Analytical Placement Techniques for Integrated Circuits: Modeling and Optimization

LIAO, Peiyu

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy
in
Computer Science and Engineering

The Chinese University of Hong Kong
May 2024

Thesis Assessment Committee

Professor XU Qiang (Chair)
Professor YU Bei (Thesis Supervisor)
Professor WONG Martin Ding Fat (Thesis Co-supervisor)
Professor YOUNG Fung Yu (Committee Member)
Professor CHANG Yao-wen (External Examiner)

Abstract

Circuit placement is a crucial process in physical design that encompasses multiple objectives and constraints. With the continuous scaling of technology, placement has become increasingly challenging and critical as it significantly impacts the overall circuit performance. Analytical placement is an essential component of this process due to its high efficiency and reliability. This thesis focuses on advanced analytical placement algorithms that can efficiently produce high-quality solutions in terms of objective modeling and optimization.

Wirelength-driven global placement is the widely adopted strategy in analytical placement, which aims to minimize the total wirelength and thus has the potential to optimize power and delay in subsequent steps of the design flow. Analytical placers formulate the global placement problem as mathematical programming with certain constraints on legality and apply numerical optimization approaches after appropriate problem relaxations. To apply gradient-based numerical optimization approaches, the wirelength model must be differentiable everywhere. This thesis proposes a new differentiable wirelength model using the Moreau envelope to approximate the half-perimeter wirelength (HPWL). The proposed model is superior to conventional models in terms of numerical stability, convexity, and approximation error. Additionally, this thesis proposes algorithms to compute the objective and corresponding gradients, with dedicated theoretical analysis.

Optimizing realistic industrial objectives is of paramount importance to the design closure of integrated circuits, in addition to wirelength. However, most existing placement algorithms based on wirelength optimizations do not directly consider realistic objectives, implying potential unpredictable quality degradation. Timing, one of the most crucial metrics in integrated circuit design, has become a formidable challenge to optimize due to its timeconsuming nature. In this thesis, we propose a timing-driven analytical placement engine with momentum-based net weighting to optimize timing in global placement. The net-based approach is applied because of its scalability to global perturbation of cells in global placement.

With technology scaling nearing its physical limits, the 3D integrated circuit (3D-IC) has emerged as a promising solution for extending Moore's Law. By vertically stacking multiple dies, 3D-IC can achieve higher transistor density and replace long 2D interconnects with shorter inter-die connections, leading to improved circuit performance. In this thesis, we further extend the algorithms from 2D scenarios to 3D. We present a new analytical 3D placement framework with a bistratal wirelength model for face-to-face (F2F) bonded 3D ICs with heterogeneous technology nodes based on the electrostatic-based density model. The proposed framework, which enables GPU acceleration, is capable of efficiently determining node partitioning and locations simultaneously, leveraging the dedicated 3D wirelength model and density model. Moreover, it is a comprehensive expression of analytical placement in terms of modeling and optimization.

摘要

電路佈局是物理設計領域中的一個關鍵過程，涵蓋了多個目標和約束。隨著製程處理規模的持續增大，佈局變得越來越具有挑戰性和關鍵性，因為它對整體電路性能有顯著影響。解析佈局由於其高效性和可靠性，成為這個過程中的一個重要組成部分。本論文關注能在目標建模和優化方面高效生成高質量解的先進解析佈局算法。

基於線長的全局佈局是目前解析佈局中廣泛採用的策略。這種策略旨在最小化電路總線長，因此有潛力在設計流程的後續步驟中優化功耗和延時。解析佈局器將全局佈局問題定義為數學規劃，並在適當的鬆弛後應用數值優化方法進行求解。為了應用基於梯度的數值優化方法，線長模型必須是處處可微的。本論文提出了一種新的可微線長模型，使用 Moreau 包絡對半週長線長（HPWL）進行近似。所提出的模型在數值穩定性、凸性和近似誤差方面都優於傳統模型。此外，本論文還提出了計算目標及相應梯度的算法，並進行了專門的理論分析。

當設計集成電路時，優化實際工業目標除了線長之外至關重要。然而，大多數現有的基於線長優化的佈局算法並未直接考慮實際目標，這意味著可能會出現難以估量的質量損失。在集成電路設計中，時序是最關鍵的指標之一，由於其顯著的耗時性質，優化時序已成為一個艱巨的挑戰。在這篇論文中，我們提出了一個時序驅動的解析佈局引擎，採用基於動量的線網加權方法來優化全局佈局中的時序。採用基於線網加權的方法是因為它對全局佈局中單元的全局干擾具有可擴展性。

隨著規模持續增大以接近其物理極限，三維集成電路（3D-IC）已成為擴展摩爾定律的一個有前途的解決方案。通過垂直堆疊多個芯片，三維集成電路可以實現更高的晶體管密度，並用較短的芯片間連接取代較長的二維互連，從而提高電路性能。在這篇論文中，我們進一步將二維情境之算法擴展到三維。我們提出了全新的三維解析佈局框架，使用基於靜電密度模型的面對面（F2F）鍵合三維集成電路的雙層線長模型。此框架支持 GPU 加速，能夠高效地利用專用的 3D 線長模型和密度模型來同時確定劃分和佈局。此外，它是解析佈局方法在建模和優化方面的全面展現。

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Along with the torrent of computing power revolution, the rise of artificial intelligence, and the ever-shrinking size of advanced technology nodes, the community of **e**lectronic **d**esign **a**utomation (EDA) has witnessed great innovation in the design of **v**ery-**l**arge **s**cale **i**ntegrated (VLSI) circuits. The design process comprises a series of consecutive tasks, each becoming more complicated and challenging for designers to accomplish due to the increasing complexity.

Placement is a crucial step in physical design, where movable instances are mapped into a layout with optimized positions under specific constraints. The fundamental unit to be placed is typically modeled by a rectangle, and millions of rectangles seek their best planar locations in terms of different objectives. Modern placers must be capable of efficiently placing extremely large-scale circuits. Therefore, designing placement algorithms with high performance and quality becomes a great challenge. Common academic and engineering works separate the placement problem into **g**lobal **p**lacement (GP) and **d**etailed **p**lacement (DP). Global placement algorithms are responsible for finding rough planar locations from scratch, ensuring that instances are roughly spread out over the die area. This is followed by a *legalization* step to strictly satisfy legality constraints. Detailed placement algorithms optimize locations incrementally based on a reliable legal initial placement. In contrast to detailed placement, which is usually handled discretely, global placement algorithms relax the complicated constrained optimization problem, making them more flexible to customize. Moreover, since global placement must efficiently find suitable locations for millions of instances from scratch, it encourages and inspires more **analytical** optimization approaches.

The analytical approaches for global placement require differentiable modeling of main objectives (wirelength, timing, congestion, *etc.*) and density constraints. The difficulty of modeling different objectives varies with analytical methods. For instance, realistic objectives are typically intractable to evaluate, model, and optimize. Simultaneously, the non-convex density models are critical to the solution quality of analytical global placement. Hence, mathematical approaches for modeling and optimization are eagerly desired to establish, enhance, and improve the current analytical techniques both in theory and in practical engineering efforts.

In this thesis, we focus on some advanced analytical placement techniques for VLSI circuits. We will cover wirelength modeling in both 2D and 3D scenarios, timing-driven optimization, and 3D placement optimization with heterogeneous technologies. This thesis is organized as follows:

In Chapter 2, the literature review is discussed. It contains an overview of previous works related to the covered topics.

Chapter 3 proposes a differentiable wirelength model based on the theoretical foundations of the proximal operator and Moreau envelope of HPWL. It is worth mentioning that we will rigorously analyze its theoretical properties, and use WATER-FILLING algorithm to efficiently compute the corresponding backward pass. Given the special structure and properties of HPWL, we will show that the general Moreau-Yosida approximation can be directly applied in the numerical optimization of analytical global placement.

In Chapter 4, we will discuss the timing-aware optimization techniques in global placement, an example of optimizing realistic objectives through analytical placement. To better incorporate timing optimization in the gradient-based numerical optimization, we dynamically update the net weights so that critical nets would become tight and thus reduce corresponding net delays, resulting in a better timing quality.

Chapter 5 focuses on the analytical 3D placement with heterogeneous technologies considered. 3D placement is not only an extension of 2D placement but also faces a new challenge of partitioning, as the realistic fabrication only allows two or more dies to vertically stack together. We propose a bistratal wirelength model that fits the die-to-die wirelength objective better. To tackle the differentiability and even convexity issue, we devise the gradient scheme with finite difference approximation, enabling differentiability and GPU parallelism. The proposed framework significantly outperforms existing 3D analytical placers in both solution quality and end-to-end runtime.

CHAPTER 2
LITERATURE REVIEW

2.1 Wirelength Models in Analytical Placement

In order to find suitable locations for circuit components, circuit placement is an essential step in physical design with various objectives and constraints. Different objectives may induce different problem formulations. The widely adopted strategy is to perform wirelength-driven global placement first, which directly minimizes the total wirelength of the entire circuit. Typically, the global placement quality is evaluated by the total routed wirelength after a subsequent routing procedure. However, it could not be easily evaluated during placement, leading to a significant difficulty for analytical approaches.

The current state-of-the-art placers like [Lin+20a] usually apply nonlinear algorithms [Lu+15a] which requires differentiability of the objective function. However, the differentiability of objective functions depends on the detailed formulation of the wirelength models we apply. The Steiner tree wirelength is the most accurate model to the routed wirelength, which is based on the construction of *Rectilinear Steiner Minimum Tree (RSMT)*. However, we do not have a general closed-form representation for Steiner tree wirelength as the well-known RSMT problem is NP-complete [GJ77]. The research discussions on the Steiner's problem can be traced back to the first half of the 19th century [CR41; Mel61; Han66]. It is demonstrated in [Han66] that the search of optimal RSMT of n planar points can be reasonably restricted to the grid points constructed by horizontal and vertical lines of all vertices, known as the *Hanan grid*. The optimal solution for $n \leq 5$ has also been proposed and analyzed [Han66]. Some fascinating research works on solving RSMT problem have discussed the exact [YW72; Hwa76; SW93; GC94] and approximated algorithms [Hwa79; KR92; Gri+94; War98; KMZ03; CW07; LCY21; Kah+24] in detail.

Accelerated algorithms of approximated Steiner tree wirelength [GGL22] may resolve the runtime performance issue to a certain degree, however, it is still apparently inconsistent with the philosophy of analytical approaches, due to its intractable differentiability issue. As a result, modern analytical placers formulate the global placement problem as mathematical programming with certain constraints on legality [Cha+07; KW06; KLM11; VPC07; BSV08; LP08; SSJ08b; Che+08; Hua+17; KM12; Kim+12], using heavily relaxed wirelength models. The common practice suggests the half-perimeter wirelegth (HPWL) as the target objective in analytical global placement, which simply calculates $\max_{1\leq i\leq n} x_i - \min_{1\leq i\leq n} x_i$ of a point set $\{(x_i, y_i)\}_{i=1}^{n}$ as its horizontal part of wirelength. The closed-form representation helps to design analytical algorithms for a parallelized implementation of global

placement.

In order to apply gradient-based numerical optimization approaches, we require the wirelength model to be differentiable everywhere. However, the most popular half-perimeter wirelength model (HPWL) function containing maximum and minimum coordinates is not everywhere differentiable [MHK15]. Therefore, modern placement algorithms heavily rely on differentiable approximations of the HPWL model. There are two main categories, quadratic approximations and nonlinear approximations.

Quadratic models like [BSV08; LP08; SSJ08b; KLM11; VPC07; KM12; Kim+12] approximate every edge cost with the squared length, which gives a strictly convex objective that is very convenient for us to optimize. The general form of quadratic objective has closed-form minimizers which can be found by solving linear systems. However, quadratic models have poor approximation error bounds. To better approximate HPWL with quadratic models, some linearization techniques [SDJ91] have been proposed. They integrate certain net weights inversely proportional to net length so that the weighted squared term can be closer to HPWL. The *Bound2Bound* (B2B) model [SSJ08b; EJ98] decomposes larger nets by selecting boundary pins and connecting them to each internal node.

Commonly, nonlinear analytical placers use differentiable functions, like the *log-sum-exp* [NDS01] model and the *weighted-average* [HCB11; HBC13] model, to approximate HPWL to arbitrary precision by controlling some hyperparameters. The state-of-the-art placers [Cha+07; KW06; Che+08; Hua+17; Lu+15a; Che+18; Lin+19] adopt nonlinear analytical models as they approximate HPWL more accurately, have closed-form gradient representations, and can be naturally optimized by gradient descent.

Some research works have been seeking new ways to optimize HPWL. The *Bivariate-Gradient-Based* (BiG) model [SC19], inspired by [LK07], calculates gradients with respect to coordinates by applying bivariate functions that approximate bivariate maximum or minimum to improve numerical stability and CPU runtime. Subgradient-based approaches [Zhu+15] directly optimize the non-smooth ℓ_1 wirelength objective with the B2B model [SSJ08b; EJ98], using the Polak-Ribière-Polyak conjugate subgradients method [PR69; Zhu+15].

The WA model [HCB11; HBC13] typically has a lower approximation error bound than the LSE model [NDS01], but it still have to face numerical stability and non-convexity issue. In comparison, the BiG model [SC19] has the advantages of numerical stability and cheap computation, but it relies on a bivariate wirelength model that needs to be specified. Non-smooth optimization approaches do not require any differentiable approximations, but they may encounter an issue of slow and poor convergence.

2.2 Timing-Driven Placement

In most modern circuit placement frameworks, the interconnect cost to be optimized is modeled by the total wirelength of all nets, which is estimated by half-perimeter wirelength (HPWL) or other derived approximations, as mentioned in Section 2.1. Besides being only an approximation, total wirelength pays equal attention to all nets instead of focusing on timing-critical nets and paths. This is contrast to timing-driven placement that specifically targets wires on timing-critical paths which often yields immediate circuit performance benefits. Optimizing a more realistic objective in analytical global

placement may heavily affect the final solution quality of the overall design flow in terms of power, performance and area.

Placement can be divided into a global placement stage and a detailed placement stage, and timing optimization can be applied to both stages. The goal of timing-driven global placement is to achieve both roughly good *worst negative slack* (WNS) and *total negative slack* (TNS). Later, timing-driven detailed placement pays more attention to WNS optimization by perturbing the current placement solution locally around critical paths. There are two types of timing-driven placement optimization: net-based and path-based approaches.

In **net-based** approaches, optimization are done on nets within the design. These approaches translate timing analysis feedbacks into changes of net weights and other constraints in order to optimize critical circuit regions. The weights of nets can be computed statically once before placement optimization based on either slack [BY85; Dun+84; Cha+02; Kon02] or sensitivity [HCS00; WTC04; XR05] statistics. The drawback of such approaches is that the timing analysis at earlier placement iterations are unreliable due to frequently-changing cell locations, leading to less effective and representative net weights. Such drawback is remedied by updating net weights dynamically across all placement iterations [BY85; EJ98; RE95; OJ04]. In addition to net weighting, timing analysis results can also be used to limit the maximum net lengths, which are called net constraint-based approaches [Kah+11; Luk91; GVL92; TK91; Raj+03]. The formulation of net constraints varies from particular placers [MHK15].

Contrast to net-based approaches, **path-based** approaches focus on direct optimization of critical timing paths [Cho+05; JK89; SS95; HCC93]. They move cells on selected critical paths to explicitly reduce the delay of these paths. Such path-based objective is often formulated as a mathematical programming problem to optimize, and can usually outperform net-based approaches in terms of solution quality. However, as the number of paths can grow near exponentially with the growth of design size, such path-based approaches are poor in their runtime scalability. Both the net-based and the path-based approaches have strengths and weaknesses regarding different targets such as solution quality and turnaround time, which involve inevitable trade-off during any timing optimization. Generally, we prefer less constrained approaches with knowledge of different placement iterations and more runtime scalability to large designs, especially considering the flexible cell placement formulation during global placement. Recent advances on timing-driven placement have greatly improved the interaction between the placer and timer, so that the entire timing optimization can be processed in a direct way by making the TNS and WNS objective completely differentiable [GL22], despite a possible divergence issue.

2.3 3D Placement

With technology scaling nearing its physical limits, the 3D integrated circuit (3D-IC) has emerged as a promising solution for extending Moore's Law. Vertically stacking multiple dies enables 3D-IC to achieve higher transistor density and replace long 2D interconnects with shorter inter-die connections, leading to improved circuit performance. Leveraging advanced packaging technology, chiplets with heterogeneous technology nodes can be integrated to achieve leading cost-effective performance.

Prominent examples of such technology adoption are Intel's Meteor Lake [Gom+22] and AMD's Zen 4 [Mun+23], which have resulted in significant performance gains and cost savings.

Conventionally, 3D-ICs are fabricated using through-silicon vias (TSVs) with large pitches and parasitics, which may limit the total number of global interconnects to avoid performance degradation [DZX10]. As an alternative approach, monolithic 3D (M3D) integration has been proposed, where tiers are fabricated sequentially and connected using monolithic inter-tier vias (MIVs) [Bat+12; Sam+16; Pan+17; KCL18]. In contrast to TSVs with microscale pitches, MIVs exhibit nanoscale dimensions [Sam+16], allowing for higher integration density with significantly reduced space requirements. Nevertheless, it is still necessary to allocate certain white space on placement regions to accommodate MIVs. Face-to-face (F2F) bonding is another approach that bonds ICs using face sides for both dies [Mor+06; Jun+14; Pan+14; Son+15]. F2F-bonded 3D ICs do not require additional silicon area for 3D connections [Jun+14], eliminating the need to reserve white space for vias and allowing much higher integration density. The silicon-space overhead-free property of F2F-bonded 3D ICs provides significant advantages in numerous applications [KCL18].

Algorithms to solve 3D placement evolve with die-stacking technologies. Conventional discrete solutions handle multiple tiers discretely. T3Place [Con+07] transforms 2D placement solutions into 3D with several folding techniques and local refinement. Early TSV-based research on partition-based approaches [DM01; DCR03; GS07; KAL09] first partitions the netlist to minimize specific targets, *e.g.*, vertical connections, followed by a simultaneous 2D placement on all tiers. The "pseudo-3D" flows utilize optimization techniques of existing 2D engines to work with projected 3D designs. Cascade2D [Cha+16] implements an M3D design using 2D commercial tools with a design-aware partitioning before placement. Recent partitioning-based approaches [Cha+16; Pan+17; KCL18; Par+20] suggest that partitioning first may not sufficiently leverage physical information and thus perform partitioning-last strategies after 2D pre-placement. Shrunk-2D [Pan+14; Pan+17] is a prominent example that performs partitioning according to a 2D pre-placement. Shrunk-2D requires geometry shrinking of standard cells and related interconnects by 50% during its 2D pre-placement for F2F-bounded 3D ICs [Pan+14] or M3D [Pan+17]. Compact-2D [KCL18] adopts placement contraction without geometry shrinking to obtain the 2D pre-placement, followed by a bin-based FM-mincut tier partitioning [FM82]. Pin-3D [Pen+20] proposes pin projection to incorporate inter-die physical information by projecting pins to other dies with fixed locations and transparent geometries, which is first applicable to heterogeneous monolithic 3D ICs. Snap-3D [VI+21] for F2F bonded 3D ICs shrinks the height of standard cell layouts by one half and labels footprint rows top vs. bottom to indicate partitioning. However, the bin-based min-cut partitioning algorithm lacks an understanding of the impact of partitioning on placement quality. TP-GNN [Lu+20], an unsupervised graph-learning-based tier partitioning framework, is proposed to address this drawback for M3D ICs using graph neural networks (GNNs). Considering that discrete algorithms are particularly sensitive to partitioning [Mur+22] and can potentially lead to performance degradation, analytical 3D placement is considered to be more promising to produce solutions with higher quality.

Analytical 3D solutions relax discrete tier partitioning and solve continuous 3D optimization problems accordingly. Typical analytical approaches include quadratic programming [KOB03; Hen+06], nonlinear programming [Tan00], and force-directed methods [GS03]. In addition, NTUPlace3-

3D [HCB11; HBC13] performs 3D analytical placement based on a bell-shaped [KW04] smooth density considering TSV insertion, and mPL6-3D [LSC13] utilizes a Huber-based local smoothing technique working with a Helmholtz-based global smoothing approach. Based on mPL6-3D [LSC13], ART-3D [Mur+22] improves placement quality using reinforcement learning-based parameter tuning. The state-of-the-art analytical placement is the ePlace family [Lu+13; Lu+15a; Lu+15b; Lu+16] where the density constraint is modeled by an electrostatic field. Lu et al. proposes a general 3D eDensity model in ePlace-3D [Lu+16] achieving analytically global smoothness along all dimensions in 3D domain. Remarkably, the ePlace family has achieved substantial success in wirelength-driven analytical placement, and their adoption of fast Fourier transform (FFT) for solving the 3D numerical solution has inspired quality enhancement [Che+18] and GPU-accelerated ultra-fast implementations [Lin+19; Liu+22].

Unfortunately, the aforementioned previous works have difficulties in considering heterogeneous technologies and specific utilization constraints, and thus lead to poor performance when applied to heterogeneous 3D placement problems. Recently, Chen et al. [Che+23a] have proposed a 3D analytical placement algorithm to optimize wirelength considering F2F-bonded 3D ICs with multiple manufacturing technologies. They devise a multi-technologies weighted-average (MTWA) wirelength model using sigmoid-based functions for pin offset transition, and establish their framework based on ePlace-3D [Lu+16]. A 2D analytical placement, considering the accurate wirelength, is employed after the 3D global placement to further refine the solution. The aforementioned works, including [Che+23a], adopt the 3D net bounding box as the wirelength model which is not capable of capturing enough information of the impact of partition on wirelength.

CHAPTER 3
WIRELENGTH MODEL

In this chapter, we propose a new differentiable wirelength model using the *Moreau envelope* [Mor65] to approximate HPWL. The proposed model is superior to previous models in numerical stability, convexity, and approximation error. We will also propose algorithms to compute the objective and gradients. To make such computation possible, we build upon the proximal mapping of the HPWL model. The major contributions are summarized as follows.

- We derive the explicit representations of proximal mapping of the non-smooth HPWL function and propose a *water-filling* algorithm similar to that in communication systems [Sha48; Wyn66; YC01; Yeu08] to solve the proximal mapping and Moreau envelope problem.
- We provide theoretical and experimental analysis to compare the proposed wirelength model and the widely-used weighted-average model [HCB11; HBC13] and verify its feasibility.
- Experimental results show that the proposed wirelength model can achieve up to 5.4% HPWL improvements and over 1% on average for the ISPD2006 contest benchmarks [Nam06]. In addition, we also achieve up to 3.0% HPWL improvements and over 1.5% on average on the recent ISPD2019 contest benchmarks [Liu+19].

The rest of the chapter is organized as follows. Section 3.1 introduces some preliminaries, including foundations of nonlinear placement, wirelength models, and the Moreau envelope. Section 3.2 presents the algorithms of the proposed methods. Then, Section 3.3 gives some theoretical properties of our wirelength model. Section 3.4 presents experimental results and some related analysis on the adopted benchmarks, followed by the conclusion in Section 3.5.

3.1 Preliminaries

In this section, we review the foundations of analytical global placement and then introduce the general form of the Moreau envelope applied to non-smooth functions. Besides, we will briefly compare properties of the weighted-average [HCB11] and the Moreau envelope.

3.1.1 Analytical Global Placement

Circuit placement usually consists of three steps: global placement, legalization, and detailed placement. At the global placement stage, we aim to find good cell locations with small total wirelength

such that the overlaps are controlled at a low level. The overlap penalty is a density term $D(\boldsymbol{x}, \boldsymbol{y})$ modeled in an electrostatic system [Lu+15a], where variables $\boldsymbol{x}$ and $\boldsymbol{y}$ are the horizontal and vertical cell locations, respectively. A typical nonlinear global placement problem is formulated as

$$\min_{\boldsymbol{x},\boldsymbol{y}} \sum_{e \in E} W_e(\boldsymbol{x}, \boldsymbol{y}) + \lambda D(\boldsymbol{x}, \boldsymbol{y}), \tag{3.1}$$

where E is the net set, $W_e(\cdot)$ is the net wirelength function of net $e \in E$, $D(\boldsymbol{x}, \boldsymbol{y})$ models the cell density penalty, and λ is the corresponding density weight in the objective.

The wirelength model is the detailed representation of the total-wirelength function $W_e(\boldsymbol{x}, \boldsymbol{y})$. For example, a *rectilinear minimum Steiner tree* (RMST) wirelength is accurate to approximate the routed wirelength but does not have an explicit closed-from representaion. For simplicity, we also use $\boldsymbol{x}, \boldsymbol{y}$ to represent corresponding pin coordinates ignoring pin offsets in this Section 3.1. Then, the most widely-adopted HPWL model is defined as $W(\boldsymbol{x}, \boldsymbol{y}) = \sum_{e \in E} W_e(\boldsymbol{x}, \boldsymbol{y})$, where the net HPWL of $e \in E$ is

$$W_e(\boldsymbol{x}, \boldsymbol{y}) = \max_{i \in e} x_i - \min_{i \in e} x_i + \max_{i \in e} y_i - \min_{i \in e} y_i. \tag{3.2}$$

The HPWL function defined in Equation (3.2) is convex and continuous but not everywhere differentiable. Fortunately, it has a simple and clean formulation. Therefore, many approximation models have been proposed. Here we focus on the *nonlinear* models.

3.1.2 Nonlinear Differentiable Models

Since most analytical placement algorithms expect to optimize HPWL, we are going to discuss nonlinear differentiable HPWL-based wirelength models. Consider a net containing n pins with horizontal coordinates $\boldsymbol{x} \in \mathbb{R}^n$. There are two widely-used exponential approximations, the *log-sum-exp* (LSE) model [NDS01] and the *weighted-average* (WA) model [HCB11; HBC13],

$$\begin{aligned}
W_e(\boldsymbol{x}) &\approx W^{\gamma}_{e,\mathrm{LSE}}(\boldsymbol{x}) = \gamma \ln \sum_{i=1}^{n} \exp\left(\frac{x_i}{\gamma}\right) + \gamma \ln \sum_{i=1}^{n} \exp\left(-\frac{x_i}{\gamma}\right), \\
W_e(\boldsymbol{x}) &\approx W^{\gamma}_{e,\mathrm{WA}}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} x_i \exp\left(\frac{x_i}{\gamma}\right)}{\sum_{i=1}^{n} \exp\left(\frac{x_i}{\gamma}\right)} - \frac{\sum_{i=1}^{n} x_i \exp\left(-\frac{x_i}{\gamma}\right)}{\sum_{i=1}^{n} \exp\left(-\frac{x_i}{\gamma}\right)},
\end{aligned} \tag{3.3}$$

where γ is a hyperparameter to control the tradeoff between the accuracy and differentiability. When $\gamma \to 0^+$, the approximation will be arbitrarily close to the real HPWL function. Usually at the earlier stages of global placement, γ is set to a large value so that the objective can be very smooth, and decreases as the number of iterations increases.

3.1.3 Moreau Envelope

The general Moreau envelope [Mor65] is defined for functions satisfying specific constraints on a real *Hilbert space*. In our placement applications, we only consider lower bounded *closed convex* functions $h(\boldsymbol{x})$ defined in $\mathbb{R}^n$ to simplify the notations. For any $t > 0$, let $H(\boldsymbol{u}, \boldsymbol{x}) = h(\boldsymbol{u}) + \frac{1}{2t}\|\boldsymbol{u} - \boldsymbol{x}\|_2^2$, then

the Moreau envelope h^t and the proximal operator are defined by

$$h^t(\boldsymbol{x}) = \min_{\boldsymbol{u}\in\mathbb{R}^n} H(\boldsymbol{u},\boldsymbol{x}),\ \operatorname{prox}_{th}(\boldsymbol{x}) = \operatorname*{argmin}_{\boldsymbol{u}\in\mathbb{R}^n} H(\boldsymbol{u},\boldsymbol{x}). \tag{3.4}$$

Hence, the objective value of the Moreau envelope can be naturally obtained as long as the proximal mapping of $h(\cdot)$ is inexpensive to calculate. Note that the proximal mapping may or may not have a closed-form representation, and it may or may not be easy to calculate, up to the form of function $h(\cdot)$. Also, we do not require function h to be smooth.

Under the constraints of $h(\cdot)$ specified above, the Moreau envelope h^t is a natural differentiable approximation of h, and its gradient is globally Lipschitz continuous. More specifically, the *envelope theorem* [Afr71] states that

$$\nabla h^t(\boldsymbol{x}) = \frac{1}{t}\left(x - \operatorname{prox}_{th}(\boldsymbol{x})\right). \tag{3.5}$$

Since $\lim_{t\to 0^+} h^t(\boldsymbol{x}) = h(\boldsymbol{x})$ converges pointwise, t is considered to be the approximation precision, similar to γ in LSE [NDS01] and WA [HCB11].

The interesting point is that, in global placement, we are inspired to consider the net HPWL function W_e, which has a clean and straightforward formulation. In this chapter, we propose an algorithm to compute its Moreau envelope W_e^t and use $W_e^t + t$ as the approximated wirelength model so that the entire objective is everywhere differentiable. In Section 3.2, we are going to show the representation of $\operatorname{prox}_{tW_e}$ for the HPWL function W_e and the *water-filling* algorithm to solve it inexpensively.

Additionally, if the learning rate in gradient descent is equal to the smoothing parameter t in the Moreau envelope, *i.e.*, the update rule is $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - t\nabla W^t(\boldsymbol{x})$, we can interpret it as a *proximal point method*, as

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - t\nabla W^t(\boldsymbol{x}^{(k)}) = \operatorname{prox}_{tW^t}(\boldsymbol{x}^{(k)}). \tag{3.6}$$

However, we also have to consider the effect of the density term, and may use a different learning rate determined by Nesterov accelerated gradient method. The real application is much more complicated.

3.1.4 Comparison with WA

In this subsection, we will briefly compare WA [HCB11] and the Moreau envelope model. Equation (3.3) uses exponential terms to assign weights to each coordinate. A larger coordinate x_i has a large weight in the smooth maximum, as $\exp(\frac{x_i}{\gamma})$ would occupy a larger proportion.

Numerical Stability: The exponential terms are sensitive to the coordinate. Note that the Δx will usually be hundreds or even over thousands in actual placement, so the γ should not be very small, as a small γ will be likely to result in a numerical overflow. This phenomenon is also stated in [SC19]. Nearly all exponential models like the LSE [NDS01] and WA [HCB11] models have to face such an issue.

Non-Convexity: There is no theoretical or experimental guarantee of convexity of the WA [HCB11] model. Let us take a 3-pin net as an example and conduct a toy experiment on its horizontal pin coordinate $\boldsymbol{x} \in \mathbb{R}^3$. Assume that we fix the $\Delta x = 100$, *i.e.*, let $\boldsymbol{x} = (0, x, 100)^\top$ without loss of

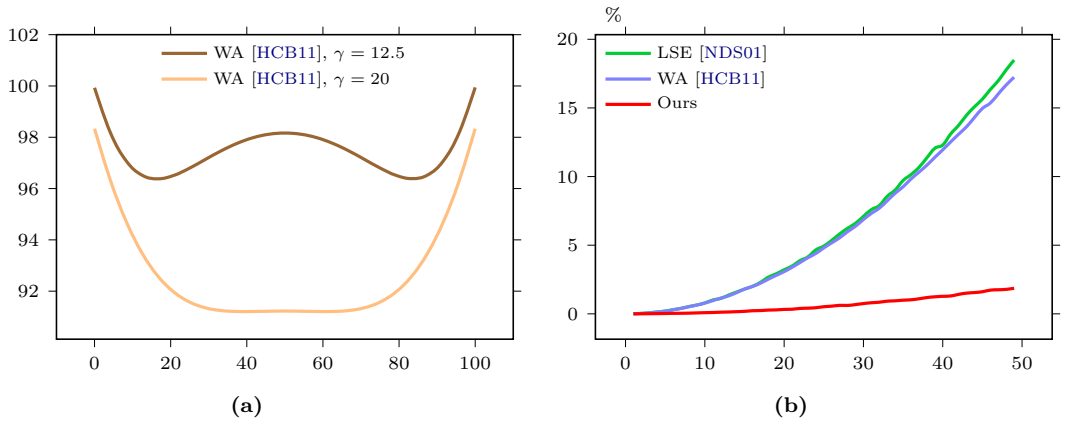


Figure 3.1: *(a) The non-convexity of WA [HCB11] on a simple 3-pin net to approximate* $\Delta x = \max\{x_{\min}, x, x_{\max}\} - \min\{x_{\min}, x, x_{\max}\}$. *(b) The average approximation error of different models LSE [NDS01], WA [HCB11] and Moreau Envelope against the smoothing parameter* γ *or* t *for 4-pin nets, under fixed* $\Delta x = 200$.

generality where $x \in [0, 100]$. We plot the function curve of $W^{\gamma}_{e,\mathrm{WA}}(\{0, x, 100\})$ for some γ values in Figure 3.1(a). As shown in the figure, even for the 3-pin net, the WA model can be non-convex. For high-degree nets, it will get more complicated. Although the optimization process of real placement problems is far more obscure than imagined, a convex wirelength model is usually preferred. As a comparison, the Moreau envelope approximation is always convex [Mor65] for HPWL.

Approximation Error: The net wirelength models can be high-dimensional and thus extremely difficult to analyze. We conduct a toy experiment on the smoothing parameters γ (for LSE [NDS01] and WA [HCB11]) and t. Given a fixed range $\Delta x = x_{\max} - x_{\min} = 200$, we randomly generate horizontal coordinates $\boldsymbol{x} \in \mathbb{R}^4$ for different smoothing parameters 3000 times, calculate the approximated Δx using different wirelength models, take the average, and draw curves in Figure 3.1(b).

Although the smoothing parameters here have different mathematical meanings in different wirelength models, the advantage of our proposed model on the approximation error is still well-illustrated in Figure 3.1(b). It is worth mentioning that a lower approximation error does not always imply better solution quality after placement, as the optimization procedure is so complicated that it can be easily influenced by various hidden factors. In addition to the approximation error, the gradient properties should also be carefully considered, which will be discussed in Section 3.3.

3.2 Algorithm

We adopt the Moreau envelope model as a differentiable approximation of net HPWL, which requires the proximal mapping of HPWL. In this chapter, we use the *rectified linear unit* (ReLU) activation function $\eta^+ = \max\{\eta, 0\}$ to represent the *positive part* of $\eta \in \mathbb{R}$.

Algorithm 1 GRADIENT ALGORITHM

Require: The horizontal (or vertical) pin coordinates $\boldsymbol{x} \in \mathbb{R}^n$, the smoothing parameter $t > 0$.

1: Sort pin coordinates $\boldsymbol{x}$ such that $x_1 \leq \cdots \leq x_n$;
2: Apply the WATER-FILLING ALGORITHM described in Algorithm 2 to solve equations

$$\sum_{i=1}^{n}(x_i - \tau_2)^+ = \sum_{i=1}^{n}(\tau_1 - x_i)^+ = t$$

to obtain water-filling parameters τ_1 and τ_2;
3: **if** $\tau_1 > \tau_2$ **then**
4: Calculate average pin coordinate $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$;
5: Assign $\tau_1 \leftarrow \bar{x}$;
6: Assign $\tau_2 \leftarrow \bar{x}$;
7: Given τ_1 and τ_2, calculate gradient $\boldsymbol{g} = \nabla W_e^t(\boldsymbol{x})$ according to Corollary 1;
8: **return** the required gradient $\boldsymbol{g}$;

3.2.1 The Gradient of Moreau Envelope

Without loss of generality, consider a single net e connecting n pins $p_1, \cdots, p_n$ with horizontal coordinates $\boldsymbol{x} \in \mathbb{R}^n$. The horizontal part of HPWL function is represented by

$$W_e(\boldsymbol{x}) = \max_{1 \leq i \leq n} x_i - \min_{1 \leq i \leq n} x_i. \tag{3.7}$$

The horizontal and vertical directions are symmetric, so we only focus on the horizontal direction for analysis.

We give an overview of the gradient computation of the Moreau envelope model here. At each placement iteration, we are required to calculate the gradient of the wirelength model w.r.t. the pin coordinates $\boldsymbol{x}$. The overall algorithm to calculate gradient $\boldsymbol{g}$ of the Moreau envelope $W_e^t(\boldsymbol{x})$ under the smoothing parameter t is decribed in Algorithm 1. The following Theorem 1 states the representation of the proximal mapping $\text{prox}_{tW_e}(\boldsymbol{x})$. Then Corollary 1 extends Theorem 1 and tells us how to calculate gradient $\boldsymbol{g}$ given parameters τ_1 and τ_2. The detailed algorithm of calculating parameters τ_1 and τ_2 is described in Algorithm 2.

Theorem 1. *The proximal mapping of $t \cdot W_e$ for a positive real number $t > 0$ is given by* $\text{prox}_{tW_e}(\boldsymbol{x}) = \boldsymbol{u}^*$ *where*

$$u_i^* = \begin{cases} \tau_2, & \text{if } x_i > \tau_2 \\ x_i, & \text{if } \tau_1 \leq x_i \leq \tau_2 \\ \tau_1, & \text{otherwise} \end{cases} \tag{3.8}$$

is defined for any $i = 1, \cdots, n$, such that

$$\sum_{i=1}^{n}(x_i - \tau_2)^+ = \sum_{i=1}^{n}(\tau_1 - x_i)^+ = t, \tag{3.9}$$

if the solution τ_1, τ_2 to Equation (3.9) satisfy $\tau_1 \leq \tau_2$, otherwise $\boldsymbol{u}^*$ is determined by the average coordinate $u_i^* = \frac{1}{n}\boldsymbol{e}^\top \boldsymbol{x}$ for any index $i = 1, \cdots, n$.

Theorem 1 discusses the explicit representation of the horizontal part of HPWL w.r.t. pin locations.

This is the core of the approximation. Note that it is NOT a complete closed-form representation, as we still need to solve Equation (3.9) for τ_1 and τ_2. The *water-filling* algorithm to solve these equations will be discussed in Section 3.2.2. Before that, we state the following corollary.

Corollary 1. *Consider the horizontal part of the net HPWL* $W_e(\boldsymbol{x}) = \max_{1\le i\le n} x_i - \min_{1\le i\le n} x_i$, *its Moreau envelope function* W_e^t *is everywhere differentiable. The gradient is* $\boldsymbol{g} = \nabla W_e^t(\boldsymbol{x})$ *where*

$$g_i = \begin{cases} \frac{1}{t}(x_i - \tau_2), & \text{if } x_i > \tau_2; \\ 0, & \text{if } \tau_1 \le x_i \le \tau_2; \\ \frac{1}{t}(x_i - \tau_1), & \text{otherwise} \end{cases} \tag{3.10}$$

is defined for any $i = 1, \cdots, n$, *such that*

$$\sum_{i=1}^{n}(x_i - \tau_2)^+ = \sum_{i=1}^{n}(\tau_1 - x_i)^+ = t, \tag{3.11}$$

if the solution τ_1, τ_2 *to Equation* (3.11) *satisfy* $\tau_1 \le \tau_2$, *otherwise* $\boldsymbol{g} = \nabla W_e^t(\boldsymbol{x})$ *is determined by the average coordinate:* $g_i = \frac{1}{t}x_i - \frac{1}{tn}\sum_{k=1}^{n} x_k$ *for any index* $i = 1, \cdots, n$.

This corollary can be directly verified by the Moreau envelope gradient $\nabla W_e^t(\boldsymbol{x}) = \frac{1}{t}(\boldsymbol{x} - \text{prox}_{tW_e^t}(\boldsymbol{x}))$ according to Equation (3.5), where the proximal mapping $\text{prox}_{tW_e^t}(\boldsymbol{x})$ is given in Theorem 1.

3.2.2 Water-filling Algorithm

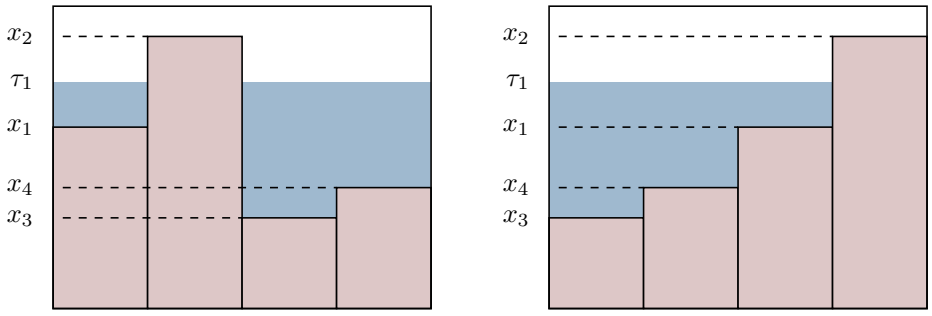
Equation (3.8) and Equation (3.9) give an explicit representation of the proximal mapping, where the values of τ_1 and τ_2 are not represented in closed-form. Therefore, we must solve equations in Equation (3.9) first to obtain the exact value of $\text{prox}_{tW_e}(\boldsymbol{x})$.

We take τ_1 as an example, *i.e.*, we are going to solve the equation $\sum_{i=1}^{n}(\tau_1 - x_i)^+ = t$. Suppose we have a 4-pin net with pin coordinate x_1, x_2, x_3, x_4, illustrated in Figure 3.2(a). We use a bar graph to illustrate the distribution, with a symbolic horizontal axis. Assume that each bar has width 1 and height x_i for $i = 1, 2, 3, 4$, then we are going to find a value τ_1 such that the blue region has a total area t.

The process for obtaining the exact value of τ_1, similar to that in the communication engineering, is called *water-filling*. One can simply imagine that we have an amount t of water in total, and want to pour it into a reservoir with an uneven bottom [Yeu08]. The final level τ_1 the water rise to is the target we desire.

To solve the water-filling, we first sort the pin coordinates. The statistics of sorted pin coordinates is illustrated in Figure 3.2(b). On the surface, there seems to be no difference compared to Figure 3.2(a). However, once we have the sorted value of pin coordinates, or sorted indices, the process can be solved in $O(n)$ time where n stands for the number of pins connected by this net. The detailed algorithm to solve the equation $\sum_{i=1}^{n}(\tau_1 - x_i)^+ = t$ is described in Algorithm 2.

The process of water-filling in Algorithm 2 is very intuitive. Assume that we have sorted the pin coordinates such that $x_1 \le x_2 \le \cdots \le x_n$. Since we have n pins in this net, we have $n - 1$ bottoms $x_1, \cdots, x_{n-1}$ to fill in before we make the bottoms of this "reservoir" even. Therefore, we create a



(a) *Pin coordinates statistics* **(b)** *Sorted pin coordinates*

Figure 3.2: *The illustration of water-filling to solve τ_1 in Equation* (3.9).

Algorithm 2 WATER-FILLING ALGORITHM

Require: The sorted coordinates $\boldsymbol{x} \in \mathbb{R}^n$ such that $x_1 \leq x_2 \leq \cdots x_n$.

Initialize water trial $q = 0$, the bottom index $k = 1$;
while $k < n$ **do**
    Calculate the amount of water we need to fill in one bottom $q' \leftarrow x_{k+1} - x_k$;
    Accumulate water trial $q \leftarrow q + kq'$ because we need to fill in k bottoms;
    if $q > t$ **then**
        Break the loop;
    Proceed to the next bottom $k \leftarrow k + 1$;
if $q < t$ **then**
    Calculate $\tau_1 \leftarrow x_n + \frac{1}{n}(t - q)$;
else
    Calculate $\tau_1 \leftarrow x_{k+1} - \frac{1}{k}(q - t)$;
return the required level value τ_1;

trial and fill in each bottom one by one to check whether the current total trial is larger than the target amount t of water. More specifically, we will find k such that $x_k \leq \tau_1 < x_{k+1}$ in Algorithm 2.

Another interpretation of this algorithm is the *Abel transformation* for discrete sequences [Abe26]:

$$\sum_{i=1}^{k}(\tau_1 - x_i) = k(\tau_1 - x_k) + \sum_{i=1}^{k-1} i(x_{i+1} - x_i), \tag{3.12}$$

where k is taken such that $x_k \leq \tau_1 < x_{k+1}$, or $k = n$ if $\tau_1 \geq x_n$. Clearly, we have $k(\tau_1 - x_k) \leq k(x_{k+1} - x_k)$ if $\tau_1 < x_n$, indicating

$$\sum_{i=1}^{k-1} i(x_{i+1} - x_i) \leq \sum_{i=1}^{n}(\tau_1 - x_i)^+ < \sum_{i=1}^{k} i(x_{i+1} - x_i), \tag{3.13}$$

when $\boldsymbol{x}$ are sorted. Therefore, given $t > 0$, we are supposed to find the index k such that either

$$\sum_{i=1}^{k-1} i(x_{i+1} - x_i) \leq t < \sum_{i=1}^{k} i(x_{i+1} - x_i) \tag{3.14}$$

or $t \geq \sum_{i=1}^{n-1} i(x_{i+1} - x_i)$ is satisfied ($k = n$ for this case). After sorting the pin coordinates, Equation (3.14) can be solved in $O(n)$ time with a single traversal. The part of solving τ_2 given t is similar.

Generally speaking, Algorithm 2 can be super fast as it visits n pins at most. The bottleneck is the sorting before each water-filling. Since we may have millions of pins in a design, an efficient sorting algorithm is critical.

3.2.3 Nonlinear Optimization

The update schemes of the precision γ in WA [HCB11] and t in ours are also essential to the solution quality. The ePlace [Lu+15a] algorithm and related placers [Lin+19] use a form of $\gamma(\phi) = \gamma_0 \left(w_{\text{bin}}^{(x)} + w_{\text{bin}}^{(y)}\right) \cdot 10^{k\phi+b}$ to update the parameter γ according to overflow ϕ. Higher overflow requires a large γ to sacrifice precision for better differentiability. To better adapt to the proposed wirelength model, we use the following tangent-based update scheme

$$t(\phi) = \frac{t_0}{2} \left(w_{\text{bin}}^{(x)} + w_{\text{bin}}^{(y)}\right) \tan\left(\frac{\pi}{2}\phi - \delta\right), \tag{3.15}$$

where t_0 is the initial value, ϕ is the density overflow, $w_{\text{bin}}^{(x)}$ and $w_{\text{bin}}^{(y)}$ stand for the horizontal and vertical bin size, respectively, and δ is a small positive number to avoid numerical overflow. For example, a configuration that $\delta = 10^{-4}$ and $t_0 = 4$ will normally give a good result for most cases.

We follow a moderate scheme similar to that of DreamPlace 3.0 [Gu+20] and elfPlace [LLP19] to update density weight λ in Equation (3.1) iteratively:

$$\begin{aligned} \lambda_{k+1} &= \lambda_k + \alpha_k, \\ \alpha_k &= \left(\alpha_{\text{H}} - \frac{\alpha_{\text{H}} - \alpha_{\text{L}}}{1 + \ln(1 + \frac{\beta D_k}{D_0})}\right) \alpha_{k-1}, \end{aligned} \tag{3.16}$$

where D_k is the density at the k-th iteration. The parameters $\alpha_{\text{H}} \geq \alpha_{\text{L}} > 1$ depict how fast the density weight increases. The parameter pair $(\alpha_{\text{L}}, \alpha_{\text{H}})$ is set to $(1.01, 1.02)$ by default. The hyper-parameter β is set to 2000 in our experiments. Note that we do not use the quadratic penalty in our formulation, so β is simply a tunable parameter to adjust the step size α_k of density weight without any physical meaning. Here α_0 is $(\alpha_{\text{L}} - 1)\lambda_0$, and λ_0 is determined according to ePlace [Lu+15a].

3.3 Wirelength Model Analysis

The HPWL model has specific properties. By controlling the precision, the approximation models should also preserve similar properties.

3.3.1 Approximation Bound

In this subsection, we give some theoretical results of the approximation bound of the Moreau envelope wirelength model. Besides, we also make comparison with the weighted-average model [HCB11].

Theorem 2. *Consider the net HPWL $W_e(\boldsymbol{x}) = \max_{1\le i\le n} x_i - \min_{1\le i\le n} x_i$ of net e. The Moreau envelope function W_e^t increases pointwise to W_e as t decreases to 0.*

Proof. The Moreau envelope is upper bounded by

$$W_e^t(\boldsymbol{x}) = \min_{\boldsymbol{u}\in\mathbb{R}^n} \left\{ W_e(\boldsymbol{u}) + \frac{1}{2t}\|\boldsymbol{u}-\boldsymbol{x}\|_2^2 \right\} \le W_e(\boldsymbol{x}), \tag{3.17}$$

because function $W_e(\boldsymbol{u})+\frac{1}{2t}\|\boldsymbol{u}-\boldsymbol{x}\|_2^2$ takes value $W_e(\boldsymbol{x})$ when $\boldsymbol{u}$ is $\boldsymbol{x}$. Additionally, we have $W_e^{t_1}(\boldsymbol{x}) \le W_e^{t_2}(\boldsymbol{x})$ if $0 < t_2 < t_1$. By the *monotone convergence theorem*, we know that the limit $\lim_{t\to 0^+} W_e^t(\boldsymbol{x})$ exists and it is non-positive. Now it only remains to show that $\lim_{t\to 0^+} W_e^t(\boldsymbol{x}) = W_e(\boldsymbol{x})$.

Before that, we first show the Lipschitz property of W_e. Taking arbitray two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, we have

$$\begin{aligned} |W_e(\boldsymbol{x}) - W_e(\boldsymbol{y})| &= \left| \max_i x_i - \max_i y_i - \min_i x_i + \min_i y_i \right| \\ &\le \left| \max_i x_i - \max_i y_i \right| + \left| \min_i x_i - \min_i y_i \right| \\ &\overset{(*)}{\le} 2\|\boldsymbol{x}-\boldsymbol{y}\|_\infty \le 2\|\boldsymbol{x}-\boldsymbol{y}\|_2. \end{aligned}$$

The inequality marked $(*)$ can be clearly explained by the symmetry property of maximum and minimum functions. Now, we have shown that W_e is 2-Lipschitz over $\mathbb{R}^n$ with respect to norm $\|\cdot\|_2$, although the constant $L = 2$ is a large estimation actually.

When $\boldsymbol{u}$ takes $\mathrm{prox}_{tW_e}(\boldsymbol{x})$, the minimum is attained. Besides, we know that $\frac{1}{t}(\boldsymbol{x} - \mathrm{prox}_{tW_e}(\boldsymbol{x})) \in \partial W_e(\mathrm{prox}_{tW_e}(\boldsymbol{x}))$ is a subgradient. Therefore, taking any subgradient $\boldsymbol{g} \in W_e(\boldsymbol{x})$, we have

$$\begin{aligned} & W_e^t(\boldsymbol{x}) - W_e(\boldsymbol{x}) \\ &= W_e(\mathrm{prox}_{tW_e}(\boldsymbol{x})) - W_e(\boldsymbol{x}) + \frac{1}{2t}\left\|W_e(\mathrm{prox}_{tW_e}(\boldsymbol{x})) - \boldsymbol{x}\right\|_2^2 \\ &\overset{(a)}{\ge} -\boldsymbol{g}^\top(\boldsymbol{x} - \mathrm{prox}_{tW_e}(\boldsymbol{x})) + \frac{1}{2t}\left\|W_e(\mathrm{prox}_{tW_e}(\boldsymbol{x})) - \boldsymbol{x}\right\|_2^2 \\ &\overset{(b)}{\ge} -\|\boldsymbol{g}\|_2\|\boldsymbol{x} - \mathrm{prox}_{tW_e}(\boldsymbol{x})\|_2 \overset{(c)}{\ge} -4t. \end{aligned}$$

The inequality marked (a) comes from the fundamental property of subgradients. The inequality marked (b) is true according to the Cauchy-Schwarz inequality. The inequality marked (c) is from Lemma 2.6 in [SS12].

From inequalities $-4t \le W_e^t(\boldsymbol{x}) - W_e(\boldsymbol{x}) \le 0$, we have the desired point-wise convergence result $\lim_{t\to 0^+} W_e^t(\boldsymbol{x}) = W_e(\boldsymbol{x})$. □

Theorem 2 gives a rough bound that describes how well the Moreau envelope function generally approximates the original one. Similar to the log-sum-exp [NDS01] and the weighted-average [HCB11; HBC13] models, the Moreau envelope function we use here also can approximate HPWL in arbitrary precision. The following theorem gives a better bound.

Theorem 3. *Consider the net HPWL $W_e(\boldsymbol{x}) = \max_{1\le i\le n} x_i - \min_{1\le i\le n} x_i$ of net e. We have*

$$-t \le W_e^t(\boldsymbol{x}) - W_e(\boldsymbol{x}) \le 0, \tag{3.18}$$

for any positive parameter $t > 0$.

Proof. Without loss of generality, let $x_1 \le x_2 \le \cdots \le x_k < \tau_1 \le \cdots \le \tau_2 < x_{n-r+1} \le \cdots \le x_n$. Then we have the water-filling equation $\sum_{i=1}^{k}(\tau_1 - x_i) = t$, since we have

$$W_e^t(\boldsymbol{x}) = \tau_2 - \tau_1 + \frac{1}{2t}\sum_{i=1}^{k}(\tau_1 - x_i)^2 + \frac{1}{2t}\sum_{i=1}^{r}(x_{n-i+1} - \tau_2)^2, \tag{3.19}$$

we consider the minimum approximation part only, and the maximum approximation part is similar. Let $a_i = \tau_1 - x_i$ for $i = 1, \cdots, k$ and $a_{\max} = \max_{1\le i\le n} a_i = a_1$. If $k \ge 2$, we obtain

$$\begin{aligned}
&\frac{1}{2t}\sum_{i=1}^{k}(\tau_1 - x_i)^2 + \min_{1\le i\le n} x_i = \frac{1}{2t}\sum_{i=1}^{k} a_i^2 - a_{\max} + \tau_1 \\
&= \frac{\sum_{i=1}^{k} a_i^2}{2t} - a_{\max} + \tau_1 = \frac{a_1^2 + \sum_{i=2}^{k} a_i^2}{2t} - a_1 + \tau_1 \\
&\overset{(*)}{\ge} \frac{a_1^2 + \frac{1}{k-1}\left(\sum_{i=2}^{k} a_i\right)^2}{2t} - a_1 + \tau_1 = \frac{ka_1^2 - 2kta_1 + t^2}{2(k-1)t} + \tau_1,
\end{aligned} \tag{3.20}$$

under constraint $ka_1 \ge t$, where the inequality marked $(*)$ is the Cauchy-Schwarz inequality. The rightmost representation is quadratic with respect to a_1, so we have

$$\frac{1}{2t}\sum_{i=1}^{k}(\tau_1 - x_i)^2 + \min_{1\le i\le n} x_i \ge -\frac{t}{2} + \tau_1. \tag{3.21}$$

If $k = 1$, the left-hand side of Equation (3.20) is directly $\frac{a_1^2}{2t} - a_1 + \tau_1 \ge -\frac{t}{2} + \tau_1$. The equality of Equation (3.21) holds when $k = 1$ and $a_1 = \tau_1 - x_{\min} = t$.

Similarly, we obtain another part

$$\frac{1}{2t}\sum_{i=1}^{r}(x_{n-i+1} - \tau_2)^2 - \max_{1\le i\le n} x_i \ge -\frac{t}{2} - \tau_2. \tag{3.22}$$

Combining Equation (3.21) and Equation (3.22), we obtain the desired result $-t \le W_e^t(\boldsymbol{x}) - W_e(\boldsymbol{x}) \le 0$. The equality of the left-hand side holds when $\tau_1 - x_{\min} = x_{\max} - \tau_2 = t$ and $k = r = 1$. □

Obviously, Theorem 3 provides a better bound with a smaller constant. This bound is clearly attainable when there is only one maximum coordinate and one minimum coordinate, and the parameter t is small enough such that only these two related pins have gradients.

Theorem 4. *Assume that there are $n_{\max} \ge 1$ pins sharing the maximum coordinates, and $n_{\min} \ge 1$ pins sharing the minimum coordinates. Then, we have*

$$-\frac{t}{2}\left(\frac{1}{n_{\max}} + \frac{1}{n_{\min}}\right) \le W_e^t(\boldsymbol{x}) - W_e(\boldsymbol{x}) \le 0, \tag{3.23}$$

for any positive parameter $t > 0$.

Proof. Similar to the proof to Theorem 3, we have

$$\begin{aligned}
&\frac{1}{2t}\sum_{i=1}^{k}(\tau_1 - x_i)^2 + \min_{1\le i\le n} x_i = \frac{1}{2t}\sum_{i=1}^{k} a_i^2 - a_{\max} + \tau_1 \\
\ge\ & \frac{n_{\min}}{2t}a_1^2 + \frac{1}{2(k-n_{\min})t}\left(t - n_{\min}a_1\right)^2 - a_1 + \tau_1 \\
=\ & \frac{n_{\min}ka_1^2 - 2kta_1 + t^2}{2(k-n_{\min})t} + \tau_1 \ge -\frac{t}{2n_{\min}} + \tau_1.
\end{aligned}$$

The maximum part is similar. The lower bound is $-\frac{t}{2n_{\max}} - \tau_2$. Therefore, combining them, we obtain the desired result. □

Theorem 4 gives a lower bound that describes how well the Moreau envelope function generally approximates the original one. Similar to LSE [NDS01] and WA [HCB11], the Moreau envelope we use here can also approximate HPWL in arbitrary precision. The proof is complicated and out of scope, and thus not attached due to page limit.

Theorem 4 indicates that the lower bound is irrelevant to the total degree of the net. Instead, for a fixed $t > 0$, more pins share the maximum or minimum coordinate, the lower bound will get closer to zero, *i.e.*, the model will be more accurate. However, we may also need to pay attention to earlier stages of global placement when the smoothing parameter t or γ is very large.

Theorem 5. *Let $\Delta x = W_e(\boldsymbol{x}) = \max_i x_i - \min_i x_i$ be the coordinate range. When the parameter t is large enough,* e.g., *$t \ge \frac{1}{2}\sum_{i=1}^{n}|x_i - \bar{x}|$, the Moreau envelope model W_e^t is a variance model. More specifically, we have $W_e^t(\boldsymbol{x}) = \frac{1}{2t}\sum_{i=1}^{n}(x_i - \bar{x})^2$, where $\bar{x} = \frac{1}{n}\boldsymbol{e}^\top\boldsymbol{x}$ is the average coordinate. The corresponding error bound is given by*

$$\frac{(\Delta x)^2}{4t} - \Delta x \le W_e^t(\boldsymbol{x}) - W_e(\boldsymbol{x}) \le -\frac{\Delta x}{2}. \tag{3.24}$$

$W_e^t(\boldsymbol{x})$ tends to zero as t increases to infinity.

Proof. When t is large enough, the approximation $W_e^t(\cdot) = \frac{1}{2t}\sum_{i=1}^{n}(x_i - \bar{x})^2$ is a variance model. Assume that the coordinates are sorted, *i.e.* $x_1 \le x_2 \le \cdots \le x_n$. By Cauchy-Schwarz inequality, we have

$$\begin{aligned}
2tW_e^t(\boldsymbol{x}) &= \sum_{i=1}^{n}|x_i - \bar{x}|^2 \ge |x_1 - \bar{x}|^2 + |x_n - \bar{x}|^2 \\
&\ge \frac{1}{2}\left(|x_{\min} - \bar{x}| + |x_{\max} - \bar{x}|\right)^2 = \frac{1}{2}(\Delta x)^2.
\end{aligned} \tag{3.25}$$

Then, we have the lower bound $W_e^t(\boldsymbol{x}) - W_e(\boldsymbol{x}) \ge \frac{(\Delta x)^2}{4t} - \Delta x$. The equality holds if and only if $x_2 = \cdots = x_{n-1} = \frac{1}{2}(x_1 + x_n)$ or there are only two pins in this net. As for the upper bound part, it is obvious that $\sum_{i:x_i\le\bar{x}}|x_i - \bar{x}| = \sum_{i:x_i>\bar{x}}|x_i - \bar{x}| \le t$ (Since t is chosen large such that

$t \geq \frac{1}{2}\sum_{i=1}^{n}|x_i - \bar{x}|$). We have

$$\begin{aligned} W_e^t(\boldsymbol{x}) &= \frac{1}{2t}\sum_{i=1}^{n}|x_i-\bar{x}|^2 = \frac{1}{2t}\sum_{i:x_i\leq\bar{x}}|x_i-\bar{x}|^2 + \frac{1}{2t}\sum_{i:x_i>\bar{x}}|x_i-\bar{x}|^2 \\ &\leq \frac{\sum_{i:x_i\leq\bar{x}}|x_i-\bar{x}|^2}{2\sum_{i:x_i\leq\bar{x}}|x_i-\bar{x}|} + \frac{\sum_{i:x_i>\bar{x}}|x_i-\bar{x}|^2}{2\sum_{i:x_i>\bar{x}}|x_i-\bar{x}|} \\ &\leq \frac{1}{2}\max_{i:x_i\leq\bar{x}}|x_i-\bar{x}| + \frac{1}{2}\max_{i:x_i>\bar{x}}|x_i-\bar{x}| = \frac{1}{2}\Delta x. \end{aligned} \tag{3.26}$$

The equality holds if and only if $t = \frac{1}{2}\sum_{i=1}^{n}|x_i-\bar{x}|$ and there exists an k such that $x_1 = \cdots = x_k \leq x_{k+1} = \cdots = x_n$. The theorem is proved. □

At the earlier stages of global placement, the smoothing parameter is set to a extremely large number for better smoothness. In ePlace [Lu+15a] algorithm, even at the later stages when cells have already spread out, the smoothing parameter is also not an "epsilon" value.

As a comparison, we also list the bound of the weighted-average model [HCB11; HBC13] $(n\Delta x)(n+\mathrm{e}^{\frac{1}{\gamma}\Delta x})^{-1}$ for $W_{e,\mathrm{WA}}^{\gamma}(\boldsymbol{x})$ in Equation (3.3). It may be hard to verify its correctness when the smoothing parameter γ is smaller than some thresholds due to the non-convexity issue shown in Figure 3.1(a). Since the convexity of $W_{e,\mathrm{WA}}^{\gamma}(\boldsymbol{x})$ is tedious to analyze, we decide not to compare them in a theoretical way. However, it can still be observed that when n is large, the lower bound given by [HCB11; HBC13] is very rough. Although this lower bound is not tight, we may imagine that the approximation error goes large for high-degree nets using the weighted-average model.

3.3.2 Gradient Properties

Differentiable approximations can be arbitrarily close to the HPWL function W_e. Ideally, when the smoothing parameter tends to zero, the gradient of any differentiable model should tend to a subgradient of W_e.

Theorem 6. *Consider the weighted-average model $W_{e,WA}^{\gamma}(\boldsymbol{x})$ defined by Equation (3.3) of a net with $x_{\max} > x_{\min}$. Assume that there are $n_{\max}$ pins sharing the maximum coordinates, and $n_{\min}$ pins sharing the minimum. Then, the gradient limit $\boldsymbol{g} = \lim_{\gamma\to 0^+}\nabla W_{e,WA}^{\gamma}$ exists and it is determined by*

$$g_i = \begin{cases} \frac{1}{n_{\max}}, & \text{if } x_i = x_{\max}; \\ 0, & \text{if } x_{\min} < x_i < x_{\max}; \\ -\frac{1}{n_{\min}}, & \text{if } x_i = x_{\min}. \end{cases} \tag{3.27}$$

Besides, $g(\boldsymbol{x}) \in \partial W_e(\boldsymbol{x})$ is a subgradient of W_e.

Proof. The WA model $W_{e,\mathrm{WA}}^{\gamma}(\cdot)$ is differentiable. Consider the smooth maximum in Equation (3.32). It is easy to verify that, for any i we have

$$\frac{\partial S_{\max}^{\gamma}}{\partial x_i} = \frac{\exp\left(\frac{x_i}{\gamma}\right)}{\sum_{j=1}^{n}\exp\left(\frac{x_j}{\gamma}\right)}\left(1 + \frac{1}{\gamma}(x_i - S_{\max}^{\gamma}(\boldsymbol{x}))\right). \tag{3.28}$$

If $x_i = x_{\max}$, there is no doubt that we have $\lim_{\gamma\to 0^+} \frac{\exp\left(\frac{x_i}{\gamma}\right)}{\sum_{j=1}^n \exp\left(\frac{x_j}{\gamma}\right)} = \frac{1}{n_{\max}}$, and

$$\lim_{\gamma\to 0^+} \frac{x_i - S^\gamma_{\max}(\boldsymbol{x})}{\gamma} = \lim_{\gamma\to 0^+} \frac{\sum_{j:x_j<x_{\max}} \frac{x_{\max}-x_j}{\gamma} \exp\left(\frac{x_j-x_{\max}}{\gamma}\right)}{n_{\max} + \sum_{j:x_j<x_{\max}} \exp\left(\frac{x_j-x_{\max}}{\gamma}\right)} = 0. \tag{3.29}$$

Then we have $\lim_{\gamma\to 0^+} \frac{\partial S^\gamma_{\max}}{\partial x_i} = \frac{1}{n_{\max}}$ in this case. If $x_i < x_{\max}$, we re-order Equation (3.28) in the following form

$$\frac{\partial S^\gamma_{\max}}{\partial x_i} = \frac{\exp\left(\frac{x_i}{\gamma}\right)}{\sum_{j=1}^n \exp\left(\frac{x_j}{\gamma}\right)} + \frac{\frac{1}{\gamma}\exp\left(\frac{x_i}{\gamma}\right)}{\sum_{j=1}^n \exp\left(\frac{x_j}{\gamma}\right)} \left(x_i - S^\gamma_{\max}(\boldsymbol{x})\right), \tag{3.30}$$

where $\lim_{\gamma\to 0^+} \left(x_i - S^\gamma_{\max}(\boldsymbol{x})\right) = x_i - x_{\max} \neq 0$. The other two limits are both 0, so we naturally have $\lim_{\gamma\to 0^+} \frac{\partial S^\gamma_{\max}}{\partial x_i} = 0$ in this case. We then establish similar results for the minimum part $S^\gamma_{\min}(\boldsymbol{x})$. Note that x_i cannot equal to both $x_{\max}$ and $x_{\min}$ as $x_{\max} > x_{\min}$. Combining the maximum and minimum parts, the theorem is proved. □

Theorem 7. *Consider the Moreau envelope $W^t_e(\boldsymbol{x})$. Then, we always have $\nabla W^t_e = \boldsymbol{g}$ when t is small enough, where $\boldsymbol{g}$ is determined by Equation* (3.27)*.*

Proof. We only need to show the limit of each component. Assume that there are $n_{\max}$ pins sharing the maximum coordinate $x_{\max}$, and let $\tilde{x}$ be the second largest coordinate. Obviously, when $t < n_{\max}(x_{\max} - \tilde{x})$, we have

$$\tau_2 = x_{\max} - \frac{t}{n_{\max}} > \tilde{x}, \tag{3.31}$$

which indicates that only these $n_{\max}$ pins with maximum coordinate have gradients, and each gradient is exactly equal to $g = \frac{1}{t}(x_{\max} - \tau_2) = \frac{1}{n_{\max}}$. The minimum part is similar, and then we have completed the proof. □

If we only consider the maximum part, the subgradients of $\max(\boldsymbol{x})$ form a convex hull: $\text{conv}\{\boldsymbol{e}_i : x_i = x_{\max}\}$, where $\boldsymbol{e}_i$ is the unit vector with the i-th entry being one. The components of any subgradient should sum to 1. The differentiable approximation models should also satisfy such a property.

Theorem 8. *The smooth maximum in the weighted-average model defined as*

$$S^\gamma_{\max}(\boldsymbol{x}) = \frac{\sum_{i=1}^n x_i \exp\left(\frac{x_i}{\gamma}\right)}{\sum_{i=1}^n \exp\left(\frac{x_i}{\gamma}\right)} \tag{3.32}$$

has a gradient whose components sum to 1.

Proof. Consider the smooth maximum in Equation (3.32) and its gradient in Equation (3.28), then

we obtain

$$\begin{aligned}\sum_{i=1}^{n}\frac{\partial S_{\max}^{\gamma}}{\partial x_i} &= \sum_{i=1}^{n}\frac{\exp\left(\frac{x_i}{\gamma}\right)}{\sum_{j=1}^{n}\exp\left(\frac{x_j}{\gamma}\right)}\left(1+\frac{1}{\gamma}(x_i - S_{\max}^{\gamma}(\boldsymbol{x}))\right) \\ &= 1+\frac{1}{\gamma}\left(\sum_{i=1}^{n}\frac{x_i\exp\left(\frac{x_i}{\gamma}\right)}{\sum_{j=1}^{n}\exp\left(\frac{x_j}{\gamma}\right)} - S_{\max}^{\gamma}(\boldsymbol{x})\right) \\ &= 1+\frac{1}{\gamma}(S_{\max}^{\gamma}(\boldsymbol{x}) - S_{\max}^{\gamma}(\boldsymbol{x})) = 1.\end{aligned} \tag{3.33}$$

The theorem is proved. □

Corollary 2. *The weighted-average wirelength $W_{e,WA}^{\gamma}(\boldsymbol{x})$ defined by Equation (3.3) has a gradient whose components sum to 0.*

Similar to Equation (3.32), we can define the smooth minimum $S_{\min}^{\gamma}(\boldsymbol{x})$. Then it is trivial to obtain the result as we always have $W_{e,\mathrm{WA}}^{\gamma}(\boldsymbol{x}) = S_{\max}^{\gamma}(\boldsymbol{x}) - S_{\min}^{\gamma}(\boldsymbol{x})$. We are going to show that the Moreau envelope model also preserve the properties in Theorem 8 and Corollary 2.

Theorem 9. *Let $\boldsymbol{g}$ be the gradient of Moreau envelop $\nabla W_e^t(\boldsymbol{x})$, then we have $\sum_{i:x_i\geq\tau_2} g_i = 1$ and $\sum_{i:x_i\leq\tau_1} g_i = -1$.*

Proof. From Equation (3.10), we know that $g_i = \frac{1}{t}(x_i - \tau_2)$ when $x_i \geq \tau_2$, therefore, it is almost trivial that

$$\sum_{i:x_i\geq\tau_2} g_i = \frac{1}{t}\sum_{i:x_i\geq\tau_2}(x_i-\tau_2) = \frac{1}{t}\sum_{i=1}^{n}(x_i-\tau_2)^{+} = 1, \tag{3.34}$$

where we use the constraint of τ_2 in Equation (3.11). Similarly, we can easily verify $\sum_{i:x_i\leq\tau_1} g_i = -1$. The theorem is proved. □

Corollary 3. *The Moreau envelope model $W_e^t(\boldsymbol{x})$ has a gradient whose components sum to 0.*

Corollary 3 indicates that $W_e^t(\boldsymbol{x})$ preserves the gradient properties like Corollary 2. No matter what wirelength model is adopted, the total gradients w.r.t. all pins should sum to zero.

3.4 Experimental Results

We use benchmarks from the ISPD2006 [Nam06] and ISPD2019 [Liu+19] contests. The circuit statistics are shown in Table 3.1. Compared to the ISPD2006 contest [Nam06] targeting at wirelength only, recent contest benchmarks may focus more on other objectives like routability, timing, and region-constrained.

We implemented our model in C++/CUDA based on the open-source analytical placer DREAMPLACE [Lin+19]. The following experiments are conducted on a 64-bit Linux workstation with Intel Xeon 2.90GHz CPUs and an NVIDIA GeForce RTX 3090 GPU.

We compare wirelength and runtime results using BiG_CHKS [SC19], LSE [NDS01], and WA [HCB11] on ISPD2006 benchmarks [Nam06] in Table 3.2 and on ISPD2019 benchmarks [Liu+19] in Table 3.3.

Table 3.1: *The statistics of ISPD2006 [Nam06] and ISPD2019 [Liu+19] contest benchmarks.*

Bench.		#Movable	#Fixed	#Nets	#Pins
ISPD2006 [Nam06]	adaptec5	842482	646	867798	3433359
	newblue1	330137	337	338901	1223165
	newblue2	440239	1277	465219	1761069
	newblue3	482833	11178	552199	1881267
	newblue4	642717	3422	637051	2455617
	newblue5	1228177	4881	1284251	4849194
	newblue6	1248150	6889	1288443	5200208
	newblue7	2481372	26582	2636820	9971913
ISPD2019 [Liu+19]	ispd_test1	8879	0	3153	17203
	ispd_test2	72090	4	72410	318245
	ispd_test3	8208	75	8953	30271
	ispd_test4	146435	7	151612	436707
	ispd_test5	28914	8	29416	80757
	ispd_test6	179865	16	179863	793289
	ispd_test7	359730	16	358720	1584844
	ispd_test8	539595	16	537577	2376399
	ispd_test9	899325	16	895253	3957481
	ispd_test10	899325	79	895253	3957499

In our experiments, we use the ePlace [Lu+15a] algorithm to perform analytical placement and DreamPlace [Lin+19] as the placer because it has enabled high-performance GPU-accelerated techniques to obtain high-quality results extremely fast. For a fair comparison on wirelength optimization, we follow the settings of [SC19] on ISPD2006 benchmarks [Nam06]. Considering that the reported wirelength and runtime results of BiG_CHKS and BiG_WA in [SC19] are roughly equal, we re-implement the BiG model proposed in [SC19] with CHKS bivariate function [CH93] in DreamPlace [Lin+19]. The CHKS function [CH93] is more representative of bivariate functions.

After global placement, legalization [SSJ08a], and detailed placement, we evaluate the results and list them in Table 3.2. We incorporate ABCDPlace [Lin+20b] as our detailed placement engine to fully leverage the GPU resources. Since the original binary of BiG in [SC19] is unavailable, as notified by the author, so we cite the performance directly from [SC19], listed in the column named "BiG_CHKS Reported in [SC19]" in Table 3.2. There exists a quality gap between their reported results and ours, so we also list the results of executing the binary of NTUPlace3 [Che+08] on our machine for reference. The wirelength results in Table 3.2 are evaluated by NTUPlace3 [Che+08] for a fair comparison. Table 3.2 shows that we can achieve more than 1% improvements on ISPD2006 [Nam06] after detailed placement. It is worth mentioning that the maximum improvement is over **5%** on `newblue1` which has large movable macros. Table 3.2 indicates that the wirelength improvements are preserved after detailed placement.

We also test our wirelength model on the recent ISPD2019 benchmarks [Liu+19]. Since NTUPlace3 [Che+08] and NTUplace4dr [Hua+17] binary executable files currently do not support ISPD2019 [Liu+19], we only compare different models incorporated in DreamPlace [Lin+19] in Table 3.3. As shown in the table, the achieved improvement is more than 1.5% and almost 2% on ISPD2019 benchmarks [Liu+19].

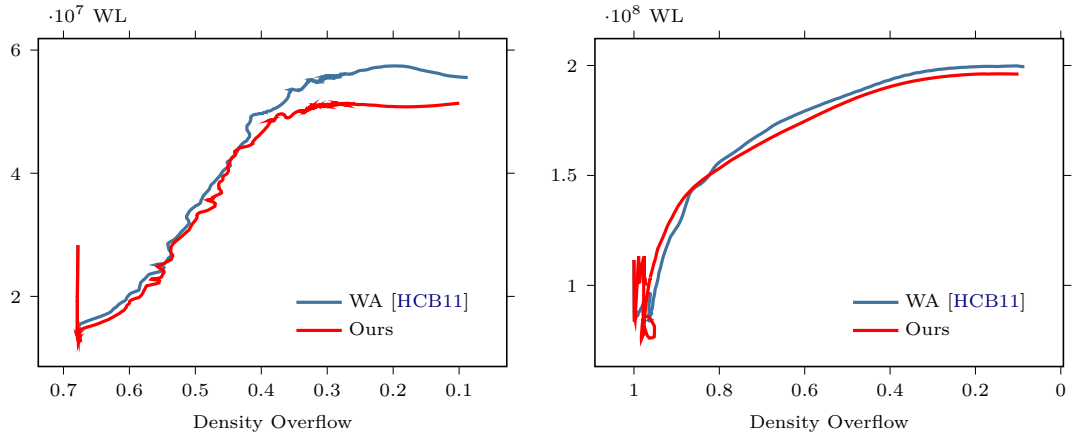
To visualize the trend of wirelength during global placement, we take `newblue1` in ISPD2006 [Nam06] and `ispd19_test10` in ISPD2019 [Liu+19] as examples and plot the curves in Figure 3.3. We can

Table 3.2: *The HPWL and runtime comparison on the ISPD2006 contest benchmarks [Nam06].* ***LGWL*** *(10^6) indicates the HPWL results after legalization.* ***DPWL*** *(10^6) indicates the HPWL results after detailed placement.* ***RT*** *(s) indicates the total runtime results including global placement, legalization, and detailed placement.*

Bench.	NTUPlace3 [Che+08]			BiG_CHKS Reported in [SC19]			DreamPlace [Lin+19] + BiG_CHKS [SC19]			DreamPlace [Lin+19] + LSE [NDS01]			DreamPlace [Lin+19] + WA [HCB11]			Ours		
	LGWL	DPWL	RT	LGWL	DPWL	RT	LGWL	DPWL	RT	LGWL	DPWL	RT	LGWL	DPWL	RT	LGWL	DPWL	RT
adaptec5	360.62	344.57	1249	281.23	270.82	1830	302.00	297.55	56.19	302.51	297.87	43.77	304.38	299.28	46.05	302.25	297.26	76.45
newblue1	62.770	60.152	319	88.350	84.150	642	61.516	60.328	35.33	61.710	60.548	29.08	60.354	58.784	30.08	57.047	55.590	56.17
newblue2	205.65	191.66	647	185.98	174.73	696	180.23	176.96	32.64	180.94	177.77	25.42	181.07	177.57	25.92	180.07	176.73	53.18
newblue3	290.59	275.91	583	278.43	258.71	1351	262.59	257.89	45.89	258.64	257.66	32.13	258.14	257.48	34.43	259.34	256.28	60.80
newblue4	260.08	247.59	916	219.10	212.20	1040	220.01	214.99	40.92	220.69	215.30	37.74	221.07	216.20	36.13	221.16	215.71	54.16
newblue5	447.30	426.91	2077	403.03	387.15	3290	388.39	381.19	81.14	390.02	382.37	64.86	389.26	382.05	65.27	387.77	380.18	118.13
newblue6	521.82	498.16	2048	433.55	417.42	2928	446.93	441.76	87.66	445.61	440.63	69.53	446.85	441.14	68.07	445.33	439.83	118.55
newblue7	1160.1	1100.2	4119	990.40	957.06	7289	944.51	931.32	148.53	942.77	929.64	131.83	945.07	931.60	128.36	941.34	926.99	193.13
Avg.	1.161	1.124	14.61	1.080	1.051	22.52	1.012	1.013	0.71	1.011	1.014	0.58	1.010	1.011	0.58	1.000	1.000	1.00

Table 3.3: *The HPWL and runtime comparison on the ISPD2019 contest benchmarks [Liu+19].* ***LGWL*** *(10^6) indicates the HPWL results after legalization.* ***DPWL*** *(10^6) indicates the HPWL results after detailed placement.* ***RT*** *(s) indicates the total runtime results including global placement, legalization, and detailed placement.*

Bench.	[Lin+19] + BiG_CHKS [SC19]			[Lin+19] + LSE [NDS01]			[Lin+19] + WA [HCB11]			Ours		
	LGWL	DPWL	RT	LGWL	DPWL	RT	LGWL	DPWL	RT	LGWL	DPWL	RT
ispd19_test1	0.41688	0.40778	7.06	0.41341	0.40561	7.09	0.41856	0.40887	5.33	0.41036	0.40225	14.32
ispd19_test2	18.211	18.027	11.74	17.858	17.705	10.96	18.047	17.890	10.83	17.500	17.345	15.97
ispd19_test3	0.75403	0.71541	7.24	0.73048	0.70335	7.60	0.76892	0.72112	7.17	0.74330	0.70771	14.42
ispd19_test4	17.182	17.021	15.23	17.153	16.983	13.37	17.189	17.027	12.78	17.087	16.903	20.68
ispd19_test5	3.5963	3.3161	11.83	3.6446	3.3097	14.30	3.5786	3.2727	14.77	3.5066	3.2281	15.78
ispd19_test6	45.497	45.097	15.28	45.402	44.987	14.27	45.138	44.755	14.68	44.707	44.351	23.33
ispd19_test7	89.457	88.373	25.79	90.731	89.496	23.52	90.105	88.831	23.87	88.526	87.561	37.74
ispd19_test8	135.21	134.06	32.99	134.60	133.57	29.92	135.05	133.89	30.43	133.04	131.89	45.66
ispd19_test9	212.99	208.71	47.53	213.40	209.51	45.01	210.98	207.16	43.79	208.83	205.14	66.53
ispd19_test10	211.68	209.46	47.78	211.21	209.04	45.81	211.99	209.73	44.46	208.48	206.32	64.59
Avg.	1.018	1.017	0.67	1.014	1.014	0.65	1.018	1.015	0.64	1.000	1.000	1.00



(a) *ISPD2006* `newblue1` **(b)** *ISPD2019* `ispd19_test10`

Figure 3.3: *(a) The wirelength curve against density overflow during global placement for ISPD2006* `newblue1`*. (b) The wirelength curve against density overflow during global placement for ISPD2019* `ispd19_test10`*. The overflow decreases as the nodes spread out.*

achieve approximately 5.4% and 1.6% improvement, respectively, compared to WA [HCB11] after detailed placement.

We set the stop overflow to $\phi' = 0.1$ for all experiments. From Figure 3.3, we can observe that all of them can successfully converge to placement solutions with low overflow. The weighted-average [HCB11] and BiG_CHKS [SC19] behave similarly, while the overflow decreases slower using our model. Nevertheless, our wirelength model reduces the overflow to a lower value at the end.

From Figure 3.3, we can observe that the wirelength result obtained with our model is better at the same density overflow during global placement on the adopted cases. The density overflow roughly reflects the overall cell overlap. Therefore, a smaller wirelength at the same density overflow implies better placement quality.

3.5 Summary

In this chapter, we propose a novel HPWL-based differentiable wirelength model. We have made both theoretical and experimental comparisons of the widely-used WA model [HCB11] and our Moreau envelope model. It has been shown that our model has the advantage of numerical stability and convexity, which is preferred in numerical optimization. Moreover, our model has a better approximation error bound. Experimental results show that our model can rapidly produce better placement solutions achieving up to 5.4% HPWL improvement and more than 1% improvement on average compared to the most widely-used nonlinear wirelength models with GPU acceleration. Since further improvement on HPWL is challenging, our future work shall focus on novel optimizers to generally improve the analytical placement quality.

CHAPTER 4
TIMING-DRIVEN PLACEMENT

Timing optimization is critical to integrated circuit (IC) design closure. In this chapter, we propose a timing-driven global placement engine with momentum-based net weighting. We choose the net-based approach for its scalability to global perturbation of cells in global placement. The major contributions are summarized as follows.

- **Momentum-based net weighting**. The weighting scheme plays an important role in our timing-driven global placement algorithm. At each timing iteration, we expect to assign weights to different nets by incorporating the current slacks within the existing criticality information. The net weights will be updated gradually by considering the new weights, computed according to slacks, to be a momentum term, which is analogous to the *momentum method* that is widely used in backpropagation learning [RHW86].
- **Preconditioning technique for net weighting**. The preconditioner proposed by the original ePlace [Lu+15a] algorithm does not consider different net weights. Considering that we may assign very different net weights to different nets to optimize timing, the numerical stability may get negatively affected, especially for those cells incident to critical paths. We enhance our preconditioner to adapt different net weights when optimizing cell locations.
- Experimental results on the ICCAD2015 contest benchmark suites [Kim+15] show that on average we can achieve 46.83% improvements on TNS, and 30.27% improvements on WNS, compared to the state-of-the-art placer [Lin+20a] after global placement and legalization.

The rest of this chapter is organized as follows. Section 4.1 provides some preliminaries including brief foundations of nonlinear placement, static timing analysis, and timing optimization. Section 4.2 presents the overall flow of our timing-driven global placement algorithms and the detailed explanations. Section 4.3 demonstrates the experimental results and some related analysis, followed by Section 4.4 summarizing the whole paper.

4.1 Preliminary

4.1.1 Nonlinear Global Placement

At the global placement stage, a given circuit is considered to be a graph where vertices model gates. Placers are expected to place millions of instances to appropriate locations such that the total

wirelength can be minimized. The netlist $\mathcal{N} = (E, V)$ consists of a net set E and a node (cell) set V. Suppose that we have n nodes in the design (*i.e.* $|V| = n$), the global placement seeks locations $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^n \times \mathbb{R}^n$ that minimize the total half-perimeter wirelength $W(\boldsymbol{x}, \boldsymbol{y})$. If we assign a net weight w_e for each net $e \in E$, the optimization formulation can be modified to minimizing the weighted sum

$$\min_{\boldsymbol{x},\boldsymbol{y}} \sum_{e \in E} w_e W(e; \boldsymbol{x}, \boldsymbol{y}). \tag{4.1}$$

There are many modern techniques to smoothly approximate the half-perimeter wirelength model $W(e; \boldsymbol{x}, \boldsymbol{y})$. Nonlinear placement adopts a *nonlinear* differentiable approximation of $W(e; \boldsymbol{x}, \boldsymbol{y})$. A widely-used approximation is the weighted-average (WA) model [HCB11; HBC13],

$$\begin{aligned} \tilde{W}_x(e, \gamma; \boldsymbol{x}, \boldsymbol{y}) &= \frac{\sum_{i \in e} x_i \mathrm{e}^{\frac{x_i}{\gamma}}}{\sum_{i \in e} \mathrm{e}^{\frac{x_i}{\gamma}}} - \frac{\sum_{i \in e} x_i \mathrm{e}^{-\frac{x_i}{\gamma}}}{\sum_{i \in e} \mathrm{e}^{-\frac{x_i}{\gamma}}}, \\ \tilde{W}(e, \gamma; \boldsymbol{x}, \boldsymbol{y}) &= \tilde{W}_x(e, \gamma; \boldsymbol{x}, \boldsymbol{y}) + \tilde{W}_y(e, \gamma; \boldsymbol{x}, \boldsymbol{y}), \end{aligned} \tag{4.2}$$

where $\tilde{W}_x(e, \gamma; \boldsymbol{x}, \boldsymbol{y})$ and $\tilde{W}_y(e, \gamma; \boldsymbol{x}, \boldsymbol{y})$ are the net wirelength along horizontal and vertical direction, respectively. γ is a hyperparameter to control the precision of this approximation. A typical non-linear placement problem can be formulated as

$$\min_{\boldsymbol{x},\boldsymbol{y}} \sum_{e \in E} w_e W(e; \boldsymbol{x}, \boldsymbol{y}) + \lambda D(\boldsymbol{x}, \boldsymbol{y}), \tag{4.3}$$

where E is the net set, $W(e; \cdot, \cdot)$ is the wirelength function that calculates the total wirelength of a specific net $e \in E$, function $D(\cdot, \cdot)$ indicates the total density penalty and λ is the corresponding density weight. This is a typical *unconstrained* optimizatin problem with arguments $\boldsymbol{x}, \boldsymbol{y}$ being the cell locations.

The objective function (4.3) is required to be everywhere differentiable so that we can use gradient-based methods to optimize the variables. Additionally, the term $W(e; \boldsymbol{x}, \boldsymbol{y})$ is a nonlinear approximation of the net wirelength.

4.1.2 Static Timing Analysis

The timing-driven placement has to be guided by timing analysis. Static timing analysis (STA) evaluates the setup/hold timing performance of a circuit under best-case and worst-case scenarios based on its delay-annotated timing graph [MS99; Hit82]. It performs forward and backward propagation to compute the arrival time and the required arrival time for each node in the graph, respectively [PHR08].

More specifically, we model the given circuit as a *directed acyclic graph* (DAG). Each node in the DAG corresponds to a pin in the circuit, and each edge in this graph represents a directed pin-to-pin connection. A complete STA process evaluates the delays of nets and cells, and then computes the arrival time and required arrival time of pins through propagation. For a specific pin p, assume that we have its arrival time $t_{\mathrm{at}}(p)$ and required arrival time $t_{\mathrm{rat}}(p)$, then the *slack* of p is defined as the

difference of its required arrival time minus arrival time,

$$s(p) = t_{\text{rat}}(p) - t_{\text{at}}(p). \tag{4.4}$$

Slack is an important metric to evaluate the timing quality of a placement solution. The *worst negative slack* (WNS) is the most commonly used timing metric, defined as the worst one among all negative slacks of timing endpoints,

$$s_{\text{wns}} = \min_{t \in P_{\text{end}}} s(t), \tag{4.5}$$

where P_{end} indicates the set of all timing endpoints, and s_{wns} is the worst negative slack. We assume that there exists at one $t \in P_{\text{end}}$ such that $s(t) < 0$, otherwise the timing constraints are perfectly satisfied. Another well-known timing closure objective is the *total negative slack* (TNS), defined as the sum of all negative slacks of timing endpoints,

$$s_{\text{tns}} = \sum_{t \in P_{\text{end}}, s(t) < 0} s(t), \tag{4.6}$$

where s_{tns} stands for the total negative slack [Raj+03].

4.1.3 Timing Optimization

Timing-driven placement pays more attention to timing optimization. Rather than the total wirelength of the circuit design, we prefer objectives that are more suitable to reflect timing metrics. TNS and WNS are both well-adopted timing metrics, however, they may emphasize different aspects. Intuitively, WNS may only provide timing information of a single critical path, while TNS gives the overall timing information of multiple or even a large number of critical paths. Empirically, TNS should be more important to guide the timing optimization during global placement, as it can integrate information of all critical paths. On the contrary, WNS should be a more important metric when optimizing timing precisely in detailed placement.

The complete formulation of timing-driven global placement can be summarized as follows.

$$\begin{aligned} \max \quad & s(\boldsymbol{x}, \boldsymbol{y}) \\ \text{s.t.} \quad & \rho_b(\boldsymbol{x}, \boldsymbol{y}) \leq \rho_t, \forall b \in B, \end{aligned} \tag{4.7}$$

where B is the set of $m \times m$ planar grids (bins) for a positive integer m, $\rho_b(\boldsymbol{x}, \boldsymbol{y})$ denote the density of a bin $b \in B$, ρ_t represents the target placement density of each bin, and the objective function $s(\boldsymbol{x}, \boldsymbol{y})$ stands for a *negative* slack function. Typically, the objective function $s(\boldsymbol{x}, \boldsymbol{y})$ can be TNS $s_{\text{tns}}(\boldsymbol{x}, \boldsymbol{y})$ or WNS $s_{\text{wns}}(\boldsymbol{x}, \boldsymbol{y})$. Equation (4.7) shares the same cell constraints but uses a completely different objective function from that of wirelength-driven analytical placement. Unlike wirelength functions that usually have closed-form representations with respect to cell locations directly, slack functions cannot be represented explicitly. That is the reason why we decide to optimize timing indirectly by implementing net weighting schemes in the wirelength optimization.

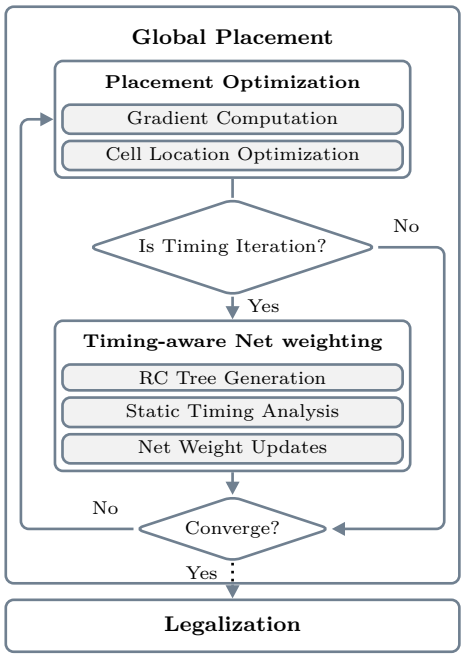


Figure 4.1: *Our overall flow with timing optimization.*

4.2 Algorithms

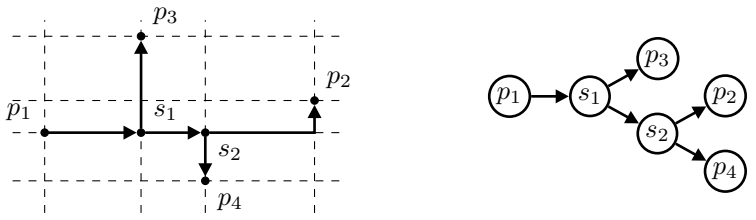
In this chapter, we expect to integrate static timing analysis into the iterations of cell location updates during analytical global placement, so that the placement solution can be optimized in terms of timing. More specifically, we expect the timing analysis tool to provide us with detailed information about how good the current placement is, which paths have intensive impact in terms of timing and how such it can affect the cell locations during the placement.

The overall flow of our placement framework with timing analysis is illustrated in Figure 4.1. Compared to modern gradient-based analytical placers, we must determine whether to perform timing analysis at each gradient-based iteration.

4.2.1 RC Tree Construction

We must construct RC trees for nets manually at every timing iteration, as the cell locations are going to be changed in every backward step. Given a *possibly illegal* placement solution, we are provided with all pin locations for each net.

For a specific net, we start with the pin locations it contains. A FLUTE [CW07] call will be performed to construct the rectilinear Steiner minimal tree of this net. This Steiner tree generally reflects how the timing propagation will be performed internally inside the timer we use. We take a simple 4-pin net as an example to illustrate how to construct the RC tree for a certain net in Figure 4.2(a) and Figure 4.2(b). The abstract RC tree hierarchy is shown in Figure 4.2(b). To construct RC information from interconnects, we require the resistance value per unit length r' and the capacitance



(a) *A Steiner tree example* **(b)** *The corresponding RC tree*

Figure 4.2: *A 4-pin net example of Steiner tree and the corresponding RC tree constructed for net delay calculation.*

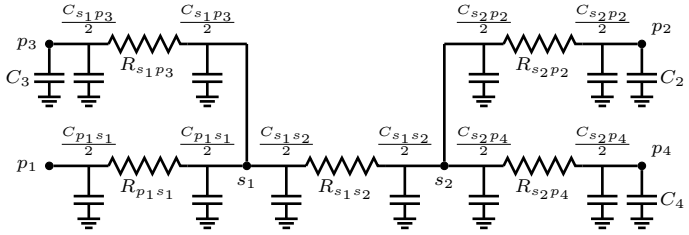


Figure 4.3: *The Elmore delay model for the above 4-pin net example.*

value per unit length c', which should be pre-determined for the given design.

4.2.2 Delay Calculation

We can enrich details on Figure 4.2(b) by adding some abstract resistors and capacitors for edges in the RC tree, illustrated in Figure 4.3. Here the Elmore delay model [Elm48] is used to approximate actual delays. More specifically, we use $\Pi-$model to break wires into RC sections. After we fill the RC information into the RC tree initialized by the timer, we then naturally proceed to the static timing analysis.

4.2.3 Momentum-based Net Weighting

Net weights are assigned to all nets in the design so that some prior knowledge of how much contribution to the objective function these nets will make can be fed into the placer during global placement. A net with a higher weight will be more sensitive to the updates of cell locations, as a perturbation to its total wire-length will lead to a greater impact on the objective we are optimizing. The optimizer will implicitly get a stronger will to place cells containing pins included in nets with higher weights closer. Without any doubt, critical nets should be reasonably assigned higher weights to remind the placer to place cells related to them closer.

Net Criticality. Our placement database considers *criticality* value as a guide to update net weights. Let c_e and s_{wns} denote the criticality value of a specific net e and the worst negative slack,

respectively. We can define the *momentum* of criticality value of a net e as

$$c_{\text{mom},e} = \begin{cases} 0, & \text{if } s_{\text{wns}} \geq 0; \\ \max\left\{0, \dfrac{s_e}{s_{\text{wns}}}\right\}, & \text{otherwise,} \end{cases} \tag{4.8}$$

where s_e is the net slack of e. If the worst negative slack s_{wns} is non-negative, everybody should be satisfied and we will do nothing to the net weights. Otherwise, $s_{\text{wns}} < 0$ is negative, and the criticality value is defined as the maximum value between 0 and the slack ratio s_e/s_{wns}.

If a net e has a non-negative slack $s_e \geq 0$, its criticality momentum will be set to $c_{\text{mom},e} = 0$, otherwise, it will be the ratio $\frac{s_e}{s_{\text{wns}}} = \frac{|s_e|}{|s_{\text{wns}}|}$. For net e, The higher slack value $|s_e|$ we obtain, the higher criticality momentum $c_{\text{mom},e}$ it will have.

Intuitively, the criticality indicates the probability of net e to be critical. Since we may obtain different timing-critical paths reported by the timer at each iteration, the criticality values should also be updated iteratively. A critical net may be related to multiple critical paths, and different nets may have different negative slacks. Hence, we are supposed to handle different critical nets in different ways. Nets with more negative slacks are considered to be more sensitive to timing metrics, and we should assign higher weights to them accordingly.

Define the criticality value of a net e at the m-th iteration as $c_e^{(m)}$. From Equation (4.8), we know that $s_e^{(m)}$ and $s_{\text{wns}}^{(m)}$, which represent the net slack and the WNS at the m-th iteration respectively, can be re-calculated at each timing iteration. Then we obtain a criticality value $c_e^{(m)}$ of net e at the m-th iteration, which corresponds to the criticality updates.

Net weighting Scheme. We introduce a momentum-based net weighting scheme. For a specific net e, let $\tilde{w}_e^{(m)} = \ln w_e^{(m)}$ and $\Delta\tilde{w}_e^{(m)}$ be the logarithmic net weight of w_e and its increment at the m-th timing iteration, respectively.

$$\tilde{w}_e^{(m+1)} = \tilde{w}_e^{(m)} + \Delta\tilde{w}_e^{(m)}, \;\; m \in \mathbb{N}. \tag{4.9}$$

The increment value $\Delta\tilde{w}_e^{(m)}$ is treated as the gradient determined by the timing metrics. Considering that we will obtain a new criticality *momentum* at each timing iteration. We expect the net weight $w_e^{(m)}$ to be emphsized by its criticality $c_e^{(m)}$. For integer $m \in \mathbb{N}$, the increment relationship can be modeled by

$$\begin{aligned} \Delta\tilde{w}_e^{(m)} &= \tilde{c}_e^{(m)}, \\ \Delta\tilde{w}_e^{(m+1)} &= \alpha\Delta\tilde{w}_e^{(m)} + (1-\alpha)\tilde{c}_{\text{mom},e}^{(m)}, \end{aligned} \tag{4.10}$$

where $\tilde{c}_e^{(m)} = \ln(1+c_e^{(m)})$, $\tilde{c}_{\text{mom},e}^{(m)} = \ln(1+c_{\text{mom},e}^{(m)})$ are the transformed criticality values and increments. The decay coefficient $\alpha \in [0,1]$ is a hyperparameter. The term $\Delta\tilde{w}_e^{(m)}$ can be considered as the velocity, from Equation (4.9).

The scheme in Equations (4.9) and (4.10) is inspired by the momentum-based gradient descent algorithm on backpropagation during neural network training. In backpropagation, the momentum term should be the negative gradient of the objective, so that the actual gradient increment value can be guided while remembering the history update at each iteration. Here we apply the momentum step to the update of criticality value. If a net has a positive criticality value instead of zero, its weight

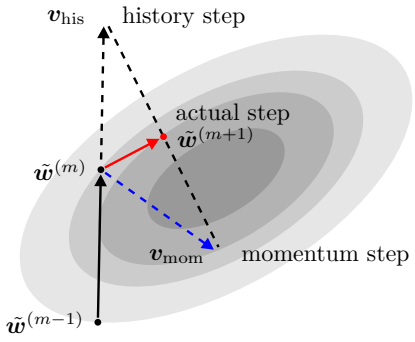


Figure 4.4: *A simple example illustrating how the momentum step vectors will affect the actual gradients.*

should be increased according to the magnitude of criticality. Besides, if a net is reported to be critical at most timing iterations, it may have a large net weight. The weight differences are acceptable as long as no value overflow is reported.

We adopt the annotations in matrix calculus, then the scheme illustrated in Equations (4.9) and (4.10) can be reformulated as

$$\Delta\tilde{\boldsymbol{w}}^{(m+1)} = \alpha\Delta\tilde{\boldsymbol{w}}^{(m)} + (1-\alpha)\tilde{\boldsymbol{c}}^{(m)}_{\text{mom}}, \tag{4.11}$$

where $\tilde{\boldsymbol{w}}^{(m)}$, $\Delta\tilde{\boldsymbol{w}}^{(m)}$, and $\tilde{\boldsymbol{c}}^{(m)}_{\text{mom}}$ indicate the logarithmic net weights, their increments, and the transformed momentum vector calculated by Equation (4.8), respectively, at the m-th timing iteration. All these vectors have the same size that is exactly the total number of nets in the design. More specifically, the i-th entry of each of these vectors indicates an attribute of the net with index i. A simple example illustrating how momentum-based net weighting works is shown in Figure 4.4.

If the momentum increment $\tilde{\boldsymbol{c}}^{(m)}_{\text{mom},e}$ has a very small magnitude in the late period of global placement, the vector $\Delta\tilde{\boldsymbol{w}}^{(m)}$ will approximately decay by the factor α with the increment of iteration m, and correspondingly the net weight $\boldsymbol{w}^{(m)}$ will gradually stabilize. Therefore, we will keep emphasizing those nets that remain critical during placement.

Unlike [EJ98] or other similar dynamic net weighting schemes, we work on all nets instead of those only on critical paths. Every net is assigned a non-trivial criticality value related to its slack at a timing iteration, and then proceeds to the net weighting.

4.2.4 Preconditioning

In numerical optimization, preconditioning is a very important step to reduce the condition number of an optimization problem. For a general unconstrained problem $\min_{\boldsymbol{x}} f(\boldsymbol{x})$, conventional preconditioning approaches aim at solving the inverse matrix of the Hessian $\boldsymbol{H}_f^{-1}$. Considering that the industrial designs may contain millions of instances, and such computation will have a huge overhead, the real implementation will become extremely unbearable.

The ePlace [Lu+15a] preconditioner only considers diagonal entries of the Hessian matrix. The objective function f is set to Equation (4.3) by default. Without loss of generality, we only consider

the horizontal direction here. Vector $\boldsymbol{x} \in \mathbb{R}^n$ represents the horizontal cell locations. The i-th diagonal entry of the Hessian matrix $\boldsymbol{H}_f^{-1}$ will be

$$\frac{\partial^2 f}{\partial x_i^2} = \sum_{e \in E} w_e \frac{\partial^2 W(e; \boldsymbol{x}, \boldsymbol{y})}{\partial x_i^2} + \lambda \frac{\partial^2 D(\boldsymbol{x}, \boldsymbol{y})}{\partial x_i^2}. \tag{4.12}$$

In the ePlace [Lu+15a] algorithm, for any net $e \in E$, the term $\frac{\partial^2 W(e;\boldsymbol{x},\boldsymbol{y})}{\partial x_i^2}$ is simply binary. We also adopt this approximation. More specifically, only if the i-th node is incident to net e will the term be set to 1. This very rough evaluation will approximate the first term in Equation (4.12) as

$$\sum_{e \in E} w_e \frac{\partial^2 W(e; \boldsymbol{x}, \boldsymbol{y})}{\partial x_i^2} \approx \sum_{e \in E_i} w_e, \tag{4.13}$$

where E_i is the net subset incident to the i-th node. Our net weighting scheme will not affect the density term, therefore we adopt the same approximation as [Lu+15a] for preconditioning.

$$\frac{\partial^2 D(\boldsymbol{x}, \boldsymbol{y})}{\partial x_i^2} = q_i \frac{\partial^2 \phi_i(\boldsymbol{x}, \boldsymbol{y})}{\partial x_i^2} \approx q_i, \tag{4.14}$$

where q_i is the quantity of electrical charge of the i-th node. The approximate preconditioning matrix on single horizontal direction will be

$$\tilde{\boldsymbol{H}}_{f_{\boldsymbol{x},\boldsymbol{x}}} = \operatorname{diag}\left(\sum_{e \in E_1} w_e + \lambda q_1, \cdots, \sum_{e \in E_n} w_e + \lambda q_n \right). \tag{4.15}$$

When net weights are all equal to 1, $\sum_{e \in E_i} w_e$ will be degraded to $|E_i|$ which stands for the total number of nets incident to the i-th node. Together with the vertical direction, the preconditioned gradient vector will be $\nabla f_{\text{precond}} = \tilde{\boldsymbol{H}}_f^{-1} \nabla f$.

4.3 Experimental Results

4.3.1 Experimental Setup

We conduct the experiments on the ICCAD 2015 contest benchmark suites [Kim+15]. Table 4.1 shows the parameters of the circuit designs. All the cases are relatively large and most of them contain millions of cells and nets. No movable macros are included in the benchmark suites. Our algorithm is implemented in C++ based on the open-source placer DreamPlace [Lin+20a] and the open-source timer OpenTimer [HW15]. Remarkably, we manage to make full use of GPU resources in both core placement [Lin+20a] and timing analysis [GHL20]. For a fair comparison, we follow the exact default hyperparameter settings of DreamPlace [Lin+20a].

4.3.2 TNS and WNS Improvement

It is important to determine when we should set up observation, perform timing analysis and update net weights. it is impossible to perform timing analysis at each iteration, as it will introduce huge overhead. Empirically, cell locations at the earlier stages are highly overlapped, and thus unreliable

Table 4.1: *Statistics of the ICCAD2015 contest benchmarks [Kim+15].*

case name	#cells	#nets	#pins	#rows
superblue1	1209716	1215710	3767494	1829
superblue3	1213253	1224979	3905321	1840
superblue4	795645	802513	2497940	1840
superblue5	1086888	1100825	3246878	2528
superblue7	1931639	1933945	6372094	3163
superblue10	1876103	1898119	5560506	3437
superblue16	981559	999902	3013268	1788
superblue18	768068	771542	2559143	1788

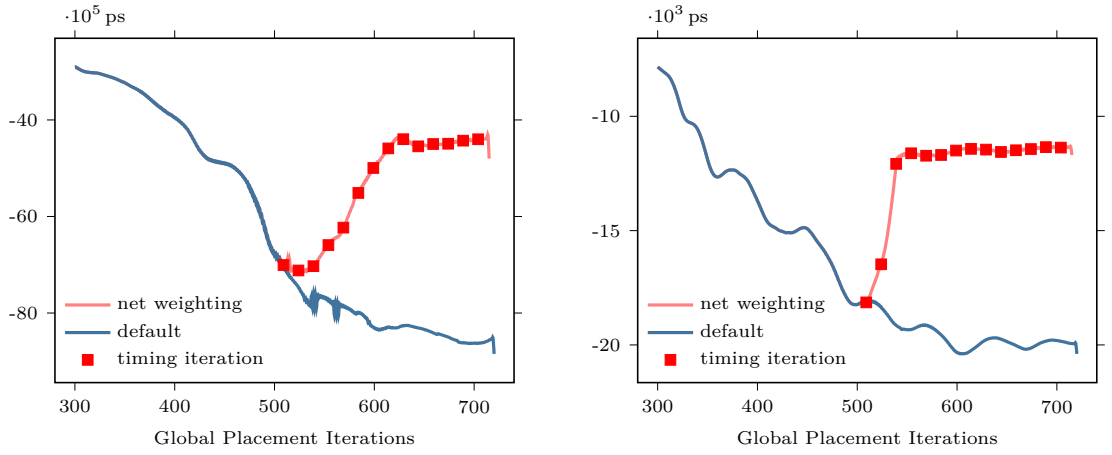
for timing analysis. A possibly appropriate time to perform timing analysis is when the cells are roughly even out by density forces. In our experiments, we evaluate timing metrics and update net weights every 15 iterations after the 500th iteration of the global placement. Additionally, we use hyperparameters manually customized in Equation (4.11) to update net weights. We use the evaluation

Table 4.2: *Comparison among DreamPlace [Lin+20a], DreamPlace [Lin+20a]+[EJ98], and our algorithm. The best results are emphasized with* ***boldface****, and the second-best results are colored in brown. The* ***TNS*** *(10^5 ps) and* ***WNS*** *(10^3 ps) stand for the total negative slack and the worst negative slack, respectively.* ***RT*** *(s) represent the end-to-end runtime.*

Bench.	DreamPlace [Lin+20a]			DreamPlace [Lin+20a]+[EJ98]			Ours		
	TNS	WNS	RT	TNS	WNS	RT	TNS	WNS	RT
superblue1	-252.359	-18.5414	**164.69**	-121.963	**-13.1548**	1320.73	**-85.0315**	-14.1031	977.56
superblue3	-88.4701	-33.2509	**153.95**	-61.2222	**-15.6518**	1247.24	**-54.7427**	-16.4341	952.11
superblue4	-196.498	-21.4654	**112.33**	-177.800	**-11.8600**	910.77	**-144.380**	-12.7808	610.26
superblue5	-208.943	-48.4825	**202.87**	-108.019	-47.7110	1758.97	**-95.7820**	**-26.7602**	1343.46
superblue7	-161.989	-20.3957	**249.32**	-84.3107	-19.9126	1968.70	**-63.8629**	**-15.2163**	1537.42
superblue10	-839.134	-33.7599	**308.81**	-786.359	**-29.0470**	1871.55	**-768.748**	-31.8796	1288.63
superblue16	-438.267	-16.8146	**102.88**	-175.543	-18.5297	875.13	**-124.181**	**-12.1115**	542.15
superblue18	-90.4280	-20.1261	**104.06**	-69.4700	**-11.7831**	887.29	**-47.2458**	-11.8705	657.47
Avg.	2.150	1.539	**0.177**	1.267	1.167	1.395	**1.000**	**1.000**	1.000

script provided by the ICCAD 2015 contest to evaluate our placement result. The results are listed in Table 4.2. All the results are evaluated after Abacus legalization [SSJ08a]. As shown in the table, we can achieve a significant improvement on both TNS (46.83% on average) and WNS (30.27% on average), compared to the DreamPlace [Lin+20a] without any timing-aware optimization.

Also, we implement the classic dynamic net weighting scheme in [EJ98]. Originally, this net weighting scheme is designed for timing-driven quadratic placement. We integrate the net weighting part for timing optimization into our implementation and make a comparison. The results are listed in the second column of Table 4.2. We use **boldface** to emphasize the best one among the three results, and color the second one with brown. As shown in the table, our net weighting scheme can outperform [EJ98] a lot on TNS. This result brings us very positive enlightenment that it is absolutely useful to consider timing-aware optimization at the global placement stage. Both TNS and WNS can be improved a lot compared to DreamPlace without any timing optimization.



(a) *The TNS curve in placement*

(b) *The WNS curve in placement*

Figure 4.5: *The TNS and WNS values at each placement iteration after the 300th iteration for* `superblue18`.

4.3.3 Visualization

To visualize the impact of net weighting on TNS and WNS, we take `superblue18` as an example and plot the TNS and WNS values after the 300th iteration in Figure 4.5. Starting from the 300th iteration, the cells have begun repelling each other, and therefore the wirelength keeps increasing, which also decreases TNS and WNS. The blue curves correspond to the results without timing optimization, while the red curves illustrate how the objectives vary with net weighting. We scatter red squares to emphasize the timing iterations.

- At nearly every timing iteration, marked with red color in Figure 4.5, TNS can get improved at once, especially when starting to break the balance of net weights.
- WNS will quickly and significantly be optimized after one or two net weighting steps. After that, it almost remains stable during the later stages of global placement.

Providing that our net weighting algorithm works on every net instead of those only on some critical paths at a timing iteration, it is quite reasonable to be effective when optimizing TNS, which may incorporate numerous critical or nearly critical paths. As for WNS, which may only give information about the worst path, it will be quickly optimized when first applying net weighting. At later stages, other critical or nearly critical paths will be taken more into consideration, and that is an important reason why it is hard to further optimize WNS during global placement.

4.3.4 Runtime Breakdown

Compared to DREAMPLACE[Lin+20a] which is very powerful to optimize cell locations with GPUs, timing-driven placement must take extra costs to perform STA and translate the feedbacks to certain operations. Hence, it is unavoidable to significantly sacrifice runtime performance for timing optimization.

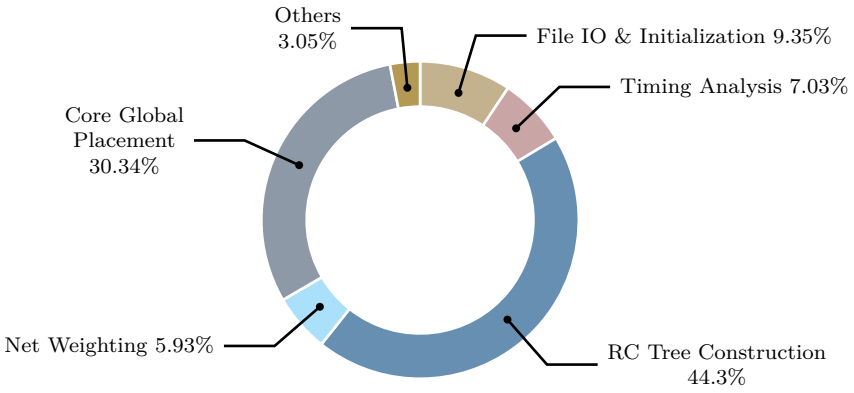


Figure 4.6: *The runtime breakdown on ICCAD2015 contest benchmark* `superblue18`.

The runtime results are listed in the third column of Table 4.2, with column name **RT**. Our weighting scheme is faster than [Lin+20a] + [EJ98], as we do not need to explicitly extract critical paths.Compared to [Lin+20a] without any timing-aware optimization, we roughly take 5 times runtime to optimize negative slacks.

Figure 4.6 plots the runtime breakdown for `superblue18`. We are still facing the runtime bottleneck dominated by the RC tree construction. It is accomplished on CPUs and thus time-consuming, especially for large nets. Considering that STA must be called multiple times to incorporate changes of cell locations, the overhead of RC tree construction and STA should be the main focus for acceleration.

4.4 Summary

In this chapter, we propose a momentum-based net weighting scheme for timing-driven global placement and improve the preconditioner accordingly. The evaluation results on ICCAD2015 contest benchmarks show that we can achieve a significant improvement on both TNS and WNS. The results of this paper enlighten us that, although most timing-aware optimization methods are performed at incremental stages, it is still very effective to consider timing at the earlier stages of physical design, especially global placement.

CHAPTER 5
ANALYTICAL 3D PLACEMENT

The emergence of 3D-IC presents challenges to traditional 2D electronic design automation methods in producing high-quality 3D circuit layouts, and the heterogeneous technology nodes further complicates the problem. Placement plays a dominant role on the overall quality of physical design, and innovations of 3D-IC placement are required to fully benefit from the 3D integration technologies. Within the context of 3D placement, 3D-IC placers are responsible for solving the optimal 3D node locations to optimize specific objectives. Such a very large-scale combinatorial optimization problem can be solved in either discrete or analytical algorithms. An analytical 3D placement algorithm is characterized by employing "true-3D" flows that handle tier partitioning continuously and devise 3D solutions directly.

Despite the various research achievements mentioned above, existing discrete and analytical 3D flows are hardly applicable to F2F-bonded 3D ICs with heterogeneous technology nodes. The discrete solutions typically fail to utilize the advantages of 3D ICs sufficiently as most of them rely on the FM-mincut tier partitioning [FM82]. However, the total cutsize is not the primary placement objective in F2F-bonded 3D ICs due to the silicon-space overhead-free property [Jun+14], resulting in sub-optimal partitioning for discrete solutions. Conventional analytical 3D placement algorithms adopt continuous optimization but they do not support heterogeneous technology nodes during global placement. Additionally, previous wirelength-driven analytical placement algorithms use inaccurate wirelength models [Lu+16] for numerical optimization, which is inconsistent with F2F-bonded scenarios. Some recent work [Che+23a] on wirelength models supports heterogeneous technology nodes in analytical placement. However, it still pays no attention to the wirelength reduction introduced by inter-die connections, remaining unsolved inaccurate estimation in 3D analytical placement.

In this chapter, we propose a new analytical 3D placement framework for F2F-bonded 3D ICs with heterogeneous technology nodes utilizing a novel and precise bistratal wirelength model. Based on the proposed placement framework, we efficiently determine the node locations along with partitioning in a single run. The main contributions are summarized as follows.

- We design a *bistratal wirelength* model, including computation strategies of the wirelength objective and gradients, that significantly outperforms the widely-used models for F2F-bonded 3D ICs.
- We propose an ultra-fast analytical 3D placement framework that leverages the *bistratal wirelength* model and eDensity-3D [Lu+16] with GPU acceleration, considering heterogeneous technology nodes.

- Experimental results show that our results achieved the best results on the ICCAD 2022 Contest Benchmarks [Hu+22] with up to 6.1% wirelength improvement and 4.1% on average, compared to the first-place winner. Remarkably, we also outperform the state-of-the-art (SOTA) analytical 3D placer [Che+23a] for heterogeneous F2F-bonded 3D ICs by up to 3.3% wirelength improvement and 2.1% on average. The usage of vertical interconnects are also significantly reduced.

The rest of this chapter is structured as follows. Section 5.1 provides some preliminaries, including previous works and foundations of analytical placement. Section 5.2 discusses the problem statement and problem formulation. Section 5.3 presents the overall flow of the proposed placement framework for heterogeneous F2F-bonded 3D ICs. Then, Section 5.4 depicts the theoretical details of the bistratal wirelength model. Section 5.5 presents experimental results and some related analysis on the adopted benchmarks, followed by the conclusion in Section 5.6.

5.1 Preliminaries

5.1.1 Analytical Placement

Global placement is performed on a netlist (V, E), where $V = \{c_1, \cdots, c_n\}$ and $E = \{e_1, \cdots, e_m\}$ are the node set and the net set, respectively. We are asked to determine the node locations $\boldsymbol{v} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ from scratch during global placement to minimize the total wirelength with little overlap allowed. A typical 3D analytical global placement problem is formulated as the following unconstrained optimization problem

$$\min_{\boldsymbol{v}} \sum_{e \in E} W_e(\boldsymbol{v}) + \lambda D(\boldsymbol{v}), \tag{5.1}$$

where $\boldsymbol{v} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ indicates the node location variables, $W_e(\cdot)$ is the net wirelength model of net $e \in E$, $D(\cdot)$ is the density model of the entire placement region evaluating the overall overlap, and λ is the density weight introduced as the Lagrangian multiplier of the density constraint. In analytical placement, we expect to make the objective differentiable and then apply numerical methods to solve Equation (5.1).

The wirelength model $W_e(\cdot)$ in the above Equation (5.1) is usually a differentiable approximation [NDS01; HCB11; HBC13; Lia+23] to the conventional net HPWL defined below.

Definition 1 (3D HPWL)**.** *Given node positions $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, the 3D HPWL of any net $e \in E$ is given by*

$$W_e(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = p_e(\boldsymbol{x}) + p_e(\boldsymbol{y}) + \alpha p_e(\boldsymbol{z}), \tag{5.2}$$

where $p_e(\boldsymbol{u}) = \max_{c_i \in e} u_i - \min_{c_i \in e} u_i$ denote the range or peak-to-peak function that evaluates the difference of maximum minus minimum in a net, and $\alpha \geq 0$ is a weight factor.

$p_e(\cdot)$ denotes partial HPWL along one axis. In real applications, it is approximated by a *differentiable* model, *e.g.* the weighted-average [HBC13] model given a smoothing parameter $\gamma > 0$,

$$p_{e,\mathrm{WA}}(\boldsymbol{u}) = \frac{\sum_{c_i \in e} u_i \mathrm{e}^{\frac{1}{\gamma} u_i}}{\sum_{c_i \in e} \mathrm{e}^{\frac{1}{\gamma} u_i}} - \frac{\sum_{c_i \in e} u_i \mathrm{e}^{-\frac{1}{\gamma} u_i}}{\sum_{c_i \in e} \mathrm{e}^{-\frac{1}{\gamma} u_i}}. \tag{5.3}$$

Other differentiable models [NDS01; Lia+23] are also applicable. Note that the z-dimension is usually defined manually, as tiers are discretely distributed in 3D scenarios. The corresponding weight factor $\alpha \geq 0$ is determined in accordance with specific objectives in real applications.

The state-of-the-art density model $D(\cdot)$ is the eDensity family [Lu+13; Lu+15a; Lu+15b; Lu+16] based on electrostatics field, where every node $c_i \in V$ is modeled by an electric charge. We implement eDensity-3D [Lu+16] as our density model with GPU acceleration in the proposed framework.

The optimization formulation in Equation (5.1) is general and thus can be applied in both 2D and 3D analytical global placement. In conventional 2D cases, the variable $\boldsymbol{v} = (\boldsymbol{x}, \boldsymbol{y})$ is optimized to find planar cell coordinates [Lu+15a; Lin+19; Liu+22]. In 3D cases, the framework is well-established in ePlace-3D [Lu+16] where the z-direction coordinates is considered to optimize $\boldsymbol{v} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$.

5.2 Problem Formulation

5.2.1 Problem Statement

In this chapter, we focus on the 3D placement problem with die-to-die (D2D) connections, specified in the ICCAD 2022 Contest [Hu+22]. The general requirement is to partition the given standard cells into two dies with different technologies, create vertical interconnections named hybrid bonding terminals (HBTs) for split nets, and determine the locations of all nodes including standard cells and HBTs so that the following constraints are satisfied:

- **Utilization Constraints**. The utilization requirements of the top die and the bottom die are provided separately, leading to different area upper bound for two dies.
- **Technology Constraints**. The cells may be fabricated using different technologies on different dies, *i.e.*, the cell characteristic, cell height, cell width, and the cell layout would be different.
- **Vertical Interconnection Constraints**. For any net e split to two dies, an HBT should be created to connect pins on two dies. All HBTs share the same size.
- **Legality Constraints**. All standard cells on both dies should be placed without overlap and aligned to rows and sites. HBTs should be placed to satisfy the spacing constraint, *i.e.*, the distance between each pair of HBTs and the distance to boundaries are lower bounded.

The objective of this 3D placement problem is the total wirelength of all nets in the given design defined in Definition 3. In short, we focus on minimizing the sum of HPWL on the two dies. The center points of the HBTs are included in the HPWL calculation for each die. We will give rigorous mathematical formulations in Section 5.2.2.

5.2.2 Problem Formulation

Consider a netlist (V, E) where $V = \{c_1, \cdots, c_n\}$ is the node set and $E = \{e_1, \cdots, e_m\}$ is the net set. A partition is determined by a 0-1 vector $\boldsymbol{\delta} \in \mathbb{Z}_2^n = \{0, 1\}^n$, where $\delta_i = 0$ indicates that cell $c_i \in V$ is placed on the bottom die, otherwise top die. In the 3D placement with D2D vertical connections,

the partition determines the total number of hybrid bonding terminals. In this section, we use $\boldsymbol{x}, \boldsymbol{y}$ to represent both node coordinates and corresponding pin coordinates ignoring pin offsets for simplicity.

Definition 2 (Net Cut Indicator)**.** *The cut indicator of a net $e \in E$ is a function of partition $\boldsymbol{\delta} \in \{0,1\}^n$ defined by*

$$C_e(\boldsymbol{\delta}) = \max_{c_i \in e} \delta_i - \min_{c_i \in e} \delta_i. \tag{5.4}$$

It is also a binary value in $\{0,1\}$. If there exist two nodes incident to net e placed on two different dies, the cut $C_e(\boldsymbol{\delta}) = 1$, otherwise it is 0.

Given a partition $\boldsymbol{\delta} \in \{0,1\}^n$, if a net $e \in E$ is a split net, *i.e.*, $C_e(\boldsymbol{\delta}) = 1$, a hybrid bonding terminal (HBT) should be inserted for this net as a vertical connection. Otherwise, all nodes incident to $e \in E$ are placed on either the top or the bottom die. Different from TSVs and MIVs going through silicon substrates, HBTs do not require silicon space. If we have $C_{e_i}(\boldsymbol{\delta}) = 1$ for a net $e_i \in E$, one and only one HBT t_i should be assigned to e_i accordingly, otherwise t_i will be discarded. We denote the set of HBTs by $T = \{t_1, \cdots, t_m\}$ with planar coordinates $\boldsymbol{x}', \boldsymbol{y}'$.

Denote the top and bottom partial nets by $e^+(\boldsymbol{\delta}) = \{c_i \in e : \delta_i = 1\}$ and $e^-(\boldsymbol{\delta}) = \{c_i \in e : \delta_i = 0\}$, respectively. Correspondingly, the complete nets on top and bottom dies are $\tilde{e}_i^+ = e_i^+ \cup \{t_i\}$ and $\tilde{e}_i^- = e_i^- \cup \{t_i\}$, respectively, including HBTs. The die-to-die (D2D) wirelength [Hu+22] of net $e \in E$ is defined as follows.

Definition 3 (D2D Net Wirelength)**.** *Given partition $\boldsymbol{\delta}$, the die-to-die (D2D) wirelength of net e is defined by $W_e = W_{\tilde{e}_i^+} + W_{\tilde{e}_i^-}$. More specifically, we have*

$$\begin{aligned} W_{\tilde{e}_i^+} &= \max_{c_j \in \tilde{e}_i^+} x_j - \min_{c_j \in \tilde{e}_i^+} x_j + \max_{c_j \in \tilde{e}_i^+} y_j - \min_{c_j \in \tilde{e}_i^+} y_j, \\ W_{\tilde{e}_i^-} &= \max_{c_j \in \tilde{e}_i^-} x_j - \min_{c_j \in \tilde{e}_i^-} x_j + \max_{c_j \in \tilde{e}_i^-} y_j - \min_{c_j \in \tilde{e}_i^-} y_j. \end{aligned} \tag{5.5}$$

If $C_{e_i}(\boldsymbol{\delta}) = 0$, it degrades to the ordinary net HPWL without HBT considered.

The D2D net wirelength in Definition 3 simply sums up the half-perimeter wirelength on two dies, demonstrating equivalence to $p_{\tilde{e}_i^+}(\boldsymbol{x}) + p_{\tilde{e}_i^-}(\boldsymbol{x}) + p_{\tilde{e}_i^+}(\boldsymbol{y}) + p_{\tilde{e}_i^-}(\boldsymbol{y})$. Since the center point of HBT t_i is included, W_e is a function of node locations $\boldsymbol{x}, \boldsymbol{y}$, HBT locations $\boldsymbol{x}', \boldsymbol{y}'$, and partition $\boldsymbol{\delta}$. Our problem is formulated as follows.

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\delta},\boldsymbol{x}',\boldsymbol{y}'} \quad & \sum_{e \in E} W_e(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\delta}, \boldsymbol{x}', \boldsymbol{y}') \\ \text{s.t.} \quad & \sum_{i=1}^{n} \delta_i a_i^+ \le a_{\text{req}}^+, \\ & \sum_{i=1}^{n} (1 - \delta_i) a_i^- \le a_{\text{req}}^-, \\ & \text{legality constraints}, \end{aligned} \tag{5.6}$$

where a_i^+, a_i^- stand for the node area of c_i on the top and bottom die, respectively. The area requirements are set to $a_{\text{req}}^+, a_{\text{req}}^-$ correspondingly. Besides of the legality constraints of standard cells, all HBTs have a specific legality rule that the distance between each other is lower bounded. It worth mentioning that HBTs are on the top-most metal layer and thus would not occupy any placement resources on both dies.

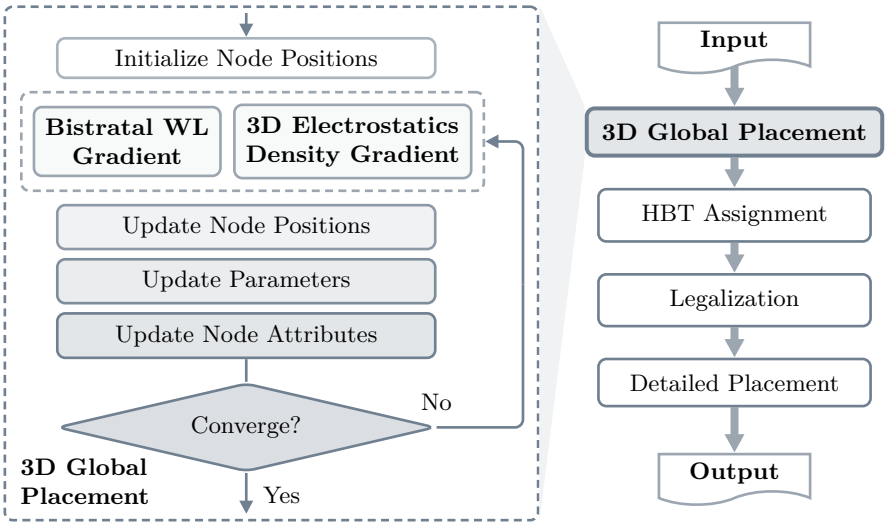


Figure 5.1: *The overall placement flow of our framework.*

5.3 Overall Placement Flow

The overall placement flow of our proposed framework is illustrated in Figure 5.1. We adopt a 3D analytical global placement to find node locations with three dimensions. After global placement, we assign HBTs and legalize all nodes including HBTs. At last, we perform detailed placement on each die to further refine the solution. The optimized circuit placement results will be output after detailed placement. Note that we do not apply 2D placement after 3D global placement and HBT assignment, as we are confident enough of our proposed 3D global placement which effectively handles partitioning and planar placement together.

5.3.1 Global Placement

In 3D placement, we assign coordinates $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R}^n$ to all nodes. Given the top die $[x^+_{\min}, x^+_{\max}] \times [y^+_{\min}, y^+_{\max}]$ and the bottom die $[x^-_{\min}, x^-_{\max}] \times [y^-_{\min}, y^-_{\max}]$, we have to make a necessary and realistic assumption that they differ very little so that the entire placement region is well-defined and the 3D placement framework makes sense under this scenario.

Assumption 1. *The die sizes of two dies are almost the same. Specifically, we have die width* $x^+_{\min} = x^-_{\min} = 0$, $x^+_{\max} = x^-_{\max}$, *and die height* $y^+_{\min} = y^-_{\min} = 0$, $\left|\frac{y^+_{\max}}{y^-_{\max}} - 1\right| < \epsilon$, *where* $\epsilon > 0$ *is a small tolerance.*

Under Assumption 1, our 3D global placement region is set to a cuboid $\Omega = [0, x^+_{\max}] \times [0, y^+_{\max}] \times [0, z_{\max}]$ by default, with a properly determined depth $z_{\max}$. For each node $c_i \in V$, along with its width and height provided by the input files, it will also be assigned a unified depth d.

Different from the 2D cases, the partition values $\boldsymbol{\delta}$ are restricted to take very discrete values in 3D placement to determine node partition. More specifically, $\boldsymbol{\delta}$ must be constrained to take binary values in $\{0, 1\}^n$ in our placement problem, described in Section 5.2.2, so that each node $c_i \in V$ has

an assigned partition indicator. We equally split the placement cuboid Ω into two parts by the plane $z = \frac{1}{2}z_{\max}$, each of which represents a die:

$$\begin{aligned} \Omega^+ &= [0, x^+_{\max}] \times [0, y^+_{\max}] \times \left[\tfrac{z_{\max}}{2}, z_{\max}\right] \\ \Omega^- &= [0, x^+_{\max}] \times [0, y^+_{\max}] \times \left[0, \tfrac{z_{\max}}{2}\right]. \end{aligned} \tag{5.7}$$

The unified node depth is $d = \frac{1}{2}z_{\max}$. Ideally, we expect every node $c_i \in V$ to be placed inside either the top part Ω^+ or the bottom part Ω^- at the end of 3D global placement. Note that every node should not be placed out of boundary, therefore z_i, which stands for the corner point coordinate of node c_i, should take values within interval $[0, \frac{1}{2}z_{\max}]$. We determine the *tentative* node partition $\boldsymbol{\delta}$ as a function of z coordinates $P(\boldsymbol{z})$, by rounding the normalized value $\frac{2}{z_{\max}}\boldsymbol{z}$ at every iteration, *i.e.*, we have

$$\delta_i = \left\lceil \frac{2z_i}{z_{\max}} - \frac{1}{2} \right\rceil, \tag{5.8}$$

for every $c_i \in V$.

An example of partition mapping $\boldsymbol{\delta} = P(\boldsymbol{z})$ is depicted in Figure 5.2(a). Node c_i is partitioned to the bottom die, *i.e.*, $\delta_i = 0$ as its corner coordinate $z_i < \frac{1}{4}z_{\max}$. The other node c_j in Figure 5.2(a) is partitioned to the top die, *i.e.*, $\delta_i = 1$ as its corner coordinate $z_j > \frac{1}{4}z_{\max}$. The exact value of the cuboid depth $z_{\max}$ should be determined properly to avoid ill-condition in numerical optimization.

We manually set the bin size of z dimension to the mean of bin sizes of the other two dimensions. More specifically, suppose the placement region is uniformly decomposed into $N_x \times N_y \times N_z$ grids, then we set

$$z_{\max} = \frac{N_z}{2}\left(\frac{x_{\max}}{N_x} + \frac{y_{\max}}{N_y}\right) \tag{5.9}$$

in our analytical placement.

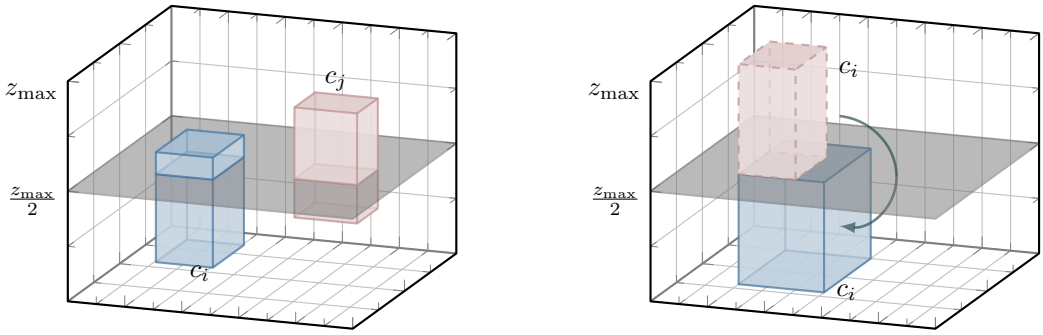
Heterogeneous Technologies. Different from ordinary analytical placement, we have to face a challenge of heterogeneous technologies that the node attributes including node sizes and pin offset values are different on the two dies.

Assume that each node $c_i \in V$ has width w_i^+ and height h_i^+ on the top die and w_i^-, h_i^- on the bottom die. At each iteration of 3D global placement, we should determine the exact node size for every node according to tentative partition $\boldsymbol{\delta} = P(\boldsymbol{z})$. More specifically, if the tentative partition $\delta_i = 1$, w_i^+, h_i^+ will be adopted for node c_i, otherwise it will use w_i^-, h_i^-. In other words, the planar node size for node $c_i \in V$ is calculated as

$$\begin{aligned} w_i &= \delta_i w_i^+ + (1 - \delta_i) w_i^-, \\ h_i &= \delta_i h_i^+ + (1 - \delta_i) h_i^- \end{aligned} \tag{5.10}$$

where the tentative partition δ_i determined by Equation (5.8) is a binary value. The node depth remains $d = \frac{1}{2}z_{\max}$ in the entire process of 3D global placement.

In addition to the node size, we also have two sets of pin offset values, although they are ignored for simplicity in previous wirelength notations. Denote all pins by $P = \{p_1, \cdots, p_l\}$, and P_i is the set of all pins on the node $c_i \in V$. Now, let $\boldsymbol{x}_{\text{offset}}, \boldsymbol{y}_{\text{offset}}, \boldsymbol{z}_{\text{offset}} \in \mathbb{R}^l$ be the pin offset vectors on three



(a) **(b)**

Figure 5.2: *The partition mapping $P(\boldsymbol{z}) : [0, z_{\max}] \to \{0, 1\}$ and the update of node attributes for heterogeneous technologies. (a) At one iteration during 3D global placement, node c_i has* tentative *partition $\delta_i = 0$ indicating the bottom die, while node c_j with $\delta_j = 1$ is assigned to the top die. (b) The node size and pin offset values of node $c_i \in V$ will change if moved to the other die.*

dimensions. For any $p_j \in P_i$, we have

$$\begin{aligned} x_{\text{offset},j} &= \delta_i x^+_{\text{offset},j} + (1 - \delta_i) x^-_{\text{offset},j}, \\ y_{\text{offset},j} &= \delta_i y^+_{\text{offset},j} + (1 - \delta_i) y^-_{\text{offset},j}, \end{aligned} \tag{5.11}$$

and $z_{\text{offset},j} = \frac{1}{4} z_{\max}$ is fixed. In other words, the pin offset values of every pin is determined by the tentative partition of the node it belongs to. Besides, we have a fact that, for any $p_j \in P_i$, node c_i's tentative partition $\delta_i = 1$ if and only if pin p_j is on the top part Ω^+: $z_i + z_{\text{offset},j} \geq \frac{z_{\max}}{2}$.

In accordance with Equation (5.10) and Equation (5.11), we update the *node attributes* including node size and pin offset at every iteration during 3D global placement. An example of updating node attributes is illustrated in Figure 5.2(b) where node c_i is moved from $z_i = \frac{1}{2} z_{\max}$ to $z_i = 0$.

Electrostatics-Based 3D Density. As mentioned and discussed in Section 5.1.1, eDensity [Lu+15a] is the state-of-the-art academic density model which analogizes every node c_i to a positive electric charge q_i. It expects an electric equilibrium so that movable objects can be evened out to reduce the overall node overlap. Extending the density model in [Lu+15a], ePlace-3D [Lu+16] computes the potential map by solving the 3D Possion's equation under Neumann boundary condition,

$$\begin{aligned} \Delta \phi &= -\rho, \quad \text{in } \Omega \\ \hat{\boldsymbol{n}} \cdot \nabla \phi &= 0, \quad \text{on } \partial\Omega, \end{aligned} \tag{5.12}$$

where $\rho = \rho(x, y, z)$ is the current density map in placement region $\Omega = [0, x_{\max}] \times [0, y_{\max}] \times [0, z_{\max}]$ computed using node locations. The second line in Equation (5.12) is the boundary condition specifying that the electric force on the boundary is zero.

Suppose the placement region Ω is uniformly decomposed into $N_x \times N_y \times N_z$ grids, the solution to Equation (5.12) under constraint $\int_\Omega \phi \, \mathrm{d}\Omega = 0$ is given by

$$\phi = \sum_{j,k,l} \frac{a_{jkl}}{\omega_j^2 + \omega_k^2 + \omega_l^2} \cos(\omega_j x) \cos(\omega_k y) \cos(\omega_l z), \tag{5.13}$$

where the tuple $(\omega_j, \omega_k, \omega_l) = (\frac{j\pi}{x_{\max}}, \frac{k\pi}{y_{\max}}, \frac{l\pi}{z_{\max}})$ stands for frequency indices. The density coefficients a_{jkl} is defined by

$$a_{jkl} = \frac{1}{N} \sum_{x,y,z} \rho \cos(\omega_j x) \cos(\omega_k y) \cos(\omega_l z). \tag{5.14}$$

where the denominator $N = N_x N_y N_z$ denotes the total number of bins. Note that the DC component of density map ρ has been removed, *i.e.*, $\int_\Omega \rho \, \mathrm{d}\Omega = 0$ is satisfied by removing $a_{000} = \frac{1}{N} \sum_{x,y,z} \rho(x, y, z)$ which equals to the average density of all bins. The electric field $\boldsymbol{E}(x, y, z) = (E_x, E_y, E_z)$ can be directly derived from Equation (5.13) by taking partial derivatives of ϕ,

$$\begin{aligned}
E_x &= \sum_{j,k,l} \frac{a_{jkl}\omega_j}{\omega_j^2 + \omega_k^2 + \omega_l^2} \sin(\omega_j x) \cos(\omega_k y) \cos(\omega_l z), \\
E_y &= \sum_{j,k,l} \frac{a_{jkl}\omega_k}{\omega_j^2 + \omega_k^2 + \omega_l^2} \cos(\omega_j x) \sin(\omega_k y) \cos(\omega_l z), \\
E_z &= \sum_{j,k,l} \frac{a_{jkl}\omega_l}{\omega_j^2 + \omega_k^2 + \omega_l^2} \cos(\omega_j x) \cos(\omega_k y) \sin(\omega_l z),
\end{aligned} \tag{5.15}$$

Equation (5.13) and Equation (5.15) are well-established in [Lu+16], demonstrating that these spectral equations can be solved efficiently using FFT with $O(N \log N)$ time complexity.

Different from the general scenarios in [Lu+16] where they may have multiple tiers, we only have two dies in our specific problem. To help the 3D electrostatic filed even out the standard cells to different dies, the node depth is set to $d = \frac{1}{2} z_{\max}$ by default, as mentioned above. Through the numerical optimization of 3D global placement, standard cells are expected to be roughly distributed within either Ω^+ or Ω^-, so that the tentative partition $\boldsymbol{\delta} = P(\boldsymbol{z})$ does not introduce significant wirelength degradation after 3D global placement.

5.3.2 HBT Assignment

During 3D global placement, we do *NOT* insert HBTs as any HBT is allowed to have overlap with standard cells. After 3D global placement, we first obtain a partition $\boldsymbol{\delta} = P(\boldsymbol{z}) \in \mathbb{Z}_2^n$ according to Equation (5.8). The convergence of global placement implies a very low overflow indicating that z_i should be close to either 0 or $\frac{1}{2} z_{\max}$ to determine the partition solution. Since the partition $\boldsymbol{\delta}$ and $\boldsymbol{x}, \boldsymbol{y}$ is already determined, we proceed to the 2D scenario with the top die layout and the bottom die layout. Every split net e should be assigned precisely one HBT.

Consider a split net $e \in E$. Ignoring pin offset values for simplicity, define x-dimension coordinates $x^+_{\text{low}} = \min_{c_i \in e^+} x_i$, $x^+_{\text{high}} = \max_{c_i \in e^+} x_i$ and vertical coordinates $y^+_{\text{low}}, y^+_{\text{high}}$ for the top partial net e^+, and similarly define corresponding variables for the bottom partial net e^-. Then, we denote the bounding box of partial nets e^+ and e^- by

$$\begin{aligned}
B_e^+ &= [x^+_{\text{low}}, x^+_{\text{high}}] \times [y^+_{\text{low}}, y^+_{\text{high}}], \\
B_e^- &= [x^-_{\text{low}}, x^-_{\text{high}}] \times [y^-_{\text{low}}, y^-_{\text{high}}],
\end{aligned} \tag{5.16}$$

respectively.

After 3D global placement, the $\boldsymbol{x}, \boldsymbol{y}$ coordinates and partition $\boldsymbol{\delta} = P(\boldsymbol{z})$ of nodes are already determined, and thus B_e^+ and B_e^- are determined for every split net e. As illustrated in Figure 5.3,

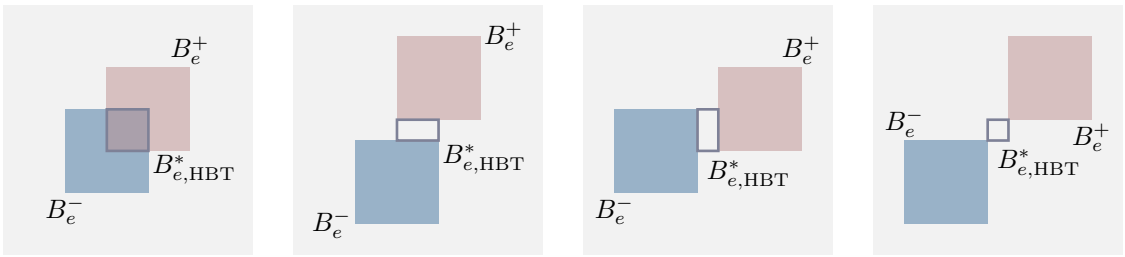


Figure 5.3: *The optimal region $B_{e,HBT}^*$ of an HBT for a split net $e \in E$ with cut $C_e(\boldsymbol{\delta}) = 1$ under several different scenarios. (a) The top net bounding box B_e^+ and the bottom net bounding box B_e^- overlap on both the x dimension and the y dimension. (b) B_e^+ and B_e^- overlap only on the x dimension. (c) B_e^+ and B_e^- overlap only on the y dimension. (d) B_e^+ and B_e^- have no overlap on both two dimensions.*

for any split net e, its HBT has a specific *optimal region*, *i.e.*, the net wirelength W_e is minimized only when its HBT is placed within this optimal region.

Theorem 10. *For a split net $e \in E$, the optimal region of its HBT is defined by $B_{e,HBT}^* = [x'_{low}, x'_{high}] \times [y'_{low}, y'_{high}]$ where*

$$\begin{aligned} x'_{low} &= \min\left\{\max\left\{x_{low}^+, x_{low}^-\right\}, \min\left\{x_{high}^+, x_{high}^-\right\}\right\}, \\ x'_{high} &= \max\left\{\max\left\{x_{low}^+, x_{low}^-\right\}, \min\left\{x_{high}^+, x_{high}^-\right\}\right\}, \end{aligned} \tag{5.17}$$

and y'_{low}, y'_{high} are defined similarly. Equivalently, coordinates x'_{low}, x'_{high} are the two median numbers of $x_{low}^+, x_{low}^-, x_{high}^+, x_{high}^-$ and the same for y'_{low}, y'_{high}.

Theorem 10 enlightens us that the total net wirelength will be minimized when every split net e has its HBT placed within the optimal region $B_{e,\mathrm{HBT}}^*$. Therefore, we intuitively assign an HBT $t(e) \in T$ for each split net such that the center point of t locates exactly at the center point of $B_{e,\mathrm{HBT}}^*$.

Note that after this *HBT assignment* step, it is likely that HBTs may overlap with each other, requiring a subsequent legalization process. To control the total number of HBTs and mitigate potential wirelength degradation caused by legalization, we carefully regulate the weight α in the objective function described in Definition 1. This enables us to mitigate wirelength degradation while minimizing the number of HBTs.

5.3.3 Legalization

After the partitioning $\boldsymbol{\delta} = P(\boldsymbol{z})$ and the HBT assignment, the mission of 3D global placement is completed. The rest is to legalize all nodes including HBTs and further refine the solution from 2D perspective. We legalize the standard cells on the top die and the bottom die separately with Tetris [Hil02] and Abacus [SSJ08a]. The HBTs are legalized similarly by treating them as ordinary standard cells with a specific terminal size.

Note that in our problem definition, HBTs share the same square size $w' \times w'$ and every pair of HBTs must satisfy the spacing constraint that the distance of boundaries should be no less than s'.

Hence, we pad every HBT to a square with size $w' + s'$ and legalize them as ordinary standard cells with row height $w' + s'$.

5.3.4 Detailed Placement

We further improve the total wirelength by applying ABCDPlace [Lin+20b] with several techniques including global swap [PVC05; Pop+14], independent set matching [Che+08], and local reordering [PVC05; Che+08] die by die. When we are performing detailed placement on one die, all other nodes on the other die and HBTs remain fixed. After the detailed placement of two dies, the optimal regions of HBTs may get affected. Therefore, we can continue to map HBTs to their updated optimal regions, followed by a new round of HBT legalization and detailed placement. While this process can be iterated infinitely, we find that only the initial few rounds yield significant benefits. Therefore, we perform one additional round of this process during the detailed placement.

5.4 Bistratal Wirelength Model

The analytical wirelength model is critical to the numerical optimization of Equation (5.1) in this problem. Previous works [HCB11; HBC13; Lu+16] use the 3D HPWL model defined in Definition 1 with the peak-to-peak function to describe the net wirelength. Chen et al. [Che+23a] propose MTWA model to consider heterogeneous technologies, but it is still based on 3D HPWL without considering the D2D wirelength. Note that $p_e(\boldsymbol{z})$ roughly reflects the cut size of net e and does not contribute to the planar net wirelength. The plain HPWL $\tilde{W}_e$ is defined as follows such that $W_e(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = \tilde{W}_e(\boldsymbol{x}, \boldsymbol{y}) + \alpha p_e(\boldsymbol{z})$.

Definition 4 (Plain HPWL)**.** *Given node positions $\boldsymbol{x}, \boldsymbol{y}$, the plain HPWL of any net $e \in E$ is given by*

$$\tilde{W}_e(\boldsymbol{x}, \boldsymbol{y}) = \max_{c_i \in e} x_i - \min_{c_i \in e} x_i + \max_{c_i \in e} y_i - \min_{c_i \in e} y_i, \tag{5.18}$$

which does not care node position $\boldsymbol{z}$ at all.

Obviously, Equation (5.18) in the above definition is equivalent to the separable representation $\tilde{W}_e(\boldsymbol{x}, \boldsymbol{y}) = p_e(\boldsymbol{x}) + p_e(\boldsymbol{y})$ using the peak-to-peak function defined in Equation (5.2).

Unfortunately, 3D HPWL model in Definition 1 based on the plain HPWL is *inaccurate* as the exact wirelength defined in Definition 3 and Equation (5.5) sums up the HPWL on the top die and bottom die. Equation (5.18) only considers the entire bounding box with the top die and the bottom die together, neglecting the pin partition and the potential presence of HBTs. Additionally, the conventional 3D HPWL wirelength model is *NOT* able to capture the wirelength variation resulting from different node partition.

Consider a net $e \in E$ connecting four pins p_1, p_2, p_3, p_4. Fix all planar locations of these pins and tentative partition of p_2, p_3, p_4. Figure 5.4(a) shows e^+ and e^- when p_3 is on the bottom die and the corresponding HBT is placed optimally. It is clear that the total wirelength of net e is $W_e = \tilde{W}_{\tilde{e}^+} + \tilde{W}_{\tilde{e}^-}$ which exactly equals to the *plain* HPWL of the entire net e. By contrast, Figure 5.4(b) shows the case when p_3 is on the top die. The HBT with the same coordinates preserves optimality, but the true wirelength $W_e = \tilde{W}_{\tilde{e}^+} + \tilde{W}_{\tilde{e}^-}$ is larger than the *plain* HPWL of net e.

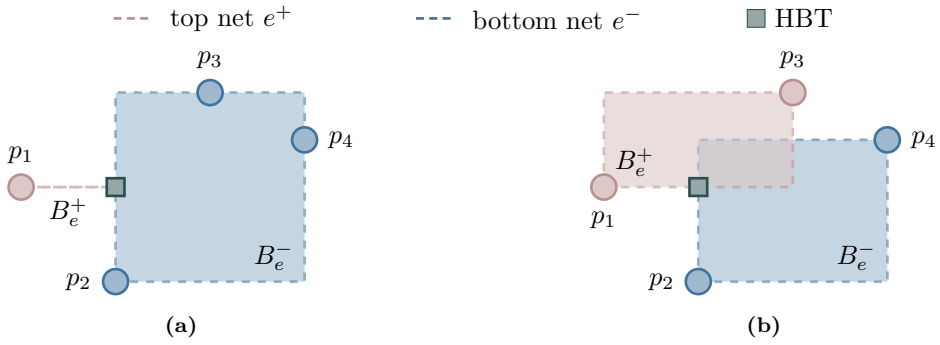


Figure 5.4: *An example where changing partition of one pin does not affect the net bounding box but increases the exact net wirelength. (a) The exact wirelength equals to the HPWL of the entire net. (b) The exact wirelength is strictly larger than the HPWL of the entire net.*

Theorem 11. *Given any partition $\boldsymbol{\delta}$ and any net $e \in E$, let $p_e(\boldsymbol{u}) = \max_{c_i \in e} u_i - \min_{c_i \in e} u_i$ be the peak-to-peak function defined in Equation* (5.2)*. Then, we always have*

$$p_e \leq \min_{x'} W_{e_x}(x') \leq 2p_e, \tag{5.19}$$

where $W_{e_x}(x')$ is the x-dimension part of the exact net wirelength defined in Equation (5.5) *with HBT coordinate x' under tentative partition $\boldsymbol{\delta}$.*

The equality of the left part of Equation (5.19) holds if and only if B_e^+ and B_e^- defined in Equation (5.16) has no overlap on the x dimension. The equality of the right part holds if and only if B_e^+ and B_e^- are the same on the x dimension. The conclusion on the y dimension can be similarly established. We will give a more detailed representation of $\min_{x'} W_{e_x}(x')$ in Theorem 12.

Corollary 4. *Given any partition $\boldsymbol{\delta}$ and any net $e \in E$, let the ordinary plain HPWL be $\tilde{W}_e$ defined in Definition 4. Then, we always have*

$$\tilde{W}_e \leq \min_{x',y'} W_e(x',y') \leq 2\tilde{W}_e, \tag{5.20}$$

where $W_e(\boldsymbol{\delta}, x', y')$ is the exact net wirelength defined in Equation (5.5) *with HBT coordinate (x', y').*

The equality of the left part of Equation (5.20) holds if and only if B_e^+ and B_e^- defined in Equation (5.16) has no overlap on both x and y dimensions. The equality of the right part holds if and only if B_e^+ and B_e^- are the same. Corollary 4 indicates that the HPWL model used in previous works [HCB11; HBC13; Lu+16] is just a lower bound of the exact bistratal wirelength in our problem. Apparently, optimizing $\tilde{W}_e$ does not necessarily benefit the exact wirelength as the error bound may get as large as the lower bound, according to Equation (5.20). We will give a precise representation of $\min_{x',y'} W_e(x',y')$ for every net e in Theorem 12.

We propose a novel *bistratal* wirelength model that can handle planar coordinates and partitioning together. Instead of optimizin the plain HPWL $\tilde{W}_e$, we try to minimize $\min_{\boldsymbol{x}',\boldsymbol{y}'} W_e(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\delta}, \boldsymbol{x}', \boldsymbol{y}')$ at every iteration according to the tentative partition. Besides of the wirelength estimation, the

computation of gradients is more critical to the numerical optimization process. In this section, we will discuss the proposed model theoretically in detail.

5.4.1 Wirelength Objective

In the forward pass of numerical optimization [Lu+15a; Lin+19], we calculate the exact or approximated wirelength. Different from 3D HPWL in Definition 1, we must consider partition for more precise wirelength estimation.

According to Corollary 4, we should approximate the exact wirelength W_e defined in Equation (5.5) as precisely as possible. Considering that HBTs are not inserted in the 3D global placement as the tentative partition $\boldsymbol{\delta}$ may vary at every iteration, we assume that each split net is assigned a dummy HBT placed within its optimal region according to the tentative partition. In other words, we target at optimizing $\min_{\boldsymbol{x}',\boldsymbol{y}'} W_e(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\delta},\boldsymbol{x}',\boldsymbol{y}')$ where the tentative partition $\boldsymbol{\delta} = P(\boldsymbol{z})$ is updated at every iteration. The following theorem reveals the explicit representation of our wirelength forward computation without any HBT inserted.

Theorem 12. *The minimal precise net wirelength on the x dimension with respect to the HBT coordinate x' of net e is given by*

$$\min_{x'} W_{e_x}(x') = \max\left\{p_e, p_{e^+} + p_{e^-}\right\}, \tag{5.21}$$

as a function of node positions $(\boldsymbol{x})$ under partition $\boldsymbol{\delta}$, where the peak-to-peak function p_e is defined by $p_e(\boldsymbol{u}) = \max_{c_i \in e} u_i - \min_{c_i \in e} u_i$ for any $\boldsymbol{u}$.

Theorem 12 gives an accurate estimation of the minimum exact net wirelength on the x dimension for split nets at every iteration during 3D global placement. Note that the right-hand side of Equation (5.21) also indicates the exact wirelength for any non-split net e as either p_{e^+} or p_{e^+} is zero. The corresponding theorem on the y dimension can be similarly established.

Given any partition $\boldsymbol{\delta}$ and any split net $e \in E$, let $B_e^+ = [x_{\text{low}}^+, x_{\text{high}}^+] \times [y_{\text{low}}^+, y_{\text{high}}^+]$ and $B_e^- = [x_{\text{low}}^-, x_{\text{high}}^-] \times [y_{\text{low}}^-, y_{\text{high}}^-]$ be the bounding boxes of partial nets e^+, e^-, respectively, defined in Equation (5.16). Define

$$W_{ex} = \begin{cases} \max\limits_{c_i \in e} x_i - \min\limits_{c_i \in e} x_i, & \text{if } x_{\text{high}} \le x_{\text{low}}, \\ x_{\text{high}}^+ - x_{\text{low}}^+ + x_{\text{high}}^- - x_{\text{low}}^-, & \text{otherwise.} \end{cases} \tag{5.22}$$

where $x_{\text{low}} = \max\{x_{\text{low}}^+, x_{\text{low}}^-\}, x_{\text{high}} = \min\{x_{\text{high}}^+, x_{\text{high}}^-\}$, and similary define W_{ey}. Then, the minimal precise net wirelength considering both x, y dimensions with respect to the HBT coordinates (x', y') of net $e \in E$ defined in Theorem 12 is equivalent to

$$\min_{x',y'} W_e(x', y') = W_{ex} + W_{ey}, \tag{5.23}$$

More intuitively, Equation (5.23) first checks whether the boxes B_e^+ and B_e^- overlap. If they overlap on one dimension, we optimize the HPWL of the top partial net e^+ and the bottom partial

net e^- on this dimension separately, as we have

$$\begin{aligned} x^+_{\text{high}} - x^+_{\text{low}} &= p_{e^+}(\boldsymbol{x}) = \max_{c_i \in e^+} x_i - \min_{c_i \in e^+} x_i, \\ x^-_{\text{high}} - x^-_{\text{low}} &= p_{e^-}(\boldsymbol{x}) = \max_{c_i \in e^-} x_i - \min_{c_i \in e^-} x_i, \end{aligned} \tag{5.24}$$

otherwise the target degrades to the ordinary HPWL function $\tilde{W}_e$ on this dimension.

For a non-split net $e \in E$ with $C_e(\boldsymbol{\delta}) = 0$, *i.e.*, it is completely within either the top or the bottom die, we treat it as an ordinary 2D net and evaluate its ordinary plain wirelength $\tilde{W}_e$ with Equation (5.18). Then, we propose the *bistratal wirelength* (BiHPWL) as follows.

Definition 5 (Bistratal Wirelength)**.** *Given 3D node position $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$, the bistratal half-perimeter wirelength of any net e is defined as*

$$W_{e,Bi}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = \max\{p_e(\boldsymbol{x}), p_{e^+}(\boldsymbol{x}) + p_{e^-}(\boldsymbol{x})\} + \max\{p_e(\boldsymbol{y}), p_{e^+}(\boldsymbol{y}) + p_{e^-}(\boldsymbol{y})\} \tag{5.25}$$

where the peak-to-peak function $p_e(\cdot)$ is defined by $p_e(\boldsymbol{u}) = \max_{c_i \in e} u_i - \min_{c_i \in e} u_i$ for any $\boldsymbol{u}$. The partial nets $e^+(\boldsymbol{\delta})$ and $e^-(\boldsymbol{\delta})$ are determined by the tentative partition $\boldsymbol{\delta} = P(\boldsymbol{z})$.

Definition 5 gives a much accurate wirelength estimation in our probelm. Combining the regularization of cut size, In our 3D global placement, we use

$$W(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) := \sum_{e \in E} W_{e,\text{Bi}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) + \alpha \sum_{e \in E} p_e(\boldsymbol{z}) \tag{5.26}$$

as the wirelength objective where the bistratal net wirelength $W_{e,\text{Bi}}$ is defined by Equation (5.25). Note that $W_{e,\text{Bi}}$ is also a function of $\boldsymbol{z}$ as the tentative partition $\boldsymbol{\delta}$ at every iteration is determined by $\boldsymbol{z}$. The second term with α weight is integrated to limit the total number of HBTs as we always expect fewer HBTs if possible. Moreover, a large number of HBTs would degrade the solution quality after legalization.

Optimizing Equation (5.26) resolves the issue that 3D HPWL approximates the true wirelength poorly when the top box B^+_e and the bottom box B^-_e overlap, illustrated in Figure 5.4(b). However, the objective in Equation (5.26) is highly non-differentiable. Therefore, we should establish the gradient approximation in detail to enable numerical optimization of 3D global placement. In the following of this section, we will discuss the gradient computation including the subgradient approximation to the planar gradients and the finite difference approximation to the depth gradient.

5.4.2 Gradient Computation

The optimization of Equation (5.26) itself is difficult as it is non-differentiable and even discontinuous with respect to $\boldsymbol{z}$. In this subsection, we will discuss our proposed strategy to find the "*gradients*" that percept the objective change with respect to variables.

Since the representations in Equation (5.22) are always in a peak-to-peak form, we use the weighted-average model [HCB11; HBC13] in Equation (5.3) to approximate them, so that W_e in Equation (5.21) is *differentiable* where $x_{\text{high}} \neq x_{\text{low}}$ and $y_{\text{high}} \neq y_{\text{low}}$ when calculating gradients. It is

straght-forward to derive the closed-form representation of gradients of the WA model[HCB11] described in Equation (5.3),

$$\frac{\partial p_{e,\mathrm{WA}}}{\partial u_i} = \frac{\mathrm{e}^{\frac{u_i}{\gamma}}\left(\gamma + u_i - S_{\max}\right)}{\gamma \sum_{c_i \in e} \mathrm{e}^{\frac{u_i}{\gamma}}} - \frac{\mathrm{e}^{-\frac{u_i}{\gamma}}\left(\gamma + S_{\min} - u_i\right)}{\gamma \sum_{c_i \in e} \mathrm{e}^{-\frac{u_i}{\gamma}}}, \tag{5.27}$$

where the smooth maximum $S_{\max} = S_{\max}(\boldsymbol{u})$ and the smooth minimum $S_{\min} = S_{\min}(\boldsymbol{u})$ are defined by

$$S_{\max} = \frac{\sum_{c_i \in e} u_i \mathrm{e}^{\frac{1}{\gamma}u_i}}{\sum_{c_i \in e} \mathrm{e}^{\frac{1}{\gamma}u_i}}, \quad S_{\min} = \frac{\sum_{c_i \in e} u_i \mathrm{e}^{-\frac{1}{\gamma}u_i}}{\sum_{c_i \in e} \mathrm{e}^{-\frac{1}{\gamma}u_i}}, \tag{5.28}$$

such that $p_{e,\mathrm{WA}} = S_{\max} - S_{\min}$. The variable $\boldsymbol{u}$ can be $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ to derive the detailed gradients of the smooth peak-to-peak on corresponding dimensions. More details of differentiable approximations are discussed in [NDS01; HCB11; HBC13; Lia+23].

Adaptive Planar Gradients. In the numerical optimization, we are supposed to derive the "gradients" of Equation (5.26) with respect to coordinates $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$. The gradients w.r.t planar coordinates $\boldsymbol{x}, \boldsymbol{y}$ guide the optimizer to find optimal placement on each die, while the gradients w.r.t $\boldsymbol{z}$ handle the partition correspondingly. It is clear that the planar gradients are determined by $\nabla_{\boldsymbol{x}} W_{e,\mathrm{Bi}}$ and $\nabla_{\boldsymbol{y}} W_{e,\mathrm{Bi}}$. Unfortunately, $W_{e,\mathrm{Bi}}$ in Equation (5.25) is non-differentiable, forcing us to consider *subgradients* instead.

Without loss of generality, we focus on the x dimension. Consider function set $\mathcal{F} = \{p_e, p_{e^+} + p_{e^-}\}$ for a given tentative partition, then the x-dimension part of wirelength $W_{e,\mathrm{Bi}}$ is $W_{e_x,\mathrm{Bi}}(\boldsymbol{x}) = \max_{f \in \mathcal{F}} f(\boldsymbol{x})$. The corresponding active function set is

$$\mathcal{I}(\boldsymbol{x}) = \{f \in \mathcal{F} : f(\boldsymbol{x}) = W_{e_x,\mathrm{Bi}}(\boldsymbol{x})\}. \tag{5.29}$$

According to the subgradient calculus rule, we know that the subdifferential of $W_{e,\mathrm{Bi}}$ is a convex hull

$$\partial W_{e_x,\mathrm{Bi}}(\boldsymbol{x}) = \mathrm{conv} \bigcup_{f \in \mathcal{I}(\boldsymbol{x})} \partial f(\boldsymbol{x}). \tag{5.30}$$

We expect to legitimately take one subgradient $g \in \partial W_{e_x,\mathrm{Bi}}(\boldsymbol{x})$ for optimization.

A non-split net is trivial as $W_{e_x,\mathrm{Bi}}(\boldsymbol{x})$ degrades to $p_e(\boldsymbol{x})$ directly. Consider a split net $e \in E$. When B_e^+ and B_e^- have overlap on x dimension, *i.e.* $p_{e^+}(\boldsymbol{x}) + p_{e^-}(\boldsymbol{x}) > p_e(\boldsymbol{x})$, $p_{e^+}(\boldsymbol{x}) + p_e^-(\boldsymbol{x}) \in \mathcal{I}(\boldsymbol{x})$ is active in Equation (5.29) and we have $W_{e_x,\mathrm{Bi}}(\boldsymbol{x}) = p_{e^+}(\boldsymbol{x}) + p_{e^-}(\boldsymbol{x})$. According to Equation (5.30), it is straight-forward to take any subgradient in $\partial p_{e^+}(\boldsymbol{x}) + \partial p_{e^-}(\boldsymbol{x})$ for numerical optimization. Empirically, differentiable approximations of p_e may be preferred to work with smooth optimizers, and thus we take $\nabla_{\boldsymbol{x}} p_{e^+,\mathrm{WA}} + \nabla_{\boldsymbol{x}} p_{e^-,\mathrm{WA}}$ as the "gradient" $\nabla_{\boldsymbol{x}} W_{e_x,\mathrm{Bi}}$, where we leverage the weighted-average model $p_{e,\mathrm{WA}}$ [HCB11; HBC13] defined in Equation (5.3). When B_e^+ and B_e^- do not overlap on x dimension, *i.e.*, $p_{e^+}(\boldsymbol{x}) + p_{e^-}(\boldsymbol{x}) < p_e(\boldsymbol{x})$, $p_e(\boldsymbol{x})$ is active in Equation (5.29), so we have $W_{e_x,\mathrm{Bi}}(\boldsymbol{x}) = p_e(\boldsymbol{x})$ and treat e as a non-split net, then apply the approximation $p_{e,\mathrm{WA}}$. When $\mathcal{I}(\boldsymbol{x})$ is not a singleton, *i.e.*, $p_{e^+}(\boldsymbol{x}) + p_{e^-}(\boldsymbol{x}) = p_e(\boldsymbol{x})$, we can take any element in the convex hull in Equation (5.30). Through this way, we define the "gradient" $\nabla W_{e,\mathrm{Bi}}$.

Definition 6 (Planar Gradient)**.** *Consider the bistratal wirelength $W_{e,Bi}$. The planar gradient*

$\nabla_{\boldsymbol{x}} W_{e,Bi} = \boldsymbol{g}$ *is defined as follows*

$$\boldsymbol{g} = \begin{cases} \nabla p_{e^+,WA} + \nabla p_{e^-,WA}, & \text{if } p_{e^+} + p_{e^-} > p_e, \\ \nabla p_{e,WA}, & \text{otherwise.} \end{cases} \tag{5.31}$$

which is an approximation of a subgradient, where the gradient of $p_{e,WA}$ *is given by Equation* (5.27).

The gradient $\nabla_{\boldsymbol{y}} W_{e,\mathrm{Bi}}$ can be defined similarly. Note that we still use $\nabla W_{e,\mathrm{Bi}}$ to denote such a subgradient approximation in Equation (5.31) although $W_{e,\mathrm{Bi}}$ itself is non-differentiable.

We consider Equation (5.31) to be the *adaptive* planar gradients w.r.t. $\boldsymbol{x}, \boldsymbol{y}$ coordinates. The term "adaptive" is named after the overlap illustrated in Figure 5.4. More specifically, we check whether B_e^+ and B_e^- overlap on x (and y) dimensions under tentative $\boldsymbol{\delta}$ for every net e at every global placement iteration. If they overlap on the x (or y) dimension, we have $p_{e^+} + p_{e^-} > p_e$ and use the first representation of $\nabla W_{e,\mathrm{Bi}}$ in Equation (5.31) and the second otherwise. Equation (5.31) is applied in our numerical optimization during the 3D global placement. With no doubt, it takes into account the physical information of pin coordinates on both dies, making it much more accurate than the 3D HPWL model.

Finite Difference Approximation of Depth Gradients. In addition to the planar gradients w.r.t. $\boldsymbol{x}$ and $\boldsymbol{y}$, we are also supposed to derive how to correctly define "gradients" w.r.t. $\boldsymbol{z}$ which is far more tricky. Finding a way to optimize $\boldsymbol{z}$ is critical to the entire optimization, as it directly determines the quality of partition.

The density gradient $\nabla_{\boldsymbol{z}} D(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ drives placer to separate nodes with depth $\frac{1}{2} z_{\max}$ to be distributed on two dies so that we can obtain a valid partition at last, neglecting wirelength optimization. The gradient $\sum_e \nabla_{\boldsymbol{z}} p_{e,\mathrm{WA}}(\boldsymbol{z})$ in Equation (5.26) with the weighted-average model [HBC13] tends to optimize the total cutsize of the design so that the total number of HBTs is limited, but there is no theoretical guarantee that a small cutsize would benefit the D2D wirelength. Hence, the most important task is to find how $W_{e,\mathrm{Bi}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ gets affected by $\boldsymbol{z}$ to evaluate the quality of partitioning. Considering that $W_{e,\mathrm{Bi}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ is even discontinuous with respect to $\boldsymbol{z}$, the gradient $\nabla_{\boldsymbol{z}} W_{e,\mathrm{Bi}}$ does not exist at all. To tackle this problem, we leverage *finite difference* to approximate the impact of $\boldsymbol{z}$ on the bistratal wirelength.

Finite difference [Tay15; BM60; Jor56; MT00] has been widely used in a large number of applications in numerical differentiation to approximate derivatives. We follow the definitions and notations in [MT00] and denote the *difference quotient* by

$$\underset{h}{\Delta} f(x) = \frac{f(x+h) - f(x)}{h} \tag{5.32}$$

using the Nörlund's operator $\underset{h}{\Delta}$ [MT00; N24] for any function f on $\mathbb{R}$ and $x, h \in \mathbb{R}$. In the classical infinitesimal calculus, the first-order derivative of f is defined by $\lim_{h \to 0} \underset{h}{\Delta} f(x)$ if f is differentiable. Both difference and derivative estimate how the function value would change with its variables, but derivative is in a continuous view while difference depends on the step size h.

Taking a net $e \in E$ and $c_i \in e$, consider the impact of z_i to the bistratal wirelength $W_{e,\mathrm{Bi}}$. For simplicity, we use $W_{e,\mathrm{Bi}}(z_i)$ to represent the bistratal wirelength of net $e \in E$ as a function of z_i and

fix all other variables. Given step size h, the difference quotient of $W_{e,\mathrm{Bi}}$ at z_i is

$$\underset{h}{\Delta} W_{e,\mathrm{Bi}}(z_i) = \frac{W_{e,\mathrm{Bi}}(z_i + h) - W_{e,\mathrm{Bi}}(z_i)}{h}, \tag{5.33}$$

combining both the *forward/advancing difference* ($h > 0$) and the *backward/receding difference* ($h < 0$). Since $W_{e,\mathrm{Bi}}$ is discontinuous with respect to variable z_i, the limit $\lim_{h\to 0} \underset{h}{\Delta} W_{e,\mathrm{Bi}}(z_i)$ does not exist. However, we could consider Equation (5.33) with a large h as we only have two dies. More specifically, we set $h = \frac{1}{4} z_{\max}$ if $\delta_i = P(z_i) = 0$ and $h = -\frac{1}{4} z_{\max}$ otherwise, so that the difference quotient in Equation (5.33) will always be non-zero. Providing that $W_{e,\mathrm{Bi}}(z_i)$ is a step function that only takes two possible values $W_{e,\mathrm{Bi}}(0)$ and $W_{e,\mathrm{Bi}}(\frac{1}{2} z_{\max})$, Equation (5.33) can be summarized as follows.

Definition 7 (Finite Difference Approximation)**.** *Consider the bistratal wirelength $W_{e,Bi}$. The finite difference approximation (FDA) $\nabla_{\boldsymbol{z}} W_{e,Bi} = \boldsymbol{g}$ is defined by*

$$\boldsymbol{g}_i = \underset{\frac{1}{4} z_{\max}}{\Delta} W_{e,Bi}(z_i) = \frac{4}{z_{\max}} \left(W_{e,Bi}\left(\frac{z_{\max}}{2}\right) - W_{e,Bi}(0) \right), \tag{5.34}$$

where $z_{\max}$ is the total depth of our placement region, defined in Equation (5.7)*.*

Equation (5.34) is intuitive that it actually evaluates the wirelength change when moving a pin to the other die. It provides a local view of benefits we can obtain when changing node partition. Note that we still use the term $\nabla_{\boldsymbol{z}} W_{e,\mathrm{Bi}}$ to denote the finite difference approximation in Definition 7, although $W_{e,\mathrm{Bi}}$ itself is non-differentiable.

From Equation (5.34), any node $c_i \in V$ accumulates depth gradients $\nabla_{\boldsymbol{z}} W_{e,\mathrm{Bi}}$ from all related nets e, therefore the finite difference approximation locally evaluates the impact of every node to the total circuit wirelength. We apply $\sum_e \nabla_{\boldsymbol{z}} W_{e,\mathrm{Bi}}$ in Equation (5.34) with cutsize gradient $\sum_e \nabla_{\boldsymbol{z}} p_{e,\mathrm{WA}}(\boldsymbol{z})$ and density gradient $\nabla_{\boldsymbol{z}} D(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ to numerical optimization in 3D global placement to obtain a good partition with an acceptable number of HBTs. Combining with the adaptive planar gradients in Definition 6, we have defined the detailed gradient computation of the proposed bistratal wirelength model.

5.5 Experimental Results

5.5.1 Experimental Setup

We conducted experiments on ICCAD 2022 contest benchmark suits [Hu+22]. The detailed design statistics are shown in Table 5.1. Note that each HBT in a specific design has a size of $w' \times w'$, and the minimum spacing s' on x and y directions between each pair of HBTs is also equal to w'. Movable macros are not included in the benchmark suits.

We implemented the proposed 3D analytical placement framework in `C++` and `CUDA` based on the open-source placer DreamPlace [Lin+19]. All the experiments were performed on a Linux machine with 20 Intel Xeon Silver 4210R cores (2.40GHz), 1 GeForce RTX 3090Ti graphics card, and 24 GB of main memory. We compared our framework with the state-of-the-art (SOTA) placers from the top-3

Table 5.1: *The statistics of the ICCAD 2022 Contest [Hu+22] Benchmark Suites where u^+, u^- stand for the utilization constraints on the top die and the bottom die, respectively. RH^+ and RH^- represent the row height on the top and bottom die. w' means the size of hybrid bonding terminals.*

Bench.	#Nodes	#Nets	#Pins	u^+	u^-	RH^+	RH^-	w'
case2	2735	2644	8118	0.70	0.75	176	252	100
case2h	2735	2644	8118	0.79	0.79	252	252	114
case3	44764	44360	142246	0.78	0.78	115	115	50
case3h	44764	44360	142246	0.68	0.78	92	115	46
case4	220845	220071	773551	0.66	0.70	92	115	62
case4h	220845	220071	773551	0.66	0.76	103	115	66

teams in ICCAD 2022 contest and recent work [Che+23a], and the reported results were evaluated by the official evaluator provided by the contest.

5.5.2 Comparison with SOTA Placers

Table 5.2 shows the experimental results of the top-3 teams, SOTA analytical 3D placer [Che+23a], and ours on the contest benchmark suites [Hu+22]. We compared the exact D2D wirelength (WL), the total number of hybrid bonding terminals (HBTs), and runtime of each case with the baselines in Table 5.2. The wirelength is evaluated using the provided official evaluator from the benchmark suites. For a fair comparison, we acquired their binary executable files and evaluated the end-to-end runtime of the baselines on our machine using their default settings.

It worths mentioning that the ICCAD 2022 Contest [Hu+22] evaluates WL as the final score. Hence, the contestants only target at optimizing WL and may not consider HBT costs explicitly. However, realistic requirements oftern expect to limit the HBT usage as well. Our framework explicitly considers the cutsize optimization as a scondary goal in the objective function in Definition 5 to find a "proper" cutsize, as simply reducing cutsize may also degrade the performance [PL22].

As illustrated in Table 5.2, our analytical 3D placement framework consistently obtained the best WL results for all the cases, demonstrating the significant advantage of our 3D placement paradigm with the dedicated bistratal wirelength model. Compared to the top-3 teams, our placer achieved 4.1%, 5.7%, and 7.2% shorter wirelength on average, respectively.

Thanks to the global optimization view of our 3D analytical approach, our placer utilized fewer HBTs and achieved better wirelength. Our framework reduced 52.3%, 21.1%, 2.0% number of HBTs on average compared to the top-3 contest winners. Our framework achieved up tp 49.2% HBT number reduction than the first place on the large cases, making our framework more competitive to reduce the hybrid bonding terminal fabrication cost for large designs in real scenarios. Leveraging the computation power of modern GPUs, our placer demonstrates better runtime scalability than the baselines, achieving 4.300× and 5.320× speedup over the first place and the second place for end-to-end placement, and achieving up to 2.925× speedup over the third place on the large cases.

We also compared our framework with the SOTA analytical 3D placer [Che+23a] on the same ICCAD 2022 benchmarks [Hu+22]. Chen et al. [Che+23a] proposed an MTWA wirelength model based on 3D HPWL in Definition 1, considering heterogeneous technologies with a weight factor α that correlates positively with net degrees. They aimed to guide the optimizer to split more

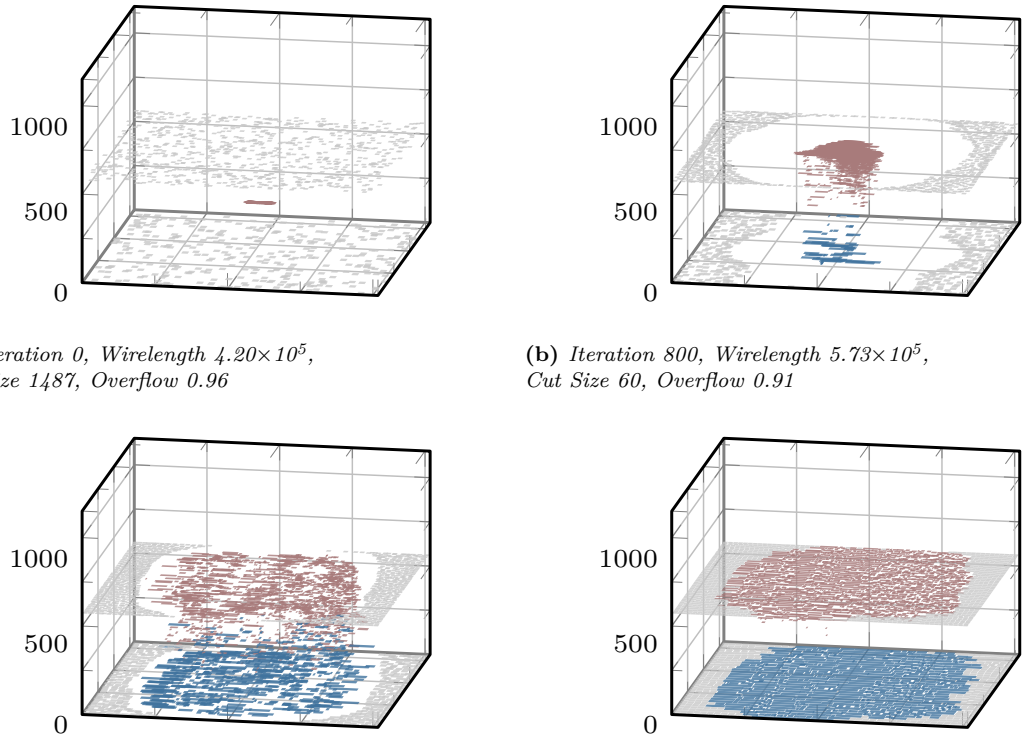
low-degree nets for wirelength reduction, which resulted in notable improvements compared to the first-place winner. However, their wirelength model in 3D analytical placement is still inaccurate and thus requires an additional 2D placement to refine node locations. Moreover, MTWA [Che+23a] is not directly partitioning-aware. The experimental results in Table 5.2 show that we achieve up to 3.4% wirelength improvement and 2.1% on average compared to [Che+23a]. Remarkably, we are confident enough of our placement framework in numerical optimization, and thus do not require a 2D placement to further refine node locations after 3D placement. In addition, we require 25% fewer HBTs and can efficiently accomplish the placement task with GPU resources. In modern VLSI design, the performance on large cases is most critical. We considered two large cases containing more than 220K standard cells in the ICCAD 2022 Contest benchmark suites [Hu+22]. As shown in Table 5.2, we significantly outperform the baseline by 1.8% and 2.5% wirelength improvement on the largest two cases `case4` and `case4h`, respectively, with more than 8× runtime acceleration, proving the scalability of our framework.

5.5.3 3D Global Placement Analysis

Our 3D analytical placement framework enables the simultaneous node partitioning and placement in the global placement stage, forming a larger solution space than previous separate partitioning and placement works [Cha+16; Pan+17; KCL18; Par+20]. Unlike previous 3D analytical placer [Lu+16] targets on multiple tiers and leverages subsequent 2D placement to refine the placement solution, our framework assigns the nodes to exact two dies and place them in a single run. Our 3D global placement is visualized in Figure 5.5.

In Figure 5.5, fillers, nodes on the top die, and nodes on the bottom die are denoted by gray, brown, and blue rectangles, respectively. The node depth is omitted for better visualization. All standard cells are randomly initialized around the center point of the design from a normal distribution, shown in Figure 5.5(a). Note that fillers are already inserted according to the given utilization requirements and uniformly initialized on two dies. During the 3D global placement, the optimizer tends to move nodes according to the gradients of wirelength (including cutsize with weight α) and density. The tentative partition $\boldsymbol{\delta}$ is updated at every intermediate iteration of global placement, shown in Figure 5.5(b) and Figure 5.5(c), until the convergence is detected. At last, the placer will find a 3D placement solution with optimized wirelength, shown in Figure 5.5(d). When the convergence is attained, most standard cells c_i with coordinate z_i satisfying $|\frac{2z_i}{z_{\max}} - \delta_i| < \epsilon$ for a sufficiently small positive number $\epsilon > 0$, implying that our framework is confident enough to partition every standard cell. The 3D global placement produces a solution with overflow 0.07, shown in Figure 5.5(d), therefore we apply the tentative partition at the 1769th iteration as the final partition $\boldsymbol{\delta}$ and proceed to the later steps including legalization and detailed placement.

In addition to the convergence visualization of our 3D global placement in Figure 5.5, we also plot the 2D placement of two dies in Figure 5.6, together with the hybrid bonding terminals. As shown in Figure 5.6(b), our HBT placement is sparse after legalization and detailed placement.



(a) *Iteration 0, Wirelength 4.20×10^5, Cut Size 1487, Overflow 0.96*

(b) *Iteration 800, Wirelength 5.73×10^5, Cut Size 60, Overflow 0.91*

(c) *Iteration 1200, Wirelength 1.48×10^6, Cut Size 652, Overflow 0.47*

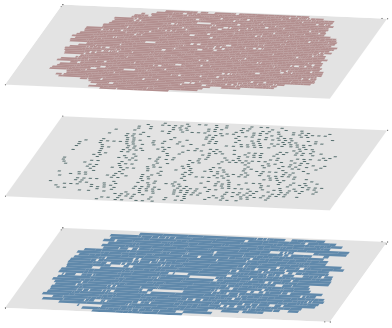
(d) *Iteration 1769, Wirelength 1.71×10^6, Cut Size 646, Overflow 0.07*

Figure 5.5: *3D global placement on* `case2` *with heterogeneous technologies. Fillers, nodes on the top die, and nodes on the bottom die are denoted by gray, brown, and blue rectangles, respectively. The node depth is omitted for better visualization. The nodes are initialized at the center point, and the fillers are randomly distributed on the two dies as shown in (a). The 3D density force combined with the wirelength force progressively drive all the nodes to the specific die, leading to a placement solution with almost perfect node partition as shown in (d).*

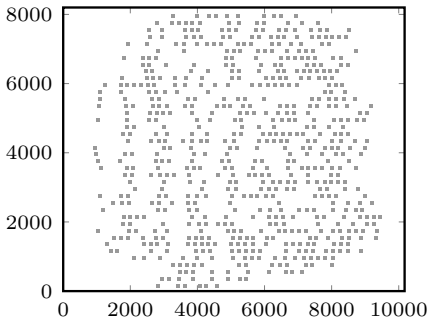
5.5.4 Ablation Studies on Wirelength Models

We evaluated the effectiveness of our proposed bistratal wirelength model by using different wirelength models in our framework on the ICCAD 2022 Contest Benchmarks [Hu+22]. The detailed experimental results are shown in Table 5.3.

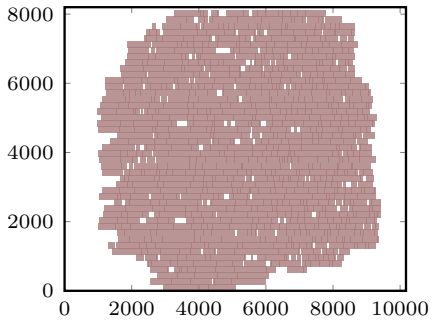
Plain HPWL stands for the conventional HPWL model $\tilde{W}_e$ defined in Definition 4. It is integrated in 3D HPWL adopted in many previous analytical placers [LSC13; HCB11; HBC13; Lu+16]. This wirelength model is very classical and has been proved to be effective in analytical 3D placement. Notably, the gradients of differentiable approximations to $\tilde{W}_e$ w.r.t. $\boldsymbol{z}$ only focus on optimization on cutsize. Hence, it achieves the best results of cutsize, with only 55.5% HBTs of ours, shown in Table 5.3. However, the wirelength reported by the evaluator is 16.9% larger than ours, as plain HPWL model could not comprehend the impact of partitioning on the exact D2D wirelength. We now validate the effectiveness of the adaptive planar gradient defined in Definition 6 and the finite difference approximation (FDA) of depth gradients in Definition 7.



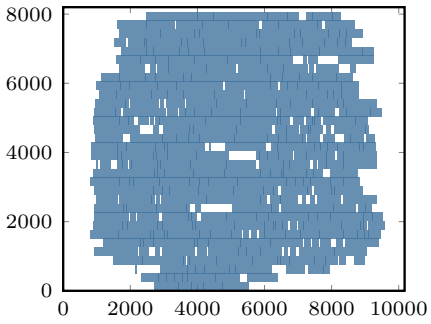
(a) *The general view of the solution to* `case2`



(b) *The hybrid bonding terminals of the solution to* `case2`



(c) *The top die layout of the solution to* `case2`



(d) *The bottom die layout of the solution to* `case2`

Figure 5.6: *The placement results of* `case2` *using our proposed method. (b) illustrates how hybrid bonding terminals are placed to connect cells on different dies. (c) and (d) plot the top die placement and the bottom die placement, respectively.*

BiHPWL in the second main column of Table 5.3 represents the bistratal wirelength in Definition 5 equipped with the adaptive planar gradient. "BiHPWL model without FDA" is equivalent to "plain HPWL with adaptive planar gradient" in terms of gradient computation. As shown in Table 5.3, the BiHPWL model without FDA achieves 3.5% wirelength improvements on average with little degradation of cutsize, compared to plain HPWL. It is intuitively rational as the adaptive planar gradient tries to figure out when the plain HPWL is inaccurate compared to the exact D2D wirelength and switches a different strategy accordingly. However, it is still far inferior to the results with FDA, as the adaptive planar gradient in Definition 6 focuses on optimizations of planar coordinates $\boldsymbol{x}, \boldsymbol{y}$ without comprehension of partitioning.

In the third main column of Table 5.3, the plain HPWL is equipped with FDA, which means that we use $\tilde{W}_e$ to replace $W_{e,\mathrm{Bi}}$ in Definition 7. However, the plain HPWL $\tilde{W}_e$ is irrelevant to $\boldsymbol{z}$ and thus insensitive to different partitioning. Therefore, nonzero gradients occur only because of changes of node attributes given heterogeneous technologies, resulting in less than 3% wirelength improvements with significant cutsize degradation. By contrast, BiHPWL is evidently sensitive to partitioning, leading to 13.4% wirelength improvement when FDA is enabled, as shown in the last main column in Table 5.3. Note that we utilize much more resources of vertical interconnects to optimize wirelength

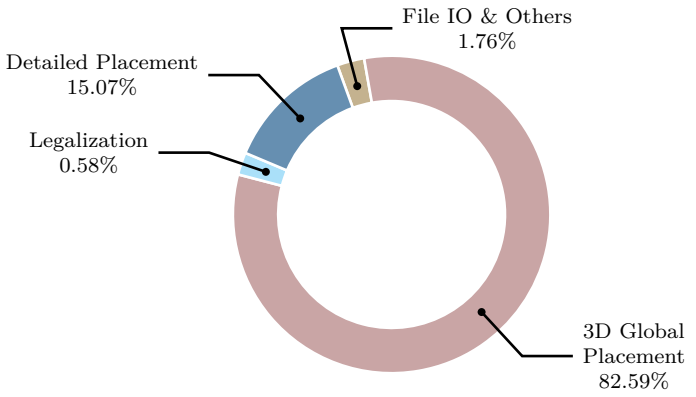


Figure 5.7: *The runtime breakdown of our proposed analytical 3D placement framework on the ICCAD 2022 Contest benchmark* `case4h` *[Hu+22]. Our global placement and detailed placement are both GPU-accelerated.*

versus plain HPWL, fully taking advantage of the benefits of F2F-bonded 3D ICs. Meanwhile, our framework still significantly outperforms the first-place winner on cutsize, preserving advantages on wirelength.

5.5.5 Runtime Breakdown

Figure 5.7 plots the runtime breakdown on `case4h` for our 3D analytical placement framework. The GPU-accelerated 3D global placement takes 82.59% of the total runtime, while the GPU-accelerated detailed placement takes 15.07%.

Similar to [Lin+19], the density and its gradients are computed with a GPU-accelerated implementation of 3D FFT in the 3D global placement. Given the ultra-fast density computation, we set the number of bins $N_z = 32$ by default for all nontrivial cases in [Hu+22] so that the discrete grids can model the 3D electric field more precisely and thus produce better results. The proposed bistratal wirelength model is also implemented based on weighted-average [HCB11; HBC13] with GPU-acceleration techniques in [Lin+19]. The computation of wirelength and density with their gradients take up the main part of runtime in global placement. It is worth mentioning that we can achieve 9.807× and 7.506× runtime speedup for the largest two designs `case4` and `case4h` over the first-place winner, demonstrating that our placement framework is scalable.

5.6 Summary

This paper proposes a new analytical 3D placement framework for face-to-face (F2F) bonded 3D ICs with heterogeneous technologies, incorporating a novel bistratal wirelength model. The proposed framework leverages high-performance GPU-accelerated implementations of both the wirelength model and the electrostatic-based density model. The experimental results on ICCAD 2022 Contest benchmarks demonstrate that our framework significantly surpass the first-place winner and the SOTA analytical 3D placer by 4.1% and 2.1% on wirelength, respectively, with much fewer vertical interconnections and conspicuous acceleration. The 3D placement framework accomplishes partitioning and

placement in a single run, proving that true 3D analytical placement can effectively handle partitioning with respect to wirelength optimization for F2F-bonded 3D ICs and thus inspire more explorations and studies on 3D analytical placement algorithms.

Table 5.2: *The experimental results on the ICCAD 2022 Contest [Hu+22] Benchmarks compared to the top-3 winners and the SOTA analytical 3D placer [Che+23a].* ***WL*** *indicates the* exact *D2D wirelength evaluated by the provided official evaluator.* ***HBTs*** *represents the cut size,* i.e., *the total number of hybrid bonding terminals.* ***RT*** *(s) stands for the total runtime.*

Bench.	1st Place			2nd Place			3rd Place			[Che+23a]			Ours		
	WL	HBTs	RT	WL	HBTs	RT	WL	HBTs	RT	WL	HBTs	RT	WL	HBTs	RT
case2	2072075	1131	47	2080647	477	7	2097487	163	5	2011447	784	33	**1944656**	646	38
case2h	2555461	1083	45	2735158	687	8	2644791	151	5	2514597	891	32	**2462553**	345	40
case3	30580336	16820	342	30969011	11257	234	33063568	14788	68	30302643	8169	141	**30062713**	8017	92
case3h	27650329	16414	224	27756492	8953	243	28372567	11211	63	27135602	7727	155	**26727327**	8887	93
case4	281315669	84069	1324	274026687	51480	1675	281378049	46468	391	272327370	53264	1189	**267400694**	42763	135
case4h	301193374	84728	1096	308359159	59896	2040	307399565	58860	427	296655075	49616	1190	**289245472**	47712	146
Avg.	1.041	2.096	4.300	1.057	1.267	5.320	1.072	1.019	1.249	1.021	1.328	3.639	1.000	1.000	1.000

Table 5.3: *The ablation study results on the ICCAD 2022 Contest [Hu+22] Benchmarks using different wirelength models with the same experimental settings.* ***WL*** *indicates the* exact *D2D wirelength evaluated by the provided official evaluator.* ***HBTs*** *represents the cut size,* i.e.*, the total number of hybrid bonding terminals. BiHPWL is the bistratal wirelength equipped with adaptive planar gradient in Definition 6. FDA indicates finite difference approximation of depth gradients in Definition 7.*

Bench.	Plain HPWL		BiHPWL w/o. FDA		Plain HPWL w/. FDA		BiHPWL w/. FDA	
	WL	HBTs	WL	HBTs	WL	HBTs	WL	HBTs
`case2`	2351813	459	2271554	454	2118450	708	**1944656**	646
`case2h`	2919815	236	2755549	245	3001905	441	**2462553**	345
`case3`	34776108	4396	33965431	4547	35577287	8086	**30062713**	8017
`case3h`	31093130	4770	30066866	4781	30748977	8544	**26727327**	8887
`case4`	309580785	19339	304667903	23261	288957440	51369	**267400694**	42763
`case4h`	330290736	18971	325610343	22195	324613980	54942	**289245272**	47712
Avg.	1.169	0.555	1.134	0.588	1.141	1.116	1.000	1.000

CONCLUSION AND FUTURE WORK

In this thesis, several analytical placement techniques in 2D and 3D scenarios are proposed and discussed to enhance and improve the quality of current placers. In addition to the rigorous theoretical analysis, we have also conducted intensive experiments to verify the efficacy of our proposed algorithms and theories.

In Chapter 3, we propose a new differentiable wirelength model using the Moreau envelope to approximate HPWL. Considering that the differentiability of wirelength models is very critical to gradient-based numerical optimization, the proposed algorithm is able to surpass the previous nonlinear models in terms of numerical stability, convexity and approximatin error. We have theoretically analyzed the rationality, feasibility, and superiority. By combining the state-of-the-art electrostatic-based placement algorithm, the experimental results demonstrate that our proposed algorithm can achieve significant HPWL improvement compared to the most widely-used nonlinear wirelength models.

In Chapter 4, we propose a timing-driven global placement algorithm leveraging a momentum-based net weighting strategy. Besides, we improve the preconditioner to incorporate our net weighting scheme. Existing global placement algorithms mostly focus on wirelength optimization without considering timing. Providing that timing is hard to analytically optimize through an explicit objective, we use the net weighting scheme to incorporate timing metrics into analytical placement so that the placer can conciously consider timing during optimization. The preconditioner is correspondingly upgraded with net weights considered to stable the process of numerical optimization and thus more robust to avoid possible divergence. Experimental results demonstrate that our algorithm can significantly improve TNS and meanwhile be beneficial to WNS.

In Chapter 5, we present a new analytical 3D placement framework with a bistratal wirelength model for F2F-bonded 3D ICs with heterogeneous technology nodes based on the electrostatic-based density model. The proposed framework, enabled GPU-acceleration, is capable of efficiently determining node partitioning and locations simultaneously, leveraging the dedicated 3D wirelength model and density model. The experimental results on the latest public contest benchmarks demonstrate the extreme efficacy and efficiency of our proposed 3D placement framework. Compared to the state-of-the-art heterogeneous 3D placers, the proposed one is much more powerful and scalable using GPUs.

With above exploration, discussion, and theoretical analysis on analytical placement techniques, we have already dived into a deep place of numerical approaches in VLSI placement. However, there still remain numerous challenges I would like to mention here, based on my knowledge of analytical placement and the research I have done. These topics may have the potential to push forward the

cutting-edge progress of acedemic and engineering efforts on analytical placement.

- **Mixed-Size Placement**. Placement with macros, especially in a large range of sizes, is extremely challenging. Nowadays, mixed-size placement has drawn people's attention as deep learning has been thriving in various modern applications. There are several works with modern deep learning [CY21; LML22; Lai+23] to accomplish macro placement and leverage reliable analytical placers to perform cell placement. However, industrial designs may become more complex than imagined, leading to a high probability for learning-based placers to fail. DREAMPLACE [Lin+19; Lin+20b; Lin+20a] is proved to be powerful on both solution quality and end-to-end runtime, but its robustness on mixed-size placement still remains unsolved. Therefore, designing robust and explainable analytical algorithms for mixed-size placement has become the bottleneck of modern wirelength-driven placement.

- **Powerful Optimizers**. Gradient-based numerical optimization heavily relies on the adopted optimizer. The most popular optimizer adopted by modern electrostatic-based placers (DREAMPLACE [Lin+19; Lin+20b; Lin+20a], REPLACE [Che+18], *etc.*) derives from the well known *Nesterov Accelerated Gradient* [Nes83]. DREAMPLACE 4.1 [Che+23b] enhances stability of macro placement by incorporating *Barzilai-Borwein* [BB88] (BB) method. Following Theorem 1 in this thesis, we are reasonably capable of handling non-smooth numerical optimization. Through this way, we can also solve global placement with proximal optimizers, which does not require any explicit differentiable approximation to the convex HPWL objective. Hence, the development of *non-smooth* optimizers based on the proximal mapping of HPWL may help to improve the numerical optimization of analytical global placement. Correspondingly, more theoretical analysis and intensive experiments are also required.

- **Ultra-Fast and Robust 3D Placement**. We have proposed a 3D analytical placer in Chapter 5. Despite its wonderful performance on the ICCAD 2022 Contest [Hu+22] Benchmarks, there is no enough data of about mixed-size 3D placement or experiments on that. Besides, 3D scenarios expand all possible issues occurring in 2D cases and make it more complex than imagined. For example, the density map calculation and the gradient aggregation for large macros would be significant when there exists another dimension. In short, efficient 3D placement with high quality is still an open challenge. Some further academic research can be conducted on our foundation of analytical 3D placement.

- **Realistic Objectives**. In Chapter 4, we discussed a net weighting scheme to enable timing optimization indirectly in normal analytical placement. Some recent works [GL22] suggest that directly optimizing differentiable timing objectives is absolutely possible. More realistic objectives might be analously considered as differentiable in analytical placement, and thus provide a more elegant way for researchers to follow.

REFERENCES

[Abe26] Niels Henrik Abel. "Untersuchungen über die Reihe: $1+\frac{m}{1}x+\frac{m\cdot(m-1)}{1\cdot 2}\cdot x^2+\frac{m\cdot(m-1)\cdot(m-2)}{1\cdot 2\cdot 3}\cdot x^3+\cdots$ u.s.w." In: *J. Reine Angew. Math* (1826).

[Afr71] SN Afriat. "Theory of maxima and the method of Lagrange". In: *SIAM Journal on Applied Mathematics* 20.3 (1971), pp. 343–357.

[Bat+12] Perrine Batude, Thomas Ernst, Julien Arcamone, Gregory Arndt, Perceval Coudrain, and Pierre-Emmanuel Gaillardon. "3-D sequential integration: A key enabling technology for heterogeneous co-integration of new function with CMOS". In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS)* 2.4 (2012), pp. 714–722.

[BB88] Jonathan Barzilai and Jonathan M Borwein. "Two-point step size gradient methods". In: *IMA journal of numerical analysis* 8.1 (1988), pp. 141–148.

[BM60] George Boole and John Fletcher Moulton. *A treatise on the calculus of finite differences*. reprinted from 1872. Dover, New York, 1960.

[BSV08] Ulrich Brenner, Markus Struzyna, and Jens Vygen. "BonnPlace: Placement of leading-edge chips by advanced combinatorial algorithms". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 27.9 (2008), pp. 1607–1620.

[BY85] Michael Burstein and Mary N Youssef. "Timing influenced layout design". In: *ACM/IEEE Design Automation Conference (DAC)*. IEEE. 1985, pp. 124–130.

[CH93] Bintong Chen and Patrick T Harker. "A non-interior-point continuation method for linear complementarity problems". In: *SIAM Journal on Matrix Analysis and Applications* 14.4 (1993), pp. 1168–1190.

[Cha+02] H Chang, Eugene Shragowitz, Jian Liu, Habib Youssef, Bing Lu, and Suphachai Sutanthavibul. "Net criticality revisited: An effective method to improve timing in physical design". In: *ispd*. 2002, pp. 155–160.

[Cha+07] Tony F Chan, Kenton Sze, Joseph R Shinnerl, and Min Xie. "MPL6: Enhanced multilevel mixed-size placement with congestion control". In: *Modern Circuit Placement*. Springer, 2007, pp. 247–288.

[Cha+16] Kyungwook Chang, Saurabh Sinha, Brian Cline, Raney Southerland, Michael Doherty, Greg Yeric, and Sung Kyu Lim. "Cascade2D: A design-aware partitioning approach to monolithic 3D IC with 2D commercial tools". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2016, pp. 1–8.

[Che+08] Tung-Chieh Chen, Zhe-Wei Jiang, Tien-Chang Hsu, Hsin-Chen Chen, and Yao-Wen Chang. "NTUplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 27.7 (2008), pp. 1228–1240.

[Che+18] Chung-Kuan Cheng, Andrew B Kahng, Ilgweon Kang, and Lutong Wang. "RePlAce: Advancing solution quality and routability validation in global placement". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 38.9 (2018), pp. 1717–1730.

[Che+23a] Yan-Jen Chen, Yan-Syuan Chen, Wei-Che Tseng, Cheng-Yu Chiang, Yu-Hsiang Lo, and Yao-Wen Chang. "Late Breaking Results: Analytical Placement for 3D ICs with Multiple Manufacturing Technologies". In: *ACM/IEEE Design Automation Conference (DAC)*. 2023.

[Che+23b] Yifan Chen, Zaiwen Wen, Yun Liang, and Yibo Lin. "Stronger Mixed-Size Placement Backbone Considering Second-Order Information". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2023, pp. 1–9.

[Cho+05] Amit Chowdhary, Karthik Rajagopal, Satish Venkatesan, Tung Cao, Vladimir Tiourin, Yegna Parasuram, and Bill Halpin. "How accurately can we model timing in a placement engine?" In: *ACM/IEEE Design Automation Conference (DAC)*. 2005, pp. 801–806.

[Con+07] Jason Cong, Guojie Luo, Jie Wei, and Yan Zhang. "Thermal-aware 3D IC placement via transformation". In: *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*. IEEE. 2007, pp. 780–785.

[CR41] Richard Courant and Herbert Robbins. *What is Mathematics?* Oxford University Press, New York, 1941.

[CW07] Chris Chu and Yiu-Chung Wong. "FLUTE: Fast lookup table based rectilinear Steiner minimal tree algorithm for VLSI design". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 27.1 (2007), pp. 70–83.

[CY21] Ruoyu Cheng and Junchi Yan. "On joint learning for solving placement and routing in chip design". In: *Annual Conference on Neural Information Processing Systems (NeurIPS)* 34 (2021), pp. 16508–16519.

[DCR03] Shamik Das, Anantha Chandrakasan, and Rafael Reif. "Design tools for 3-D integrated circuits". In: *IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC)*. 2003, pp. 53–56.

[DM01] Yangdong Deng and Wojciech P Maly. "Interconnect characteristics of 2.5-D system integration scheme". In: *ACM International Symposium on Physical Design (ISPD)*. 2001, pp. 171–175.

[Dun+84] Alfred E Dunlop, Vishwani D Agrawal, David N Deutsch, MF Jukl, Patrick Kozak, and Manfred Wiesel. "Chip layout optimization using critical path weighting". In: *ACM/IEEE Design Automation Conference (DAC)*. IEEE. 1984, pp. 133–136.

[DZX10] Xiangyu Dong, Jishen Zhao, and Yuan Xie. “Fabrication cost analysis and cost-aware design space exploration for 3-D ICs”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 29.12 (2010), pp. 1959–1972.

[EJ98] Hans Eisenmann and Frank M Johannes. “Generic global placement and floorplanning”. In: *ACM/IEEE Design Automation Conference (DAC)*. 1998, pp. 269–274.

[Elm48] William C Elmore. “The transient response of damped linear networks with particular regard to wideband amplifiers”. In: *Journal of applied physics* 19.1 (1948), pp. 55–63.

[FM82] Charles M Fiduccia and Robert M Mattheyses. “A linear-time heuristic for improving network partitions”. In: *ACM/IEEE Design Automation Conference (DAC)*. 1982, pp. 175–181.

[GC94] Joseph L Ganley and James P Cohoon. “A faster dynamic programming algorithm for exact rectilinear Steiner minimal trees”. In: *ACM Great Lakes Symposium on VLSI (GLSVLSI)*. IEEE. 1994, pp. 238–241.

[GGL22] Zizheng Guo, Feng Gu, and Yibo Lin. “GPU-accelerated rectilinear Steiner tree generation”. In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2022, pp. 1–9.

[GHL20] Zizheng Guo, Tsung-Wei Huang, and Yibo Lin. “GPU-accelerated static timing analysis”. In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2020, pp. 1–9.

[GJ77] Michael R Garey and David S. Johnson. “The rectilinear Steiner tree problem is NP-complete”. In: *SIAM Journal on Applied Mathematics* 32.4 (1977), pp. 826–834.

[GL22] Zizheng Guo and Yibo Lin. “Differentiable-timing-driven global placement”. In: *ACM/IEEE Design Automation Conference (DAC)*. 2022, pp. 1315–1320.

[Gom+22] Wilfred Gomes, Slade Morgan, Boyd Phelps, Tim Wilson, and Erik Hallnor. “Meteor Lake and Arrow Lake Intel Next-Gen 3D Client Architecture Platform with Foveros”. In: *IEEE Hot Chips 34 Symposium (HCS)*. IEEE Computer Society. 2022, pp. 1–40.

[Gri+94] Jeff Griffith, Gabriel Robins, Jeffrey S Salowe, and Tongtong Zhang. “Closing the gap: Near-optimal Steiner trees in polynomial time”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 13.11 (1994), pp. 1351–1365.

[GS03] Brent Goplen and Sachin Sapatnekar. “Efficient thermal placement of standard cells in 3D ICs using a force directed approach”. In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2003, pp. 86–89.

[GS07] Brent Goplen and Sachin Spatnekar. “Placement of 3D ICs with thermal and interlayer via considerations”. In: *ACM/IEEE Design Automation Conference (DAC)*. 2007, pp. 626–631.

[Gu+20] Jiaqi Gu, Zixuan Jiang, Yibo Lin, and David Z Pan. “DREAMPlace 3.0: Multi-electrostatics based robust VLSI placement with region constraints”. In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2020, pp. 1–9.

[GVL92] Tong Gao, Pravin M Vaidya, and CL Liu. "A Performance Driven Macro-Cell Placement Algorithm." In: *ACM/IEEE Design Automation Conference (DAC)*. 1992, pp. 147–152.

[Han66] Maurice Hanan. "On Steiner's problem with rectilinear distance". In: *SIAM Journal on Applied mathematics* 14.2 (1966), pp. 255–265.

[HBC13] Meng-Kai Hsu, Valeriy Balabanov, and Yao-Wen Chang. "TSV-aware analytical placement for 3-D IC designs based on a novel weighted-average wirelength model". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 32.4 (2013), pp. 497–509.

[HCB11] Meng-Kai Hsu, Yao-Wen Chang, and Valeriy Balabanov. "TSV-aware analytical placement for 3D IC designs". In: *ACM/IEEE Design Automation Conference (DAC)*. 2011, pp. 664–669.

[HCC93] Takeo Hamada, Chung-Kuan Cheng, and Paul M Chau. "Prime: A timing-driven placement tool using a piecewise linear resistive network approach". In: *ACM/IEEE Design Automation Conference (DAC)*. 1993, pp. 531–536.

[HCS00] Bill Halpin, CY Roger Chen, and Naresh Sehgal. "A sensitivity based placer for standard cells". In: *Proceedings of the 10th Great Lakes symposium on VLSI*. 2000, pp. 193–196.

[Hen+06] Renato Hentschke, Guilherme Flach, Felipe Pinto, and Ricardo Reis. "Quadratic placement for 3d circuits using z-cell shifting, 3d iterative refinement and simulated annealing". In: *Proceedings of Symposium on Integrated Circuits and Systems Design (SBCCI)*. 2006, pp. 220–225.

[Hil02] Dwight Hill. *Method and system for high speed detailed placement of cells within an integrated circuit design*. US Patent 6,370,673. 2002.

[Hit82] Robert B Hitchcock. "Timing verification and the timing analysis program". In: *ACM/IEEE Design Automation Conference (DAC)*. IEEE. 1982, pp. 594–604.

[Hu+22] Kai-Shun Hu, I-Jye Lin, Yu-Hui Huang, Hao-Yu Chi, Yi-Hsuan Wu, and Chin-Fang Cindy Shen. "2022 ICCAD CAD Contest Problem B: 3D Placement with D2D Vertical Connections". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2022, pp. 1–5.

[Hua+17] Chau-Chin Huang, Hsin-Ying Lee, Bo-Qiao Lin, Sheng-Wei Yang, Chin-Hao Chang, Szu-To Chen, Yao-Wen Chang, Tung-Chieh Chen, and Ismail Bustany. "NTUplace4dr: A detailed-routing-driven placer for mixed-size circuit designs with technology and region constraints". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 37.3 (2017), pp. 669–681.

[HW15] Tsung-Wei Huang and Martin DF Wong. "OpenTimer: A high-performance timing analysis tool". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2015, pp. 895–902.

[Hwa76] Frank K Hwang. "On Steiner minimal trees with rectilinear distance". In: *SIAM journal on Applied Mathematics* 30.1 (1976), pp. 104–114.

[Hwa79] Frank K Hwang. "An $O(n \log n)$ algorithm for rectilinear minimal spanning trees". In: *Journal of the ACM (JACM)* 26.2 (1979), pp. 177–182.

[JK89] Michael AB Jackson and Ernest S Kuh. "Performance-driven placement of cell based IC's". In: *ACM/IEEE Design Automation Conference (DAC)*. 1989, pp. 370–375.

[Jor56] Charles Jordan. *Calculus of finite differences*. Chelsea, New York, 1956.

[Jun+14] Moongon Jung, Taigon Song, Yang Wan, Yarui Peng, and Sung Kyu Lim. "On enhancing power benefits in 3D ICs: Block folding and bonding styles perspective". In: *ACM/IEEE Design Automation Conference (DAC)*. 2014, pp. 1–6.

[Kah+11] Andrew B Kahng, Jens Lienig, Igor L Markov, and Jin Hu. *VLSI physical design: from graph partitioning to timing closure*. Springer Science & Business Media, 2011.

[Kah+24] Andrew B Kahng, Robert R Nerem, Yusu Wang, and Chien-Yi Yang. "NN-Steiner: A mixed neural-algorithmic approach for the rectilinear Steiner minimum tree problem". In: *AAAI Conference on Artificial Intelligence (AAAI)*. Vol. 38. 12. 2024, pp. 13022–13030.

[KAL09] Dae Hyun Kim, Krit Athikulwongse, and Sung Kyu Lim. "A study of through-silicon-via impact on the 3D stacked IC layout". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2009, pp. 674–680.

[KCL18] Bon Woong Ku, Kyungwook Chang, and Sung Kyu Lim. "Compact-2D: A physical design methodology to build commercial-quality face-to-face-bonded 3D ICs". In: *ACM International Symposium on Physical Design (ISPD)*. 2018, pp. 90–97.

[Kim+12] Myung-Chul Kim, Natarajan Viswanathan, Charles J Alpert, Igor L Markov, and Shyam Ramji. "MAPLE: Multilevel adaptive placement for mixed-size designs". In: *ACM International Symposium on Physical Design (ISPD)*. 2012, pp. 193–200.

[Kim+15] Myung-Chul Kim, Jin Hu, Jiajia Li, and Natarajan Viswanathan. "ICCAD-2015 CAD contest in incremental timing-driven placement and benchmark suite". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2015, pp. 921–926.

[KLM11] Myung-Chul Kim, Dong-Jin Lee, and Igor L Markov. "SimPL: An effective placement algorithm". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 31.1 (2011), pp. 50–60.

[KM12] Myung-Chul Kim and Igor L Markov. "ComPLx: A competitive primal-dual Lagrange optimization for global placement". In: *ACM/IEEE Design Automation Conference (DAC)*. 2012, pp. 747–752.

[KMZ03] Andrew B Kahng, Ion I Măndoiu, and Alexander Z Zelikovsky. "Highly scalable algorithms for rectilinear and octilinear Steiner trees". In: *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*. 2003, pp. 827–833.

[KOB03] Idris Kaya, Markus Olbrich, and Erich Barke. "3-D placement considering vertical interconnects". In: *IEEE International System-on-Chip Conference (SOCC)*. IEEE. 2003, pp. 257–258.

[Kon02] Tim Kong. "A novel net weighting algorithm for timing-driven placement". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2002, pp. 172–176.

[KR92] Andrew B Kahng and Gabriel Robins. "A new class of iterative Steiner tree heuristics with good performance". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 11.7 (1992), pp. 893–902.

[KW04] Andrew B Kahng and Qinke Wang. "Implementation and extensibility of an analytic placer". In: *ACM International Symposium on Physical Design (ISPD)*. 2004, pp. 18–25.

[KW06] Andrew B Kahng and Qinke Wang. "A faster implementation of APlace". In: *ACM International Symposium on Physical Design (ISPD)*. 2006, pp. 218–220.

[Lai+23] Yao Lai, Jinxin Liu, Zhentao Tang, Bin Wang, Jianye Hao, and Ping Luo. "Chipformer: Transferable chip placement via offline decision transformer". In: *International Conference on Machine Learning (ICML)*. PMLR. 2023, pp. 18346–18364.

[LCY21] Jinwei Liu, Gengjie Chen, and Evangeline FY Young. "REST: Constructing rectilinear steiner minimum tree via reinforcement learning". In: *ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2021, pp. 1135–1140.

[Lia+23] Peiyu Liao, Hongduo Liu, Yibo Lin, Bei Yu, and Martin Wong. "On a Moreau Envelope wirelength model for analytical global placement". In: *ACM/IEEE Design Automation Conference (DAC)*. 2023.

[Lin+19] Yibo Lin, Shounak Dhar, Wuxi Li, Haoxing Ren, Brucek Khailany, and David Z Pan. "DREAMPlace: Deep learning toolkit-enabled GPU acceleration for modern VLSI placement". In: *ACM/IEEE Design Automation Conference (DAC)*. 2019, pp. 1–6.

[Lin+20a] Yibo Lin, Zixuan Jiang, Jiaqi Gu, Wuxi Li, Shounak Dhar, Haoxing Ren, Brucek Khailany, and David Z Pan. "Dreamplace: Deep learning toolkit-enabled gpu acceleration for modern vlsi placement". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2020).

[Lin+20b] Yibo Lin, Wuxi Li, Jiaqi Gu, Haoxing Ren, Brucek Khailany, and David Z Pan. "ABCD-Place: Accelerated batch-based concurrent detailed placement on multithreaded CPUs and GPUs". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 39.12 (2020), pp. 5083–5096.

[Liu+19] Wen-Hao Liu, Stefanus Mantik, Wing-Kai Chow, Yixiao Ding, Amin Farshidi, and Gracieli Posser. "ISPD 2019 initial detailed routing contest and benchmark with advanced routing rules". In: *ACM International Symposium on Physical Design (ISPD)*. 2019, pp. 147–151.

[Liu+22] Lixin Liu, Bangqi Fu, Martin D. F. Wong, and Evangeline F. Y. Young. "Xplace: an extremely fast and extensible global placement framework". In: *ACM/IEEE Design Automation Conference (DAC)*. 2022, pp. 1309–1314.

[LK07] Chen Li and Cheng-Koh Koh. "Recursive function smoothing of half-perimeter wirelength for analytical placement". In: *IEEE International Symposium on Quality Electronic Design (ISQED)*. IEEE. 2007, pp. 829–834.

[LLP19] Wuxi Li, Yibo Lin, and David Z Pan. "elfPlace: Electrostatics-based placement for large-scale heterogeneous FPGAs". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2019, pp. 1–8.

[LML22] Yao Lai, Yao Mu, and Ping Luo. "Maskplace: Fast chip placement via reinforced visual representation learning". In: *Annual Conference on Neural Information Processing Systems (NeurIPS)* 35 (2022), pp. 24019–24030.

[LP08] Tao Luo and David Z Pan. "DPlace2.0: A stable and efficient analytical placement based on diffusion". In: *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*. IEEE. 2008, pp. 346–351.

[LSC13] Guojie Luo, Yiyu Shi, and Jason Cong. "An analytical placement framework for 3-D ICs and its extension on thermal awareness". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 32.4 (2013), pp. 510–523.

[Lu+13] Jingwei Lu, Pengwen Chen, Chin-Chih Chang, Lu Sha, Dennis Jen-Hsin Huang, Chin-Chi Teng, and Chung-Kuan Cheng. "FFTPL: An analytic placement algorithm using fast fourier transform for density equalization". In: *IEEE International Conference on ASIC (ASICON)*. IEEE. 2013, pp. 1–4.

[Lu+15a] Jingwei Lu, Pengwen Chen, Chin-Chih Chang, Lu Sha, Dennis Jen-Hsin Huang, Chin-Chi Teng, and Chung-Kuan Cheng. "ePlace: Electrostatics-based placement using fast fourier transform and Nesterov's method". In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 20.2 (2015), pp. 1–34.

[Lu+15b] Jingwei Lu, Hao Zhuang, Pengwen Chen, Hongliang Chang, Chin-Chih Chang, Yiu-Chung Wong, Lu Sha, Dennis Huang, Yufeng Luo, Chin-Chi Teng, et al. "ePlace-MS: Electrostatics-based placement for mixed-size circuits". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 34.5 (2015), pp. 685–698.

[Lu+16] Jingwei Lu, Hao Zhuang, Ilgweon Kang, Pengwen Chen, and Chung-Kuan Cheng. "ePlace-3D: Electrostatics based placement for 3D-ICs". In: *ACM International Symposium on Physical Design (ISPD)*. 2016, pp. 11–18.

[Lu+20] Yi-Chen Lu, Sai Surya Kiran Pentapati, Lingjun Zhu, Kambiz Samadi, and Sung Kyu Lim. "TP-GNN: A graph neural network framework for tier partitioning in monolithic 3D ICs". In: *ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2020, pp. 1–6.

[Luk91] Wing K Luk. "A fast physical constraint generator for timing driven layout". In: *ACM/IEEE Design Automation Conference (DAC)*. 1991, pp. 626–631.

[Mel61] Zdzislaw Alexander Melzak. "On the problem of Steiner". In: *Canadian Mathematical Bulletin* 4.2 (1961), pp. 143–148.

[MHK15] Igor L Markov, Jin Hu, and Myung-Chul Kim. "Progress and challenges in VLSI placement research". In: *Proceedings of the IEEE* 103.11 (2015), pp. 1985–2003.

[Mor+06] PR Morrow, C-M Park, S Ramanathan, MJ Kobrinsky, and M Harmes. "Three-dimensional wafer stacking via Cu-Cu bonding integrated with 65-nm strained-Si/low-k CMOS technology". In: *IEEE Electron Device Letters (EDL)* 27.5 (2006), pp. 335–337.

[Mor65] Jean-Jacques Moreau. “Proximité et dualité dans un espace hilbertien”. In: *Bulletin de la Société mathématique de France* 93 (1965), pp. 273–299.

[MS99] Naresh Maheshwari and Sachin S Sapatnekar. “Timing Analysis of Sequential Circuits”. In: *Timing Analysis and Optimization of Sequential Circuits*. Springer, 1999, pp. 7–31.

[MT00] Louis Melville Milne-Thomson. *The calculus of finite differences*. reprinted from 1933. American Mathematical Soc., 2000.

[Mun+23] Benjamin Munger, Kathy Wilcox, Jeshuah Sniderman, Chuck Tung, Brett Johnson, Russell Schreiber, Carson Henrion, Kevin Gillespie, Tom Burd, Harry Fair, et al. ““Zen 4”: The AMD 5nm 5.7 GHz x86-64 microprocessor core”. In: *IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE. 2023, pp. 38–39.

[Mur+22] Gauthaman Murali, Sandra Maria Shaji, Anthony Agnesina, Guojie Luo, and Sung Kyu Lim. “ART-3D: Analytical 3D placement with reinforced parameter tuning for monolithic 3D ICs”. In: *ACM International Symposium on Physical Design (ISPD)*. 2022, pp. 97–104.

[N24] Niels Erik Nörlund. *Vorlesungen über differenzenrechnung*. Springer, Berlin, 1924.

[Nam06] Gi-Joon Nam. “ISPD 2006 placement contest: Benchmark suite and results”. In: *ACM International Symposium on Physical Design (ISPD)*. 2006, pp. 167–167.

[NDS01] William C Naylor, Ross Donelly, and Lu Sha. *Non-linear optimization system and method for wire length and delay optimization for an automatic electric circuit placer*. US Patent 6,301,693. 2001.

[Nes83] Yurii Nesterov. “A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$”. In: *Doklady Akademii Nauk*. Vol. 269. 3. Russian Academy of Sciences. 1983, pp. 543–547.

[OJ04] Bernd Obermeier and Frank M Johannes. “Quadratic placement using an improved timing model”. In: *ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2004, pp. 705–710.

[Pan+14] Shreepad A Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim. “Design and CAD methodologies for low power gate-level monolithic 3D ICs”. In: *IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. 2014, pp. 171–176.

[Pan+17] Shreepad Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim. “Shrunk-2-D: A physical design methodology to build commercial-quality monolithic 3-D ICs”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 36.10 (2017), pp. 1716–1724.

[Par+20] Heechun Park, Bon Woong Ku, Kyungwook Chang, Da Eun Shim, and Sung Kyu Lim. “Pseudo-3D approaches for commercial-grade RTL-to-GDS tool flow targeting monolithic 3D ICs”. In: *ACM International Symposium on Physical Design (ISPD)*. 2020, pp. 47–54.

[Pen+20] Sai Surya Kiran Pentapati, Kyungwook Chang, Vassilios Gerousis, Rwik Sengupta, and Sung Kyu Lim. "Pin-3D: A physical synthesis and post-layout optimization flow for heterogeneous monolithic 3D ICs". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2020, pp. 1–9.

[PHR08] David Z Pan, Bill Halpin, and Haoxing Ren. "21 Timing-Driven Placement". In: *Handbook of Algorithms for Physical Design Automation* (2008), p. 423.

[PL22] Sai Pentapati and Sung Kyu Lim. "Metal layer sharing: A routing optimization technique for monolithic 3D ICs". In: *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)* 30.9 (2022), pp. 1355–1367.

[Pop+14] Sergiy Popovych, Hung-Hao Lai, Chieh-Min Wang, Yih-Lang Li, Wen-Hao Liu, and Ting-Chi Wang. "Density-aware detailed placement with instant legalization". In: *ACM/IEEE Design Automation Conference (DAC)*. 2014, pp. 1–6.

[PR69] Elijah Polak and Gerard Ribiere. "Note sur la convergence de méthodes de directions conjuguées". In: *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique* 3.R1 (1969), pp. 35–43.

[PVC05] Min Pan, Natarajan Viswanathan, and Chris Chu. "An efficient and effective detailed placement algorithm". In: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2005, pp. 48–55.

[Raj+03] Karthik Rajagopal, Tal Shaked, Yegna Parasuram, Tung Cao, Amit Chowdhary, and Bill Halpin. "Timing driven force directed placement with physical net constraints". In: *ispd*. 2003, pp. 60–66.

[RE95] Bernhard M Riess and Gisela G Ettelt. "Speed: Fast and efficient timing driven placement". In: *IEEE International Symposium on Circuits and Systems (ISCAS)*. Vol. 1. IEEE. 1995, pp. 377–380.

[RHW86] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: 323.6088 (1986), pp. 533–536.

[Sam+16] Sandeep Kumar Samal, Deepak Nayak, Motoi Ichihashi, Srinivasa Banna, and Sung Kyu Lim. "Monolithic 3D IC vs. TSV-based 3D IC in 14nm FinFET technology". In: *IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference (S3S)*. IEEE. 2016, pp. 1–2.

[SC19] Fan-Keng Sun and Yao-Wen Chang. "BiG: A bivariate gradient-based wirelength model for analytical circuit placement". In: *ACM/IEEE Design Automation Conference (DAC)*. 2019, pp. 1–6.

[SDJ91] Georg Sigl, Konrad Doll, and Frank M Johannes. "Analytical placement: A linear or a quadratic objective function?" In: *ACM/IEEE Design Automation Conference (DAC)*. 1991, pp. 427–432.

[Sha48] Claude Elwood Shannon. "A mathematical theory of communication". In: *Bell System Technical Journal* 27.3 (1948), pp. 379–423.

[Son+15] Taigon Song, Arthur Nieuwoudt, Yun Seop Yu, and Sung Kyu Lim. "Coupling capacitance in face-to-face (F2F) bonded 3D ICs: Trends and implications". In: *IEEE Electronic Components and Technology Conference (ECTC)*. IEEE. 2015, pp. 529–536.

[SS12] Shai Shalev-Shwartz. "Online learning and online convex optimization". In: *Foundations and Trends® in Machine Learning* 4.2 (2012), pp. 107–194.

[SS95] William Swartz and Carl Sechen. "Timing driven placement for large standard cell circuits". In: *ACM/IEEE Design Automation Conference (DAC)*. 1995, pp. 211–215.

[SSJ08a] Peter Spindler, Ulf Schlichtmann, and Frank M Johannes. "Abacus: Fast legalization of standard cell circuits with minimal movement". In: *ACM International Symposium on Physical Design (ISPD)*. 2008, pp. 47–53.

[SSJ08b] Peter Spindler, Ulf Schlichtmann, and Frank M Johannes. "Kraftwerk2—A fast force-directed quadratic placement approach using an accurate net model". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 27.8 (2008), pp. 1398–1411.

[SW93] Jeffrey S Salowe and David M Warme. "An exact rectilinear Steiner tree algorithm". In: *IEEE International Conference on Computer Design (ICCD)*. IEEE. 1993, pp. 472–475.

[Tan00] Thitipong Tanprasert. "An analytical 3-D placement that reserves routing space". In: *IEEE International Symposium on Circuits and Systems (ISCAS)*. Vol. 3. IEEE. 2000, pp. 69–72.

[Tay15] Brook Taylor. *Methodus incrementorum directa & inversa*. Gul. Innys, London, 1715.

[TK91] Ren-Song Tsay and Juergen Koehl. "An analytic net weighting approach for performance optimization in circuit placement". In: *ACM/IEEE Design Automation Conference (DAC)*. 1991, pp. 620–625.

[VI+21] Pruek Vanna-Iampikul, Chengjia Shao, Yi-Chen Lu, Sai Pentapati, and Sung Kyu Lim. "Snap-3D: A constrained placement-driven physical design methodology for face-to-face-bonded 3D ICs". In: *ACM International Symposium on Physical Design (ISPD)*. 2021, pp. 39–46.

[VPC07] Natarajan Viswanathan, Min Pan, and Chris Chu. "FastPlace 3.0: A fast multilevel quadratic placement algorithm with placement congestion control". In: *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*. 2007, pp. 135–140.

[War98] David Michael Warme. *Spanning trees in hypergraphs with applications to Steiner trees*. University of Virginia, 1998.

[WTC04] Ting-Yuan Wang, Jeng-Liang Tsai, and Charlie Chung-Ping Chen. "Sensitivity guided net weighting for placement driven synthesis". In: *ACM International Symposium on Physical Design (ISPD)*. 2004, pp. 124–131.

[Wyn66] Aaron Daniel Wyner. "The Capacity of the Band-Limited Gaussian Channel". In: *Bell System Technical Journal* 45.3 (1966), pp. 359–395.

[XR05] Zhong Xiu and Rob A Rutenbar. "Timing-driven placement by grid-warping". In: *ACM/IEEE Design Automation Conference (DAC)*. 2005, pp. 585–591.

[YC01] Wei Yu and John M Cioffi. "On constant power water-filling". In: *IEEE International Conference on Communications (ICC)*. Vol. 6. IEEE. 2001, pp. 1665–1669.

[Yeu08] Raymond W Yeung. *Information theory and network coding*. Springer Science & Business Media, 2008.

[YW72] Y Yang and Omar Wing. "Suboptimal algorithm for a wire routing problem". In: *IEEE Transactions on Circuit Theory* 19.5 (1972), pp. 508–510.

[Zhu+15] Wenxing Zhu, Jianli Chen, Zheng Peng, and Genghua Fan. "Nonsmooth optimization method for VLSI global placement". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 34.4 (2015), pp. 642–655.