents fast algorithms for computing the homogeneous and inhomogeneous temperature solutions. This chapter introduces an $\mathcal{O}(n \lg n)$ chip-level thermal analysis method based on the multilayer heat conduction Green's function, which is named LOTAGre. Here, n indicates that there are n heat source blocks and that the temperatures of n blocks need to be evaluated. Traditional Green's function-based methods compute the temperature distribution of the chip by the following process: first, the methods model heat sources as discrete blocks; next, the methods compute the temperatures of the observed regions by the sum of the products of the heat source power densities and the Green's function values [21, 78]. The computing process can be abstracted as a matrix-vector product operation on an $n \times n$ matrix and an n-dimensional vector. Therefore, the traditional Green's function-based methods need $\mathcal{O}(n^2)$ computations to obtain the required temperatures. Such quadratic increase of the computational time becomes intolerable, especially when the methods are to be used in an inner loop for many iterations. Several techniques have been proposed to reduce the time complexity of the Green's function-based methods for parasitic exaction. In [47], FastCap used a multipole accelerated technique to compute the dense matrix-vector product in $\mathcal{O}(n)$ time. In [51], an $\mathcal{O}(n \lg n)$ precorrected-FFT technique was proposed, which was instead faster in many cases than the $\mathcal{O}(n)$ multipole accelerated technique. To speed up the dense matrix-vector production operation, both the techniques exploit the multipole expansion of the free-space Green's function based on the spherical harmonics. In comparison, LOTAGre introduced in the chapter is an $\mathcal{O}(n \lg n)$ method and meantime uses the multilayer heat conduction Green's function. Because of the application of the eigen-expansion technique and the use of orthogonal trigonometric functions, LOTAGre employs the discrete cosine transform (DCT) and the inverse discrete cosine transform (IDCT) to accelerate thermal simulation for ULSI chips. The final portion of the chapter presents experimental results to demonstrate the accuracy, speed, and scalability of LOTAGre.

1.3.3 Thermal Optimization in Cell Placement

Chapter V is focused on the optimization of the temperature distribution of the chip in the cell placement stage. In the ULSI chip physical design flow, the cell placement stage determines the module locations and significantly impacts the temperature distribution of the chip. The cell placement stage was traditionally intended to minimize the total interconnect length. However, the temperature distribution of the chip recently became very important, and in the literature, several thermal optimization methods were proposed for cell placement. Thermal simulation is an obstacle to thermal optimization at the cell placement stage. To optimize the temperature distribution of a chip, a large number of thermal simulations must be run across the entire parameter space. With LOTAGre to accelerate thermal simulation, thermal optimization becomes much faster at the cell placement stage. Importantly, this chapter introduces an optimal power budget model with the use of LOTAGre. Stipulated by several design constraints, the optimal power budget model determines the optimal allocation of heat sources to different regions of the chip to reduce the number of hot-spots inside the chip. The chapter describes the procedure to integrate the optimal power budget model into the Capo placement tool to perform thermal optimization.

In addition to the aforementioned three major chapters, chapter IV briefly addresses the interconnect thermal issue. Interconnect temperature impacts an interconnect wire primarily in two aspects. One is interconnect timing, as the variation of the interconnect temperature causes the variation of the interconnect resistivity and concurrently the variation of delay in propagating signals through the interconnect wire. The other is interconnect electromigration, as the MTTF of an interconnect wire decreases exponentially with the increase of its temperature [13]. Therefore, it is necessary to accurately determine the temperature distribution of an interconnect wire. This chapter presents an interconnect temperature distribution model and an efficient $\mathcal{O}(n)$ approach to calculate the temperature distribution of an interconnect wire.

Finally, chapter VI provides the concluding remarks and summarizes the major results and contributions of this dissertation. Possible future research directions to address the chip-level thermal issues are also suggested in the chapter.

CHAPTER II

Derivation of Homogeneous and Inhomogeneous Temperature Solutions

2.1 Steady-State Heat Conduction Problem

2.1.1 Steady-State Heat Conduction Equation

Inside a chip, temperature and power density are two interrelated physical quantities. The intensify of the power densities of the devices raises their temperatures, and rising temperatures also strengthen the power densities of the devices because of the increase of sub-threshold currents. In IC thermal analysis, there are two main methodologies for addressing the coupling between temperature and power density. One method solves the time-dependent heat conduction equation at discrete-time steps: at each time step, the method imposes the known power densities and solves the time-dependent heat conduction equation for the temperatures within the chip; then this method uses the obtained temperatures to estimate the power densities at the next time step by either transient circuit simulation or from circuit power models [27, 76]. The other method decomposes IC thermal analysis into two steps. First, this method applies circuit power models to estimate the power densities of the devices based on their initial temperatures. Next, it imposes the estimated power densities and solves the steady-state heat conduction equation to update the temperatures of the devices. It then repeats the previous two steps until convergence

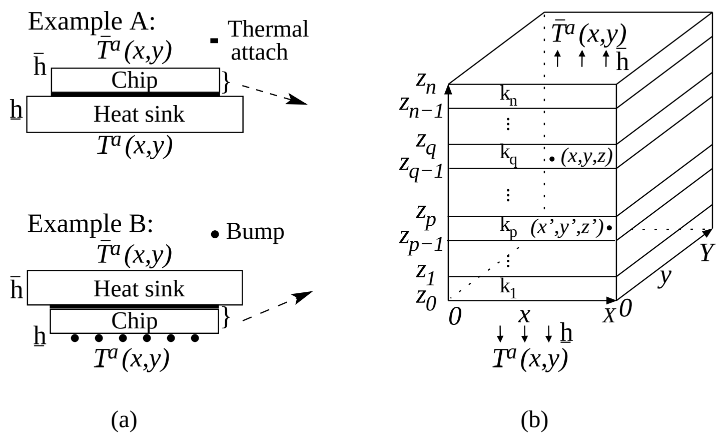


Figure 2.1: Illustration of multilayer thermal model: (a) two chip examples and (b) multilayer thermal model.

[22, 21]. Because the power densities of the devices highly depend on the transistor switching patterns, the latter method is much more efficient in ULSI chip design [21]. Hence, this dissertation work is based primarily on the steady-state heat conduction equation.

In thermal analysis, a chip can be abstracted as a multilayer thermal (MLT) model, as shown in Fig.2.1(b). On the left, Fig.2.1(a) shows two chip examples: example A, which has wire-bonding packaging, and example B, which has flip-chip packaging [26, 50]. The MLT model is capable of modeling chips with either packaging. Traditional Green's function-based methods used only the SLT model, i.e., an MLT model with only one layer. Geometrically, the SLT model includes only the active region of the chip and needs to approximate the other regions of the chip as well as the packaging materials by the heat transfer rates on the top and bottom surfaces of the chip [22]. In contrast, the MLT model can include the geometry of the entire chip as well as the packaging. Hence, this dissertation uses the MLT model shown in Fig.2.1(b).

The temperature distribution of a chip can be solved from the 3-D steady-state heat conduction equation, which in the Cartesian coordinate system is given by

$$\nabla \cdot \left[k\left(z\right)\nabla T\left(x,y,z\right)\right] = -f\left(x,y,z\right) \tag{2.1}$$

where T denotes temperature (in kelvin or K); the small letter k denotes material thermal conductivity (in W/m K); and f denotes power density (in W/m^3). Given a multilayer chip, the thermal conductivity k is modeled by a piecewise constant function, with k in each layer being a constant value. As shown in Fig.2.1(b), k is only z-axis dependent, with $k\left(z\right) = k_m$ for $z_{m-1} < z < z_m$ and $1 \leq m \leq n$.

Traditional Green's function-based methods specified the top ambient temperature $\bar{T}^a\left(x,y\right)$ and the bottom ambient temperature $\underline{T}^a\left(x,y\right)$ to take the same constant value. In reality, $\bar{T}^a$ and $\underline{T}^a$ may differ considerably and have large spatial variations in the $x - y$ plane, because of the temperature gradients inside the chip and the imbalance of heat flow from the different surfaces of the chip to the outer environment. To be accurate, in this dissertation $\bar{T}^a$ and $\underline{T}^a$ is represented as 2-D functions. Furthermore, the ambient environment surrounding the four sidewalls of the chip is assumed to have the same temperature. This sidewall ambient temperature is chosen as the reference, i.e., T in (2.1), $\bar{T}^a$ and $\underline{T}^a$ are temperature differences from the sidewall ambient temperature.

2.1.2 Heat Conduction Boundary Conditions

Three sets of boundary conditions (BCs) are specified for the heat conduction equation (2.1): BCs for the four sidewalls of the chip, named sidewall BCs; BCs for the horizontal inner interfaces between adjacent layers, named inter-layer BCs; and BCs for the top and bottom surfaces of the chip, named top-bottom BCs. Details are given below.

Sidewall BCs

If the four sidewalls of the chip, i.e., $x = 0, X$ and $y = 0, Y$, are insulated from the ambient environment, the Neumann's sidewall BCs are specified:

$$\begin{aligned} \left.\frac{\partial T(x,y,z)}{\partial x}\right|_{x=0,X} &= 0 \\ \left.\frac{\partial T(x,y,z)}{\partial y}\right|_{y=0,Y} &= 0. \end{aligned} \tag{2.2}$$

If the four sidewalls of the chip remain at a specific sidewall ambient temperature, the Dirichlet's sidewall BCs are specified:

$$\begin{aligned} T(x,y,z)|_{x=0,X} &= 0 \\ T(x,y,z)|_{y=0,Y} &= 0 \end{aligned} \tag{2.3}$$

where both the right-hand sides are zero because the sidewall ambient temperature is chosen as the reference.

Here capital letters X and Y denote the dimensions of the chip along the x and y axes, respectively. The chip vertical dimension is specified by the interval $[z_0, z_n]$. First, the Neumann's sidewall BCs (2.2) are imposed to solve the steady-state heat conduction equation.

Inter-layer BCs

At any horizontal inner interface z_m, for $0 < m < n$, inter-layer BCs are specified to ensure the continuity of temperature, described by (2.4a), and the continuity of per unit-area heat flux through the interface, described by (2.4b):

$$T(x,y,z_{m+}) = T(x,y,z_{m-}) \tag{2.4a}$$

$$k_{m+1}\left.\frac{\partial T(x,y,z)}{\partial z}\right|_{z=z_{m+}} = k_m\left.\frac{\partial T(x,y,z)}{\partial z}\right|_{z=z_{m-}}. \tag{2.4b}$$

Top-bottom BCs

On the top and bottom surfaces of the chip, the phenomenon of heat transfer with the ambient environment is described by the heat convection BCs:

$$\text{(2.5a)} \qquad k_1 \left. \frac{\partial T\left(x,y,z\right)}{\partial z} \right|_{z=z_0} - \underline{h} T\left(x,y,z_0\right) = -\underline{h}\,\underline{T}^a\left(x,y\right)$$

$$\text{(2.5b)} \qquad -k_n \left. \frac{\partial T\left(x,y,z\right)}{\partial z} \right|_{z=z_n} - \bar{h} T\left(x,y,z_n\right) = -\bar{h}\bar{T}^a\left(x,y\right).$$

Here, $\underline{h}$ (or $\bar{h}$) is the heat transfer rate between the bottom (or top) surface z_0 (or z_n) of the chip and the ambient environment, with units W/(m^2 K).

Given a power density distribution $f\left(x,y,z\right)$, the temperature distribution of the chip, $T\left(x,y,z\right)$, can be solved from (2.1), under the three sets of heat conduction BCs. Grid-based methods often discretize the Laplacian operator by the second-order central FD scheme to form a resistive network [65]. Even if sophisticated linear solvers are used to solve the resultant system of equations, the large problem size makes the solution process very expensive. Allocating large amounts of grids to those regions that are of little interest renders the grid-based methods inefficient in performing chip-level thermal analysis. In contrast, Green's function-based methods avoid such costly modeling and improve thermal simulation time by orders of magnitude [21, 67, 78]. This dissertation introduces a multilayer heat conduction Green's function-based fast thermal analysis method named LOTAGre [73, 75].

2.2 Homogeneous and Inhomogeneous Temperature Solutions

Because the top-bottom BCs in (2.5) are inhomogeneous, the temperature distribution of the chip is separated into two parts: one is a homogeneous solution, denoted by T^h, which satisfies the Laplace's equation and inhomogeneous top-bottom BCs; and the other is an inhomogeneous solution, denoted by T^i, which satisfies the

Poisson's equation and homogeneous top-bottom BCs. Therefore, at a given chip location (x, y, z), the temperature is represented by the superposition of the homogeneous temperature $T^h(x, y, z)$ and the inhomogeneous temperature $T^i(x, y, z)$:

$$T(x, y, z) = T^h(x, y, z) + T^i(x, y, z).$$

2.2.1 Homogeneous Temperature Solution

The homogeneous solution T^h satisfies the homogeneous heat conduction equation, which is obtained by setting the right-hand side of (2.1) to zero. Meanwhile, T^h satisfies the three sets of heat conduction BCs given in Section 2.1.2. In summary, the complete equations that govern T^h are described by

$$\nabla \cdot \left[k(z) \nabla T^h(x, y, z)\right] = 0 \tag{2.6a}$$

$$k_1 \frac{\partial T^h}{\partial z} - \underline{h} T^h \bigg|_{z=z_0} = -\underline{h}\,\underline{T}^a(x, y) \tag{2.6b}$$

$$-k_n \frac{\partial T^h}{\partial z} - \bar{h} T^h \bigg|_{z=z_n} = -\bar{h}\bar{T}^a(x, y) \tag{2.6c}$$

$$T^h\big|_{z=z_{m+}} = T^h\big|_{z=z_{m-}} \tag{2.6d}$$

$$k_{m+1} \frac{\partial T^h}{\partial z}\bigg|_{z=z_{m+}} = k_m \frac{\partial T^h}{\partial z}\bigg|_{z=z_{m-}} \tag{2.6e}$$

$$\frac{\partial T^h}{\partial x}\bigg|_{x=0,X} = 0, \quad \frac{\partial T^h}{\partial y}\bigg|_{y=0,Y} = 0. \tag{2.6f}$$

2.2.2 Inhomogeneous Temperature Solution

The inhomogeneous solution T^i satisfies the heat conduction equation (2.1) and all three sets of heat conduction BCs, except that the top-bottom BCs (2.5) are replaced by homogeneous top-bottom BCs. In other words, the right-hand sides of (2.5) are set to zero. The complete equations that govern T^i, except the sidewall and inter-layer BCs, are given by

$$\nabla \cdot \left[k\left(z\right) \nabla T^{i}\left(x,y,z\right) \right] = -f\left(x,y,z\right) \tag{2.7a}$$

$$k_{1}\frac{\partial T^{i}\left(x,y,z\right)}{\partial z}\bigg|_{z=z_{0}} - \underline{h}T^{i}\left(x,y,z_{0}\right) = 0 \tag{2.7b}$$

$$-k_{n}\frac{\partial T^{i}\left(x,y,z\right)}{\partial z}\bigg|_{z=z_{n}} - \bar{h}T^{i}\left(x,y,z_{n}\right) = 0. \tag{2.7c}$$

The inhomogeneous solution T^i can be obtained by using the Green's function of (2.7), i.e., the heat conduction Green's function, denoted by $G(x,y,z|x',y',z')$ here. G corresponds to the temperature distribution of the chip when the power density distribution $f\left(x,y,z\right)$ is a Dirac delta function $\delta(x-x',y-y',z-z')$, i.e., a unit-strength heat source at the location (x',y',z') of the chip. The complete equations that govern G, except the sidewall BCs, are given by

$$\nabla \cdot \left[k\left(z\right) \nabla G\left(x,y,z|x',y',z'\right) \right] = -\delta\left(x-x',y-y',z-z'\right) \tag{2.8a}$$

$$G|_{z=z_{m+}} = G|_{z=z_{m-}} \tag{2.8b}$$

$$k_{m+1}\frac{\partial G}{\partial z}\bigg|_{z=z_{m+}} = k_{m}\frac{\partial G}{\partial z}\bigg|_{z=z_{m-}} \tag{2.8c}$$

$$k_{1}\frac{\partial G}{\partial z} - \underline{h}G\bigg|_{z=z_{0}} = 0 \tag{2.8d}$$

$$-k_{n}\frac{\partial G}{\partial z} - \bar{h}G\bigg|_{z=z_{n}} = 0. \tag{2.8e}$$

Then the inhomogeneous solution $T^i\left(x,y,z\right)$ can be represented by the spatial convolution of the power density distribution f with G:

$$T^{i}\left(x,y,z\right) = \int_{\mathcal{V}} G\left(x,y,z|x',y',z'\right) f\left(x',y',z'\right) dx'dy'dz' \tag{2.9}$$

where $\mathcal{V}$ denotes the entire volume space of the simulated chip.

Assuming the homogeneous solution T^h being zero, traditional Green's function-based methods mainly considered the inhomogeneous solution T^i [21, 78]. In this dissertation, the homogeneous solution T^h is considered and general 2-D functions are used to model the ambient temperatures at the top and bottom surfaces of the simulated chip. Since the homogeneous solution T^h and the inhomogeneous solution T^i are independent and the former does not depend on the power density distribution f, T^h should be computed only once for a given ambient condition.

The Green's function for the Poisson's equation has been discussed in the literature. In [28], the Green's function for the 2-D Poisson's equation was considered under various types of BCs and geometrical configurations. In [80], the Green's function for the 3-D Poisson's equation was derived for parasitic extraction, under the assumption that the chip horizontal dimensions were infinite. For substrate modeling, in [48] the Green's function was derived with the Neumann's sidewall BCs imposed and with the potential or potential gradient at the top and bottom surfaces of the chip set to zero; and in [77] the numerical stability issue was further discussed. In thermal analysis, however, general heat convection BCs need to be imposed on the top and bottom surfaces of the chip. Considering one homogeneous heat conduction material, [21] derived the Green's function by assuming a heat insulation BC on the chip top surface, and [78] derived the Green's function under the Neumann's sidewall BCs. To consider heterogeneous materials, the MLT model and the multilayer heat conduction Green's function should be used.

In the following, fully analytical formulas are derived for the homogeneous solution T^h, the inhomogeneous solution T^i, and the multilayer heat conduction Green's function, including the s-domain version.

2.3 Derivation of Homogeneous Temperature Solution

To obtain the homogeneous solution T^h from (2.6), the eigen-expansion technique is used [14]. Consider the Neumann's sidewall BCs (2.6f), which approximate that there is no heat transfer between the simulated chip and the surrounding environment via the four sidewalls of the chip because the thickness of the chip is much smaller than the horizontal dimensions of the chip. Consequently, orthogonal cosine functions are chosen as eigenfunctions.

The domain of the homogeneous solution $T^h\left(x,y,z\right)$ was initially limited to the entire chip volume space $\mathcal{V}$. To obtain an eigen-expansion of $T^h\left(x,y,z\right)$, the domain of $T^h\left(x,y,z\right)$ is expanded from space $\mathcal{V}$ to the entire 3-D space, and $T^h\left(x,y,z\right)$ is expanded to a periodic even function of x of period $2X$ as well as a periodic even function of y of period $2Y$. Similarly, this even periodic expansion is applied to the top and bottom ambient temperature functions $\bar{T}^a\left(x,y\right)$ and $\underline{T}^a\left(x,y\right)$. The three eigen-expansions are given by

$$T^h\left(x,y,z\right)=\sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\phi_{ij}\left(x,y\right)t_{ij}^h\left(z\right) \tag{2.10a}$$

$$\bar{T}^a\left(x,y\right)=\sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\phi_{ij}\left(x,y\right)\bar{t}_{ij}^a \tag{2.10b}$$

$$\underline{T}^a\left(x,y\right)=\sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\phi_{ij}\left(x,y\right)\underline{t}_{ij}^a. \tag{2.10c}$$

Here t_{ij}^h, $\bar{t}_{ij}^a$ and $\underline{t}_{ij}^a$ are eigen-expansion coefficients, and $\phi_{ij}\left(x,y\right)$ is the eigen-function: $\phi_{ij}\left(x,y\right)=\cos\left(\frac{i\pi x}{X}\right)\cos\left(\frac{j\pi y}{Y}\right)$. Eigen-expansions in (2.10a) ensure that T^h satisfies the Neumann's sidewall BCs (2.6f).

Insert (2.10) into (2.6) and eliminate the $\phi_{ij}\left(x,y\right)$ terms in those equations. The

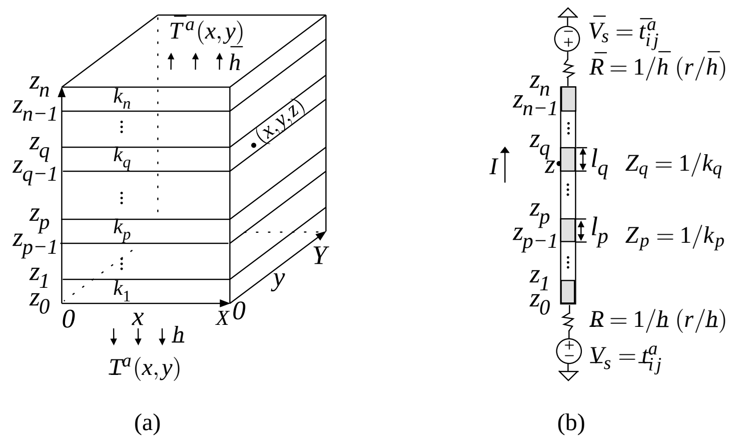


Figure 2.2: Equivalent circuit for deriving eigen-expansion coefficient t^h_{ij}: (a) MLT model for 3-D governing equation (2.6), with top and bottom ambient temperature functions imposed and power density distribution set to zero; and (b) an equivalent circuit that describes 1-D governing equation (2.11).

complete equations that govern the eigen-expansion coefficient t^h_{ij} are obtained:

$$\frac{d^2 t^h_{ij}(z)}{dz^2} - \gamma^2_{ij} t^h_{ij}(z) = 0 \tag{2.11a}$$

$$t^h_{ij}(z_{m+}) = t^h_{ij}(z_{m-}) \tag{2.11b}$$

$$k_{m+1} \left. \frac{dt^h_{ij}(z)}{dz} \right|_{z=z_{m+}} = k_m \left. \frac{dt^h_{ij}(z)}{dz} \right|_{z=z_{m-}} \tag{2.11c}$$

$$k_1 \left. \frac{dt^h_{ij}(z)}{dz} \right|_{z=z_0} - \underline{h} t^h_{ij}(z_0) = -\underline{h}\underline{t}^a_{ij} \tag{2.11d}$$

$$-k_n \left. \frac{dt^h_{ij}(z)}{dz} \right|_{z=z_n} - \bar{h} t^h_{ij}(z_n) = -\bar{h}\bar{t}^a_{ij} \tag{2.11e}$$

where $\gamma_{ij} = \sqrt{\frac{i^2\pi^2}{X^2} + \frac{j^2\pi^2}{Y^2}}$. Accordingly, the homogeneous solution T^h can be computed in the following way: given the 2-D top and bottom ambient temperature functions $\bar{T}^a(x,y)$ and $\underline{T}^a(x,y)$, first calculate their eigen-expansion coefficients $\bar{t}^a_{ij}$ and $\underline{t}^a_{ij}$ based on formulas (2.10b) and (2.10c); then solve the eigen-expansion coefficient $t^h_{ij}(z)$ from (2.11). Finally, calculate $T^h(x,y,z)$ based on (2.10a).

The eigen-expansion coefficient t^h_{ij} is derived in the following.

2.3.1 Eigen-expansion Coefficient t_{ij}^{h} $(i = j = 0)$

When $i = j = 0$, γ_{ij} becomes 0, then the governing equations in (2.11) can be shown to be equivalent to the circuit equations for an n-section non-uniform line conductor of per unit-length (PUL) conductance $k\left(z\right)$. Each section of the line conductor corresponds to one layer in the MLT model in Fig.2.2(a) and has a length that equals the thickness of that layer. The two ends of the line conductor are terminated by two resistors of resistances $\bar{R} = 1/\bar{h}$ and $\underline{R} = 1/\underline{h}$, and the two resistors are driven by two voltage sources of magnitudes $\bar{V}_s = \bar{t}_{ij}^{a}$ and $\underline{V}_s = \underline{t}_{ij}^{a}$. The equivalent line conductor circuit is shown in Fig.2.2(b). According to the shown equivalent circuit, the eigen-expansion coefficient $t_{ij}^{h}\left(z\right)$ corresponds to the voltage at the location z on the line (assume that location z is in the q-th section of the line conductor); (2.11a) corresponds to the Kirchhoff's current law at that location; (2.11b) and (2.11c) are equivalent to the current and voltage continuity conditions at the interface between the m-th and the $(m+1)$-th line sections. The last two equations in (2.11) correspond to the circuit equations that govern the two ends of the line conductor.

Therefore, by solving the voltage at the location z, the eigen-expansion coefficient $t_{00}^{h}\left(z\right)$ is obtained:

(2.12)
$$t_{00}^{h}\left(z\right) = \bar{t}_{00}^{a}\bar{H}_{00}^{a}\left(z\right) + \underline{t}_{00}^{a}\underline{H}_{00}^{a}\left(z\right)$$

where

(2.13a)
$$\bar{H}_{00}^{a}\left(z\right) = \frac{\underline{Z}_q + Z_q\left(z - z_{q-1}\right)}{\underline{Z}_q + Z_q l_q + \bar{Z}_q}$$

(2.13b)
$$\underline{H}_{00}^{a}\left(z\right) = \frac{\bar{Z}_q + Z_q\left(z_q - z\right)}{\underline{Z}_q + Z_q l_q + \bar{Z}_q}.$$

The symbols used previously are explained hereafter. In this dissertation, when a non-uniform line (either a line conductor or a transmission line) is used in an equivalent circuit, for the m-th line section, $\underline{Z}_m$ denotes the input impedance seen from the bottom boundary of that section toward the bottom side of the equivalent circuit; $\bar{Z}_m$ denotes the input impedance seen from the top boundary of that section toward the top side of the circuit; Z_m denotes the characteristic impedance of that section, with $Z_m = 1/k_m$; and l_m is the length of that section. There are two special cases: $\bar{Z}_0$ denotes the input impedance seen from location $z = z_0$ to the top side of the circuit, and $\underline{Z}_{n+1}$ denotes the input impedance seen from location $z = z_n$ to the bottom side of the circuit. For the equivalent line conductor circuit shown in Fig.2.2(b), $\bar{Z}_q = \bar{R} + \sum_{m=q+1}^{n} Z_m$ and $\underline{Z}_q = \underline{R} + \sum_{m=1}^{q-1} Z_m$.

2.3.2 Eigen-expansion Coefficient t_{ij}^h $(i + j > 0)$

When $i + j > 0$, a similar circuit equivalence that will facilitate solving t_{ij}^h from (2.11) can be established by comparing (2.11a) to the transmission line equations

$$\begin{aligned} \frac{dV(z)}{dz} &= -(R + sL)\, I(z) \\ \frac{dI(z)}{dz} &= -(G + sC)\, V(z) \end{aligned}$$

or in an alternative form $\frac{d^2V(z)}{dz^2} - \gamma^2 V(z) = 0$. The governing equations of t_{ij}^h in (2.11) can be shown to be equivalent to the circuit equations for an n-section non-uniform transmission line (TL) of propagation constant $\gamma = \sqrt{(R + sL)(G + sC)}$ and characteristic impedance $Z = \sqrt{\frac{R+sL}{G+sC}}$. In the equivalent TL circuit, each line section corresponds to one layer in the MLT model and has a length that equals the thickness of that layer. The PUL parameters of the m-th TL section, R, L, C, and G satisfy $\sqrt{RG} = \gamma_{ij}$, $\sqrt{R/G} = Z_m = 1/k_m$, and $L = C = 0$. The two ends of the TL are terminated by two resistors of resistances $\bar{R} = \gamma/\bar{h}$ and $\underline{R} = \gamma/\underline{h}$, which are driven by the same voltage sources as those in the line conductor circuit. Consequently, the same circuit diagram as that for the line conductor circuit is used to illustrate the equivalent TL circuit, as shown by Fig.2.2(b) again. Note that the two terminating resistors choose the resistance values enclosed in the parentheses.

According to Fig.2.2(b), the eigen-expansion coefficient $t_{ij}^{h}(z)$ corresponds to the voltage at the location z in the q-th TL section. Therefore, by solving the voltage at that location, $t_{ij}^{h}(z)$ is obtained:

$$t_{ij}^{h}(z) = \bar{t}_{ij}^{a}\bar{H}_{ij}^{a}(z) + \underline{t}_{ij}^{a}\underline{H}_{ij}^{a}(z) \tag{2.14}$$

where

$$\begin{aligned}\bar{H}_{ij}^{a}(z) &= \frac{\underline{Z}_{n+1}}{\bar{\xi}\left(\underline{Z}_{n+1}+\bar{R}\right)}\frac{\underline{Z}_q\cosh\gamma\underline{l}_q + Z_q\sinh\gamma\underline{l}_q}{\underline{Z}_q\cosh\gamma l_q + Z_q\sinh\gamma l_q}\\ \underline{H}_{ij}^{a}(z) &= \frac{\bar{Z}_0}{\underline{\xi}\left(\underline{R}+\bar{Z}_0\right)}\frac{\bar{Z}_q\cosh\gamma\bar{l}_q + Z_q\sinh\gamma\bar{l}_q}{\bar{Z}_q\cosh\gamma l_q + Z_q\cosh\gamma l_q}.\end{aligned}$$

Here $\bar{l}_q = z_q - z$, $\underline{l}_q = z - z_{q-1}$, and

$$\begin{aligned}\bar{\xi} &= \prod_{m=q+1}^{n}\left(\cosh\gamma l_m + \frac{Z_m}{\underline{Z}_m}\sinh\gamma l_m\right)\\ \underline{\xi} &= \prod_{m=1}^{q-1}\left(\cosh\gamma l_m + \frac{Z_m}{\bar{Z}_m}\sinh\gamma l_m\right).\end{aligned}$$

Those input impedances $\underline{Z}$'s and $\bar{Z}$'s have recurrence formulas: for one TL section in a non-uniform TL, the input impedance at its one boundary, denoted by Z_{in}, has a recurrence formula:

$$Z_{in} = ZC\frac{ZL + ZC\tanh\gamma L}{ZC + ZL\tanh\gamma L} \tag{2.16}$$

where ZL is the load impedance at the other boundary, ZC is the characteristic impedance of this TL section, and L is the length of the section.

In the previous procedure, the homogeneous solution T^h has been derived under the Neumann's sidewall BCs (2.2). When the Dirichlet's sidewall BCs (2.3) are imposed, the same procedure as the previous can be followed to derive the corresponding homogeneous solution after the eigenfunction is changed to $\phi_{ij}(x,y) = \sin\left(\frac{i\pi x}{X}\right)\sin\left(\frac{j\pi y}{Y}\right)$.

2.4 Derivation of Inhomogeneous Temperature Solution

To obtain the inhomogeneous solution T^i, the heat conduction Green's function G should be solved from its governing equations (2.8) under the BCs imposed. In the following, G is derived by using the same procedure as that which derives the homogeneous solution T^h, i.e., employing the eigen-expansion technique and the transmission line theory. First, impose the Neumann's sidewall BCs (2.2); therefore, G satisfies the following sidewall BCs:

$$\left.\frac{\partial G\left(x,y,z|x',y',z'\right)}{\partial x}\right|_{x=0,X} = 0 \tag{2.17a}$$

$$\left.\frac{\partial G\left(x,y,z|x',y',z'\right)}{\partial y}\right|_{y=0,Y} = 0. \tag{2.17b}$$

Similarly, the even periodic expansion is applied to the heat conduction Green's function. The following eigen-expansion of G results:

$$G\left(x,y,z|x',y',z'\right) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\phi_{ij}\left(x,y\right)G_{ij}\left(z|x',y',z'\right) \tag{2.18}$$

where the eigenfunction remains being $\phi_{ij}\left(x,y\right) = \cos\left(\frac{i\pi x}{X}\right)\cos\left(\frac{j\pi y}{Y}\right)$. The above eigen-expansion ensures that G satisfies the Neumann's sidewall BCs in (2.17).

Insert (2.18) into (2.8), multiply the two sides of (2.8a) by $\phi_{ij}\left(x,y\right)$, and integrate over the x and y dimensions of the chip. Then (2.19a) results, due to the orthogonality of eigenfunctions [14]. Simplifying the remaining equations in (2.8) leads to

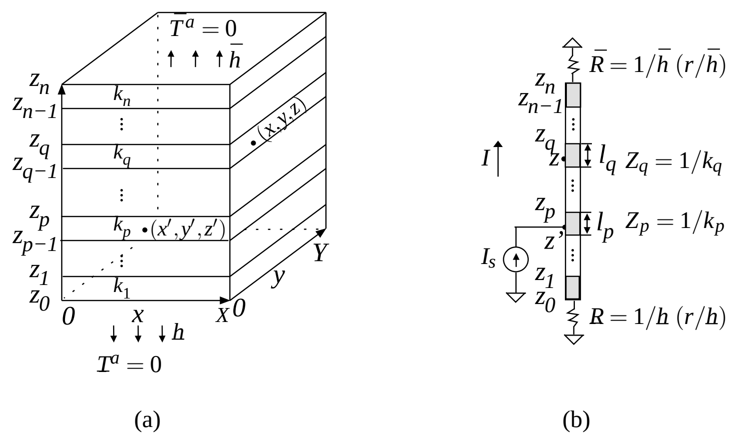


Figure 2.3: Equivalent circuit for deriving eigen-expansion coefficient G_{ij}: (a) MLT model for 3-D governing equation (2.8), with Dirac delta heat source imposed at location (x', y', z') and ambient temperatures set to zero; and (b) an equivalent circuit that describes 1-D governing equation (2.19), (small letter z denotes a location, while capital letter Z denotes TL characteristic impedance).

the complete equations that govern the eigen-expansion coefficient G_{ij}:

$$\text{(2.19a)} \qquad \frac{d^2 G_{ij}(z|x', y', z')}{dz^2} - \gamma_{ij}^2 G_{ij}(z|x', y', z') = -\frac{c_{ij}\phi_{ij}(x', y')\,\delta(z - z')}{k(z)}$$

$$\text{(2.19b)} \qquad G_{ij}(z_{m+}|x', y', z') = G_{ij}(z_{m-}|x', y', z')$$

$$\text{(2.19c)} \qquad k_{m+1}\left.\frac{dG_{ij}(z|x', y', z')}{dz}\right|_{z=z_{m+}} = k_m\left.\frac{dG_{ij}(z|x', y', z')}{dz}\right|_{z=z_{m-}}$$

$$\text{(2.19d)} \qquad k_1\left.\frac{dG_{ij}(z|x', y', z')}{dz}\right|_{z=z_0} - \underline{h}G_{ij}(z_0|x', y', z') = 0$$

$$\text{(2.19e)} \qquad -k_n\left.\frac{dG_{ij}(z|x', y', z')}{dz}\right|_{z=z_n} - \bar{h}G_{ij}(z_n|x', y', z') = 0.$$

Here γ_{ij} remains being $\gamma_{ij} = \sqrt{\frac{i^2\pi^2}{X^2} + \frac{j^2\pi^2}{Y^2}}$, and $c_{ij} = 2^{2-\delta_{i0}-\delta_{j0}}/XY$, where δ_{i0} and δ_{j0} are Kronecker deltas. Once the eigen-expansion coefficient G_{ij} is solved from (2.19), G can be obtained from (2.18). Since G_{ij}'s governing equations (2.19) have some similarities to (2.11), the transmission line theory is employed again to derive G_{ij}.

2.4.1 Eigen-expansion Coefficient G_{ij} $(i = j = 0)$

Let $i = j = 0$, and γ_{ij} becomes 0. Then the governing equations (2.19) can be shown to be equivalent to the circuit equations for an n-section non-uniform line conductor of PUL conductance $k(z)$. The two ends of the line are terminated by two resistors of resistances $\underline{R} = 1/\underline{h}$ and $\bar{R} = 1/\bar{h}$. Fig.2.3(b) shows this line conductor. In comparison, this line conductor differs in two aspects from the equivalent circuit in Fig.2.2(b) for deriving T^h: there is a current source input I_s of intensity $c_{00}\phi_{00}(x', y')$ at the source location z' in the p-th section of this line conductor, and there are no voltage sources at the two ends of this line conductor.

According to Fig.2.3(b) and (2.19), the eigen-expansion coefficient $G_{00}\left(z|x', y', z'\right)$ corresponds to the voltage at the target location z in the q-th section of this line conductor. Therefore, by solving the voltage at that location, G_{00} is obtained:

$$G_{00}(z|x', y', z') = c_{00}\phi_{00}(x', y')H_{00}(z|z') \tag{2.20}$$

where H_{00} is the transfer impedance from the source location z' to the target location z, given by

$$H_{00}(z|z') = \frac{\left[\underline{Z}_p + Z_p(z' - z_{p-1})\right]\left[\bar{Z}_q + Z_q(z_q - z)\right]}{\underline{Z}_p + Z_p l_p + \bar{Z}_q}. \tag{2.21}$$

Here the same symbols are used as in the previous equations.

2.4.2 Eigen-expansion Coefficient G_{ij} $(i + j > 0)$

When $i + j > 0$, an equivalent TL circuit can be constructed to derive G_{ij}. Compare (2.19a) to the transmission line equations:

$$\begin{aligned} \frac{dV(z)}{dz} &= -(R + sL)I(z) \\ \frac{dI(z)}{dz} &= -(G + sC)V(z) + I_s\delta(z - z') \end{aligned}$$

or in the form of

$$\frac{d^2V(z)}{dz^2} - \gamma^2 V(z) = -\gamma Z I_s \delta(z - z'). \tag{2.22}$$

It is evident that $G_{ij}(z|x', y', z')$ corresponds to the voltage at the location z in an n-section non-uniform TL of propagation constant $\gamma = \gamma_{ij}$ and characteristic impedance $Z = 1/k(z)$ when a current source input of intensity $I_s = \frac{c_{ij}}{\gamma_{ij}}\phi_{ij}(x', y')$ is imposed at the location z'. The PUL parameters of the m-th line section satisfy $\sqrt{RG} = \gamma_{ij}$, $\sqrt{R/G} = Z_m = 1/k_m$, $L = 0$, and $C = 0$. The equivalent TL circuit is shown by Fig.2.3(b) again. Note that the two terminating resistors choose the resistance values enclosed in the parentheses, i.e., $\bar{R} = \gamma/\bar{h}$ and $\underline{R} = \gamma/\underline{h}$.

Since $G_{ij}(z|x', y', z')$ corresponds to the voltage at the location z of the TL under a current source input I_s, $G_{ij}(z|x', y', z')$ is derived with the help of the transfer impedance between the source location z' in the p-th line section and the target location z in the q-th line section. The obtained G_{ij}, for $i + j > 0$, is given by

$$G_{ij}(z|x', y', z') = c_{ij}\phi_{ij}(x', y')H_{ij}(z|z') \tag{2.23}$$

where H_{ij} is the normalized transfer impedance from the location z' to the location z by the propagation constant γ, with $\gamma = \gamma_{ij} = \sqrt{\frac{i^2\pi^2}{X^2} + \frac{j^2\pi^2}{Y^2}}$. The normalized transfer impedance H_{ij} is given by

$$H_{ij}(z|z') = \frac{\xi\left(\underline{Z}_p \cosh \gamma \underline{l} + Z_p \sinh \gamma \underline{l}\right)\left(\bar{Z}_q \cosh \gamma \bar{l} + Z_q \sinh \gamma \bar{l}\right)}{Z_p(\underline{Z}_p + \bar{Z}_p)\cosh \gamma l_p + (Z_p^2 + \underline{Z}_p \bar{Z}_p)\sinh \gamma l_p} \tag{2.24}$$

where $\underline{l} = z' - z_{p-1}$, $\bar{l} = z_q - z$, and ξ is given by

$$\xi = \frac{Z_p \prod_{m=p}^{q-1} \bar{Z}_m}{\gamma \prod_{m=p+1}^{q}(\bar{Z}_m \cosh \gamma l_m + Z_m \sinh \gamma l_m)}. \tag{2.25}$$

To demonstrate the multilayer heat conduction Green's function derived here, the multilayer structure considered by Albers is taken as an example [5].

2.4.3 Surface Temperature Solution for Multilayer Structure

In [5], Albers gave a recursion relation for the steady-state surface temperature of a multilayer structure and showed that the recursion analytically agreed with Kokkas' solution for up to 3 layers [43]. Based on the previous multilayer Green's function, this section gives a surface temperature solution that agrees with Albers' recursion relation for an arbitrary number of layers. The procedure is described below.

The multilayer structure used by Albers becomes a special case of the MLT model in Fig.2.1(b) after let $\bar{h} = 0$, $\underline{h} = \infty$, $\bar{T}^a(x,y) = \underline{T}^a(x,y) = 0$, and $f(x,y,z) = P_0 u(x,y)\,\delta(z - z_n)$. Therefore, according to (2.9), (2.18), (2.20), and (2.23), the surface temperature is represented by

$$T(x,y,z_n) = P_0 \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} c_{ij} U(i,j)\,\phi_{ij}(x,y)\,H_{ij}(z_n|z_n) \tag{2.26}$$

where $U(i,j) = \int_0^X \int_0^Y u(x',y')\,\phi_{ij}(x',y')\,dx'dy'$.

Insert $\bar{Z}_n = \infty$ into (2.21). $H_{00}(z_n|z_n)$ is obtained:

$$H_{00}(z_n|z_n) = \underline{Z}_n + Z_n l_n = \sum_{m=1}^{n} \frac{l_m}{k_m}. \tag{2.27}$$

Insert $\underline{l} = l_n$ and $\bar{Z}_n = \infty$ into (2.24) and apply (2.16). $H_{ij}(z_n|z_n)$, for $i + j > 0$, is obtained:

$$H_{ij}(z_n|z_n) = \frac{1}{\gamma}\underline{Z}_{n+1} = \frac{1}{\gamma} Z_n \frac{\underline{Z}_n + Z_n \tanh\gamma l_n}{Z_n + \underline{Z}_n \tanh\gamma l_n}. \tag{2.28}$$

Note that since $\underline{h} = \infty$, $\underline{Z}_2 = Z_1 \tanh\gamma l_1$. Clearly, after a simple transformation $\tau_m = \underline{Z}_{m+1} k_m$, (2.26), (2.27) and (2.28), obtained by employing the multilayer heat conduction Green's function derived here, are the same as the analytical formulas derived by Albers [5]. For the numerical computation of the surface temperature, Kokkas presented extensive numerical results in [43] and discussed the convergence issue of the series expansion (2.26).

By integrating the eigen-expansion technique and the transmission line theory, this section derives the multilayer heat conduction Green's function, with the Neumann's sidewall BCs (2.2) imposed. With the multilayer heat conduction Green's function, the steady-state temperature distribution for a given arbitrary power density distribution can be computed. For example, this section derives a surface temperature solution that agrees with Albers' recursion relation for a multilayer structure. The same methodology can still be followed to derive the multilayer heat conduction Green's function under the other types of sidewall BCs. For example, consider using the Dirichlet's sidewall BCs (2.3), i.e., assume that the sidewall temperatures remain a constant value. After the eigenfunction is changed to $\phi_{ij}(x, y) = \sin\left(\frac{i\pi x}{X}\right)\sin\left(\frac{j\pi y}{Y}\right)$, the same form of equations as those in (2.19) result for G_{ij}. Therefore, the heat conduction Green's function under the Dirichlet's sidewall BCs can be derived in the same way as that described previously. [63] has presented a comprehensive library of eigenfunctions, which can be employed to derive the Green's function under other types of sidewall BCs.

In calculating the Green's function derived here, the relationship between the eigen-expansion coefficients and circuit transfer functions leads to the following observation: when the heat transfer rate $\underline{h}$ or $\bar{h}$ is zero or close to zero, the load impedances at the end sides of the equivalent circuits become infinite or too large to be represented in a floating-point number system; therefore, instead of using impedance formulations, using admittance functions in calculation can avoid numerical overflows. The following asymptotic estimation about H_{ij} can also be made by expanding the hyperbolic functions in the explicit formulas (2.24) and (2.25): as γ increases, $\bar{Z}$'s and $\underline{Z}$'s are close to some constant values according to (2.16); therefore, when $z \neq z'$, the trend of H_{ij} is dominated by the exponential decrease at a rate of

$e^{\gamma\left(\underline{l}+\bar{l}-\sum_{m=p}^{q} l_m\right)}$, and when $z = z'$, the decrease of H_{ij} is due to the $\frac{1}{\gamma}$ term in (2.25). In the special case that $z = z' = z_n$, H_{ij} has a concise form (2.28), by which H_{ij} can be efficiently computed.

Proceeding as previously, the following section derives the multilayer heat conduction Green's function for the time-dependent heat conduction equation.

2.4.4 s-domain Multilayer Heat Conduction Green's Function

The time-dependent heat conduction equation for the MLT model in Fig.2.1(b) is described by

$$\nabla \cdot \left[k\left(z\right)\nabla T\left(x,y,z,t\right)\right] - \rho\left(z\right)c\left(z\right)\frac{\partial T\left(x,y,z,t\right)}{\partial t} = -f\left(x,y,z,t\right) \tag{2.29}$$

where ρ is the density of the material and c is the specific heat. The multilayer heat conduction Green's function for (2.29), denoted by $\mathcal{G}(x,y,z,t|x',y',z')$, is the temperature solution to (2.29) under zero initial temperature distribution and zero ambient temperatures when a Dirac delta source is imposed as the power density distribution, i.e., $f\left(x,y,z,t\right) = \delta\left(x-x', y-y', z-z', t\right)$.

Consequently, in the s-domain the Laplace transform of $\mathcal{G}\left(x,y,z,t|x',y',z'\right)$, i.e., the s-domain heat conduction Green's function $\mathcal{G}\left(x,y,z,s|x',y',z'\right)$, satisfies

$$\nabla^2\mathcal{G} - \frac{s}{d\left(z\right)}\mathcal{G} = \frac{\delta\left(x-x', y-y', z-z'\right)}{k\left(z\right)} \tag{2.30}$$

where d is the material thermal diffusivity, with $d\left(z\right) = k\left(z\right)/\rho\left(z\right)c\left(z\right)$. Insert the eigen-expansion of $\mathcal{G}$,

$$\mathcal{G}\left(x,y,z,s|x',y',z'\right) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\phi_{ij}\left(x,y\right)\mathcal{G}_{ij}\left(z,s|x',y',z'\right), \tag{2.31}$$

into (2.30). After let $\gamma_{ij} = \sqrt{\frac{i^2\pi^2}{X^2} + \frac{j^2\pi^2}{Y^2} + \frac{s}{d(z)}}$, the same set of governing equations as those in (2.19) result for $\mathcal{G}_{ij}$. Therefore, the equivalent circuit in Fig.2.3(b) is employed again to derive the eigen-expansion coefficient $\mathcal{G}_{ij}$.

Note that since γ_{ij} is now z-axis dependent, circuit parameters in Fig.2.3(b) should be altered in the following way. For the m-th line section, its propagation constant and characteristic impedance are specified as $\gamma^{(m)} = \sqrt{\frac{i^2\pi^2}{X^2} + \frac{j^2\pi^2}{Y^2} + \frac{s}{d_m}}$ and $Z_m = \frac{1}{k_m\gamma^{(m)}}$, respectively. Here d_m is the thermal diffusivity of the material in the m-th layer. The intensity of the current source input and the resistances of the two terminating resistors are specified by $I_s = c_{ij}\phi_{ij}(x', y')$, $\bar{R} = 1/\bar{h}$, and $\underline{R} = 1/\underline{h}$, respectively.

The formulas for $\mathcal{G}_{ij}$ obtained here are similar to (2.23) and (2.24):

$$\begin{aligned}
\mathcal{G}_{ij}(z, s|x', y', z') &= c_{ij}\phi_{ij}(x', y')\,\mathcal{H}_{ij}(z, s|z') \\
\mathcal{H}_{ij}(z, s|z') &= \frac{\eta\left(\underline{Z}_p \cosh\gamma^{(p)}\underline{l} + Z_p \sinh\gamma^{(p)}\underline{l}\right)}{\left(\underline{Z}_p + \bar{Z}_p\right)\cosh\gamma^{(p)}l_p + \left(Z_p + \underline{Z}_p\bar{Z}_p/Z_p\right)\sinh\gamma^{(p)}l_p} \\
\eta &= \frac{\left(\bar{Z}_q \cosh\gamma^{(q)}\bar{l} + Z_q \sinh\gamma^{(q)}\bar{l}\right)\prod_{m=p}^{q-1}\bar{Z}_m}{\prod_{m=p+1}^{q}\left(\bar{Z}_m \cosh\gamma^{(m)}l_m + Z_m \sinh\gamma^{(m)}l_m\right)}.
\end{aligned} \tag{2.32}$$

When $\gamma^{(m)}$ goes to zero, (2.32) can be reformulated by employing the following formula:

$$\lim_{\gamma^{(m)}\to 0} Z_m \sinh\gamma^{(m)}l = \frac{l}{k_m}.$$

Apparently, changing the eigenfunction ϕ_{ij} will lead to the s-domain multilayer Green's function under other types of BCs.

Application of s-domain Multilayer Heat Conduction Green's Function in Computing Thermal Transfer Impedance

With the s-domain multilayer heat conduction Green's function, the thermal transfer impedance from an arbitrary-shape input volume iv to an arbitrary-shape output volume ov, denoted by $R_{th}^{(iv,ov)}(s)$ here, can be given by

$$R_{th}^{(iv,ov)}(s) = \frac{\int_{ov}\int_{iv}\mathcal{G}(x, y, z, s|x', y', z')\,dx'dy'dz'dxdydz}{\int_{ov}dxdydz\int_{iv}dx'dy'dz'}. \tag{2.33}$$

For the thermal transfer impedance at the top surface of a multilayer structure, Kokkas gave the solution for up to three layers [43], and the general solution for an arbitrary number of layers was given implicitly in [29] by a system of linear equations and explicitly in [10] by the product of 2×2 transfer matrices. Based on (2.33), the thermal transfer impedance at the surface is obtained below in a concise form.

The multilayer structure used in the literature is a special case of the MLT model in Fig.2.1(b) by letting $\bar{h} = 0$ and $\underline{h} = \infty$. Therefore at the top surface of the MLT model in Fig.2.1(b), the thermal transfer impedance to any point pt of location (x, y, z_n) from an arbitrary-shape heat source region hs of area $\mathcal{A}$ can be obtained by inserting (2.31) and (2.32) into (2.33):

$$R_{th}^{(hs,pt)}(s) = \frac{1}{\mathcal{A}} \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} I_{ij} c_{ij} \phi_{ij}(x, y)\, \mathcal{H}_{ij}(z_n, s|z_n) \tag{2.34}$$

where $I_{ij} = \int_{hs} \phi_{ij}(x', y')\, dx'dy'$.

Insert $\underline{l} = l_n$, $\bar{l} = 0$ and $\bar{Z}_n = \infty$ into (2.32) and apply (2.16). The following formula results:

$$\mathcal{H}_{ij}(z_n, s|z_n) = \underline{Z}_{n+1} = Z_n \frac{\underline{Z}_n + Z_n \tanh \gamma^{(n)} l_n}{Z_n + \underline{Z}_n \tanh \gamma^{(n)} l_n}. \tag{2.35}$$

Because $\underline{h} = \infty$, $\underline{R} = 0$ and $\underline{Z}_2 = Z_1 \tanh \gamma^{(1)} l_1$. By employing (2.34) and (2.35), the thermal transfer impedance at the surface can be efficiently computed. The complex locus results computed by (2.34) and (2.35) for a structure examined in the literature (Fig.8 in [10]; Fig.5 and 6 in [62]; Fig.17 in [61]) are plotted in Fig.2.4, which shows good agreement with the results in [10, 62] and [61].

To establish compact thermal models, the required thermal-transfer impedance matrices can also be generated by using the s-domain multilayer heat conduction Green's function, as formulated in (2.33) [10, 46, 29]. Identical to the 2×2 transfer matrix approach in [10], the presented method by (2.33) leads to fully analytical double Fourier series such as (2.34). With the explicit formulas for the coefficient $\mathcal{H}_{ij}$, the efficiency of establishing compact thermal models can be improved. For

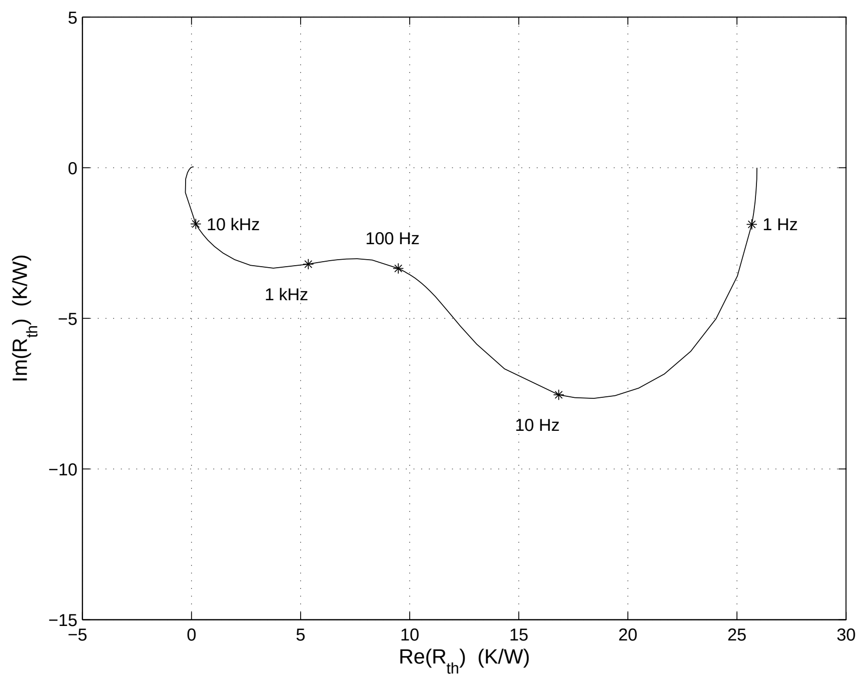


Figure 2.4: Complex locus of thermal impedance for the structure examined in [10, 62] and [61], computed by (2.34) and (2.35).

example, to compute the surface thermal transfer impedance, a system of equations should be solved in an implicit method, and several times more complex number calculations are required by a transfer matrix method than by (2.35).

This chapter derives the multilayer heat conduction Green's function with the inclusion of the s-domain version and demonstrates the Green's function usage by the known examples in the literature. The rest of this dissertation primarily focuses on the steady-state thermal issue and will apply the heat conduction Green's function $G\left(x, y, z | x', y', z'\right)$ derived in this chapter.

CHAPTER III

Computation of Homogeneous and Inhomogeneous Temperature Solutions

The previous chapter presents analytical formulas for both the homogeneous solution T^h and the inhomogeneous solution T^i. This chapter considers the fast computation of the two solutions. Traditional Green's function-based methods consider only the inhomogeneous solution and use a matrix-vector product to compute T^i: multiply the power density function, given as a vector, by a matrix of Green's function values. For n heat source blocks and n temperature observation regions, these methods require $\mathcal{O}(n^2)$ computations to obtain the inhomogeneous temperatures. To speed up the thermal analysis of ULSI chips, this chapter introduces algorithms of $\mathcal{O}(n \lg n)$ complexity to compute both the homogeneous solution and the inhomogeneous solution.

3.1 Computation of Homogeneous Temperature Solution

Section 2.3 gives fully analytical formulas for the homogeneous solution T^h. Based on those formulas, this section introduces an $\mathcal{O}(n \lg n)$ algorithm to compute T^h. The introduced algorithm decomposes a target region in a layer (e.g., layer q) of vertical dimension spanning $[z_{q1}, z_{q2}]$ into $A \times B$ uniform cells, as illustrated by Fig.3.1(a) and (b). Inside a given layer, the cell that is the $(a+1)$-th in the x direction and

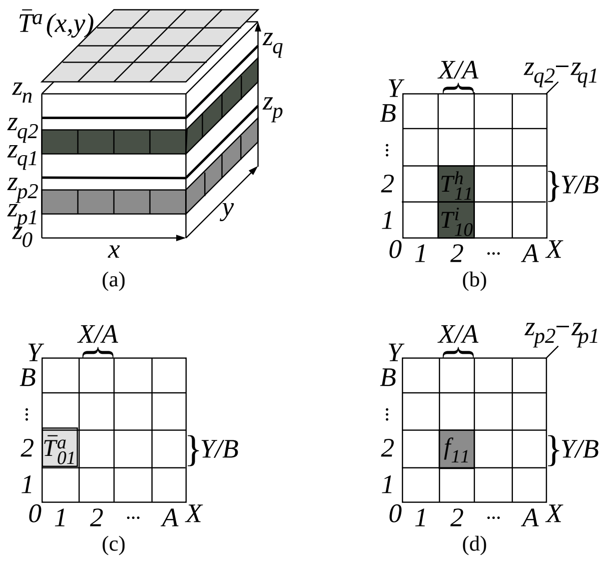


Figure 3.1: Illustration of discrete models for homogeneous temperatures, inhomogeneous temperatures, ambient temperatures, and heat sources: (a) 3-D heat source region with $z \in [z_{p1}, z_{p2}]$, target region with $z \in [z_{q1}, z_{q2}]$, and domain of 2-D top ambient temperature function $\bar{T}^a(x, y)$; (b) discrete homogeneous and inhomogeneous temperature models for target layer q; (c) discrete ambient temperature model for $\bar{T}^a(x, y)$; and (d) discrete heat source model for heat source layer p.

the $(b + 1)$-th in the y direction is named cell (a, b), where $0 \leq a \leq A - 1$ and $0 \leq b \leq B - 1$. Each cell in a target region is of dimensions $\frac{X}{A} \times \frac{Y}{B} \times (z_{q2} - z_{q1})$ and has a uniform homogeneous temperature. For a given cell (a, b) in layer q, its average homogeneous temperature is denoted by T^h_{ab}, as shown in Fig.3.1(b). Similarly, the algorithm partitions the domains of the top and bottom 2-D ambient temperature functions $\bar{T}^a(x, y)$ and $\underline{T}^a(x, y)$ into $A \times B$ uniform cells, as illustrated by Fig.3.1(a) and (c). Each cell discretizing the domains of the top and bottom ambient temperature functions is of dimensions $\frac{X}{A} \times \frac{Y}{B}$ and has a uniform ambient temperature, as shown in Fig.3.1(c). For a given cell (a, b) that discretizes the domain of an ambient temperature function, its ambient temperature is denoted by $\bar{T}^a_{ab}$ or $\underline{T}^a_{ab}$, depending on which ambient temperature function the cell represents.

3.1.1 Eigen-expansion Coefficient $\bar{t}^a_{ij}$ and $\underline{t}^a_{ij}$

The eigen-expansion coefficient $\bar{t}^a_{ij}$ in the eigen-expansion (2.10b) is defined by an integral:

$$\bar{t}^a_{ij} = \frac{2^{2-\delta_{i0}-\delta_{j0}}}{XY} \int_0^X \int_0^Y \bar{T}^a(x,y)\,\phi_{ij}(x,y)\,dxdy.$$

Apply the introduced discretization scheme and carry out the above integral. $\bar{t}^a_{ij}$ is reformulated to

$$\begin{aligned} \bar{t}^a_{ij} &= \frac{2^{2-\delta_{i0}-\delta_{j0}}}{XY} \sum_{a=0}^{A-1}\sum_{b=0}^{B-1} \bar{T}^a_{ab} \int_{aX/A}^{(a+1)X/A} \int_{bY/B}^{(b+1)Y/B} \phi_{ij}(x,y)\,dxdy \\ &= \frac{2^{2-\delta_{i0}-\delta_{j0}}}{ij\pi^2} \sin\left(\frac{i\pi}{2A}\right) \sin\left(\frac{j\pi}{2B}\right) \bar{t}^n_{ij} \end{aligned} \tag{3.1}$$

where $\bar{t}^n_{ij}$ is given by

$$\bar{t}^n_{ij} = \sum_{a=0}^{A-1}\sum_{b=0}^{B-1} 4\bar{T}^a_{ab} \cos\frac{i\pi(2a+1)}{2A} \cos\frac{j\pi(2b+1)}{2B}. \tag{3.2}$$

In formulating $\bar{t}^a_{ij}$ to (3.1), the following trigonometric identity has been used:

$$\begin{aligned} \int_{aX/A}^{(a+1)X/A} \int_{bY/B}^{(b+1)Y/B} \phi_{ij}(x',y')\,dx'dy' &= \frac{XY}{ij\pi^2} \times \\ \left[\sin\left(\frac{i\pi(2a+1)}{2A} + \frac{i\pi}{2A}\right) - \sin\left(\frac{i\pi(2a+1)}{2A} - \frac{i\pi}{2A}\right)\right] &\times \\ \left[\sin\left(\frac{j\pi(2b+1)}{2B} + \frac{j\pi}{2B}\right) - \sin\left(\frac{j\pi(2b+1)}{B} - \frac{j\pi}{2B}\right)\right] & \\ = \frac{4XY}{ij\pi^2} \sin\frac{i\pi}{2A} \sin\frac{j\pi}{2B} \cos\frac{i\pi(2a+1)}{2A} \cos\frac{j\pi(2b+1)}{2B}. & \end{aligned} \tag{3.3}$$

Similarly, from (2.10b), the eigen-expansion coefficient $\underline{t}^a_{ij}$ is reformulated to

$$\underline{t}^a_{ij} = \frac{2^{2-\delta_{i0}-\delta_{j0}}}{ij\pi^2} \sin\left(\frac{i\pi}{2A}\right) \sin\left(\frac{j\pi}{2B}\right) \underline{t}^n_{ij} \tag{3.4}$$

where

$$\underline{t}^n_{ij} = \sum_{a=0}^{A-1}\sum_{b=0}^{B-1} 4\,\underline{T}^a_{ab} \cos\frac{i\pi(2a+1)}{2A} \cos\frac{j\pi(2b+1)}{2B}. \tag{3.5}$$

Formulas (3.2) and (3.5) correspond to the 2-D DCTs of the top and bottom ambient temperatures $\bar{T}^a_{ab}$ and $\underline{T}^a_{ab}$. Therefore, all $\bar{t}^n_{ij}$ and $\underline{t}^n_{ij}$, for $0 \le i \le A-1$ and $0 \le j \le B-1$, can be computed in $\mathcal{O}(AB \lg(AB))$.

3.1.2 Computation of Homogeneous Solution by $\mathcal{O}(n \lg n)$ Algorithm

For a given cell (a, b) in the target region, its average homogeneous temperature T^h_{ab} can be obtained by the integral of $T^h(x, y, z)$ in (2.10a) over cell (a, b):

$$\begin{aligned} T^h_{ab} = {} & \frac{AB}{XY(z_{q2} - z_{q1})} \times \\ & \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \int_{aX/A}^{(a+1)X/A} \int_{bY/B}^{(b+1)Y/B} \int_{z_{q1}}^{z_{q2}} \phi_{ij}(x, y)\, t^h_{ij}(z)\, dxdydz \\ = {} & \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} 2^{d-2} TH_{ij} \cos\frac{i\pi(2a+1)}{2A} \cos\frac{i\pi(2b+1)}{2B}. \end{aligned} \tag{3.6}$$

Here

$$\begin{aligned} TH_{ij} &= \frac{2^{4-d} AB \sin\left(\frac{i\pi}{2A}\right) \sin\left(\frac{j\pi}{2B}\right)}{ij\pi^2 (z_{q2} - z_{q1})} \int_{z_{q1}}^{z_{q2}} t^h_{ij}(z)\, dz \\ &= \frac{16AB \sin^2\frac{i\pi}{2A} \sin^2\frac{j\pi}{2B}}{i^2 j^2 \pi^4 (z_{q2} - z_{q1})} \left[\bar{t}^n_{ij} \overline{IH}^a_{ij} + \underline{t}^n_{ij} \underline{IH}^a_{ij}\right] \end{aligned} \tag{3.7}$$

where $\overline{IH}^a_{ij}$ and $\underline{IH}^a_{ij}$ are the integrals of $\bar{H}^a_{ij}$ and $\underline{H}^a_{ij}$ over the interval $[z_{q1}, z_{q2}]$, i.e.,

$$\overline{IH}^a_{ij} = \int_{z_{q1}}^{z_{q2}} \bar{H}^a_{ij}(z)\, dz \text{ and } \underline{IH}^a_{ij} = \int_{z_{q1}}^{z_{q2}} \underline{H}^a_{ij}(z)\, dz. \tag{3.8}$$

Then truncate the series representation of T^h_{ab} in (3.6) into the following form:

$$T^h_{ab} \approx \sum_{i=0}^{A-1}\sum_{j=0}^{B-1} 2^{d-2} TH_{ij} \cos\frac{i\pi(2a+1)}{2A} \cos\frac{i\pi(2b+1)}{2B} \tag{3.9}$$

which corresponds to the 2-D IDCT of TH_{ij}. Therefore all T^h_{ab}, for $0 \le a \le A-1$ and $0 \le b \le B-1$, can be computed in $\mathcal{O}(AB \lg(AB))$. Based on the previous formulas, this section introduces an $\mathcal{O}(n \lg n)$ algorithm, named Compute-T^h, to compute the homogeneous solution: first compute the 2-D DCTs of the discrete top and bottom

```
Begin Compute-$T^h$
    1. Compute the 2-D DCTs of the given top and bottom ambient temperatures $\bar{T}^a_{ab}$ and $\underline{T}^a_{ab}$ by
       (3.2) and (3.5). Then obtain $\bar{t}^n_{ij}$ and $\underline{t}^n_{ij}$.
    2. Compute all $TH_{ij}$, for $0 \leq i \leq A-1$ and $0 \leq j \leq B-1$, by (3.7), (3.10) and (3.11).
    3. Compute the 2-D IDCT of $TH_{ij}$ by (3.9). Then obtain $T^h_{ab}$.
End Compute-$T^h$
```

Figure 3.2: Compute-T^h: $\mathcal{O}(n \lg n)$ algorithm for computing homogeneous solution T^h.

ambient temperatures $\bar{T}^a_{ab}$ and $\underline{T}^a_{ab}$ by (3.2) and (3.5) to obtain $\bar{t}^n_{ij}$ and $\underline{t}^n_{ij}$, ; then compute all TH_{ij} by (3.7) for $0 \leq i \leq A-1$ and $0 \leq j \leq B-1$; finally, compute the 2-D IDCT of TH_{ij} by (3.9) to obtain T^h_{ab}. Fig.3.2 shows the algorithm Compute-T^h.

Note that the truncation of the infinite series (3.6) up to orders A and B allows the use of the 2-D IDCT to achieve an $\mathcal{O}(AB \lg (AB))$ algorithm. The accuracy of this truncation approximation can be improved by folding back spectral components of orders higher than A and B into spectral components of orders lower than A and B, i.e., adding DCT coefficients TH_{ij} with $i \geq A$ and $j \geq B$ to the corresponding DCT coefficients TH_{ij} with $i < A$ and $j < B$, because of the periodicity of $\cos \frac{i\pi(2a+1)}{2A} \cos \frac{j\pi(2b+1)}{2B}$. The later experimental results will demonstrate that the simple truncation given by (3.9) already provides sufficient accuracy, notwithstanding without folding back any high-order spectral components.

3.1.3 DCT Coefficients

From (2.13) and (2.15), $\overline{IH}^a_{ij}$ and $\underline{IH}^a_{ij}$ in (3.8) are obtained.

For $i + j = 0$,

$$\overline{IH}^a_{00} = \frac{\underline{Z}_q l_{qv} + Z_q l_{qv} \underline{l}_{q12c}}{\underline{Z}_q + Z_q l_q + \bar{Z}_q} \tag{3.10a}$$

$$\underline{IH}^a_{00} = \frac{\bar{Z}_q l_{qv} + Z_q l_{qv} \bar{l}_{q12c}}{\underline{Z}_q + Z_q l_q + \bar{Z}_q} \tag{3.10b}$$

where $\underline{l}_{q12c} = \frac{z_{q2}+z_{q1}}{2} - z_{q-1}$ and $\bar{l}_{q12c} = z_q - \frac{z_{q2}+z_{q1}}{2}$.

For $i + j > 0$,

$$\overline{IH}^{a}_{ij} = \frac{\underline{Z}_q + Z_q}{2\gamma\bar{\xi}\left(1 + \bar{R}/\underline{Z}_{n+1}\right)} \frac{e^{\gamma \underline{l}_{q2}} - e^{\gamma \underline{l}_{q1}} + \underline{D}^q_{ij}\left(e^{-\gamma \underline{l}_{q1}} - e^{-\gamma \underline{l}_{q2}}\right)}{\underline{Z}_q \cosh\gamma l_q + Z_q \sinh\gamma l_q} \tag{3.11a}$$

$$\underline{IH}^{a}_{ij} = \frac{\bar{Z}_q + Z_q}{2\gamma\underline{\xi}\left(1 + \underline{R}/\bar{Z}_{0}\right)} \frac{e^{\gamma \bar{l}_{q1}} - e^{\gamma \bar{l}_{q2}} + \bar{D}^q_{ij}\left(e^{-\gamma \bar{l}_{q2}} - e^{-\gamma \bar{l}_{q1}}\right)}{\bar{Z}_q \cosh\gamma l_q + Z_q \sinh\gamma l_q} \tag{3.11b}$$

where

$$\begin{aligned} \underline{l}_{q1,2} &= z_{q1,2} - z_{q-1} \\ \bar{D}^q_{ij} &= \frac{\bar{Z}_q - Z_q}{\bar{Z}_q + Z_q} \\ \underline{D}^q_{ij} &= \frac{\underline{Z}_q - Z_q}{\underline{Z}_q + Z_q}. \end{aligned}$$

In fact, $\bar{D}^q_{ij}$ is the reflection coefficient of the q-th TL section, seen from its top boundary toward the top side of the circuit, and $\underline{D}^q_{ij}$ is the reflection coefficient of the q-th TL section, seen from its bottom boundary toward the bottom side of the equivalent TL circuit. $\bar{D}^q_{ij}$ and $\underline{D}^p_{ij}$ are functions of a single parameter γ, where $\gamma = \sqrt{\frac{i^2\pi^2}{X^2} + \frac{j^2\pi^2}{Y^2}}$. Therefore, for a given process technology, regardless of A and B, these coefficients can be characterized into 1-D look-up tables indexed by the parameter γ. Then in the pre-characterization of $\overline{IH}^a_{ij}$ and $\underline{IH}^a_{ij}$ for given values of A and B, the required values of $\bar{D}^q_{ij}$ and $\underline{D}^p_{ij}$ can be obtained from the 1-D look-up tables.

3.2 Computation of Inhomogeneous Temperature Solution

First introduce a heat source model to describe the power density distribution f. In the heat source model, uniform cells are employed to discretize the heat source regions and the target regions. Heat sources in one layer, e.g., the p-th layer, are partitioned into $A \times B$ uniform cells, each being of dimensions $\frac{X}{A} \times \frac{Y}{B} \times (z_{p2} - z_{p1})$ and having a uniform power density, as shown in Fig.3.1(a) and (d). To simplify notations, the numbers of cells in the x and y directions are still denoted by A and B, respectively, although the number of cells employed in calculating the inhomogeneous solution can be different from that in calculating the homogeneous solution. For a given cell (a, b), its power density is denoted by f_{ab}, its average inhomogeneous temperature is denoted by T^i_{ab}, and its overall temperature is denoted by T_{ab}: $T_{ab} = T^i_{ab} + T^h_{ab}$.

According to (2.9), when there are multiple layers of heat sources, the inhomogeneous solution T^i at a given target layer, e.g., layer q, can be obtained by superposing each inhomogeneous solution at layer q caused by a single layer of heat sources. Therefore, it is adequate to provide an algorithm that evaluates the inhomogeneous solution T^i at layer q, caused by the heat sources at only one layer, e.g., layer p. Layer q is illustrated in Fig.3.1(a) and (b). To obtain the inhomogeneous solution at layer q caused by heat sources in layer p, traditional Green's function-based methods need $\mathcal{O}\left(A^2B^2\right)$ computations because of the dense matrix-vector product. Based on the multilayer heat conduction Green's function, this section introduces a fast yet accurate algorithm to compute the inhomogeneous solution in $\mathcal{O}\left(AB \lg\left(AB\right)\right)$.

3.2.1 Inhomogeneous Temperature for One Layer of Heat Sources

Consider heat source layer p, whose thickness is $z_{p2} - z_{p1}$, as illustrated in Fig.3.1(d). Insert eigen-expansion (2.18) into (2.9) and carry out the integral by convoluting the multilayer heat conduction Green's function with the power density distribution at layer p. Then the inhomogeneous temperature at an arbitrary location (x, y, z),

$T^i(x,y,z)$, is obtained:

$$\begin{aligned}
T^i(x,y,z) &= \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} c_{ij}\phi_{ij}(x,y)\times \\
&\int_0^X\int_0^Y\int_{z_{p1}}^{z_{p2}} \phi_{ij}(x',y')H_{ij}(z|z')f(x',y',z')dx'dy'dz' \\
&= \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \frac{2^{2-\delta_{i0}-\delta_{j0}}\sin\frac{i\pi}{2A}\sin\frac{j\pi}{2B}}{ij\pi^2} F_{ij}\phi_{ij}(x,y)\int_{z_{p1}}^{z_{p2}} H_{ij}(z|z')dz' \qquad (3.12)
\end{aligned}$$

where

$$\begin{aligned}
F_{ij} &= \int_0^X\int_0^Y \frac{ij\pi^2}{XY}\csc\frac{i\pi}{2A}\csc\frac{j\pi}{2B}\phi_{ij}(x',y')f(x',y',z')dx'dy' \\
&= \frac{ij\pi^2}{XY}\csc\frac{i\pi}{2A}\csc\frac{j\pi}{2B}\times \\
&\sum_{a=0}^{A-1}\sum_{b=0}^{B-1} f_{ab}\int_{aX/A}^{(a+1)X/A}\int_{bY/B}^{(b+1)Y/B} \phi_{ij}\left(x',y'\right)dx'dy' \\
&= \sum_{a=0}^{A-1}\sum_{b=0}^{B-1} 4f_{ab}\cos\frac{i\pi(2a+1)}{2A}\cos\frac{j\pi(2b+1)}{2B}. \qquad (3.13)
\end{aligned}$$

The F_{ij} formulated above is exactly the 2-D DCT of f_{ab}. Therefore, all F_{ij}, for $0 \le i \le A-1$ and $0 \le j \le B-1$, can be computed in $\mathcal{O}\left(AB\lg(AB)\right)$. For i,j outside that range, the value of F_{ij} can be obtained by exploiting the periodicity of F_{ij}: $F_{(2A-i)j} = -F_{ij}$ and $F_{i(2B-j)} = -F_{ij}$. Note that in (3.13), the trigonometric identify in (3.3) has been used.

3.2.2 Computation of Inhomogeneous Solution by $\mathcal{O}(n\lg n)$ Algorithm

Consider the target layer q, whose thickness is $z_{q2}-z_{q1}$, as illustrated in Fig.3.1(b). For a given cell (a,b) in layer q, its average inhomogeneous temperature T^i_{ab} is obtained by the integral of the inhomogeneous temperature $T^i(x,y,z)$ in (3.12) over cell (a, b):

$$T^i_{ab} = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} F_{ij} \frac{2^{2-\delta_{i0}-\delta_{j0}} AB \sin\frac{i\pi}{2A}\sin\frac{j\pi}{2B}}{ijXY\pi^2(z_{q2}-z_{q1})} \int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}} H_{ij}(z|z')dz'dz$$

$$\times \int_{aX/A}^{(a+1)X/A}\int_{bY/B}^{(b+1)Y/B} \phi_{ij}(x,y)dxdy$$

$$= \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} 2^{-\delta_{i0}-\delta_{j0}} F_{ij} IH_{ij} \cos\frac{i\pi(2a+1)}{2A}\cos\frac{j\pi(2b+1)}{2B} \tag{3.14}$$

where IH_{ij} is given by

$$IH_{ij} = \frac{16AB\sin^2\frac{i\pi}{2A}\sin^2\frac{j\pi}{2B}}{i^2j^2\pi^4(z_{q2}-z_{q1})} \int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}} H_{ij}(z|z')dz'dz. \tag{3.15}$$

Note that in simplifying (3.14), the trigonometric identify in (3.3) has been used.

Then truncate the series representation of T^i_{ab} in (3.14) into the following form:

$$T^i_{ab} \approx \sum_{i=0}^{A-1}\sum_{j=0}^{B-1} 2^{-\delta_{i0}-\delta_{j0}} F_{ij} IH_{ij} \cos\frac{i\pi(2a+1)}{2A}\cos\frac{j\pi(2b+1)}{2B}. \tag{3.16}$$

The above truncated series representation of T^i_{ab} is exactly the 2-D IDCT of $F_{ij}IH_{ij}$. As a result, the algorithm named Compute-T^i to evaluate the inhomogeneous solution at layer q, caused by heat sources at layer p, consists of one 2-D DCT procedure to compute F_{ij} based on (3.13) and another 2-D IDCT procedure to compute T^i_{ab} based on (3.16). The time complexity of the algorithm is $\mathcal{O}(AB\lg(AB))$. The complete thermal analysis method named LOTAGre, which integrates algorithms Compute-T^i and Compute-T^h, is shown in Fig.3.3. Note that folding back high-order spectral components $F_{ij}IH_{ij}$ can also improve the accuracy of the truncation approximation (3.16).

Both Compute-T^i and Compute-T^h use the 2-D DCT and 2-D IDCT procedures to achieve $\mathcal{O}(n\lg n)$ run-time. However, the involved DCT and IDCT coefficients have different physical meanings. In Compute-T^i, coefficient IH_{ij} is related to the transfer impedance of the equivalent circuits. In Compute-T^h, coefficients $\overline{IH}_{ij}$ and $\underline{IH}_{ij}$ are related to the voltage transfer functions of the equivalent circuits.

```
Begin Compute-$T^i$
    1. Compute all $IH_{ij}$, for $0 \leq i < A$ and $0 \leq i < B$, by (3.15), (3.17) and (3.18).
    2. Given one layer of heat sources, whose power densities are defined by a 2-D array made of
       $f_{ab}$, for $0 \leq a < A$ and $0 \leq b < B$, compute the 2-D DCT of $f_{ab}$ by (3.13) to obtain $F_{ij}$.
    3. Form an array made of $F_{ij}IH_{ij}$, for $0 \leq i < A$ and $0 \leq i < B$, and then compute the 2-D
       IDCT of that array by (3.16) to obtain $T^i_{ab}$.
End Compute-$T^i$
Begin LOTAGre
    1. Apply algorithm Compute-$T^i$ to compute $T^i_{ab}$.
    2. Apply algorithm Compute-$T^h$ to compute $T^h_{ab}$.
    3. Sum up $T^i_{ab}$ and $T^h_{ab}$ to obtain temperature $T_{ab}$.
End LOTAGre
```

Figure 3.3: LOTAGre: $\mathcal{O}(n \lg n)$ multilayer heat conduction Green's function-based thermal analysis method.

3.2.3 Pre-characterization of IH_{ij}

To simulate a chip, all IH_{ij}'s should be pre-characterized only once. Then, for any given power density distribution in the form of f_{ab}, multiplying its 2-D DCT F_{ij} by the pre-calculated value of IH_{ij} and applying the 2-D IDCT of $F_{ij}IH_{ij}$ will obtain the corresponding inhomogeneous temperature T^i_{ab}. The following details the procedure to pre-characterize IH_{ij}.

Let $\hat{H}_{ij}$ denote the integral term in (3.15), i.e. $\hat{H}_{ij} = \int_{z_{q1}}^{z_{q2}} \int_{z_{p1}}^{z_{p2}} H_{ij}(z|z')dz'dz$. To compute $\hat{H}_{ij}$, it should be noted that in the representations of H_{ij} given in (2.21) and (2.24), it is assumed that either the source layer p is lower than the target layer q, or both $p = q$ and $z' < z$, to simplify the presentation. Beyond the assumption, H_{ij} can be obtained by the reciprocity of transfer functions: if $p = q$ and $z' > z$, H_{ij} can be obtained by exchanging z and z' in (2.21) and (2.24); otherwise, if $p > q$, H_{ij} can be obtained by exchanging the subscripts p and q, as well as z and z', in (2.21) and (2.24). Therefore, three cases are considered in computing $\hat{H}_{ij}$.

The case that layer p and layer q are the same ($p = q$)

From (2.21) and (2.24), $\hat{H}_{ij}$ is obtained:

$$(3.17) \qquad \hat{H}_{ij} = \begin{cases} \alpha l_{pv}^2 \left[\left(\dfrac{\underline{Z}_p}{Z_p} + \dfrac{2z_{p1} + z_{p2}}{3} - z_{p-1} \right) \right. \\ \left. \times \left(\dfrac{\bar{Z}_q}{Z_q} + z_q - \dfrac{2z_{q2} + z_{q1}}{3} \right) - \dfrac{l_{qv}^2}{36} \right] & i = j = 0 \\ \\ \left[\underline{D}_{ij}^p \left(e^{-\gamma l_{2c}} - e^{-\gamma l_{1c}} \right)^2 + \bar{D}_{ij}^q \left(e^{\gamma l_{1c}} - e^{\gamma l_{2c}} \right)^2 \right. \\ \qquad + 2\underline{D}_{ij}^p \bar{D}_{ij}^q e^{-\gamma l_q} \left(e^{\gamma l_{qv}} - \gamma l_{qv} - 1 \right) & i + j > 0 \\ \qquad \left. + \, 2e^{\gamma l_p} \left(e^{-\gamma l_{pv}} + \gamma l_{pv} - 1 \right) \right] E_{ij}. \end{cases}$$

Here $l_{pv} = z_{p2} - z_{p1}$, $l_{qv} = z_{q2} - z_{q1}$, $l_{1c} = z_{p1} - \frac{z_p + z_{p-1}}{2}$, and $l_{2c} = z_{p2} - \frac{z_p + z_{p-1}}{2}$.

The case that $p < q$

From (2.21) and (2.24), $\hat{H}_{ij}$ is obtained:

$$(3.18) \qquad \hat{H}_{ij} = \begin{cases} \alpha l_{pv} l_{qv} \left(\dfrac{\underline{Z}_p}{Z_p} + \dfrac{z_{p1} + z_{p2}}{2} - z_{p-1} \right) \times \\ \qquad \left(\dfrac{\bar{Z}_q}{Z_q} + z_q - \dfrac{z_{q1} + z_{q2}}{2} \right) & i = j = 0 \\ \\ E_{ij} \left[\underline{D}_{ij}^p \left(e^{-\gamma \underline{l}_{p1}} - e^{-\gamma \underline{l}_{p2}} \right) + e^{\gamma \underline{l}_{p2}} - e^{\gamma \underline{l}_{p1}} \right] \times \\ \qquad \left[\bar{D}_{ij}^q \left(e^{-\gamma \bar{l}_{q2}} - e^{-\gamma \bar{l}_{q1}} \right) + e^{\gamma \bar{l}_{q1}} - e^{\gamma \bar{l}_{q2}} \right]. & i + j > 0 \end{cases}$$

Here $\underline{l}_{p1,2} = z_{p1,2} - z_{p-1}$ and $\bar{l}_{q1,2} = z_q - z_{q1,2}$.

The case that $p > q$

The expression of $\hat{H}_{ij}$ is similar to (3.18). In this case, $\hat{H}_{ij}$ can be obtained by exchanging the subscripts p and q in (3.18) and also the subscripts p and q in coefficients α, $\underline{D}_{ij}^p$, $\bar{D}_{ij}^q$ and E_{ij}.

The previous coefficients are given by

$$
\begin{aligned}
\alpha &= \frac{Z_p Z_q}{\underline{Z}_p + Z_p l_p + \bar{Z}_q} \\
\underline{D}^p_{ij} &= \frac{\underline{Z}_p - Z_p}{\underline{Z}_p + Z_p} \\
E_{ij} &= \frac{\xi}{4\gamma^2} \frac{\left(\underline{Z}_p + Z_p\right)\left(\bar{Z}_q + Z_q\right)}{Z_p\left(\underline{Z}_p + \bar{Z}_p\right)\cosh\gamma l_p + \left(Z_p^2 + \underline{Z}_p \bar{Z}_p\right)\sinh\gamma l_p}.
\end{aligned}
$$

According to the equivalent TL circuit shown in Fig.2.1(d), $\underline{D}^p_{ij}$ is the reflection coefficient of the p-th TL section, seen from its bottom boundary toward the bottom side of the circuit.

The coefficients $\underline{D}^p_{ij}$ and E_{ij} also depend upon a single parameter γ, where $\gamma = \sqrt{\frac{i^2\pi^2}{X^2} + \frac{j^2\pi^2}{Y^2}}$. Therefore, 1-D lookup tables can also be constructed for these coefficients to speed up the pre-characterization of IH_{ij}. With the required values of $\underline{D}^p_{ij}$ and E_{ij}, IH_{ij} can be computed from (3.15), (3.17), and (3.18).

3.3 Experimental Results

3.3.1 Accuracy and Speed of LOTAGre

The $\mathcal{O}\left(n \lg n\right)$ multilayer heat conduction Green's function-based thermal analysis method is named LOTAGre. The method was verified by comparisons with a sophisticated computational fluid dynamics tool, called FLUENT. As introduced in chapter II, the MLT model in Fig.2.1(b) used in LOTAGre can consider different types of chip packaging scenarios, e.g., wire-bonding packaging and flip-chip packaging. The following uses a die with flip-chip packaging as an example.

Fig.3.4 shows a chip example which has a structure similar to the PowerPC[1] chip [50, 76]. The figure shows two heat conduction paths in the chip. One heat conduction path transfers the heat generated in the active region of the chip through the silicon bulk, the thermal adhesive, and the heat sink to the top ambient environment.

[1]PowerPC is a trademark of IBM Corp., used under license by Motorola Inc.

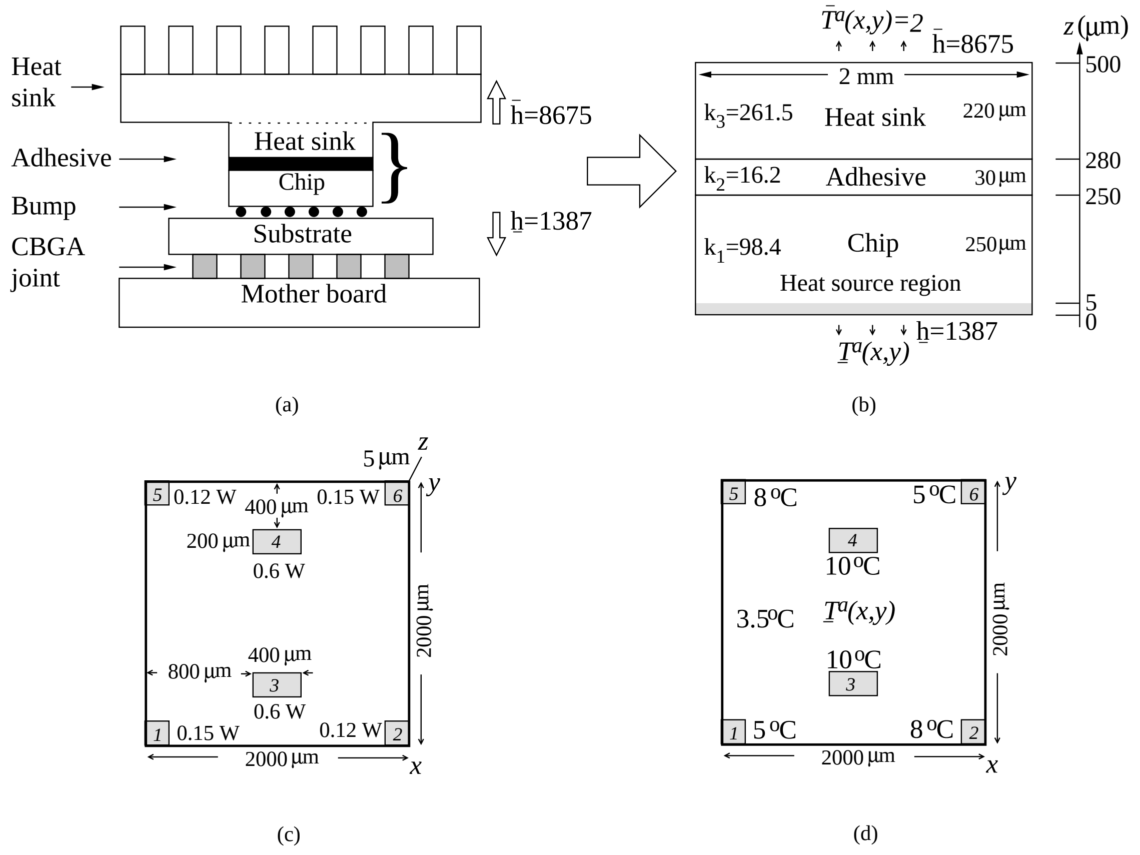


Figure 3.4: Example chip of flip-chip packaging: (a) real chip structure; (b) MLT model for the given chip; (c) specification of power density distribution f for heat source region; and (d) specification of 2-D bottom ambient temperature function $\underline{T}^a(x, y)$, which models thermal effects of the bumps.

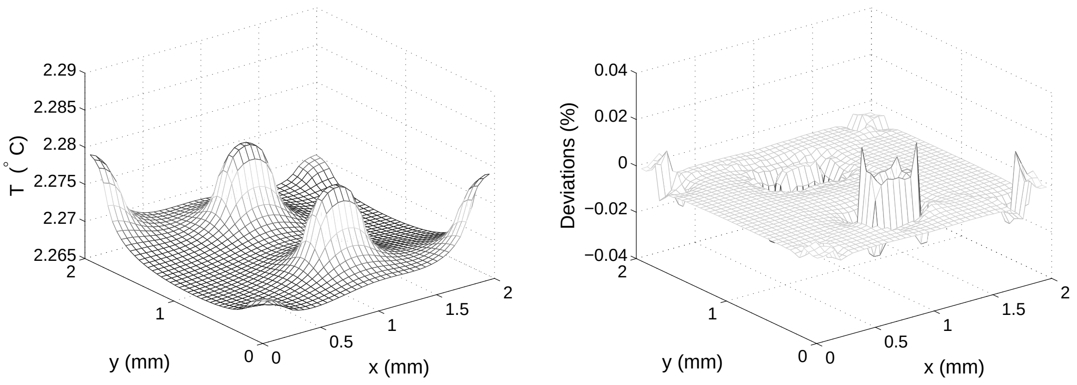
The other heat conduction path transfers the heat through the bump, the under-fill materials, and the substrate to the bottom ambient environment. Fig.3.4(b) shows a three-layer MLT model for the example chip. In the three-layer MLT model, the bottom two layers incorporate the entire chip and the thermal adhesive, and the top layer models one portion of the heat sink. The thermal effects of the other regions excluded in the MLT model are addressed by the top heat transfer rate $\bar{h}$, the bottom heat transfer rate $\underline{h}$, and the top and bottom ambient temperature functions $\bar{T}^a(x, y)$ and $\underline{T}^a(x, y)$. The two heat transfer rates $\bar{h}$ and $\underline{h}$ can be determined by either empirical formulas or experimental data fitting [50, 22]. In the experiments, $\bar{h}$ was set to 8675 W/(m^2 K), and $\underline{h}$ was set to 1387 W/(m^2 K). The chip horizontal dimensions were 2×2 mm^2. Fig.3.4(b) also shows the thickness and thermal conductivity of each layer. Inside the chip layer of the MLT model, there was a 5 μm thick heat source region, where six rectangular heat sources were placed. Fig.3.4(c) shows the six heat sources, and the power of each heat source is given near the heat source box.

The temperature distribution of the example chip was analyzed by LOTAGre and FLUENT. In LOTAGre, A and B were set to 40. However, it is recommended that A and B be the powers of 2, to facilitate the DCT and IDCT algorithms. At first, the homogeneous temperature distribution was analyzed. The 2-D top ambient temperature function $\bar{T}^a(x, y)$ was assumed to take a constant value. The 2-D bottom ambient temperature function $\underline{T}^a(x, y)$ was specified as in Fig.3.4(d). As shown by Fig.3.4(c) and (d), the specified bottom ambient temperature function was very similar to the specified power density function, except that the ambient temperatures replaced the powers in Fig.3.4(c).

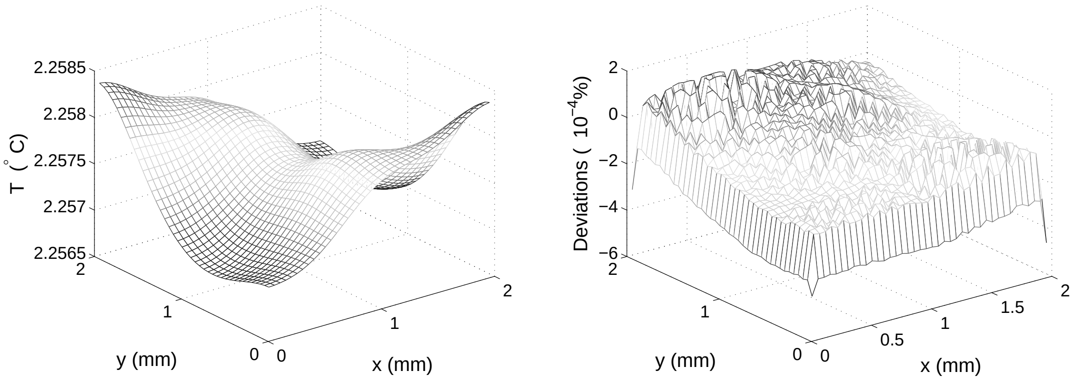
The results for the homogeneous temperature distribution are shown in Fig.3.5(a) and (b), where the left graphs give the homogeneous temperature distributions obtained from LOTAGre, and the right graphs show the relative differences of the calculated temperatures from FLUENT. Fig.3.5(a) and (b) demonstrate that the homogeneous temperature deviations between the two methods were within 0.04% in the heat source region and within 0.001% on the top surface of the MLT model. In terms of CPU usages, LOTAGre took 1.193 s to pre-characterize all $\overline{IH}_{ij}$ and $\underline{IH}_{ij}$ and then took only 47 ms to calculate the homogeneous temperature distribution, while FLUENT took 269 s to obtain the solution. A SUN Blade 1500 machine was used in running the experiments.

Fig.3.6(a) and (b) show the results for the inhomogeneous temperature distribution. The left graphs give the inhomogeneous temperature distributions obtained by LOTAGre, and the right graphs show the relative temperature differences from FLUENT. Fig.3.6(a) and (b) again demonstrate the accuracy of LOTAGre: the inhomogeneous temperature deviations between the two methods were within 1.18% in the heat source region and within 0.529% on the top surface of the MLT model. The pre-characterization of IH_{ij} took 1.283 s. The evaluation of the inhomogeneous temperature distribution after the pre-characterization only took 44 ms, while FLUENT took 205 s to obtain the solution.

Different sets of parameters $\underline{h}$, k_2 and $\bar{h}$, as shown in Table 3.1, were also experimented for the MLT model shown in Fig.3.4(b), and the temperature distributions on the top surface and in the heat source region of the MLT model were again solved by LOTAGre and FLUENT, respectively. The results are shown in Table 3.2, where letter H indicates the homogeneous temperature results, while letter I indicates the inhomogeneous temperature results; *Max.* indicates the maximum temperature; *Pre.* indicates the pre-characterization time taken by LOTAGre; *Eva.* indicates

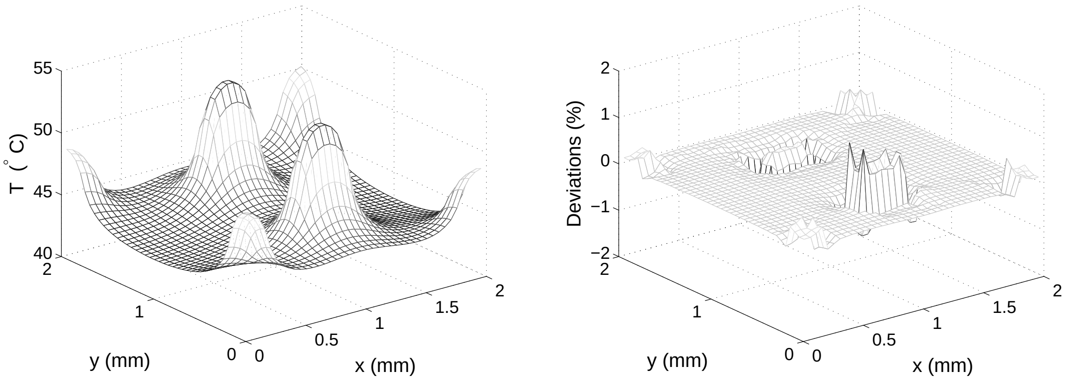


(a) Homogeneous temperature distribution in heat source region.

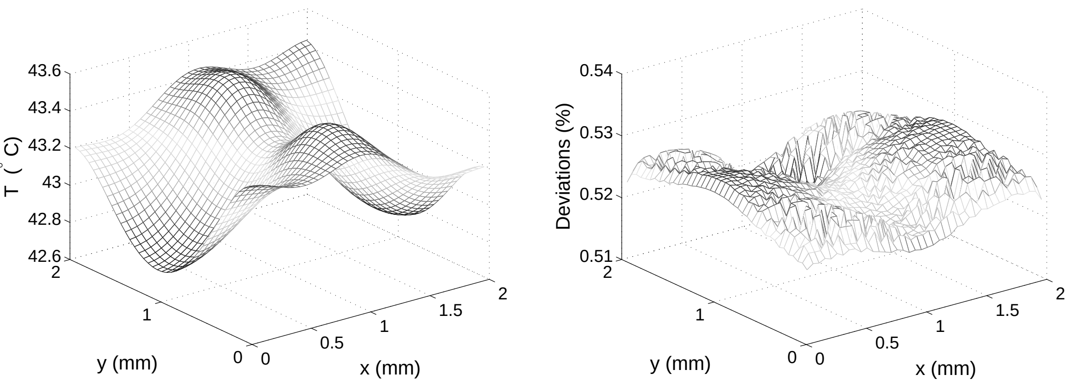


(b) Homogeneous temperature distribution on top surface of MLT model.

Figure 3.5: Comparison between LOTAGre and FLUENT in computing the homogeneous temperature distribution. In (a) and (b), left graphs show temperature distributions computed by LOTAGre, and right graphs show relative temperature differences from FLUENT in percentages.



(a) Inhomogeneous temperature distribution in heat source region.



(b) Inhomogeneous temperature distribution on top surface of MLT model.

Figure 3.6: Comparison between LOTAGre and FLUENT in computing the inhomogeneous temperature distribution. In (a) and (b), left graphs show temperature distributions computed by LOTAGre, and right graphs show relative temperature differences from FLUENT in percentages.

Table 3.1: Parameters of the examples used in comparing LOTAGre and FLUENT.

	EX1	EX2	EX3	EX4	EX5	EX6	EX7	EX8	EX9	EX0
$\underline{h}$ (W/m^{2}K)	257	5212	9826	1387	2715	1462	512	1128	1682	21
k_2 (W/m K)	12.2	32.4	16.2	5.1	382	1715	10.4	15.3	60.1	70.2
h (W/m^{2}K)	8675	5419	2371	9451	7213	8415	800.1	3410	13215	9898

Table 3.2: Comparisons between LOTAGre and FLUENT for the example chip in Fig.3.4 under wide parameter variations.

EXS	*H-Max.*	*H-Pre.*	*H-Eva.*	*H-FLU.*	*H-Dev.*	*I-Max.*	*I-Pre.*	*I-Eva.*	*I-FLU.*	*I-Dev.*
	(^{o}C)	(s)	(ms)	(s)		(^{o}C)	(s)	(ms)	(s)	
EX1	2.058	1.238	52	290	0.02%	59.44	1.215	42	51	0.95%
EX2	2.968	1.203	45	83	0.59%	49.51	1.266	42	182	1.27%
EX3	3.587	1.17	57	134	0.74%	43.94	1.327	46	202	1.44%
EX4	2.269	1.14	47	166	0.05%	51.71	1.255	48	373	1.23%
EX5	2.542	1.211	48	93	0.26%	52.51	1.226	53	34	1.16%
EX6	2.295	1.187	47	291	0.04%	52.88	1.214	43	140	1.16%
EX7	2.74	1.197	50	941	0.38%	340.9	1.238	42	553	0.55%
EX8	2.482	1.154	51	238	0.85%	105.5	1.32	46	232	0.55%
EX9	2.233	1.171	48	145	0.07%	38.58	1.27	42	364	1.67%
EX0	2.004	1.195	48	223	0.01%	53.56	1.227	45	81	1.17%

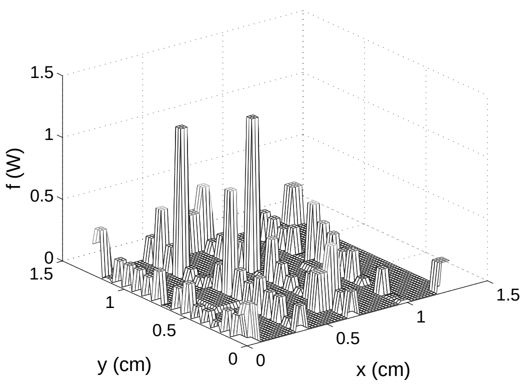
the temperature evaluation time taken by LOTAGre; *FLU.* indicates the run-time of FLUENT; and *Dev.* indicates the maximum of the relative temperature differences from FLUENT. Table 3.2 shows that the homogeneous temperature differences between LOTAGre and FLUENT were within 0.9%, and the inhomogeneous temperature differences between the two methods were within 1.7%. The results in the table have demonstrated the accuracy of LOTAGre, despite the large parameter variations. They also demonstrate the superior speed advantage of LOTAGre, which was around two orders of magnitude faster than FLUENT if the pre-characterization time is taken into consideration. If LOTAGre is used in an inner loop and iterated many times for different power density distributions, the pre-characterization of coefficients needs to be done only once. Therefore, LOTAGre can asymptotically be thousands of times faster than FLUENT.

3.3.2 Scalability of LOTAGre

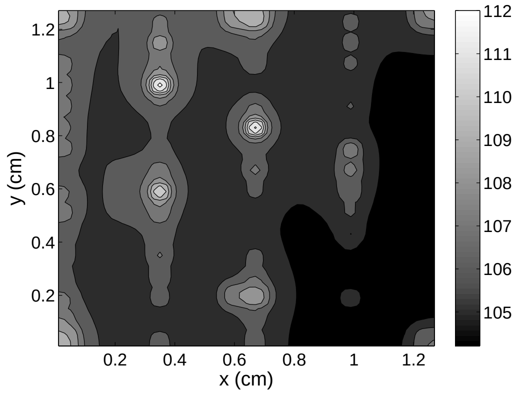
Theoretically, LOTAGre is of $\mathcal{O}(n \lg n)$ complexity, while traditional Green's function-based thermal analysis methods are of quadratic complexity [21, 78]. To demonstrate the scalability, LOTAGre was employed to analyze the example chip in Fig.3.4, but the $x - y$ dimensions were extended to 1.28×1.28 cm^2 to accommodate more standard cells. A randomly generated heat source distribution f_{ab} was imposed on the 5 μm thick heat source region of the chip, which is indicated in Fig.3.4(b). Only the inhomogeneous temperature distribution was analyzed, since the homogeneous temperature distribution did not depend on the power density distribution and could be computed only once for a given ambient condition. The power density distribution and the calculated inhomogeneous temperature distribution in the heat source region are shown in Fig.3.7(a) and (b).

The CPU usages taken by LOTAGre during pre-characterization are shown in Table 3.3 (*Pre-char.*), for the number of cells $A \times B$ varying from 32×32 to 1024×1024. Note that the CPU usages during pre-characterization will be amortized in an iterative thermal analysis flow, since LOTAGre conducts the pre-characterization of IH_{ij} only once. The table shows that the pre-characterization time is almost linearly related to $A \times B$, as all IH_{ij} for $0 \le i \le A-1$ and $0 \le j \le B-1$ were computed by LOTAGre. As introduced in Section 3.2.3, the pre-characterization time would be reduced further if 1-D look-up tables were established for coefficients $\underline{D}^p_{ij}$, $\bar{D}^q_{ij}$ and E_{ij} before running the experiments.

For comparisons, a matrix-vector product program was implemented to simulate the traditional Green's function-based methods [21, 78]. Fig.3.7 shows the randomly generated heat source distribution used in the experiments and the resultant inhomogeneous temperature distribution computed by LOTAGre. Table 3.3 compares the CPU usages of LOTAGre and the traditional methods. According to the table, when the number of cells $A \times B$ doubled, the temperature evaluation time by LOTAGre (*LOTAGre*) increased a little more than two-fold, while the time by the traditional



(a) Randomly generated heat source distribution f.



(b) Inhomogeneous temperature distribution in heat source region.

Figure 3.7: Applied heat source distribution in testing the scalability of LOTAGre and the resultant inhomogeneous temperature distribution computed by LOTAGre.

A	B	*Pre-char.* (s)	*LOTAGre* (s)	*Trad.* (s)
32	32	0.038	0.021	0.012
32	64	0.073	0.023	0.046
64	64	1.442	0.026	0.192
64	128	2.866	0.032	0.85
128	128	5.825	0.047	4.733
128	256	11.501	0.082	18.75
256	256	22.713	0.171	82.892
256	512	45.663	0.407	707.07
512	512	90.79	1.236	N/A
512	1024	181.428	3.282	N/A
1024	1024	362.350	7.537	N/A

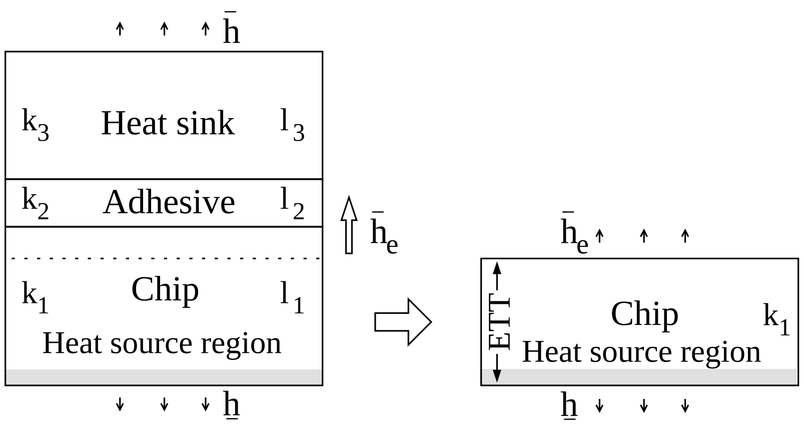
Table 3.3: Scalability of LOTAGre: comparison of CPU usages by LOTAGre and traditional Green's function-based thermal analysis methods.

methods (*Trad.*) increased around four-fold. These run-time data closely matched the theoretical complexities of LOTAGre and the traditional methods. LOTAGre clearly had superior computing speed and was also scalable to large problem size. For example, when $A \times B$ increased to 1024×1024, a chip of one million standard cells was analyzed in less than 8 s by LOTAGre. In contrast, the traditional methods became extremely slow even when the number of cells was no more than 256×512, or one-eighths of one million.

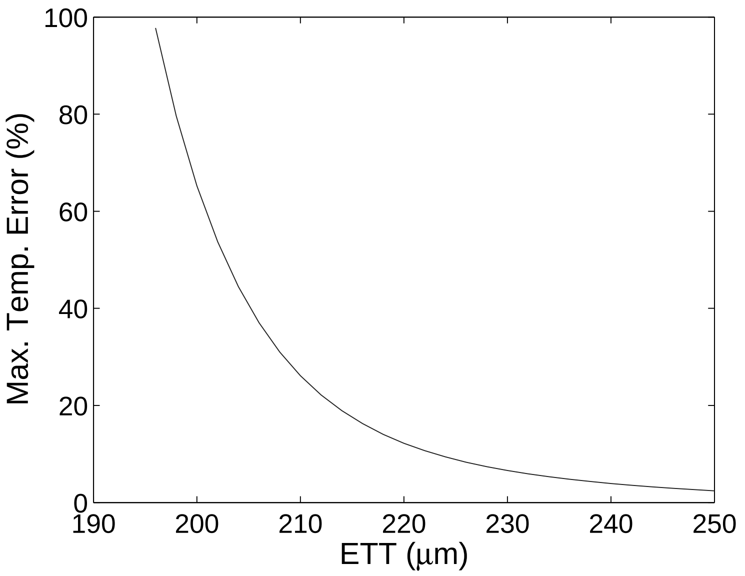
3.3.3 Single-layer Thermal Model Versus Multilayer Thermal Model

Novel 3-D ICs that vertically integrate multiple active layers can significantly reduce interconnect lengths and improve transistor density [9]. However, the thermal management is exacerbated by the low thermal conductivity of bonding layers [36]. For example, one active layer typically has a thermal conductivity of 150 W/(m K), while the bonding material between two active layers has only a thermal conductivity of 0.05 W/(m K). LOTAGre is able to accurately analyze the temperature distribution of 3-D ICs, as it uses the multilayer heat conduction Green's function and is based on the MLT model. The heat conduction path in a 3-D IC or a traditional IC consists of multilayer heterogeneous heat conduction materials. Traditional Green's function-based thermal analysis methods treated the chip heat conduction path as a one-layer structure, by using the SLT model complemented with effective heat transfer rates to the ambient. Fig.3.8(a) demonstrates the SLT model (the rightmost diagram). In the figure, $\bar{h}_e$ is the effective heat transfer rate from the top of the SLT model to the ambient, determined by the approach in [22]; and ETT is named the effective thermal thickness.

To determine the accuracy of the SLT model for the example chip shown in Fig.3.4, the SLT model in the rightmost diagram in Fig.3.8(a) was used in LOTAGre to analyze the temperature distribution of the heat source region. The temperatures obtained by using this SLT model were compared with the temperatures obtained from FLUENT simulation of the MLT model in Fig.3.4(b). The maximum percentage errors of the calculated temperatures based on the SLT model, versus the effective thermal thickness ETT, are plotted in Fig.3.8(b). The figure shows that the accuracy of the SLT model was very sensitive to the effective thermal thickness ETT. For the simulated chip, when $ETT = 250$ μm, the maximum percentage error of the



(a) Multilayer chip, and its single-layer thermal model (the rightmost diagram).



(b) Accuracy of single-layer thermal model versus *ETT*.

Figure 3.8: Single-layer thermal model, and its accuracy versus effective thermal thickness *ETT*.

calculated temperatures using the SLT model was 2.41%; when $ETT = 240\ \mu$m, the maximum percentage error was 3.93%; however, when $ETT = 210\ \mu$m, the maximum percentage error was as large as 26.14%. However, to ensure the temperature errors are within 2.4%, both the chip region and the thermal adhesive should be modeled, which is beyond the capability of the SLT model. Since the active layers in 3-D ICs will become thinner, the use of the SLT model in 3-D IC thermal analysis and optimization will be very limited. In order to use the SLT model, the effective thermal thickness ETT must be determined accurately, because the accuracy of the SLT model is very sensitive to ETT. LOTAGre can be used to estimate ETT, as it has a low time complexity.

3.4 Error Analysis of LOTAGre

LOTAGre utilizes the DCT and IDCT algorithms to achieve the $\mathcal{O}(n \lg n)$ complexity. Consider the inhomogeneous solution. In order to apply the IDCT algorithm, however, the infinite series (3.14) must be truncated to the finite-summation form (3.16). This section analyzes the truncation error of (3.16) and establishes connections between the sampling theory and the discrete heat-source model by the Fourier analysis.

Given a function $u(x, y)$, its Fourier transform $U(\alpha, \beta)$ and the inverse transform are given by

$$U(\alpha, \beta) = \mathcal{F}[u(x, y)] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} u(x, y)\, \phi(x, y, \alpha, \beta)\, dxdy \tag{3.19a}$$

$$u(x, y) = \mathcal{F}^{-1}[U(\alpha, \beta)] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} U(\alpha, \beta)\, \phi^{-1}(x, y, \alpha, \beta)\, d\alpha d\beta \tag{3.19b}$$

where $\phi(x, y, \alpha, \beta) = \exp[-\ddot{\imath}2\pi(\alpha x + \beta y)]$.

3.4.1 Upper Bound of Truncation Error in LOTAGre

The discrete heat-source model describes a power density function, i.e., $f(x', y', z')$, which is defined in the region $x' \in [0, X]$, $y' \in [0, Y]$ and $z' \in [z_{p1}, z_{p2}]$. Let the even expansion of $f(x', y', z')$ be denoted by $\hat{f}(x', y', z')$, which is defined in the expanded region $x' \in [-X, X]$, $y' \in [-Y, Y]$ and $z' \in [z_{p1}, z_{p2}]$. From (3.19a), it can be shown that the F_{ij} defined in (3.13) relates to the Fourier transform of $\hat{f}(x', y', z)$ at the frequency point $\left(\frac{i}{2X}, \frac{j}{2Y}\right)$ by

$$F_{ij} = \frac{ij\pi^2}{4XY} \csc\frac{i\pi}{2X} \csc\frac{j\pi}{2B}\, \mathcal{F}\left[\hat{f}(x', y', z')\right]\Big|_{\alpha=\frac{i}{2X}, \beta=\frac{j}{2Y}}.$$

Unless the power density function $f(x', y', z')$ is band-limited, (3.14) must contain infinite terms. Therefore, the truncation of the infinite series (3.14) will incur numerical errors in LOTAGre.

The truncation error of the series (3.16), being denoted as ϵ_{tr}, is given by

$$\epsilon_{tr} = \sum_{i=A}^{\infty}\sum_{j=1}^{\infty} \epsilon_{tr(i,j)} + \sum_{i=1}^{A-1}\sum_{j=B}^{\infty} \epsilon_{tr(i,j)} + \frac{1}{2}\sum_{i=A}^{\infty} \epsilon_{tr(i,0)} + \frac{1}{2}\sum_{j=B}^{\infty} \epsilon_{tr(0,j)} \tag{3.20}$$

where

$$\epsilon_{tr(i,j)} = F_{ij} I H_{ij} \cos\frac{i\pi(2a+1)}{2A} \cos\frac{j\pi(2b+1)}{2B}.$$

As mentioned in Chapter III, F_{ij} is periodic; therefore, there must exist an upper bound F_{ij}^{max} such that $|F_{ij}| \leq F_{ij}^{max}$, for $i \geq 0$, $j \geq 0$. Before determining the truncation error ϵ_{tr}, first estimate a bound for H_{ij}. The H_{ij} in (2.24) is rewritten to

$$H_{ij}(z|z') = \frac{\xi\left(\underline{Z}_p \cosh\gamma\underline{l} + Z_p \sinh\gamma\underline{l}\right)\left(\bar{Z}_q \cosh\gamma\bar{l} + Z_q \sinh\gamma\bar{l}\right)}{Z_p(\underline{Z}_p + \bar{Z}_p)\cosh\gamma l_p + (Z_p^2 + \underline{Z}_p\bar{Z}_p)\sinh\gamma l_p}. \tag{3.21}$$

Upper and Lower Bounds for $\underline{Z}_p$ and $\bar{Z}_q$

The input impedances $\underline{Z}_p$ and $\bar{Z}_q$ in (3.21) can be bounded based on (2.16). Consider that when $x > 0$, $\tanh x$ is an increasing function of x, and $0 < \tanh x < 1$.

From (2.16), it is clear that for $\gamma \geq \gamma_{min} > 0$, the input impedance Z_{in} satisfies

$$Z_{in}^{min}(ZL) \leq Z_{in} \leq Z_{in}^{max}(ZL)$$

where

$$\begin{aligned} Z_{in}^{max}(ZL) &= ZC\frac{ZL + ZC}{ZC + ZL\tanh\gamma_{min}L} \\ Z_{in}^{min}(ZL) &= ZC\frac{ZL + ZC\tanh\gamma_{min}L}{ZC + ZL}. \end{aligned}$$

Further, it can be shown that when $ZL \geq 0$, both $Z_{in}^{min}(ZL)$ and $Z_{in}^{max}(ZL)$ are increasing functions of ZL.

Therefore, when $\gamma \geq \gamma_{min} > 0$, the upper and lower bounds of $\underline{Z}_p$ can be determined at the maximum and minimum of the loading impedance $\underline{Z}_{p-1}$: $\underline{Z}_p$ satisfies

$$\underline{Z}_p^{min} \leq \underline{Z}_p \leq \underline{Z}_p^{max}$$

where

$$\underline{Z}_p^{max} = \begin{cases} Z_{p-1}\left(Z_{p-1} + \underline{Z}_{p-1}^{max}\right) / \left(Z_{p-1} + \underline{Z}_{p-1}^{max}\tanh\gamma_{min}l_{p-1}\right), & p > 2 \\ Z_1\coth\gamma_{min}l_1, & p = 2 \\ \gamma/\underline{h}, & p = 1 \end{cases} \tag{3.22}$$

and

$$\underline{Z}_p^{min} = \begin{cases} Z_{p-1}\left(\underline{Z}_{p-1}^{min} + Z_{p-1}\tanh\gamma_{min}l_{p-1}\right) / \left(\underline{Z}_{p-1}^{min} + Z_{p-1}\right), & p > 2 \\ Z_1\left(\frac{\gamma_{min}}{\underline{h}} + Z_1\tanh\gamma_{min}l_1\right) / \left(\frac{\gamma_{min}}{\underline{h}} + Z_1\right), & p = 2 \\ \gamma/\underline{h}, & p = 1. \end{cases} \tag{3.23}$$

Similarly, when $\gamma \geq \gamma_{min} > 0$, the upper and lower bounds of $\bar{Z}_q$ can be determined at the maximum and minimum of the loading impedance $\bar{Z}_{q+1}$: $\bar{Z}_q$ satisfies

$$\bar{Z}_q^{min} \leq \bar{Z}_q \leq \bar{Z}_q^{max}$$

where

$$(3.24)\qquad \bar{Z}_q^{max} = \begin{cases} Z_{q+1}\left(Z_{q+1} + \bar{Z}_{q+1}^{max}\right) / \left(Z_{q+1} + \bar{Z}_{q+1}^{max} \tanh \gamma_{min} l_{q+1}\right), & q < n-1 \\ Z_n \coth \gamma_{min} l_n, & q = n-1 \\ \gamma/\bar{h}, & q = n \end{cases}$$

and

$$(3.25)\qquad \bar{Z}_q^{min} = \begin{cases} Z_{q+1}\left(\bar{Z}_{q+1}^{min} + Z_{q+1} \tanh \gamma_{min} l_{q+1}\right) / \left(\bar{Z}_{q+1}^{min} + Z_{q+1}\right), & q < n-1 \\ Z_n\left(\frac{\gamma_{min}}{\bar{h}} + Z_n \tanh \gamma_{min} l_n\right) / \left(\frac{\gamma_{min}}{\bar{h}} + Z_n\right), & q = n-1 \\ \gamma/\bar{h}, & q = n. \end{cases}$$

Upper Bound for ξ

Given the upper or lower bounds for $\underline{Z}_p$ and $\bar{Z}_q$, an upper bound for the ξ in (2.25) can be obtained.

Clearly, for $\gamma \geq \gamma_{min} > 0$,

$$\frac{1}{2}e^{\gamma l_m} \leq \cosh \gamma l_m \leq \frac{1 + e^{-2\gamma_{min} l_m}}{2} e^{\gamma l_m}$$

$$(3.26)\qquad \frac{1 - e^{-2\gamma_{min} l_m}}{2} e^{\gamma l_m} \leq \sinh \gamma l_m \leq \frac{1}{2} e^{\gamma l_m}.$$

Hence

$$\xi \leq \xi^{max} = \frac{\dot{\xi}}{\gamma} \exp\left(-\gamma \sum_{m=p+1}^{q} l_m\right)$$

where

$$(3.27)\qquad \dot{\xi} = \frac{Z_p \prod_{m=p}^{q-1} \bar{Z}_m^{max}}{\prod_{m=p+1}^{q} \frac{1}{2}\left[\bar{Z}_m^{min} + Z_m\left(1 - e^{-2\gamma_{min} l_m}\right)\right]}.$$

Upper Bound for H_{ij}

Given the upper and lower bounds for $\underline{Z}_p$, $\bar{Z}_q$ and ξ, it can be shown that the H_{ij} in (3.21) is bounded by

$$H_{ij}\left(z|z'\right) \leq H_{ij}^{max} = \frac{1}{\gamma}\dot{H}_{ij}\dot{\xi}\exp\left[\gamma\left(z'-z\right)\right] \tag{3.28}$$

where

$$\dot{H}_{ij} = \frac{1}{2}\frac{\left[\underline{c}_h\underline{Z}_p^{max}+Z_p\right]\left[\bar{c}_h\bar{Z}_q^{max}+Z_q\right]}{\left[Z_p\left(\underline{Z}_p^{min}+\bar{Z}_p^{min}\right)+\left(Z_p^2+\underline{Z}_p^{min}\bar{Z}_p^{min}\right)\left(1-e^{-2\gamma_{min}l_p}\right)\right]}. \tag{3.29}$$

Here $\underline{c}_h$ and $\bar{c}_h$ are given by

$$\underline{c}_h = 1+e^{-2\gamma_{min}\left(z_{p1}-z_{p-1}\right)}$$

$$\bar{c}_h = 1+e^{-2\gamma_{min}\left(z_q-z_{q2}\right)}.$$

To simplify the upper bound H_{ij}^{max}, different combinations of p, q, z, z' are considered below.

- $p=1$ and $q<n$

In this case, according to (3.24) and (3.25), $\bar{Z}_q^{max}$ and $\bar{Z}_q^{min}$ are constants, while $\underline{Z}_p^{max}$ and $\underline{Z}_p^{min}$ are given in the form of $\gamma/\underline{h}$. It is clear that $\dot{\xi}$, given in (3.27), is a constant.

Accordingly, the $\dot{H}_{ij}$ in (3.29) satisfies

$$\dot{H}_{ij} \leq \frac{1}{2}\frac{\left(\underline{c}_h+\frac{Z_p}{\gamma_{min}}\underline{h}\right)\left(\bar{c}_h\bar{Z}_q^{max}+Z_q\right)}{Z_p+\bar{Z}_p^{min}\left(1-e^{-2\gamma_{min}l_p}\right)}. \tag{3.30}$$

Consequently, by (3.28), $H_{ij}\left(z|z'\right)$ is bounded by a function in the form of $\alpha e^{\gamma\left(z'-z\right)}/\gamma$:

$$H_{ij}\left(z|z'\right) \leq \alpha e^{\gamma\left(z'-z\right)}/\gamma$$

where α is a coefficient determined by (3.27) and (3.30).

Since the transfer function $H_{ij}\left(z|z'\right)$ is reciprocal, i.e., $H_{ij}\left(z|z'\right)=H_{ij}\left(z'|z\right)$, the above analysis is also applicable when $p>1$ and $q=n$.

- $p = 1$ and $q = n$

In this case, $\underline{Z}_p^{max}$ and $\underline{Z}_p^{min}$ are of the form $\gamma/\underline{h}$; $\bar{Z}_q^{max}$ and $\bar{Z}_q^{min}$ are of the form $\gamma/\bar{h}$.

When $p = q$, $\dot{\xi}$ is a constant. Because $\bar{Z}_p^{min} = \gamma/\bar{h}$, the $\dot{H}_{ij}$ in (3.29) satisfies

$$\dot{H} \leq \frac{1}{2\left(1 - e^{-2\gamma_{min} l_p}\right)} \left(\underline{c}_h + \frac{Z_p}{\gamma_{min}} \underline{h}\right) \left(\bar{c}_h + \frac{Z_q}{\gamma_{min}} \bar{h}\right). \tag{3.31}$$

Hence, from (3.28), $H_{ij}(z|z')$ is also bounded by a function in the form of $\alpha e^{\gamma(z'-z)}/\gamma$, with α determined by (3.27) and (3.31).

When $p < q$, the $\dot{\xi}$ and $\dot{H}$ in (3.27) and (3.29) need to be changed to

$$\begin{aligned} \dot{\xi} &= \frac{Z_p \prod_{m=p}^{q-1} \bar{Z}_m^{max}}{\prod_{m=p+1}^{q-1} \frac{1}{2}\left[\bar{Z}_m^{min} + Z_m\left(1 - e^{-2\gamma_{min} l_m}\right)\right]} \\ \dot{H} &= \frac{\left(\underline{c}_h + \frac{Z_p}{\gamma_{min}} \underline{h}\right) \left(\bar{c}_h + \frac{Z_q}{\gamma_{min}} \bar{h}\right)}{Z_p + \bar{Z}_p^{min}\left(1 - e^{-2\gamma_{min} l_p}\right)}. \end{aligned} \tag{3.32}$$

Then from (3.28), $H_{ij}(z|z')$ is bounded by a function in the form of $\alpha e^{\gamma(z'-z)}/\gamma$ as well, with α determined by (3.32).

- $p > 1$ and $q < n$

In this case, $\dot{\xi}$ in (3.27) and $\dot{H}$ in (3.29) are constants. Then by (3.28), $H_{ij}(z|z')$ is also bounded by a function in the form of $\alpha e^{\gamma(z'-z)}/\gamma$, with α determined by (3.27) and (3.29).

Upper Bound of Truncation Error

As previously demonstrated, $H_{ij}(z|z') \leq \alpha e^{\gamma(z'-z)}/\gamma$, where α is a coefficient contingent on p, q, z, z' and γ_{min}. Then by (3.20), an upper bound for the truncation error ϵ_{tr} can be given:

$$\epsilon_{tr} \leq \epsilon_{tr}^{max} = \frac{\alpha F_{ij}^{max}}{z_{q2} - z_{q1}} \left(\epsilon_{tra} + \epsilon_{trb} + \epsilon_{trc} + \epsilon_{trd}\right), \tag{3.33}$$

with[2]

$$\begin{aligned}
\epsilon_{tra} &= \frac{16AB}{\pi^4}\sum_{i=A}^{\infty}\sum_{j=1}^{\infty}\int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}}\psi_{221}\left(i,j,z'-z\right)dz'dz\\
\epsilon_{trb} &= \frac{16AB}{\pi^4}\sum_{i=1}^{A-1}\sum_{j=B}^{\infty}\int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}}\psi_{221}\left(i,j,z'-z\right)dz'dz\\
\epsilon_{trc} &= \frac{2AX}{B}\sum_{i=A}^{\infty}\int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}}e^{i\pi(z'-z)/X}\sin^2\frac{i\pi}{2A}/\left(i\pi\right)^3dz'dz\\
\epsilon_{trd} &= \frac{2BY}{A}\sum_{j=B}^{\infty}\int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}}e^{j\pi(z'-z)/Y}\sin^2\frac{j\pi}{2B}/\left(j\pi\right)^3dz'dz.
\end{aligned}$$

Here $\psi_{abc}\left(i,j,x\right) = e^{\gamma x}\sin^2\frac{i\pi}{2A}\sin^2\frac{j\pi}{2B}/i^aj^b\gamma^c$. As $\gamma = \sqrt{\left(i\pi/X\right)^2+\left(j\pi/Y\right)^2}$, to calculate α, let $\gamma_{min} = \pi\cdot\min\left(A/X,B/Y\right)$.

Let $S_{dbl}\left[a,b,c,x\right]$ denote the double summation of $\psi_{abc}\left(i,j,x\right)$:

$$S_{dbl}\left[a,b,c,x\right] = \sum_{i=1}^{A-1}\sum_{j=B}^{\infty}\psi_{abc}\left(i,j,x\right)+\sum_{i=A}^{\infty}\sum_{j=1}^{\infty}\psi_{abc}\left(i,j,x\right).$$

Then $\epsilon_{tra}+\epsilon_{trb}$, denoted by ϵ_{trab}, is represented by

$$\begin{aligned}
\epsilon_{trab} &= \epsilon_{tra}+\epsilon_{trb}\\
&= \frac{16AB}{\pi^4}\left\{S_{dbl}\left[2,2,3,z_{p2}-z_{q1}\right]-S_{dbl}\left[2,2,3,z_{p2}-z_{q2}\right]\right.\\
&\qquad\left.-S_{dbl}\left[2,2,3,z_{p1}-z_{q1}\right]+S_{dbl}\left[2,2,3,z_{p1}-z_{q2}\right]\right\}. \qquad (3.34)
\end{aligned}$$

Let $S_{sgl}\left[a,\rho,\theta,K\right]$ denote the single summation below:

$$S_{sgl}\left[a,\rho,\theta,K\right] = \sum_{k=K}^{\infty}\frac{e^{k\rho}}{k^a}\sin^2\left(\frac{k\theta}{2}\right).$$

[2]Here assume that $z_{p1} < z_{p2} < z_{q1} < z_{q2}$. When $z_{p1} = z_{q1}$ and $z_{p2} = z_{q2}$, ϵ_{tra} – ϵ_{trd} need to be determined from $\int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}}H_{ij}\left(z|z'\right)dz'dz \leq \frac{2\alpha}{\gamma^3}\left[\gamma\left(z_{q2}-z_{q1}\right)+e^{\gamma\left(z_{q1}-z_{q2}\right)}-1\right]$.

Then ϵ_{trc} is represented by

$$\begin{aligned}\epsilon_{trc} = \frac{2AX^3}{B\pi^5} \Big\{ & S_{sgl}\left[5, \frac{\pi}{X}\left(z_{p2} - z_{p1}\right), \frac{\pi}{A}, A\right] \\ & - S_{sgl}\left[5, \frac{\pi}{X}\left(z_{p2} - z_{q2}\right), \frac{\pi}{A}, A\right] \\ & - S_{sgl}\left[5, \frac{\pi}{X}\left(z_{p1} - z_{q1}\right), \frac{\pi}{A}, A\right] \\ & - S_{sgl}\left[5, \frac{\pi}{X}\left(z_{p1} - z_{q2}\right), \frac{\pi}{A}, A\right]\Big\}, \end{aligned} \tag{3.35}$$

and ϵ_{trd} is represented by

$$\begin{aligned}\epsilon_{trd} = \frac{2BY^3}{A\pi^5} \Big\{ & S_{sgl}\left[5, \frac{\pi}{Y}\left(z_{p2} - z_{p1}\right), \frac{\pi}{B}, B\right] \\ & - S_{sgl}\left[5, \frac{\pi}{Y}\left(z_{p2} - z_{q2}\right), \frac{\pi}{B}, B\right] \\ & - S_{sgl}\left[5, \frac{\pi}{Y}\left(z_{p1} - z_{q1}\right), \frac{\pi}{B}, B\right] \\ & - S_{sgl}\left[5, \frac{\pi}{Y}\left(z_{p1} - z_{q2}\right), \frac{\pi}{B}, B\right]\Big\}. \end{aligned} \tag{3.36}$$

With (3.34), (3.35), and (3.36), the upper bound of truncation error ϵ_{tr}^{max} in (3.33) can be computed.

3.4.2 Computation of Upper Bound of Truncation Error

Computation of $S_{sgl}\left[a, \rho, \theta, K\right]$

To compute ϵ_{tr}^{max}, first consider the single summation $S_{sgl}\left[a, \rho, \theta, K\right]$. It is reformulated to

$$S_{sgl}\left[a, \rho, \theta, K\right] = \sum_{k=K}^{W-1} \frac{e^{k\rho}}{k^a} \sin^2\left(\frac{k\theta}{2}\right) + \sum_{k=W}^{\infty} \frac{e^{k\rho}}{k^a} - \sum_{k=W}^{\infty} \frac{e^{k\rho}}{k^a} \frac{1 + \cos k\theta}{2} \tag{3.37}$$

where W is an integral multiple of $2\pi/\theta$. In the above formula, the second right-hand-side term can be reformulated to

$$\sum_{k=W}^{\infty} \frac{e^{k\rho}}{k^a} = \mathrm{Li}_a\left(e^{\rho}\right) - \sum_{k=1}^{W-1} \frac{e^{k\rho}}{k^a} \tag{3.38}$$

where $\mathrm{Li}_a\left(\cdot\right)$ is the poly-logarithm function: $\mathrm{Li}_a\left(x\right) = \sum_{k=1}^{\infty} x^k / k^a$.

Consider the last right-hand-side term in (3.37). Clearly, π is an integral multiple of θ. Then let $\kappa = \pi/\theta$, where κ is an integer. There is a lemma.

Lemma III.1. *When $\rho \leq 0$,*

$$0 \leq \sum_{k=W}^{W+2\kappa-1} \frac{e^{k\rho}}{k^a}\frac{1+\cos k\theta}{2} - \int_W^{W+2\kappa} \frac{e^{x\rho}}{x^a}\frac{1+\cos x\theta}{2}dx \leq \left.\frac{e^{k\rho}}{k^a}\frac{1+\cos k\theta}{2}\right|_{W+2\kappa}^{W}. \tag{3.39}$$

Proof. Rewrite

$$\int_W^{W+2\kappa} \frac{e^{x\rho}}{x^a}\frac{1+\cos x\theta}{2}dx = \int_W^{W+\kappa} \left[\left(\frac{e^{x\rho}}{x^a} - \frac{e^{(x+\kappa)\rho}}{(x+\kappa)^a}\right)\frac{1+\cos x\theta}{2} + \frac{e^{(x+\kappa)\rho}}{(x+\kappa)^a}\right]dx.$$

For $x \in [W, W+\kappa]$, both $e^{x\rho}/x^a - e^{(x+\kappa)\rho}/(x+\kappa)^a$ and $(1+\cos x\theta)/2$, being non-negative, are decreasing functions of x. Further, $e^{(x+\kappa)\rho}/(x+\kappa)^a$ is also a decreasing function of x. Hence, the right-hand-side integrand in the above formula must be a decreasing function of x. Consequently,

$$\begin{aligned}\int_W^{W+2\kappa} \frac{e^{x\rho}}{x^a}\frac{1+\cos x\theta}{2}dx &\leq \sum_{k=W}^{W+k-1} \left[\left(\frac{e^{k\rho}}{k^a} - \frac{e^{(k+\kappa)\rho}}{(k+\kappa)^a}\right)\frac{1+\cos k\theta}{2} + \frac{e^{(k+\kappa)\rho}}{(k+\kappa)^a}\right] \\ &= \sum_{k=W}^{W+2k-1} \frac{e^{k\rho}}{k^a}\frac{1+\cos k\theta}{2}\end{aligned}$$

and

$$\begin{aligned}\int_W^{W+2\kappa} \frac{e^{x\rho}}{x^a}\frac{1+\cos x\theta}{2}dx &\geq \sum_{k=W+1}^{W+k} \left[\left(\frac{e^{k\rho}}{k^a} - \frac{e^{(k+\kappa)\rho}}{(k+\kappa)^a}\right)\frac{1+\cos k\theta}{2} + \frac{e^{(k+\kappa)\rho}}{(k+\kappa)^a}\right] \\ &= \sum_{k=W+1}^{W+2k} \frac{e^{k\rho}}{k^a}\frac{1+\cos k\theta}{2}.\end{aligned}$$

□

Since (3.39) can be generalized to address any summation in the form of

$$\sum_{k=W+2m\kappa}^{W+2m\kappa+\kappa} \frac{e^{k\rho}}{k^a}\frac{1+\cos k\theta}{2},$$

where m is an integer, a lemma follows.

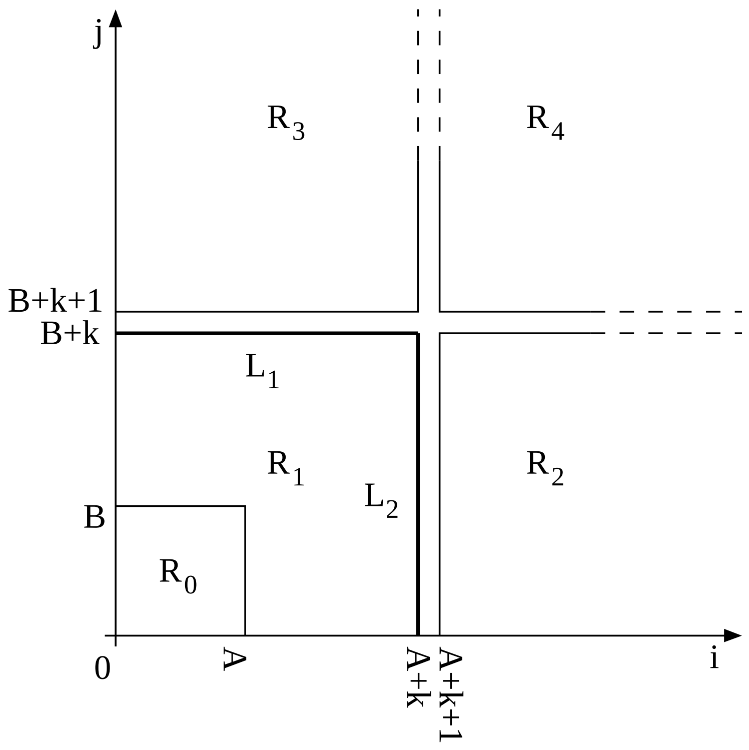


Figure 3.9: Illustration of computation of $S_{dbl}\left[a,b,c,x\right]$.

Lemma III.2. *When $\rho \leq 0$,*

$$0 \quad \leq \quad \sum_{k=W}^{\infty} \frac{e^{k\rho}}{k^a}\frac{1+\cos k\theta}{2} \quad - \int_W^{\infty} \frac{e^{x\rho}}{x^a}\frac{1+\cos x\theta}{2}dx \quad \leq \quad \frac{e^{W\rho}}{W^a}\frac{1+\cos W\theta}{2}. \tag{3.40}$$

By (3.38) and (3.40), $S_{sgl}\left[a,\rho,\theta,K\right]$ can be approximated by

$$S_{sgl}\left[a,\rho,\theta,K\right] \approx \mathrm{Li}_a\left(e^{\rho}\right) - \sum_{k=1}^{K-1}\frac{e^{k\rho}}{k^a} - \sum_{k=K}^{W-1}\frac{e^{k\rho}}{k^a}\frac{1+\cos k\theta}{2} - \int_W^{\infty}\frac{e^{x\rho}}{x^a}\frac{1+\cos x\theta}{2}dx \tag{3.41}$$

with an absolute error no more than

$$\frac{e^{W\rho}}{W^a}\frac{1+\cos W\theta}{2}.$$

In order to meet the error tolerance, a sufficiently large W can be chosen.

Computation of $S_{dbl}[a,b,c,x]$

As shown in Fig.3.9, $S_{dbl}[a,b,c,x]$ corresponds to the summation of $\psi_{abc}(i,j,x)$ for all i and j in the regions $R_1 - R_4$. Let $S_{dbl}[a,b,c,x]$ be approximated by

$$S_{dbl}[a,b,c,x] \approx \sum_{i,j\in R_1} \psi_{abc}(i,j,x)\,. \tag{3.42}$$

Then the approximation error, denoted as ϵ_{dbl}, is given by

$$\epsilon_{dbl} = \sum_{i,j\in R_2\cup R_3\cup R_4} \psi_{abc}(i,j,x)\,.$$

To estimate ϵ_{dbl}, use the following inequality: for $x<0$,

$$e^{\gamma x} \le e^{(s_i i + s_j j)x}, \text{ with } s_i = \frac{\pi}{\sqrt{2}}\frac{1}{X} \text{ and } s_j = \frac{\pi}{\sqrt{2}}\frac{1}{Y}, \tag{3.43}$$

which holds because

$$\gamma = \sqrt{\left(\frac{i\pi}{X}\right)^2 + \left(\frac{j\pi}{Y}\right)^2} \ge \frac{\pi}{\sqrt{2}}\left(\frac{i}{X}+\frac{j}{Y}\right).$$

By (3.43),

$$\sum_{i,j\in R_2} \psi_{abc}(i,j,x) = \sum_{i=A+k+1}^{\infty}\sum_{j=1}^{B+k} \psi_{abc}(i,j,x) \le \epsilon^{max}_{dbl(R_2)} \tag{3.44}$$

where

$$\begin{aligned}\epsilon^{max}_{dbl(R_2)} &= \sum_{i=A+k+1}^{\infty} \frac{e^{s_i i x}\sin^2\frac{i\pi}{2A}}{i^a} \sum_{j=1}^{B+k} \frac{e^{(s_j j + s_i)x}\sin^2\frac{j\pi}{2B}}{j^b\gamma^c_{(A+k)j}} \\ &= S_{sgl}\left[a, s_i x, \frac{\pi}{A}, A+k+1\right] \sum_{j\in L_2} \frac{e^{(s_j j + s_i)x}\sin^2\frac{j\pi}{2B}}{j^b\gamma^c_{(A+k)j}}.\end{aligned}$$

Here L_2 is the set of js falling on the vertical line $i = A+k$, $j = 1,\cdots,B+k$.

By the same means,

$$\sum_{i,j\in R_3} \psi_{abc}(i,j,x) = \sum_{j=B+k+1}^{\infty}\sum_{i=1}^{A+k} \psi_{abc}(i,j,x) \le \epsilon^{max}_{dbl(R_3)}, \tag{3.45}$$

where

$$\epsilon^{max}_{dbl(R_3)} = S_{sgl}\left[b, s_j x, \frac{\pi}{B}, B+k+1\right] \sum_{i=1}^{A+k} \frac{e^{(s_i i + s_j)x} \sin^2 \frac{i\pi}{2A}}{i^a \gamma^c_{i(B+k)}},$$

and

$$\sum_{i,j \in R_4} \psi_{abc}(i,j,x) = \sum_{i=A+k+1}^{\infty} \sum_{j=B+k+1}^{\infty} \psi(i,j) \le \epsilon^{max}_{dbl(R_4)}, \tag{3.46}$$

where

$$\epsilon^{max}_{dbl(R_4)} = \frac{S_{sgl}\left[a, s_i x, \frac{\pi}{A}, A+k+1\right] \cdot S_{sgl}\left[b, s_j x, \frac{\pi}{B}, B+k+1\right]}{\gamma^c_{(A+k+1)(B+k+1)}}.$$

In summary,

$$\epsilon_{dbl} \le \epsilon^{max}_{dbl} = \epsilon^{max}_{dbl(R_2)} + \epsilon^{max}_{dbl(R_3)} + \epsilon^{max}_{dbl(R_2)}. \tag{3.47}$$

When k is sufficiently large, ϵ^{max}_{dbl} will meet the given error tolerance. As a result, $S_{dbl}[a,b,c,x]$ can be satisfactorily approximated by the double summation of $\psi_{abc}(i,j,x)$ for i and j in the region R_1.

The above describes the approaches to compute $S_{sgl}[a,\rho,\theta,K]$ and $S_{dbl}[a,b,c,x]$. With these approaches, the numerical value for the upper bound of the truncation error, ϵ^{max}_{tr}, can be obtained from (3.33), (3.34), (3.35) and (3.36). Accordingly, a numerical program was developed to calculate the upper bound of the truncation error in LOTAGre. For the results in Table 3.2 for the example chip in Fig.3.4, Table 3.4 shows the upper bounds of the truncation errors for the temperatures computed by LOTAGre for the heat-source region, and Table 3.5 shows the upper bounds of the truncation errors for the temperatures computed by LOTAGre for the top surface of the chip.

Since LOTAGre uses the first $A \times B$ terms in (3.14) to compute the temperature distribution of the chip, the next $15A \times B$ terms in (3.14) were also used to estimate

Table 3.4: Upper bounds of truncation errors for temperatures computed by LOTAGre in the heat-source region of the example chip in Fig.3.4 under wide parameter variations.

$\underline{h}$ (W/m^2 K)	k_2 (W/m K)	h (W/m^2 K)	$\underline{I} - Max$ °C	ϵ_{tr}^{max} °C	ϵ_{tr}^{sum} °C
257	12.2	8675	59.44	0.8568	0.6900
5212	32.4	5419	49.51	0.8516	0.6897
9826	16.2	2371	43.94	0.8562	0.6893
1387	5.1	9451	51.71	0.8594	0.6899
2715	382	7213	52.51	0.8304	0.6898
1462	1715	8415	52.88	0.8247	0.6899
512	10.4	800.1	340.9	0.8574	0.6900
1128	15.3	3410	105.5	0.8559	0.6899
1682	60.1	13215	38.58	0.8462	0.6899
21	70.2	9898	53.56	0.8446	0.6900

Table 3.5: Upper bounds of truncation errors for temperatures computed by LOTAGre on the top surface of the example chip in Fig.3.4 under wide parameter variations.

$\underline{h}$ (W/m^2 K)	k_2 (W/m K)	h (W/m^2 K)	$\bar{I} - Max$ °C	ϵ_{tr}^{max} 10^{-16}°C	ϵ_{tr}^{sum} 10^{-16}°C
257	12.2	8675	48.97	3.464	2.667
5212	32.4	5419	41.02	7.196	5.538
9826	16.2	2371	35.82	4.371	3.360
1387	5.1	9451	40.23	1.594	1.227
2715	382	7213	43.98	10.38	8.074
1462	1715	8415	44.16	4.032	3.162
512	10.4	800.1	331.6	3.024	2.329
1128	15.3	3410	95.98	4.173	3.214
1682	60.1	13215	29.49	10.01	7.709
21	70.2	9898	44.26	10.64	8.202

the truncation error:

$$\epsilon_{tr}^{sum} = F_{ij}^{max}\left(\sum_{i=0}^{4A-1}\sum_{j=0}^{4B-1} 2^{-\delta_{i0}-\delta_{j0}} IH_{ij} - \sum_{i=0}^{A-1}\sum_{j=0}^{B-1} 2^{-\delta_{i0}-\delta_{j0}} IH_{ij}\right).$$

Table 3.4 and Table 3.5 give the corresponding values of ϵ_{tr}^{sum}. Compared to those ϵ_{tr}^{max}s, the shown ϵ_{tr}^{sum}s give some clue to the magnitudes of the truncation errors, but unlike ϵ_{tr}^{max}s, they cannot bound the truncation errors.

3.4.3 Accuracy of Discrete Heat-Source Model

Let the power density function $f(x', y', z')$ be uniform in the heat source region $[z_{p1}, z_{p2}]$. Then $f(x', y', z')$ and its even expansion $\hat{f}(x', y', z')$ can be simply written as $f(x', y')$ and $\hat{f}(x', y')$, respectively. From (2.9), the average inhomogeneous temperature in the target region $[z_{q1}, z_{q2}]$ at the location (x, y), $T^i(x, y)$, is obtained:

$$T^i(x,y) = \frac{1}{4XY}\sum_{i=0}^{\infty}\sum_{j=0}^{\infty} 2^{2-\delta_{i0}-\delta_{j0}} \hat{F}\left(\frac{i}{2X}, \frac{j}{2Y}\right)\tilde{H}_{ij}\phi_{ij}(x,y) \tag{3.48}$$

where

$$\hat{F}(\alpha,\beta) = 4\int_0^X\int_0^Y f(x', y')\,\phi(x', y', \alpha, \beta)\,dx'dy'$$
$$\tilde{H}_{ij}(z) = \int_{z_{q1}}^{z_{q2}}\int_{z_{p1}}^{z_{p2}} H_{ij}(z|z')\,dz'dz.$$

According to (3.19a), $\hat{F}\left(\frac{i}{2X}, \frac{j}{2Y}\right)$ is actually the Fourier transform of $\hat{f}(x', y')$ at the frequency point $\left(\frac{i}{2X}, \frac{j}{2Y}\right)$:

$$\hat{F}\left(\frac{i}{2X}, \frac{j}{2Y}\right) = \mathcal{F}\left[\hat{f}(x', y')\right]\Big|_{\left(\frac{i}{2X}, \frac{j}{2Y}\right)}.$$

When $f(x', y')$ is a power density function under the discrete heat-source model, $\hat{F}\left(\frac{i}{2X}, \frac{j}{2Y}\right)$ relates to F_{ij}, given in (3.13), by

$$\hat{F}\left(\frac{i}{2X}, \frac{j}{2Y}\right) = \frac{4XY}{ij\pi^2}\sin\frac{i\pi}{2A}\sin\frac{j\pi}{2B}F_{ij}.$$

According to (3.19b), the Fourier series (3.48) is actually the inverse Fourier transform of $\hat{F}(\alpha,\beta)$ multiplied by an infinite-delta sequence:

$$(3.49)\quad T^i(x,y) = \mathcal{F}^{-1}\left[\hat{F}(\alpha,\beta) \underset{\rightarrow}{\times} \frac{1}{4XY}\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}\tilde{H}_{ij}(z)\,\delta\left(\alpha-\frac{i}{2X},\beta-\frac{j}{2Y}\right)\right].$$

In this dissertation, $\underset{\rightarrow}{\times}$ denotes multiplication and $\underset{\rightarrow}{\otimes}$ denotes convolution.

The discrete heat-source model shown in Fig.3.1 is an approximation to the actual power density distribution of the chip. To preserve the total power of the chip, the total power inside each cubic cell of the discrete heat-source model needs to match that of the corresponding region in the chip. In the frequency domain, the preservation of total power by the discrete heat-source model can be analyzed as follows:

1. Convolute $\hat{f}(x',y')$, the even expansion of the power density function, with a 2-D window function of dimensions $\frac{X}{A}\times\frac{Y}{B}$ and strength $\frac{AB}{XY}$. The window function is denoted by $W_{\frac{X}{A}\times\frac{Y}{B},\frac{AB}{XY}}(x',y')$ where

$$(3.50)\qquad W_{a\times b,c}(x',y') = \begin{cases} c, & |x'|\le\frac{a}{2}, |y'|\le\frac{b}{2} \\ 0, & \text{otherwise.}\end{cases}$$

2. Sample the result from Step 1 at the locations $\left(\frac{aX}{A}+\frac{X}{2A},\frac{bY}{B}+\frac{Y}{2B}\right)$ by an infinite-delta sequence $\delta^*(x',y')$, which is defined by

$$(3.51)\qquad \delta^*(x',y') = \sum_{a=-\infty}^{\infty}\sum_{b=-\infty}^{\infty}\delta\left(x'-\left(a+\frac{1}{2}\right)\frac{X}{A},y'-\left(b+\frac{1}{2}\right)\frac{Y}{B}\right).$$

3. Convolute the result from Step 2 with $W_{\frac{X}{A}\times\frac{Y}{B},1}(x',y')$.

The Fourier transform of the window function (3.50) is in the well-known form:

$$\mathcal{F}\left[W_{a\times b,c}(x',y')\right] = abc\,\mathrm{sinc}(\alpha a)\,\mathrm{sinc}(\beta b)$$

where $\operatorname{sinc}(x) = \sin(\pi x)/\pi x$. The Fourier transform of $\delta^*(x', y')$, the infinite-delta sequence in (3.51), is given by

$$\begin{aligned}
\mathcal{F}\left[\delta^*(x', y')\right] &= e^{-\ddot{\imath}\pi(\alpha X/A+\beta Y/B)}\mathcal{F}\left[\sum_{a=-\infty}^{\infty}\sum_{b=-\infty}^{\infty}\delta\left(x' - \frac{aX}{A}, y' - \frac{bY}{B}\right)\right] \\
&= e^{-\ddot{\imath}\pi(\alpha X/A+\beta Y/B)}\frac{AB}{XY}\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}\delta\left(\alpha - \frac{iA}{X}, \beta - \frac{jB}{Y}\right) \\
&= \frac{AB}{XY}\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}(-1)^{i+j}\,\delta\left(\alpha - \frac{iA}{X}, \beta - \frac{jB}{Y}\right).
\end{aligned}$$

Therefore, in the frequency domain, the power density distribution under the discrete heat-source model, denoted by $f_{hs}(x', y')$, relates to $f(x', y')$, the actual power density distribution of the chip, by

$$\begin{aligned}
\mathcal{F}\left[\hat{f}_{hs}(x', y')\right] &= \hat{F}(\alpha, \beta)\operatorname{sinc}\left(\frac{\alpha X}{A}\right)\operatorname{sinc}\left(\frac{\beta Y}{B}\right) \\
&\underset{\to}{\otimes}\frac{AB}{XY}\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}(-1)^{i+j}\,\delta\left(\alpha - \frac{iA}{X}, \beta - \frac{jB}{Y}\right) \\
&\underset{\to}{\times}\frac{XY}{AB}\operatorname{sinc}\left(\frac{\alpha X}{A}\right)\operatorname{sinc}\left(\frac{\beta Y}{B}\right)
\end{aligned} \tag{3.52}$$

where $\hat{f}_{hs}(x', y')$ is the even expansion of $f_{hs}(x', y')$.

According to (3.52), the inhomogeneous temperature distribution computed based on the discrete heat-source model differs from the actual inhomogeneous temperature distribution of the chip for the following reasons. First, under-sampling the actual power density distribution can cause the temperature differences. When $\hat{f}(x', y')$ is band-limited with α^{max} and β^{max} being the respective maximum frequencies, to ensure that the sampling is sufficient, A and B in the discrete heat-source model should satisfy

$$A \geq 2\alpha^{max}X, B \geq 2\beta^{max}Y. \tag{3.53}$$

If A and B are smaller than $2\alpha^{max}X$ and $2\beta^{max}Y$, the high-frequency components of $f(x', y')$ can fold around half the sampling frequencies, leading to frequency aliasing.

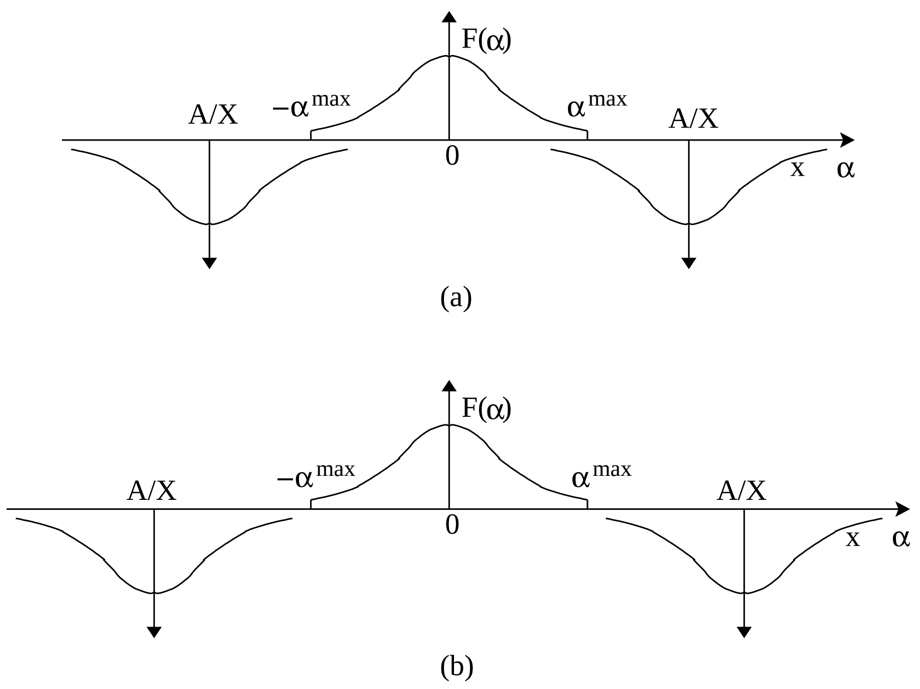


Figure 3.10: Illustration of under-sampling (1-D version). $F(\alpha)$ is the Fourier transform of a function in 1-D space. The figure shows the convolution of $F(\alpha)$ with the Fourier spectrum of the 1-D version of the infinite-delta sequence (3.51). (a) Under-sampling, and (b) sufficiently sampling.

Fig.3.10 uses a 1-D example to illustrate the under-sampling issue. The other reason is that the frequency spectrum $\mathcal{F}\left[\hat{f}_{hs}(x', y')\right]$ in the discrete heat-source model is the actual frequency spectrum $F\left[\hat{f}(x', y')\right]$ modulated by the sinc functions twice.

In summary, two types of errors exist in LOTAGre. One type of error is the sampling error that occurs when the discrete heat-source model is used to approximate the actual power density distribution of the chip. The sampling resolution of the discrete heat-source model should satisfy (3.53), which is the same as that required by the general sampling theory, albeit here the sampling method is quite different. The other type of error is the truncation error that occurs when LOTAGre truncates the fully analytical series solutions to the temperature distribution of the chip into finite-summation forms. The previous theoretical analysis and experimental results have demonstrated that the truncation error of LOTAGre is insignificant.

CHAPTER IV

Interconnect Thermal Modeling

4.1 Overview

Temperature impacts on-chip interconnect wires primarily in two ways. First, temperature affects interconnect timing. With progressive technology scaling, delays continue to increase in propagating signals through on-chip interconnect wires because of the shrinking interconnect cross-sectional areas and the increasing interconnect lengths [60]. To reduce the signal propagation delay of a critical path, buffers must be appropriately inserted into the related interconnect wires, with the consideration of interconnect RC, RLC, or transmission line effects [6]. Because temperature affects interconnect conductivity, the delays may be different when propagating signals through a set of even identically shaped interconnect wires that are at regions of different temperatures. In order to avoid timing failures, the buffer-insertion stage must consider temperature gradients within the chip. Similarly, temperature gradients must also be considered to contain the clock skews when designing an on-chip clock distribution network that exposes to a large portion of the chip. Precisely matching the clock network geometry may not ensure the timely delivery of signals to the clock sinks, because of the underlying temperature gradients [4].

Second, temperature affects interconnect electromigration. Voids or open circuits can occur in a metal wire because of the transport of the metal ions activated by the electron winds resulting from the flowing current. This electromigration-induced MTTF for a metal wire is generally depicted by the Black's equation:

$$MTTF\,(T) = AJ^{-2}e^{\phi/kT}$$

where J is the current density, ϕ is the activation energy, and A is a technology-dependent parameter [13]. Accordingly, the MTTF of a metal wire is affected by both the temperature and the average current density. It reduces exponentially with increasing temperature: for example, a temperature difference of 9.4°C leads to 30 percent difference in the MTTF of a metal wire for an activation energy of 0.55 eV [58]. To improve the MTTF, the average current density must also meet the design rules for the interconnect wire. In very deep sub-micron technology nodes, miniaturizing interconnect cross-sectional dimensions reduces the heat dissipation areas and increases the thermal impedances from interconnect wires to the heat sink of the chip. Therefore, manifesting as the power dissipation, the root mean square current density instead of the average current density determines the lifetime of an interconnect wire [8].

To alleviate the timing and reliability issues, it is necessary to accurately compute the temperature distribution within an interconnect wire. In [19] the FD method was used to analyze the temperatures of power lines. When the power lines were paralleled by the signal lines, as in Fig.4.1(a), simulation showed that thermal coupling from the signal lines reduced the temperatures of the nearby power lines by negligible amounts, less than 3% for the typical spacings in an IC. When the power lines were orthogonal to the signal lines, as in Fig.4.1(b), for a fixed ratio of width to separation, w_2/s, the maximum temperature increased in the power lines with the increase of s. For multilevel interconnect configuration, simulation manifested

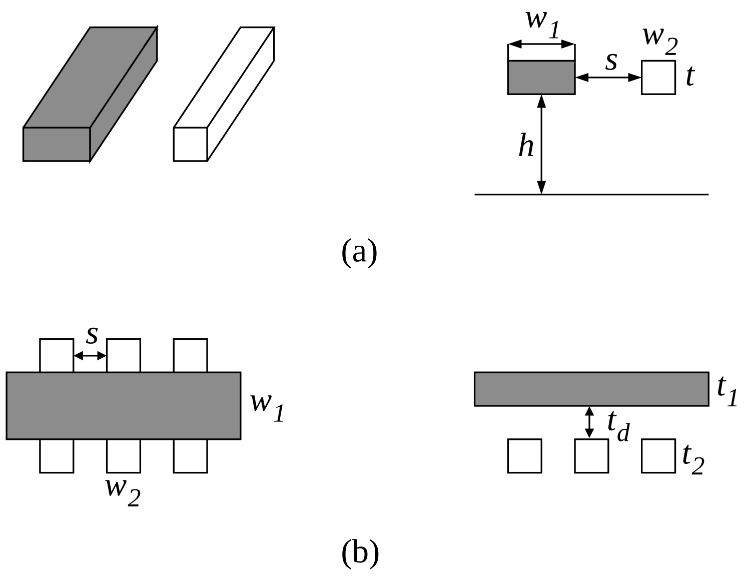


Figure 4.1: (a) Parallel and (b) orthogonal interconnect configurations (gray boxes are power lines and blank boxes are signal lines).

that the nearby metal levels mainly affected the temperature distributions of power lines. Guided by such simulation results from the FD and FE methods, the temperature distribution of the entire on-chip interconnect network is usually analyzed by firstly partitioning the interconnect network into individual wire segments and then determining the temperature distribution of each wire segment separately [19, 23]. Consequently, the Schafft's model, which originally modeled only the interconnect electromigration [58], becomes widely accepted in modeling the temperature distribution of an interconnect wire. For example, the Schafft's model has been used to analyze the clock skews induced by the substrate temperature gradients [4]. To improve the precision, 2-D thermal characterization can be used to determine the parameters of the 1-D Schafft's model as well as its variants, namely the Schafft-type models [12, 58, 33, 4].

The temperature distribution of an interconnect wire is affected by many factors, including the chip packaging, ambient temperatures, and multiple heat conduction

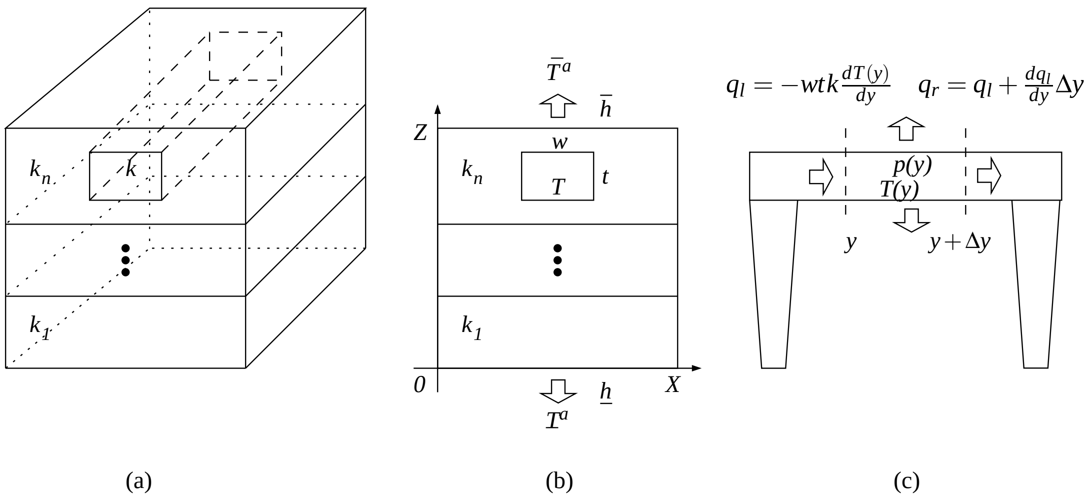


Figure 4.2: 1-D interconnect temperature distribution model: (a) 3-D interconnect configuration; (b) 2-D modeling of heat dissipation in interconnect cross-sectional area; (c) 1-D interconnect temperature distribution model in longitudinal direction (to be general, assume two vias are at the two line ends).

paths in the chip. Based on the original Schafft's model, this chapter introduces an accurate 1-D interconnect temperature distribution model.

4.2 Interconnect Temperature Distribution Model

As shown in Fig.4.2, this section introduces a 1-D interconnect temperature distribution model. Similar to the original Schafft's model, the introduced 1-D temperature distribution model assumes that heat is either vertically dissipated through the insulation materials around the interconnect or conducted along the interconnect longitudinal direction [58]. The 1-D temperature distribution model can be established in two steps for an interconnect wire embedded in a 3-D structure. First, the 2-D heat conduction equation is solved for the interconnect cross-sectional area to estimate the thermal impedances between the interconnect wire and its surrounding materials [see Fig.4.2 (b)]. With the estimated thermal impedances, a formula results for the amount of heat lost at the wire location y vertically through the surrounding insulation materials, denoted by $p_v(y)$:

$$p_v(y) = \frac{T(y)}{R} - \frac{\bar{T}^a(y)}{\bar{R}} - \frac{\underline{T}^a(y)}{\underline{R}} \tag{4.1}$$

where $\bar{T}^a(y)$ and $\underline{T}^a(y)$ are the top and bottom ambient temperatures of the chip at the location y; R, $\bar{R}$, and $\underline{R}$ are the self thermal impedance of the interconnect, and the thermal impedances from the interconnect to the top and bottom surfaces of the chip, respectively. Compared to the $p_v(y)$ in (4.1), the vertical heat loss considered in the traditional Schafft-type models relied upon only one of the two ambient temperatures, for example, the substrate temperature [58, 4]. Consequently, the traditional models included only one heat conduction path (downward heat dissipation) and also excluded the heat transfer rate between the bottom surface of the chip and the ambient environment, i.e. $\underline{h}$.

Furthermore, the traditional Schafft-type models neglected the effect of the temperature gradients in the interconnect wire. Consider the three identical heat conduction plates in Fig.4.3. If the three plates have the same temperature distribution at their boundaries, their interior temperature distributions must also be the same because they satisfy the same 2-D Laplace's equation. Therefore, in this case, there is no heat flow among the three heat conduction plates when they are attached together. This is exactly an assumption under the traditional Schafft-type models. However, when the three plates have different temperatures at their boundaries, e.g., $T_l < T_r < T_m$, they must have similar interior temperature gradients. That is, at an interior location, the temperature of the left plate is the lowest and that of the middle one is the highest. Therefore, when the three heat conduction plates are attached together, heat flows from the middle one to the left and right ones, and the total vertical heat loss of the middle one increases. The above example demonstrates that the interconnect temperature gradients can affect the amount of heat dissipated

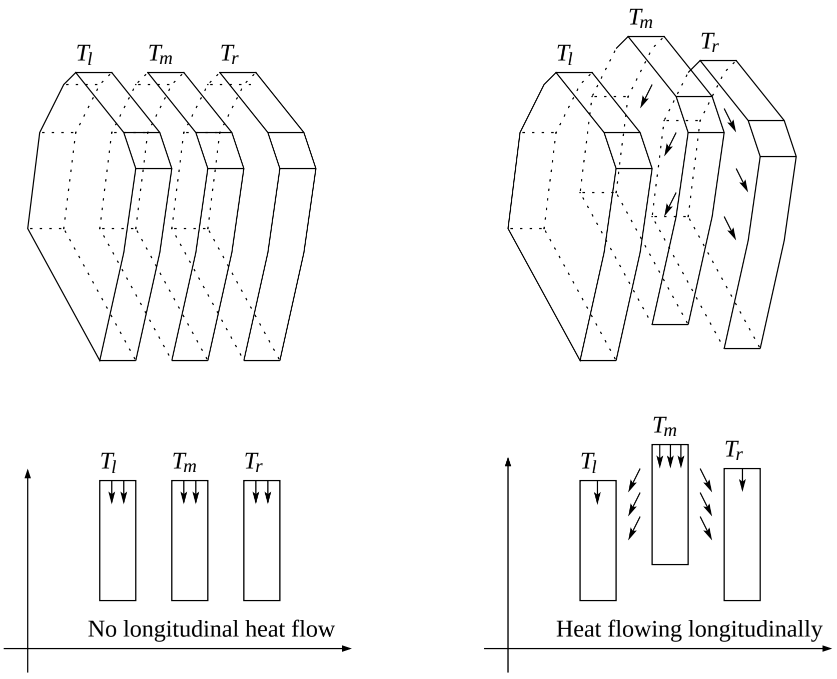


Figure 4.3: Effect of temperature gradients on interconnect vertical heat dissipation.

vertically from the interconnect wire. Without considering such an effect, the traditional Schafft-type models tended to overestimate the temperature gradients in the interconnect wire. Therefore, a new vertical heat loss model, denoted by $p_v^+(y)$, is introduced, which linearly approximates the effect of temperature gradients:

$$p_v^+(y) = p_v(y) - \beta_1 \frac{\partial^2 T(y)}{\partial y^2} + \beta_2 \frac{\partial^2 \bar{T}^a(y)}{\partial y^2} + \beta_3 \frac{\partial^2 \underline{T}^a(y)}{\partial y^2}. \tag{4.2}$$

The coefficients β_1, β_2 and β_3 are non-negative numbers to be determined experimentally. Based on the formula (4.2), an interconnect temperature distribution model is introduced below.

Let the interconnect width be w, thickness be t, length be L, thermal conductivity be k, and power density at location y be $p(y)$. Consider one incremental interconnect segment of length Δy, e.g., the box $[y, y+\Delta y]$ shown in Fig.4.2(c). The total heat entering into the box from the left face is given by

$$q_l = -wtk\frac{dT(y)}{dy},$$

and that leaving the box from the right face is given by

$$q_r = q_l + \frac{d}{dy}\left[-wtk\frac{dT\left(y\right)}{dy}\right]\Delta y.$$

The net heat generated in the box is given by

$$p_{gen} = \left[p\left(y\right) - p_v^+\left(y\right)\right] wt\Delta y.$$

The law of energy conservation implies that

$$q_r - q_l = p_{gen}.$$

Therefore, the temperature distribution within the interconnect wire satisfies

$$(k+\beta_1)\frac{d^2T\left(y\right)}{dy^2} - \frac{T\left(y\right)}{R} = -f\left(y\right) \tag{4.3}$$

where

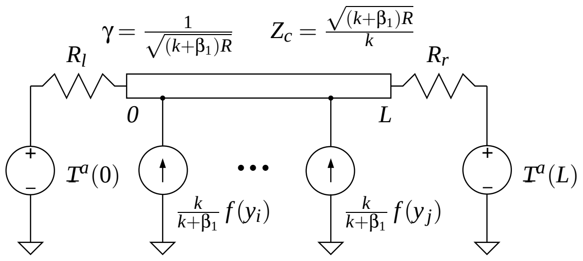
$$f\left(y\right) = p\left(y\right) - \frac{\bar{T}^a\left(y\right)}{\bar{R}} - \frac{\underline{T}^a\left(y\right)}{\underline{R}} - \beta_2\frac{\partial^2\bar{T}^a\left(y\right)}{\partial y^2} - \beta_3\frac{\partial^2\underline{T}^a\left(y\right)}{\partial y^2}. \tag{4.4}$$

The boundary conditions for (4.3) are specified by

$$\begin{aligned} k\left.\frac{dT\left(y\right)}{dy}\right|_{y=0} &= \frac{T\left(0\right)-\underline{T}^a\left(0\right)}{R_l} \\ -k\left.\frac{dT\left(y\right)}{dy}\right|_{y=L} &= \frac{T\left(L\right)-\underline{T}^a\left(L\right)}{R_r} \end{aligned} \tag{4.5}$$

where R_l (R_r) is the thermal impedance from the left (right) end of the line to the bottom ambient environment. Here assume that heat dissipates from the two ends of the line to the top ambient environment in negligible amounts, compared to that which dissipates through the low thermal-impedance vias to the bottom ambient environment.

In contrast to the traditional Schafft-type models, the introduced 1-D interconnect temperature distribution model considers the ambient temperatures, the thermal impedances of the vias, and the effect of temperature gradients. The following



$\gamma = \frac{1}{\sqrt{(k+\beta_1)R}}$
$Z_c = \frac{\sqrt{(k+\beta_1)R}}{k}$
R_l
R_r
0
L
$T^a(0)$
$T^a(L)$
$\frac{k}{k+\beta_1} f(y_i)$
$\frac{k}{k+\beta_1} f(y_j)$

Figure 4.4: Equivalent TL circuit for solving interconnect temperature distribution from (4.3).

introduces an $\mathcal{O}(n)$ method to solve the interconnect temperature distribution from (4.3).

4.3 Computation of Interconnect Temperature Distribution

To solve (4.3), this section again employs the transmission line theory to construct an equivalent TL circuit, which is shown in Fig.4.4. In the shown circuit, the TL propagation constant $\gamma = \frac{1}{\sqrt{(k+\beta_1)R}}$, the TL characteristic impedance $Z_c = \frac{\sqrt{(k+\beta_1)R}}{k}$, and $\frac{k}{k+\beta_1} f(y)$ is a distributive current source along the TL. The two ends of the TL are driven by the two voltage sources $\underline{T}^a(0)$ and $\underline{T}^a(L)$ through the two resistors R_l and R_r, respectively.

Based on Fig.4.4, the temperature at the location y, $T(y)$, can be derived:

$$T(y) = T_l^a(y) + \frac{k}{k+\beta_1} \int_0^L f(y') Z(y|y') \, dy' + T_r^a(y) \tag{4.6}$$

where $Z(y|y')$ denotes the transfer impedance from the location y' to the location y at the TL [69], and

$$\begin{aligned} T_l^a(y) &= \frac{H_l(L-y)\,\underline{T}^a(0)}{1+R_l/Z_l} \\ T_r^a(y) &= \frac{H_r(y)\,\bar{T}^a(L)}{1+R_r/Z_r} \\ Z(y|y') &= Y_\gamma\left(R_l \cosh\gamma\underline{y} + Z_c \sinh\gamma\underline{y}\right) \cdot \\ &\qquad \left[R_r \cosh\gamma(L-\bar{y}) + Z_c \sinh\gamma(L-\bar{y})\right]. \end{aligned}$$

Here

$$\begin{aligned}
H_{\{l,r\}}(y) &= \frac{R_{\{r,l\}}\cosh\gamma y + Z_c\sinh\gamma y}{R_{\{r,l\}}\cosh\gamma L + Z_c\sinh\gamma L}\\
Z_{\{l,r\}} &= Z_c\frac{R_{\{r,l\}} + Z_c\tanh\gamma L}{Z_c + R_{\{r,l\}}\tanh\gamma L}\\
Y_\gamma &= Z_c/\left[Z_c(R_l+R_r)\cosh\gamma L + (Z_c^2 + R_lR_r)\sinh\gamma L\right]\\
\underline{y} &= \min(y,y')\\
\bar{y} &= \max(y,y').
\end{aligned}$$

In deriving (4.6), the superposition principle has been used.

In general, $f(y)$ is given at discrete locations: $f(y_1),\ldots,f(y_{n+1})$, where $0 = y_1 < \cdots < y_{n+1} = L$. Further, $f(y)$ can be approximated by a piecewise-linear function or a smooth function consisting of n pieces. By using generic numerical integration methods to evaluate (4.6), $T(y)$ at each location y_i, $0 \le i \le n+1$, can be computed in $\mathcal{O}(n)$ time; however, to calculate $T(y)$ at all the locations $y_1,\ldots,y_{n+1}$ requires $\mathcal{O}(n^2)$ computations. To improve the efficiency, an $\mathcal{O}(n)$ algorithm is introduced to compute $T(y)$ at all the discrete locations.

From (4.6), $T(y_i)$ is rewritten into the form of

$$T(y_i) = T_l^a(y_i) + \alpha_l(y_i)(L-y_i)S_i^l + \alpha_r(y_i)S_i^r + T_r^a(y_i) \tag{4.7}$$

where

$$\begin{aligned}
\alpha_{\{l,r\}}(y) &= \frac{kY_\gamma}{k+\beta_1}\left(R_{\{r,l\}}\cosh\gamma y + Z_c\sinh\gamma y\right)\\
S_i^l &= \sum_{j=2}^{i}\int_{y_{j-1}}^{y_j} f(y')\,g_l(y')\,dy'\\
S_i^r &= \sum_{j=i+1}^{n+1}\int_{y_{j-1}}^{y_j} f(y')\,g_r(L-y')\,dy'\\
g_{\{l,r\}}(y') &= R_{\{l,r\}}\cosh\gamma y' + Z_c\sinh\gamma y'.
\end{aligned}$$

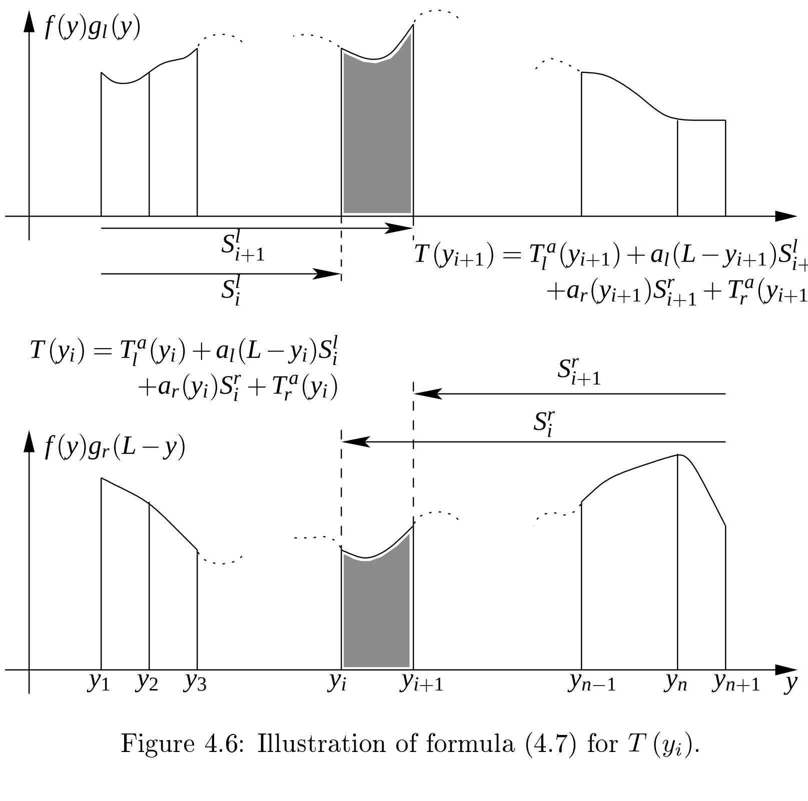
```
Begin Compute-wire-temp

  1. let $S_1^l = 0$ and compute $S_1^r$;

  2. For $i = 1$ to $n + 1$
         Compute $T_l^a(y_i)$, $T_r^a(y_i)$, $\alpha_l(L - y_i)$ and $\alpha_r(y_i)$;
         $T(y_i) = T_l^a(y_i) + T_r^a(y_i) + \alpha_l(L - y_i) S_i^l + \alpha_r(y_i) S_i^r$;
         $S_{i+1}^l = S_i^l + \int_{y_i}^{y_{i+1}} f(y') g_l(y') dy'$;
         $S_{i+1}^r = S_i^r - \int_{y_i}^{y_{i+1}} f(y') g_l(L - y') dy'$;
     End For

End Compute-wire-temp
```

Figure 4.5: Algorithm Compute-wire-temp for evaluating interconnect temperature $T(y)$ at locations $y_1, \ldots, y_{n+1}$.



$f(y)g_l(y)$
S_{i+1}^l
S_i^l
$T(y_{i+1}) = T_l^a(y_{i+1}) + a_l(L - y_{i+1})S_{i+1}^l + a_r(y_{i+1})S_{i+1}^r + T_r^a(y_{i+1})$
$T(y_i) = T_l^a(y_i) + a_l(L - y_i)S_i^l + a_r(y_i)S_i^r + T_r^a(y_i)$
S_{i+1}^r
S_i^r
$f(y)g_r(L-y)$
y_1
y_2
y_3
y_i
y_{i+1}
y_{n-1}
y_n
y_{n+1}
y

Figure 4.6: Illustration of formula (4.7) for $T(y_i)$.

Fig.4.6 illustrates the terms $T(y_i)$, $T(y_{i+1})$, S_i^l, S_i^r, S_{i+1}^l, and S_{i+1}^r. According to the figure, $T(y)$ can be computed sequentially from the locations y_1 to y_{n+1} by recursively calculating S_i^l and S_i^r. Therefore, the temperatures at all the discrete locations can be computed by an $\mathcal{O}(n)$ algorithm named Compute-wire-temp, which is shown in Fig.4.5. The values of $\int_{y_{j-1}}^{y_j} f(y')\, g_l(y')\, dy'$ and $\int_{y_{j-1}}^{y_j} f(y')\, g_r(L-y')\, dy'$ are usually given by analytical formulas, especially when $f(y)$ is a piecewise-linear function.

4.4 Experimental Results

4.4.1 Accuracy of Interconnect Temperature Distribution Model

The experiments used an interconnect array. The interconnect temperature distribution was obtained from both FLUENT 3-D simulation and the 1-D interconnect temperature distribution model combined with FLUENT 2-D characterization. Fig.4.7(a) shows the interconnect array, where each line is of length $L = 100$ μm, width $w = 1\,\mu$m, thermal conductivity $k = 144$ W/(m K), resistivity $\rho = 5.05 \times 10^{-6}$ Ω·cm, current density $J = 2$ MA/cm^2, and power density $p = 2.02 \times 10^{13}$ W/m^3. Two types of dielectric materials were used: SiO_2, with a thermal conductivity of 1.2 W/(m K), and polymer, with a thermal conductivity of 0.3 W/(m K). The parameters were chosen to be consistent with those in [19, 23]. Different line separations and inter-level dielectric (ILD) thickness were experimented.

FLUENT simulation was used to characterize the thermal properties of the cross-sectional area of the interconnect array. In the array, the thermal conductance $(1/R)$ from the boundary of the cross-sectional area of a metal line to the substrate was obtained by measuring the total heat flux out of the substrate when a 1 °C temperature was applied to the boundary, as shown in Fig.4.7(b). With the thermal conductance, algorithm Compute-wire-temp was used to compute the temperature

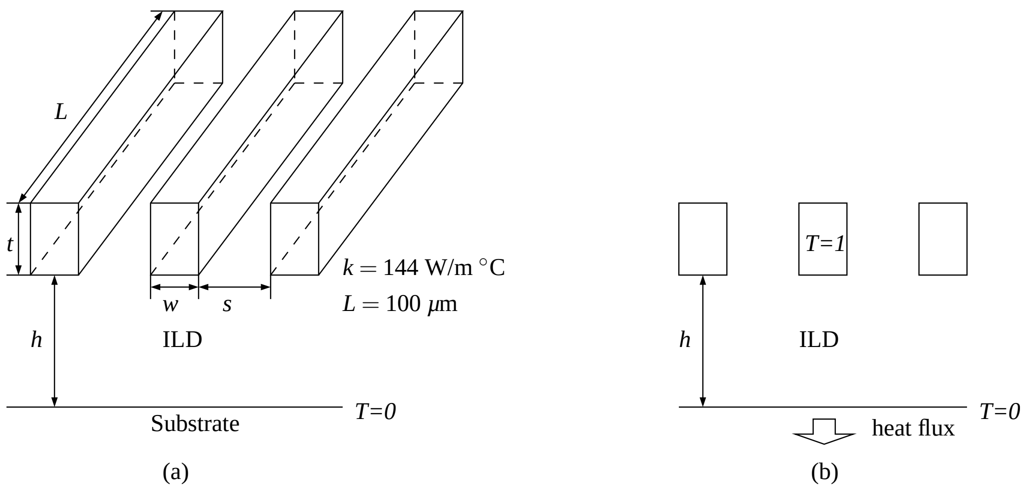


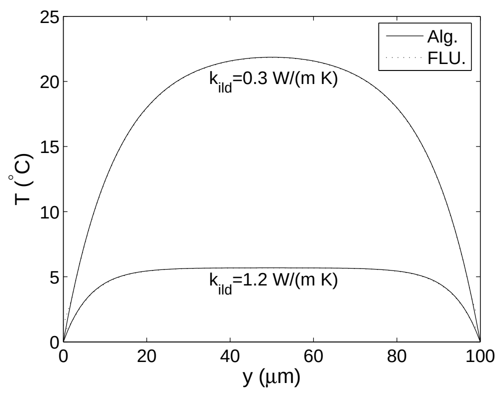
Figure 4.7: Interconnect array.

distribution of the central metal line.

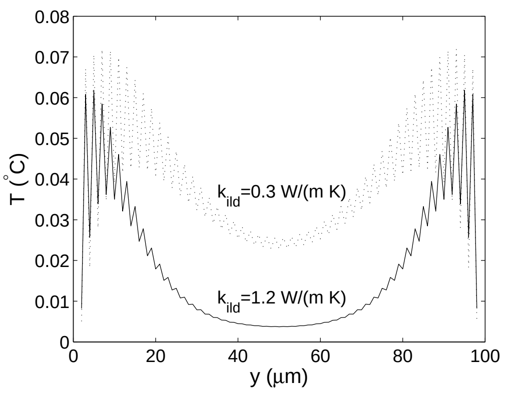
Fig.4.8 compares algorithm Compute-wire-temp and FLUENT, with parameters given by $s = 0.3\ \mu$m and $h = t = 0.8\ \mu$m (corresponding to the 0.1 μm technology node). The thermal conductance obtained from 2-D characterization was 3.55×10^{12} W/(K m^3) when the ILD was SiO_2 and 8.875×10^{11} W/(K m^3) when the ILD was polymer. The maximum temperature of the line increased from 5.685 °C to 21.863 °C when the ILD was changed from SiO_2 to polymer. To observe the effect of the ILD thickness, h was increased from 0.8 μm to 1.6 μm. Fig. 4.9 shows the results. The thermal conductance obtained from 2-D characterization was 1.824×10^{12} W/(K m^3) when the ILD was SiO_2 and 4.562×10^{11} W/(K m^3) when the ILD was polymer. The maximum temperature increased from 10.99 °C to 38.989 °C when the ILD was changed from SiO_2 to polymer. Table 4.1 further compares the results when the line width $w = 1\ \mu$m and separation $s = 0.5\ \mu$m.

4.4.2 Effect of Temperature Gradients

Next, different β_1 factors were tested to observe the effect of temperature gradients. Fig.4.10 shows the results for the case that $s = 0.5\,\mu$m, $w = 1\,\mu$m, $h = 1.6\,\mu$m

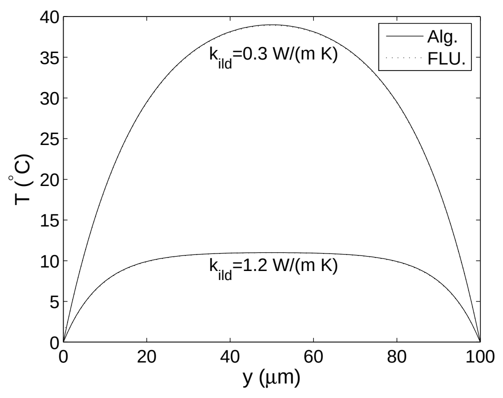


(a) Compute-wire-temp versus FLUENT.

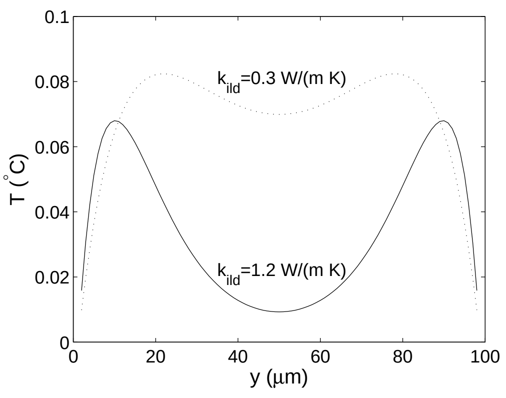


(b) Errors for y from 2 to 98 μm.

Figure 4.8: Comparison between 1-D interconnect temperature distribution model and FLUENT 3-D simulation: $s = 0.3$ μm and $h = t = 0.8$ μm.



(a) Compute-wire-temp versus FLUENT.

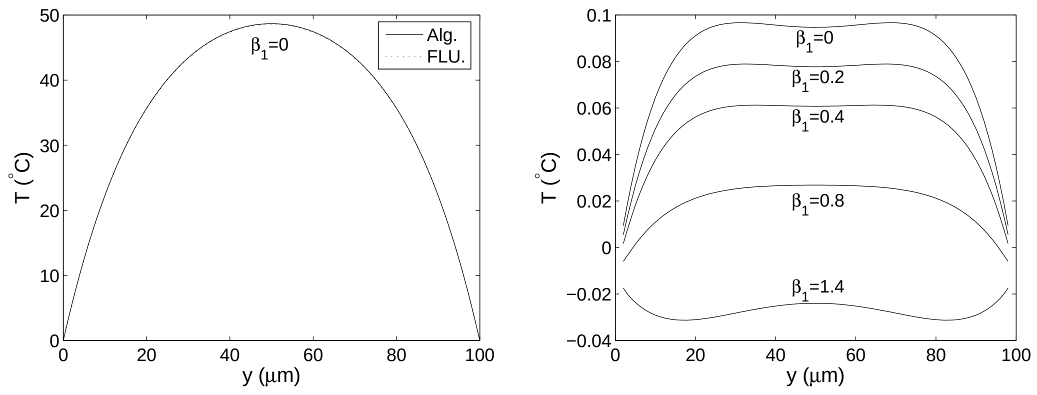


(b) Errors for y from 2 to 98 μm.

Figure 4.9: Comparison between 1-D interconnect temperature distribution model and FLUENT 3-D simulation: $s = 0.3$ μm, $h = 1.6$ μm and $t = 0.8$ μm.

h (μm)	0.8		1.6	
k_{ild} (W/m K)	0.3	1.2	0.3	1.2
1/R ($\times 10^{11}$ W/K m^3)	6.709	26.840	3.433	13.730
Max. Temp. (°C)	28.126	7.510	48.672	14.486
Max. Error (°C)	0.034	0.030	0.097	0.070

Table 4.1: Comparison between 1-D interconnect temperature distribution model and FLUENT 3-D simulation: $s = 0.5\,\mu$m and $w = 1\,\mu$m.



(a) Compute-wire-temp versus FLUENT. (b) Errors for y from 2 to 98 μm for different β_1's.

Figure 4.10: Effect of temperature gradients: accuracy of 1-D interconnect temperature distribution model versus β_1. Parameters: $s = 0.5\,\mu$m, $w = 1.0\,\mu$m, $h = 1.6\,\mu$m and $t = 0.8\,\mu$m.

β_1	0	0.2	0.4	0.8	1.4
Max. Error (°C)	0.097	0.079	0.061	0.027	0.031

Table 4.2: Effect of temperature gradients: accuracy of 1-D interconnect temperature distribution model versus β_1.

and $k_{ild} = 0.3$ W/(m K), i.e., the case in the fourth column of Table 4.1. The maximum absolute temperature errors are listed in Table 4.2.

Note that slightly increasing the value of β_1 reduced by as large as 70% the maximum absolute temperature error, which was however much smaller than the actual temperature. The experimental results have demonstrated that the accuracy of the introduced interconnect temperature distribution model can be comparable to that of FLUENT, and that temperature gradients within an interconnect wire can be overestimated if their own effect is neglected.

In summary, 3-D simulation by the FD and FE methods should be the most accurate in analyzing the interconnect temperature distribution. However, to improve computational time, 1-D interconnect temperature distribution models have been proposed in the literature with the combination of 2-D thermal characterization. Such models are reasonably accurate and lead to figures of merit for planning on-chip interconnect wires, for example, designing global interconnects and routing clock trees. In this chapter, an accurate Schafft-type interconnect temperature distribution model is presented, which considers the ambient temperatures and the effect of temperature gradients. Finally, an $\mathcal{O}(n)$ algorithm is introduced to solve the interconnect temperature distribution from the presented model.

CHAPTER V

Thermal Optimization in Cell Placement

5.1 Overview

In the top-down IC physical design flow, the cell placement stage is focused on reducing the total length of the interconnect wires and the overall area of the chip, as well as meeting the circuit timing requirements. The cell placement stage may lead to high temperatures, large temperature gradients, and numerous hot spots inside the chip if thermal optimization is not considered. High chip temperature aggravates interconnect electromigration and thus compromises the reliability of the chip. Meanwhile, large temperature gradients within the chip can cause logic faults because of the induced spatial variation of the interconnect and gate timing across the chip. In the literature, many cell placement algorithms have been proposed. However, a large portion of them have neglected the thermal issue or inadequately addressed it in lieu of the current chip thermal management criteria. To alleviate the chip thermal issue, this chapter introduces an optimal power budget model and discusses the integration of the model into the widely distributed Capo cell placement tool. First, this chapter reviews several representative cell placement algorithms that have the capability of thermal optimization: the matrix-synthesis approach, the simulated-annealing-based approach, the force-directed approach, and the partition-driven approach.

5.1.1 Matrix-Synthesis Approach

A matrix-synthesis problem can be described as follows: Given mn real numbers $x_0, x_1, \ldots, x_{nm-1}$, formulate a matrix $M_{m\times n}$ with these numbers such that the maximal sub-matrix sum of M, denoted by $\mu_t(M)$, is minimized. Notations are given below:

- $S_t(M)$, the set of all $t \times t$ sub-matrices in M.
- $\sigma(A)$, the sum of all elements in an arbitrary matrix A.
- $\mu_t(M)$, defined by

$$\mu_t(M) = \max_{A \in S_t(M)} \sigma(A).$$

In [25], a cell placement problem is transformed into a matrix synthesis problem. Assume that the cell placement problem requires $m \times n$ cells to be placed into $m \times n$ slots in the chip. Let the cell powers be denoted by $x_0, x_1, \ldots, x_{nm-1}$. Apparently, a solution to the matrix synthesis problem corresponds to a thermally optimized placement of the cells. As the matrix-synthesis problem is NP-complete, in [25] three approximation algorithms were given with proved bounds. The idea behind the approximation algorithms is to assign high-power cells into remotely located regions of the chip. The first approximation algorithm, called $A1$ in [25], is reviewed below.

Let $m = n = tq$, and divide the chip into $q \times q$ blocks, with each block containing $t \times t$ slots. Without loss of generality, the mn real numbers $x_0, x_1, \ldots, x_{nm-1}$, which represent the cell powers, are in non-increasing order: $x_0 \geq x_1 \geq \cdots \geq x_{nm-1}$. The location of each slot is specified by a tuple (i, j) with $0 \leq i, j \leq n-1$. Then for any

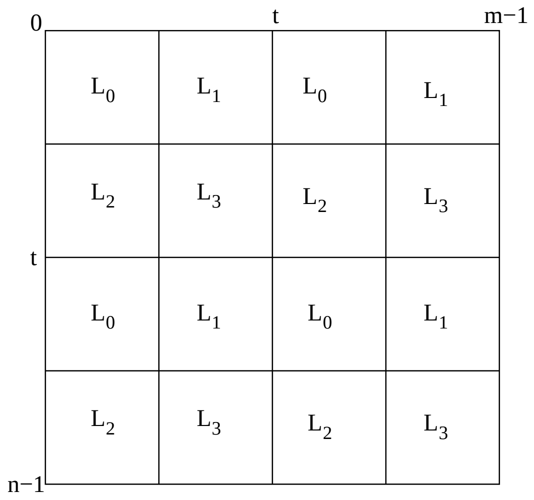


Figure 5.1: Labeling mechanism in matrix-synthesis approach: $m = n = 4$ and $t = 2$.

slot at the location (i, j), a label L_k is assigned such that

$$i \equiv \lfloor k/t \rfloor \pmod t$$
$$j \equiv (k \bmod t) \pmod t$$

where $0 \le k \le t^2 - 1$. Fig.5.1 illustrates the labeling mechanism for $t = 2$.

Algorithm $A1$ divides the mn real numbers into t^2 equisized groups in the natural order. For example, the first group named G_0 contains the first q^2 numbers $x_0, x_1, \ldots x_{q^2-1}$, and the last group named G_{t^2-1} contains the last q^2 numbers $x_{mn-q^2}, x_{mn-q^2+1}, \ldots x_{mn-1}$. Then the algorithm randomly assigns all the numbers in the same group, e.g., G_k, to the slots that have the same label L_k only. Consequently, the algorithm scatters the high-power cells across the chip rather than aggregate them at nearby locations to form hot spots. Algorithm A_1 was shown to have the maximal sub-matrix sum, denoted by $\mu_t(A1)$, bounded by

$$\mu_t(A1) \le 2\mu_t(OPT)$$

where $\mu_t(OPT)$ is the optimal solution. Therefore, the maximal sub-matrix sum of the placement produced by algorithm $A1$ is no more than two times that of the optimal placement.

Besides thermal, the matrix synthesis approach can meanwhile optimize both the total wire length and the overall chip area. Although aimed to distribute the cell powers evenly, the matrix-synthesis approach may not lead to an optimized temperature distribution because it does not consider the thermal boundary conditions for the chip.

5.1.2 Simulated-Annealing-Based Approach

Simulated-Annealing Algorithm

The simulated-annealing algorithm solves a combinatorial optimization problem by simulating the annealing process of finding ground states of matter under the control of a temperature schedule [42]. Mathematically, the simulated-annealing algorithm performs the Metropolis-Hastings method with the use of the Boltzmann distribution $p_B(\epsilon)$:

$$p_B(\epsilon) = \frac{g(\epsilon)\, e^{-\epsilon/KT}}{Z}$$

where $g(\epsilon)$ denotes the degeneracy of energy ϵ, or the number of states for particles with energy ϵ, and Z is the partition function or the normalization factor.

The Metropolis-Hastings method produces samples to meet a given probability distribution $p(\epsilon)$ based on the Markov chain. Given a sampled value of ϵ at the time step t, denoted by $\epsilon^{(t)}$, the method decides the next sampled value of ϵ, denoted by $\epsilon^{(t+1)}$, by first proposing a new value ϵ' and then examining the ratio $\alpha = \frac{p(\epsilon')}{p(\epsilon^t)}$. If $\alpha \geq 1$, the method sets $\epsilon^{(t+1)}$ to be x'; Otherwise, it sets $\epsilon^{(t+1)}$ to be x' with only probability α. In summary,

$$\epsilon^{(t+1)} = \begin{cases} \epsilon' & \alpha \geq 1 \text{ or } \alpha > \text{random}(0,1) \\ \epsilon^{(t)} & \text{otherwise} \end{cases}$$

where random$(0,1)$ returns a uniformly distributed random number between 0 and 1. Iterating the process, the method produces sampled values of ϵ that converge to the stationary distribution of the underlying Markov chain, $p(\epsilon)$. Therefore, when the Metropolis-Hastings method iterates with the Boltzmann distribution, the occurrence of small values of ϵ in the resulting stochastic process will dominate that of the larger values of ϵ at an exponential rate.

The simulated-annealing algorithm performs the Metropolis-Hastings method under a controlled temperature, the parameter T in $p_B(\epsilon)$. Take a minimization problem, for example. The simulated-annealing algorithm allows the parameter values to increase the cost function. However, the chance of allowing such parameter values becomes smaller with the decrease of the temperature. By the uphill-climbing of the cost function, the algorithm escapes the local minimum, and by a controlled temperature schedule, it avoids enumerating the entire parameter space in the process of minimizing the cost function. The algorithm can reach the global optimum with probability approaching 1 with the extending of the temperature schedule [53]. Fig.5.2 shows the pseudo-code, which comprises two loops. The inner loop runs the Metropolis-Hastings method to reach the stationary distribution at the set temperature. With the temperature gradually reduced, the outer loop expedites the process to find the global optimum.

Simulated-Annealing-Based Approach for Cell Placement

To apply the simulated-annealing algorithm in cell placement, the inner-loop condition, the outer-loop condition, the *next_temperature_scheduled* function and the *generate_next_configuration* function need to be specified. The inner-loop condition can force the algorithm to exit from the inner loop if many placements

```
$T = initial\_temperature\_scheduled\,()$;
$t = 0$;
$X = initial\_configuration\,()$;
while algorithm stopping criterion not satisfied
    while stopping criterion at $T$ not satisfied
        $X = generate\_next\_configuration\left(X^{(t)}\right)$;
        $\epsilon' = cost\_of\_configuration\,(X)$;
        if $\epsilon' < \epsilon^{(t)}$ or $\exp\left[-\left(\epsilon' - \epsilon^{(t)}\right)/kT\right] >$ random$(0,1)$
        then
            $X^{(t+1)} = X$;
            $\epsilon^{(t+1)} = \epsilon'$;
            if $\epsilon' <$ bestSeenValue then
                bestSeenValue $= \epsilon'$;
                bestConfiguration $= X$;
            end if;
        else
            $X^{(t+1)} = X^{(t)}$;
            $\epsilon^{(t+1)} = \epsilon^{(t)}$;
        end if;
        $t = t + 1$;
    end while;
    $T = next\_temperature\_scheduled\,(T)$;
end while.
```

Figure 5.2: Pseudo code of simulated-annealing algorithm.

attempted have been rejected or the maximal number of iterations has been reached. The outer-loop condition can terminate the algorithm if it has not improved the placement at several consecutive temperatures. The *next_temperature_scheduled* function is often in the form of

$$T_{new} = \alpha\left(T_{old}\right)T_{old}$$

where $0 < \alpha < 1$ [42, 59]. Accordingly, the new temperature scheduled, T_{new} reduces exponentially. The *generate_next_configuration* function generates a new placement based mainly on two mechanisms: one is to swap the locations of two cells, and the other is to move one cell from its current location to a new location. In the TimberWolf placement and routing package, an *exchange class mechanism* is used to exchange only cells in the same class, and the movement of a cell is confined to a rectangular window which shrinks with the decrease of the temperature [59].

Because exchanging cells may cause overlaps, the placement cost includes also the cost of cell overlaps in addition to the total half-perimeter wire length (HPWL) of all the nets.

To optimize the temperature distribution of the chip, a cost similar to the cost of cell overlaps can be added to the placement cost. The cell placement approach in [65] specifies a power budget for the chip. Then in the cell placement stage, if a generated placement violates the power budget, the placement will be either rejected or assigned a large placement cost. To specify the power budget, first the average temperature of the chip is obtained from an empirical model:

$$T_{averge} = T_{ambient} + P_{total} \cdot R^{th}$$

where $T_{average}$, $T_{ambient}$, P_{total}, and R^{th} are the average temperature, the ambient temperature, the total power dissipation, and the equivalent thermal resistance of the chip, respectively [65]. Then for each region of the chip, a temperature slack is added to $T_{average}$ to form an envelop temperature for that region. With the thermal transfer matrix, an envelop power for each region, called the power budget of the region, can be computed from the envelop temperatures.

5.1.3 Forced-Directed Approach

The force-directed approach originated from an analogy to Hook's law on elastic materials [52]. Let the cost of connection between two cells numbered i and j at locations $\vec{p_i} = (x_i, y_i)$ and $\vec{p_j} = (x_j, y_j)$ be approximated by the squared Euclid distance:

$$c_{ij}\left[(x_i - x_j)^2 + (y_i - y_j)^2\right]$$

where c_{ij} is a weighting factor. To minimize the total connection cost, the optimal location of cell i must satisfy

$$\begin{aligned} &\sum_{j=1}^{n} c_{ij}\left(x_i - x_j\right) = 0 \\ &\sum_{j=1}^{n} c_{ij}\left(y_i - y_j\right) = 0 \end{aligned} \tag{5.1}$$

where n is the number of cells directly connecting to cell i. The above optimal conditions resemble Hook's law. Treat the connection between two cells as a spring so that the longer the spring is stretched, the stronger the tension between the two cells. Hence, in the optimal placement, each cell should receive a zero total force; otherwise, a nonzero force will displace the cell to a different location.

With only the attractive forces, the cells will be squeezed together. To reduce the cell overlaps, a repulsive force is introduced between any two cells. Let $D\left(x,y\right)$ denote the density of the cells at the location (x,y):

$$D\left(x,y\right) = \sum_{i} a_i\left(x,y\right) - a_{avg}$$

where $a_i\left(x,y\right) = 1$ if cell i covers location (x,y); otherwise, $a_i\left(x,y\right) = 0$. a_{avg} is the average density of the cells in the chip: $a_{avg} = \frac{\sum_i module\_area_i}{chip\_area}$ [30, 45]. If $D\left(x,y\right) > \alpha_{avg}$, $D\left(x,y\right)$ is positive and it behaves like a positive charge density. On the other hand, if $D\left(x,y\right) < \alpha_{avg}$, $D\left(x,y\right)$ is negative and it behaves like a negative charge density. When a cell is placed at a location with a large density of cells, the cell resembles a positive charge at a location of large positive charge density, and it receives a large repulsive force. The repulsive force, denoted by $\vec{f_r}$, is governed by

$$\nabla \cdot \vec{f_r} = kD\left(x,y\right).$$

With the free-space Green's function [30, 40], $\vec{f_r}$ is represented in the form of

$$\vec{f_r} = \int_S D\left(x',y'\right) \frac{\vec{r} - \vec{r'}}{\left|\vec{r} - \vec{r'}\right|^2} dr'.$$

```
$generate\_initial\_placement$();
while placement is improving
    calculate $\vec{d}$ due to fixed cells;
    calculate $\vec{f_r}$ due to cell overlaps or other constraints;
    update cell locations based on (5.2);
end while;
post-processing to legalize the placement.
```

Figure 5.3: Pseudo code for the forced-directed approach.

In matrix form, the placement cost is given by

$$\frac{1}{2}\vec{p}^T C\vec{p} + \vec{d}^T\vec{p} + \vec{e}$$

where $\vec{p}$ is the vector of cell positions. Apparently, the first term is contributed by the connections between two movable cells, the second term is contributed by the connections between one movable cell and another fixed cell, and the last term is contributed by the connections between two fixed cells. Combining the repulsive force $\vec{f_r}$, the optimal conditions in (5.1) are modified to be

$$C\vec{p} = \alpha\vec{d} + \beta\vec{f_r} \tag{5.2}$$

where α and β are weighting factors to balance the attractive forces and the repulsive forces. Fig.5.3 gives the pseudo code for the forced-directed approach.

To optimize the temperature distribution, the forced-directed approach in [32] introduces thermal forces. At a given location, in addition to the attractive and repulsive forces, a cell also receives a thermal force, denoted by $\vec{f}_{therm}$, which is the negative temperature gradient at the location. Then the optimal location of each cell satisfies

$$C\vec{p} = \alpha\vec{d} + \beta\vec{f_r} + \gamma\vec{f}_{therm}$$

where α, β, and γ are coefficients to balance the attractive forces, the repulsive forces, and the thermal forces.

5.1.4 Partition-Driven Approach

Hyper-graph Partition

A hyper-graph is a special graph such that an edge in the graph may connect to more than two vertices. The number of vertices connected to an edge is called the degree of the edge. Partitioning a hyper-graph is to divide the vertices into different parts. Mathematically, a hyper-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined by the set of vertices, $\mathcal{V}$, and the set of edges, $\mathcal{E}$. Each $v \in \mathcal{V}$ has a size named s_v. Each edge $\epsilon \in \mathcal{E}$ is a set of vertices in $\mathcal{V}$ and has a cost named c_ϵ. The cost of edge ϵ can also be denoted by c_{xy}, where $x, y \in \epsilon$. A partition Π of the graph is defined by a collection of subsets $\Pi \equiv \{\Pi_1, \cdots \Pi_k\}$ such that $\Pi_i \cap \Pi_j = \emptyset$ for any $i \neq j$ and $\cup_{i=1}^{k} \Pi_i = \mathcal{V}$. Each subset corresponds to one part of the partition. The quality of partition Π is measured by the cost of the cut-set C_Π. An edge is said to be in the cut-set C_Π if its vertices are assigned to more than one part. The cut-set cost can be defined by the number of edges or the weighted sum of the costs of the edges in C_Π. The objective of hyper-graph partition is to assign all $v \in V$ to different parts such that the cost of C_Π is minimized. Hyper-graph partition may be subject to additional constraints such as balancing the number of vertices in each part.

One special hyper-graph partition problem is the bisection problem, in which a hyper-graph of $2n$ vertices is to be optimally divided into two parts of equal size such that the cut-set cost is minimized. Normally, a k-way hyper-graph partition problem, in which the vertices are to be assigned to k parts, is transformed into a series of bisection problems. The hyper-graph bisection problem is NP-complete. Two well-known heuristic approaches for the bisection problem are reviewed below.

Kernighan-Lin Algorithm

The Kernighan-Lin algorithm is an exchange-based bisection algorithm [44, 56]. Consider a bi-partition of two parts A and B. For a vertex $a \in A$, the external cost of a, denoted by E_a, is the total cost of edges that connect a to vertices in the other part B:

$$E_a = \sum_{x \in B} c_{ax}.$$

The internal cost of a, denoted by I_a, is the total cost of edges that connect a to vertices in the same part A:

$$I_a = \sum_{x \in A} c_{ax}.$$

Moving a from A to B reduces the cut-set cost by D_a:

$$D_a = E_a - I_a.$$

Similarly, moving a vertex $b \in B$ from B to A reduces the cut-set cost by D_b. Then the gain from interchanging a and b, denoted by g_{ab}, is given by

$$g_{ab} = D_a + D_b - 2c_{ab}.$$

That is, interchanging a and b reduces the cut-set cost by g_{ab}.

At each step, the KL algorithm interchanges one pair of vertices that have the maximum gain, locks them, and then updates the gains of the other pairs of vertices affected. The algorithm repeats the previous step until all the vertices are locked. Denote the two vertices interchanged in the i-th step by a_i and b_i, respectively, and the gain by g_i. Because g_i may be negative, the algorithm makes permanent only the first k interchanges such that $\sum_{i=1}^{k} g_i$ maximizes $\sum_{i=1}^{n} g_i$. This constitutes a single pass of the KL algorithm. After one pass, all the vertices are unlocked and the next pass begins. The algorithm stops if the present pass has not improved the cut-set cost.

Because in the i-th step, the KL algorithm needs to choose the pair with the maximum gain from the $(n-i)^2$ pairs, the time complexity of one pass is $\mathcal{O}(n^3)$. To reduce the run-time, sort D_{ax} for every vertex ax in A and D_{by} for every vertex by in B such that

$$D_{a1} \geq D_{a2} \geq \cdots D_{an}$$

and

$$D_{b1} \geq D_{b2} \geq \cdots D_{bn}.$$

Then examine D_a's and D_b's in the sorted order until a pair D_{ai} and D_{bj} is encountered such that $D_{ai} + D_{bj}$ is less than the present maximal gain. Then all the pairs after D_{ai} and D_{bj} can be discarded because their gains must not exceed the present maximal gain. Using the above sort procedure reduces the time complexity of the KL algorithm to $\mathcal{O}(pn^2 \lg n)$, where p is the total number of passes.

Fiduccia-Mattheyses Algorithm

The Fiduccia-Mattheyses algorithm is a move-based bisection algorithm [31, 56]. Consider a bi-partition of two parts A and B. Given a vertex i, let $F(i)$ be the *From* part, i.e., the part currently containing vertex i, and let $T(i)$ be the *To* part, i.e., the part whereto vertex i can be moved. Given an edge ϵ, let $F(\epsilon)$ be the number of vertices in the *From* part that ϵ connects to, and let $T(\epsilon)$ be the number of vertices in the *To* part that ϵ connects to. Then the gain from moving vertex i from $F(i)$ to $T(i)$, denoted by $g(i)$, is given by

$$g(i) = \sum_{\{\epsilon \in \mathcal{E} | i \in \epsilon\}} c_\epsilon \delta_{1,F(\epsilon)} - \sum_{\{\epsilon \in \mathcal{E} | i \in \epsilon\}} c_\epsilon \delta_{0,T(\epsilon)}$$

where $\delta_{i,j}$ denotes the Kronecker delta such that

$$\delta_{x,y} = \begin{cases} 1, & x = y \\ 0, & \text{otherwise.} \end{cases}$$

In other words, moving vertex i from its present part to the other part reduces the cut-set cost by $g(i)$.

To maintain the sizes of A and B, the FM algorithm imposes a balance criterion

$$r \times \sum_{v \in \mathcal{V}} s_v - \max\{s_v, v \in \mathcal{V}\} \le \sum_{v \in A} s_v \le r \times \sum_{v \in \mathcal{V}} s_v + \max\{s_v, v \in \mathcal{V}\} \tag{5.3}$$

where r is the balance ratio, e.g., 0.5 for a balanced bisection. Define the base vertex as a vertex that has the maximal gain and is free to move without violating the balance criterion. At each step, the FM algorithm moves the base vertex from its $From$ part to its To part, locks the vertex, and then updates the gains of the other vertices affected. The algorithm repeats the previous step until no base vertices exist. The procedure constitutes one pass of the FM algorithm. Denote the gain at the i-th steps by g_i. At the end of one pass, the FM algorithm makes permanent the moves at the first k steps such that $\sum_{i=1}^{k} g_i$ is the maximal. Then the algorithm unlocks all the vertices and begins the next pass if the latest pass has improved the cut-set cost.

After moving one vertex, say v, the FM algorithm applies the procedure in Fig.5.4 to update the gains of the other free vertices efficiently. Using a bucket data structure, the FM algorithm runs approximately in linear time with respect to the number of vertices.

5.1.5 Thermal Optimization in Partition-Driven Approach

The partition-driven approach recursively divides the chip into smaller bins until the bins become small enough to be handled by an end-case placer [54, 37]. Given

```
update_gains($v$)
begin
    $F = F(v), T = T(v)$;
    for each edge $\epsilon$ on vertex $v$
    begin
        if $F(\epsilon) = 1$ then
            decrease the gains of the other free
                vertices on $\epsilon$ by $c_\epsilon$;
        else if $F(\epsilon) = 2$ then
            increase the gain of the other free
                vertex in $F$ by $c_\epsilon$;
        end if;
        if $T(\epsilon) = 1$
            decrease the gain of the only free
                vertex in $T$ by $c_\epsilon$;
        else if $T(\epsilon) = 0$
            increase the gains of the other free
                vertices on $\epsilon$ by $c_\epsilon$;
        end if;
        $F(\epsilon) = F(\epsilon) - 1, T(\epsilon) = T(\epsilon) + 1$;
    end for;
end
```

Figure 5.4: Procedure to update gains of free vertices in Fiduccia-Mattheyses algorithm after moving base vertex v.

a bin and the set of cells contained, the partition-driven approach first determines the location of the cut-line to divide the given bin into two halves, called the child bins. Then the approach uses the FM algorithm to assign the cells to the two child bins, with the cost of the connections across the two child bins minimized. Then the approach repeats the previous procedure on each of the two child bins until the produced bins become small enough so that the exact locations of the contained cells can be determined trivially. The approach is shown in Fig.5.5.

The cut-line to divide a given bin runs either horizontally or vertically, depending on the placement style and the shape of the given bin. To formulate the bisection problem, in the hyper-graph, one vertex represents a free cell, the vertex size represents the area of the cell, and an edge represents a net. The edge cost is given by the HPWL calculated when the vertices on the edge are assumed to be at the centers

```
En-queue the top-level bin that contains the whole chip;
while the queue is not empty
    de-queue a bin;
    if the bin is small enough then
        runs the end-case placer;
    else
        determine the cut-line location;
        form a bisection problem;
        run the $FM$ algorithm to partition the bin
           into two child bins;
        en-queue the two child bins;
    end if;
end while.
```

Figure 5.5: Partition-based approach

of the two child bins. The balance criterion in the FM algorithm avoids overlaps between cells and also maintains the white-space ratio. Using the linear-time complexity FM algorithm, the partition-driven approach can place multiple million cells efficiently, compared to the simulated-annealing-based approach.

The partition-driven approach in [20] considers thermal optimization. First, the approach runs thermal simulation for a few random placements and determines the temperature budget, denoted by T_{budget}, for each region of the chip based on the maximal and minimal temperatures in the random placements. Then the approach modifies the FM algorithm such that a base vertex must not only satisfy the area balance criterion (5.3) but also ensure the resultant chip temperature distribution to be within T_{budget}.

5.2 Optimal Power Budget Model for Cell Placement

This section describes an optimal power budget model that determines the best allocation of cell powers to different regions of the chip so that the resultant temperature distribution most closely approximates the target temperature distribution. Based on the optimal power budget model, this section introduces a top-level thermal optimizer and a front-level thermal optimizer that use LOTAGre to solve the optimal power budget for use in the partition-driven approach. Particularly, this section presents the integration of the optimal power budget model with the Capo placement tool, a sophisticated partition-driven placement tool, to perform chip-level thermal optimization.

5.2.1 Optimal Power Budget Model

First, introduce the optimal power budget model. In the partition-driven approach, the recursive bin-splitting procedure forms a binary tree of bins called the partition tree. In the tree, each bin is a geometrical union of its two child bins. Particularly, the bin at the root of the tree represents the entire chip layout. Define the level of one bin as the distance of the bin from the root of the tree. Assume not considering the variations of cell powers in the routing stage. Then, under a fixed total chip power, there exists an optimal allocation of powers (or optimal power budget) for the bins of the same level so that the resultant temperature distribution of the chip is optimal. Hence, if the partition algorithm closely complies with the optimal power budget when splitting bins of the same level, the generated placement should not contain a significant number of hot-spots.

The optimal power budget model plans the total power of each bin of the same level, with the given total powers of all the parent bins of one level above, to improve the temperature distribution of the chip the most effectively. To establish the optimal power budget model, the chip layout is meshed by the discrete heat-source model so that each bin contains a set of mesh grids in a rectangular region. Next, the grids in the same bin are organized into several clusters, with each cluster comprising one or more grids. Several notations are given below: Denote the bins of the same level by $B_1, B_2, \ldots B_n$; for each bin B_i, denote the clusters contained by $C_{i1}, C_{i2}, \ldots C_{ik}$;

for each cluster C_{ij}, denote its area by a_{ij} and its total power by x_{ij}. Finally, denote the total power of bin B_i by p_i, where

$$p_i = \sum_{j=1}^{k} x_{ij}. \tag{5.4}$$

Then reformulate (5.4) into an implicit condition:

$$x_{i1} = p_i - \sum_{j=2}^{k} x_{i2}. \tag{5.5}$$

Let all the x_{ij}s such that $i > 1$ form a vector x, named the power vector. Then the chip temperature distribution can be given by

$$RMx + c \tag{5.6}$$

where R is the thermal transfer matrix for the chip, and M is a mapping matrix such that if cluster C_{ij} has a total power x_{ij}, the matrix-vector product Mx contributes a negative power $-x_{ij}$ to cluster C_{i1}. The mapping matrix is shown in (5.7), where the first row and the first column of the matrix are labels for the clarity of presentation.

$$(5.7)\quad M = \left[\begin{array}{cccccccc}
 & x_{12} & \cdots & x_{1u} & \cdots & x_{n2} & \cdots & x_{nv} \\
C_{12} & 1/a_{12} & & & & & & \\
\vdots & \vdots & & & & & & \vdots \\
C_{12} & 1/a_{12} & & & & & & \\
 & & \ddots & & & & & \\
C_{1u} & & & 1/a_{1u} & & & & \\
\vdots & & & \vdots & & & & \vdots \\
C_{1u} & & & 1/a_{1u} & & & & \\
 & & & & \ddots & & & \\
C_{n2} & & & & & 1/a_{n2} & & \\
\vdots & & & & & \vdots & & \\
C_{n2} & & & & & 1/a_{n2} & & \\
\vdots & & & & & & \ddots & \vdots \\
C_{nv} & & & & & & & 1/a_{nv} \\
\vdots & & & & & & & \vdots \\
C_{nv} & & & & & & & 1/a_{nv} \\
C_{11} & -1/a_{11} & \cdots & -1/a_{11} & & & & \\
\vdots & \vdots & \ddots & \vdots & & & & \\
C_{11} & -1/a_{11} & \cdots & -1/a_{11} & & & & \\
\vdots & & & & \ddots & & & \\
C_{n1} & & & & & -1/a_{n1} & \cdots & -1/a_{n1} \\
\vdots & & & & & \vdots & \ddots & \vdots \\
C_{n1} & & & & & -1/a_{n1} & \cdots & -1/a_{b1} \\
0 & \cdots & & & & & & \\
 & & & & \ddots & & \cdots & 0
\end{array}\right] = \left[\begin{array}{c} E \\ C \\ 0 \end{array}\right]$$

In the mapping matrix (5.7), the rows with the same label (e.g., C_{1k}) account for the grids in the same cluster. Sub-matrix E contains the rows numbered from C_{12} to C_{nv}. Sub-matrix C contains the rows numbered from C_{11} to C_{n1}. The zero matrix indicates that the power vector x has no impact on the related grids. Therefore, Mx produces a power map such that each cluster C_{ij} has a total power x_{ij} and each grid in the cluster has a power density x_{ij}/a_{ij}. For the first cluster C_{i1} of each bin B_i, the matrix-vector product Mx contributes a power density of $-\frac{1}{a_{i1}}\sum_{j=2}^{k} x_{ik}$ to each grid contained, and the power constraint (5.5) induces an additional fixed power density p_i/a_{i1} to the grid. Accordingly, the constant vector c denotes the temperature distribution incurred by the fixed power densities at the first clusters of all the bins and the fixed cells in the chip.

Let scalar τ denote the target average temperature for the chip and τe denote the target temperature distribution for the chip, with e being the normalized temperature distribution. Then the optimal power budget model is given in the least-square form:

$$\min_{x} \|RMx + c - \tau e\|^2 . \tag{5.8}$$

The objective of the least-square form is to find the optimal power vector x such that the resulting temperature distribution most closely approximates the target temperature distribution τe. The least-square form (5.8) requires that the optimal power vector, denoted by x^*, satisfy

$$(RM)^T RMx^* = (RM)^T (\tau e - c) . \tag{5.9}$$

If τ is also one parameter to be optimized, the optimal τ, denoted by τ^*, which minimizes (5.8), must satisfy

$$\tau^* = \frac{e^T}{\|e\|^2} (RMx^* + c) . \tag{5.10}$$

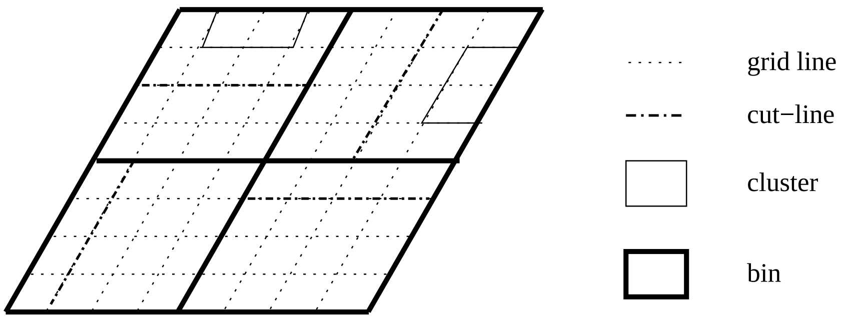


Figure 5.6: Calculating optimal power budget by summation of optimal powers of grids.

The optimal power budget for the bins of the same level can be determined from the optimal power vector x^* solved for the parent bins of one level above. As shown in Fig.5.6, the optimal power vector x^* determines the power of each grid in the layout. Therefore, the summation of the powers of all the grids contained by a bin gives the optimal power budget for the bin.

Because of its high efficiency, LOTAGre is used to solve the optimal power vector x^* from (5.9) and (5.10). In LOTAGre, the thermal transfer matrix R is given by

$$R = D^{-1}GD \tag{5.11}$$

where D is the DCT matrix, D^{-1} is the IDCT matrix, and G is a diagonal matrix of the Green's function values. The DCT matrix has the property that $D^{-1} = D^T$, which can be verified by the MATLAB formulas [1]: the DCT coefficient, denoted as B_{pq}, is given by

$$B_{pq} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \alpha_p \alpha_q A_{mn} \cos \frac{\pi (2m+1) p}{2M} \cos \frac{\pi (2n+1) q}{2N}$$

and the IDCT coefficient, denoted as A_{mn}, is given by

$$A_{mn} = \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} \alpha_p \alpha_q B_{pq} \cos \frac{\pi (2m+1) p}{2M} \cos \frac{\pi (2n+1) q}{2N}.$$

Then apply the thermal transfer matrix (5.11) to reformulate (5.9) and (5.10) to

$$\begin{aligned} M^T D^{-1} G^2 D M x^* &= (RM)^T (\tau^* e - c) \\ \tau^* &= \frac{e^T}{\|e\|^2} (RMx^* + c) . \end{aligned} \tag{5.12}$$

Generally, a constant temperature is the desired target temperature distribution for the chip because it does not produce hot spots. Therefore, e is assumed to be a vector of ones, and then the right-hand sides of (5.12) can be simplified to

$$\begin{aligned} &(RM)^T c = M^T D^{-1} G D D^{-1} G D p = M^T D^{-1} G^2 D p \\ &(RM)^T e = M^T D^{-1} G D e = 0 \end{aligned} \tag{5.13}$$

where p is the vector of fixed powers.

5.2.2 Top-Level Thermal Optimizer

The optimal power vector x^* can be solved from (5.12) by an iterative linear solver. At the top few levels of the partition tree, the numbers of the bins of the same level are relatively small, and for these levels, a *top-level* thermal optimizer solves x^* from (5.12). The top-level thermal optimizer uses a clustering mechanism that requires each cluster in a bin to contain only a single grid. The clustering mechanism leads to the highest resolution because the power of each grid in the same bin can differ from those of the other grids. In contrast, if a cluster contains more than one grid, all the grids in the cluster must have the same power, i.e., the average power of the cluster. Moreover, the top-level thermal optimizer requires the bins of the same level to form a partition of the entire chip layout. This requirement is often satisfied for about the top 10 levels of the partition tree. Using the high-resolution clustering mechanism, the top-level thermal optimizer reduces the mapping matrix M to the following form:

$$(5.14)\qquad M = \left[\begin{array}{c|ccccccc} & x_{12} & \cdots & x_{1u} & \cdots & x_{n2} & \cdots & x_{nv} \\ C_{12} & 1 & & & & & & \\ \vdots & & \ddots & & & & & \\ C_{1u} & & & 1 & & & & \\ \vdots & & & & \ddots & & & \vdots \\ C_{n2} & & & & & 1 & & \\ \vdots & & & & & & \ddots & \\ C_{nv} & & & & & & & 1 \\ C_{11} & -1 & \cdots & -1 & & & & \\ \vdots & & & & \cdots & & & \\ C_{n1} & & & & & -1 & \cdots & -1 \end{array}\right] = \begin{bmatrix} I \\ C \end{bmatrix}$$

where I is the identity matrix. Directly solving x^* from (5.12) with an iterative linear solver is computationally expensive because the optimal power vector x^* may contain close to one million or even millions of unknowns. Therefore, the top-level thermal optimizer solves x^* from (5.12) in an alternative way.

Define a vector y such that

$$(5.15)\qquad y = \begin{bmatrix} y_u \\ y_l \end{bmatrix} = D^{-1}G^2DMx^*$$

where y_l is an $n \times 1$ vector, called the guess vector. From (5.12), (5.13) and (5.14),

$$M^Ty = y_u + C^Ty_l = -\left(RM\right)^T c.$$

Hence, the vector y_u can be represented in terms of the guess vector y_l:

$$(5.16)\qquad y_u = -\left(RM\right)^T c - C^Ty_l.$$

From (5.15) and (5.14),

$$Mx^* = D^{-1}G^{-2}Dy$$

and

$$\begin{bmatrix} I \\ C \end{bmatrix} x^* = D^{-1}G^{-2}Dy = \begin{bmatrix} z_u \\ z_l \end{bmatrix} \tag{5.17}$$

where z_l is a newly defined $n \times 1$ vector.

From (5.17), an $n \times 1$ residue vector r is defined such that

$$r = Cz_u - z_l \tag{5.18}$$

i.e., r is the residue for (5.17) incurred by the guess vector y_l. The goal is to find the right guess vector y_l such that the residue r computed based on (5.15), (5.17) and (5.18) is zero. Clearly, at this stage an iterative linear solver can be employed to find the right guess vector y_l and then derive the optimal power vector x^*.

The residue vector r is linearly related to the guess vector y_l:

$$r = Ay_l - b$$

where A is a matrix and b is a constant vector. Apparently, the vector b can be obtained by negating the residue vector r for a zero guess vector y_l: $b = Ay_l|_{y_l=0} - r = -r$. The matrix-vector product Ay_l for a given guess vector y_l can be obtained from

$$Ay_l = r + b.$$

Note that the residue r can always be computed from (5.16), (5.17) and (5.18). With the known vector b and the procedure to obtain the matrix-vector product Ay_l, the generalized minimal residue (GMRES) method is utilized to solve the implicit linear equation

$$Ay_l = b. \tag{5.19}$$

```
1. Assign a zero vector to $y_l$ and compute the residue $r$.
   Then let the right hand side vector $b$ of linear
   equations (5.19) be
                         $b = -r.$
2. Apply the GMRES method to solve (5.19), where the
   matrix-vector product $Ay_l$ for a given guess vector
   $y_l$ is computed below:
    2a. Compute
                         $y_u = -M^T D^{-1} G^2 Dp - C^T y_l$
    2b. Apply LOTAGre to compute the vector $z$:
                         $z = \begin{bmatrix} z_u \\ z_l \end{bmatrix} = D^{-1} G^{-2} Dy$
    2c. Compute the residue vector $r$ by
                         $r = Cz_u - z_l$
    2d. Compute the matrix-vector product $Ay_l$ by
                         $Ay_l = r + b$
3. The optimal power vector $x^*$ is substituted by the $z_u$
   vector at the last iteration of the GMRES method.
```

Figure 5.7: Top-level thermal optimizer for computing optimal power budget.

Once the GMRES method solves the guess vector y_l from (5.19), the optimal power vector x^* can be substituted by the vector z_u . The above steps to solve (5.12) constitute the top-level thermal optimizer, which is shown in Fig.5.7. Note that the number of unknowns to be solved by the GMRES method equals the number of bins, i.e., n, which is in the order of hundreds to thousands.

5.2.3 Front-Level Thermal Optimizer

As the recursive bin-splitting procedure continues, the height of the partition tree increases and the partition tree may become incomplete — some bins at the same level of the partition tree have no child bins. The underlying reason is that some bins are sufficiently small so that an end-case placer can directly handle the bins without further splitting. Although the top-level thermal optimizer uses a clustering mechanism that provides the highest resolution, the optimizer requires the bins of the same level to form a partition of the entire chip layout. When the partition tree starts to become incomplete, the bins at the lowest level (also called the front level) no longer form a partition of the layout. Therefore, this case is particularly handled by a *front-level* thermal optimizer that directly solves the optimal power vector x^* from (5.12). Like the top-level thermal optimizer, the front-level thermal optimizer starts with a guess vector, which is actually x^*, and iteratively improves the accuracy of x^*, however, by the conjugate gradient (CG) method.

Unlike the top-level thermal optimizer, which solves a linear system of size equal to the number of bins, the front-level thermal optimizer solves a linear system of size equal to the number of clusters. In order to improve the efficiency of the CG method, the front-level thermal optimizer reduces the number of clusters for the bins at the lowest level. Unlike the top-lever thermal optimizer, the front-level thermal optimizer deals with small bins that may contain only several grids. Furthermore, the cut-line to divide a bin is normally very close to the center of the bin. Hence, the front-level thermal optimizer constructs at most nine clusters for a bin by the intersection of at most four straight lines, as illustrated in Fig.5.8. In the figure, because the bin spans an odd number of grids horizontally, it is divided by two vertical lines separated by one grid. Similarly, because the bin spans an even number of grids vertically, it is evenly divided by a single horizontal line. As a result, a total of six clusters are constructed for the shown bin. By using this clustering mechanism, the front-level thermal optimizer solves a linear system that has a number of unknowns at most nine times the number of bins. Fig.5.9 shows the front-level thermal optimizer.

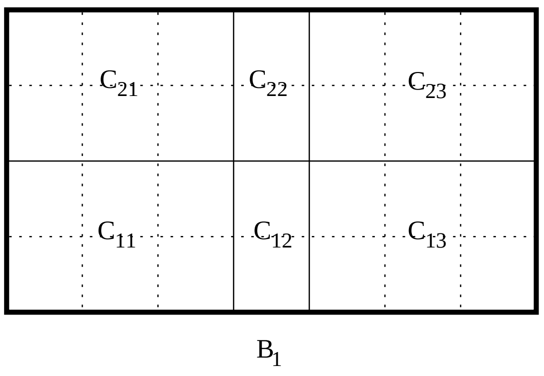


Figure 5.8: Clustering mechanism in front-level thermal optimizer.

```
1. Construct the clusters for each bin;
2. Compute $M^T D^{-1} G^2 Dp$ as the right hand side vector;
3. Solve (5.12) by the conjugate gradient method, with $x^*$
   as the unknown vector. The matrix-vector product is
   computed by
```

$$M^T D^{-1} G^2 DMx^*.$$

Figure 5.9: Front-level thermal optimizer for computing optimal power budget.

5.2.4 Computation in Top-Level and Front-Level Thermal Optimizers

The top-level thermal optimizer applies the GMRES method to solve the formulated linear equations, while the front-level thermal optimizer applies the CG method. Both the GMRES method and the CG method are well-known iterative methods [34, 55]. To solve a linear system $Ax = b$, an iterative method starts from an initial solution $x^{(0)}$ and then iteratively improves the solution until reaching an acceptable accuracy. Denote the exact solution of the linear system by x^*, the error at the i-th iteration by $e^{(i)}$, where $e^{(i)} = x^{(i)} - x^*$, and the residue by $r^{(i)}$, where $r^{(i)} = Ae^{(i)}$.

Generalized Minimal Residue Method

The GMRES method seeks from the Krylov subspace $K^i\left(A, r^{(0)}\right)$ an approximate solution x to x^* that minimizes the residue norm [55]:

$$\min_x \left\|r^{(0)} - Ax\right\|_2, x \in K^i\left(A, r^{(0)}\right)$$

where $K^i\left(A, r^{(0)}\right) \equiv span\left(r^{(0)}, Ar^{(0)}, A^2r^{(0)}, \cdots, A^{(i-1)}r^{(0)}\right)$. The method applies the Arnoldi process to construct the basis of $K^i\left(A, r^{(0)}\right)$, denoted by $V^{(i)}$:

$$AV^{(i)} = V^{(i+1)}H^{(i)}$$

where $H^{(i)}$ is an $(n+1) \times n$ matrix. $H^{(i)}$ consists of an upper Hessenberg matrix and an additional row vector which has only the last element being non-zero.

Let $x = Vy$. Then the following identify holds:

$$\left\|r^{(0)} - Ax\right\|_2 = \left\|e_1||r^{(0)}|| - H^{(i)}y\right\|_2. \tag{5.20}$$

Hence, the x vector that minimizes the residue norm can be computed from the related y vector. Since $H^{(i)}$ is almost triangular, the Givens rotation can be applied to obtain the optimal y vector efficiently to minimize $\left\|e_1||r^{(0)}|| - H^{(i)}y\right\|_2$.

To apply the GMRES method in the top-level thermal optimizer, the matrix-vector product Aw, where w denotes any column vector of $V^{(i)}$, can be computed by the procedure described in the previous section.

Conjugate Gradient Method

The CG method seeks an approximate solution x to x^* along a set of A-orthogonal directions $d^{(0)}, d^{(1)}, \cdots d^{(i)}$, where

$$\left(d^{(i)}\right)^T Ad^{(j)} = 0, \text{ for } i \neq j.$$

The CG method ensures that any direction $d^{(i)}$ is searched only once and never searched again [34]. Hence, in the CG method, the residue at the i-th iteration is orthogonal to all the previous search directions:

$$r^{(i)}d^{(j)} = 0, \text{ for } j < i.$$

Using a procedure similar to the Gram-Schmidt process, the CG method constructs the A-orthogonal search directions from the residue vectors $r^{(0)}, r^{(1)}, \cdots r^{(i)}$. If the initial error is represented in terms of the search directions by

$$e^{(0)} = \sum_{j=0}^{i} \delta_j d^{(j)}$$

the step size of the CG method, $\alpha^{(j)}$, given by

$$\alpha^{(j)} = \frac{\left(d^{(j)}\right)^T Ae^{(j)}}{\left(d^{(j)}\right)^T Ad^{(j)}} = -\delta_j$$

guarantees that the method eliminates one component of the initial error at each iteration.

Compared to the GMRES method, the CG method is only applicable to a Hermitian matrix. Because of the high efficiency, the front-level thermal optimizer adopts the CG method to solve the formed symmetric linear system.

Matrix Computation in Top-Level and Front-Level Thermal Optimizers

The previous sections briefly describe the matrix computations in the top-level and front-level thermal optimizers to solve the formulated linear equations. This section presents the details.

First, consider the top-level thermal optimizer. To compute the $M^T D^{-1} G^2 Dp$ in Fig.5.7, LOTAGre is employed to compute $D^{-1} G^2 Dp$, which is the temperature distribution caused by a fixed power vector p under the matrix of Green's function

```
1. Compute the 2-D DCT of the matrix of fixed powers $p$;
2. Multiply the result with $G^2$;
3. Compute the 2-D IDCT of the result at step 2;
4. Let the temperature result at step 3 be denoted by $\tilde{T}$;
5. Compute $M^T\tilde{T}$:
   for each bin do
       compute the average temperature of the first
       cluster using $\tilde{T}$;
       for each cluster other than the first cluster do
           compute its average temperature using $\tilde{T}$;
           compute the difference from the average
           temperature of the first cluster; The result
           is an entry of $M^T\tilde{T}$;
       end for;
    end for.
```

Figure 5.10: Procedure to compute $M^TD^{-1}G^2Dp$.

```
1. Compute the 2-D DCT of the matrix formed by $y$;
2. Multiply the result with $G^2$ and ;
3. compute the 2-D IDCT of the result at step 2;
3. Let the result at step 3 be denoted by $Z$:
   for each bin do
      compute the summation of the elements in $Z$ that
      represent the clusters inside this bin;
      Negate the result to form an entry of $r$;
   end for.
```

Figure 5.11: Procedure to compute $r = Cz_u - z_l$.

values G^2. Designate $\tilde{T}$ the temperature distribution computed: $\tilde{T} = D^{-1}G^2Dp$. Then each element of the vector $M^T\tilde{T}$ represents the average temperature difference between the first cluster and one of the other clusters in the same bin. Fig.5.10 shows the procedure to compute $M^TD^{-1}G^2Dp$.

With the vector $-M^T\tilde{T}$ computed, the vector $y_u = -M^TD^{-1}G^2Dp - C^Ty_l$ can be computed as follows: each element in y_l is assigned to the entry of $-M^T\tilde{T}$ that is for the first cluster of the related bin. Then each element of the vector y_u can be obtained by adding the entries in $-M^T\tilde{T}$ that are for the related bin. To compute $r = Cz_u - z_l$, where $z = D^{-1}G^2Dy$, Fig.5.11 shows the procedure.

Fig.5.10 and Fig.5.11 show that the $\mathcal{O}(n \lg n)$ DCT and IDCT procedures dom-

```
1. Let $Mx^*$ be denoted by $x'$;
2. for each bin do
       initialize all entries of $x'$ for this bin to $0$.
       for each cluster other than the first cluster do
          assign the first entry of $x^*$ that represents
          the cluster to the first entry of $x'$;
          subtract the same value from the first entry
          of $x'$ that represents the first cluster;
       end for;
       for each cluster do
           compute the average value of $x'$ for this cluster;
           assign the average value to every entry of $x'$
           for this cluster;
       end for;
   end for.
```

Figure 5.12: Compute Mx^*.

inate the run-time of one GMRES iteration. Here n denotes the number of grids for discretizing the layout. Note that although the GMRES method performs the Arnoldi process and solves a triangular system at the last iteration, these steps are inexpensive because the number of unknowns equals the number of bins, which is orders of magnitude smaller than the number of grids. However, the total number of GMRES iterations determines the overall time complexity of the top-level thermal optimizer.

Consider the front-level thermal optimization. To compute the $M^TD^{-1}G^2DMx^*$ in Fig.5.9, follow the steps shown in Fig.5.10. First Mx^* is computed, and denote the result by x'. Then follow the steps in Fig.5.10 to compute $M^TD^{-1}G^2Dx'$. In fact, Mx^* represents a power density distribution as the guess vector x^* is a vector of powers. The detailed procedure is shown in Fig.5.12.

Compared to the top-level thermal optimizer, the front-level thermal optimizer solves a larger linear system of size equal to the number of clusters. However, each iteration of the CG method employed requires fewer vector multiplications. As a result, the $\mathcal{O}(n \lg n)$ DCT and IDCT procedures still dominate the run-time of the front-level thermal optimizer in each iteration.

5.3 Application of Thermal Optimization in Capo

The partition tree is implicitly constructed level by level in the partition-driven approach. For the bins at the same level of the tree, the optimal power budget represents a power density map and can be solved by the top-level or front-level thermal optimizer. To optimize thermal during partitioning a bin, one method is from the optimal power budget to determine a power threshold for each child bin. Then the partition algorithm is modified not to move a cell to a child bin if the move causes the total power of the bin to exceed the power threshold. This is similar to the approach in [20]. An alternative method is to add to the partition objective, i.e., the placement cost, a penalty cost that measures the amount of power of the child bin that deviates from the optimal power budget. The latter method must trade off between the traditional placement cost, such as the HPWL, and the total power deviation of the child bin from the optimal power budget.

Assume that the FM algorithm is used in partitioning. Imposing power thresholds to optimize thermal is similar to the approach in [20]. When the FM algorithm moves a cell to a target bin, the move must neither incur cell overlaps or cause the total power of the target bin to exceed the power threshold. Furthermore, the cell moved must have the maximum gain among all the cells that satisfy the previous two conditions. On the other hand, to augment the placement cost for thermal optimization, the traditional placement cost needs to be slightly changed. If thermal optimization is not considered, the placement cost is actually the cut-set cost. When thermal optimization is considered, the placement cost is changed to the product of the cut-set cost and a penalty cost that accounts for the total power deviation of each child bin from its optimal power budget. Then in each pass of the FM algorithm, after one cell is moved, the placement cost is recalculated, and the initial moves that lead to the minimum placement cost in the pass are made permanent.

5.3.1 Optimal Power Budget in FM Based Algorithms

In partitioning bins, the Capo placement tool uses the FM algorithm and the multilevel FM ($MLFM$) algorithm, which is an extension to the FM algorithm. When a hyper-graph is to be partitioned by the $MLFM$ algorithm, first tightly connected nodes are grouped into clusters. Then a reduced hyper-graph is constructed by representing the clusters as nodes and retaining the node connectivity in the original graph. Next, perform the FM algorithm to partition the reduced hyper-graph, and convert the partition result into an initial partition for the original graph. From the initial partition, execute the FM algorithm again to obtain a final partition for the original graph. Because the $MLFM$ algorithm essentially employs the FM algorithm for partitioning, there hardly exist any differences in augmenting the FM algorithm and the $MLFM$ algorithm for thermal optimization.

The procedure to modify the placement cost for thermal optimization is detailed here. Given a placement, denote the cut-set cost by δ, the optimal power budget and the total power of the first bin by P_1^{opt} and P_1, respectively, and the optimal power budget and the total power of the second bin by P_2^{opt} and P_2, respectively. Let the placement cost when thermal optimization is considered be denoted by δ'. Then δ' is given by the product of δ and a penalty cost $p\left(\delta''\right)$ that measures the total power deviation of each bin from its optimal power budget:

$$\delta' = \delta \cdot p\left(\delta''\right)$$

where

$$p\left(\delta''\right) = \begin{cases} \left(1 + \delta'' - min\_power\_deviation\right)^{\alpha}, & \delta'' > min\_power\_deviation \\ \beta, & \text{otherwise} \end{cases}$$

$$\delta'' = \begin{cases} 1 - P_2/P_2^{opt}, & P_1^{opt} \leq 0 \\ 1 - P_1/P_1^{opt}, & P_2^{opt} \leq 0 \\ \max\left(\text{abs}(1 - P_2/P_2^{opt}), \text{abs}\left(1 - P_1/P_1^{opt}\right)\right), & \text{otherwise.} \end{cases}$$

The above formula for δ'' is explained below. The percentage of power deviation from the optimal power budget (called the percentage of power deviation) for the first bin and that for the second bin are given by abs $\left(1 - P_1/P_1^{opt}\right)$ and abs $\left(1 - P_2/P_2^{opt}\right)$, respectively. If the optimal power budget for the first bin is non-positive, then δ'' chooses the percentage of the power deviation of the second bin as its value. Similarly, if the optimal power budget for the second bin is non-positive, then δ'' chooses the percentage of the power deviation of the first bin as its value. Using this strategy avoids negative values for δ'' and effectively integrates into the placement cost the total power deviation of each placement bin from its optimal power budget. If both the optimal power budget values are positive, then δ'' chooses the maximum of the two percentages of power deviation as the value. The penalty cost $p\left(\delta''\right)$ is defined such that if δ'' is large than $min\_power\_deviation$, $p\left(\delta''\right)$ incurs a large penalty cost $\left(1 + \delta'' - min\_power\_deviation\right)^{\alpha}$, compared to the penalty cost β otherwise.

5.3.2 Optimal Power Budget in Branch and Bound Algorithm

Besides the FM and $MLFM$ algorithms, the Capo placement tool also includes a branch and bound partition algorithm to handle the end-case cell placements [15]. To place n cells in two bins, the branch and bound algorithm performs a depth-first traverse of a binary tree with n levels. In the binary tree, each node represents a cell, each branch under the node indicates a partition decision for the cell, and the path from the root to the node indicates a partial placement of all the nodes on the path. When visiting a node, the algorithm first estimates a lower bound for the cost of the partial placement of all the parent nodes (namely the bounding step). If the lower bound is larger than the cost of a complete placement previously generated by the algorithm, the sub-tree under the visited node will not be traversed (namely the pruning step). Otherwise, the algorithm diverges into two partition decisions for the visited node (namely the branching step), when deciding to assign the node to either the first bin or the second. The traverse process is recursive and continues until all the branches in the binary tree have been either visited or pruned. Because the search space can be extremely large, the traverse process may stop earlier if it has visited a predefined number of nodes.

Given a partial placement of cells, a lower bound for the placement cost can be straightforwardly given if thermal optimization is not considered. For instance, the Capo tool uses the cut-set cost of the partial placement as the lower bound. The cut-set cost must be a valid lower bound because the cut-set cost can only be increased, and any complete placement after the partial placement will only introduce more nets to the cut-set. Denote the lower bound when thermal optimization is not considered by B:

$$B = \sum_{\epsilon \in C_\Pi} c_\epsilon$$

i.e., the lower bound B is the summation of the cost of edges in the cut-set for the partial partition Π. When thermal optimization is considered, a lower bound for the cost of the partial placement can be given by the product of B and a lower bound for the percentages of power deviation. After the partial placement of cells, the total power of the first bin becomes P_1 and that of the second bin becomes P_2. Denote the lower bound when thermal optimization is considered by B'. Then B' is given by the product of B and the penalty cost $p\left(B''\right)$, where B'' gives a lower bound for the percentages of power deviation:

$$B' = B \cdot p\left(B''\right)$$

where

$$B'' = \begin{cases} \left(P_1 - P_1^{opt}\right)/P_2^{opt}, & P_1^{opt} \leq 0 \\ \left(P_2 - P_2^{opt}\right)/P_1^{opt}, & P_2^{opt} \leq 0 \\ B''', & \text{otherwise} \end{cases}$$

$$B''' = \begin{cases} \max\left(\frac{P_1 - P_1^{opt}}{P_1^{opt}}, \frac{P_1 - P_1^{opt}}{P_2^{opt}}\right), & P_1 > P_1^{opt} \\ \max\left(\frac{P_2 - P_2^{opt}}{P_1^{opt}}, \frac{P_2 - P_2^{opt}}{P_2^{opt}}\right), & P_2 > P_2^{opt} \\ min\_power\_deviation, & \text{otherwise.} \end{cases}$$

The above formulas are explained below. When P_1^{opt}, the optimal power budget for the first bin, is non-positive, $\left(P_1 - P_1^{opt}\right)/P_2^{opt}$ must be a lower bound for the percentages of power deviation because at least an amount of power $P_1 - P_1^{opt}$ will never be allocated to the second bin. Similarly, if the optimal power budget for the second bin, P_2^{opt}, is non-positive, $\left(P_2 - P_2^{opt}\right)/P_1^{opt}$ must be a lower bound for the percentages of power deviation. When both the optimal power budget values are positive, if neither bin has overfilled the optimal power budget, the lower bound for the percentages of power deviation cannot be estimated. The reason is that any estimation within $(0, 1)$ may be invalidated by constructing a partition such that the optimal power budget for every bin is satisfied. In this case, the predefined value *min_power_deviation* is used as a lower bound. However, if either of the bin has overfilled the optimal power budget, the amount of power overfill can be used to derive a lower bound.

Assume that the first bin has overfilled the optimal power budget, i.e., $P_1 > P_1^{opt}$. Then the percentage of power deviation for the first bin is at least $\left(P_1 - P_1^{opt}\right)/P_1^{opt}$, i.e., the amount of power overfilling the first bin divided by the optimal power budget. Furthermore, the percentage of power deviation for the second bin is at least $\left(P_1 - P_1^{opt}\right)/P_2^{opt}$, i.e., the amount of power under-filling the second bin divided by the optimal power budget. Obviously, the maximum of the two values, $\max\left(\left(P_1 - P_1^{opt}\right)/P_1^{opt}, \left(P_1 - P_1^{opt}\right)/P_2^{opt}\right)$, is a lower bound for the two percentages of power deviation. Similarly, if the second bin has overfilled the optimal power budget, $\max\left(\left(P_2 - P_2^{opt}\right)/P_1^{opt}, \left(P_2 - P_2^{opt}\right)/P_2^{opt}\right)$ must be a valid lower bound.

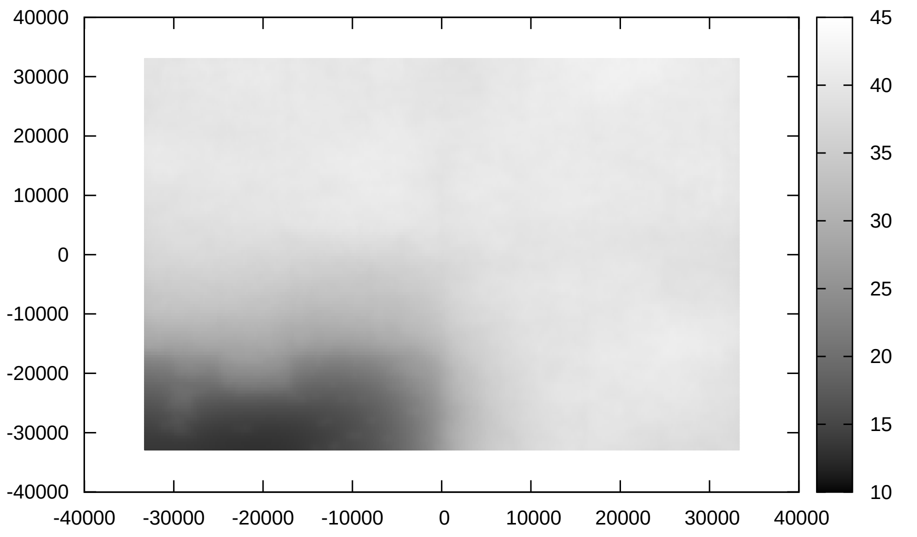
5.4 Experimental Results

The optimal power budget model was incorporated into version 45 of the Capo placement tool to optimize thermal. The top-level and front-level thermal optimizers adopt the implementation of the GMRES and CG methods from the iterative methods library (IML++) [2]. The experiments were based on the IBM-PLACE 2.0 benchmark suites [3]. The die sizes of the benchmark circuits were fixed to 2cm $\times$ 2cm. The other thermal parameters were the same as those for the example chip in Fig.3.4: the chip consisted of three layers, $\bar{h} = 8675$ W/(m^2·K), $\underline{h} = 1387$ W/(m^2·K), $k_1 = 98.4$ W/m·K, $k_2 = 16.2$ W/m·K, and $k_3 = 261.5$ W/m·K. The powers of the cells in each benchmark circuit were generated using a uniform distribution, with a maximal cell power and a total power set for the chip.

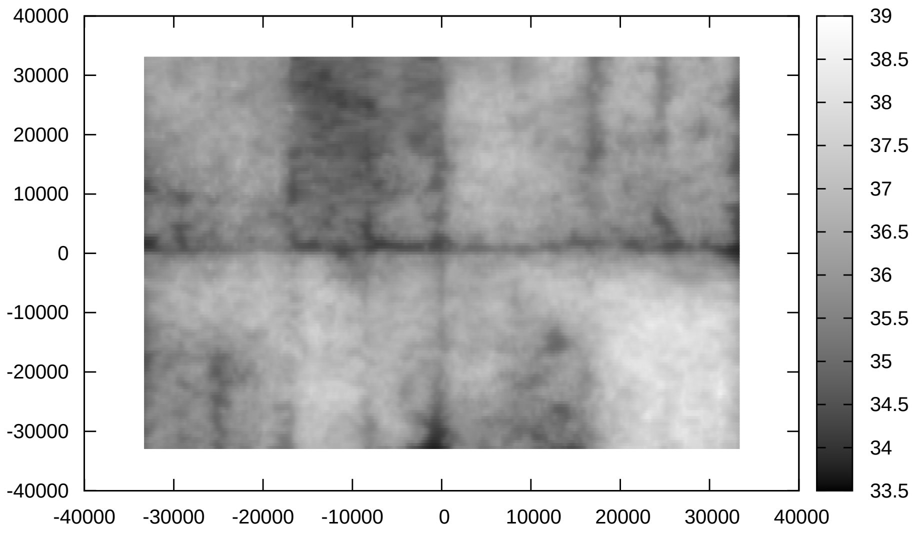
First, the parameters α, β and $min\_power\_deviation$ were set to 2, $1e-3$, and 5%. These parameter values implied that in the partition procedure by the FM

algorithm, if δ'', which reflected the percentages of power deviations for the two bins, was no more than 5%, the penalty cost was only $1e-3$; however, if δ'' was larger than 5%, the penalty cost was $(\delta'' - min\_power\_deviation + 1)^2$, which was at least 1000 times the value $1e-3$. Therefore, a placement that had δ'' larger than 5% was highly unlikely to be selected as the final placement, compared to another placement that had δ'' no more than 5%. However, when the placements considered all had δ'' larger than 5%, the chance for a placement to be selected as the final placement highly depended on the cut-set cost or the total HPWL.

Fig.5.13 shows the temperature distribution results for the IBM01 circuit. The total power of the chip was 140 W and the maximal cell power was 0.05 W. The first placement for the circuit was produced by Capo without thermal optimization. Thermal simulation showed that the temperatures in the placement were within $[12.9131°\text{C}, 42.2375°\text{C}]$. The total HPWL of the placement was $5.11015e7$. Then the second placement was produced for the circuit by Capo with thermal optimization. The temperatures in the second placement were within $[33.7226°\text{C}, 38.5310°\text{C}]$. In comparison, the temperature spread of the first placement was 551% larger than that of the second placement. The total HPWL of the second placement increased by 5.1% to $5.37073e7$. The average temperatures of the two placements were both 36.0927°C. However, the temperature standard deviation of the first placement was 7.13684°C, which was 767% larger than that of the second placement, 0.8229°C. With thermal optimization, the run-time of Capo increased by 6.25% from 80 s to 85 s (the Capo tool was run on a Debian Linux machine configured with an Intel Dual Core 2.4GHz CPU). Fig.5.14(a) shows the temperature histograms for the two placements. Clearly, the temperature spread was significantly reduced in the second placement because of thermal optimization by Capo.



(a) Temperature distribution of the first placement without thermal optimization.



(b) Temperature distribution of the second placement with thermal optimization.

Figure 5.13: Temperature distribution results for IBM01 circuit with and without thermal optimization: $\alpha = 2, min\_power\_deviation = 5\%$.

Table 5.1 summarizes the experimental results for the entire benchmark circuits. In the table, $\delta T_{range} = \frac{T_{max}-T_{min}}{T_{max}^{opt}-T_{min}^{opt}} - 1$, where T_{max}^{opt} and T_{min}^{opt} are the maximal and minimal temperatures in the placement generated by Capo with thermal optimization. δT_{range} measures the reduction of temperature spread in the placement with thermal optimization. The rate of the HPWL increase is denoted by $\delta W = \frac{W^{opt}}{W} - 1$, where W^{opt} denotes the HPWL for the placement with thermal optimization. Table 5.2 shows the temperature statistics for the placements. In summary, for the placements with thermal optimization, the average increase of the HPWL was 5.14%, the average reduction of the temperature spread was 288%, and the average reduction of the temperature standard deviation was 326%. The average increase of the run-time because of thermal optimization was 17.53%, which was 10-20 times smaller than the run-time results reported in [20]. Experiments demonstrated that using the optimal power budget model significantly reduced the temperature spreads and evenly distributed the temperatures in the chip. Results also showed that LOTAGre was very fast for thermal optimization purposes, compared to the other thermal analysis methods.

Then the α parameter was changed to $\alpha = 3$, while the other parameter values were retained. Table 5.3 and Table 5.4 show the results. As expected, because of the increased penalty cost for the percentages of power deviations, the average reduction of the temperature standard deviation increased from 326% to 412%. However, unexpectedly, the average increase of the HPWL was slightly reduced from 5.14% to 5.08%. The slight reduction may be explained by the randomness inherent in the Capo placement tool.

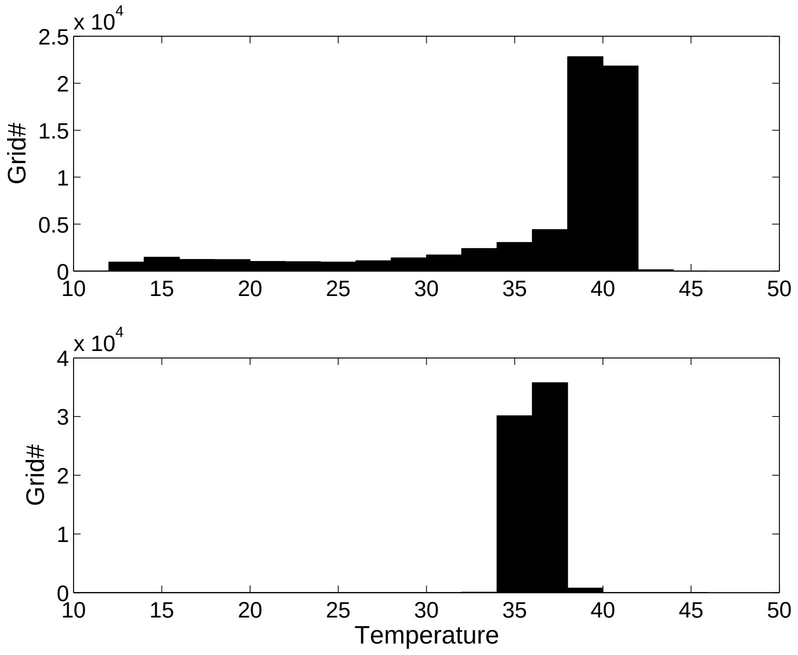
Next, the *min_power_deviation* parameter was increased to 10%, while the other parameters values were retained: $\alpha = 2, \beta = 0.001$. Because the penalty cost for the

Circuit	T_{min}	T_{max}	δT_{range} $T_{max} - T_{min}$	W $(\times 1e8)$	δW	t (s)	δt
IBM01	12.9131	42.2375		0.5110		80	
	33.7226	38.5310	510%	0.5371	5.10%	85	6.25%
IBM02	26.1408	38.6625		1.4499		179	
	32.5521	37.9126	134%	1.5106	4.19%	193	7.82%
IBM07	61.2283	95.2807		3.4109		428	
	79.1161	93.5680	136%	3.5590	4.34%	551	28.74%
IBM08	30.6825	70.1595		3.5244		465	
	56.8059	65.7273	343%	3.7082	5.22%	551	18.50%
IBM09	27.3360	51.0066		3.0331		458	
	43.7113	49.2200	330%	3.2144	5.98%	573	25.11%
IBM10	48.2706	86.2589		5.9315		756	
	73.3701	81.7177	355%	6.1607	3.86%	890	17.73%
IBM11	64.0321	96.9821		4.4554		687	
	79.5660	95.2184	111%	4.6820	5.09%	798	16.16%
IBM12	37.0579	102.7010		7.7678		869	
	83.4075	96.8102	390%	8.3363	7.32%	1042	19.91%
Average			288%		5.14%		17.53%

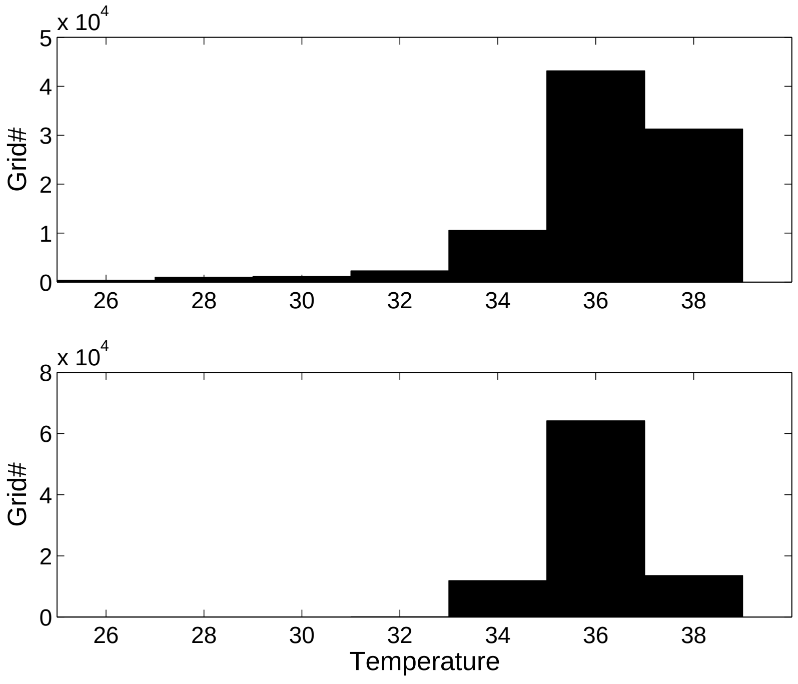
Table 5.1: Thermal optimization results for IBM-PLACE 2.0 benchmark circuits: $\alpha = 2$, $min\_power\_deviation = 5\%$.

Circuit	T_{avg}	T_{avg}^{opt}	σ	σ^{opt}	$\delta\sigma$
IBM01	36.0927	36.0927	7.1368	0.8229	767%
IBM02	36.0994	36.0998	1.8187	1.8187	95%
IBM07	87.7027	87.7042	6.5670	2.9188	125%
IBM08	61.9048	61.9049	5.5929	1.4061	298%
IBM09	47.1092	47.1092	3.6620	0.9563	283%
IBM10	77.3945	77.3973	7.2137	1.4613	394%
IBM11	90.2907	90.2902	7.1385	2.8893	147%
IBM12	90.2931	90.2917	12.8482	2.1340	502%
Average					326%

Table 5.2: Temperature statistics for the placements with and without thermal optimization:$\alpha = 2$, $min\_power\_deviation = 5\%$.

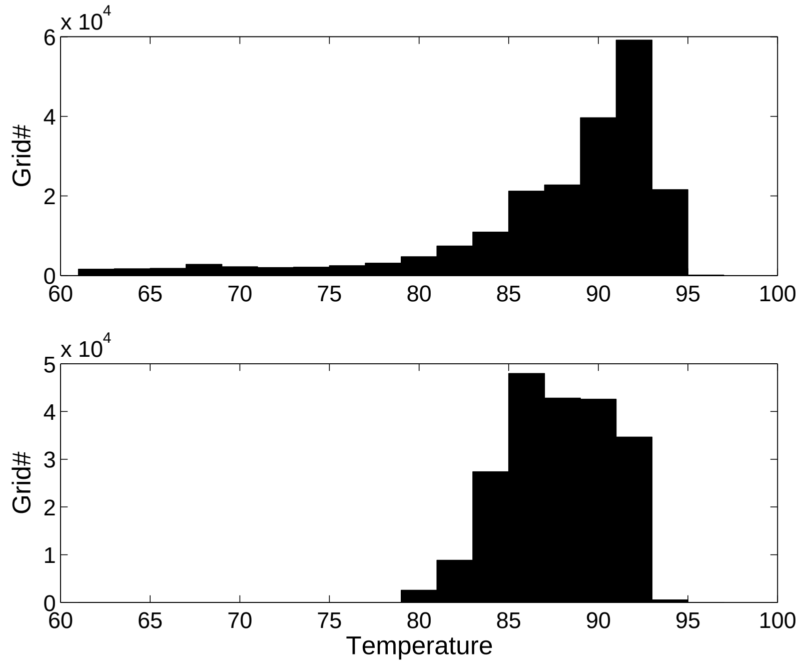


(a) Temperature histograms for IBM01 circuit.

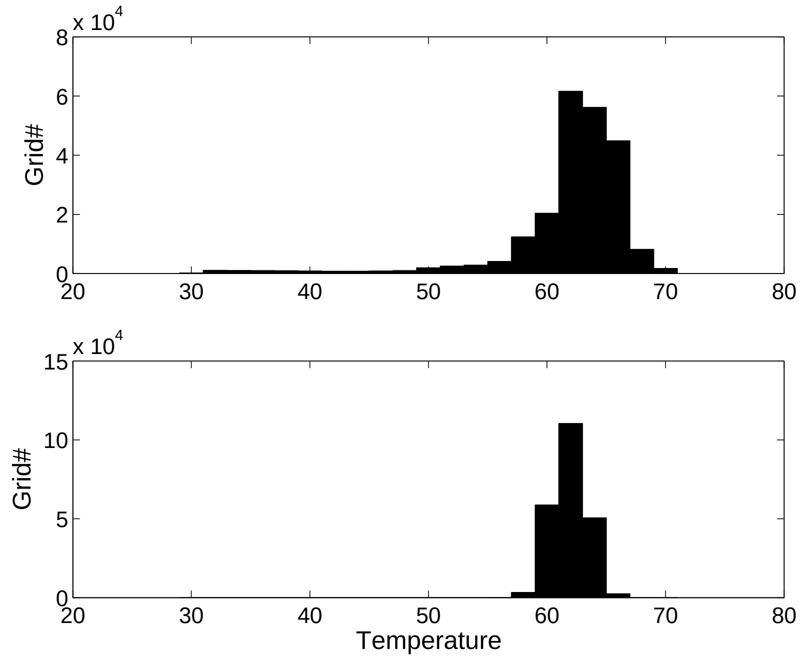


(b) Temperature histograms for IBM02 circuit.

Figure 5.14: Temperature histograms for IBM01 and IBM02 circuits with and without thermal optimization: $\alpha = 2, min\_power\_deviation = 5\%$. Upper diagram for the placement without thermal optimization and lower diagram for the placement with thermal optimization.

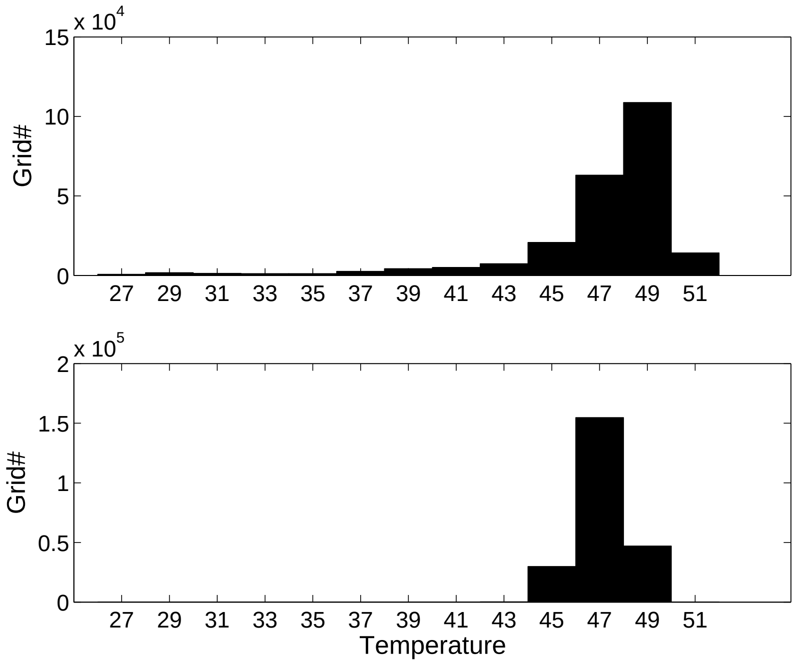


(a) Temperature histograms for IBM07 circuit.

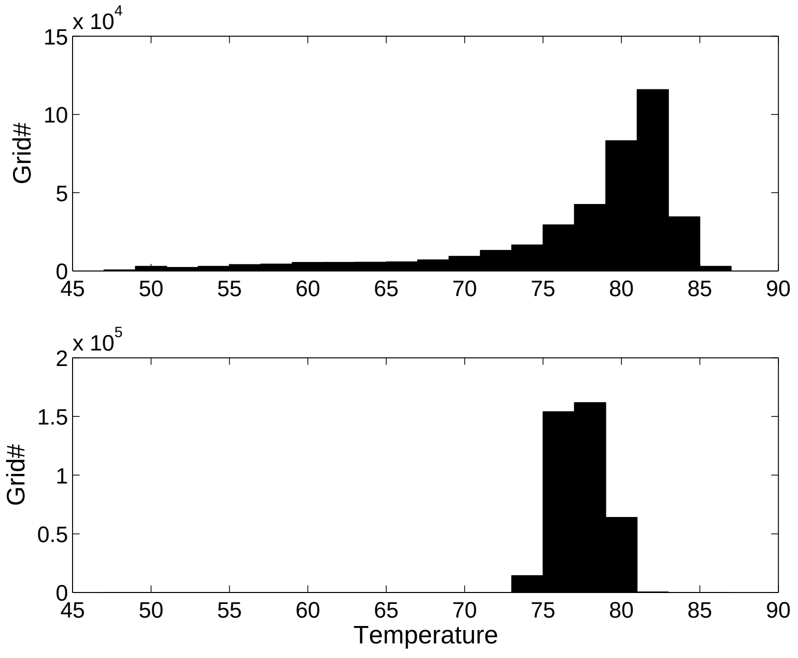


(b) Temperature histograms for IBM08 circuit.

Figure 5.15: Temperature histograms for IBM07 and IBM08 circuits with and without thermal optimization: $\alpha = 2, min\_power\_deviation = 5\%$. Upper diagram for the placement without thermal optimization and lower diagram for the placement with thermal optimization.

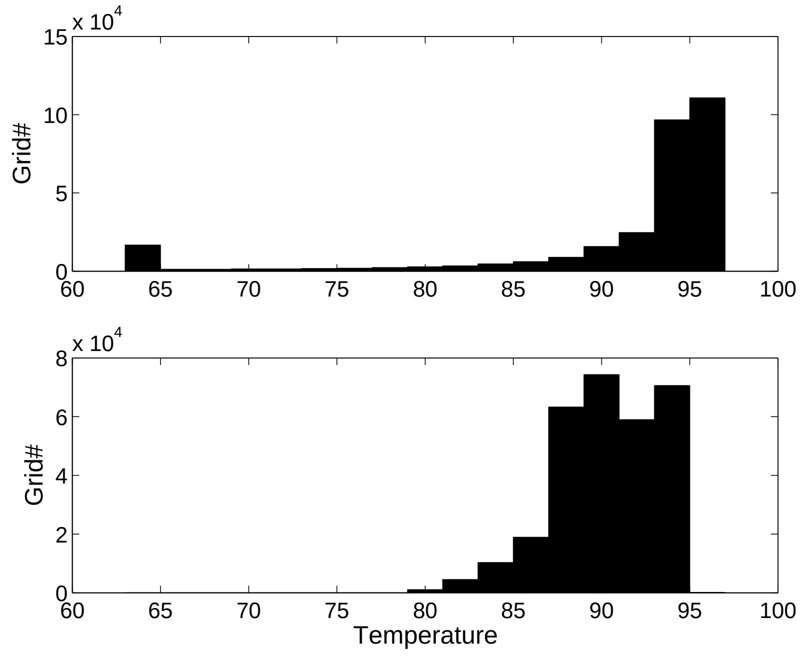


(a) Temperature histograms for IBM09 circuit.

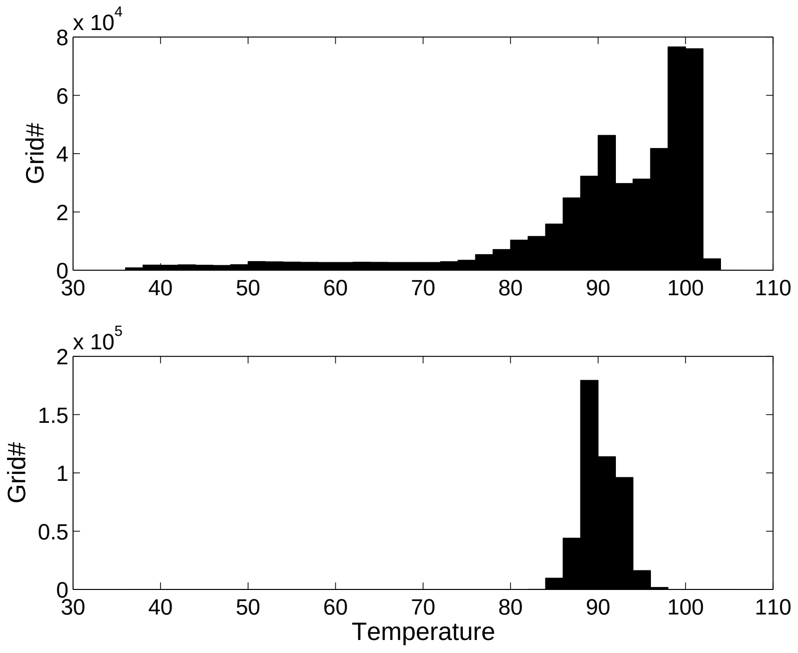


(b) Temperature histograms for IBM10 circuit.

Figure 5.16: Temperature histograms for IBM09 and IBM10 circuits with and without thermal optimization: $\alpha = 2, min\_power\_deviation = 5\%$. Upper diagram for the placement without thermal optimization and lower diagram for the placement with thermal optimization.



(a) Temperature histograms for IBM11 circuit.



(b) Temperature histograms for IBM12 circuit.

Figure 5.17: Temperature histograms for IBM11 and IBM12 circuits with and without thermal optimization: $\alpha = 2, min\_power\_deviation = 5\%$. Upper diagram for the placement without thermal optimization and lower diagram for the placement with thermal optimization.

Circuit	T_{min}	T_{max}	δT_{range}	W	δW	t	δt
			$T_{max} - T_{min}$	$(\times 1e8)$		(s)	
IBM01	12.9131	42.2375		0.5110		80	
	33.2534	37.9748	521%	0.5386	5.40%	90	12.50%
IBM02	26.1408	38.6625		1.4499		179	
	32.5877	37.9544	133%	1.5132	4.36%	197	10.6%
IBM07	61.2283	95.2807		3.4109		428	
	80.5412	92.0629	196%	3.5641	4.49%	566	32.24%
IBM08	30.6825	70.1595		3.5244		465	
	57.9678	64.8735	472%	3.7634	6.78%	557	19.79%
IBM09	27.3360	51.0066		3.0331		458	
	43.2227	48.8663	319%	3.1881	5.11%	591	29.04%
IBM10	48.2706	86.2589		5.9315		756	
	73.0699	81.0288	377%	5.9923	1.02%	933	23.41%
IBM11	64.0321	96.9821		4.4554		687	
	83.3869	93.4120	229%	4.7894	7.50%	842	22.56%
IBM12	37.0579	102.7010		7.7678		869	
	83.403	95.4998	443%	8.2312	5.97%	1065	22.56%
Average			336%		5.08%		21.52%

Table 5.3: Thermal optimization results for IBM-PLACE 2.0 benchmark circuits: $\alpha = 3$, $min\_power\_deviation = 5\%$.

Circuit	T_{avg}	T_{avg}^{opt}	σ	σ^{opt}	$\delta\sigma$
IBM01	36.0927	36.0927	7.1368	0.6724	961%
IBM02	36.0994	36.1005	1.8187	0.9628	89%
IBM07	87.7027	87.7078	6.5670	2.0105	227%
IBM08	61.9048	61.9060	5.5929	1.0999	408%
IBM09	47.1092	47.1092	3.6620	0.9112	302%
IBM10	77.3945	77.3916	7.2137	1.4200	408%
IBM11	90.2907	90.2914	7.1385	1.3857	415%
IBM12	90.2931	90.2918	12.8482	2.1948	485%
Avg					412%

Table 5.4: Temperature statistics for the placements with and without thermal optimization: $\alpha = 3$, $min\_power\_deviation = 5\%$.

Circuit	T_{min}	T_{max}	δT_{range} $T_{max} - T_{min}$	W $(\times 1e8)$	δW	t (s)	δt
IBM01	12.9131	42.2375		0.5110		80	
	30.1853	39.580	212%	0.5365	4.99%	89	11.25%
IBM02	26.1408	38.6625		1.4499		179	
	29.4837	38.7822	35%	1.4997	3.43%	194	8.38%
IBM07	61.2283	95.2807		3.4109		428	
	73.3678	94.5063	61%	3.5137	3.01%	500	16.82%
IBM08	30.6825	70.1595		3.5244		465	
	56.2841	65.3251	337%	3.7106	5.28%	528	13.55%
IBM09	27.3360	51.0066		3.0331		458	
	42.0756	50.6158	177%	3.2202	6.17%	541	18.12%
IBM10	48.2706	86.2589		5.9315		756	
	64.4119	84.5991	88%	6.1433	3.57%	865	14.42%
IBM11	64.0321	96.9821		4.4554		687	
	72.9936	96.6296	39%	4.6399	4.14%	780	13.54%
IBM12	37.0579	102.7010		7.7678		869	
	75.6084	98.6316	185%	8.4000	8.14%	1071	23.25%
Average			142%		4.84%		14.92%

Table 5.5: Thermal optimization results for IBM-PLACE 2.0 benchmark circuits: $\alpha = 2$, $min\_power\_deviation = 10\%$.

percentages of power deviations were reduced after the $min\_power\_deviation$ parameter, it was predicted that the temperature distributions of the placements worsened and the HPWLs improved. The results are shown in Table 5.7 and Table 5.8. Consistent with the prediction, the average reduction of the temperature spread decreased from 288% to 142%, the average reduction of the temperature standard deviation decreased from 326% to 114%, and the average increase of the HPWL decreased from 5.14% to 4.84%, compared to the case that $\alpha = 2, min\_power\_deviation = 5\%$.

The parameter values were set to $\alpha = 3, min\_power\_deviation = 10\%$ in the final set of experiments. Table 5.7 and Table 5.8 show the results. Compared to the case that $\alpha = 2, min\_power\_deviation = 10\%$, the average reduction of the temperature standard deviation and the average increase of the HPWL increased as expected. The slight decrease in the average reduction of the temperature spread may still be explained by the randomness inherent in the Capo placement tool.

Circuit	T_{avg}	T_{avg}^{opt}	σ	σ^{opt}	$\delta\sigma$
IBM01	36.0927	36.0927	7.1368	2.1921	226%
IBM02	36.0994	36.1022	1.8187	1.7201	6%
IBM07	87.7027	87.7057	6.5670	4.8125	36%
IBM08	61.9048	61.9054	5.5929	1.7888	213%
IBM09	47.1092	47.1092	3.6620	1.5838	131%
IBM10	77.3945	77.3929	7.2137	4.1254	75%
IBM11	90.2907	90.2904	7.1385	4.8059	49%
IBM12	90.2931	90.2908	12.8482	4.6153	178%
Average					114%

Table 5.6: Temperature statistics for the placements with and without thermal optimization: $\alpha = 2$, $min\_power\_deviation = 10\%$.

Circuit	T_{min}	T_{max}	δT_{range} $T_{max} - T_{min}$	W $(\times 1e8)$	δW	t (s)	δt
IBM01	12.9131	42.2375		0.5110		80	
	30.1267	40.9214	272%	0.5429	6.25%	91	13.75%
IBM02	26.1408	38.6625		1.4499		179	
	29.7709	38.8480	38%	1.5290	5.45%	191	6.70%
IBM07	61.2283	95.2807		3.4109		428	
	71.8602	93.9542	54%	3.5651	4.52%	542	26.64%
IBM08	30.6825	70.1595		3.5244		465	
	55.5729	65.6557	292%	3.7766	7.16%	545	17.20%
IBM09	27.3360	51.0066		3.0331		458	
	40.1923	51.2919	113%	3.1465	3.74%	589	28.60%
IBM10	48.2706	86.2589		5.9315		756	
	66.9832	84.2087	121%	6.0130	1.37%	873	15.48%
IBM11	64.0321	96.9821		4.4554		687	
	76.2216	95.9343	67%	4.7371	6.32%	781	13.68%
IBM12	37.0579	102.7010		7.7678		869	
	74.0894	99.7213	156%	8.4618	13.9%	1090	25.43%
Average			139%		6.09%		18.44%

Table 5.7: Thermal optimization results for IBM-PLACE 2.0 benchmark circuits: $\alpha = 3$, $min\_power\_deviation = 10\%$.

Circuit	T_{avg}	T_{avg}^{opt}	σ	σ^{opt}	$\delta\sigma$
IBM01	36.0927	36.0927	7.1368	2.1731	228%
IBM02	36.0994	36.1045	1.8187	1.5983	14%
IBM07	87.7027	87.7062	6.5670	4.9034	34%
IBM08	61.9048	61.9086	5.5929	1.7576	218%
IBM09	47.1092	47.1092	3.6620	1.5194	141%
IBM10	77.3945	77.3998	7.2137	3.6386	98%
IBM11	90.2907	90.2924	7.1385	4.3066	66%
IBM12	90.2931	90.2917	12.8482	5.1360	150%
Avg					119%

Table 5.8: Temperature statistics for the placements with and without thermal optimization: $\alpha = 3$, $min\_power\_deviation = 10\%$.

In general, increasing the α parameter will reduce the temperature standard deviation and increase the HPWL. Similarly, relaxing the $min\_power\_deviation$ parameter will increase the temperature standard deviation and reduce the HPWL. However, exceptions can occur because of the randomness inherent in the Capo placement tool or the correlation between the powers of the cells and the HPWL. For example, consider two placements A and B, where the percentage of power deviation of A is 4% and that of B is 3%. Let $\alpha = 2$. If the $min\_power\_deviation$ parameter is set to 5%, selecting either A or B as the final placement depends on which placement has a larger HPWL. However, when the percentage of power deviation of A increases to 8%, it becomes unlikely to select A as the final placement. If the power deviation of B also increases to over 5%, such as 7%, the chance of selecting A as the final placement depends on if the HPWL of B is at least 1.97% (i.e.,$\left(\frac{1+0.03}{1+0.02}\right)^2 - 1$) larger than that of A. One step further, let $min\_power\_deviation$ increase to 6%. It is expected that the final placement will have a smaller HPWL and a larger temperature standard deviation, compared to the case that $\alpha = 2, min\_power\_deviation = 5\%$. However, calculations show a contradiction. When $min\_power\_deviation$ is 6%, the likelihood of selecting A as the final placement depends on if the HPWL of B is at least 1.99% (i.e.,$\left(\frac{1+0.02}{1+0.01}\right)^2 - 1$) larger than that of A. Assume that the HPWL of B is 1.98% larger than that of A. When $min\_power\_deviation = 5\%$, A is selected as the final placement. But when $min\_power\_deviation$ is increased to 6%, B is selected as the final placement. In other words, relaxing the $min\_power\_deviation$ parameter causes an increase of the HPWL and a decrease of the power deviation. This type of counter-intuitive result, together with the randomness inherent in the Capo placement tool, may complicate the experimental results. In summary, by slightly trading off the total HPWL, using the optimal power budget model in the

Capo placement tool significantly improved the temperature distribution of the chip.

CHAPTER VI

Conclusions and Future Works

The continual scaling of transistors and interconnects exacerbates the thermal management problems for ULSI chips. Accurate estimation and effective optimization of the temperature distribution of a ULSI chip become utterly important in predicting and ensuring the performance and reliability of the chip before actual fabrication. Motivated by the design challenges, this dissertation aims at a detailed study of the chip-level thermal issues. In summary, the dissertation contributes primarily in three areas: chip-level thermal analysis, interconnect thermal modeling, and thermal optimization in cell placement. First, the dissertation introduces LOTAGre, a high-efficiency $\mathcal{O}(n \lg n)$ multilayer Green's function based thermal analysis method. Next, the dissertation presents a Schafft-type interconnect temperature distribution model and an $\mathcal{O}(n)$ algorithm to compute the interconnect temperature distribution from the model. Finally, the dissertation introduces an optimal power budget model for thermal optimization in the cell placement stage and details the integration of the model into the widely distributed Capo placement tool.

6.1 Contributions to Thermal Analysis

This dissertation introduces a chip-level thermal analysis method called LOTAGre. Compared to grid-based methods such as the FE and FD methods, LOTAGre utilizes the multilayer heat conduction Green's function to avoid dispensing large numbers of grids to chip regions with no heat sources and no monitored temperatures. Using the DCT and IDCT algorithms, LOTAGre achieves $\mathcal{O}(n \lg n)$ run-time in thermal analysis. Comparisons have shown that LOTAGre can be orders of magnitude faster than a sophisticated computational fluid dynamics tool called FLUENT, a typical grid-based tool, while providing the same accuracy. Using the multilayer thermal model, LOTAGre is capable of handling chips consisting of multilayer heterogeneous heat conduction materials, with either wire-bonding packaging or flip-chip packaging.

This dissertation also discusses the ambient temperature effects on temperature distribution within the chip. Traditional thermal analysis methods have assumed a uniform ambient temperature surrounding the chip. The assumption may cause large errors because the temperature gradients at different boundaries of the chip are dissimilar and the heat flow from different surfaces of the chip to the outer environment is unbalanced. Using general 2-D functions to model the ambient temperatures at the top and bottom surfaces of the chip, this dissertation separates the temperature distribution of the chip into two parts: (a) homogeneous temperature distribution attributed to ambient temperatures, and (b) inhomogeneous temperature distribution attributed to the heat sources inside the chip. Both the temperature distributions are computed by highly efficient procedures of $\mathcal{O}(n \lg n)$ complexity in LOTAGre.

In analyzing the inhomogeneous temperature distribution, this dissertation integrates the eigen-expansion technique and the transmission line theory to derive fully analytical formulas for the multilayer heat conduction Green's function, including the s-domain version. With the multilayer heat conduction Green's function, the temperature distribution caused by an arbitrarily shaped heat source can be computed, and most important, thermal transfer impedance between any two locations can be given, and compact thermal models can be established for the critical components in the chip.

This dissertation also analyzes the errors in LOTAGre. One type of error is caused by truncation of the infinite series. The dissertation provides a bounding technique to determine an upper bound for the truncation error. Theoretical and numerical results show that the truncation error in LOTAGre is insignificant. The other type of error is caused by the sampling of power density distribution in the chip. The dissertation applies the Fourier analysis technique to obtain a power density sampling criterion similar to the Nyquist sampling criterion.

6.2 Contributions to Interconnect Thermal Modeling

The Schafft's model was initially used to model interconnect electromigration. Recently, the model was used to analyze the temperature distribution within an interconnect. Based on the Schafft's model, this dissertation introduces an interconnect temperature distribution model which includes flexible parameters to accurately model the thermal effects of packaging, ambient temperatures, and multiple heat conduction paths in the chip.

In existing interconnect temperature distribution models, the law of energy conservation is used to set up the appropriate differential equations. However, existing models have inadequately addressed the amount of heat dissipated vertically from the interconnect to the heat sink of the chip, and have neglected the effect of the temperature gradients within the interconnect. In establishing the interconnect temperature distribution model, this dissertation considers the effect of the temperature gradients to avoid overestimating the temperature variations within the interconnect.

Despite the increased number of parameters in the presented model, this dissertation gives an efficient $\mathcal{O}(n)$ approach to solve the interconnect temperature distribution.

6.3 Contributions to Thermal Optimization

This dissertation introduces an optimal power budget model for thermal optimization in the cell placement stage. The optimal power budget model determines the optimal allocation of cell powers to different regions of the chip so that the resultant temperature distribution most closely approximates the target temperature distribution for the chip. To solve the optimal power budget from the formulated least-square form, the dissertation employs the GMRES method and the CG method as well as LOTAGre to construct highly efficient top-level and front-level thermal optimizers.

The dissertation then presents the procedures to incorporate the optimal power budget model into the partition-driven Capo placement tool for thermal optimization. The Capo placement tool augmented can rely on the top-level and front-level thermal optimizers to optimize the temperature distribution of the chip in the cell placement stage. Experiments showed that the placements generated by Capo with thermal optimization had significantly narrower temperature spreads than the placements without thermal optimization. Results also demonstrated that LOTAGre was advantageous in thermal optimization because of its superior speed over the grid-based methods.

6.4 Future Works

In chapter II, this dissertation derives the multilayer heat conduction Green's function, including the s-domain version. One possible future work is to apply the multilayer heat conduction Green's function to estimate the thermal transfer impedance between two interested locations in the chip and establish compact thermal models for the thermally critical components in the chip. By studying the thermal transfer properties of the on-chip components, insights may be gained into the temperature distribution of the chip to provide better thermal management design.

In chapter IV, this dissertation introduces a new interconnect temperature distribution model. One future research work is to apply the new model to study a large set of interconnect configurations and build figures of merit on the temperature distributions of the interconnect wires to aid the IC physical design processes, e.g., global routing, detail routing, and buffer insertion, in alleviating the ULSI thermal problems.

In chapter V, this dissertation reviews several cell placement approaches and details the incorporation of the optimal power budget model into the partition-driven Capo placement tool. Possible future research directions are: apply the model to the simulated-annealing-based approach and the force-directed approach; compare the thermal optimization results by these approaches; and apply the optimal power budget model for thermal optimization to the earlier floorplanning stage to further improve the temperature distribution of ULSI chips.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] http://www.mathworks.com/access/helpdesk/help/.

[2] http://math.nist.gov/iml++/.

[3] http://er.cs.ucla.edu/benchmarks/ibm-place2/.

[4] A.H. Ajami, K. Banerjee, and M. Pedram. Modeling and analysis of nonuniform substrate temperature effects on global ulsi interconnects. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 24(6):849–861, June 2005.

[5] J. Albers. An exact recursion relation solution for the steady-state surface temperature of a general multilayer structure. *IEEE Trans. Compon., Packag., Manuf. Technol. A*, 18(1):31–38, Mar. 1995.

[6] H.B. Bakoglu. *Circutis, Interconnects, and Packaging for VLSI.* Addison-Wesley publishing company, 1990.

[7] K. Banerjee, A. Amerasekera, G. Dixit, and Chenming Hu. The effect of interconnect scaling and low-*k* dielectric on the thermal characteristics of the IC metal. In *IEDM Tech. Dig.*, pages 65–68, Dec. 1996.

[8] K. Banerjee, A. Mehrotra, A. Sangiovanni-Vincentelli, and Chenming Hu. On thermal effects in deep sub-micron VLSI interconnects. In *Proc. ACM/IEEE Design Automation Conf.*, pages 885–891, June 1999.

[9] K. Banerjee, S.J. Souri, P. Kapur, and K.C. Saraswat. 3-D ICs: a novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration. *Proc. of IEEE*, 89(5):602–633, May 2001.

[10] W. Batty, C.E. Christoffersen, A.J. Panks, S. David, C.M. Snowden, and M.B. Steer. Electrothermal CAD of power devices and circuits with fully physical time-dependent compact thermal modeling of complex nonlinear 3-d systems. *IEEE Trans. Compon. Packag. Technol.*, 24(4):566–590, Dec. 2001.

[11] S. Bilbao and J.O.S.III Smith. Finite difference schemes and digital waveguide networks for the wave equation: stability, passivity, and numerical dispersion. *IEEE Trans. Acoust., Speech, Signal Process.*, 11(3):255–266, May 2003.

[12] A.A. Bilotti. Static temperature distribution in IC chips with isothermal heat sources. *IEEE Trans. Electron Devices*, 21(3):217–226, Mar. 1974.

[13] J.R. Black. Electromigration—a brief survey and some recent results. *IEEE Trans. Electron Devices*, 16(4):338–347, Apr. 1969.

[14] J.W. Brown and R.V. Churchill. *Fourier Series and Boundary Value Problems.* McGraw-Hill, 5th edition, 1993.

[15] A. E. Caldwell, A. B. Kahng, and I. L. Markov. Optimal partitioners and end-case placers for standard-cell layout. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 19(11):1304–1314, Nov. 2000.

[16] B.H. Calhoun, A. Wang, and A. Chandrakasan. Modeling and sizing for minimum energy operation in subthreshold circuits. *IEEE J. Solid-State Circuits*, 40(9):1778–1786, Sept. 2005.

[17] Mario R. Casu, Mariagrazia Graziano, Guido Masera, Gianluca Piccinini, M. M. Prono, and Maurizio Zamboni. Clock distribution network optimization under self-heating and timing constraints. In *PATMOS*, pages 198–208, Sept. 2002.

[18] M.R. Casu, M. Graziano, G. Masera, G. Piccinini, and M. Zamboni. An electromigration and thermal model of power wires for a priori high-level reliability prediction. *IEEE Trans. VLSI Syst.*, 12(4):349–358, April 2004.

[19] Danqing Chen, Erhong Li, E. Rosenbaum, and Sung-Mo Kang. Interconnect thermal modeling for accurate simulation of circuit timing and reliability. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 19(2):197–205, Feb. 2000.

[20] G. Chen and S. Sapatnekar. Partition-driven standard cell thermal placement. In *Proc. Int. Symp. Physical Design*, pages 75–80, Apr. 2003.

[21] Yi-Kan Cheng and Sung-Mo Kang. A temperature-aware simulation environment for reliable ULSI chip design. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 19(10):1211–1220, Oct. 2000.

[22] Yi-Kan Cheng, P. Raha, Chin-Chi Teng, E. Rosenbaum, and Sung-Mo Kang. ILLIADS-T: an electrothermal timing simulator for temperature-sensitive reliability diagnosis of CMOS VLSI chips. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 17(8):668–681, Aug. 1998.

[23] Ting-Yen Chiang, K. Banerjee, and K.C. Saraswat. Compact modeling and SPICE-based simulation for electrothermal analysis of multilevel ULSI interconnects. In *Proc. ACM/IEEE Int. Conf. on Computer-Aided Design*, pages 165–172, Nov. 2000.

[24] Tai-Yu Chou and Z.J. Cendes. Capacitance calculation of IC packages using the finite element method and planes of symmetry. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 13(9):1159–1166, Sept. 1994.

[25] C.C.N. Chu and D.F. Wong. A matrix synthesis approach to thermal placement. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 17(11):1166–1174, Nov. 1998.

[26] J. Clementi, J. McCreary, T.M. Niu, J. Palomaki, J. Varcoe, and G. Hill. Flip-chip encapsulation on ceramic substrates. In *Proc. Electronic Components and Technology Conf.*, pages 175–181, June 1993.

[27] L. Codecasa, D. D'Amore, and P. Maffezzoni. An Arnoldi based thermal network reduction method for electro-thermal analysis. *IEEE Trans. Compon., Packag., Manuf. Technol. A*, 26(1):186–192, Mar. 2003.

[28] R. Crampagne, M. Ahmadpanah, and J.-L. Guiraud. A simple method for determining the Green's function for a large class of MIC lines having multilayered dielectric structures. *IEEE Trans. Microw. Theory Tech.*, 26(2):82–87, Feb. 1978.

[29] A. Csendes, V. Szekely, and M. Rencz. An efficient thermal simulation tool for ICs, microsystem elements and MCMs: the μS-THERMANAL. *Microelectronics Journal*, 29(4):241–255, Apr. 1998.

[30] H. Eisenmann and F.M. Johannes. Generic global placement and floorplanning. In *Proc. ACM/IEEE Design Automation Conf.*, pages 269–274, 1998.

[31] C.M. Fiduccia and R.M. Mattheyses. A linear-time heuristic for improving network partitions. In *Proc. ACM/IEEE Design Automation Conf.*, pages 175–181, Jun. 1982.

[32] B. Goplen and S. Sapatnekar. Efficient thermal placement of standard cells in 3d ICs using a force directed approach. In *Proc. ACM/IEEE Int. Conf. on Computer-Aided Design*, pages 86–89, 2003.

[33] D. Harmon, J. Gill, and T. Sullivan. Thermal conductance of IC interconnects embedded in dielectrics. In *IEEE International Integrated Reliability Workshop final report*, pages 1–9, Oct. 1998.

[34] Magnus R. Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Res. of Nat. Burean of Stand.*, 49(6):409–436, Dec. 1952.

[35] Wei Huang, M.R. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy. Compact thermal modeling for temperature-aware design. In *Proc. ACM/IEEE Design Automation Conf.*, pages 878–883, June 2004.

[36] Sungjun Im and K. Banerjee. Full chip thermal analysis of planar (2-D) and vertically integrated (3-D) high performance ICs. In *IEDM Tech. Dig.*, pages 727–730, Dec. 2000.

[37] J.A.Roy and I.L.Markov. Seeing the forest and the trees: Steiner wirelength optimization in placement. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 26(4):632–644, Apr. 2007.

[38] J.V.Beck, K. Cole, and A. Haji-Sheikh. *Heat Conduction Using Green's Functions.* Hemisphere, 1992.

[39] K. Kanda, K. Nose, H. Kawaguchi, and T. Sakurai. Design impact of positive temperature dependence on drain currentin sub-1-V CMOS VLSIs. *IEEE J. Solid-State Circuits*, 36(10):1559–1564, Oct. 2001.

[40] A Kennings and K.P. Vorwerk. Force-directed methods for generic placement. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 25(10):2076–2087, Oct. 2006.

[41] C.H.-I. Kim, H. Soeleman, and K. Roy. Ultra-low-power DLMS adaptive filter for hearing aid applications. *IEEE Trans. VLSI Syst.*, 11(6):1058–1067, Dec. 2003.

[42] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 220, 4598:671–680, 1983.

[43] A.G. Kokkas. Thermal analysis of multiple-layer structures. *IEEE Trans. Electron Devices*, 21(11):674–681, Nov. 1974.

[44] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

[45] Fan Mo, A. Tabbara, and R.K. Brayton. A force-directed macro-cell placer. In *Proc. ACM/IEEE Int. Conf. on Computer-Aided Design*, pages 177–180, 2000.

[46] M.Rencz, V.SzÃ©kely, A.Poppe, and B.Courtois. Inclusion of RC compact models of packages into board level thermal simulation tools. In *Proc. of the 18th IEEE Semiconductor Thermal Measurement and Management Symposium*, pages 71–76, 2002.

[47] K. Nabors and J. White. FastCap: a multipole accelerated 3-D capacitance extraction program. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 10(11):1447–1459, Nov. 1991.

[48] A.M. Niknejad, R. Gharpurey, and R.G. Meyer. Numerically stable Green function for modeling and analysis of substrate coupling in integrated circuits. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 17(4):305–315, Apr. 1998.

[49] A. Odabasioglu, M. Celik, and L.T. Pileggi. PRIMA: passive reduced-order interconnect macromodeling algorithm. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 17(8):645–654, Aug. 1998.

[50] J. Parry, H. Rosten, and G.B. Kromann. The development of component-level thermal compact models of a C4/CBGA interconnect technology: the Motorola PowerPC 603 and PowerPC 604 risc microprocessors. *IEEE Trans. Compon., Packag., Manuf. Technol. A*, 21(1):104–112, Mar. 1998.

[51] J.R. Phillips and J.K. White. A precorrected-FFT method for electrostatic analysis of complicated 3-D structures. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 16(10):1059–1072, Oct. 1997.

[52] N. Quinn and M. Breuer. A forced directed component placement procedure for printed circuit boards. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 26(6):377–388, Jun. 1979.

[53] F. Romeo and Alberto L. Sangiovanni-Vincentelli. Probabilistic hill climbing algorithms: Properties and applications. Technical report, EECS Department, University of California, Berkeley, 1984.

[54] J.A. Roy, S.N. Adya, D.A. Papa, and I.L. Markov. Min-cut floorplacement. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 25(7):1313–1326, Jul. 2006.

[55] Y. Saad and M Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3):856–869, Jul. 1986.

[56] Sadiq M. Sait and Habib Youssef. *VLSI Physical Design Automation: Theory and Practice.* World Scientific Pub Co. Inc., 1999.

[57] T. Sakurai and A.R. Newton. Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas. *IEEE J. Solid-State Circuits*, 25(2):584–594, Apr. 1990.

[58] H.A. Schafft. Thermal analysis of electromigration test structures. *IEEE Trans. Electron Devices*, 34(3):664–672, Mar. 1987.

[59] C. Sechen and A. Sangiovanni-Vincentelli. The TimberWolf placement and routing packages. *IEEE J. Solid-State Circuits*, 20(2):510–522, Apr. 1985.

[60] SIA. *International Technology Roadmap for Semiconductors.* Semiconductor Industry Association, 2001.

[61] V. Szekely. Identification of RC networks by deconvolution: chances and limits. *IEEE Trans. Circuits Syst. I*, 45(3):244–258, Mar. 1998.

[62] V. Szekely. THERMODEL: a tool for compact dynamic thermal model generation. *Microelectron. J.*, 29:257–267, 1998.

[63] Vladimir Szekely, Andras Poppe, Marta Rencz, Miklos Rosental, and Tamas Teszeri. THERMAN: a thermal simulation tool for IC chips, microstructures and PW boards. *Microelectron. J.*, 40:517–524, 2000.

[64] L.H. Thomas. Elliptic problems in linear difference equations over a network. In *Watson Sci. Comput. Lab. Rept., Columbia University*, 1949.

[65] Ching-Han Tsai and Sung-Mo Kang. Cell-level placement for improving substrate thermal distribution. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 19(2):253–266, Feb. 2000.

[66] Baohua Wang and Pinaki Mazumder. Subgridding method for speeding up FD-TLM circuit simulation. In *Proc. Int. Symp. Circuits and Systems*, pages 20–23, Thailand, May 2003.

[67] Baohua Wang and Pinaki Mazumder. Fast thermal analysis for VLSI circuits via semi-analytical Green's function in multi-layer materials. In *Proc. Int. Symp. Circuits and Systems*, volume 2, pages 409–412, Canada, May 2004.

[68] Baohua Wang and Pinaki Mazumder. On optimality of adiabatic switching in MOS energy-recovery circuit. In *Proc. Int. Symp. Low Power Electronics and Design*, pages 236–239, Aug. 2004.

[69] Baohua Wang and Pinaki Mazumder. EM wave coupling noise modeling based on Chebyshev approximation and exact moment formulation. In *Proc. Conf. on Design, Auto. and Test in Europe*, pages 976–981, Germany, Mar. 2005.

[70] Baohua Wang and Pinaki Mazumder. Integrating lumped networks into full wave TLM/FDTD methods using passive discrete circuit models. In *Proc. Int. Symp. Circuits and Systems*, pages 1948–1951, Japan, May 2005.

[71] Baohua Wang and Pinaki Mazumder. Multivariate normal distribution based statistical timing analysis using global projection and local expansion. In *Proc. Int. Conf. on VLSI Design*, pages 380–385, India, Jan. 2005.

[72] Baohua Wang and Pinaki Mazumder. Bounding supply noise induced path delay variation using a relaxation approach. In *Proc. Int. Conf. on VLSI Design*, pages 349–354, India, Jan. 2006.

[73] Baohua Wang and Pinaki Mazumder. A logarithmic full-chip thermal analysis algorithm based on multi-layer Green's function. In *Proc. Conf. on Design, Auto. and Test in Europe*, volume 1, pages 39–44, Germay, Mar. 2006.

[74] Baohua Wang and Pinaki Mazumder. Optimization of circuit trajectories: an auxiliary network approach. In *Proc. Asia and South Pacific Design Automation Conf.*, pages 416–421, Japan, Jan. 2006.

[75] Baohua Wang and Pinaki Mazumder. Accelerated chip-level thermal analysis using multilayer Green's function. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 26(2):325–344, Feb. 2007.

[76] Ting-Yuan Wang and C.C.P. Chen. Thermal-ADI - a linear-time chip-level dynamic thermal-simulation algorithm based on alternating-direction-implicit (ADI) method. *IEEE Trans. VLSI Syst.*, 11(4):691–700, Aug. 2003.

[77] Chenggang Xu, T. Fiez, and K. Mayaram. On the numerical stability of Green's function for substrate coupling in integrated circuits. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 24(4):653–658, Apr. 2005.

[78] Yong Zhan and S.S. Sapatnekar. Fast computation of the temperature distribution in VLSI chips using the discrete cosine transform and table look-up. In *Proc. Asia and South Pacific Design Automation Conf.*, volume 1, pages 87–92, Jan. 2005.

[79] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan. Hotleakage: A temperature-aware model of subthreshold and gate leakage for architects. Technical report, Univ. of Virginia, Dept. of CS, Mar. 2003.

[80] Jinsong Zhao, W.W.M. Dai, S. Kadur, and D.E. Long. Efficient three-dimensional extraction based on static and full-wave layered Green's functions. In *Proc. ACM/IEEE Design Automation Conf.*, pages 224–229, Jun. 1998.