The Dissertation Committee for Xiaoqing Xu
certifies that this is the approved version of the following dissertation:

Standard Cell Optimization and Physical Design in Advanced Technology Nodes

Committee:

David Z. Pan, Supervisor

Brian Cline

Michael Orshansky

Nan Sun

Nur A. Touba

Standard Cell Optimization and Physical Design in Advanced Technology Nodes

by

Xiaoqing Xu, B.S.; M.S.E.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2017

Dedicated to my parents: XU Min and LIU Shufang

for their unconditional love

To my wife Shaoting Zhang

Acknowledgments

I sincerely appreciate the generous guidance and support from my adviser, Professor David Z. Pan. He is a wise, humble and patient leader. Throughout my graduate study, he not only provides insightful technical suggestions toward important research problems, but also shapes my personality to be more humble and patient. Moreover, he is a nice friend in daily life and teaches me how to communicate with senior professionals and benefit from their life experiences. None of my research studies would have been possible without his support and trust.

I express great appreciations to other committee members for their precious efforts on this dissertation. In particular, I would like to thank Professor Michael Orshansky for his technical suggestions during the development of this dissertation. I would like to thank Professor Nan Sun for his comments on the concepts and scopes of my research studies. I would like to thank Professor Nur A. Touba for his kindness and support to this dissertation. I would like to thank Dr. Brian Cline for his detailed technical suggestions and guidance. He has been a dear friend and great mentor during our continuous research collaborations and my internships at ARM Inc.

Besides, I deeply appreciate the help from other industrial mentors and collaborators: Dr. Greg Yeric at ARM Inc., Dr. Tetsuaki Matsunawa, Dr.

Shigeki Nojima and Dr. Chikaaki Kodama at Toshiba Corporation, Dr. Gi-Joon Nam, Dr. Gustavo Tellez and Dr. Hua Xiang at IBM T. J. Watson Research Center, Dr. Zhuo Li and Dr. Charles Alpert from Cadence Design System, and Dr. Lars Liebmann from GLOBALFOUNDRIES. This work would not be possible without their technical suggestions.

It has been a great learning experience to work with other UTDA members: Bei Yu, Jerrica Gao, Subhendu Roy, Yibo Lin, Jiaojiao Ou, Meng Li, Vinicius Livramento, Che-Lun Hsu, Biying Xu and Wei Ye. This dissertation has benefited a lot from the productive collaborations and discussions with them.

I am deeply thankful for my parents back in China, XU Min and LIU Shufang. Although they grow up and live in the countryside of China, they have always believed that knowledge changes life and fate. They have also been providing unconditional love and support for my graduate studies at the very beginning. I am also sincerely grateful for my wife, Shaoting Zhang, for her companion and encouragement, especially during those tiring and gloomy days during my Ph.D. studies.

Portions of this work were supported by SRC, NSF, Toshiba, SPIE BACUS Scholarship and University Graduate Continuing Fellowship.

Standard Cell Optimization and Physical Design in Advanced Technology Nodes

Publication No. ____________

Xiaoqing Xu, Ph.D.
The University of Texas at Austin, 2017

Supervisor: David Z. Pan

Integrated circuits (ICs) are at the heart of modern electronics, which rely heavily on the state-of-the-art semiconductor manufacturing technology. The key to pushing forward semiconductor technology is IC feature-size miniaturization. However, this brings ever-increasing design complexities and manufacturing challenges to the $340 billion semiconductor industry. The manufacturing of two-dimensional layout on high-density metal layers depends on complex design-for-manufacturing techniques and sophisticated empirical optimizations, which introduces huge amounts of turnaround time and yield loss in advanced technology nodes. Our study reveals that unidirectional layout design can significantly reduce the manufacturing complexities and improve the yield, which is becoming increasingly adopted in semiconductor industry [61, 89]. The lithography printing of unidirectional layout can be tightly controlled using advanced patterning techniques, such as self-aligned double and quadruple patterning. Despite the manufacturing benefits, unidirectional layout leads to more restrictive solution space and brings significant impacts on the IC design automation flow for routing closure. Notably, unidirectional routing limits the standard cell pin accessibility, which further exacerbates the resource competitions during routing. Moreover, for post-routing optimization, traditional redundant-via insertion has become obsolete under unidirectional routing style, which makes the yield enhancement task extremely challenging. Regardless of complex multiple patterning and design-for-manufacturing approaches, mask optimization through resolution enhancement techniques remains as the key strategy to improve the yield of the semiconductor manufacturing processes. Among them, Sub-Resolution Assist Feature (SRAF) generation is a very important method to improve lithographic process windows. Model-based SRAF generation has been widely used to achieve high accuracy but it is time-consuming and hard to obtain consistent SRAFs.

This dissertation proposes novel CAD algorithms and methodologies for standard cell optimization and physical design in advanced technology nodes, which ultimately reduces the design cycle and manufacturing cost of IC design. First, a standard cell pin access optimization engine is proposed to evaluate the pin accessibility of a given standard cell library. We further propose novel pin access planning techniques and concurrent pin access optimizations to efficiently resolve the routing resource competitions, which generates much better routing solutions than state-of-the-art, manufacturing-friendly routers. To systematically improve the manufacturing yield in the post-routing stage, a global optimization engine has been introduced for redundant local-loop insertion considering advanced manufacturing constraints. Finally, we propose the first machine learning-based framework for fast yet consistent SRAF generation with the high quality of results.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

As technology nodes continue to scale down, minimum feature size of the manufacturing process has been shrinking well beyond the resolution limits of the state-of-the-art photolithography tool, i.e., $193nm$ wavelength immersion lithography with complex resolution enhancement techniques (RETs). Due to the delay of emerging lithography candidates, such as extreme ultraviolet (EUV) lithography, electron-beam lithography (E-beam) and directed self-assembly (DSA), multiple patterning lithography (MPL) with complicated design-for-manufacturing (DFM) techniques has been widely adopted in semiconductor industry to further push the geometric scaling in advanced technology nodes as shown in Fig. 1.1. For example, the $14nm$ node has introduced double patterning lithography into the semiconductor manufacturing process to maintain the pitch scaling. The $10nm$ node further increases the level of design complexities and manufacturing constraints by forcing triple patterning lithography and sidewall image transfer solutions [61]. In general, there exist two types of MPL, Litho-Etch-Litho-Etch (LELE) type and self-aligned type. LELE-type of MPL splits the target layout into different masks, which allows stitch insertions and two-dimensional patterns [49,98,113,114,122], but coloring and overlay compensation schemes become extremely complicated for

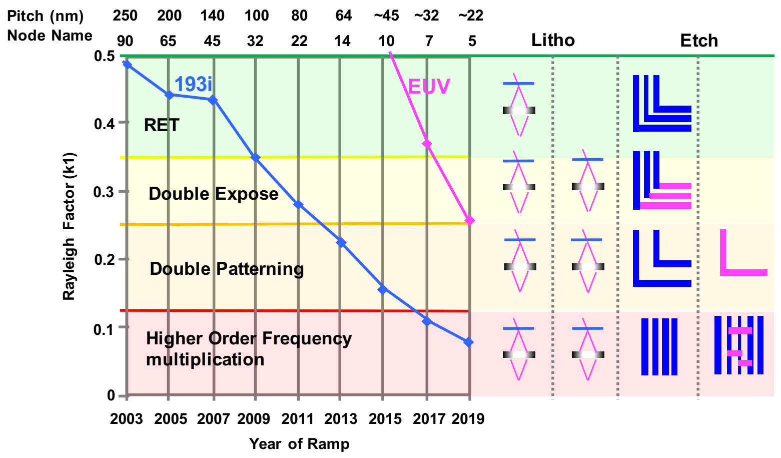


Figure 1.1: Lithography roadmap in advanced technology nodes [61].

triple patterning lithography and beyond [28, 55, 65, 99, 118, 120]. Self-aligned type of MPL depends on sidewall image transfer techniques, which can minimize electrical variations from overlay and line-edge-roughness, but induces sophisticated coloring and metal line-end constraints for two-dimensional geometries [67, 71, 93]. Regardless of different manufacturing solutions, MPL introduces a significant amount of DFM constraints and design complexities, such as layout coloring, overlay compensations and restrictive design rules, into the physical design flow.

To simplify the DFM constraints aforementioned, unidirectional layout patterns are strongly preferred for lower metal layers with high-density metal patterns in advanced technology nodes [43, 82, 84, 89, 119]. Unidirectional layout significantly reduces the coloring complexities while using a track-based coloring scheme, because each layout pattern created by the physical design tool automatically has a legal coloring solution. However, the unidirectional layout style still generates a set of challenging restrictions, such as restrictive solution space and sophisticated metal line-end rules. This further induces routing-limited scaling in the $7nm$ technology node, which means routing densities and routing resource competitions on lower metal layers are becoming increasingly high, such that design area has to be relaxed to obtain routing closure [61].

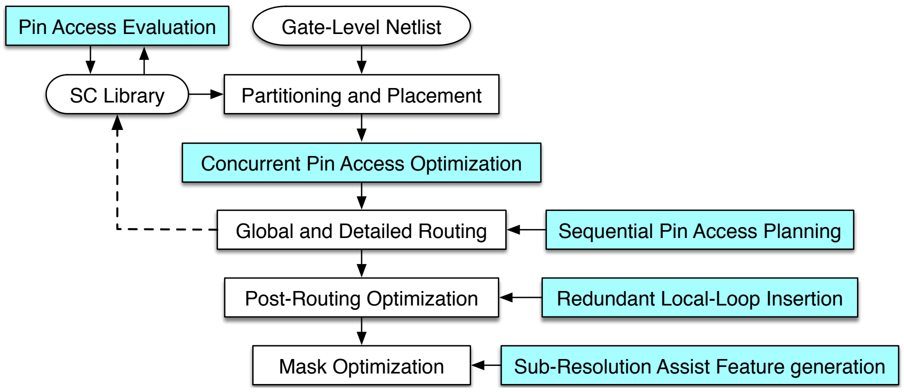
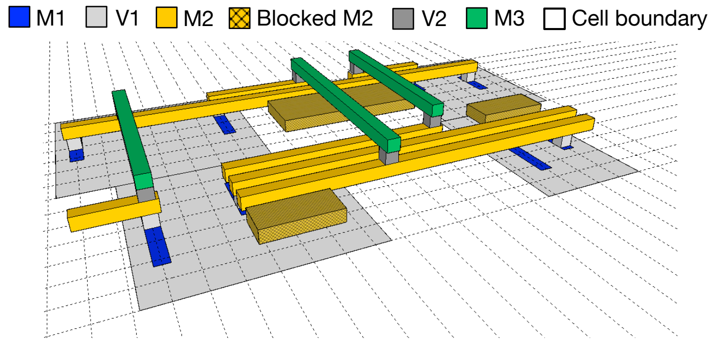

Pin Access Evaluation
Gate-Level Netlist
SC Library
Partitioning and Placement
Concurrent Pin Access Optimization
Global and Detailed Routing
Sequential Pin Access Planning
Post-Routing Optimization
Redundant Local-Loop Insertion
Mask Optimization
Sub-Resolution Assist Feature generation

Figure 1.2: Integrate proposed studies (light blue boxes) into the physical design flow in advanced technology nodes.

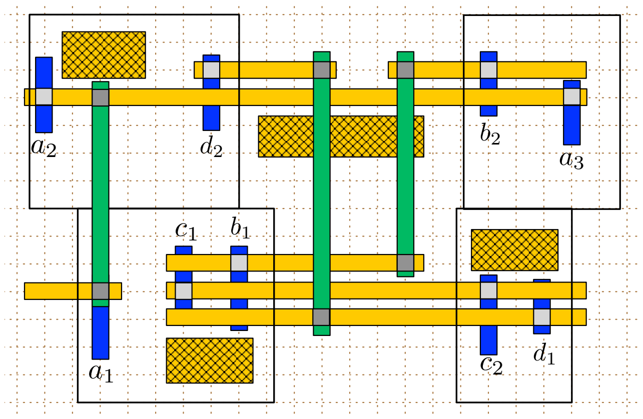
Despite the problem complexities aforementioned, a typical physical design flow relies heavily on empirical optimizations in an iterative manner as shown in Fig. 1.2, which incurs huge amounts of human intervention and turnaround time. Our study identifies the root cause lies in the restrictive solution space from unidirectional routing, which imposes significant impacts on different design stages across the standard cell (SC) library design, routing and post-routing optimization. Besides the complex DFM optimization through unidirectional layout design, mask optimization remains as the key strategy to improve the lithographic process window and the yield of the volume production in advanced technology nodes [10, 12, 30, 54, 95]. This means a holistic approach is strongly needed to obtain fast and high-quality design closure in advanced technology nodes.

For the SC library design, our study reveals that pin accessibility is a major bottleneck for unidirectional routing closure. To improve standard cell pin access, one simple option is to go through the physical design flow using a specific design instance. This would detect "hard-to-access" cells for iterative standard cell Input/Output (I/O) pin optimizations in a manual procedure as denoted by the dashed line in Fig. 1.2 [61]. This straightforward approach leads to large turnaround time and design dependency, which motivates the pin access evaluation research for fast and accurate evaluations of standard cell pin accessibility.

For the routing stage, a router is required to follow complex manufacturing rules and obtain net connections using lithography-friendly routing patterns. However, in advanced technology nodes, routing on lower metal layers is becoming more complicated than dealing with manufacturing constraints. An example of unidirectional routing on metal-2 (M2) and metal-3 (M3) layers is shown in Fig. 1.3. This routing instance includes 4 cells, 9 metal-1 pins ($a_1, a_2, a_3, b_1, b_2, c_1, c_2, d_1, d_2$) and 4 nets ($\{a_1, a_2, a_3\}$, $\{b_1, b_2\}$, $\{c_1, c_2\}$ and $\{d_1, d_2\}$). Under a routing grid with horizontal and vertical tracks, we observe severe routing resource competitions among multiple nets, especially on the



(a)



(b)

Figure 1.3: Unidirectional routing on high-density metal layers, (a) 3D view, (b) top view.

M2 layer, where only a small number of horizontal routing tracks overlaps each metal-1 pin within cell boundaries and one routing track may be shared among different pins [111, 112]. Thus, it is important to globally allocate routing resources among different pins so that all nets can be connected. In addition, routing resources on lower metal layers, such as M2 and M3, are primarily reserved for short nets. For short-net routing, via minimization is particularly important in advanced technology nodes [24]. Furthermore, compared to the previous routing studies [6,81], unidirectional routing generates a valuable opportunity to explore the solution space using interval-based search instead of grid-based search. This motivates the research for novel approaches, including sequential pin access planning and concurrent pin access optimization, to effectively resolve routing resource competitions and enable via minimizations.

For the post-routing optimization stage, the unidirectional routing style has made conventional redundant via insertion obsolete for yield improvement. This is because unidirectional routing patterns forbid off-track routing, i.e., wire bending, for the metal coverage of redundant vias. To enhance the yield of unidirectional routing patterns, we identify redundant local-loop insertion as a new way of inserting redundant vias due to its compatibility with the unidirectional layout style [8, 15]. However, redundant local loop insertion introduces complex long-range conflict constraints and timing impacts are no longer negligible for certain redundant local loop candidates. This motivates the research for global optimization techniques to achieve better performance than the traditional approaches.

For the mask optimization stage, to date, the $193nm$ wavelength immersion lithography with low $k1$ value is still the mainstream technique to achieve smaller feature size. However, low image contrast and complex target pattern shapes make it extremely difficult for low-$k1$ lithography to obtain acceptable lithographic process windows [86]. Besides the complex DFM techniques, mask optimization through resolution enhancement techniques remains as the key strategy to improve the lithographic process window and the yield of volume production in advanced technology nodes [10,12,30,54,95]. Among them, SRAF generation is particularly important to improve the lithographic process window of target patterns. Conventional SRAF generation approaches suffer from large computational overhead or lithographic performance degradations. This motivates the research to adopt machine learning-based approaches for SRAF generation, which demonstrates a new perspective to achieve promising trade-offs between computational efforts and lithographic performance.

This chapter first briefly summarizes the aforementioned physical design challenges in advanced technology nodes. We further overview this dissertation in terms of novel solutions to advanced physical design challenges.

1.1 Advanced Physical Design Challenges

Standard Cell Optimization and Evaluation Standard cell library design involves significant amounts of human interventions because the industrial routine follows full custom design and optimization. Typically, designers depend on figures of merit from placed and routed digital blocks to iteratively assess and improve the quality of a standard cell library, which incurs large turnaround time for the design closure. Therefore, an automated standard cell library optimization and evaluation tool is crucial for the designers to obtain quick feedback on the layout design quality considering complex DFM constraints.

Pin Access and Routing Co-Optimization With unidirectional routing, only a small number of routing tracks are available to access each standard cell I/O pin, which further leads to severe pin access interference, i.e., routing interference caused by pin access, on lower metal layers. The pin access interference significantly degrades the routing solution quality, which makes pin access and routing co-optimization extremely important for unidirectional routing closure in advanced technology nodes.

Redundant Local-Loop Insertion The conventional redundant-via insertion with two-dimensional metal patterns has become obsolete under restrictive unidirectional routing style, because the off-track metal patterns covering redundant vias are strictly forbidden. To enhance the yield of metal and via patterns, redundant local-loop insertion is a new way of inserting redundant vias that is compatible with unidirectional routing but induces complicated long-range conflict constraints, which typically require global optimization techniques to resolve.

Sub-Resolution Assist Feature Generation Regardless of complex DFM techniques, mask optimization through resolution enhancement techniques remains as the key strategy to improve the yield of the semiconductor manufacturing processes. Among them, Sub-Resolution Assist Feature (SRAF) generation is a very important method to improve lithographic process windows. Conventional SRAF generation approaches suffer from large computational overhead or lithographic performance degradations. We propose machine learning-based approaches for fast and high-quality SRAF generations.

1.2 Overview of this Dissertation

This dissertation proposes novel CAD algorithms and methodologies to systematically enable standard cell optimization and physical design in advanced technology nodes.

Chapter 2 presents the first cell-level evaluation engine to quantify the pin accessibility of a given standard cell library. The proposed study is based on local routing pattern enumeration and optimization, which has been prototyped by the leading IP vendor, ARM Inc.

Chapter 3 proposes pin access and routing co-optimization techniques to enhance the routing solution quality in advanced technology nodes. To efficiently resolve the severe pin access interference, we first propose pin access planning schemes to improve the pin accessibility during a sequential routing procedure. We further propose concurrent pin access optimization techniques to obtain fast and high-quality routing solutions. We demonstrate that the pin access and routing co-optimization studies can provide much better solution qualities than state-of-the-art manufacturing-friendly routers.

Chapter 4 presents the first global optimization engine for redundant local-loop insertion with bounded timing impact while accommodating advanced manufacturing constraints. This proposed study features an optimal integer linear programming formulation and scalable iterative linear programming solving with an incremental search scheme.

Chapter 5 proposes a machine learning-based framework for sub-resolution assist feature generation. The classification-based learning model can deliver competitive lithography performance with significant speed up compared with a commercial setup for SRAF generation. The proposed SRAF generation approach has been prototyped within the memory division in Toshiba Microelectronics.

Chapter 6 summarizes this dissertation and discusses potential future research directions.

Chapter 2

Standard Cell Optimization and Evaluation

2.1 Introduction

Due to the resolution limits of 193nm photolithography, multiple patterning techniques and unidirectional layout have been widely used to extend semiconductor process technology scaling [83, 93, 117]. The design rules that enable multiple patterning (color decomposition, forbidden pitches, etc.) are much more restrictive than the basic rules used previously in technology nodes larger than 20nm. In addition, the expectation to continue Moore's Law translates to the same density and area scaling every node. That means the physical design tools need to access Standard Cell (SC) Input/Output (I/O) pins in more congested areas with increasingly restrictive rules.

One way that SC designers can assist physical design tools is through intelligent, optimized SC I/O pin design. Unfortunately, the complex design rules and neighbor interactions that exist due to various multiple patterning techniques like Litho-Etch-Litho-Etch (LELE) and Self-Aligned Double Patterning (SADP) make human-driven layout almost impossible at 14nm tech-

This chapter is based on the journal "Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Z Pan. Self-aligned double patterning aware pin access and standard cell layout co-optimization. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 34(5):699712, 2015." I am the main contributor in charge of problem formulation, algorithm development and experimental validations.

nologies and below. That means automated SC layout design and optimization are needed to provide flexible I/O pin access.

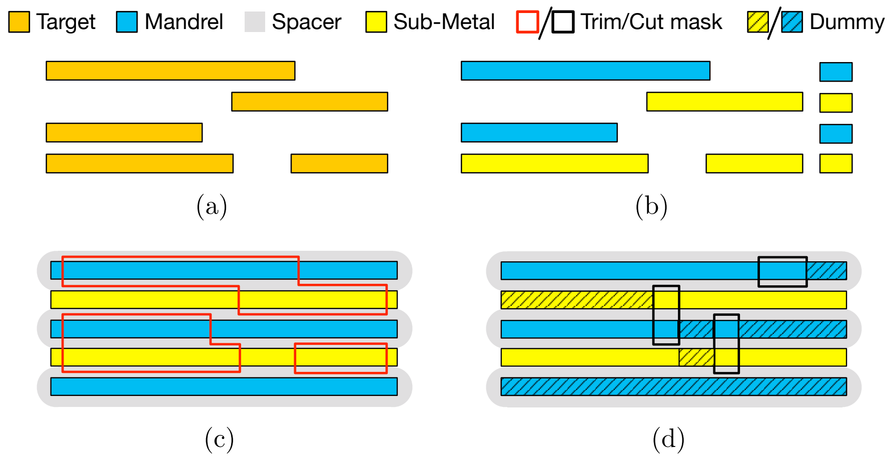
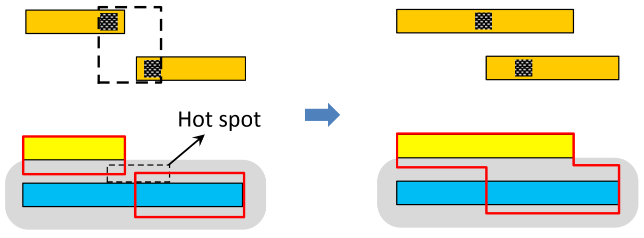


Figure 2.1: Track-based coloring for unidirectional layout patterns, (a) unidirectional target patterns, (b) track-based coloring, (c) SADP with trim masks, (d) SADP with cut masks.

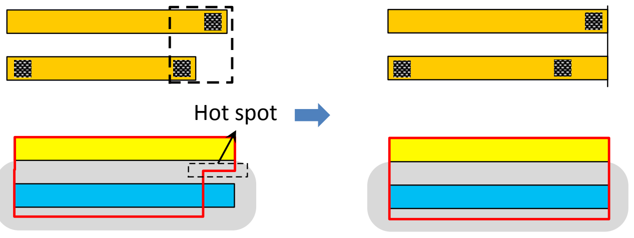
SADP, in particular, is a viable candidate for lower layer metalization with regular patterns at the 10nm technology node, due to better overlay and Line Edge Roughness (LER) [72] control compared to LELE. To deploy the SADP technique for routing layers in practical designs, designers need to ensure that layout patterns are SADP-friendly to achieve successful layout decomposition. The SADP layout decomposition problem has been studied, as shown in [11, 31, 103, 123]. For the regular or unidirectional layout, the line-space array decomposition method can efficiently decompose SADP-based geometries and achieve good pattern fidelity and process margin [67,93]. An example of line-space array decomposition is demonstrated in Fig. 2.1. Fig. 2.1(a) shows regular layout on horizontal tracks. We can assign different colors to patterns on neighbor tracks in Fig. 2.1(b). With spacer defining the gap between neighboring patterns, the trim mask and cut mask obtain the target layout as demonstrated in Fig. 2.1(c) and Fig. 2.1(d), respectively. For the cut mask solution in Fig. 2.1(d), dummy patterns may be created without affecting the underlying electrical connections.

To incorporate SADP constraints into early design stages, there are several studies [26, 29, 52, 76] dealing with the SADP-aware routing problem. However, to date, works studying how multiple patterning and decomposition impact SC I/O pin design are lacking, especially as pin congestion and routability become increasingly critical to the overall physical design results. Since most modern-day SC designs primarily use Metal-1 (M1) for local connections and I/O pins, Metal-2 (M2) design is essential for SC I/O pin access. There are several works [80, 81] on detailed routing accounting for pin access. [80] focuses on the gridless routing and proposes efficient algorithms to improve the pin accessibility and routability. [81] presents the formulation and solution for the detailed routing for dense pin clusters. None of these previous studies explicitly addresses the pin accessibility problem at the SC level under complex SADP-specific constraints.

SADP-based M2 wires, in particular, present a new set of problems to SC I/O pin access. Specifically, because of the decomposition of SADP into the mandrel and trim masks, one cannot simply rely on via locations to determine line-end positions of M2 wires. As shown in Fig. 2.2, SADP yield can be enhanced by line-end extensions that are dependent on both via



(a)



(b)

Figure 2.2: Line-end extension techniques, (a) anti-parallel line ends, (b) parallel line ends.

placement and neighboring wire placement. For example, in Fig. 2.2(a) and (b), respectively, the extension of anti-parallel line ends and parallel line-end alignment both help to avoid hot spots on the trim masks [71].

In general, the ideal location of geometries is not as straight-forward under SADP constraints and is more dependent on the neighborhood around the geometry in question. This means all SADP-based metal designs, including pin access and within-cell connections have to be optimized simultaneously during the I/O pin design phase. Previously, line-end extension techniques have been deployed to deal with the cut mask manufacturability in [124, 125]. To minimize the effective gaps in the layout, [125] proposes a greedy optimal algorithm to extend the line ends of metal wires. [124] presents a mask cost reduction formulation to improve the layout manufacturability with the line-end extension techniques. Alternatively, our study enables trim mask friendly, SC-level pin access design under SADP constraints, a priori, before the SC library is ever used in a placed-and-routed block.

In addition, the Via-1 layer is used to connect the M2 wires and M1 I/O pins. Due to the high density of Via-1's in future technology nodes, multiple patterning, e.g., LELE, is needed to manufacture the Via-1 layer. Hence, for the 10nm technology used in this study, we need to enable LELE friendly Via-1 design to achieve legal pin access at the SC level. Determining the double patterning decomposability of given layout patterns is a well-studied topic [49, 98, 113, 114, 122]. If the conflict graph deduced from the given layout patterns does not contain odd cycles, the layout patterns are double patterning (LELE) friendly. The odd-cycle free LELE-aware Via-1 design is equally important to determining SC pin access as SADP-friendly M2 design.

This chapter proposes the first study on the SC-level I/O pin access problem and prototype our methodology on the predictive 10nm SC library. To solve this problem efficiently, we present a Depth First Search (DFS)-based method to achieve LELE-friendly Via-1 assignment and a Mixed Integer Linear Programming (MILP)-based technique that simultaneously optimizes the M2 wires for pin access and within-cell connections of SCs. Moreover, using the backtracking method, we extend this technique to each pin access scenario and maximize the pin accessibility of the predictive 10nm library.

2.2 Preliminaries

2.2.1 Standard Cell Architecture

Due to complex design-for-manufacturing constraints, there are finite options of Standard Cell (SC) architecture for a specific advanced technology node [18, 61, 92]. Fig. 2.3 illustrates a 7.5-track SC architecture at $7nm$ technology node. An SC architecture pre-determines the cell height, which is typically an integer multiple of the M2 routing pitch. The SC boundary decides the region of intra-cell layout patterns on M1 and lower manufacturing layers, which are used for intra-cell connections and SC I/O pins. While occasional M2 usages are allowed for intra-cell connections for complex SCs [105], M2 and upper layers are mainly used for routing connections. Horizontal M2 tracks in the middle of Fig. 2.3(a) are routing tracks used by the router for cell-to-cell I/O connections. The M2 tracks on the top and bottom of SC boundary typically align with the power/ground rails of a specific design, providing a robust power/ground network. The M1 I/O pins of "Inverter" and "AND-OR-Inverter" are shown in Fig. 2.3(b) and Fig. 2.3(c), respectively. In the routing level, these M1 I/O pins can only be accessed with fixed number of M2 routing tracks. As M2 routing patterns are purely unidirectional, there exists a limited number of M2 accessing points (see Fig. 2.7(b)) for each I/O pin, which makes routing density and resource competition increasingly high on lower metal layers. This also makes SC pin access optimization on lower metal layers particularly important for unidirectional routing closure.

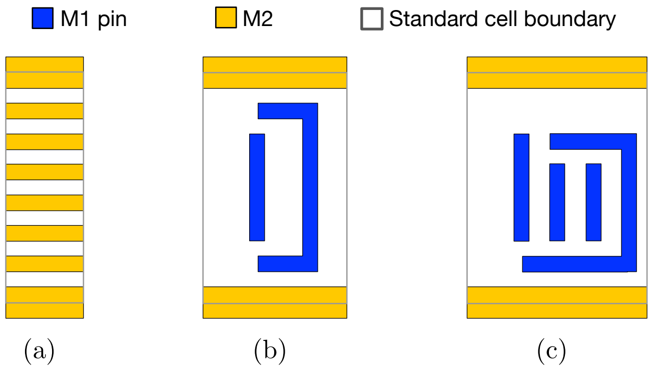


Figure 2.3: SC architecture, (a) 7.5 track architecture, (b) "Inverter" layout on M1 and M2, (c) "AND-OR-Inverter" layout on M1 and M2.

2.2.2 Track-based Coloring

The continued geometric scaling of process technology depends on multiple patterning and increased layout regularity [93]. Thus, at the 10nm node and beyond, we assume that the M2 layout will be extremely regular. Furthermore, in the 10nm commercial technology we used, the preferred direction of M2 routing is horizontal. After studying the M2 routing tracks, we make the following observation.

Observation 1. *A single color is assigned to metal patterns on even routing tracks. The alternate color is then assigned to metal patterns on odd routing tracks.*

M2 routing is becoming increasingly congested as we continue to scale towards the 10nm technology node because of the increasing density of transistors and SC I/O pins. Hence, increasing M2 congestion leads to a higher likelihood of having M2 wires on neighbor tracks at the same time, which

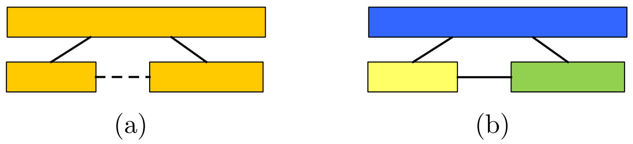


Figure 2.4: The potential odd-cycle conflicts of the coloring graph, (a) potential M2 layout, (b) potential odd-cycle conflicts.

leads to the layout patterns illustrated in Fig. 2.4. In Fig. 2.4(a), a solid edge denotes a coloring conflict and a dashed edge denotes a potential coloring conflict. Fig. 2.4(b) demonstrates how a dashed edge changes to solid edge if we assign different colors to M2 wires on the same track, which leads to an odd-cycle conflict in the coloring graph. The SADP technique does not allow stitches during the layout decomposition stage, which means odd-cycle conflicts must be strictly forbidden in SADP-friendly layout patterns [11]. The color assignment strategy from Observation 1 avoids potential odd cycles in the coloring stage for SADP-friendly layout.

2.2.3 Design Rule Formulation and Legalization

Table 2.1 defines SADP-specific notations for the $10nm$ technology node [66]. First, the minimum area constraint of the M2 layout is converted to the minimum wire length design rule (l_0) due to the fixed width of M2 wires. Then, we define the space between M2 line ends on the same routing track as **OnTrackSpace(l_1)**, as shown in Fig. 2.5(a). We use **OffTrackOverlap(l_2)** and **OffTrackSpace(l_3)** to define the prohibited region for the anti-parallel M2 wires [66], as shown in Fig. 2.5(b) and Fig. 2.5(c), respectively. Finally,

Table 2.1: Notations for design rule formulation

M2 layer	MetalWidth	w
	MetalSpace	s
	minMetalArea	A_{min}
Spacer deposits	SpacerDepositWidth	w_{sp}
Trim mask	minResistWidth	w_r
	minResistSpace	s_r
	EtchBias	w_e
Design rules	minMetalLength (*Rule* 0)	$l_0 \geq \frac{A_{min}}{w}$
	OnTrackSpace (*Rule* 1)	$l_1 \geq w_r - 2 \cdot w_e$
	OffTrackOverlap (*Rule* 2)	$l_2 \geq s_r + 2 \cdot w_e$
	OffTrackSpace (*Rule* 3)	$l_3 \geq \sqrt{(w_r - 2 \cdot w_e)^2 - {w_{sp}}^2}$
	OffTrackOffset (*Rule* 4)	$l_4 \geq w_r \quad \text{or} \quad l_4 = 0$

for parallel M2 wires illustrated in Fig. 2.5(d), the line-end design constraint is defined as **OffTrackOffset(l_4)**. Table 2.1 summarizes the design rules for M2 layout patterns [66, 71].

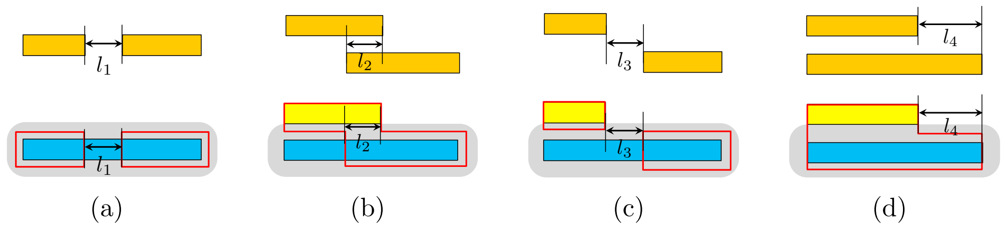


Figure 2.5: Design rule formulation, (a) OnTrackSpace, (b) OffTrackOverlap, (c) OffTrackSpace, (d) OffTrackOffset.

The minimum coloring distance (d_0) and minimum different mask distance (d_1) are introduced to enable LELE friendly Via-1 assignment. Suppose successful layout decomposition of Via-1 patterns from Fig. 2.6(a) to Fig. 2.6(b), d_0 denotes the minimum center to center spacing of Via-1's on the same mask and d_1 denotes the minimum center to center spacing of Via-1's on different masks.

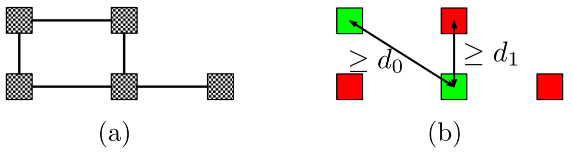


Figure 2.6: (a) Via-1 patterns, (b) layout decomposed to two masks and Via-1 design rules.

2.3 Problem Formulation

The SC-level pin access optimization involves both LELE-aware Via-1 assignment and SADP-aware M2 wire design. Our I/O pin access design and optimization is based on the practical layout of SCs. For each cell in the library, we observe that I/O pins generally exist on either the M1 or M2 layer. To properly formulate the pin access design and optimization problem, we have the following definitions.

Definition 1 (Hit Point)**.** *The overlap of an M2 routing track (which is pre-determined by the place and route tool) and an I/O pin shape is defined as a Hit Point for that particular I/O pin.*

Each hit point determines the range of positions for the corresponding Via-1 (the M1 to M2 connection). In the 10nm technology used, M2 is unidirectional and runs horizontally. Thus, for each hit point, there are two accessing directions possible, either from left to right or from right to left. To access one hit point, we first need to determine the legal location for the Via-1 connection. Then, we need to design the M2 wire assuming one accessing direction for the hit point. In order to connect to every I/O pin in a cell, we need to determine the accessing directions for a set of hit points. Hence, we have several definitions as follows.

Definition 2 (Hit Point Combination)**.** *A set of hit points (with a defined access direction, left or right) where each I/O pin in the SC is accessed exactly once is defined as a Hit Point Combination for that cell.*

Definition 3 (Valid Hit Point Combination)**.** *If a hit point combination induces zero design rule violations, it is considered a Valid Hit Point Combination. Otherwise, it is considered to be invalid.*

Definition 4 (Valid Hit Point)**.** *If a hit point can be accessed from both directions within some valid hit point combinations for one cell, it is considered a valid hit point. Otherwise, it is considered to be invalid.*

An example of the I/O pin access design for one hit point combination is shown in Fig. 2.7. Fig. 2.7(a) demonstrates the M1 I/O pin layout and M2 routing tracks running horizontally above M1. Fig. 2.7(b) shows how the hit points, which represent valid Via-1 locations for I/O pin access, are derived from the overlap of M2 routing tracks and the M1 I/O pins. For most hit points, the length of the hit point is short because it is decided by the minimum width of the vertical M1. However, if an M1 wire runs horizontally, this leads to a long hit point, which allows more flexibility for the Via-1 position. The

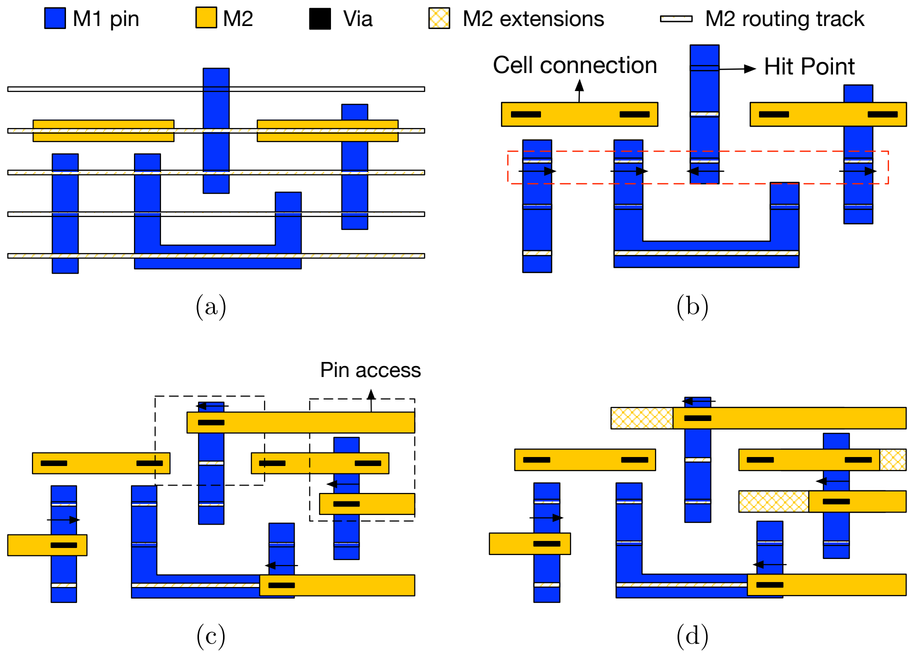


Figure 2.7: SC pin access optimization, (a) SC I/O pins and M2 routing tracks, (b) hit points and M2 within-cell connections, (c) a hit point combination with M2 pin access, (d) SADP-friendly pin access with M2 extensions.

set of hit points within the dashed box in Fig. 2.7(b) shows one hit point combination and the dark arrows denote the arbitrary accessing directions chosen for the hit points. Fig. 2.7(c) illustrates one way to access the cell using that hit point combination. After choosing one hit point for each I/O pin and the accessing direction for that hit point, we need to determine the legal Via-1 locations for that hit point combination such that the final Via-1 patterns are LELE friendly. Given the LELE-legal Via-1 assignment, the M2 wires can be designed for pin access, accounting for the minimum enclosure design rule for M2 over Via-1. The dashed boxes in Fig. 2.7(c) denote all pairs of line ends that cause hot spots in trim mask designs. Fig. 2.7(d) demonstrates that line-end extension techniques can be adopted to fix those hot spots in the trim mask.

In SC design, it is important to achieve both LELE-aware Via-1 assignment and SADP-aware M2 design, i.e., all hot spots are fixable via line-end extension techniques. However, the engineering efforts and iterations involved to fix all of the potential hot spots across the SC library are too large for the average layout design team. Therefore, a general methodology is needed to determine optimal Via-1 assignment and design the M2 wires for pin access and within-cell connections simultaneously. We can now define the SADP-aware pin access optimization problem as follows.

Problem 1 (Pin Access Optimization (PAO))**.** *Given the SC layout and a specific hit point combination, determine whether or not it is possible to achieve a legal LELE-friendly Via-1 assignment and subsequently optimize the M2 wires for pin access and within-cell connections given the Via-1 assignment under SADP constraints. If possible, show legal Via-1 assignment and all SADP-friendly M2 wires.*

As shown in Fig. 2.7(b), we may have multiple hit points for one I/O pin, which leads to numerous hit point combinations for one cell. For one SC, we define the pin access and cell layout co-optimization problem as follows.

Problem 2 (PICO)**.** *Given the SC layout, the Pin Access and Cell Layout Co-Optimization (PICO) problem is to show all Via-1 assignment and M2 wiring* *cases with successful PAO's and maximize the pin access flexibility under LELE and SADP constraints.*

2.4 Pin Access Optimization

Given a hit point combination, we seek the LELE-friendly Via-1 assignment, based on which we pre-design the M2 wires for pin access. Then, we propose an MILP-based method to enable efficient SADP-aware M2 design. Finally, the Linear Programming (LP) relaxation is presented to explore the trade-off between runtime and performance for the pin access optimization.

Table 2.2: Notations for via assignment

v_k	the k^{th} Via-1 for cell connections
hp_k	the k^{th} hit point within given hit point combination
hp_k^m	the m^{th} candidate Via-1 location for k^{th} hit point
S_v	the LELE-Aware Via-1 assignment
S_v^o	the optimal LELE-Aware Via-1 Assignment
g_0	the Via-1 grid size

2.4.1 Via Assignment

In general, there are two kinds of Via-1's at the SC level. The first kind is for internal M1 to M2 (within-cell) connections due to the complexity of SC layout in advanced technology nodes. These connections cannot be modified in the pin access optimization stage. The other kind is for the pin access connections from M2 wires to M1 I/O pins, which is the design target in the Via-1 assignment problem. For instance, in Fig. 2.8(a), there are four internal

Via-1's denoted as $v_1 - v_4$ and our target is to determine the location of Via-1's for the four hit points denoted as $hp_1 - hp_4$. We can drop the Via-1's in the center of short hit points, such as hp_1, hp_2 and hp_4. However, long hit points, such as hp_3, allow for more flexibility for the Via-1 location. We can build the conflict graph for the Via-1 layer given a specific hit point combination, as illustrated in Fig. 2.8(b). We put a solid edge between conflicting Via-1's of which locations are decided. A dashed edge is added when the location of some Via-1 is not decided and there exists a potential conflict between two Via-1's. For example, the Via-1 associated with the long hit point in Fig. 2.8(b) always conflicts with the Via-1 for hp_2 , but only conflicts with that for hp_1 when dropped on the left end and conflicts with that for hp_4 when dropped on the right end.

To make the best use of the flexibility from the long hit point, we propose a grid-based segmentation, which leads to multiple potential Via-1 candidates for a hit point. For example, we can see several Via-1 candidates for hp_3 in Fig. 2.8(c). The grid size depends on the granularity needed and is constrained by the manufacturing grid imposed by a particular technology node, i.e., the grid size chosen must be greater-than-or-equal-to the manufacturing grid. Related notations are given in Table 2.2. There is an accessing direction assigned to each hit point within the given hit point combination. Hence, we prefer the legal Via-1 location to be on the right end of hp_3 because the hit point will be accessed from right to left. The right most legal Via-1 candidate for hp_3 leads to shortest M2 wire to access that hit point. To dif-

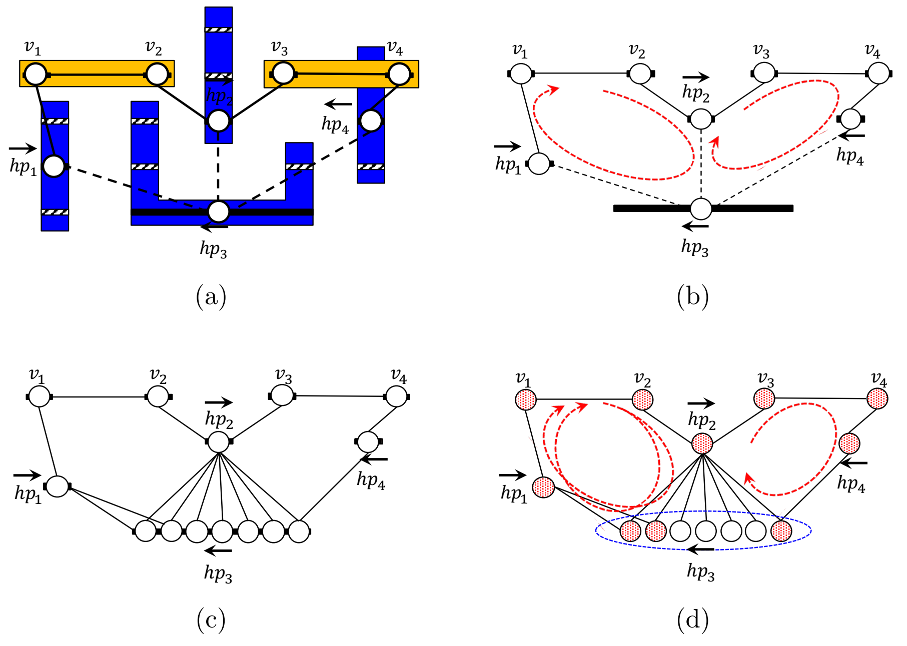


Figure 2.8: (a) SC I/O pins and Via-1's for within-cell connections, (b) conflict graph for hit points and Via-1's, (c) conflict graph with grid-based segmentation for long hit points, (d) odd cycles in the conflict graph.

ferentiate multiple Via-1 candidates for the long hit point, we assign a cost to each candidate according to the accessing direction and distance from the Via-1 location to the left/right end of the hit point. For instance, the cost of the right most Via-1 for the long hit point in Fig. 2.8(c) will be 0 and the cost of the second right most Via-1 will be g_0. In general, from right to left, the cost of the n^{th} Via-1 will be $(n-1)*g_0$. We assign zero cost to internal Via-1's and Via-1's associated with short hit points. After the grid-based segmentation and cost assignment for each Via-1 candidate, we can build the whole conflict graph for the Via-1 layer, as illustrated in Fig. 2.8(d). There is no conflict among Via-1 candidates for the same hit point.

The overall objective of this step is to choose a set of Via-1 locations, including existing Via-1's for internal cell connections and target Via-1's for each hit point, such that the total cost is minimized and the Via-1 layer is LELE compliant. It is a well-established argument that the layout is LELE compliant if and only if the conflict graph deduced does not contain odd cycles [49]. This means that odd cycles are forbidden in the optimal legal Via-1 assignment.

Algorithm 1 demonstrates the details for achieving optimal LELE-friendly Via-1 assignment. Lines 3-16 explain how to build the conflict graph given existing Via-1's for cell connections and a specific hit point combination. In line 18, all of the odd cycles in the conflict graph are reported (e.g., the 3 odd cycles in Fig. 2.8(d) would be reported). If an odd cycle is detected in the conflict graph, the conflicting set of Via-1 candidates cannot be chosen simultaneously during the legal Via-1 assignment. [49] deploys the Breadth First Search (BFS) and double linked list to iteratively report odd cycles. In this case, our target is different because we need to find all of the odd cycles simultaneously in a given conflict graph. Hence, the Depth First Search (DFS) technique is used. We keep track of the DFS stack while coloring the conflict graph. Once a coloring conflict is reported for double patterning, we back-track the DFS stack and report the cycle associated with the current coloring conflict, which is repeated until all nodes are visited. The optimal Via-1 assignment is achieved in line 20 with a depth-first-search (DFS) technique. The optimal Via-1 assignment is an empty set if no legal Via-1 assignment can be found, as explained in lines 21-25. Assume n hit points for the input hit point combination and the number of valid Via-1 candidates for the i^{th} hit point is $k_i, 0 \leq i \leq n-1$, the total number of Via-1 combinations enumerated will be $\Pi_{i=0}^{i=n-1} k_i$. Due to the linear time complexity of DFS, the theoretical runtime bound for Algorithm 1 will be $O(k^n), k = \max_i k_i$. The time complexity of Algorithm 1 grows exponentially with the number of I/O pins, but most SC libraries have the cell I/O pin counts bounded to the order of 10. This means that for a typical SC library, Algorithm 1 can execute in a reasonable runtime.

Algorithm 1 Via-1 Assignment Algorithm

```
Input: a set of Via-1's for cell connections ($S_c$), a hit point combination ($HPC$) and Via-1
    grid size ($g_0$);
 1: Define $CFGraph$ as the conflict graph for Via-1 layer;
 2: Define $S_{via}$ as the 2-D set of Via-1's;
 3: for each element $hp$ in $HPC \cup S_c$ do;
 4:     Define $S_{pos}$ as the set of Via-1's for $hp$;
 5:     if $hp$ is a long hit point then;
 6:         Segment $hp$ in the grid size $g_0$;
 7:         Push the center of each segment to $S_{pos}$;
 8:     else
 9:         Push the center of $hp$ to $S_{pos}$;
10:     end if
11:     for each via location $pos$ in $S_{pos}$ do;
12:         Add $pos$ to $CFGraph$ and assign cost;
13:         Detect and add conflict edges;
14:     end for
15:     Push $S_{pos}$ to $S_{via}$
16: end for
17: Define $oddCycles$ as the 2-D set of nodes in $CFGraph$;
18: Report all odd cycles in $CFGraph$ to $oddCycles$;
19: Define $S_v^o$ as the optimal Via-1 assignment;
20: $S_v^o$ = DFS($S_{via}$, $oddCycles$);
21: if $S_v^o = \emptyset$ then;
22:     $HPC$ is invalid;
23: else
24:     $S_v^o$ is the optimal legal Via-1 assignment;
25: end if
```

Table 2.3: Notations for metal pattern optimization

c_L, c_R	left or right boundary of cell
c_W	cell width, $c_W = c_R - c_L$
S_m	set of M2 wires
n	total number of M2 wires
S_k	set of pairs of wires for rule $k, \forall k \in \{1, 2, 3, 4\}$
x_{iL}, x_{iR}	the left or right line end of i^{th} wire
x_{iL}^0, x_{iR}^0	the initial line ends of i^{th} wire

The LELE-aware Via-1 assignment yields a legal Via-1 location for each hit point within the hit point combination. Furthermore, the solution of Algorithm 1 decides the boundary condition for the SADP-aware pin access design of the M2 layer.

2.4.2 Metal Pattern Optimization

Given optimal Via-1 assignment and the accessing direction for the corresponding hit point, we can determine the line-end position of the M2 wire for pin access accounting for the minimum enclosure design rule for M2 over Via-1. For pin access design, we focus on SADP-aware layout optimization within an SC boundary. Hence, if one hit point is next to the right boundary of a cell and the access direction is from the right, the right line end of the corresponding M2 wire will be extended to the right boundary. We have similar pre-design if the hit point is accessed from the left boundary of a cell. Fig. 3.1(c) is an example of M2 wires for pin access after the pre-design stage. This is the most common hit point access scenario from the SC perspective. Our primary goal is to achieve the first-order pin accessibility figure-of-merit. Hence, we exclude special scenarios like Metal-3 wire going down directly through two vias (Metal-3 to M2 and M2 to M1), and also pin access through complex maze routing, instead of straight routing. The pre-design method induces the following observation.

Observation 2. *For M2 wires after the pre-design stage, right line ends can only be extended to the right and left line ends can only be extended to the left.*

As illustrated in Fig. 3.1(c), if we simply use the Via-1 locations to determine the line end of M2 wires, the SADP constraints may invalidate some hit point combinations. The line-end extension techniques enable us to legalize the M2 layout and ensure SADP-friendly design. The conventional layout migration issue has been formulated as a linear programming problem in [42]. A similar approach has also been deployed to deal with LELE double patterning layout decomposition in [32]. In addition, the line-end extension techniques have also been deployed in [124, 125] to improve the manufacturability of the cut mask process instead of the trim mask process in this work. [125] has proposed a greedy optimal algorithm to minimize the effective gaps in the cut mask process. The constrained shortest path algorithm is presented to optimize the cut mask cost. Instead of mask cost reduction, this study is design rule oriented and guarantees to find an optimal solution with the minimum amount of line-end extensions if a feasible solution exists for the given Via-1 assignment from Section 2.4.1. The greedy algorithm in [125] aims at the gap distribution optimization in cut masks, which is different from the trim mask optimization in this work. The techniques used in [32, 42, 124] cannot be directly applied to SADP-aware I/O pin access design because the relative order of the metal line ends may change during the line-end extension stage, as shown in Fig. 2(a). Instead, we propose an MILP-based optimization methodology to determine the M2 wire design for each specific hit point combination. Table 2.3 shows the related notations. We will first give the mathematical formulation for the SADP-aware pin access problem. Then, we transfer the mathematical formulation to an MILP formulation. The results of the MILP can determine whether feasible solutions exist for the M2 line ends of a particular hit point combination. If feasible solutions exist, the line-end positions of each M2 wire are decided while minimizing the total amount of line-end extensions.

Mathematical Formulation Observation 2 allows us to quantify the total amount of extension in terms of line-end positions. It is known that line-end extension techniques benefit SADP-based wires [71]. However, in next generation technology nodes, the routing resources are becoming increasingly limited, so line-end extensions of M2 wires should be used judiciously. Additionally, line-end extensions can potentially increase both coupling capacitance and ground capacitance on M2 routes. Therefore, line-end extension minimization is a necessity for pin access optimization. The minimization of the total amount of line-end extensions is formulated as the objective function, as shown in (1).

Constraints (1a) - (1c) define the line-end extension limits and minimum wire length design rule (*Rule* 0 in Table 2.1) for each M2 wire. The initial relative order can be determined for each pair of M2 wires. Suppose the i^{th} wire is on the left of the j^{th} wire, as demonstrated in Fig. 2.5(a). Constraint (1d) is formulated to define *Rule* 1. In set S_2, the line ends originally overlap each other and constraints (1e) and (1f) represent *Rule* 2. In set S_3, the line ends initially have no overlap. After extension, the line ends may or may not overlap each other. Constraint (1g) satisfies *Rule* 2 or *Rule* 3. Then, constraints (1h) and (1i) are formulated to specify *Rule* 4 for each pair of M2 wires in set S_4.

$$
\begin{aligned}
&\min \sum_{i=0}^{n-1} (x_{iL}^0 - x_{iL}) + (x_{iR} - x_{iR}^0) && && (1)\\
\text{s.t. } & c_L \le x_{iL} \le x_{iL}^0 && \forall i \in S_m && (1a)\\
& x_{iR}^0 \le x_{iR} \le c_R && \forall i \in S_m && (1b)\\
& x_{iR} - x_{iL} \ge l_0 && \forall i \in S_m && (1c)\\
& x_{jL} - x_{iR} \ge l_1 && \forall (i,j) \in S_1 && (1d)\\
& x_{iR} - x_{jL} \ge l_2 && \forall (i,j) \in S_2 && (1e)\\
& x_{jR} - x_{iL} \ge l_2 && \forall (i,j) \in S_2 && (1f)\\
& x_{jL} - x_{iR} \ge l_3 \text{ or } x_{iR} - x_{jL} \ge l_2 && \forall (i,j) \in S_3 && (1g)\\
& |x_{iL} - x_{jL}| \ge l_4 \text{ or } x_{iL} - x_{jL} = 0 && \forall (i,j) \in S_4 && (1h)\\
& |x_{iR} - x_{jR}| \ge l_4 \text{ or } x_{iR} - x_{jR} = 0 && \forall (i,j) \in S_4 && (1i)
\end{aligned}
$$

MILP formulation Next, we explain how to convert (1) into an MILP formulation. We can simplify the objective function by omitting item x_{iL}^0

and x^0_{iR}, which are constants for a specific hit point combination. We also need to convert constraints ($1g$)-($1i$) to linear constraints based on the big-M transformation [5]. Note that $|x_{iL} - x_{jR}| \leq c_W, \forall i, j \in S_m$ and c_W is the width of the cell. Hence, at the SC level, the cell width c_W is an appropriate big-M parameter for the formulation. Constraint ($1g$) can be formulated as linear constraints ($2a$) - ($2c$) given below. s_k is an additional integer variable introduced so that both constraints can be satisfied at the same time.

$$x_{jL} - x_{iR} + (c_W + l_3)\cdot s_k \geq l_3 \tag{2a}$$

$$x_{iR} - x_{jL} + (c_W + l_2)\cdot (1 - s_k) \geq l_2 \tag{2b}$$

$$s_k \in \{0, 1\} \qquad \forall (i, j) \in S_3 \tag{2c}$$

Similarly, constraints ($1h$) and ($1i$) can also be converted to linear constraints by introducing integer variables as follows.

$$x_{jL} - x_{iL} + (c_W + l_4)\cdot s_{m1} \geq l_4\cdot (1 - t_{n1}) \tag{2d}$$

$$x_{iL} - x_{jL} + (c_W + l_4)\cdot (1 - s_{m1}) \tag{2e}$$

$$\geq l_4\cdot (1 - t_{n1}) + (c_W + l_4)\cdot t_{n1} \tag{2f}$$

$$s_{m1} + t_{n1} \leq 1, s_{m1}, t_{n1} \in \{0, 1\} \qquad \forall (i, j) \in S_4 \tag{2g}$$

$$x_{jR} - x_{iR} + (c_W + l_4)\cdot s_{m2} \geq l_4\cdot (1 - t_{n2}) \tag{2h}$$

$$x_{iR} - x_{jR} + (c_W + l_4)\cdot (1 - s_{m2}) \tag{2i}$$

$$\geq l_4\cdot (1 - t_{n2}) + (c_W + l_4)\cdot t_{n2} \tag{2j}$$

$$s_{m2} + t_{n2} \leq 1, s_{m2}, t_{n2} \in \{0, 1\} \qquad \forall (i, j) \in S_4 \tag{2k}$$

For the MILP formulation, the optimization results will decide whether it is possible to achieve a legal solution for the M2 design given the Via-1 assignment of one hit point combination. In particular, the optimal solution of the MILP formulation generates the legal line-end position of each M2 wire with the minimum amount of extensions.

Linear Programming Relaxation In the MILP formulation, integer variables are introduced to allow changing the relative order of metal line ends. This approach is guaranteed to determine the feasibility of the given Via-1 assignment with minimized total line-end extension. However, the MILP formulation may lead to long runtime for large cell designs. To explore the necessity of the MILP formulation and the possibility of speed up, we relax the MILP formulation into a Linear Programming (LP) formulation. Specifically, the relative order of line ends is pre-determined based on the initial position of metal lines, which further converts constraints ($1g$)-($1i$) to linear constraints [106]. As we can see in Section 2.6, this formulation can not optimally determine the feasibility of the given Via-1 assignment but can significantly reduce the runtime, compared with the MILP formulation.

2.5 Pin Access and Cell Layout Co-Optimization

Previously, we have shown that the LELE-aware Via-1 assignment and MILP-based optimization for M2 wires determine whether a single hit point combination is valid or not. If it is valid, we can achieve optimal LELE friendly Via-1 assignment and subsequently optimize the M2 wires for pin access and cell connections simultaneously given the Via-1 assignment. However, as shown in Fig. 3.1, multiple hit points for one I/O pin lead to numerous hit point combinations for one SC. In general, the more valid hit point combinations we have for one cell, the more flexibility we can provide to the routing stage. Thus, we extend the Pin Access Optimization (PAO) to validate all hit point combinations of an SC.

The overall algorithm for the pin access and cell layout co-optimization (PICO) is given in Algorithm 2. First, as shown in lines 1-7, the pre-processing steps determine the set of hit points for each I/O pin. Then, in line 10, the standard backtracking method is adopted to obtain a table of potential valid hit point combinations for the SC. From line 11 to line 17, we call the LELE-aware Via-1 assignment and MILP-based optimization for each entry in the table of potential hit point combinations. Assume n I/O pins for the input SC and h_i is the number of hit points for the i^{th} I/O pin, $0 \leq i \leq n-1$, the number of hit point combinations enumerated is $\Pi_{i=0}^{i=n-1} 2 * h_i$. The goal is to optimize each hit point combination, individually, hence, the time complexity will be $O((2*h)^n), h = \max_i h_i$. The parametric constant associated with the runtime complexity is given by the PAO optimization. For each cell, all valid pin access designs are stored in a table, which can be incorporated into the SC library design. Hence, we have maximized the pin access flexibility of one cell for the routing stage.

The following check heuristics prune out invalid hit point combinations.

Algorithm 2 PICO Algorithm

```
Input: Cell layout and M2 routing tracks;
 1: Define $C$ as set of M2 wires for within-cell connections;
 2: Define $IO$ as set of I/O pins;
 3: Define $S_{hp}$ as the 2-D set of hit points for each I/O pin;
 4: for each pin $p_k \in IO$ do
 5:     Get the set of hit points, $P_k$, for pin $p_k$;
 6:     Add $P_k$ to $S_{hp}$;
 7: end for
 8: Define $MTable$ as table of M2 layout design;
 9: Define $HPC$ as an empty set for hit point combination;
10: $MTable$ = Backtracking($HPC$, $S_{hp}$, $C$);
11: for each entry $H_k$ in $MTable$ do;
12:     if PAO for $H_k \cup C$ is feasible then
13:         Replace $H_k$ with the solution from PAO;
14:     else
15:         Delete $H_k$;
16:     end if
17: end for
```

1. Avoid two hit points that are on the same routing track who's end-to-end spacing is less than the pre-defined threshold, i.e., l_1 from Table 2.1.

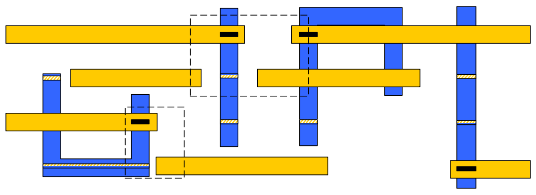
2. Existing M2 wires used for within-cell connections invalidate the hit points they cover as well as hit points that are too close in proximity.

It should be noted that we can further consider other pruning metrics during the backtracking stage. For example, cell robustness metrics, such as pin density, are closely related to the pin access design at the SC level and it could be another metric used to prune out invalid hit point combinations.

2.6 Experimental Results

We have implemented our algorithm in C++ and tested it using an industrial 14nm SC library that has been scaled and compacted to 10nm-representative dimensions. We use CLP [2] as our LP solver, CBC [1] as our MILP solver and all experiments are performed on a Linux machine with a 3.33GHz Intel(R) Xeon(R) CPU X5680. The width and space of M2 wires are assumed to be 24nm [66]. The spacer deposit width is set as 24nm. For trim mask design, the minimum resist width and space are set as 44nm and 46nm, respectively. The etch bias is set as 6nm [66]. The industrial 10nm design rules are set for the Via-1 layer and grid size g_0 for the segmentation of hit points is set as 2nm. Next, we demonstrate the strength of our optimization methodology by showing the results from pin access design for a predictive 10nm SC library consisting of around 700 cells.

Fig. 2.9 demonstrates a typical cell layout design in the library we used. The I/O pins for this cell are on the M1 layer. Due to the complexity of this cell, M2 wires are used for within-cell connections. Fig. 2.9(a) shows the M2 layout design if we simply use Via-1 locations after LELE aware Via-1 assignment to determine the line-end positions. A design rule check will reveal multiple violations in the dashed boxes. However, as illustrated in Fig. 2.9(b), the same M2 wires for pin access and within-cell connections can be co-optimized to enable SADP-friendly layout. The MILP-based optimization ensures the minimum amount of line-end extension and avoids potential engineering efforts from design rule violation fixes.


(a)

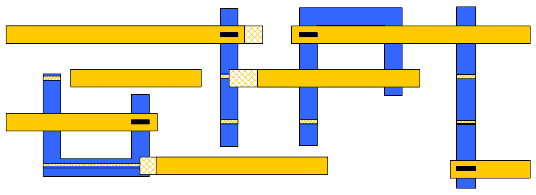

(b)

Figure 2.9: Pin access and cell connection co-optimization for one hit point combination. (a) design rule violations in layout, (b) MILP-based optimization result.

For PICO, the SC pin access flexibility is quantified using the number of valid hit point combinations (VHPCs) and valid hit points (VHPs) for each SC. For a specific SC, a larger number of VHPCs and VHPs leads to more pin access flexibility during the detailed routing stage. Our pin access optimization engine (PICO) aims at maximizing the pin accessibility at the SC level. We compare different pin access evaluation/optimization schemes, including design rule check ("DRC"), "PICO + MILP" and "PICO + LP" in Fig. 2.10. Among the cell-dependent results in Fig. 2.10(a), the "PICO + MILP" scheme consistently achieves the best performance, i.e., largest number of VHPCs. For "Cell 1", three schemes obtain a similar number of VHPCs, which means the MPL constraints introduce slight degradations on the pin accessibility. However, the "DRC" scheme achieves zero VHPC for "Cell 5"

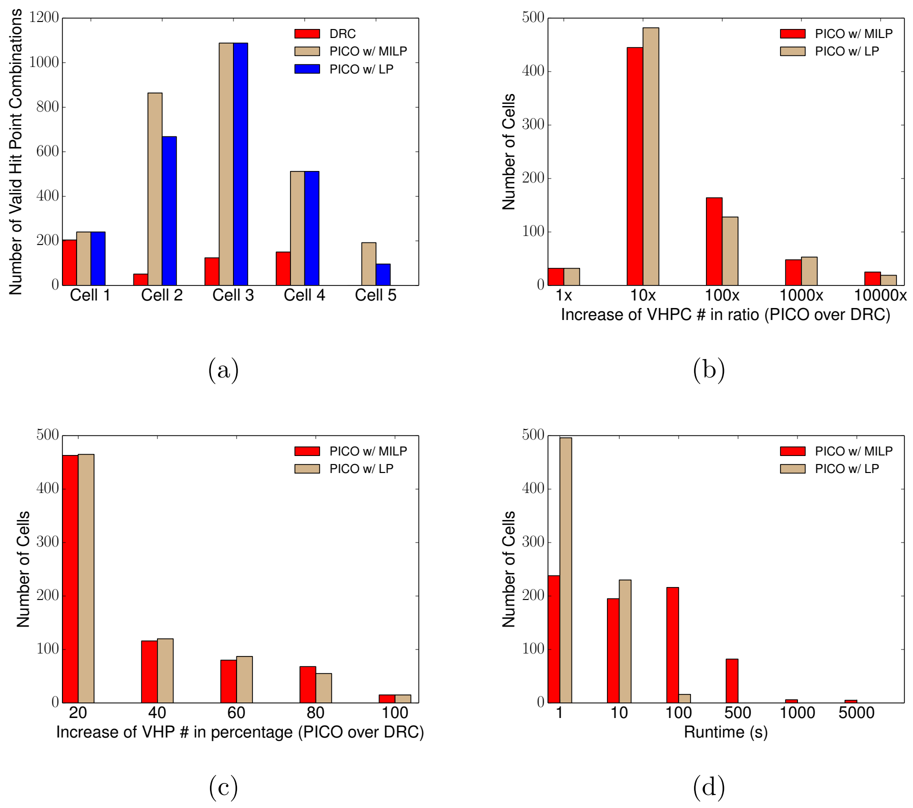


Figure 2.10: Standard cell pin accessibility improvement from PICO, (a) increase in number of VHPCs for different cells, (b) increase of VHPC # in ratio across the entire library, (c) increase of VHP # in percentage across the entire library.

while the "PICO + MILP" scheme recovers around 200 VHPCs. To evaluate the library-level benefits, we apply the PICO engine on each cell in the $10nm$ predictive library and illustrate the improvement from PICO over DRC in Fig. 2.10(b) and Fig. 2.10(c). In Fig. 2.10(b), the improvement is calculated as the "increase of VHPC # in ratio", i.e., the number of VHPCs obtained from PICO divided by that obtained from DRC. The histogram in Fig. 2.10(b) demonstrates 10X or more improvement in terms of VHPCs for most SCs. The increase in the number of VHPs is further calculated as a percentage and plotted in Fig. 2.10(c), which shows that over 25% of the cells achieve 30% or more improvement for the 10*nm* library. In Fig. 2.10(b) and Fig. 2.10(c), we also find that the "PICO + MILP" scheme is better than the "PICO + LP" scheme in terms of solution qualities since more cells tend to achieve a larger amount of increase in VHPCs and VHPs. The runtime for cells within the library is shown in Fig. 2.10(d) and the pin access optimization can be finished for most cells within 500s. Although the "PICO + MILP" scheme leads to longer runtime compared to the "PICO + LP" scheme, it is still affordable at the cell level as the PICO is a one-time computation task for a specific SC library.

2.7 Summary

This chapter proposes two techniques, PAO for a specific I/O hit point combination and PICO for an SC, which maximize the pin access flexibility for a 10*nm* SC library. To the best of our knowledge, this is the first work that addresses LELE-aware Via-1 assignment and SADP-aware I/O pin access design simultaneously. Compared to the conventional approach, we achieve significant improvement in pin accessibility of the predictive 10*nm* SC library. This also means the PICO engine can provide maximized flexibility for the routing stage.

Chapter 3

Pin Access and Routing Co-Optimization

3.1 Introduction

In sub-20nm technology nodes, pin access has become a critical challenge for detailed routing [43]. Due to the density increase or area reduction of the technology scaling, a very small number of routing tracks are available for the Standard Cell (SC) design. It makes the local SC I/O pin access challenging because the access points of each pin available for the detailed router are limited and they interfere with each other due to the extreme pitch scaling [43, 105]. Furthermore, the continued technology scaling of the lower routing layers in 14nm node and beyond depends on the complex design-for-manufacturability (DFM) strategies [83]. Unidirectional routing is strongly recommended to provide tight control of lithographic printing for lower metal layers, such as Metal-2 (M2) and Metal-3 (M3), which also complies with underlying self-aligned multiple patterning (SAMP) techniques (with cut/trim masks), e.g., SADP discussed in Chapter 2 [33, 84, 102]. Although unidirectional layout simplifies the coloring scheme during rout-

This chapter is partially based on the journal "Xiaoqing Xu, Bei Yu, Jhih-Rong Gao, Che-Lun Hsu, and David Z Pan. PARR: Pin-access planning and regular routing for self-aligned double patterning. ACM Transactions on Design Automation of Electronic Systems (TODAES), 21(3):42, 2016." I am the main contributor in charge of problem formulation, algorithm development and experimental validations.

ing [33, 102, 104, 110, 119], it makes local SC pin access more challenging due to the loss of I/O access points and more complicated neighborhood interactions. To mitigate the SC pin access issue, designers can perform extensive layout optimization with the assistance from local-routing enumeration and optimization, i.e., SC pin access optimization and evaluation as discussed in Chapter 2. However, this is constrained by modern SC architectures, e.g., 7.5-track or 9-track cell architecture [61], and a detailed router now has to manage increasingly complicated pin access interference and finish all net connections while accommodating complex DFM constraints.

Detailed routing has to account for pin accessibility while searching for an exact route of each net. A typical detailed routing strategy performs pathfinding of the nets sequentially. For SC pin access, the access point selection of the local I/O pins of the net being routed could impact the routability of the remaining nets. A typical example is illustrated in Fig. 3.1. The pre-routed M2 wires in Fig. 3.1(a) blocks the I/O pin on the right side of *Cell* 1, which makes the remaining net unroutable. A different access point selection scheme is shown in Fig. 3.1(b), where the accessing points of net *A* are changed and all nets are successfully routed. In addition, the routing order of nets is also critical for the routability when each I/O pin has a limited number of access points interfere with each other [81]. Thus, the pin access planning, including access point selection at the SC level and net order prediction, is very important for the detailed router to achieve better pin accessibility. Meanwhile, the rip-up and reroute scheme is a classical method to improve the quality of detailed routing solutions [20, 21]. To fix the pin access failure in Fig. 3.1(a), two different rip-up and reroute procedures are shown in Fig. 3.1(c) and (d). The routing wires involved in the rip-up and reroute procedures are shown in dashed lines. In Fig. 3.1(c), net B is ripped up and the rerouting results block the I/O pin in the middle of *Cell* 1 due to the M2 spacing rules. Successful rerouting can be achieved if net A is ripped up as shown in Fig. 3.1(d). Therefore, pin access driven rip-up and reroute schemes are strongly needed to route today's complex designs.

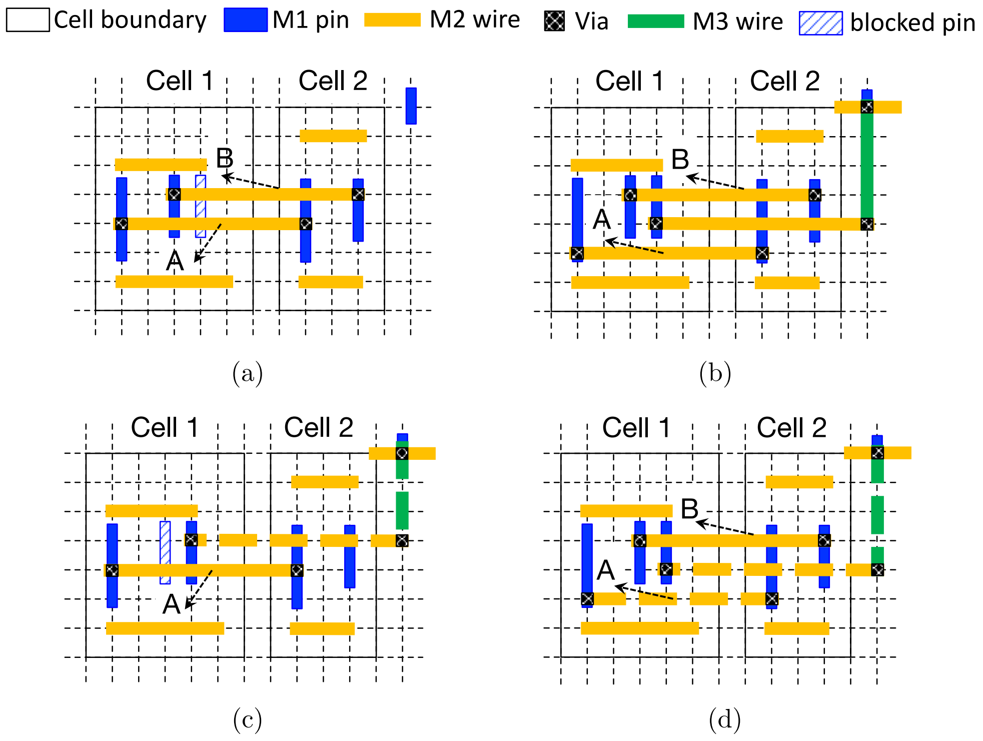


Figure 3.1: Pin access for detailed routing, (a) pin access failure, (b) pin access success with smart access point selection, (c) rip-up and reroute failure, (d) rip-up and reroute success.

A wide range of academic research across various design stages have been dedicated to the pin access issue in advanced technology nodes, including SC design [43, 105, 116], placement mitigation [97], global routing [7, 87] and detailed routing [80, 81]. Among them, SC I/O pin access and detailed routing play an important role due to their direct impact on the detailed routability. [105] addresses the I/O pin access issue for each cell in isolation under SADP-specific constraints but the related detailed routing scheme is not explicitly presented. [81] proposes an escape routing strategy to improve the detailed routability for dense pin clusters instead of each SC within the design. [80] focuses on the gridless pin access in the detailed routing stage, which cannot be directly applied in grid-based designs in advanced technology nodes [93].

To deal with complex design-for-manufacturing constraints, existing SAMP-aware routing studies mainly focus on the coloring scheme of routing patterns while accommodating specific design rules for cut/trim masks with novel routing grid models [23, 26, 27, 29, 52, 64, 76, 77, 94, 111]. Furthermore, [27] and [94] propose to address the cut mask complexities during the track assignment and global routing stages, respectively. The detailed routing procedures of existing studies follow the paradigms of sequential routing [29, 52, 64, 76, 77, 94, 111] or negotiation-congestion-based routing schemes [23, 26]. In general, negotiation-congestion-based routing technique can resolve routing congestions more efficiently than a sequential scheme, because a detailed router avoids following a specific net ordering with a history-based heuristic [75].

In advanced technology nodes, detailed routing on lower metal layers is becoming more complicated than simply dealing with patterning constraints. The reasons are two-fold. First, the ever-increasing SC pin access interference needs to be effectively addressed so that all nets can be routed simultaneously, as shown in Figure 3.1. Conventional rip-up and reroute schemes generally lead to huge runtime overheads due to severe routing resource competition, which makes concurrent optimization techniques particularly important for fast routing closure. Second, routing resources on lower metal layers are primarily reserved for short or local nets. For short-net routing, via minimization is particularly important in advanced technology nodes [24].

This chapter first proposes PARR, a sequential routing framework that explicitly addresses SC I/O pin access and regular routing under SADP-specific constraints. Our framework starts with the pin accessibility study of a given SC library. Then, based on the placement-level information, the pin access planning strategies are proposed to guide the detailed router for much better pin accessibility. Our main contributions are summarized as follows:

- The local pin access planning scheme is proposed to enable smart access point selection for the single-net routing.
- We propose the global pin access planning strategy based on the concept of a pin access graph to guide the regular routing for the ultimate routability.

- A pin access driven rip-up and reroute scheme is proposed to further improve the detailed routability.

However, a sequential routing procedure cannot optimally resolve pin access interference on one metal layer, such as M2, which generates significant overhead on the routing solution qualities. To further address the routing challenges aforementioned, we propose the concurrent pin access optimization study for unidirectional routing. This study is unique to previous detailed routing studies [6, 81, 126] due to following reasons. This study focuses on interval-based search instead of grid-based search. We demonstrate that concurrent pin access optimization is critical to resolving routing resource competition, including pin access interference. We perform unidirectional routing while explicitly addressing the complex manufacturing constraints from self-aligned double patterning (SADP). Our contributions are listed as follows.

- We take advantage of the unidirectional routing style to propose track-based pin access interval generation and linear conflict set detection.
- The concurrent pin access optimization is modeled as a weighted interval assignment problem, which is further formulated as a binary integer linear programming problem.
- We propose an iterative Lagrangian relaxation algorithm to obtain scalable solutions for the concurrent pin access optimization problem.

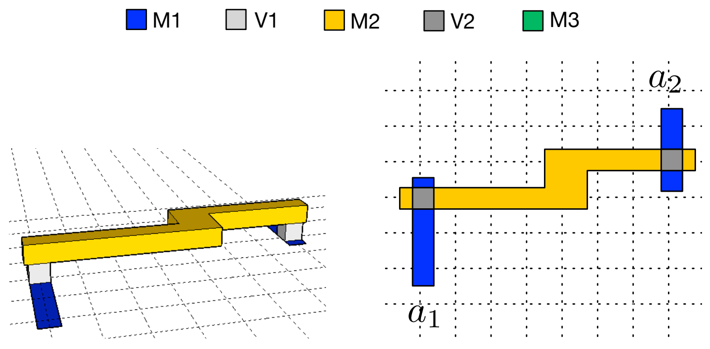
- We implement a concurrent pin access router (CPR) to obtain much better routing results than the state-of-the-art SADP-friendly routers.

3.2 Sequential Pin Access Planning

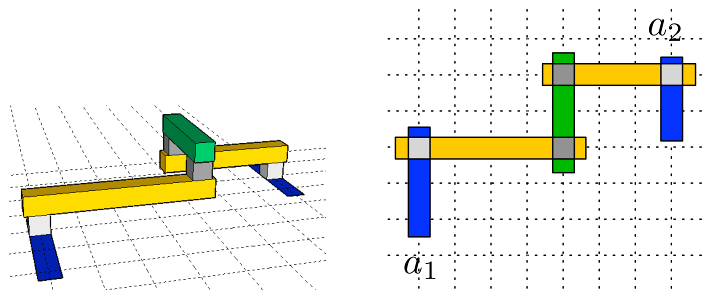
3.2.1 Preliminaries and Problem Formulation

Unidirectional Routing Patterns In advanced technology nodes, regular routing wires with tight pitches are preferred to be unidirectional [93]. Two-dimensional (2-D) and unidirectional routing patterns for a simple net is shown in Fig. 3.2. Compared with the 2-D routing patterns with jogs in Fig. 3.2(a), unidirectional routing patterns in Fig. 3.2(b) have better printability but extra vias will be introduced when the detailed router changes the routing directions. In addition, different access points will be selected for the unidirectional routing patterns. Hence, smart pin access planning strategies are crucial for the quality of the routing solutions with unidirectional patterns.

Related Design Rules In the detailed routing stage, a set of design rules need to be followed by the detailed router so as to achieve legal regular layout patterns. Practically, these design rules lead to additional difficulties to access the local I/O pins of the SCs for the detailed router. Related design rules on metal patterns are defined in Table 2.1. With unidirectional layout patterns on the single routing layer, only one direction, horizontal or vertical, is allowed for routing paths on the same layer. To change the direction of a routing path, a detailed router will have to insert vias and switch interconnect layers



(a)



(b)

Figure 3.2: Two dimensional vs unidirectional routing patterns, 3D view (left) and top view (right), (a) two dimensional routing patterns with a jog, (b) unidirectional routing patterns with extra vias.

among the multi-layer grids for the detailed router as shown in Fig. 3.2(b). Thus, the via rule is another important factor for the quality of the routing solution. The minimum center-to-center spacing rule for vias is considered for the completeness of the pin access and routing co-optimization framework, which can be extended to other complex via rules such as multiple patterning related constraints. In Chapter 2, pin access is pre-computed on the M2 layer, M1 I/O pins are promoted to the M2 layer and the via rule is only imposed on the M2 and layers above.

Problem definition For an SC library, the set of feasible intra-cell I/O pin access point combinations for each cell in isolation can be pre-computed under a set of design rules on the M2 layer as discussed in Section 2.2.3. However, in practical designs, SCs are placed next to each other instead of being isolated, which means the inter-cell pin access needs to be further addressed. Since the number of cells within a library is finite, the intra-cell and inter-cell pin access can be pre-computed and stored in a look-up table (LUT). With the LUT for the pin accessibility, we propose efficient pin access planning strategies to guide the regular routing and enable handshaking between the SC pin access and detailed routing. Meanwhile, design rule legalizations are performed to guarantee legal routing results. Hence, we define the pin access guided regular routing as follows.

Problem 3 (Pin Access Guided Regular Routing)**.** *Given a netlist, a grid routing plane, cell placement, pin access LUT of the library and a set of design rules, the pin access planning guided regular routing performs the regular routing and design rule legalization simultaneously to achieve SADP-friendly routing results.*

To evaluate the effectiveness of the proposed techniques for improving the quality of routing solutions, we have the following definition.

Definition 5 (Routability)**.** *The routability is defined as the percentage of nets routed over the total number of nets in a design.*

3.2.2 Pin Accessibility Prediction

This section aims at the pre-computation of the pin accessibility under a set of design rules, which yields an LUT of the inter-cell and intra-cell pin accessibility for the given SC library.

Intra-Cell Pin Access The SC I/O pin access problem has been explicitly addressed in Chapter 2, where the pin accessibility is determined while minimizing the total amount of line-end extensions. However, line-end extensions can be automatically performed by the detailed router to fix related rule violations [26,64]. In addition, a router is responsible for dynamically choosing the access direction of a specific accessing point of the SC I/O pin. This means a feasibility study at the SC level is enough to guide the detailed routing. Thus, we adapt the pin access and SC layout co-optimization (PICO) method in Section 2.5 to determine the intra-cell pin accessibility. For convenience, we adopt the definitions of *Hit Point (HP)*, *Valid Hit Point (VHP)*, *Hit Point Combination (HPC)* and *Valid Hit Point Combination* from Section 2.3.

Inter-Cell Pin Access We have discussed intra-cell pin access feasibility for each isolated cell. However, the pin accessibility may interfere and degrade when two cells are placed next to each other. We quantify the inter-cell pin accessibility for a given SC pair with each cell assigned a specific VHPC. A typical example of a cell pair, denoted as $pair_{ij}$, is illustrated in Fig. 3.3, where c_i is placed to the left of c_j with the gap distance as g. For pragmatic placement, g is an integer multiple of the placement pitch. The pin access interference is expected in Fig. 3.3(a) because the pin access boundaries of the two cells are next to each other. To demonstrate the pin accessibility interference, we choose the m^{th} VHPC for c_i and the n^{th} VHPC for c_j in Fig. 3.3(b). The M2 wires for intra-cell pin access associated with the selected VHPC's are also shown in Fig. 3.3(b), where pin access M2 wires introduce extra rule violations even with VHPC's for c_i and c_j in isolation. To further explore inter-cell pin accessibility, additional line-end extensions are required to fix the violations and make $pair_{ij}$ accessible in Fig. 3.3(b).

The additional line-end extensions based on the selected VHPC's for $pair_{ij}$ has the exact same formulation as the PAO in Section 2.4. If two cells c_i and c_j are assigned the m^{th} and n^{th} VHPC and the gap distance is set to g in Fig. 3.3(b), the feasibility of fixing extra violations can be evaluated with PAO on the set of M2 wires, i.e., $pa_i^m \cup pa_j^n$. In particular, from Fig. 3.3(a), the pin access boundary is extended beyond the cell boundary. For practical implementation, the left and right boundary of the cell could be extended by minimum M2 length, which preserves the validness of the HPs close to the cell boundary while satisfying the minimum length rule for M2 wires. This pin access boundary only applies to the pin accessibility prediction phase, which does not introduce additional placement or routing constraints. Since the detailed router only cares about the pin accessibility at the SC level, we only report VHP's and VHPC's to the detailed router and each VHP is accessible from the M2 layer.

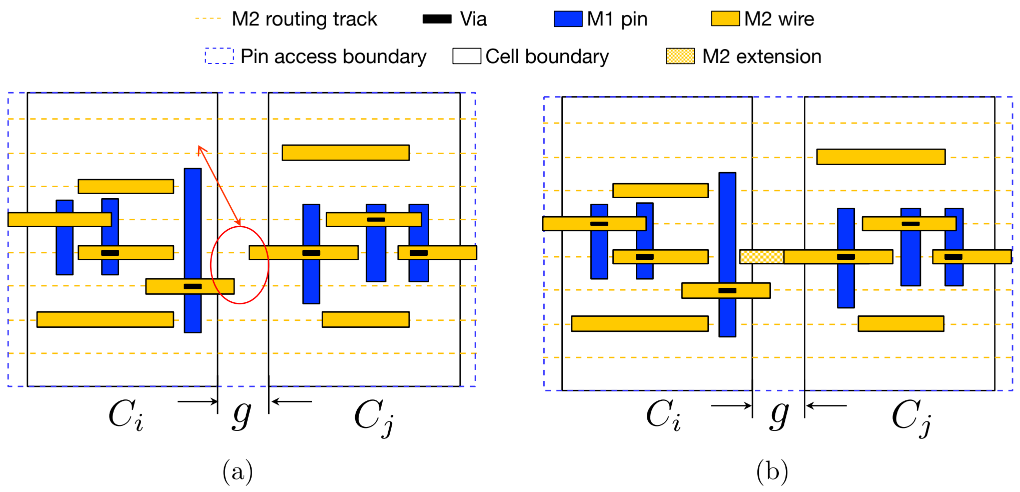


Figure 3.3: The inter-cell pin access, (a) two cells placed next to each other and intra-cell pin access introduces a rule violation, (b) fixing the rule violation by line-end extension.

Look-Up Table Construction Here, the LUT construction is presented for the library-wide inter-cell pin accessibility prediction. We assume that M2 routing layer is horizontal and pin accessibility is pre-computed on the M2 layer. Modern SC cell architecture creates horizontal power/ground rails on the M1 layer, which further splits the design into horizontal rows on the M2 layer. Thus, pin access interactions between adjacent rows are non-critical and the pin access LUT only needs to be calculated for cells abutted horizontally. Connecting cells in adjacent rows involves M2 (horizontal) and M3 (vertical) routing layers simultaneously, which is performed by the detailed router. For each cell c_i in the library, we have computed a set of VHPC's from the intra-cell pin access feasibility study. We further evaluate the inter-cell pin access feasibility of each cell pair ($pair_{ij}$ with gap distance g) using the PAO technique, which will be stored in an LUT. Specifically, if two cells c_i and c_j are assigned the m^{th} and n^{th} VHPC and the gap distance is set to g, the inter-cell pin access feasibility can be evaluated on the set of M2 wires, i.e., $pa_i^m \cup pa_j^n$. Then $LUT(i, m, j, n, g)$ stores a Boolean value denoting whether inter-cell pin access is feasible or not when c_i is to the left of c_j. Here, the gap distance g is also an index of the LUT because changing the gap distance between cells has potential impacts on the inter-cell pin accessibility. For example, in Fig. 3.3(b), in spite of extra rule violations when c_i is to the left of c_j with gap distance as g, the violation can be fixed by additional line-end extensions. Thus, the element $LUT(i, m, j, n, g)$ will store a "true" value. Cell flipping across the horizontal axis is also considered and related values are stored during LUT construction.

Depending on the SC library design, the LUT size could be large or even unaffordable due to large number VHPC's for each SC. For the library we used, the maximum number of VHPC's for one cell is 7745 and the average number of VHPC's for each cell is 589.1. To control the number of entries within the LUT, our LUT is constructed only on critical pin-access cells. Specifically, critical pin-access cells are those cells with a small number of VHPC's (e.g., < 500) or some I/O pins of the cells have limited number of VHP's (e.g., < 5). For each cell pair in the LUT, we require both cells to be critical. We assume that if two cells are placed next to each other and only one of them is critical, the other cell has enough pin access flexibility to compensate for the pin accessibility of its neighbor. Suppose the number of pin-access critical cells is N, the maximum HPC per cell is M, and the maximum gap is g, then the LUT size is at most $4 * N^2 * M^2 * g$. Since each LUT item is just true or false for the inter-cell pin accessibility, we only store the false entries for our practical implementation, which further reduces the size of the entire LUT.

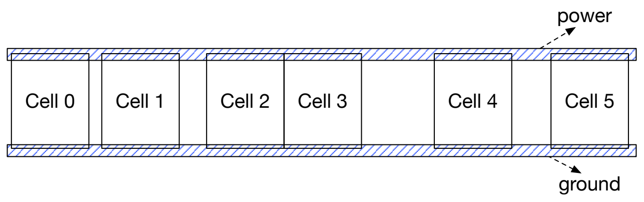
3.2.3 Pin Access Planning Guided Regular Routing

The pin access graph (PAG) is proposed to dynamically determine the pin accessibility of a single SC or a row of SCs. Then, local and global pin access planning strategies are presented, which further guide the regular routing to improve pin accessibility. In addition, a pin access driven rip-up and reroute scheme is proposed to increase the ultimate routability.

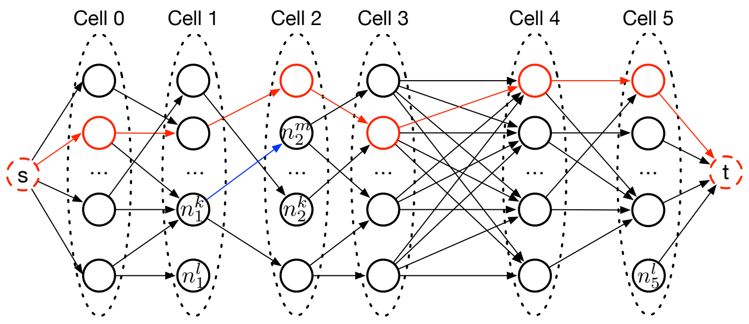
3.2.3.1 Single Row Pin Access Graph

In the row-structure for placement in Fig. 3.4(a), SCs are aligned horizontally and share the same height. The power and ground rails go from the very left to the very right of the die area. Given the position for each SC and a placement row, we build the single row pin access graph. As illustrated in Fig. 3.4(b), the single row PAG is a directed graph starting from the virtual source node (s) on the left to the virtual target node (t) on the right. For each SC placed within the row, we introduce a set of nodes into the PAG and each node corresponds to one of the VHPC's for that particular SC, where n_i^m denotes the node for the m^{th} VHPC for *Cell i*. We add edges between s and n_0^m, for each m. Similarly, edges will be introduced between n_5^m and t, for each

m. No edges will be added between n_i^m and n_i^k, namely, nodes for the same SC. For neighboring SCs, such as $Cell$ 1 and $Cell$ 2, an edge, denoted as a blue arrow, will be added between n_1^k and n_2^m since the item $LUT(1, k, 2, m, g)$ is true, where g is the gap distance for $pair_{ij}$. In contrast, no edge is introduced between n_1^l and n_2^k since $LUT(1, l, 2, k, g)$ is false.



(a)

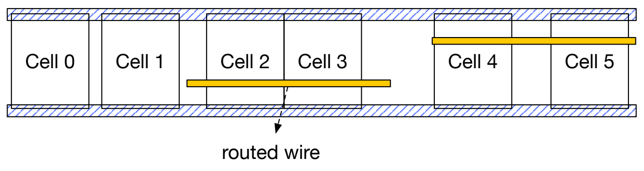


(b)

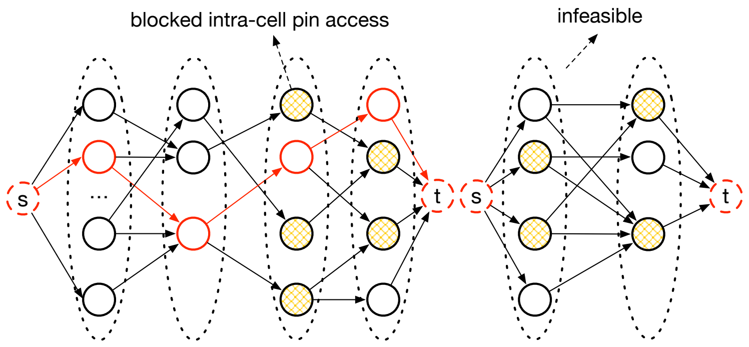
Figure 3.4: Single row pin access graph, (a) cell placement, (b) initial pin access graph.

For the PAG, we have the following observation.

Observation 3. *The pin accessibility of the SCs on the M2 layer within the single row is equivalent to the existence of a path from s to t of the PAG associated with that particular row. We define a PAG component to be infeasible for pin access when no feasible path exists from s to t of that PAG component.*



(a)



(b)

Figure 3.5: (a) cell placement with pre-routed wires, (b) simplified pin access graphs with blocked nodes.

If there exists a path from s to t, the SCs within the row are accessible using the set of VHPC's and corresponding pin access M2 wires on the path. An example is shown in Fig. 3.4(b). Moreover, routing wires on the M2 layer will be created on top of the SCs during the routing stage, as shown in Fig. 3.5(a). The routing wires over the SC create routing blockages, which block some specific HPs of the SCs. This means the associated VHPC's for the SC will be unavailable for pin access. As demonstrated in Fig. 3.5(b), some nodes will become invalid, indicated by the dashed orange nodes due to the routing wires on top of the $Cell\ 2-5$.

Graph simplification For the single row PAGs constructed, some graph nodes have no children or parents and can be safely removed without impacting the feasible paths from s to t within each PAG. As shown in Fig. 3.4(b), the node n_1^l has no children and the node n_5^l has no parents, which are removed in Fig. 3.5(b). In addition, we can perform the graph partitioning from Fig. 3.4(b) to Fig. 3.5(b) since an edge exists between n_3^m and n_4^l, for any (m, l) pair. The graph partitioning creates several components for each single row PAG. Then, pin accessibility of the SCs with the row is equivalent to the existence of paths from s to t on two independent components in Fig. 3.5(b). However, no feasible path exists on the right component of the PAG after the creation of pre-routed wires in Fig. 3.5(b), which means pre-routed wires need to be ripped up to preserve the routability of remaining nets. To achieve quick update on the PAGs, graph partitioning is applied to all the PAGs associated with the placement rows of the design. As discussed in Section 3.2.4, the graph partitioning technique controls the size of the single component of the PAGs, which leads to feasible pin access planning guided detailed routing within affordable runtime cost.

Furthermore, each component of the PAG is related to a set of SCs in proximity, which is bounded by a determined bounding box. Since the search for impacted components of the PAG needs to be done whenever a net is routed, we adopt R-tree [38] to enable the fast indexing bounding box of each component of the PAG.

3.2.3.2 Local Pin Access Planning

An intra-cell pin accessibility study yields a set of VHPC's, denoted as $VHPC_i$ for each cell c_i within the library. An HP is invalid if there are no VHPC's associated with that HP. Therefore, invalid HPs are inaccessible and should be removed before the detailed routing stage, which avoids unnecessary routing efforts. Next, we discuss how to differentiate among various VHPs for a particular I/O pin.

The pre-computation of the intra-cell pin access achieves a set $VHPC_i$ for cell c_i. For each I/O pin for c_i, we propose a dynamic hit point scoring strategy to differentiate among various VHPs for that particular I/O pin. In general, a higher score is assigned to an HP if that particular HP has a larger number of VHPC's associated with it than other HPs of the same I/O pin. Hence, we calculate the score for the h^{th} HP of the k^{th} I/O pin for c_i, namely hp_i^{kh}, as follows.

$$score(hp_i^{kh}) = \frac{\text{number of VHPC's associated with } hp_i^{kh}}{\text{total number of VHPC's for } c_i} \tag{2}$$

In the sequential routing scheme, the M2 wires created for routed nets become blockages, an example of which is shown in Fig. 3.5(a). As discussed above, some VHPC's for blocked cells become invalid as illustrated in Fig. 3.5(b). Therefore, the score for each HP should be updated dynamically during the detailed routing stage. We only update the scores of the HPs of the cells impacted by the routed nets. For practical implementation, we adopt the "unordered map" data structure to store the VHPC's for each cell for fast updating. For the single-net routing, the router prefers selecting the HPs with higher scores for source and target pins of the net being routed.

3.2.3.3 Global Pin Access Planning

The local pin access planning enables smart HP selection for single net routing. During sequential detailed routing, the routed wires block some VHPs of the I/O pins not yet routed, which degrades the routability of the remaining nets. Thus, this subsection addresses the global, rather than the local prediction of the pin accessibility.

For sequential detailed routing, the relative order of routing nets has a potential impacts on the routability, as discussed in Section 3.1. Here, we introduce two techniques to enable net deferring and improve pin accessibility. First, the routability of a net relates to the accessibility of each pin within this net. This motivates us to defer the routing of nets with robust source and target pins, which both have many VHP's available. Second, the PAG for each placement row determines the accessibility of the cells within that row. To preserve the global pin accessibility of the remaining nets, we dynamically maintain the source-to-target path existence of each component of the PAGs. Specifically, we update the pin access graph based on the routed wires from the trial routes of each net. If no feasible path exists from source to sink in some pin access graph, we will rip-up and increase the deferring cost of that net and reroute that net later. Otherwise, we keep the routing results of the net. The weight for the order of the net n_k is calculated as follows.

$$order(n_k) = HPWL(n_k) \cdot (1 + \alpha \cdot \min\{hp_s, hp_t\}) + DCost(n_k) \qquad (3)$$

In Eq. (3), $HPWL(n_k)$ denotes the half-perimeter wirelength of net n_k, α is a user-defined parameter, hp_s and hp_t denote the number of VHPs available for source and target pins, respectively. $DCost(n_k)$ is the deferring cost of n_k. With the net deferring scheme, one net may be deferred for several times due to its impact on the routability of the remaining nets. To quantify that, we have the following definition.

Definition 6 (Deferring Cycle)**.** *Deferring cycle is the maximum number of times that a net is deferred before reaching the cost upper bound.*

In one deferring cycle, if routed M2 wires of some net make some PAG component infeasible for pin access, the routing of that net will be deferred and current routing patterns will be cleared to preserve the routing of remaining nets. An example has been illustrated in Fig. 3.6. Only 4 nets, including $\{a_1,a_2\}$, $\{b_1,b_2\}$, $\{c_1,c_2\}$, and $\{d_1,d_2\}$ are explicitly drawn with routing patterns and other nets, including net_set1 and net_set2, in the design are shown here for clear demonstration. The original order of routing is $\{a_1,a_2\}$, $\{b_1,b_2\}$, $\{c_1,c_2\}$, net_set1, $\{d_1,d_2\}$, net_set2 based on Eq. (3). As shown in Fig. 3.6(a), the net deferring strategy detects the pin d_1 is blocked after the routing of $\{c_1,c_2\}$ is completed. With the PAGs constructed, blocking pin d_1 makes the PAG component associated with the pin to be infeasible for pin access as discussed in Fig. 3.5(b). Instead of waiting until the nets in net_set1 are routed to find out that $\{d_1,d_2\}$ is unroutable, the net deferring strategy will increase the $DCost$ of $\{c_1,c_2\}$ and net $\{d_1,d_2\}$ will be routed before $\{c_1,c_2\}$. As shown in Fig. 3.6(b), the routing has been successfully achieved with the net deferring scheme. In particular, the net deferring scheme is dynamic, which means the net will be deferred whenever necessary.

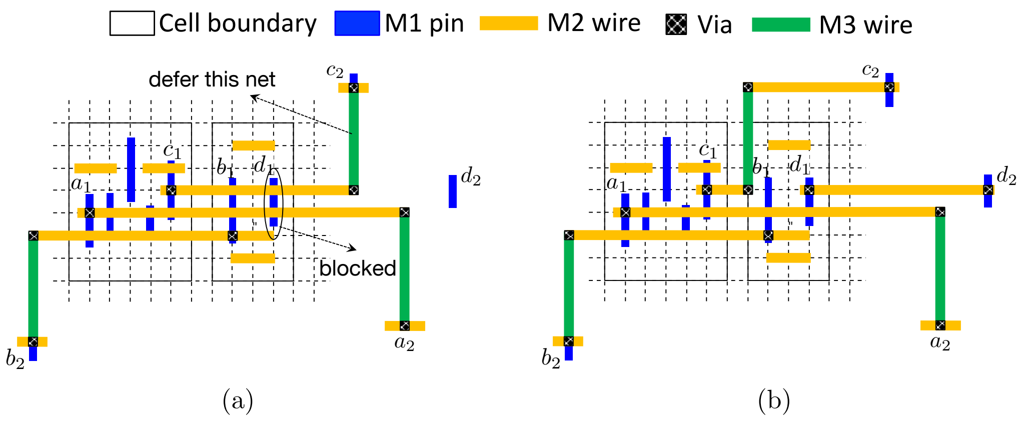


Figure 3.6: A net deferring example, (a) defer the routing of net $\{c_1,c_2\}$, (b) successful routing.

The detailed net deferring scheme is shown in Algorithm 3. From line 1 to line 6, the minimum heap for routing nets is built based on the order of each net computed using Eq. 3. In each loop when the *net_heap* is not empty, the *net* with the minimum order weight is extracted from the *net_heap* and the single-net routing is performed in line 8 and line 9. From the routing wires of the *net*, the impacted components of PAGs are updated on line 10. Impacted components are those components containing nodes blocked by the newly created M2 wires as shown in Fig. 3.5(b). As discussed in Section 3.2.3.1, R-tree enables quick search of the impacted components when new routing wires are created. If the routing results of the *net* break the pin accessibility of the PAG from line 11 to line 15, the deferring cost is increased and the *net* is deferred and pushed back to the *net_heap* when the accumulated deferring cost is within the maximum bound. Otherwise, the *net* is added to the *nets_deferred* in line 17. In line 21, the *nets_deferred* will be returned as the input of the pin access driven rip-up and reroute.

Algorithm 3 Net Deferring Algorithm

```
Input: a set of nets ($Nets$), maximum deferring cost ($maxCost$), increasing unit for deferring cost ($unit$) and pin access graphs for placement rows ($PAGs$);
Output: a set of nets ($nets\_deferred$)
 1: Define $net\_heap$ as the minimum heap for $Nets$;
 2: for each net $n_k$ in $Nets$ do;
 3:     Set $DCost(n_k) = 0$;
 4:     Compute $order(n_k)$ based on equation (3);
 5:     $insert\_heap(n_k, net\_heap)$;
 6: end for
 7: while $net\_heap$ is not empty do;
 8:     Define $net = extract\_min(net\_heap)$;
 9:     Perform A* search for $net$ and achieve trial routes;
10:     Update impacted components of $PAGs$;
11:     if $PAGs$ are infeasible for pin access then;
12:         $DCost(net) = DCost(net) + unit$;
13:         if $DCost(net) < maxCost$ then;
14:             Defer $net$ and update $PAGs$;
15:             $insert\_heap(net, net\_heap)$;
16:         else
17:             Add $net$ to $nets\_deferred$;
18:         end if
19:     end if
20: end while
21: return $nets\_deferred$;
```

Pin Access Driven Rip-up and Reroute The local and global pin access planning schemes guide the detailed router for better pin accessibility. There might be some remaining nets, i.e., $nets\_deferred$ from Algorithm 3, that are not yet routed. Thus, an additional pin access driven rip-up and reroute scheme is strongly needed to improve the ultimate routability.

As discussed in Fig. 3.1(c) and (d), different rip-up and reroute procedures may lead to pin access success or failure, depending on the nets to be ripped up and the outcome of the A* searches. To improve the solution quality, a typical rip-up and reroute procedure involves various trials or several iterations [20, 21]. Our pin access driven rip-up and reroute scheme follows this direction with local pin access information. Fig. 3.7 gives two rip-up and reroute examples, where only 6 nets, i.e., $\{a_1,a_2\}$, $\{b_1,b_2\}$, $\{c_1,c_2\}$, $\{d_1,d_2\}$, $\{e_1,e_2\}$ and $\{f_1,f_2\}$ are drawn for clear demonstrations. As shown in Fig. 3.7(a), the U-shape pin f_1 is blocked by the routing patterns from two nets, i.e., $\{a_1,a_2\}$ and $\{b_1,b_2\}$. If the net $\{a_1,a_2\}$ is ripped up in Fig. 3.7(a), the rerouting result in Fig. 3.7(b) shows that pin a_1 is blocked, which leads to a rerouting failure. In Fig. 3.7(c), if we rip-up the net $\{b_1,b_2\}$ blocking the hit point with the highest score from Eq. (2), successful rip-up and reroute can be achieved as shown in Fig. 3.7(d). Thus, the pin access driven rip-up and reroute scheme makes use of the dynamic hit point scoring in Eq. (2) to determine the rip-up and reroute procedures. It shall be noted that the pin access driven rip-up and reroute is different from the net deferring scheme because only local pin access information is used to guide the rip-up and reroute iterations. Since the global pin access planning is not an optimal strategy, the pin access driven rip-up and reroute aims at improving the routability with an extra iterative step. The details of our rip-up and reroute algorithm are skipped here because it can be achieved by incorporating the greedy metric from Eq. (2) into the traditional rip-up and reroute procedures [20, 21].

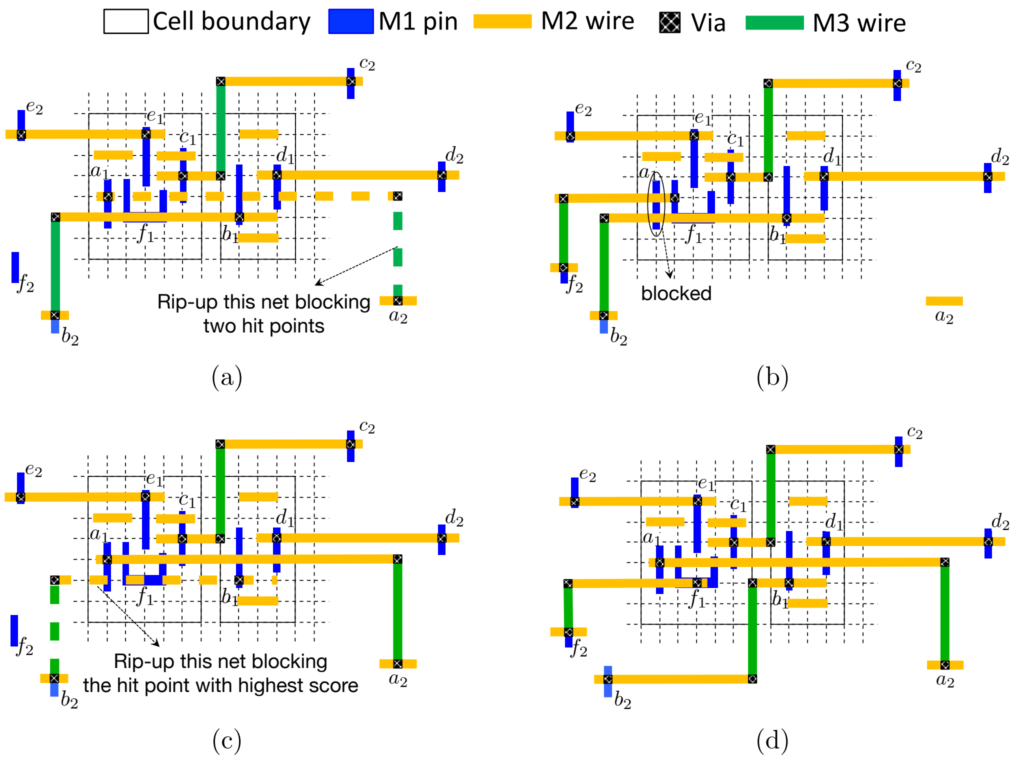

Cell boundary
M1 pin
M2 wire
Via
M3 wire
Rip-up this net blocking two hit points
blocked
Rip-up this net blocking the hit point with highest score
(a)
(b)
(c)
(d)

Figure 3.7: A rip-up and reroute example, (a) rip up net $\{a_1,a_2\}$, (b) reroute net $\{f_1,f_2\}$ and pin a_1 is blocked, (c) rip-up net $\{b_1,b_2\}$, (d) successful rip-up and reroute.

Sequential Routing Flow Our routing strategy adopts the grid-based routing model and targets the unidirectional layout patterns friendly to the design rules introduced in Section 3.2.1. The line-end extensions are performed to fix the violations relevant to the minimum length rule and metal line-end rules for SADP in Table 2.1. Since both intra-cell and inter-cell pin accessibility are pre-computed on the M2 layer, we only report those M2 HP's of I/O pins accessible from the M2 layer to the router in the pre-processing phase. Thus, for the routing of each net, all source/target M1 pins can be accessed through HP's on the M2 layer. Considering the spacing rule for via layer, we impose the forbidden grids once a legal via is inserted for the routed net. Specifically, neighboring grids surrounding the via position are forbidden to be used for the remaining nets [111]. The via rule is imposed on the M2 and upper layers.

We focus on a routing grid model, which can easily be extended to the multi-layer routing framework. Then, our detailed router follows the paradigm of an A* search, which is guided by the local and global pin access planning strategies. The cost of the routing grid is calculated while performing the A* search. If we consider a routing path from grid g_i to grid g_j, the cost of the grid g_j, denoted as $C(g_j)$, can be computed as follows:

$$\begin{aligned} C(g_j) = C(g_i) + \theta \cdot (1 - score(hp_j)) + \eta \cdot C(forbid(j)) \\ + \beta \cdot C(WL_{ij}) + \gamma \cdot C(Via_{ij}) \end{aligned} \tag{4}$$

In Eq. 4, $score(hp_j)$ is the dynamic hit point score for g_j if g_j is a source or target grid. Otherwise, $score(hp_j)$ is set to 1. In general, the A* search prefers routing grids with lower cost. Thus, the term $score(hp_j)$ enables the local pin access planning, which prefers selecting the HPs with higher scores for the source or target pins of the net being routed. $C(forbid(j))$ is the forbidden cost for the grids if the grid g_j is within the prohibited region of some pre-routed wires [26, 67]. This cost is set to help the design rule legalization for the trim mask rules. $C(WL_{ij})$ and $C(Via_{ij})$ are the amount of wirelength and vias for the routing path from g_i to g_j. $\beta, \gamma, \theta, \eta$ are user-defined parameters to adjust weights for of cost.

The overall routing flow is demonstrated in Fig. 3.8. The intra-cell and inter-cell pin accessibility for an SC library is pre-computed and stored in a LUT. Then, based on the placement level information and the LUT, the PAGs are constructed for the design being routed. With the PAGs, the net deferring algorithm makes use of the local and global pin access planning strategies to achieve better pin accessibility. For those unrouted nets, the pin access driven rip-up and reroute scheme improves the ultimate routability with several rerouting iterations.

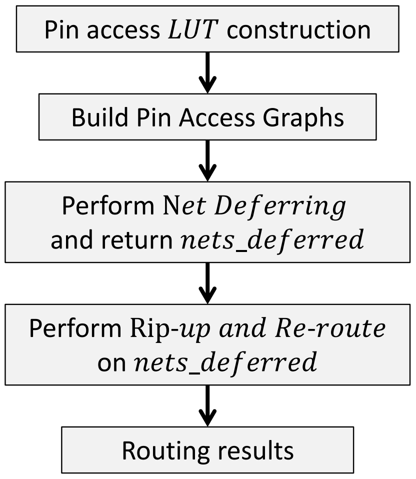

Pin access LUT construction
Build Pin Access Graphs
Perform Net Deferring and return nets_deferred
Perform Rip-up and Re-route on nets_deferred
Routing results

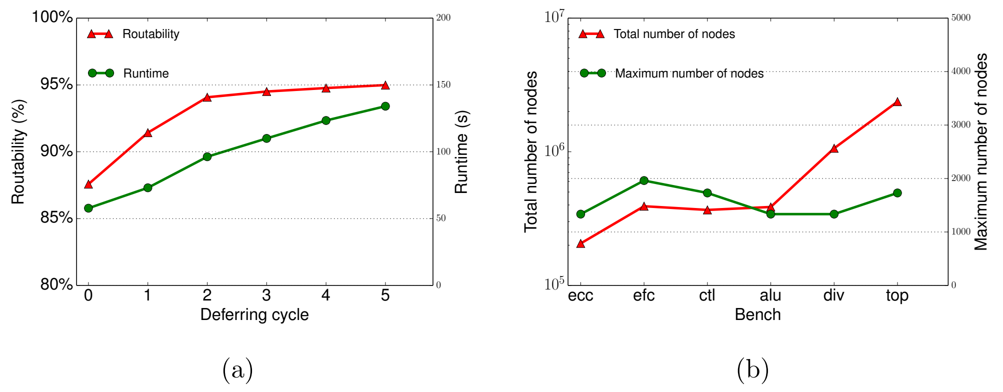
Figure 3.8: Sequential routing flow.

3.2.4 Experimental Results

We have implemented PARR in C++ and all experiments are performed on a Linux machine with 3.4GHz Intel(R) Core and 32GB memory. With the help from the authors of [64], the 2-D SADP-aware routing results are generated on a Linux machine with 2.0GHz CPU and 72GB memory. We modify and scale the NanGate 45nm open cell library [78] to represent the pin access scenario in advanced technology nodes, where M2 wires may be introduced in the SC layout design as discussed in Chapter 2. For the LUT construction, we only store the false entries in our implementation since each entry is just true or false for the inter-cell pin accessibility checking. The number of entries in the LUT constructed in our implementation is around $1.8 * 10^6$. As illustrated in Table 3.1, modules from the OpenSparc T1 are synthesized with Design Compiler [96]. The placement results are generated using Cadence SOC encounter [16] with the utilization rate set to 70%. All benchmarks are scaled and compacted to 10nm-representative dimensions. Since our study targets improving the pin accessibility in the detailed routing stage, global nets are treated as pre-routed nets and beyond the scope of our detailed routing framework. A net is treated as a global net if its bounding box crosses more than M horizontal or vertical routing tracks. M is set as 40 in our implementation. We focus on the two-layer (M2 and M3) regular routing, where the routing directions of M2 and M3 are horizontal and vertical, respectively. We adopt Gurobi [37] as our MILP solver for pin accessibility prediction. The upper bound on the gap distance of a cell pair is set to $g = 2$

and the unit is min-sized placement pitch. The width and space of M2 and M3 wires are assumed to be $24nm$ [66]. The minimum length of the metal wires is set to $48nm$. The minimum center-to-center spacing of the vias is set to $96nm$. SADP related parameters are the same as those in Section 2.6. The spacer deposit width is set to $24nm$. The width and height of the routing grid are set to $48nm$. For trim mask design, the minimum resist width and space are set to $44nm$ and $46nm$, respectively [66]. The user-defined parameters in equation (3) and (4) are set to $\alpha = 0.15, \theta = 4, \eta = 10, \beta = 1, \gamma = 5$. The iteration upper bound is set as $l_0 = 3$ for the pin access driven rip-up and reroute scheme.

As discussed in Section 3.2.3.3, one single net may be deferred several times. The deferring cost bound is further set to avoid too much rerouting effort. We define $deferring\ cycle$ as the maximum number of times that a net is deferred before reaching the deferring cost upper bound. The "routability" is defined as the number of nets routed over the total number of nets in the design. The trade-off between routability, runtime and deferring cycle is illustrated in Fig. 3.9(a) for the benchmark "alu". As the deferring cycle increases, both the routability and runtime increase monotonically. The net deferring scheme improves the routability significantly during first few deferring cycles. After that, the runtime increases quasi-linearly while the amount of routability improvement degrades. Thus, the $deferring\ cycle$ is set to 3 for the global pin access planning scheme, which is enough for the routability improvement exploration. For results in Table 3.2, the step of increasing deferring cost is



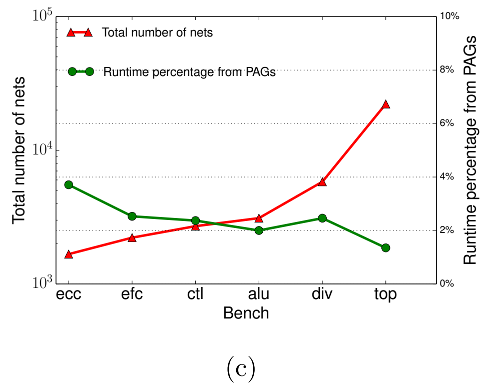


Figure 3.9: The computational efforts related to PAGs, (a) routability and runtime tradeoff varying the net deferring cost, (b) PAG size for benchmarks with different sizes, (c) runtime for updating PAGs during detailed routing.

set to 300 for better routability.

Apart from the A* search for the trial routes, the extra computational cost for pin access planning schemes is mainly associated with the construction and queries of the PAGs. Fig. 3.9(b) shows the node size of the PAGs constructed for various benchmarks. Since the graph partitioning is applied to each single row PAG, the PAGs of each design consist of various components. As shown in Fig. 3.9(b), the total number of nodes of the PAGs increases as the size of the design increases. On the other hand, the maximum number of nodes for each component is bounded by 2000. This means that, the runtime of determining feasible paths in each component of the PAGs is bounded by a small constant in Algorithm 3. Fig. 3.9(c) illustrates the analysis on the percentage of runtime associated with the PAG construction and queries. The number of nets increases monotonically as the design complexity increases. However, the runtime percentage from PAG construction and queries is bounded by a small constant, i.e., 5%. The trend of runtime percentage decrease from PAG construction and queries means the routing cost dominates the overall computational cost as the design complexity increases.

This study targets two-layer unidirectional detailed routing, which means no jogs are allowed on an individual routing layer. The main focus of this study is on the pin access planning during the detailed routing stage, while considering the difficulties from SADP-specific constraints. We believe comparing with a third party SADP-aware router can empirically validate the effectiveness of our proposed methodologies. We choose [64] as our baseline router due to its reported performance and efficiency and we obtain related routing results on our set of benchmarks with the help from the authors of [64]. In Table 3.2, we compare different SADP-aware detailed routing schemes, including the baseline SADP-aware routing from [64], regular routing with local pin access planning ("LPAP"), regular routing with local pin access planning and rip-up and reroute ("LPAP + RR"), regular routing with local & global pin access planning ("LPAP + GPAP") and regular routing with local & global pin access planning and rip-up and reroute ("LPAP + GPAP +RR"), i.e., overall routing flow. Overall, it is difficult to achieve 100% routability within reasonable runtime owing to problem complexity. To quantify the difference on the via count among various routing schemes, we average the number of vias from M3 to M2 over the number of routed nets, namely via number per routed net, which is listed as "V.p.n". The total wirelength, listed as "WL*", is the summation of the half perimeter wirelength for unrouted nets and actual wirelength for routed nets in terms of routing grid count. "OLL" denotes the total side overlay length defined in [64]. "Rout." denotes the percentage of routed nets over the total number of nets in the design and "CPU" denotes the runtime in seconds.

Table 3.2 demonstrates the strength of the pin access planning schemes for the regular routing over the 2-D detailed routing from [64]. For the "LPAP", the net order is computed with Eq. (3) with the term $DCost(n_k)$ ignored. Our router legalizes the routing wires for each routed net sequentially under the given set of design rules. The routing wires are all protected by the spacer after line-space array decomposition [67], which leads to the zero side overlay length. Moreover, the "LPAP" achieves better routability and shorter runtime by avoiding the extra efforts to maintain the overlay constraint graph from [64]. With "LPAP", our router gives better routability with competitive even shorter runtime compared with [64]. The cost is the 1.4% increase in the "V.p.n" and 5.8% increase in the "WL*". Since the regular routing does not allow jogs, extra vias are needed when routing wires change directions compared with 2-D routing scheme. We believe the 5.8% increase in the wire-length lower bound is reasonable considering the better routability and shorter runtime from the "LPAP" scheme. Furthermore, there is some variance in the routability improvement with "LPAP" and further improvement on the pin accessibility of all benchmarks are quite necessary for the ultimate routability. For example, for "ecc" and "ctl", the "LPAP" scheme gives similar routability as [64], which means the "LPAP" scheme could be further improved. Therefore, we compare two different approaches on top of "LPAP", i.e., "LPAP + RR" and "LPAP + GPAP". Compared to "LPAP" approach, the routability has been improved by 4.8% and 7.0% with the "LPAP + RR" and "LPAP + GPAP" approach, respectively. The routability improvement is still at the cost of slight degradations on the "V.p.n" and "WL*". Moreover, "LPAP + GPAP" approach is better than "LPAP + RR" with 2.8% routability improvement and better results on "V.p.n" and "WL*". In terms of runtime, all approaches are with similar scalability for small benchmarks, but "LPAP + GPAP" consumes larger runtime in the "top" benchmark due to heavy computational cost of the trial routes from net deferring procedure while the "LPAP + RR" stops at the stage of no routability improvement or iteration upper bound. The routing results with "LPAP + GPAP + RR" are further demonstrated in Table 3.2. The "LPAP + GPAP + RR" scheme achieves the best routability (96.4% on average) for all benchmarks, which is over 10% increase from [64], 9.2% increase from "LPAP" scheme, 4.0% increase from "LPAP + RR" scheme and 1.2% increase from the "LPAP + GPAP" scheme.

To achieve this high routability, there are some degradations on “V.p.n” and “WL*” and runtime increase compared with other schemes. Despite this overhead, we believe “LPAP + GPAP +RR” is a reasonable trade-off for the ultimate routability improvement. This study aims at pin accessibility, i.e., routability, and design rule clean routing results with some cost from other metrics. For different routing strategies, It shall be noted that the “WL*” and “V.p.n” for [64], “LPAP”, “LPAP + RR” or “LPAP + GPAP” might increase if similar routability was achieved since the remaining nets are difficult to route. The runtime from “LPAP + GPAP” and ”LPAP + GPAP + RR” is higher than other strategies because the net deferring procedure may perform trial routes of a net several times. To the best of our knowledge, this is the first work explicitly addressing the pin access planning strategies during detailed routing stage and we can achieve the best routability with the proposed techniques.

Table 3.1: Benchmark statistics for routing

Ckt	ecc	efc	ctl	alu	div	top
Net#	1671	2219	2706	3108	5813	22201
Cell#	1302	1197	1725	1802	3260	12576
Size(um^2)	21 x 21	20 x 19	24 x 24	20 x 19	31 x 31	57 x 56

Table 3.2: Comparisons on different sequential routing approaches

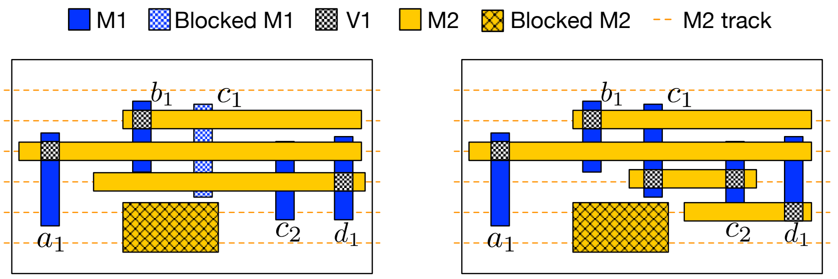
	[64]					LPAP					LPAP + RR					LPAP + GPAP					LPAP + GPAP + RR				
Ckt	V.p.n	WL*	OLL	Rout.	CPU(s)	V.p.n	WL*	OLL	Rout.	CPU(s)	V.p.n	WL*	OLL	Rout.	CPU(s)	V.p.n	WL*	OLL	Rout.	CPU(s)	V.p.n	WL*	OLL	Rout.	CPU(s)
ecc	2.31	41497	2775	91.1%	16.77	2.51	45102	0	91.2%	14.21	2.59	46248	0	94.4%	23.09	2.66	46588	0	96.4%	19.98	2.67	46519	0	97.4%	26.06
efc	2.31	54459	4703	82.5%	100.5	2.25	56457	0	88.1%	22.86	2.45	58848	0	92.1%	35.20	2.40	57834	0	94.9%	34.52	2.45	58717	0	96.2%	45.40
ctl	2.24	67470	5255	87.3%	93.80	2.26	71643	0	88.3%	22.39	2.45	72984	0	92.7%	36.05	2.42	72388	0	95.3%	37.14	2.50	73112	0	96.8%	47.31
alu	2.26	68491	5713	79.4%	143.4	2.22	73430	0	87.5%	28.32	2.44	76424	0	92.0%	53.09	2.44	75679	0	95.2%	45.92	2.47	75814	0	95.8%	61.92
div	2.29	139309	11267	79.4%	253.5	2.34	150356	0	87.9%	57.79	2.55	156615	0	91.8%	100.5	2.51	155704	0	94.6%	106.0	2.58	157625	0	96.2%	132.2
top	N/A	N/A	N/A	N/A	N/A	2.32	496228	0	88.2%	253.7	2.47	514499	0	92.7%	378.5	2.47	513366	0	95.3%	763.2	2.52	516902	0	96.0%	791.0
Avg.	0.901	0.902		0.870	1.943	0.915	0.960		0.918	0.361	0.984	0.997		0.960	0.567	0.982	0.992		0.988	0.912	1.000	1.000		1.000	1.000

3.3 Concurrent Pin Access Optimization

Section 3.2 proposes novel sequential planning schemes to address the pin access interference and enhance the routing solution qualities. However, pin access planning schemes depend on conventional grid-based search algorithms to obtain routing paths, where the pin access interference is incrementally resolved in a sequential routing procedure. This means the pin access interference on one metal layer, such as M2, is not optimally resolved, which leads to significant overheads on the runtime and solution qualities. To further addresses the pin access challenges, we propose the concurrent pin access optimization study for unidirectional routing. Our study takes advantage of the unidirectional routing to enable interval-based search and optimally the pin access interference on one metal layer.

3.3.1 Preliminaries and Problem Formulation

Pin Access Interference A typical outcome of routing resource competition on the M2 layer is SC pin access interference among I/O pins close to each other [106]. An example of pin access interference is shown in Fig. 3.10(a), where pin c_1 is blocked on the M2 layer due to the routing patterns connected to other I/O pins. However, pin access interference can be avoided with superior pin access optimization results as shown in Fig. 3.10(b). This illustrative example demonstrates that pin access optimization, including track location and spans of pin access intervals, is critical to resolving the pin access interference on the M2 layer.



(a)

(b)

Figure 3.10: Pin access interference on the M2 layer, (a) pin access failure, (b) pin access success.

In this section, we demonstrate that the concurrent pin access problem can be resolved through weighted interval assignment, which is more complex than conventional track assignment. Track assignment is an intermediate step between the global and detailed routing stages and targets the maximum assignment of routing intervals from global routing to a set of routing tracks [13, 50]. However, for pin access interference, track assignment is impossible without routing intervals for each I/O pin, which are actually the outcome of the detailed routing itself. This chicken and egg issue makes the pin access optimization problem more difficult than the conventional track assignment problem.

Problem Definition The concurrent pin access optimization problem builds on a set of pin access intervals. We take advantage of the unidirectional routing style on lower metal layers and propose track-based pin access interval generation for each I/O pin. Concurrent pin access optimization is performed in a panel-by-panel manner, where a panel corresponds to either a row (horizontal layer) or a column (vertical layer). We define the problem of concurrent pin access optimization free of routing conflict as follows.

Problem 4 (Concurrent Pin Access Optimization)**.**
Given a netlist, I/O pins and a set of routing tracks, concurrent pin access optimization has to find a set of pin access intervals such that each I/O pin is connected to one pin access interval, where there is no overlap among pin access intervals belonging to different nets.

Our approach guarantees that a feasible solution exists for Problem 4, which yields conflict-free pin access intervals for all I/O pins. We further implement a concurrent pin access router while accommodating advanced manufacturing constraints from Section 3.2.1, which greatly reduces initial routing congestion and leads to high-quality routing solutions.

3.3.2 Pin Access Interval Generation

A traditional placed-and-routed design with power/ground rails is inherently separated into panels, e.g., standard cell rows in the physical design floorplan. Without loss of generality, concurrent pin access optimization problem is solved panel-by-panel, which can also handle multiple panels simultaneously for scalable solutions.

We introduce track-based pin access interval generation for each I/O pin, where pin access intervals are generated for each pin within its net bounding box. Related notations are defined in Table 3.3. Fig. 3.11(a) enumerates

Table 3.3: Notations for concurrent pin access optimization

P	the set of pins
p_j	j^{th} pin in P
I_i	i^{th} pin access interval
I	the set of pin access intervals
$I_i^{p_j}$	i^{th} pin access interval for p_j
S_j	the complete pin access interval set for p_j on multiple tracks
l_i	the length for I_i
$f(I_i)$	the profit function for I_i
x_i	the binary variable for I_i
C_m	the m^{th} conflict pin access interval set
λ_m	the Lagrangian multiplier for C_m
C	sets of conflict interval set $\{C_m\}$

the pin access intervals for pin a_1, which is part of the net containing three pins (a_1, a_2, and a_3). The left and right edge of the net bounding box is by a_2 and a_3, respectively. For each I/O pin, a *minimum pin access interval* is the smallest metal strip to cover the pin, while a *maximum pin access interval* is the largest metal strip available within the net bounding box[1]. On track t_1, the maximum pin access interval of pin a_1 starts from the left edge to right edge of the net bounding box. On track t_2, the maximum pin access interval has to stop before the routing blockage. On track t_3, it shows the general scenario for the pin access interval generation for a_1. The net bounding box contains I/O pins (b_1 and d_1) that do not belong to the same net. For a_1, b_1 and d_1 are referred as diff-net pins. Then, when generating the pin access intervals for a_1, it is important to enumerate all the pin access intervals that

[1]If M2 routing is not favored for long nets, we can constrain pin access interval generation using an estimated M2 routing bounding box for its corresponding net, instead of using the net bounding box.

end at the vertical cutting lines of each diff-net pin, including $I_1^{a_1}$ and $I_2^{a_1}$. The reason is that $I_1^{a_1}$ is the maximum length of interval that can be used to access a_1 without blocking b_1. We do not enumerate all grids points between a_1 and b_1 because a router has the flexibility to choose any grid point on $I_1^{a_1}$ and accommodate metal/via constraints. Similarly, $I_2^{a_1}$ is generated to consider the pin access interference between a_1 and d_1. This principle of generation controls the number of pin access intervals while exactly capturing the pin access interference among I/O pins. There are 8 pin access intervals generated for pin a_1 across 3 tracks.

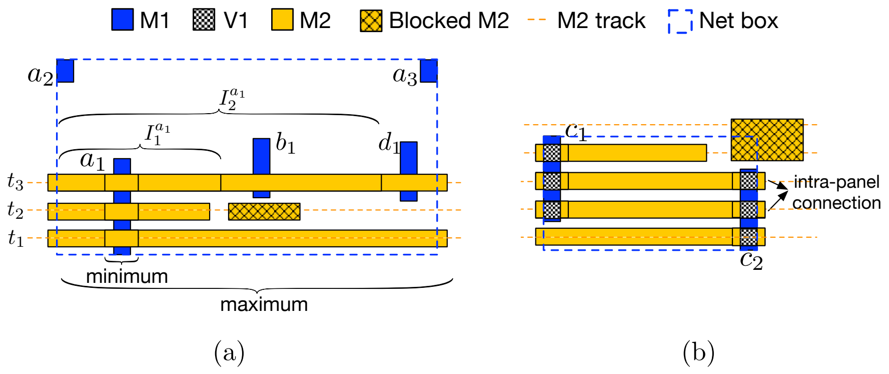
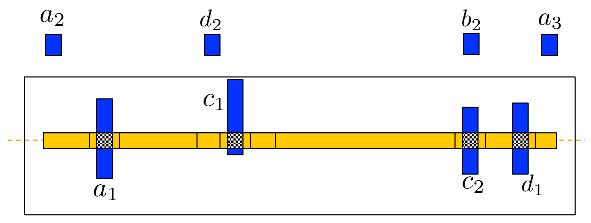


Figure 3.11: Pin access interval generation, (a) for pin a_1 across 3 tracks, (b) for pin c_1 and c_2 with intra-panel connection.

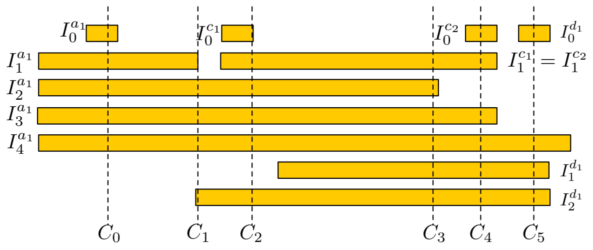
Suppose a given pin p with m pins on the left and n pins on the right within the net bounding box, the number of pin access intervals generated is $\mathcal{O}(m \cdot n)$ to enumerate all possible left and right edges of a pin access interval. The minimum and maximum pin access intervals can also be generated for pins c_1 and c_2 in Fig. 3.11(b). In particular, those pin access intervals resulting in intra-panel connections shall be preferred in the pin access optimization result, because they connect same-net pins without using external routing resources outside of the panel. Suppose conflict-free pin access intervals are assigned to I/O pins (one interval per pin) with a maximum and balanced interval length, providing much better pin accessibility for a router. Specifically, a router can choose any grid point on the pin access interval of an I/O pin to access that particular pin, without introducing extra pin access interference, i.e., routing congestion. This also reduces the number of vias because pin access intervals discourage switching metal layers, which reduces congestion.

3.3.3 Linear Conflict Set Detection

After track-based pin access interval generation, the pin access intervals may overlap/conflict with each other. We define a set of intervals to be a conflict interval set if the intersection of the intervals is non-empty. Fig. 3.12 shows an example of conflict interval sets. For a routing track in Fig. 3.12(a), all the intervals on the track are shown in Fig. 3.12(b). There are six conflict interval sets on the track, i.e., $C_0, C_1, \ldots, C_5$. Routing intervals $I_0^{a_1}$, $I_1^{a_1}$, $I_2^{a_1}$, $I_3^{a_1}$ and $I_4^{a_1}$ define the first conflict interval set C_0, because they share a common horizontal range. Similarly, pin access intervals $I_1^{a_1}$, $I_2^{a_1}$, $I_3^{a_1}$, $I_4^{a_1}$ and $I_2^{d_1}$ form another conflict set C_1. The target of conflict detection is to collect all the conflict interval sets on a track without redundancy. In general, this can be realized by generating a vector of intervals and scanning from the left to the right to detect the overlaps. The number of conflict interval sets generated is linear to the size of pin access intervals [50]. As I/O pins overlap a set of tracks, we further collect conflicts among all routing tracks to obtain a complete set of conflict interval sets.



(a)



(b)

Figure 3.12: Linear conflict set detection for pin access intervals, which can be applied to multiple tracks obtain complete conflict sets.

3.3.4 Weighted Interval Assignment Problem

Among all pin access intervals for all pins, the concurrent pin access optimization is a weighted interval assignment problem, which aims at assigning conflict-free intervals to I/O pins while maximizing the overall interval length. If a router treats conflict-free intervals as partial routes, maximum and balanced pin access interval lengths among all I/O pins lead to better pin accessibility during a routing procedure. The concurrent pin access problem requires that each pin is assigned exactly one interval to guarantee pin access feasibility, while the track assignment targets the maximum assignment of routing intervals among a set of tracks [13, 50]. This means previous track assignment solutions cannot be directly applied here.

To obtain the optimal solution of the current pin access optimization, we formulate the weighted interval assignment problem as an integer linear programming problem. Related notations are defined in Table 3.3. The objective is to achieve the maximum and balanced length of pin access intervals for all pins without any conflict. For the ILP formulation in Formula (3.1), a pin access interval I_i corresponds to a binary variable x_i that indicates whether I_i is selected or not. The objective (3.1a) explicitly favors the selection of an interval covering multiple I/O pins by counting its corresponding variable multiple times. To obtain an interval selection with balanced interval length, we set $f(I_i) = \sqrt{l_i}$ because the square root function generates more balanced solutions while maximizing the interval length, compared to a linear function. Constraint (3.1b) denotes only one interval can be selected for pin p_j. Furthermore, only one pin access interval within any conflict interval set should be selected. A simple way to forbid conflicts is to add a constraint $x_i + x_{i'} \leq 1$ for every conflict interval pair $(I_i, I_{i'})$, which introduces large numbers of constraints, i.e., the number of constraints will be quadratic to the number of pin access intervals. To avoid an unmanageable number of constraints, we utilize the linear conflict interval set detection from Section 3.3.3. For each conflict set C_m, only one constraint is added to avoid all possible pin access interval conflicts from C_m. This means at most one interval in C_m can be selected and the equivalent linear constraint is shown in (3.1c). The complete ILP formulation is given as follows.

$$
\begin{aligned}
\textbf{max} \quad & \sum_{p_j \in P} \sum_{I_i \in S_j} f(I_i) \cdot x_i && && \text{(3.1a)}\\
\textbf{s.t.} \quad & \sum_{I_i \in S_j} x_i = 1, && \forall p_j \in P, && \text{(3.1b)}\\
& \sum_{I_i \in C_m} x_i \le 1, && \forall C_m \in C, && \text{(3.1c)}\\
& x_i \in \{0, 1\}, && \forall I_i \in I. && \text{(3.1d)}
\end{aligned}
$$

Due to the minimum pin access interval generation of each I/O pin shown in Fig. 3.11, we have the following theorem for the ILP formulation.

Theorem 1. *The ILP formulation* (3.1) *has a feasible solution.*

Proof. We can select the minimum pin access interval for each I/O pin, which leads to a set of pin access intervals without conflict. It corresponds to a feasible solution for (3.1). □

3.3.5 Lagrangian Relaxation

The main challenge in solving the ILP comes from the conflict constraints in Eq. (3.1c). To overcome the scalability issue of the ILP formulation, we propose a Lagrangian relaxation (LR)-based approach, which relaxes the conflict constraints to the objective as penalties. The problem formulation for the LR problem is shown in Formula (3.2). A set of Lagrangian multipliers (LMs) $\{\lambda_m\}$ is introduced to relax the conflict constraints, while the equality constraints are kept in Eq. (3.1b). In particular, as the ILP problem is feasible according to Theorem 1, the LR problem in Eq. (3.2a) is bounded during the iterative solving procedure.

$$\begin{aligned} \mathbf{max} \quad & \sum_{p_j \in P} \sum_{I_i \in S_j} f(I_i) \cdot x_i - \sum_{C_m \in C} \lambda_m \cdot (\sum_{I_i \in C_m} x_i - 1) && (3.2a) \\ \mathbf{s.t.} \quad & \text{(3.1b) and (3.1d).} && (3.2b) \end{aligned}$$

For the LR-based algorithm, we first solve the concurrent pin access optimization problem without conflict constraints in Eq. (3.1c), i.e., all λ_m are set to 0. As will be discussed later, each iteration of LR is solved using an efficient greedy algorithm. After obtaining an initial solution, the pin access interval assignments with conflict constraint violations are detected. For any violation, we gradually increase the penalty in the objective by adjusting the corresponding λ_m. The method for updating λ_m based on the current solution is critical for the convergence of the LR-based algorithm. To guarantee convergence, we adopt the subgradient descent method for updating LMs, defined in Eq. (3.3), where k is the number of iterations and t_k is the step size [41]. In our experiments, we set t_k to $1/k^\alpha \cdot L_m$, where α is 0.95 and L_m is the length of intersection among all conflict intervals in C_m.

$$\lambda_m^{k+1} = \max(0, \lambda_m^k + t_k \cdot (\sum_{I_i \in C_m} x_i - 1)). \tag{3.3}$$

In each iteration to solve the LR subproblem, the objective defined in Eq. (3.2a) is a weighted summation of x_i with fixed LMs. Let the weight of

Algorithm 4 LR Sub-Routines

```
1: function maxGains($I, gains$)
2:     Define $sel$ as selected intervals;
3:     Sort $I$ in the non-increasing order of $gains$;
4:     Select intervals from $I$ to $sel$ until all pins are covered;
5:     Return $sel$;
6: end function
7: function penalize($sel, C, penalties$)
8:     Define $vio \leftarrow 0$ as the violation number;
9:     for each conflict $C_m \in C$ do
10:        if more than 1 interval selected in $sel$ from $C_m$ then;
11:            $vio \leftarrow vio + 1$;
12:            Update $penalties[i]$ for each $I_i \in C_m$ (Eq. (3.3));
13:        end if
14:    end for
15:    Return $vio$;
16: end function
```

each x_i represent the gain of the interval. We restate Formula (3.2) as follows. Given a set of pins and corresponding intervals with different gains, select one interval for each pin such that the total gain is maximized. We develop a greedy algorithm to compute the maximum total gain efficiently, shown as function maxGains in Algorithm 4. All the intervals are sorted in the non-increasing order of *gains* in line 3. We break the ties of equal gains by the number of same-net pins an interval covers. We prefer selecting pin access intervals covering more same-net I/O pins because intra-panel connections are preferred as mentioned in Section 3.3.2. As one pin can only be assigned one pin access interval, we skip an interval if its corresponding pin has been assigned another interval in line 4. The penalty of each interval I_i is defined as the summation of its corresponding LMs in Eq. (3.2a). In function penalize, we scan through each conflict (C_m) in lines 9 to 14. If multiple intervals are selected within C_m, the corresponding penalty is updated in line 12.

An iterative LR approach generates good convergence behavior if we can optimally solve LR subproblem. Given notations in Table 3.3, the optimality condition for the greedy algorithm is analyzed as follows.

Theorem 2. *The greedy LR sub-routines in Algorithm 4 solves Formula* (3.2) *optimally if* $S_i \cap S_j = \emptyset, \forall p_i, p_j \in P$.

Proof. $S_i \cap S_j = \emptyset, \forall p_i, p_j \in P$ means that no pin access interval is associated with more than one pin. Objective (3.2a) generates the net profit for each pin access interval candidate, i.e., the net multiplier before each binary variable. Therefore, we greedily select pin access interval with the maximum profit for each pin, which delivers optimal solution to Formula (3.2). $\square$

Although intra-panel routing connections may introduce dependencies among pin access interval sets for I/O pins, i.e., breaking the optimality condition in (2), the greedy algorithm empirically works well in computing the solution and reaching convergence in our experiments. The LR-based algorithm is shown in Algorithm 5. We first compute the profit of each interval in line 1. Within each iteration of the kernel loop from line 4 to line 10, the relaxed problem is solved by our greedy algorithm in MAXGAINS in line 5. In line 6, we compute the penalty for each interval and update LMs according to Eq. (3.3) in PENALIZE. The LR iterations continue until there is no violation or the iteration upper bound is reached. We observe small oscillations on the

Algorithm 5 LR-based Pin Access Optimization

```
Input: Pins ($P$), pin access interval vector ($I$), complete conflict interval sets ($C$) and iteration upper bound ($UB$).
Output: A vector of pin access intervals $PI$ for $P$.
 1: Initialize $profits$ as the profit vector for $I$ using Eq. (3.2a);
 2: Define $penalties$ as penalty vector with zeros for $I$;
 3: Define $k \leftarrow 0$, $min\_vio \leftarrow \infty$;
 4: while $min\_vio > 0$ and $k < UB$ do;
 5:     $sel \leftarrow$ maxGains($I, profits - penalties$);
 6:     $vio \leftarrow$ penalize($sel, C, penalties$);
 7:     if $vio < min\_vio$ then
 8:         $min\_vio \leftarrow vio$, $PI \leftarrow sel$;
 9:     end if
10: end while
11: Greedy conflict removal among $PI$;
12: Return $PI$;
```

solutions when the number of violations approaches zero, so we record the best solution throughout the LR iterations in lines 7 to 9. The LR scheme usually reduces violations rapidly, but it cannot guarantee zero conflicts. Greedy refinement is performed to remove remaining conflicts in line 11. For any conflict interval set, this refinement process tries to shrink conflict intervals into minimum intervals to remove that particular conflict.

3.3.6 Concurrent Pin Access Routing

The routing resource competitions among those pin access intervals from concurrent pin access optimization have been efficiently resolved. Therefore, we implement a concurrent pin access router (CPR) to take advantage of these pin access intervals and obtain net connections. The pin access intervals are treated as partial routes connected to its associated pins, which are fed into a negotiation-congestion-based router [23]. As shown in Fig. 3.13(a), the M2 pin access intervals from pin access optimization are connected to pins a_1, a_2 and a_3. Some detours may happen as metal-1 (M1) pins are connected to pin access intervals (partial routes) on the M2 layer. In Fig. 3.13(b), the routing results are obtained directly with a negotiation-congestion-based routing scheme. This introduces more vias because the router tries to avoid congested routing regions and routing blockages with unidirectional routing patterns. However, via minimization is particularly important for short-net routing in advanced technology nodes [24].

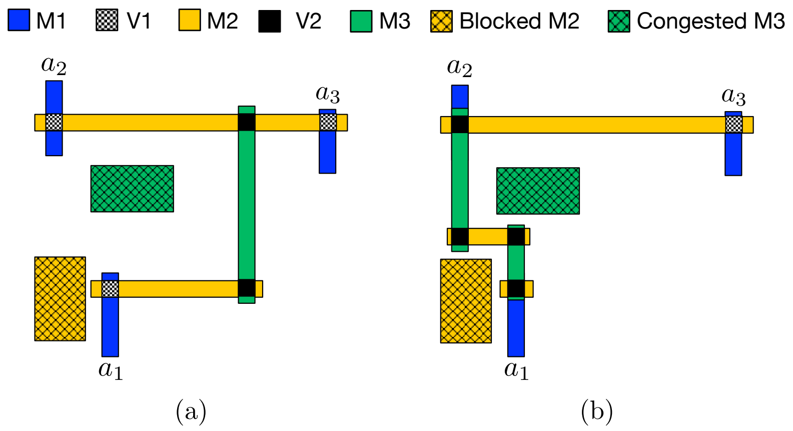


Figure 3.13: (a) Routing with pin access optimization introducing extra routed wirelength; (b) Routing without pin access optimization introducing extra routed vias.

As shown in Figure 3.14, given the netlist and routing grid plane, CPR starts with concurrent pin access optimization on lower metal layers, which efficiently resolves routing resource competition and greatly reduces the amount of

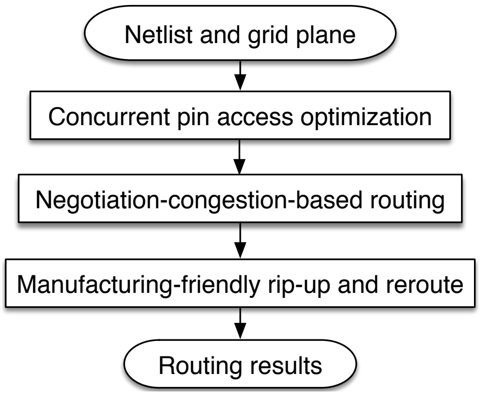

Netlist and grid plane
Concurrent pin access optimization
Negotiation-congestion-based routing
Manufacturing-friendly rip-up and reroute
Routing results

Figure 3.14: Concurrent pin access routing flow.

grid congestion for the subsequent routing steps. I/O pin promotions are performed based on the pin access intervals from pin access optimization. Treating pin access intervals as partial routes, the negotiation-congestion-based routing is performed to finish all net connections, where design rule violations are mitigated using forbidden grid costs [23].

Our router follows the negotiation-congestion-based routing scheme to reduce the net ordering effects during sequential detailed routing [75]. The routing of each net on the grid plane follows the A^* search and only unidirectional routing patterns are allowed on a specific routing layer. The cost $C(g_i)$ for grid g_i is calculated as follows.

$$C(g_i) = base(g_i) + forbid(g_i) + \beta \cdot usage(g_i) + hist_k(g_i), \tag{3.4}$$

$$overflow(g_i) = \max(0, usage(g_i) - 1), \tag{3.5}$$

$$hist_k(g_i) = hist_{k-1}(g_i) + \gamma \cdot overflow(g_i). \tag{3.6}$$

In Eq. (3.4), $base(g_i)$ is the grid base cost, $forbid(g_i)$ is the forbidden cost for via grid to enable minimum center-to-center spacing rule [111], $usage(g_i)$ is the number of nets currently using the grid and $hist_k(g_i)$ is the history cost in the current iteration. In Eq. (3.6), $hist_{k-1}(g_i)$ is the history cost in the previous iteration and the grid overflow, $overflow(g_i)$ is calculated using Eq. (3.5). β and γ are user-defined parameters. The routing procedure consists of two phases, where the first phase contains independent routing iterations and the second phase is a negotiation-congestion-based rip-up and reroute [23]. For independent routing iterations, all nets can freely use the available routing resources. The usage cost, $\beta \cdot usage(g_i)$, is updated to differentiate multiple optimal paths for each net. The independent routing iterations stop when the total grid overflow (based on Eq. (3.5) of the current iteration is larger than that of the previous iteration. The second phase is negotiation-congestion-based rip-up and reroute. A queue of congested routing grids is constructed from the routing results of independent routing iterations. Then, the congestion on each grid is resolved through rip-up and reroute and the history cost of each grid is updated according to Eq. (3.6). The negotiation-congestion-based rip-up and reroute stop when the routing grid queue becomes empty.

To make the negotiation-congestion-based routing aware of SC pin access, during the routing of each net, only the pins and associated pin access intervals of the current net are available, while the pins and associated pin access intervals from remaining nets are treated as blockages. We further perform line-end extensions and rip-up and reroute to accommodate the same manufacturing constraints discussed in Section 3.2.1 and enable SADP-friendly cut masks shown in Figure 1.3. CPR is extendable to technology-dependent manufacturing constraints, e.g., SAMP with unidirectional routing.

3.3.7 Experimental Results

We implement concurrent pin access router (CPR) in C++ and all experiments are performed on a Linux machine with 3.4GHz Intel®core and 32GB memory. All benchmarks are from Section 3.2.4 and mapped to a routing grid plane. The iteration upper bound (UB) for LR is set as 200. The metal and via manufacturing rules are set the same as discussed in Section 3.2.1. For the grid cost computation, the base cost is set as 1 for metal and via grids. The forbidden cost is set as 10 for via grids, β is set as 10^{-4} and γ is set as 0.25. We perform the concurrent pin access optimization on the M2 layer, where one SC row (10 x M2 tracks) is one routing panel. All experiments are performed using the same benchmarks, design rule settings and evaluation metrics listed in Section 3.2.4. We quantify routing solutions with the number of routed nets over the total number of nets ("Rout."), the number of vias ("Via#") and wirelength ("WL"). "Via#" is the total number of vias for all nets estimated by via per routed net. "WL" is the summation of half perimeter wirelength of unrouted nets and actual grid wirelength for routed nets.

3.3.7.1 Quantifying the LR Suboptimality

The comparisons on runtime scalability and solution quality in terms of objective value between LR and ILP are shown in Figs. 3.15(a) and 3.15(b),

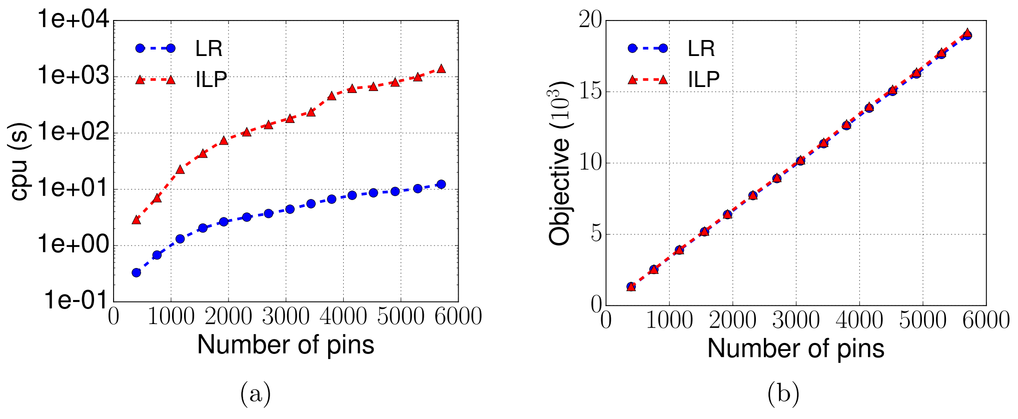
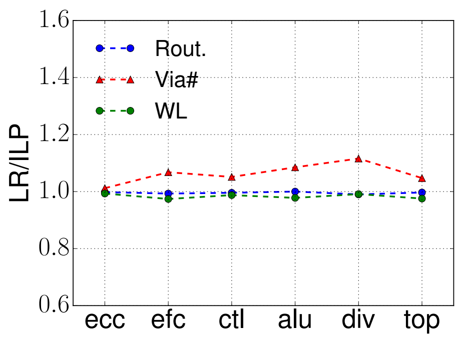


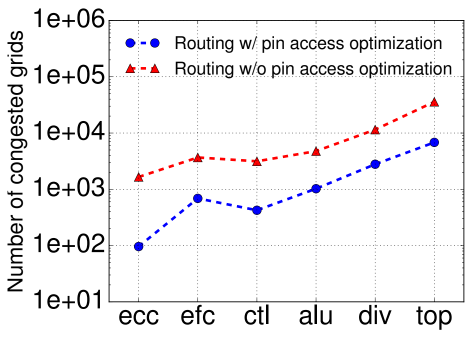
Figure 3.15: Comparisons between LR and ILP for different number of pins, (a) runtime, (b) objective value.

respectively. For Fig. 3.15(a), we observe that the runtime of the ILP is not scalable to a large number of pins and the runtime of the LR algorithm has much better scalability. As shown in Fig. 3.15(b), the objective value achieved with LR algorithm is close to the optimal value returned by the ILP solution. Therefore, our proposed LR algorithm delivers a scalable solution to the concurrent pin access optimization problem.

Fig. 3.16(a) further demonstrates the comparisons of the routing solution qualities obtained with LR-based and ILP-based pin access optimizations. We observe that the "Rout." and "WL" are similar across different designs with LR-based and ILP-based pin access optimization schemes. Furthermore, the "Via#" for the routing solution with LR-based pin access optimization is around 5% more than that with ILP-based pin access optimization. The reason is that ILP-based optimization delivers optimal and balanced pin access intervals for each I/O pin, which leaves more flexibility for via locations



(a)



(b)

Figure 3.16: (a) LR over ILP after obtaining the routing solutions, (b) the number congested routing grids before rip-up and reroute stage.

during the routing phase. The LR-based optimization depends on a greedy conflict removal phase to generate legal pin access intervals for each I/O pin, which penalizes the solution quality of the pin access optimization. We believe the LR-based pin access optimization represents a desirable trade-off between solution quality and runtime scalability, compared to the ILP-based approach.

3.3.7.2 Comparison with Related Work

Table 3.4 compares concurrent pin access router (CPR) with two state-of-the-art SADP-friendly routers [23,111]. Section 3.2 [111] proposes sequential pin access planning schemes with design rule legalizations during sequential routing of each net. [23] devises a negotiation-congestion-based router with modified routing grid models for SADP, with explicit considerations of pin access and routing blockages[2]. To make the negotiation-congestion-based routing aware of SC pin access, during the routing of each net, only the pins of the current net are available, while the pins from remaining nets are treated as blockages [23]. For [23], the design rules violations are mitigated with forbidden via grid cost and rip-up and reroute iterations. CPR combines the concurrent pin access optimization results with the same negotiation-congestion-routing scheme as [23]. Although [23] and our router can initially finish all net connections (100% routability), they introduce many design rule violations because design rule violations can only be mitigated through grid cost manipulations. We treat those nets introducing violations as unrouted nets, which generates design-rule-clean routing results for fair comparisons.

CPR obtains 1.5% routability improvement, 23.8% via number and 16.0% wirelength reductions compared to sequential pin access planning in Section 3.2 [111] as shown in Table 3.4. The reason is that the baseline router in Section 3.2 [111] depends on pin access planning schemes to improve routability, which resolves routing resource competitions in a greedy manner to generate design-rule-clean routing results. Although the net deferring technique enables dynamic net reordering, the routing scheme in Section 3.2 [111] is still sequential in nature, which introduces significant overhead in terms of runtime, wirelength and via number. Our approach provides the maximum and balanced amount of horizontal M2 partial routes connected to each pin,

[2]The authors of [23] make their router explicitly consider design rules listed in Section 3.2.1 and run on a Linux machine with 2.4GHz Intel(i5) core and 8GB memory.

which leads to more flexible via locations and pin access. We further compare with [23] (routing w/o pin access optimization), the data of which are directly provided by the authors of [23]. CPR improves the "Rout." by 3.8% and reduces the "Via#" by 10.8%, while the trade-off is only 0.2% "WL" degradation. In particular, this "WL" degradation is negligible considering 3.8% more routability from CPR because remaining nets from [23] are difficult to route and may introduce more "WL" overhead. The "Via#" reductions and slight "WL" degradations are both due to the partial routes from pin access intervals shown in Fig. 3.13(a). From the solutions of [23], there exist routing patterns shown in Fig. 3.13(b), which means the router tries to switch between metal layers to resolve routing congestion and minimize routing cost. CPR reduces "Via#" by $> 10\%$ with pin access optimization as shown in Fig. 3.13(b) in spite of some added detours, i.e., 0.2% wirelength degradation on average in Table 3.4. In advanced technology nodes, via minimization is particularly important as the via resistance has been increasing dramatically [24].

The runtime of CPR consists of the runtime from concurrent pin access optimization and unidirectional routing. In Table 3.4, CPR is significantly faster (12.7×) than the sequential planning planning in Section 3.2 [111] and estimated to be 2.5× faster than [23] considering different machine configurations. The major reason is that the routing resource competitions on the M2 layer are concurrently resolved using the pin access optimization approach, which greatly reduces the initial congested routing grids for the router. Congested routing grids are those grids occupied by more than one net. The sequential pin access planning depends on detours and rip-up and reroute to avoid congestions, which significantly increases the runtime as the design size becomes larger. [23] depends on history cost and rip-up and reroute to reduce congestion, which is computationally more expensive than our pin access optimization approach in terms of congestion reduction. We demonstrate efficient congestion reduction from concurrent pin access optimization in Fig. 3.16(b). The negotiation-congestion routing consists of two stages, i.e., the independent routing stage and the rip-up and reroute stage [23]. The rip-up and reroute stage takes congested routing grids as input and resolves grid congestion one by one. With concurrent pin access optimization, we achieve a 5-10$\times$ reduction in terms of congested routing grids, which greatly reduces the rip-up and reroute efforts. This further generates significant speed-up compared to state-of-the-art manufacturing-friendly routers [23, 111].

Table 3.4: Comparisons on solution qualities of different routing approaches

Ckt	Net#	Size(um^2)	Sequential pin access planning [111]				Routing w/o pin access optimization [23]				CPR			
			Rout.(%)	Via#	WL	cpu(s)	Rout.(%)	Via#	WL	cpu(s)	Rout.(%)	Via#	WL	cpu(s)
ecc	1671	21x21	96.41	6482	46588	19.98	94.55	5409	38428	10.00	97.25	4907	40465	2.01
efc	2219	20x19	94.91	8558	57834	34.52	92.83	7989	52329	15.60	96.80	7418	51973	3.69
ctl	2706	24x24	95.27	10573	72388	37.14	92.42	9327	64217	17.80	96.86	8757	63900	3.69
alu	3108	20x19	95.17	11645	75679	45.92	93.37	10496	64604	20.10	97.01	9371	62249	5.24
div	5813	31x31	94.60	22829	155704	106.0	92.12	21001	139811	47.70	95.89	19665	139201	24.32
top	22201	57x56	95.33	82644	513366	763.2	93.44	73487	434051	147.4	96.79	65167	436972	40.37
Avg.			95.28	23789	153593	167.8	93.12	21285	132240	43.1	96.77	19214	132460	13.22
Ratio			0.985	1.238	1.160	12.69	0.962	1.108	0.998	3.26	1.000	1.000	1.000	1.000

3.4 Summary

This chapter proposes sequential pin access planning and concurrent pin access optimization studies for unidirectional routing in advanced technology nodes. The sequential pin access planning features pin access LUT construction for a given library, local and global pin access planning to improve the pin accessibility during the SADP-aware regular routing. Compared to the 2-D SADP-aware detailed router, the sequential pin access planning scheme can achieve significant improvement in terms of the overlay and routability.

The concurrent pin access optimization is modeled as a weighted interval assignment problem and solved optimally with an ILP formulation. We further propose an LR-based algorithm for scalable solutions to the weighted interval assignment problem, which leads to similar routing solution quality as an ILP-based approach. Due to the 5-10$\times$ reduction in initial routing congestions, CPR generates much better routing solutions than the state-of-the-art SADP-friendly routers.

Chapter 4

Redundant Local-Loop Insertion

4.1 Introduction

Due to the continued scaling of semiconductor technologies, the manufacturing process is becoming more and more sensitive to process variations and random failures. In particular, via and wiring failures are major causes for the yield loss of integrated circuits (ICs) during back-end-of-line (BEOL) processes [15]. To reduce potential via and wiring failures at the post-routing stage, redundant via [58, 68] and redundant wire [17, 48] insertions have been proposed for manufacturing yield improvement. In advanced technology nodes, metal interconnect stack consists of multiple metal layers with various metal widths and spacings [43].

For upper metal layers, the wiring width and spacing are relaxed to provide low resistance connections for robust power delivery and improved timing (due to less RC parasitics) for long nets. For lower metal layers, the geometric scaling leads to high metal density, which is enabled by complex

This chapter is based on the journal "Xiaoqing Xu, Yibo Lin, Meng Li, Jiaojiao Ou, Brian Cline, and David Z. Pan. Redundant local-loop insertion for unidirectional routing. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), preprint, 2017." I am the main contributor in charge of problem formulation, algorithm development and experimental validations."

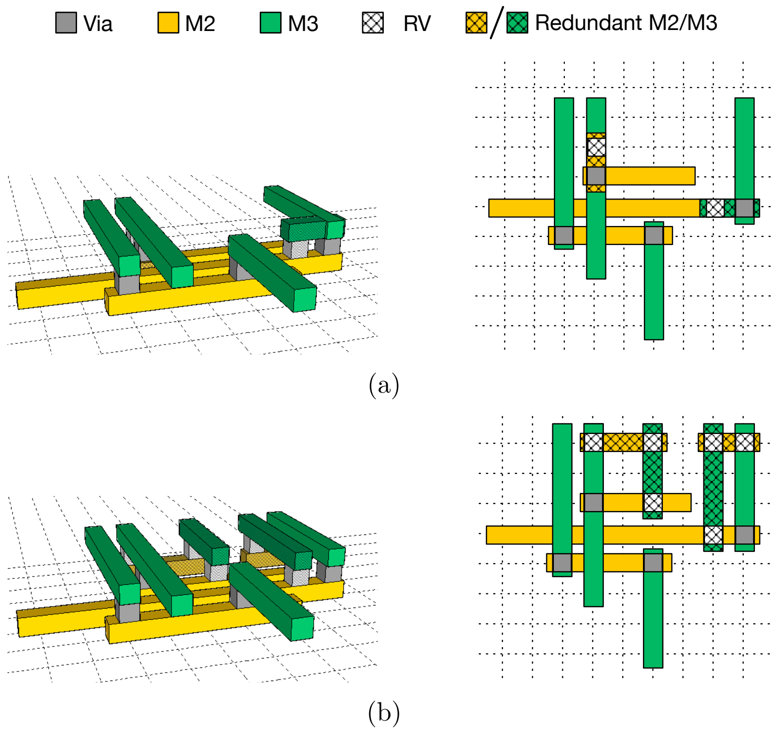


Figure 4.1: 3D view (left) and top view (right) of (a) RVI with wire bending, (b) RLLI for unidirectional routing patterns.

design-for-manufacturing constraints, such as multiple patterning lithography and unidirectional layout. Redundant via insertion (RVI) has been widely used in industry to improve the manufacturing yield of metal interconnections, where non-minimum width wires and wire bending are allowed. In $10nm$ and beyond, RVI is still feasible, but only for upper metal layers with relaxed metal pitches. On the other hand, for lower metal layers with tight pitches, unidirectional routing has become the mainstream routine, which complies with the underlying multiple patterning constraints [61, 89]. These unidirectional routing constraints make conventional RVI, e.g., double via, obsolete because unidirectional routing patterns forbid off-track wiring, i.e., wire bending, for the metal coverage of redundant vias. As shown in Fig. 4.1(a), two redundant vias have been inserted for associated single vias. The Metal-2 (M2) and Metal-3 (M3) tracks are horizontal and vertical, respectively. The redundant via (RV) on the M2 track introduces M3 wire bending and vice versa, which are strictly forbidden under the restrictive unidirectional routing style. Instead, as demonstrated in Fig. 4.1(b), redundant local-loop insertion (RLLI), as a supplementary scheme for lower metal layers, can simultaneously insert redundant vias and redundant wires for yield improvement of unidirectional routing [15]. A redundant local loop (RLL) introduces via and wiring redundancy, e.g., the redundant M2/M3 patterns in Fig. 4.1(b), such that all redundant metal patterns are on-track, which complies with the unidirectional routing style.

[15] and [8] first introduced the local loop concept back in 2006. An ad-hoc local-loop insertion routine was proposed at the post-routing stage. A recent work [44] studied the yield and timing impact of local loops of different sizes. It confirmed the yield enhancement and demonstrated very small or negligible timing impact using local loops with empirical timing simulations. However, comprehensive delay model analysis and timing simulations for various RLL configurations were not provided [44]. Furthermore, continued technology scaling has imposed advanced manufacturing constraints on via layers. Among them, self-aligned via (SAV) [9,88] and via density constraints [47] are particularly important for manufacturing via patterns in advanced technology

nodes. In addition, our predictive timing simulations show that significant timing degradations could be introduced with specific kinds of RLLs. This means that those kinds of RLLs should be strictly forbidden to guarantee a bounded and negligible timing impact for RLLI. It should be noted that for full-chip local-loop insertion, these advanced manufacturing constraints on via patterns and the bounded timing impact from RLLI have to be considered, but the prior work [8, 15, 44] focused on proving and validating the local-loop concept, and did not address the advanced manufacturing constraints and timing impact aforementioned.

Moreover, all prior approaches for RLLI are greedy in nature [8, 15, 44] and no systematic algorithms were proposed for local-loop insertion in a full-chip manner. In contrast, traditional RVI has been extensively studied with various advanced optimization schemes, including the maximum independent set problem formulation, solved with fast heuristics [58–60], and 0-1 integer linear programming (ILP) formulation, solved with speed-up techniques [56, 57, 59, 62]. The conflict constraints for the traditional RVI problem are purely local, which means only redundant-via candidates for neighboring single vias will introduce conflicts during RVI. This is the major reason for the high quality solutions obtained from fast heuristics and speed-up techniques for RVI.

However, for RLLI, one RLL candidate may consist of multiple via and metal grids, which induces long-range conflict constraints among RLL candidates. Meanwhile, redundant vias within one RLL candidate may cross multiple density windows. This means redundant vias within RLL candidates need to be balanced under via density constraints across multiple density windows. Therefore, global optimization schemes are strongly needed to achieve better performance than the simple greedy scheme [8, 15, 44]. Moreover, RLLI is not merely about maximizing insertion rate, because reducing the insertion cost is critical to routing resource usage, which impacts both timing and yield for various redundant local-loop candidates (RLLCs). It is important to combine the optimization of insertion rate and cost while accommodating conflict and density constraints, which makes ILP a competitive candidate for the global optimization scheme.

To the best of our knowledge, as technology moves toward unidirectional routing and vias continue to scale to extremely small geometries, which are difficult to yield and have high parasitic resistance, we expect increasing adoption of local loops in $10nm$ and beyond. This study proposes the first global optimization engine for full-chip local-loop insertion, in consideration of advanced manufacturing constraints and bounded timing impact guided by SPICE simulations. Under a bounded search space, we enumerate the RLLCs for each single via. The timing impact is quantified and bounded with comprehensive Elmore delay model analysis and timing simulations. With routing grid models, we further analyze the conflict constraints among RLLCs and formulate the RLLI issue as a binary ILP problem, which simultaneously improves the insertion rate and reduces the overall cost of inserted RLLs. Although the number of RLLCs generated is controllable by limiting the search space, it is still much larger than that of the traditional RVI problem, which means the ILP formulation is not scalable to large designs with large numbers of single vias. However, the theoretical analysis reveals that, due to the special constraint structures of the ILP formulation, linear programming (LP) relaxation leads to a solution that is intrinsically close to the ILP-integral solution with appropriate rounding schemes [19, 50]. Thus, we further propose the iterative relaxation and linear programming solving (IRLS) with an incremental search scheme to achieve scalable solutions with affordable performance degradations [109].

4.2 Preliminaries

4.2.1 Redundant Local Loop

For unidirectional routing patterns, RLLs are inserted to reduce the failure rate of within-loop single vias (SVs) and one RLL could cover multiple SVs as shown in Fig. 4.2, where RLLs are inserted involving the M3, M2 and Via-2 (V2) layers. RLL_1 consists of 3 redundant M3 grids, 3 redundant M2 grids and 1 redundant-via grid while RLL_2 includes 5 redundant M3 grids, 3 redundant M2 grids and 2 redundant-via grids. RLL_1 has 3, instead of 5, redundant M3 grids due to existing M3 routing patterns, which are constrained by upper-level metal connections within a routing solution. In practice, we need to differentiate the cost of RLLs with different configurations and the reasons are twofold. First, the RLLs with less number of redundant vias are preferred since a larger number of vias leads to a larger timing impact of a local-loop structure [44], which will be demonstrated with comprehensive SPICE simulations in Section 4.4.1. Second, RLLs with different configurations may lead to distinct yield impacts, which highly depends on the technological setup and analysis [15,44]. Therefore, we define the configuration and cost of an RLL as follows.

Definition 7 (Redundant Local Loop)**.** *An RLL with the configuration as* $rm_{x+1} \times rm_x \times rv_x$ *is defined as a loop structure with* rm_{x+1} *redundant grids on the* $x+1$ *metal layer,* rm_x *redundant grids on the* x *metal layer and* rv_x *redundant grids on the* x *via layer. Its cost is defined as* $\alpha \cdot rm_{x+1} + \beta \cdot rm_x + \gamma \cdot rv_x$*, where* α*,* β *and* γ *are user-defined parameters.*

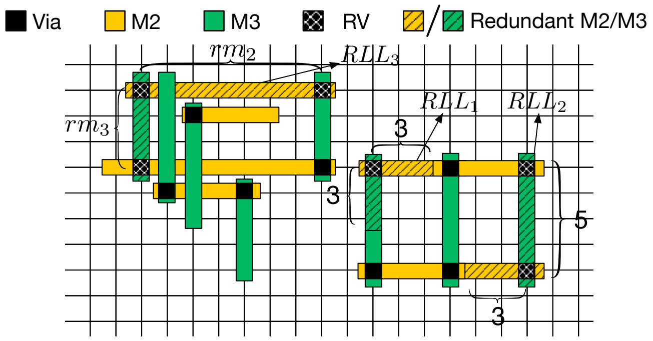


Figure 4.2: RLL_1, RLL_2 and RLL_3 with configurations as $3 \times 3 \times 1$, $5 \times 3 \times 2$ and $rm_3 \times rm_2 \times 3$, respectively.

Since the RLLI problem attempts to insert local-loop structures for SVs, we can enumerate valid RLLCs for an SV within a limited local search space [44]. We limit the local search space by bounding the number of redundant metal grids (i.e., rm_{x+1} or rm_x) by a pre-determined parameter. Moreover, various RLLCs for an SV can be differentiated using the cost definition above, where α, β and γ are set to make the cost inversely proportional to the yield and timing improvement for an RLLC.

4.2.2 Advanced Manufacturing Constraints

Via-Pattern Constraints In the $10nm$ node and beyond, the center-to-center spacing of lower via layers is in the sub-lithographic domain [61]. SAV patterning [9, 88] is a promising candidate for the sub-lithographic printing of via patterns. As shown in Fig. 4.3(a), the neighboring vias in the horizontal direction are merged into "Via Cuts" to enable SAV patterning for vias within the same via cut. The SAV patterning takes advantage of the line spacers from manufacturing procedure and selectively etches the via trenches defined by the via cuts, which enables sub-lithographic printing of via patterns within the same via cut. The patterning of via cuts is technology dependent and correlated to the M2/M3 patterning on the lower/upper metal layer. The details on the manufacturing procedures are introduced in [9, 88]. This study considers basic SAV constraints in [88]. For a single via in a routed net, the neighboring via grids along the upper metal layer direction, such as vertical M3, are not available for RVs and the neighboring via grids along the lower metal direction, such as horizontal M2, are only available for same-net redundant vias. An example is shown in Fig. 4.3(b). Our RLLI scheme only depends on the routing grid model and can be easily adapted to incorporate more complicated SAV constraints.

Much as lithographic constraints - such as SAV - are imposed on via

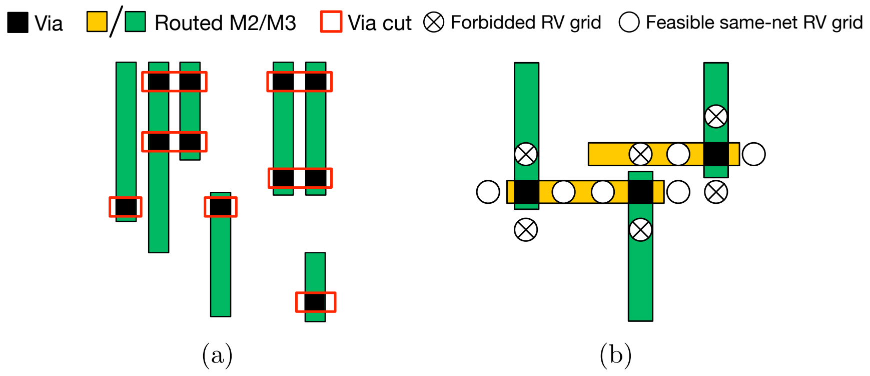


Figure 4.3: (a) Via cuts for SAVs, (b) SAV design contraint.

patterns by manufacturers to control the variation due to lithography, via density constraints are also introduced in order to control the variation from other processing steps like chemical-mechanical-polishing (CMP) and etch. These processing steps will have varying outcomes across a design, depending on the local layout densities [47]. If the number of RVs inserted within a density window is not well controlled, the violation of via density constraints could be detrimental to the yield of the design [59]. The windows for via density control are illustrated in Fig. 4.4. Specifically, a via layer is partitioned into a set of square regions. Each of the squares has a width of λ and the total number of vias, including SVs and RVs, within each window cannot exceed the pre-set upper bound. A possible scenario of the inserted RLLs is shown in detail for neighboring windows, i.e., W_{n+2} and W_{n+3}, in Fig. 4.4. Within the local search space for an SV's RLLCs, an RLL may cross multiple density windows, such as W_{n+2} and W_{n+3}. This means that the optimal RLLI scheme should simultaneously consider all density windows to globally balance the via densities and maximize the number of RLLs inserted.

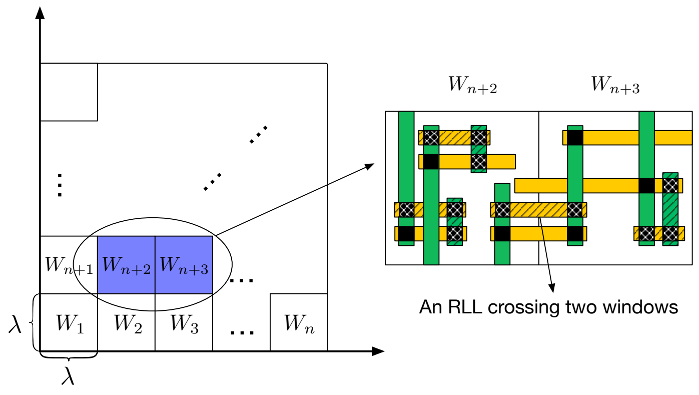


Figure 4.4: Via density windows.

Metal-Pattern Constraints Unidirectional metal patterns are manufacturing friendly in the sub-lithographic domain, which enables track-based coloring schemes and self-aligned techniques for multiple patterning lithography. This study addresses basic metal-pattern constraints including minimum metal tip-to-tip rules (1 metal grid) for line-end control and minimum metal length rules (2 metal grids) to avoid short metal defects. In advanced technology nodes, complex metal pattern rules are technology dependent. For example, self-aligned multiple patterning introduces complex line-end rules, which can be explicitly considered by avoiding prohibited line-end positions during the RLLC generation [67,105]. Our study primarily focuses on the global optimization of the RLLI and provides a design-rule-checker interface, which enables the incorporation of technology-specific metal and via pattern constraints (see Section 4.3.2).

4.2.3 Problem Definition

Given a design with unidirectional routing for each net on a routing grid, this study focuses on the RLLI with bounded timing impact for routed single vias at the post-routing stage. Apart from the advanced manufacturing constraints aforementioned, we need to explicitly consider the following constraints to obtain legal RLLIs. First, at most one RLL should be inserted for one single via. Second, two RLLs inserted cannot occupy the same grid unless they belong to the same net. Third, the RLLI scheme should minimize the total cost of RLLs inserted since the cost is defined to be inversely proportional to the timing and yield improvement of a specific RLLC. Thus, we define the RLLI problem as follows.

Problem 5 (Redundant Local-Loop Insertion)**.** *Given a unidirectional routing design and a set of density windows, the RLLI problem is to insert RLLs to cover as many routed single vias as possible while reducing the total cost of the inserted RLLs and accommodating advanced manufacturing constraints and bounded timing impact.*

4.3 Redundant Local-Loop Insertion

4.3.1 Timing Impact Analysis

As shown in Fig. 4.2, an RLLC consists of redundant metal wires and vias, which generates a loop structure in the routing solution. This makes timing analysis more complicated than that of the original routing tree. With the simplified RC network, our Elmore delay model provides a closed-form solution of the impact on signal delay computation in non-tree RC networks [63], which extends the empirical timing analysis in [44]. Fig. 4.5(a) shows the simplified RC network for an RLLC, where notations are given in Table 4.1. The primary path (from routing tree) and secondary path (from RLLC) from driver to load are denoted by black and red lines, respectively. This RC network can model any RLL defined in Definition 7 if the related resistance and capacitance are provided. The Elmore delay computation in non-tree topology [63] is based on the split RC network shown in Fig. 4.5(b). The key idea is that node N can be split into three independent nodes, i.e., N_1, N_2 and N_3, and the delay of the three nodes are the same. Further delay analysis and computation yield the following closed-form solution of the delay impact from driver to load due to RLLI[1].

$$\Delta delay = C_s \cdot R_d + \frac{C_s \cdot R_p \cdot R_s - C_p \cdot R_p^2 - 2 \cdot C_l \cdot R_p^2}{2 \cdot (R_p + R_s)}$$

The close-form solution of $\Delta delay$ reveals that the delay impact from RLLI could be quite different depending on specific resistance and capacitance parameters. In general, timing degradations could be arbitrarily small (even negligible) if the resistance (R_s) and capacitance (C_s) on the secondary path are much smaller compared to those (R_p and C_p) on the primary path. If secondary-path resistance and capacitance are relatively large, associated RLLCs should not be inserted due to prohibitively large timing degradations.

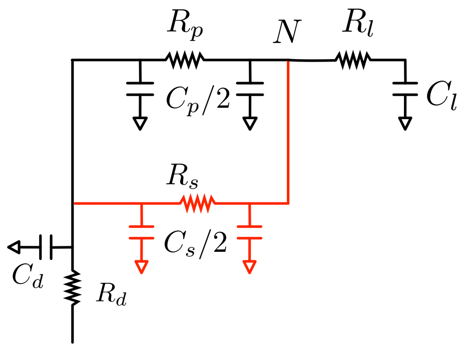
The closed-form solution (Elmore delay) is computed using simplified

[1]The detailed principles of computation for $\Delta delay$ are given in [63].

RC networks for the primary and secondary paths, where via and metal resistance/capacitance on the primary/secondary path are combined for one pi model as shown in Fig. 4.5(a). For accurate timing evaluations, Elmore delay is conservative compared to SPICE simulations. Moreover, the via/metal resistance and capacitance shall be modeled separately for better accuracy, which generates a much more complex RC network. Therefore, comprehensive SPICE simulations shall be performed with complex RC networks aforementioned to determine the actual timing impact. As will be discussed in Section 4.4.1, the RLLI could introduce a wide range of timing impact, i.e., delay increase or decrease, depending on specific RLL configurations. To enable practical adoption of RLLI, we bound the resulting timing impact due to RLLI by forbidding RLLCs with timing degradations greater than a preset timing impact amount. Therefore, the timing impact analysis and simulation yield a look-up table (LUT) of forbidden RLLC configurations under the preset timing impact bound, which will be used for RLLC generation and pruning. It shall be noted that our RLLI scheme is independent of the timing engine as long as an LUT of forbidden RLLC configurations is given.

Table 4.1: Notations for RC networks

C_d, C_l	the driving and loading capacitance
R_d, R_l	the driving and loading resistance
C_p, C_s	the capacitance on primary and secondary path
R_p, R_s	the resistance on primary and secondary path
C_{N1}, C_{N2}, C_{N3}	the effective capacitance on floating nodes



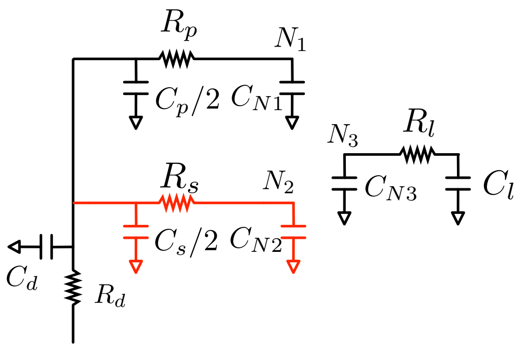


Figure 4.5: Elmore delay model for one RLLC, (a) simplified RC network, (c) split RC network.

4.3.2 RLLC Generation and Pruning

We first discuss the RLLC generation for each single via (SV). As mentioned in Section 4.2, rm_{x+1} and rm_x are bounded by a preset parameter for limiting the search space of RLLCs for a single via. The region for valid RLLCs of a single via is subjected to a distance constraint, i.e., the horizontal and vertical distance from the single via to the farthest corner of the RLLC is bounded by a preset parameter. Thus, all valid RLLCs for a single via are bounded by a rectangular region. Fig. 4.6 illustrates an example of RLLC generation for a single via in a net with rm_3 and rm_2 bounded by 5. For sv_1, the region for valid RLLCs is shown with the rectangle with dashed lines. During the RLLC generation for sv_1, we scan the rectangular region and skip forbidden redundant-via (RV) grids and metal grids occupied by a different net for valid RLLCs of sv_1. For example, due to the SAV constraints, forbidden RV grids are shown in Fig. 4.6 and RLLCs occupying those via grids are invalid.

One valid RLLC for sv_1 is further shown in Fig. 4.6, which covers two single vias, i.e., sv_1 and sv_2. It shall be noted that, during the sequential RLLC generation, this RLLC will be generated twice for sv_1 and sv_2. In general, the sequential RLLC generation leads to duplicates, i.e., equivalent RLLCs, for single vias belonging to the same net. We define the equivalence of two RLLCs as follows.

Definition 8 (Equivalence of RLLCs)**.** *An RLLC is defined to be equivalent to another RLLC when they share the same metal and via grids associated with the same net.*

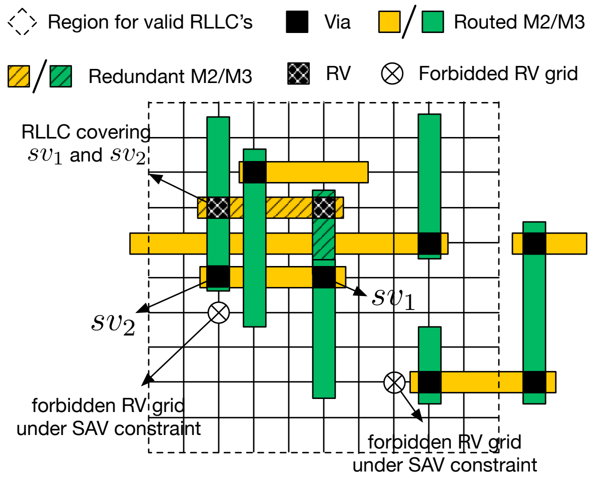

Region for valid RLLC's
Via
Routed M2/M3
Redundant M2/M3
RV
Forbidded RV grid
RLLC covering sv_1 and sv_2
sv_1
sv_2
forbidden RV grid under SAV constraint
forbidden RV grid under SAV constraint

Figure 4.6: RLLC generation for one single via.

We propose net-based RLLC generation and pruning technique to remove duplicates and achieve a compact set of RLLCs for each single via. The details of the RLLC generation and pruning scheme are shown in Algorithm 6. The RLLCs are generated in a net-by-net manner, which is explained from line 3 to line 18 of Algorithm 6. For each net, we define $RLLC_{net}$ in line 4 to store a compact set of RLLCs belonging to that net. Then, the single vias within the net are traversed from line 5 to line 9. We enumerate the RLLCs for the single via v as the set $\{rllc\}$ in line 6. In line 7, we further remove illegal RLLC configurations based on the timing analysis in Section 4.3.1 and the design rule checker (DRC). Notable design rules included here are the minimum metal tip-to-tip rule (1 metal grid), minimum metal-length rule (2 metal grids) and self-aligned via design rules in Section 4.2.2. In advanced technology nodes, complex metal and via pattern rules are technology dependent, which can be easily incorporated into our optimization engine by inventing technology-specific DRC. The enumerated set $\{rllc\}$ will be further combined into $RLLC_{net}$ in line 8. In particular, since $RLLC_{net}$ is a set data structure, it will automatically remove duplicates according to the definition of the RLLC equivalence in Definition 8. In lines 10 to 13, we index the RLLCs generated for current net based on the size of compact RLLC vector $RLLC$, which are further merged into $RLLC$. We need a compact vector of RLLCs because only one binary variable is generated for an RLLC covering multiple single vias in mathematical formulations. From line 14 to line 17, we obtain a vector to index the RLLCs for each single via in the net. The RLLC vector for each single via ($RLLC_{v_i}$) will only be selected from the compact set of RLLCs, i.e., $RLLC_{net}$, computed for the net. This guarantees no duplicate among the RLLCs generated for each single via.

We analyze the complexity of Algorithm 6. Related notations are de-

Algorithm 6 RLLC Generation and Duplicate Removal

```
Input: Routed patterns for nets ($\{nets\}$), a lookup table ($LUT$) of forbidden RLLCs from
    bounding timing analysis and a design rule checker ($DRC$);
Output: A set of RLLC vectors indexed by single vias ($\{RLLC_{v_i}\}$) and a compact vector
    of RLLCs ($RLLC$);
 1: Define $RLLC$ as a compact vector of RLLCs for all SVs;
 2: Define $\{RLLC_{v_i}\}$ as the set of RLLC vectors indexed by single vias;
 3: for each $net$ in $\{nets\}$ do;
 4:     Define $RLLC_{net}$ as the set of RLLCs for $net$;
 5:     for each single via $v_i$ in $net$ do;
 6:         Enumerate RLLCs for $v_i$ as set $\{rllc\}$;
 7:         Remove illegal RLLCs in $\{rllc\}$ with bounded timing ($LUT$) and design rule
    checker ($DRC$);
 8:         $RLLC_{net} = RLLC_{net} \cup \{rllc\}$;
 9:     end for
10:     for each $llc$ in $RLLC_{net}$ do;
11:         Index $llc$ with the size of $RLLC$;
12:         Push $llc$ to $RLLC$;
13:     end for
14:     for each single via $v_i$ in $net$ do;
15:         Define $RLLC_{v_i}$ as the RLLC vector for $v_i$;
16:         Select $RLLC_{v_i}$ from $RLLC_{net}$;
17:     end for
18: end for
19: for $RLLC_{v_i}$ indexed by SV $v_i$ do;
20:     Prune redundant candidates in $RLLC_{v_i}$;
21: end for
22: Return $\{RLLC_{v_i}\}$ and $RLLC$;
```

fined in Table 4.2. The net-based RLLC generation and pruning form a loop over the total number of nets (N) in a design, which means the complexity is linear to N. Within each loop, all RLLCs enumerated are stored in a set ($RLLC_{net}$). The maximum size of $RLLC_{net}$ is bounded by $O(M \cdot R)$, i.e., the maximum number of single vias per net times the maximum number of RLLCs per single via. Thus, the set ($RLLC_{net}$) construction time (from line 4 to line 9 in Algorithm 6) is $O(M \cdot R \cdot log(M \cdot R))$. In lines 10 to 13 and lines 14 to 17, merging $RLLC_{net}$ to $RLLC$ and RLLC selection for each single via both take $O(M \cdot R)$ time. The enumeration loop in lines 3 to 18 takes $O(N \cdot M \cdot R \cdot log(M \cdot R))$ time. In line 20, we sort the RLLCs for each SV based on cost and a linear-time scan can prune out redundant candidates, which altogether takes $O(R \cdot log(R))$. The total number of single vias is bounded by $O(N \cdot M)$. The pruning technique in lines 19 to 21 takes $O(N \cdot M \cdot R \cdot \log(R))$ time. Therefore, the time complexity of Algorithm 6 is $O(N \cdot M \cdot R \cdot log(M \cdot R))$. We further assume each net has approximately the same number of single vias and each single via has approximately the same number of RLLCs, i.e., $M \simeq |SV|/N$ and $R \simeq L/|SV|$. Then, the time complexity of Algorithm 6 reduces to $O(N \cdot |SV|/N \cdot L/|SV|) \cdot log(|SV|/N \cdot L/|SV|)$, i.e., $O(L \cdot log(L/N))$.

The last step in Algorithm 6 from line 19 to line 21 is to further prune RLLCs without loss of optimality. For $RLLC_{v_i}$ associated with a single via v_i, if there exists an isolated RLLC with a cost of c_i, we can potentially prune some RLLCs in $RLLC_{v_i}$. Those RLLCs shall have a larger cost than c_i and occupy the same via density windows as the isolated RLLC with a cost of c_i. An isolated RLLC means no other RLLCs conflict with it on the routing grid except for those RLLCs belonging to the same single via. Suppose any pruned RLLC is eventually selected in an optimal RLLI solution, we can always replace it with the isolated one for a lower cost and meanwhile the window density distributions remain the same. This contradicts with the optimality of the given RLLI solution. Thus, the pruning technique guarantees optimality. As discussed in the following section, each RLLC has an associated binary variable. Then, the RLLC pruning techniques in Algorithm 6 help to reduce the number of variables and constraints for our ILP formulation[2].

Table 4.2: Notations for RLLI

N	the number of nets
M	the maximum number of SVs for one net
R	the maximum number of RLLCs for each SV
L	the total number of RLLCs
SV	the pre-defined set of SVs
sv_j	j^{th} SV
llc_i	i^{th} RLLC
c_i	the cost for llc_i
n_i	the number of single vias that llc_i covers
x_i	the binary variable for llc_i
X_j	the variable set for the RLLCs covering sv_j
G_k	the k^{th} variable set for RLLCs occupying the same grid
SA_k	the k^{th} variable set for RLLCs occupying conflicting SAV grids
X, G, SA	sets of variable sets for $\{X_i\}$, $\{G_k\}$ and $\{SA_k\}$, respectively
W_k	the k^{th} density window
v_{ik}	the number of RVs of llc_i in W_k
DB_k	via density upper bound for W_k
W	the set of density windows $\{W_k\}$

4.3.3 Integer Linear Programming Formulation

With the RLLCs generated for each single via, Problem 5 becomes a general assignment problem to cover as many single vias as possible while reducing the total cost of inserted RLLs. Related notations are shown in Table 4.2. We assume each RLLC has binary variable x_i that denotes whether the i^{th} RLLC llc_i is selected to cover associated single vias. In particular, only one

[2]This pruning technique could be restrictive under long-range conflict constraints and via density constraints, which may limit the effectiveness for practical test cases.

binary variable is generated for an RLLC covering more than one single via. Then, Problem 5 can be formulated as a binary ILP problem. The objective of our ILP formulation consists of the weighted summation of the associated RLLC binary variable x_i, which should be maximized to improve the insertion rate and reduce the cost of inserted RLLCs. The objective (1) consists of two terms. The term $(CB \cdot \sum_{x_i} n_i \cdot x_i)$ is the first objective, which improves the insertion rate. The parameter n_i is added before x_i to consider that one RLLC may cover multiple single vias. The selection of that particular RLLC can cover n_i single vias. The term $(-\sum_{x_i} c_i \cdot x_i)$ is the second objective, which reduces the overall cost of inserted RLLs. The two terms are balanced using the parameter CB, where $CB > \max_i c_i$. Therefore, our formulation can simultaneously improve the insertion rate and reduce the overall cost of inserted RLLs.

$$
\begin{aligned}
\max \quad & CB \cdot \sum_{x_i} n_i \cdot x_i - \sum_{x_i} c_i \cdot x_i && && (1)\\
\text{s.t.} \quad & \sum_{x_i \in X_j} x_i \le 1 && \forall X_j \in X && (C1)\\
& \sum_{x_i \in A} x_i \le 1 && \forall A \in G \cup SA && (C2)\\
& \sum_{llc_i \in W_k} v_{ik} \cdot x_i \le DB_k && \forall W_k \in W && (C3)\\
& x_i \in \{0, 1\} && \forall x_i \in X_j \in X && (C4)
\end{aligned}
$$

To ensure legal assignment results for the RLLI problem, we consider three sets of constraints listed as follows.

- At most one RLLC is assigned to each single via. Thus, the summation of all $x_i \in X_j$ is bounded by one to guarantee that at most one binary variable will be 1 as listed in constraint $(C1)$.

- Conflict constraints are primarily related to two reasons. First, one single grid can only be occupied by one RLLC. Second, conflicting via grids cannot be occupied simultaneously under SAV constraints. Related constraints are represented by constraint $(C2)$[3].

- Via density constraints are applied to each density window. Since one RLLC may have multiple vias and cross multiple density windows, the value of coefficient v_{ik} denotes the number of RVs from llc_i in window W_k. The total number of vias in a window should not exceed an upper bound DB_k and the related constraints are listed in constraint $(C3)$.

The RLLI works on the given routing grid model. The metal and via grids store references to the RLLCs occupying those grids during the RLLC generation and pruning. Then, a simple grid traversal can set up all the constraints aforementioned. The objective of the ILP formulation combines two objectives for simultaneous insertion rate and RLL cost optimization. Moreover, the ILP formulation can globally balance the via densities for constraints $(C3)$, which generates much better solution qualities than a simple greedy scheme.

[3]Similar conflicting constraints can be added once technology-dependent design rules are given for metal and via patterns.

4.3.4 Special Constraint Structures

Although optimal solutions can be obtained with the ILP formulation for Problem 5, the exponential time complexity of the ILP formulation makes it unscalable to large problem sizes. It shall be noted that a typical ILP solving approach for Problem 5 involves two steps [19,50]. The first step is to relax the binary constraints in constraint $(C4)$ into the linear constraints $(0 \leq x_i \leq 1)$ and solve the LP-relaxed problem instance. The second step is to retrieve the binary solution via branch and bound scheme with the bounding information provided by LP-relaxed solution, which can be prohibitively time-consuming for a complex ILP problem.

Fortunately, the special constraint structures in Problem 5 lead to an LP-relaxation solution that is intrinsically close to a binary solution. Specifically, the selection constraints in $(C1)$ and conflicting constraints in $(C2)$ are strong valid inequalities [50] and can reduce the feasible region (polytope) of the LP-relaxed problem instance, which makes the LP-relaxed solution close to a binary solution. Strong invalid inequality means that each constraint in $(C1)$ and $(C2)$ is tight. Supposing k RLLCs conflict with each other, only one variable within $\{x_i, \forall i \in [1, k]\}$ can be assigned as 1, which can be formulated as two kinds of linear constraints, including loose inequality in $(C5)$ and strong invalid inequality in $(C6)$.

The selection constraints in $(C1)$ and conflict constraints in $(C2)$ are generated for each single via and each routing grid, which makes them strong in nature. For the geometric solution space of an LP-relaxed instance, the

$$x_i + x_j \leq 1 \qquad \forall i,\ j \in [1, k] \qquad (C5)$$

$$\sum_{i=1}^{k} x_i \leq 1 \qquad (C6)$$

strong valid inequalities, such as ($C6$), play as cutting planes to bound the LP solution to close to integral [50]. The via density constraints in ($C3$) turn out to be the same as capacity constraints for the ILP formulation of a constrained multiple knapsack problem [19]. It has been shown that LP-based relaxation effectively delivers integral solutions to the ILP formulation of the constrained multiple knapsack problem [19]. Therefore, due to special constraint structures in Problem 5, LP relaxation provides valuable opportunities to obtain close-to-integral solution.

Non-integral solutions generally exist from LP-relaxed instance even with strong valid inequalities, i.e., ($C1$) and ($C2$), and capacity constraints, i.e., ($C3$), for a constrained multiple knapsack problem. Thus, randomized rounding schemes are adopted in [19,50], where final solutions can be bounded to optimal solutions with a certain probability. However, randomized rounding scheme generates uncertainty in the solution, which is not preferred for the RLLI problem. Instead, we propose a deterministic greedy scheme after solving LP-relaxed instances, which empirically delivers close-to-optimal solutions.

4.3.5 IRLS with Incremental Search

To take advantage of the special constraint structures aforementioned, we propose an IRLS with incremental search scheme to iteratively solve the LP-relaxed problem instance followed by an incremental search step for a scalable solution. Within each iteration, we determine the integral assignment, ignore non-integer results from the optimal LP solution and update the problem instance itself. The iteration stops when no integral assignment can be achieved with the LP relaxation for the problem instance. For those single vias without RLLCs assigned from the iterative LP solving procedure, an incremental search step is adopted, i.e., greedy rounding for IRLS, to improve the quality of results.

The details of the IRLS scheme for Problem 5 are shown in Algorithm 7. The main loop for IRLS is from line 2 to line 24. Within each iteration, the LP relaxation of the RLLI problem is solved and optimal solutions are stored in X_{opt} at line 3. From line 6 to line 8, we obtain the integral assignment A_1 from X_{opt}, where only variables with solutions as 1 are collected into A_1. From line 12 to line 18, we iterate through the conflict constraints for constraints ($C1$) and ($C2$). We define the variable set in constraint C as X_c in line 13. In line 15, if one of the variables in X_c has been assigned as 1 in A_1, we can deduce that all other binary variables should be assigned 0 (add to A_0) since each conflict constraint requires that only one binary variable can be assigned to 1. In line 16, we update the problem instance $RLLI$ by removing related conflict constraints and variables because associated binary variables

Algorithm 7 IRLS with Incremental Search

```
Input: An instance of the RLLI problem ($RLLI$) and the single via set ($SV$);
Output: The assignment ($F$) of RLL for each SV;
 1: Define $F$ as the assignment of RLL for each single via;
 2: while True do;
 3:     Find the optimal solution $X_{opt}$ with $LP(RLLI)$;
 4:     Define $A_0 = \emptyset$ as the integral assignment with 0's;
 5:     Define $A_1 = \emptyset$ as the integral assignment with 1's;
 6:     for each $x_i = 1$ in $X_{opt}$ do;
 7:         Add $x_i = 1$ to $A_1$;
 8:     end for
 9:     if $A_1 = \emptyset$ then
10:         Break;
11:     end if
12:     for each conflict constraint $C$ in $(C1)$ and $(C2)$ do;
13:         Define $X_C$ as the variable set in $C$;
14:         if $X_C \cap A_1 \neq \emptyset$ then;
15:             Add $X_C - A_1$ to $A_0$ with assignment 0;
16:             Remove $C$ and $X_C$ from $RLLI$;
17:         end if
18:     end for
19:     for each window density constraint $C$ in $(C3)$ do;
20:         Update $DB_k$ with assignment $A_1 \cup A_0$;
21:     end for
22:     Update the objective (1) with assignment $A_1 \cup A_0$;
23:     $F = F \cup A_1 \cup A_0$;
24: end while
25: $F \leftarrow$ GreedySearch($RLLI, SV, F$);
26: Return $F$;
```

have become constants with assigned 0/1 values. From line 19 to line 21, the density upper bound DB_k is updated for each density window considering the integral assignment $A_1 \cup A_0$. Each RLLC with a binary variable as 1 in integral assignment A_1 consumes the density budgets in related density windows. This means the density upper bound of the related density windows shall be updated for the next iteration of integral assignment. In line 22, the objective is further updated with the integral assignment $A_1 \cup A_0$ since variables in $A_1 \cup A_0$ become constants for the next iteration. With the update on the problem instance $RLLI$, the next iteration of LP relaxation performs another round of integral assignment on top of the previous assignment results. We stop the iteration when no integral assignment can be obtained from the X_{opt} as shown in line 9 to line 11.

Algorithm 8 RLLI with Greedy Search

```
 1: function GreedySearch($RLLI, SV, F$)
 2:     for each single via $v_i$ in $SV$ do;
 3:         if no RLLC assignment to $v_i$ in $F$ then;
 4:             Obtain the set of RLLCs ($RLLC_{v_i}$) for $v_i$;
 5:             Sort $RLLC_{v_i}$ in the increasing order of cost;
 6:             Traverse $RLLC_{v_i}$ for the first legal candidate add it to $F$;
 7:         end if
 8:     end for
 9:     Return $F$;
10: end function
```

An incremental greedy search step is explained Algorithm 8. During the incremental search, for each single via without RLL inserted (line 3), we traverse associated RLLCs in the increasing order of cost (line 5) until a legal RLLC can be inserted or no legal RLLC is obtained (line 6). This is equivalent to a greedy rounding step if no integral solution is achieved for a specific variable after IRLS procedure.

In particular, the problem scale, including the number of variables and constraints, of the later iteration will be much smaller than that of the former iteration and the LP problem instance will be updated based on fixed values (0/1). Then the solution space, i.e., polytope, of the LP instance for the later iteration will be different (smaller dimension) from the former iteration. One vertex of the polytope is one feasible solution of the LP instance. A typical LP solver adopts the simplex algorithm to seek an optimal solution by moving from one vertex to another vertex of the polytope. The optimal solution of the former iteration is a vertex of the polytope for the corresponding LP instance. It is within the solution space but not necessarily a vertex of the polytope for the later LP instance. By moving among vertices of the polytope for the updated LP instance, the IRLS scheme keeps updating the solution of the RLLI. In theory, it is possible that the first iteration of IRLS produces non-integral values for all variables. The solution for each iteration of IRLS also depends on specific ILP solver and underlying solving schemes. But in practice, the corner case aforementioned rarely happens in real experiments because LP relaxation provides a close-to-integral solution. Moreover, an incremental search step is performed after the IRLS solving. This means our IRLS with incremental search scheme can deliver insertion results at least as good as the greedy scheme.

For the IRLS with incremental search scheme, each iteration of LP solving simultaneously considers all conflict constraints and via density constraints. With an incremental search step, it gives much better solution qualities than the simple greedy scheme. Meanwhile, the LP relaxation leads to polynomial time complexity within each iteration, which makes the runtime of the IRLS with incremental search scheme much more scalable than the optimal ILP approach.

4.4 Experimental Results

We have implemented the RLLI algorithms in C++ and all experiments are performed on a Linux machine with a 2.9GHz Intel(R) Core and 192 GB memory. Gurobi [37] is adopted as our ILP/LP solver. For RLL cost, the parameters are set as $\alpha = \beta = 1.0$ and $\gamma = 5.0$. The cost upper bound for weight computation is set as $CB = 100.0$. For limiting the search space of RLLCs for an SV, the upper bound of rm_{x+1} and rm_x is set as 20. For windows of via density control, the width is set as $\lambda = 20$ routing grids and the via density upper bound within a window is set as 40 for results in Table 4.4 and 4.5. The benchmark statistics are listed in Table 4.3, where modules from OpenSparc T1 are synthesized with Design Compiler [96] and placed using Cadence SOC Encounter [16] with the standard cell utilization rate set as 0.7. The unidirectional routing results are generated using a state-of-the-art unidirectional router in Section 3.2 and mapped onto a routing grid model. The routing density of the original unidirectional routing results is relatively low. Extra blockages are created on the M2 and M3 layer to represent dense routing cases in advanced technology nodes. Specifically, we take the sparse cases, i.e., the original unidirectional routing results, and add metal blockages every 1 out of 3 tracks (everywhere the routing grid is not occupied), which attempts to increase the routing utilization by 33% over the sparse cases. Our experiments end up with sparse and dense routing cases in Table 4.4 and Table 4.5, respectively. Our framework is based on a grid structure and supports partitioning of large benchmarks to make the runtime and memory more practical, e.g, test case “sparc” is partitioned into 9 parts evenly in the physical dimension. We further run our algorithm on each part separately for affordable memory and runtime. Our RLLI optimization framework takes the unidirectional routing patterns on the routing grid model as the input for the post-routing RLLI.

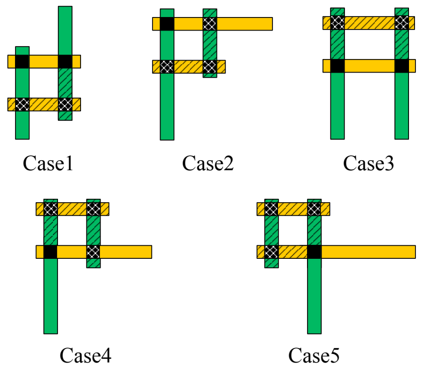
4.4.1 Bounded Timing Impact

Our timing simulations build on the 10*nm* predictive technology setup, where wire resistance and capacitance are set by the ITRS roadmap [3] and via resistance is provided by our industrial collaborator. The driver is set as the INV_X1 from the NanGate 15*nm* library [79] scaled to 10*nm* dimensions with PTM models [4]. We assume the M2 and M3 layers share the same resistance/capacitance and the input routing solution has no detour. Under bounded local search space, we only consider RLL configurations shown in Fig. 4.7(a), where each RLLC is in the rectangular shape and covers at most 2 single vias. A rectangular RLLC covers exactly 4 vias. If an RLLC covers more than 2 single vias, that means the input routing solution has detour because the router should have selected the path containing redundant vias (less than 2) for short wirelength and a smaller number of single vias. For SPICE simulations of fan-out-4 delay impact in Fig. 4.7(b), we adopt the pi model for each wire segment and via leading to more accurate RC network for delay simulations than that in Fig. 4.5. The primary path consists of M2 and M3 wire segment both in 4*um* length [44].

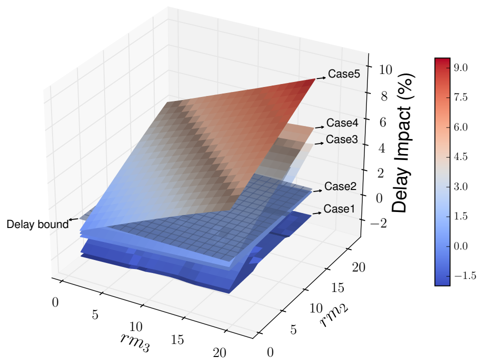
As shown in Fig. 4.7(b), a wide range of delay impact, from -1.5% to +9%, is introduced for different configurations of RLLCs and the delay bound is set as 1%. The general observation from timing analysis is that more redundant vias in an RLLC will introduce larger secondary-path resistance and capacitance, which further leads to larger timing impact. Specifically, for one RLL configuration such as Case5, the timing degradations increase as the number of redundant metal grids increases as shown in Fig. 4.7(b). Moreover, for the same number of rm_2 and rm_3, the secondary-path resistance and capacitance increase monotonically from Case1 to Case5, which makes timing degradations monotonically increase as well. For Case4 and Case5, the timing degradations can be prohibitively large (9%). The delay impact bound is set as 1% in our experiments during RLLC generation, which means we strictly forbid the RLLCs with more than 1% delay impact. Although our timing simulations are preliminary, more accurate timing analysis can be easily introduced to the RLLI framework as long as an LUT of forbidden RLLC configurations is provided.

Table 4.3: Benchmark statistics for RLLI

Ckt	ecc	efc	ctl	alu	div	top	sparc
SV#	4013	4619	5873	6683	12878	48847	360239
Grid Size	446x436	421x406	503x496	408x406	646x636	1179x1176	3969x3966
RLLC# per SV (sparse)	47.3	39.0	43.7	32.6	36.0	35.2	44.2
RLLC# per SV (dense)	11.2	9.6	11.0	8.2	8.9	8.6	10.7
Gen. time(s) (sparse)	4.5	4.9	6.5	6.3	13.0	49.4	443.2
Gen. time(s) (dense)	3.1	3.4	4.3	4.6	9.1	34.8	290.6



(a)



(b)

Figure 4.7: (a) A complete set of RLLC configurations assuming no detour in a routing solution, (b) timing impact evaluation.

4.4.2 Comparisons on Different RLLI Schemes

In Table 4.4 and 4.5, we compare four RLLI schemes, including the greedy scheme, the ILP scheme, the IRLS without (w/o) and with (w/) incremental search scheme, in terms of solution qualities for the RLLI problem. The greedy scheme is adapted from [8, 44] with explicit considerations of SAV and via density constraints. The greedy scheme is the same as the greedy search in Algorithm 8 without the initial assignment results from IRLS. In Table 4.4 and 4.5, the insertion rate, denoted as "I.R.", is defined as the number of SVs with an RLL inserted over the total number of SVs in the design. The "RLL#" is the total number of RLLs inserted. To quantify the redundant-via (RV) usage for each RLL inserted, we average the number RVs from the inserted RLLs over the total number of inserted RLLs, i.e., RV number per RLL, denoted as "R.p.R". "T" denotes the runtime. In general, better RLLI schemes should lead to larger "RLL#" and higher "I.R." for yield improvement, less "R.p.R" for less resistance and smaller timing impact as shown in Fig. 4.7. Although the greedy scheme runs very fast due to linear time complexity, it has several drawbacks in terms of solution qualities for both sparse and dense routing cases. First, the "I.R." is not consistent and highly depends on the test cases. For "ecc", the "I.R." is more than 98% (sparse) and 82% (dense) while for "alu", the "I.R." is less than 81% (sparse) and 61% (dense). Second, the "R.p.R" is higher than the IRLS w/ incremental search scheme by 13.1% (sparse) and 9.0% (dense) on average. This means the IRLS w/ incremental search can effectively select those RLLCs covering multiple SVs, which in general induces less timing impacts as shown in Fig. 4.7. Third, for the benchmark "top": the greedy scheme has around 10% less "I.R." and $\geq$ 10% higher "R.p.R" compared to the IRLS w/ incremental search scheme for both sparse and dense routing cases.

The incremental search step is critical to the quality of Algorithm 7. In Table 4.4 and 4.5, more than 16% "I.R" can be obtained with the incremental search on top of IRLS results with negligible runtime impact. This is because non-integral solutions may still exist from LP-relaxed instances, which means RLLCs are not assigned to some SVs after IRLS procedure. As RLLC generation and most of the assignment task has been performed after IRLS procedure, the incremental search takes linear time complexity, which is negligible compared to the runtime of RLLC generation and IRLS.

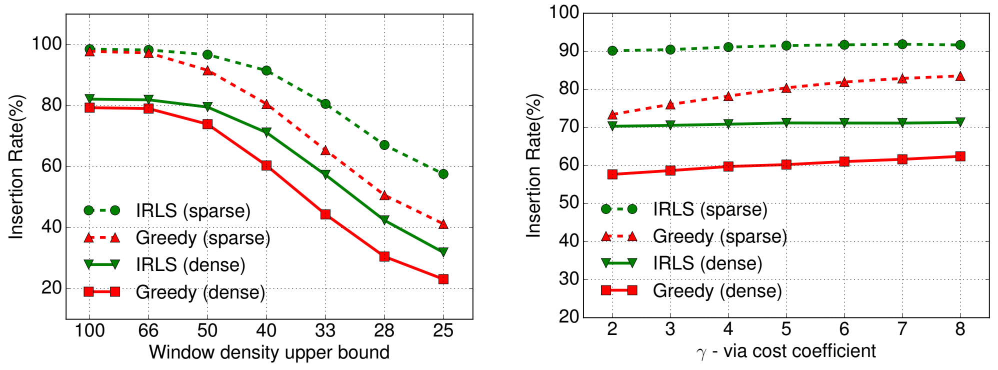
Table 4.4: Comparisons on sparse routing cases

Sparse	Greedy				0-1 ILP				IRLS w/o incremental search				IRLS w/ incremental search			
Ckt	I.R.(%)	RLL#	R.p.R	T(s)	I.R.(%)	RLL#	R.p.R	T(s)	I.R.(%)	RLL#	R.p.R	T(s)	I.R.(%)	RLL#	R.p.R	T(s)
ecc	98.26	2542	2.45	4.6	99.85	2119	2.11	207.5	88.21	1874	2.11	53.5	99.75	2136	2.13	53.5
efc	92.35	2799	2.45	4.9	99.06	2465	2.11	500.9	82.37	2056	2.12	61.3	98.00	2476	2.14	61.3
ctl	95.23	3543	2.42	6.5	99.66	3055	2.08	355.7	84.62	2598	2.09	81.1	99.05	3059	2.10	81.1
alu	80.40	3232	2.34	6.4	N/A	N/A	N/A	$> 10^5$	66.68	2271	2.04	82.5	91.50	3221	2.10	82.5
div	88.12	7103	2.40	13.0	98.11	6612	2.09	9285.4	78.42	5258	2.08	166.1	96.12	6579	2.12	166.2
top	83.00	24705	2.36	49.8	N/A	N/A	N/A	$> 10^5$	72.53	17902	2.02	844.2	93.08	23680	2.08	844.4
sparc	94.64	216143	2.42	445.4	N/A	N/A	N/A	$> 10^5$	82.74	155836	2.09	7218.7	98.17	188394	2.12	7219.7
Avg.	0.935	1.133	1.131	0.062					0.822	0.818	0.977	1.000	1.000	1.000	1.000	1.000

Table 4.5: Comparisons on dense routing cases

Dense	Greedy				0-1 ILP				IRLS w/o incremental search				IRLS w/ incremental search			
Ckt	I.R.(%)	RLL#	R.p.R	T(s)	I.R.(%)	RLL#	R.p.R	T(s)	I.R.(%)	RLL#	R.p.R	T(s)	I.R.(%)	RLL#	R.p.R	T(s)
ecc	82.28	2226	2.52	3.2	85.47	2010	2.29	29.3	76.70	1810	2.30	11.1	85.12	2015	2.30	11.1
efc	73.91	2330	2.51	3.4	82.69	2272	2.29	135.0	67.34	1859	2.30	13.1	81.11	2258	2.31	13.1
ctl	79.70	3119	2.50	4.3	85.71	2921	2.28	174.5	72.94	2514	2.30	17.5	84.62	2915	2.30	17.5
alu	60.23	2527	2.41	4.7	74.52	2729	2.18	11469.6	53.05	1947	2.18	18.5	71.18	2674	2.22	18.5
div	69.27	5890	2.49	9.2	N/A	N/A	N/A	$> 10^5$	64.81	4795	2.26	35.3	78.31	5868	2.28	35.3
top	63.50	19800	2.43	35.1	N/A	N/A	N/A	$> 10^5$	59.85	15941	2.17	150.3	74.15	20257	2.21	150.4
sparc	77.34	187495	2.51	294.5	N/A	N/A	N/A	$> 10^5$	71.40	150731	2.29	1389.8	82.64	175980	2.31	1390.5
Avg.	0.907	1.054	1.090	0.217					0.837	0.847	0.992	1.000	1.000	1.000	1.000	1.000

The ILP scheme maximizes a weighted sum to simultaneously improve the insertion rate and reduce the overall insertion cost. The best solution qualities are reported for the first three and four test cases for sparse and dense routing, respectively. However, the exponential time complexity of the ILP scheme leads to unaffordable runtime for large test cases. In particular, for the sparse routing case, ILP cannot generate a solution for "alu" under 10^5 seconds although "alu" has similar problem scale as "ctl". The "I.R." for "alu" is 14.83% and 7.55% less than that of "ctl" for greedy and IRLS w/ incremental search scheme, respectively. This means that the constraints for "alu" are very hard to resolve than those for "ctl", which forbids ILP scheme reaching an optimal solution within 10^5 seconds. Actually, IRLS w/ incremental search scheme achieves competitive solution qualities as the "I.R." is less than the ILP scheme by only 1% for the first three test cases. We achieve significant speed-up from the IRLS w/ incremental search scheme as the runtime is much more scalable and affordable across various test cases compared to the ILP approach. In addition, compared with the greedy scheme, the IRLS w/ incremental search scheme improves the "I.R." by 17.8% and 16.3% with 18.2% and 15.3% increase in "RLL#" on average for sparse and dense routing, respectively. In practice, the problem scale of later iterations of the IRLS procedure reduces dramatically, which makes it more scalable than the ILP scheme among test cases with different sizes shown in Table 4.4 and 4.5. Therefore, we argue that the IRLS w/ incremental search is the best scheme among the four schemes aforementioned since it has much better solution qualities than the greedy scheme and significant speed-up compared to the ILP approach.



(a) (b)

Figure 4.8: Parameter analysis with benchmark "alu", (a) degradations of insertion rates as the window density upper bound decreases with γ set as 5, (b) change of insertion rate when varying γ with window density upper bound set as 40.

As mentioned in Section 4.2, via density constraints are critical for the RLLI problem because one RLL may cross multiple density windows. This not only introduces complicated conflicting constraints but also brings the opportunity to globally balance the via densities with efficient RLLI schemes. The strength of the IRLS w/ incremental search scheme is further demonstrated by the trade-off between the insertion rate and the window density upper bound in Fig. 4.8(a) for both the sparse and dense routing cases. The insertion rates are approximately the same for the greedy and IRLS w/ incremental search scheme when the upper bound is at 100 due to loose via density constraints. However, when the upper bound gradually decreases, the insertion rate rapidly decreases for the greedy scheme while the decrease for the IRLS scheme is much slower. Since the IRLS w/ incremental search scheme globally considers the via density constraints in each iteration, we observe better insertion rates when the upper bound is pushed to its lower limit.

For RLL cost, we assume metal patterns with the same length on M2 and M3 share the same resistance and capacitance. Thus, α and β are normalized to 1.0. In $10nm$ technology setup, the single via resistance will be larger than a piece of metal with minimum length (2 metal pitches in our experiments). Thus, we set via cost coefficient (γ) larger than α and β. Fig. 4.8(b) further demonstrates the impact on insertion rate from different via cost coefficient. In general, a larger via cost coefficient leads to lower cost for RLLCs with less number of redundant vias, i.e., RLLCs covering more than one single via. As shown in Fig. 4.4, these RLLCs usually cross more than one density window, which globally balances via densities and generates higher insertion rate. When γ increases, the insertion rate from the IRLS w/ incremental search scheme increases more slowly than that for the greedy scheme. The IRLS w/ incremental search scheme is still more favorable than the greedy scheme due to consistently larger insertion rates. Moreover, for the IRLS w/ incremental scheme, the insertion rate saturates as γ increases beyond 5 as shown in Fig. 4.8(b). We empirically set γ as 5, which approximately denotes the starting point of saturation for the IRLS w/ incremental scheme in our experiment setup.

4.4.3 Comparisons with Redundant Via Insertion

To elaborate the potential usages of RLLI in advanced technology nodes, we compare traditional double-via insertion (DVI) and RLLI in terms of timing impact, random failure rate [44], routing resource usage and problem complexity. We assume off-track metal coverage for DVI, the width and length of which are set as 1 and 2 metal grids, respectively.

Timing Impact We have performed the timing impact analysis and simulation for RLLI in Section 4.3.1 and Section 4.4.1. We evaluate the fan-out-4 delay with DVI using the same technology setup in Section 4.4.1. Fig. 4.9 compares the timing impact between DVI and various cases of RLLCs. DVI for one single via improves the fan-out-4 delay by 0.3% under our technology setup. In general, RLLI generates more redundant resistance and capacitance to form a local-loop structure comparing to DVI. Thus, RLLI introduces more timing impact than DVI except Case 1. This becomes the major motivation to control the timing degradations (within 1% in this study) during RLLI.

Random Failure Rate We compare the random failure rate between DVI and RLLI to elaborate practical usages of RLLI. To simplify the analysis, we consider via failure rate while assuming metal patterns are free of opens or shorts [44]. Fig. 4.10(a) illustrates different cases of DVI and each case corresponds to a case of RLLC in Fig. 4.7(a). We assume the random failure probability of a single via is p and via failures are independent of each other.

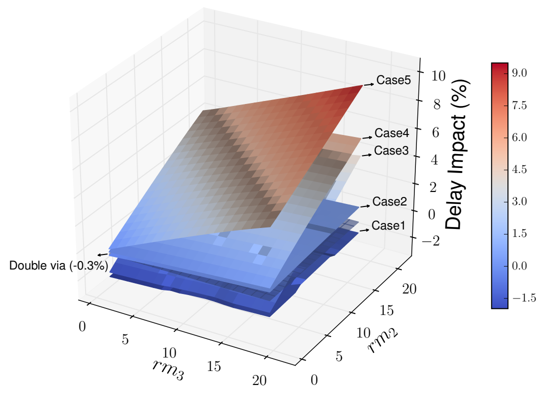
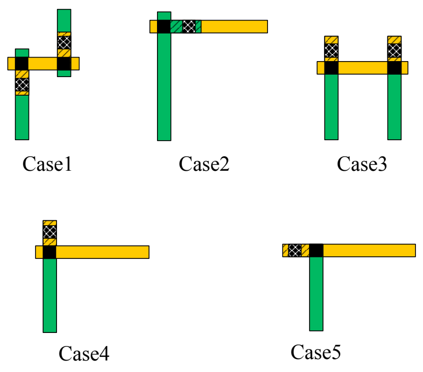


Figure 4.9: RLLI vs DVI in terms of timing.

With probability calculation, we obtain the failure probability of each double-via and redundant local-loop insertion case as shown in Table 4.6 [44]. If we vary p in the unit of "failure per billion (fpb)" [44], we can compute the failure rate ratio between RLLI and DVI for different cases as shown in Fig. 4.10(b). In general, the failure rate of a redundant local loop structure is 1 or 2 times larger than that of a double-via structure. Although RLL is not as robust as double via, RLL still provides better robustness than single via. Typically p is a very small value, which makes the values of $(1-(1-p)^2)^2$ (for Case 1/3) and $p \cdot (1-(1-p)^3)$ (for Case 2/4/5) much smaller than p itself. For instance, suppose $p = 1$fpb, then $(1-(1-p)^2)^2 \simeq 4 \cdot 10^{-9}$fpb and $p \cdot (1-(1-p)^3) \simeq 3 \cdot 10^{-9}$fpb. Thus, RLLI is a promising candidate for yield enhancement of unidirectional routing.



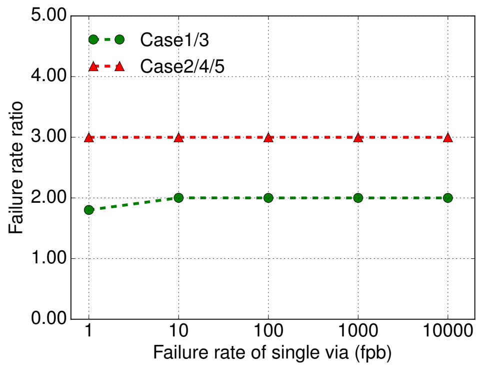


Figure 4.10: (a) DVI in different cases and each case corresponds to a case of RLLC in Fig. 4.7(a), (b) failure rate ratio, i.e., RLLI over DVI across various via failure rate.

Table 4.6: Comparisons between DVI and RLLI

	Failure rate		Routing resource	
	DVI	RLLI	DVI	RLLI
Case 1	$1-(1-p^2)^2$	$(1-(1-p)^2)^2$	6	$rm_2 + rm_3 + 2$
Case 2	p^2	$p \cdot (1-(1-p)^3)$	3	$rm_2 + rm_3 + 3$
Case 3	$1-(1-p^2)^2$	$(1-(1-p)^2)^2$	6	$rm_2 + 2 \cdot rm_3 + 2$
Case 4	p^2	$p \cdot (1-(1-p)^3)$	3	$rm_2 + 2 \cdot rm_3 + 3$
Case 5	p^2	$p \cdot (1-(1-p)^3)$	3	$2 \cdot rm_2 + 2 \cdot rm_3 + 3$

Routing Resource We further compare routing resource usages between DVI and RLLI. For a specific case of DVI or RLLI, we quantify the routing resource as the total number of redundant metal (rm_2 and rm_3) and via grids. As shown in Table 4.6, the routing resource usage of RLLI varies from case to case, and RLLC covering more than one single via consumes less amount of routing resource. For instance, Case 5 uses much more routing resource than Case 1 for RLLI. In general, RLLI consumes more routing resource than

DVI due to the local-loop structure. RLLI and RVI are typically performed at the post-routing stage, routing resources (i.e., empty grids on metal and via layers) are given as input for RLLI/RVI engine to maximize the insertion rate. For metal layers where unidirectional routing is strictly enforced, RLLI is still a feasible candidate to improve manufacturing yield. Given limited routing resources for RLLI, it is important to perform global optimization and maximize insertion rate with bounding timing impact.

Problem Complexity The RLLI problem is different from the conventional RVI problem although the ILP formulation looks similar. In Table 4.3, we calculate the average number of RLLCs for each SV, denoted as "R.p.R", to be 39 and 10 on average for sparse and dense routing cases, respectively, which are larger than the number of RV candidates available for each SV in the conventional RVI problem. Through our experiments, we find that the complexities of the RLLI problem make the fast techniques from the conventional RVI problem obsolete. Specifically, the pre-selection technique, i.e., selecting the conflict-free RLLC with minimum cost for an SV, breaks the optimality of the solution under tight via density constraints as an RLLC may cross multiple density windows as shown in Fig. 4.4. Independent component computation does not work well due to complicated conflicting constraints across several horizontal/vertical tracks. In our experiments, the largest independent component of each constructed conflict graph consists of around 99% and 85% of nodes within the entire graph for sparse and dense routing cases, respectively.

Independent component computation is not effective for the RLLI problem and can not be directly applied under via density constraints. Thus, it is not incorporated into our optimization engine.

4.5 Summary

This chapter proposes the first global optimization engine on RLLI considering advanced manufacturing constraints on via patterns and bounded timing impact from RLLI. The proposed RLLI framework is independent of timing simulation setup as long as an LUT of forbidden RLLC configurations is given by the timing engine under bounded timing impact. With the net-based RLLC generation and pruning, we further propose the binary ILP formulation and the IRLS with incremental search scheme to obtain scalable solutions with negligible performance degradations. Our experimental results demonstrate that the IRLS with incremental search scheme achieves more consistent solution qualities compared to the greedy scheme and more scalable runtime compared to the ILP approach.

Chapter 5

Sub-Resolution Assist Feature Generation

5.1 Introduction

As the technology node continues scaling down, the $193nm$ wavelength photolithography with low $k1$ value is the mainstream technique to achieve smaller feature size. However, low image contrast and complex target pattern shapes make it extremely difficult for low-$k1$ lithography to obtain acceptable lithographic process windows [86]. Besides the design for manufacturability techniques, like multiple patterning and litho-friendly layout design, mask optimization through resolution enhancement techniques (RETs) remains as the key strategy to improve the lithographic process window and the yield of the volume production in advanced technology nodes [10, 12, 30, 54, 95]. Major RETs include source mask co-optimization, sub-resolution assist feature (SRAF) generation and optical proximity correction (OPC). Among them, the SRAF generation is particularly important to improve the lithographic process window of target patterns. The key physical mechanism behind is that, without printing themselves, the small SRAF patterns would deliver light to the positions of target patterns at proper phase so that the printing of target patterns will be more robust to the lithographic variations. The lithographic process window is quantified with the process variation (PV) band area, which should be minimized to obtain a robust mask optimization solution. An example demonstrating the benefit of SRAF generation is shown in Fig. 5.1. An isolated target contact with the OPC pattern is shown in Fig. 5.1(a) and the target pattern is optimized only with OPC in Fig. 5.1(b), while the optimization in Fig. 5.1(c) is done with both SRAF generation and OPC. It can be clearly observed that much smaller PV band area is achieved in Fig. 5.1(c). Therefore, fast SRAF generation with high quality is of great importance for the mask optimization.

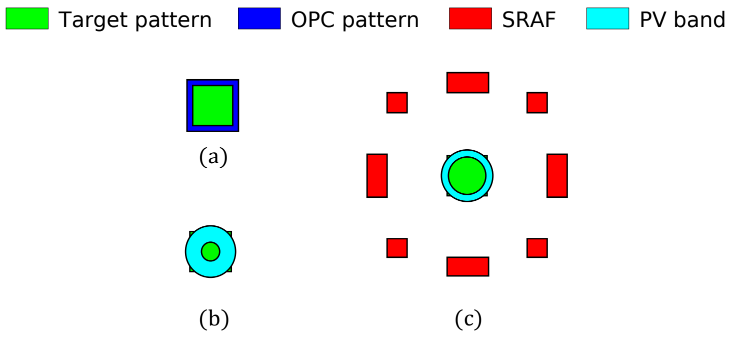


Figure 5.1: (a) An isolated contact, (b) printing with OPC, (c) printing with SRAF generation and OPC.

Multiple SRAF generation approaches, including model-based and rule-based approaches, have been developed and widely used in standard mask optimization flows. The rule-based approach is widely adopted due to its fast execution time and acceptable performance for simple designs and regular target patterns [46, 53, 86]. However, the rule-based SRAF is hard to deal with complex two-dimension (2D) shapes as it requires significant engineering efforts to setup and maintain the rule table [46]. Model-based SRAF generation methods can be divided into two categories based on the lithographic computations involved. One is to use simulated aerial images to seed the SRAF generation [90, 91, 101, 115]. The other is to apply inverse lithography technology (ILT) and compute the image contour to guide the SRAF generation [51, 85]. Despite better lithographic performance and generalization capabilities compared to the rule-based approach, the model-based SRAF is known to be very time-consuming and it is difficult to achieve the same SRAFs around the same layout configurations, i.e., consistent SRAFs [46, 86].

Recently, the machine learning technique has been introduced to the computational lithography domain, with applications to lithographic hotspot detection [22, 25, 73, 100, 121] and OPC [35, 69, 70, 74]. The machine learning technique calibrates a mathematical model with respect to an objective from the training data set based on accurate lithographic computations. Then, the calibrated model can predict the objective values, like a hotspot or non-hotspot for the hotspot detection and the shifting distance of an edge segment for the OPC, on the testing data. The machine learning technique usually demonstrates a trade-off between computational efforts and lithographic performance, which makes it particularly interesting for the SRAF generation problem. However, to the best of our knowledge, there is no prior art in applying the machine learning technique to the SRAF generation issue. In this study, we propose the first machine learning-based framework for the SRAF generation. Our methodology can achieve fast yet consistent SRAFs with high quality in a 2D grid plane. Our main contributions are summarized as follows:

- A machine learning-based framework is proposed for the SRAF generation, where a classification model is calibrated for SRAF predictions using model-based SRAFs as the training data.

- We propose a robust feature extraction scheme by adapting the concentric circle with area sampling considering SRAF-specific constraints. We further propose a novel feature compaction technique taking advantage of illumination source symmetry properties to reduce the training data size and improve the SRAF consistency.

- Logistic regression and decision tree models are calibrated for fast SRAF predictions due to the large data set size and high feature vector dimension. Different from conventional label predictions, we propose predictions with probability maxima in the 2D grid plane to generate manufacturing-friendly SRAFs.

- Our machine learning-based framework achieves 10X speed-up in layout windows with comparable lithographic performance, compared with an industry strength model-based approach.

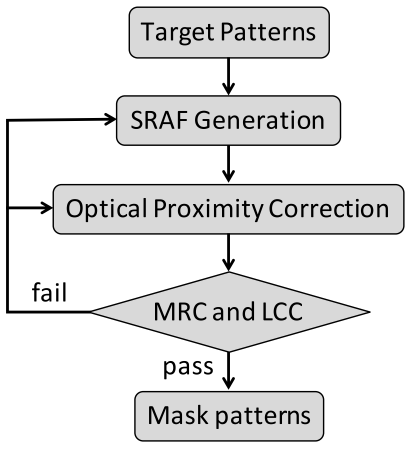
5.2 Preliminaries

5.2.1 Mask Optimization Flow

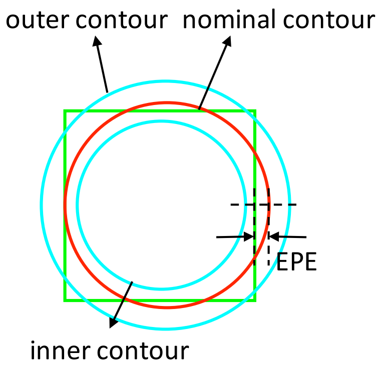
A standard mask optimization flow consists of several stages, including SRAF generation, OPC, mask manufacturing rule check (MRC) and lithography compliance check (LCC) as shown in Fig. 5.2(a) [86]. Depending on the outcome of MRC and LCC, iterative optimizations may be applied to achieve legal mask patterns. The MRC will check whether mask patterns satisfy a set of mask manufacturing rules. The LCC means lithography simulations are performed to check whether lithographic constraints are satisfied. In the stage of SRAF generation, small SRAFs will be added and isolated patterns on the mask will become dense patterns as shown in Fig. 5.1(b). SRAFs will not be printed themselves but will benefit the lithographic process windows of target patterns. In the next stage, OPC will shift the edges of OPC patterns to compensate for the optical proximity effects. Then, for the MRC, we assume the target patterns are MRC-clean and some typical mask manufacturing rules are applied to the SRAFs since this study mainly focuses on the SRAF generation. Typical mask manufacturing rules for SRAFs include maximum width (max_width) rule, minimum space (min_space) rule and maximum length (max_length) rule. The LCC will introduce a lithographic process window involving a set of {focus, dose} conditions [36]. Lithography simulations at various conditions are performed to check whether the metrics, such as PV band and edge placement error (EPE), meet the criteria.

5.2.2 Evaluation Metrics

We introduce several metrics to evaluate the performance of mask optimization results. An example of lithography simulation results is shown in Fig. 5.2(b). Inner and outer contours are explicitly drawn to demonstrate the lithographic printing variations due to the imposed {focus, dose} conditions.



(a)



(b)

Figure 5.2: Mask optimization: (a) mask optimization flow, (b) lithography simulation contours.

Nominal contour represents the lithographic printing at the best {focus, dose} condition. To quantify the lithographic variations, we define PV band and EPE as follows.

Definition 9 (PV Band)**.** *Given the lithography simulation contours at a set of {focus, dose} conditions, the process variation (PV) band is defined as the area between the outer contour and inner contour.*

Definition 10 (EPE)**.** *Given the lithography simulation contour at the best {focus, dose} condition, i.e., nominal contour and a measurement point, the edge placement error (EPE) is defined as the distance between the target pattern contour and nominal contour.*

Thus, in Fig. 5.2(b), the area between the outer contour and inner contour is the PV band. A measurement point is drawn with a dashed line orthogonal to the vertical edge of the target pattern in Fig. 5.2(b) and the EPE can be explicitly quantified. The SRAF consistency is an important issue since it is closely related to the process variations on wafer [86]. Consistent SRAFs are preferred around the same target pattern configurations because different SRAFs lead to different OPC results, which potentially introduce extra process variations. We define the consistent SRAF generation as follows.

Definition 11 (Consistent SRAF generation)**.** *Consistent SRAF generation means the same SRAF patterns should be generated for the same target layout configurations.*

5.3 Problem Formulation

The machine learning-based SRAF generation framework works on a 2D grid plane with a specific grid size. The training data consist of a set of layout clips, where each layout clip includes a set of target patterns and model-based SRAFs. With the 2D grid plane and the training patterns, training samples can be extracted at each grid point. To clearly explain the training data, we define the SRAF label as follows.

Definition 12 (SRAF label)**.** *Given model-based SRAFs on the 2D grid plane, the SRAF label of a grid is 1 or 0, where 1 denotes an SRAF is inserted at that grid and 0 vice versa.*

Specifically, a training data point includes a feature vector and an

SRAF label. The feature vector represents the optical conditions of the grid point with respect to the target patterns. With the training data, we define the classification-based SRAF as follows.

Problem 6 (Classification-based SRAF)**.** *Given the 2D grid plane and training patterns with model-based SRAFs, feature vectors and SRAF labels of all grid points are extracted and a classification model is calibrated to predict the SRAF insertion at each grid of testing patterns.*

In the testing phase, the classification model can predict the SRAF label at each grid for testing patterns. Those grids with SRAF labeled as 1 can not directly be treated as the final SRAFs and further simplifications are needed to generate SRAFs accommodating mask manufacturing rules. Thus, we define the SRAF generation as follows.

Problem 7 (SRAF Generation)**.** *Given the classification model and test patterns, SRAFs are generated while accommodating the mask manufacturing rules.*

The machine learning-based framework for SRAF generation is demonstrated in Fig. 5.3, where Problem 6 is mainly related to the training phase and Problem 7 is associated with the testing phase. In particular, this study aims at the machine learning-based method instead of traditional model-based or rule-based approaches. A complete mask optimization flow is achieved by combining the machine learning-based framework in Fig. 5.3 with other stages shown in Fig. 5.2.

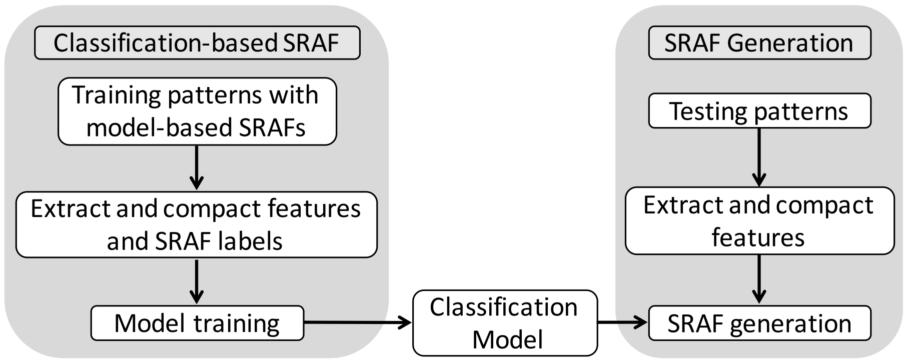

Classification-based SRAF
Training patterns with model-based SRAFs
Extract and compact features and SRAF labels
Model training
Classification Model
SRAF Generation
Testing patterns
Extract and compact features
SRAF generation

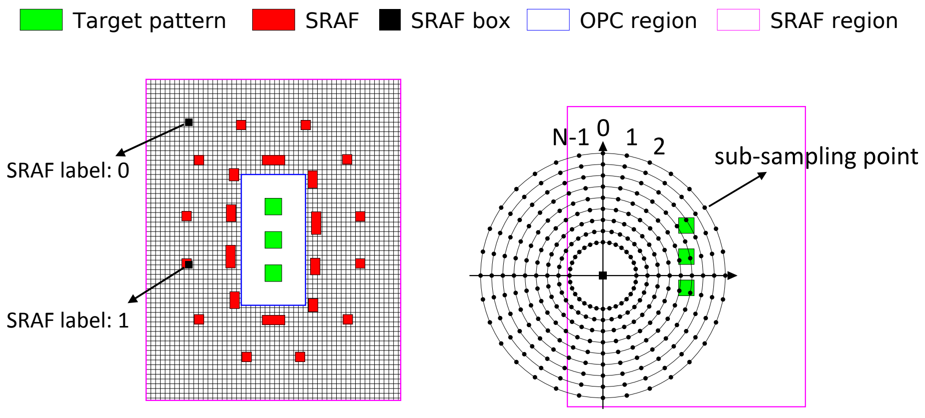
Figure 5.3: Machine learning-based SRAF generation.

5.4 Classification-based SRAF

5.4.1 Data Preparation

SRAF Label Extraction Given training patterns with model-based SRAFs on a 2D grid plane, we need to extract the training data, including the SRAF label and feature vector for each grid. As shown in Fig. 5.4(a), a 2D grid plane is imposed on the target patterns and model-based SRAFs. The coordinates of each grid are determined by the pre-set grid size. An SRAF box is introduced at each grid to decide the SRAF label from model-based SRAFs. Specifically, the SRAF box is a rectangle and the size is a parameter, which could be different from the grid size. The SRAF label of the grid is 0 if no model-based SRAF covering the SRAF box on the grid. The SRAF label is 1 when there is a model-based SRAF covering the entire SRAF box area. Therefore, the grid size of the 2D grid plane decides the granularity of the training data extraction while the SRAF box provides an alternative control on the SRAF label extraction accuracy. The SRAF label extraction will give a set of labels for all the grids, denoted as $\{y_0\}$.

In addition, an OPC region and an SRAF region are explicitly drawn in Fig. 5.4(a) to demonstrate SRAF-specific constraints. SRAF generation is not allowed in the OPC region since it is reserved for the OPC stage after the SRAF generation. Since the optical interference happens within some specific lithographic interaction window, the SRAF generation outside of the pre-determined SRAF region can be ignored. Both OPC region and SRAF region are created by expanding the edges of the target patterns by some specific distance. We define the distance of expansion for the OPC region and SRAF region as d_{opc} and d_{sraf}, respectively.



(a) (b)

Figure 5.4: (a) SRAF label extraction and sampling constraints, (b) CCCAS at one grid point.

Feature Extraction and Compaction The layout feature extraction plays an important role in the classification model calibration and prediction. The SRAFs benefit the printing of target patterns by delivering light to the positions of target patterns at proper phase. Thus, we need a layout feature that represents this physical phenomenon. The concentric circle with area sampling is an ideal candidate since it represents the information related to the concentric propagation of diffracted light from mask patterns [74]. We adapt it to the constrained concentric circle with area sampling (CCCAS) by incorporating the OPC region and SRAF region constraints discussed in Section 5.4.1. The CCCAS at one grid is illustrated in Fig. 5.4(b), where each circle centers at the grid and the minimum and maximum radius of the CCCAS are determined by d_{opc} and d_{sraf}, respectively. After transforming target patterns into the bitmap on the 2D plane, the CCCAS yields an $M \times N$ matrix, denoted as X_0, where M is the row number and N is the column number. As shown in Fig. 5.4(b), the column index of X_0 starts at the positive Y-axis with 0 and increases clockwise to $N-1$. The sub-sampling points, denoted as the black dots in Fig. 5.4(b), sharing the same angle to the origin are in the same column of X_0. The row index of X_0 starts with 0 at the circle with the smallest radius and increases to $M-1$ as the circle radius becomes larger. The sub-sampling points on the same circle are on the same row of X_0.

The consistent SRAF generation is an important issue, which means the same SRAFs will be generated surrounding the same target pattern configurations, i.e., the same optical conditions [46]. For example, in Fig. 5.5, the four grids are on axial symmetric positions of the grid plane with respect to the target patterns. If we assume the annular shape of the illumination source, the optical conditions of these four grids are the same and consistent SRAF generation scheme will give the same SRAF results. However, since the CCCAS at these four grids are different, denoted by different colors in Fig. 5.5, it is difficult for a classification model to achieve the same SRAF predictions. To achieve better SRAF consistency, we propose a novel feature compaction technique taking advantage of the illumination source symmetry. As shown in Fig. 5.5, this feature compaction technique transforms the CCCAS of symmetric grids into the CCCAS of the same grid, denoted as g, in the lower left of the grid plane. The sampling region of the grid (g) can be divided into four quadrants, i.e., I, II, III and IV. The target patterns mainly locate at the quadrant I of the sampling region of the grid (g), while target patterns mainly locate at different quadrants for other symmetric grids. For clearer explanations, we define the main quadrant as follows.

Definition 13 (Main Quadrant)**.** *The main quadrant for a grid is defined as the quadrant of the CCCAS region where target patterns mainly locate.*

By flipping the CCCAS of other symmetric grids with X or Y-axis as shown in Fig. 5.5, target patterns will always locate at the quadrant I of the sampling regions for symmetric grids, which leads to the same CCCAS results. Then, the classification model will give consistent SRAF predictions for axial symmetric grids.

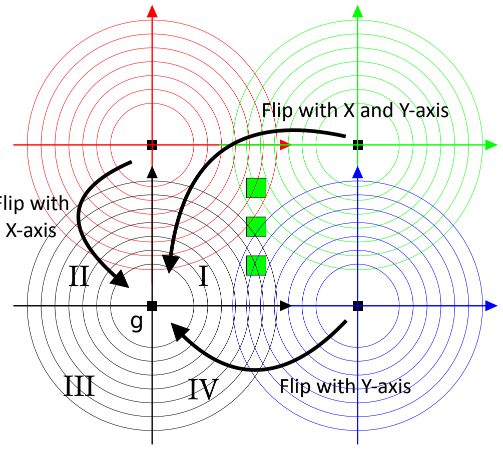


Figure 5.5: Feature compaction based on symmetry.

The details of the feature compaction technique are explained in Algorithm 9. The 2D feature matrix from CCCAS contains the bitmap value at each sub-sampling point. The bitmap values of sub-sampling points within each quadrant of the sampling region correspond to a set of columns in the 2D feature matrix. Thus, the main quadrant should have the maximum summation of bitmap values at the 2D feature matrix. From lines 3 to 9, we scan through the four quadrants of the CCCAS region and decide the main quadrant. The flipping of 2D matrix is performed in line 10 to transform the main quadrant to quadrant I as demonstrated in Fig. 5.5. For practical implementation, the flipping of 2D matrix can be achieved with simple column index switching. In line 11, the 2D feature matrix is flattened into a one-dimension (1D) feature vector for the classification model calibration. It shall be noted that, the illumination source symmetry-based feature compaction scheme can be easily extended to other symmetric scenarios, such as rotational symmetry.

Overall, the feature extraction and compaction will yield a set of 1D feature vectors with $M \times N$ dimensions.

Algorithm 9 Feature Compaction

```
Input: A $M \times N$ feature matrix $X_0$;
Output: Optimized feature vector $x_0$;
 1: Define $main\_quadrant = 1$ as the main quadrant;
 2: Define $max\_sum = 0$ as the maximum summation;
 3: for $index = 0$, $index < 4$, $index{+}{+}$ do;
 4:     Define $sum$ = summation of $X_0$ from column $index \times N/4$ to column $(index + 1) \times N/4$;
 5:     if $sum > max\_sum$ then;
 6:         $max\_sum = sum$;
 7:         $main\_quadrant = index + 1$;
 8:     end if
 9: end for
10: Flip $X_0$ based on $main\_quadrant$;
11: Flatten $X_0$ into a vector $x_0$ and return $x_0$;
```

5.4.2 Model Training

With the SRAF labels and feature vectors, a classification model is calibrated for SRAF predictions. In particular, the size of training data set and feature vector dimension can both be very large because high sampling accuracy is needed for classification-based SRAF. Moreover, the large training and testing data set sizes make the classification-based SRAF problem difficult to adopt support vector machine (SVM) and other advanced classification models from the perspective of calibration and prediction runtime. Thus, we adopt decision tree (DTree) and logistic regression (LGR) models for SRAF predictions with reasonable performance and runtime.

Decision Tree The DTree model is simple yet powerful, which partitions the feature space into a set of rectangles and calibrate a simple model (like a constant label for classification model) in each one [40]. Specifically, the DTree model calibration is to construct a decision tree from labeled training data in a top-down manner, using a metric to decide the best splits of a set of training data at each step [40]. For this application, the Gini index metric is used to decide the best splits at each step of calibration with the CART (Class and Regression Trees) algorithm. With the Gini index, the DTree classifier can estimate the probability of a label for each data sample by calculating the fraction samples of that label in a leaf [40].

Logistic Regression The LGR adopts the logistic function as the probabilistic estimation for each label of the training or testing data. The model calibration is typically achieved with the maximum likelihood method [40]. The LGR model is especially powerful for binary classification, which makes the calibration and prediction scalable to a large data set. Due to the large training data set in classification-based SRAF issue, $L2$ regularization is added to the LGR model to avoid overfitting. The LGR model provides the direct probabilistic estimation of labels for each data sample.

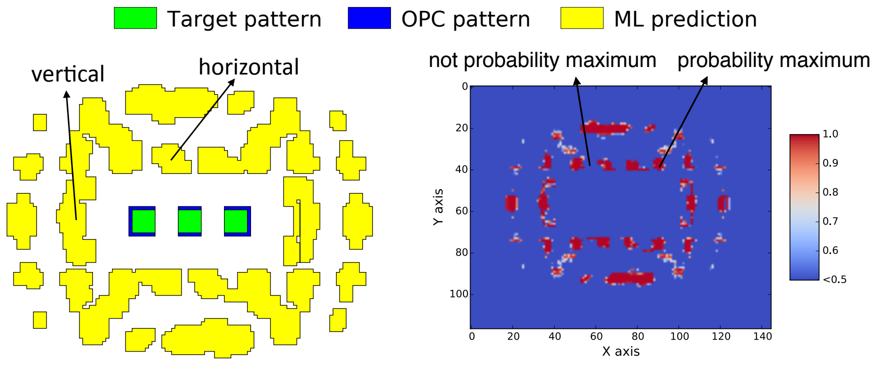
5.5 SRAF Generation

5.5.1 Predictions with Probability Maxima

The typical prediction with a binary classification model will be a label, i.e., 0 or 1, for each testing data. With the label prediction for each grid, clusters of grids will be labeled as 1, denoted as yellow grids, as shown in Fig. 5.6(a). After the label prediction, clusters of grids in Fig. 5.6(a) cannot be directly treated as SRAFs because they may violate the mask manufacturing rules or be printed due to large critical dimensions. Instead of using SRAF label for the grid prediction, we propose predictions with probability maxima to simplify the clusters of SRAF grids. When a classification model is calibrated, the probability of the label to be 1, denoted as p_1, can be calculated for LGR and DTree as explained in Section 5.4.2. Then, a probability map on the 2D grid plane can be attained as shown in Fig. 5.6(b). To simplify the clusters of grids for SRAF generation, we only insert SRAFs at grids with probability maxima. A grid with probability maximum means the probability (p_1) at that particular grid is larger than that at any other neighboring grids. The idea of predictions with probability maxima originates from the model-based SRAF approach. Model-based SRAFs are generated using the guidance map from lithographic computations [90,91,101,115]. A guidance map is also grid based and has intensity assigned to each grid, where SRAFs will only be inserted at those intensity maxima. Thus, we adopt the similar idea during predictions with probability maxima since model-based SRAFs are used as the training data for the classification model calibration.

5.5.2 SRAF Simplification

Using predictions with probability maxima, clusters of grids will be predicted as SRAFs on the 2D grid plane but the mask manufacturing and SRAF



(a) (b)

Figure 5.6: SRAF predictions: (a) label predictions, (b) predictions with probability maxima.

printing issues are not fully resolved. The SRAF simplification phase aims at simplifying these clusters of grids into SRAFs satisfying the mask manufacturing rules. A greedy simplification scheme is proposed by grid merging and shrinking the SRAFs into rectangular shapes while accommodating mask manufacturing rules.

The overall algorithm of SRAF generation is shown in Algorithm 10. In lines 1-2, we predict SRAFs at grids with probability maxima from the classification model. In line 3, the grids with probability maxima are merged into polygons, which is followed by the spacing rule check and shrinking the polygons to remove violations in line 4. From line 5 to 14, each polygon is processed to generate a rectangular SRAF. Particularly, in line 7, the main direction of SRAF is detected based on the bounding box of target patterns. In line 8, the bounding box of the polygon is shrunk to achieve a rectangular SRAF parallel to that of target patterns. As illustrated in Fig. 5.6(a), the main

Algorithm 10 SRAF generation

```
Input: A 2D grid plane, a classification models, a set of mask manufacturing
    rules;
Output: The mask manufacturing friendly set $SRAF$;
 1: Compute the probability of label 1 for each grid;
 2: SRAF predictions at grids with probability maxima;
 3: Merge SRAF grids into a polygon set $SRAF_{pg}$;
 4: Spacing rule check and shrink polygons in $SRAF_{pg}$ to remove violations;
 5: for each polygon in $SRAF_{pg}$ do;
 6:     Define $BBox$ as the bounding box of polygon;
 7:     Detect the main direction of polygon as direction;
 8:     Shrink $BBox$ size based on direction;
 9: end for
10: Rule check and shrink rectangles in $SRAF$ to remove violations;
11: Return $SRAF$;
```

direction of the polygon on the top is horizontal while the main direction on the left is vertical, both of which are parallel to the bounding box of target patterns. With the SRAF simplification, the mask manufacturing-friendly SRAFs can be generated for testing patterns.

5.6 Experimental Results

We have implemented the machine learning framework in Python and accelerated with Cython [14] and parallel computations. The optical model, model-based SRAF, MRC/LCC recipes and the SRAF simplification are implemented using Calibre script language with the industry-strength setup. All experiments are performed on an 8-core Linux machine with 3.4GHz Intel(R) Core and 32GB memory. For the optical model, the wavelength (λ) and numerical aperture (NA) are set as 193nm and 1.35, respectively. The annular illumination source is used with outer sigma as 0.9 and inner sigma as 0.6. Compact model 1 from Calibre is adopted as the resist model. In the LCC, the outer/inner contours are generated using lithographic process window conditions as a focus variation of $\pm 30nm$ and a dose variation of $\pm 3.0\%$. For model-based SRAF generation, process window conditions above are considered and SRAF manufacturing rules are set as max_width = $40nm$, min_space = $60nm$, max_length = $90nm$. We test the SRAF generation framework on two types of contact patterns. One type is dense contact arrays with contact width and space fixed as $70nm$. We have dense contact patterns because redundant vias are needed to improve yield during layout design. The other type is sparse contact patterns, where the contact width is $70nm$ but the space between contact holes is random and the minimum space is $70nm$. For CCCAS, the grid size is set as $10nm$, the SRAF box size is set as $40nm$ and radius step size is $s_r = 15nm$. For SRAF-specific constraints, d_{opc} and d_{sraf} are set as $100nm$ and $600nm$, respectively.

Model Training: A set of training patterns and model-based SRAFs are needed to extract the training data and calibrate the classification model for SRAF predictions on both dense and sparse testing patterns. From the extensive experiments, the training patterns in Fig. 5.7 yield the best training and testing accuracy. For the dense contact patterns in Fig. 5.7(a), the width and space are fixed as $70nm$. The width of sparse contact patterns in Fig. 5.7(b) is $70nm$, while the space is $350nm$. In particular, since the training patterns are symmetric and feature compaction scheme has been proposed, we only need to sample the lower left part of the layout clip for training data. This is beneficial for the classification model calibration since the training data size can be reduced by 3/4 without losing the critical SRAF information. The training data set statistics are summarized in Table 5.1. For CCCAS, the number of circle is set as $M = \lfloor (d_{sraf} - d_{opc})/s_r \rfloor = 33$. The number of sub-sampling points in each circle is set as $N = 32$ to guarantee sampling accuracy for sparse contact patterns. Then, the feature vector dimension is $M \times N = 1056$. By combining the training data from dense and sparse patterns, we have 14609 training samples. We have 95412 testing samples from dense patterns and 803756 testing samples from sparse patterns.

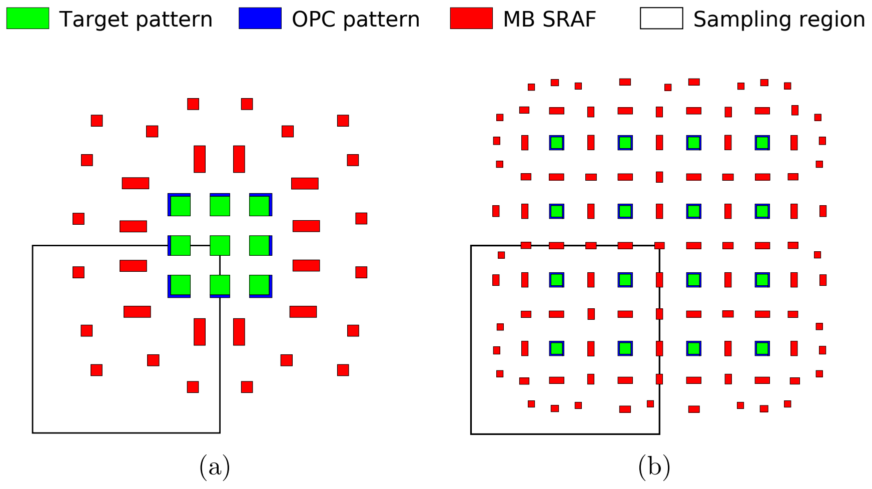


Figure 5.7: Training layout: (a) dense contact patterns, (b) sparse contact patterns.

The data set statistics demonstrate the high feature vector dimension and large training data size. The feature vector dimension is difficult to be further reduced since each sub-sampling point contains the information related to the target patterns. Moreover, each training data sample within the lithographic interaction window is considered valuable, so there is little redundancy within the training data set. As discussed in Section 5.4.2, advanced classification models, such as support vector machine, are not applicable to the classification-based SRAF domain due to large training and testing data sets. In practice, we find that the reasons are twofold. First, the training of advanced classification models has high complexity, which is not as scalable as the simple classification models to large training data set. Second, advanced classification models have more complex prediction model calibrated, which means the testing time for each testing sample would be longer than that of simple models. Since the grid-based approach is used here, the runtime overhead accumulates and even becomes unaffordable over the huge amount of testing samples extracted from testing patterns.

Table 5.1: Data set statistics for classification

feature vector dimension	1056
Training sample #	14609
Testing sample # from dense patterns	95412
Testing sample # from sparse patterns	803756

We further compare different classification models, including DTree and LGR, for the SRAF generation framework and data statistics are shown in Table 5.2. The F_1 *score* is computed as:

$$F_1\ score = \frac{2 * precision * recall}{(precision + recall)} \tag{5.1}$$

Table 5.2: Comparisons on different classification models

		DTree	LGR
Training	F_1 *score*(1)	0.9983	0.9938
	F_1 *score*(0)	0.9855	0.9462
	Calibration time(s)	5.66	1.06
Dense Testing Patterns	F_1 *score*(1)	0.9499	0.9557
	F_1 *score*(0)	0.3319	0.4132
	Testing time(s)	3.83	3.83
Sparse Testing Patterns	F_1 *score*(1)	0.8787	0.8724
	F_1 *score*(0)	0.2646	0.2629
	Testing time(s)	50.44	41.47

where *precision* is the number of true positive results divided by the number of all positive results and *recall* is the number of true positive results divided by the number of positive results that should have been returned [34]. In particular, F_1 *score* is best at 1.0 and worst at 0.0 for comparison. Since either label 1 or label 0 can be treated as a positive result, we compute the F_1 *score* for both labels, denoted as F_1 *score*(1) and F_1 *score*(0), respectively. From Table 5.2, although DTree achieves better F_1 *score*(1) and F_1 *score*(0) on training data, the testing F_1 *score*(0) and F_1 *score*(1) are worse than LGR on dense patterns by 0.081 and 0.006, respectively. We only see a difference within 0.01 in F_1 *score* for sparse testing patterns. This means LGR is better than DTree due to its better testing accuracy and LGR is less prone to the overfitting with large training data set and high dimension feature vectors. In addition, the model calibration time and testing time on sparse testing patterns for LGR are less than DTree but the difference is non-significant. Moreover, we observe the F_1 *score*(1) is much better than F_1 *score*(0) on all testing data for both classification models. This means that most grids with SRAFs of testing patterns are labeled as 1 correctly but some other grids without SRAFs that should be labeled as 0 are incorrectly labeled as 1. This also proves the necessity of predictions with probability maxima and the SRAF simplification stage later on, which essentially reduces the number of grids labeled as 1 and potentially improves the $F_1\ score(0)$ for testing data.

SRAF Generation We add testing layout clips to demonstrate the strength of predictions with probability maxima and SRAF simplification schemes. Since the LGR based approach performs better than the DTree based approach, we only show the SRAFs from LGR predictions for clear explanations. As illustrated in Fig. 5.8, we compare the SRAFs generated using different machine learning (ML) predictions, i.e., label predictions and predictions with probability maxima, followed by the SRAF simplification phase. Predictions with probability maxima can simplify the clusters of grids labeled as 1, i.e., breaking large clusters into small clusters, which benefits the SRAF simplification stage. Thus, the SRAFs generated using predictions with probability maxima in Fig. 5.8(b) are much better than those in Fig. 5.8(a) in terms of PV band from the LCC.

SRAF Consistency We further demonstrate the benefit of SRAF consistency improvement from the feature compaction technique in Section 5.4.1. The SRAF generation from the model-based method using Calibre [Calibre,

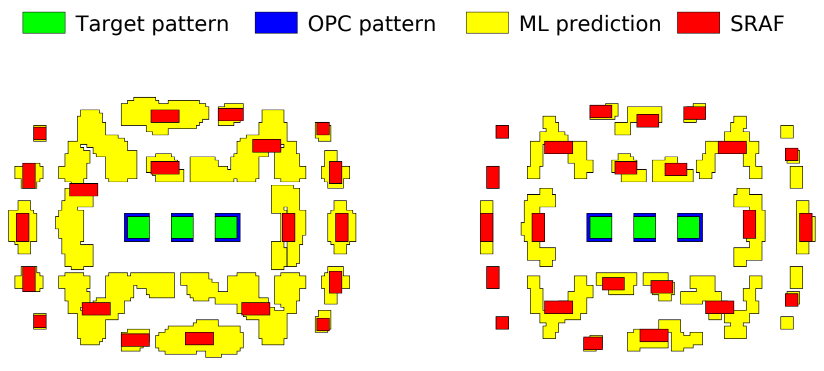


Figure 5.8: SRAF generations: (a) label predictions, (b) predictions with probability maxima.

v2015.2_36.27], LGR without feature compaction and LGR with feature compaction are shown in Fig. 5.9(a), (b) and (c), respectively. Since annular illumination source is used, axial symmetric grids share the same optical environment and the consistent SRAF generation should yield the same SRAFs at axial symmetric grids. The feature compaction scheme would transform the feature matrices extracted from axial symmetric grids to the same feature vector. Therefore, the SRAFs in Fig. 5.9(c) are more consistent than those in Fig. 5.9(b). Moreover, we have even achieved SRAFs with better consistency than the model-based method shown in Fig. 5.9(a).

We further demonstrate the SRAFs from LGR predictions on both dense and sparse testing patterns as shown in Fig. 5.10. The layout clip of dense contact patterns is defined as $m \times n$ dense contact patterns, if the layout clip contains a contact array with m rows and n columns. Fig. 5.10(a) and (b)

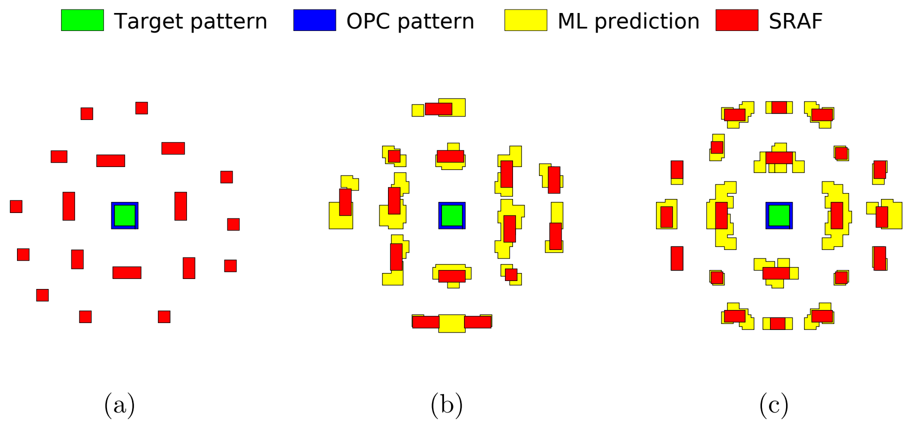
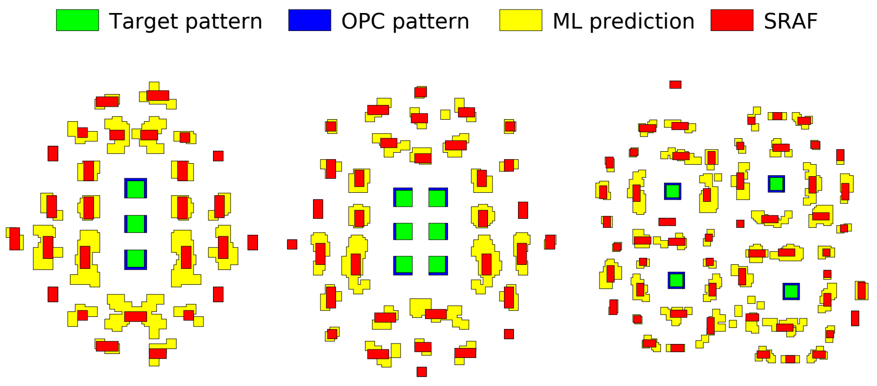


Figure 5.9: SRAFs for the isolated contact pattern: (a) model-based, (b) LGR without feature compaction, (c) LGR with feature compaction.

show two cases of redundant vias in real designs. The LGR based SRAF generation can obtain acceptable SRAFs on these dense contact patterns but the degradation of SRAF consistency is observed. The reasons are twofold. First, the training data with the model-based SRAFs are not perfectly consistent as shown in Fig. 5.7. Then, it is difficult to guarantee the consistent SRAF generation with the classification model calibrated with these training data. Second, the CCCAS results may be slightly different for axial symmetric grids due to the grid error within the 2D grid plane. Fig. 5.10(c) illustrates a small layout clip of random contact patterns, which proves the capability of the machine learning-based SRAF generation on random sparse contact patterns.

Lithography Compliance Check To evaluate the practical lithographic performance, we combine the SRAF generation with a complete mask opti-



(a) (b) (c)

Figure 5.10: Testing contact patterns: (a) 3×1 dense contact patterns, (b) 3×2 dense contact patterns, (c) sparse contact patterns

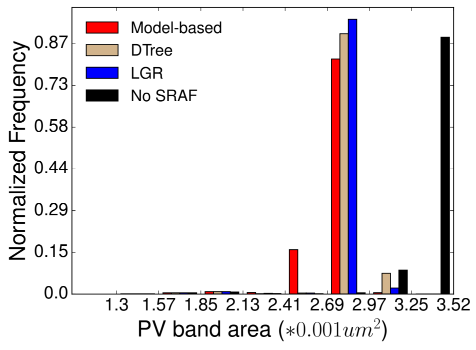
mization flow as shown in Fig. 5.2, where model-based OPC and LCC are performed using Mentor Calibre tool. We compare the model-based, LGR and DTree approaches in terms of PV band and EPE on both dense and sparse testing patterns. Specifically, we collect the PV band value for each contact and the EPE value at the center of the four edges of each contact at nominal conditions. The mean values are summarized and compared in Table 5.3. We add the PV band without SRAFs to better demonstrate the benefit from SRAF generation. The model-based approach reduces the PV band from 3.3064 to 2.7705, which is 16.21% reduction compared to no SRAF insertion. Meanwhile, we obtain 13.37% and 14.09% PV band reduction from DTree and LGR, respectively. The LGR based SRAF generation achieves better performance as expected because LGR model is less prone to overfitting compared to DTree model. In particular, there is only 2.12% PV band degradation from the model-based approach to LGR based approach. We take the absolute values when calculating the EPE mean to avoid the cancellations between positive and negative values of EPE. The LGR based SRAF generation yields the smallest EPE mean value, which even outperforms the model-based approach. This means there is some trade-off between PV band and EPE because different SRAF results lead to different OPC results. It is very difficult to improve the PV band and EPE simultaneously with a robust mask optimization flow.

Table 5.3: Comparisons on PV band and EPE

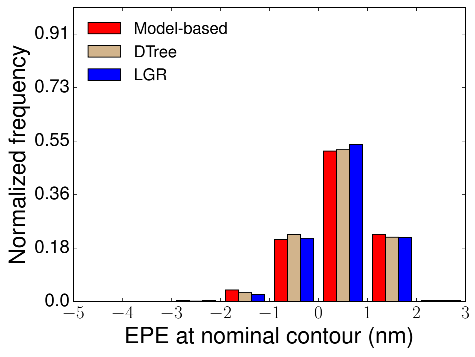
Mean value	No SRAF	Model-based	DTree	LGR
PV band ($.001um^2$)	3.306	2.771	2.864	2.841
Absolute of EPE (nm)	3.636	0.539	0.523	0.501

We collect the PV band and EPE values for each contact and further plot the data in histograms as shown in Fig. 5.11. Fig. 5.11(a) shows that SRAF insertion significantly improves the PV band and model-based SRAF gives the best performance. The LGR based approach is slightly worse than model-based method but performs better than DTree based method. Fig. 5.11(b) shows that LGR based SRAF generation achieves the best EPE performance.

Run Time We compare our machine learning-based SRAF generation with the commercial Calibre tool, i.e., model-based SRAFs. The mask optimization techniques, including SRAF generation and OPC, usually apply to small layout windows due to the high computational cost [12]. Therefore, we choose



(a)



(b)

Figure 5.11: Comparison among different schemes: (a) PV band distribution, (b) EPE distribution.

small layout windows with dense contact patterns for runtime comparisons between different SRAF generation approaches. In Fig. 5.12, different dense contact patterns, denoted as $m \times n$ contact patterns, are used for runtime comparisons. The areas of these layout windows considering SRAF regions are in the range from $1um^2$ to $2um^2$. The runtime for the machine learning-based approach includes runtime for feature extraction and compaction, predictions with probability maxima and SRAF simplification. The model calibration time is not included since it is one-time computation.

The average runtime for model-based SRAF, LGR based SRAF and DTree based SRAF are $5.14s$, $0.41s$ and $0.41s$, respectively. Although we are using a different programming language and database from the commercial tool, we still obtain over 10X speed-up from machine learning-based SRAF generation compared to the model-based approach in Calibre [Calibre, v2014.4_18.13]. We also check the runtime of the model-based OPC from different SRAF gen-

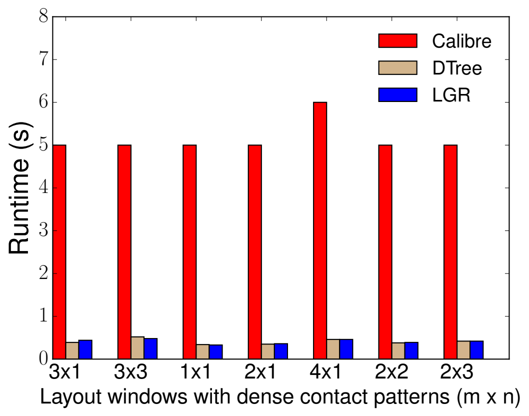


Figure 5.12: Run time comparison among different schemes on different layout windows.

eration approaches and ensure that they are approximately the same.

5.7 Summary

This chapter demonstrates a machine learning-based framework for SRAF generation. A robust feature extraction scheme is proposed by adapting the concentric circle with area sampling considering SRAF-specific constraints. We further propose a novel feature compaction technique to reduce the training data size and improve the SRAF consistency. During SRAF predictions with a classification model, instead of using conventional label predictions, predictions with probability maxima are proposed to achieve mask manufacturing-friendly SRAFs. Experimental results show that LGR based SRAF generation obtains 10X speed-up in layout windows and better EPE with affordable degradation in PV band, compared to the commercial Calibre tool.

Chapter 6

Conclusion and Future Work

This dissertation proposes novel CAD algorithms and methodologies to enable standard cell optimization and physical design in advanced technology nodes. A standard cell pin accessibility evaluation and optimization engine is presented to guide standard cell designers for incremental cell layout optimization. Our cell evaluation tool has been prototyped by leading IP vendor, **ARM Inc.** Pin access planning schemes and concurrent optimization techniques are discussed to efficiently resolve the pin access interference for a sequential router and a negotiation-congestion-based router, respectively. Our pin access and routing co-optimization studies have demonstrated much better solution qualities than state-of-the-art manufacturing-friendly routers. In the post-routing stage, redundant local-loop insertion is comprehensively studied to improve manufacturing yield under advanced manufacturing constraints. Furthermore, machine learning-based SRAF generation techniques are proposed to achieve promising trade-offs between computational efforts and lithographic performance, compared to conventional model-based and rule-based approaches. Our SRAF generation tool has been prototyped within the memory division in **Toshiba Microelectronics**.

Unidirectional routing style is becoming the mainstream industrial routine for high-density metal layers in advanced technology nodes. SRAF generation will continue to play an important role in manufacturing yield enhancement for contacts/vias with smaller and smaller sizes. For better design closure in advanced technology nodes, we anticipate more and more research efforts from following perspectives.

- Generic standard cell library evaluation tool. Hit points and hit point combinations are elementary metrics for pin accessibility evaluation and optimization. For manual design and optimization, a generic scoring function is favorable while considering pin accessibility, colorability [107, 108] and manufacturing-specific design rules.

- Placement mitigation. The complexities of cell coloring, inter-cell pin accessibility and other manufacturing constraints interfere with each other, which makes it difficult to predict the appropriate spacing among standard cells [39, 97]. Advanced placement engine is needed to properly allocate white space among standard cells for better design closure.

- Concurrent multi-layer detailed routing. Concurrent pin access optimization can only resolve routing resource competitions on one metal layer. It remains an open research problem of how to enable concurrent multi-layer detailed routing beyond negotiation-congestion-based routing without significant wirelength or runtime overhead [45].

- Global and detailed routing co-optimization. Most existing manufacturing-friendly routing studies focus on detailed routing phase as detailed router creates real physical patterns, which are directly associated with manufacturing constraints. However, this is no longer an effective approach when the routing resource competitions are severe and many local nets cannot be routed due to inaccurate routing resource modeling for global routing, which requires global and detailed routing co-optimization.

- Advanced learning techniques for SRAF generation. Many advanced learning techniques, such as boosting and active learning, have been proposed in the machine learning area, which demonstrate much better learning accuracy than simple learning models. This leads to a much broader research scope for the machine learning-based SRAF generation.

Bibliography

[1] Cbc. `http://www.coin-or.org/projects/Cbc.xml`.

[2] Clp. `http://www.coin-or.org/Clp/index.html`.

[3] ITRS. `http://www.itrs.net`.

[4] Predictive Technology Model ver. 2.1. `http://ptm.asu.edu`.

[5] Ashish Agarwal, Sooraj Bhat, Alexander Gray, and Ignacio E Grossmann. Automating mathematical program transformations. In *Practical Aspects of Declarative Languages*, pages 134–148. Springer, 2010.

[6] Markus Ahrens, Michael Gester, Niko Klewinghaus, Dirk Muller, Sven Peyer, Christian Schulte, and Gustavo Tellez. Detailed routing algorithms for advanced technology nodes. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 34(4):563–576, 2015.

[7] C.J. Alpert, Z. Li, C.N. Sze, and Y. Wei. Consideration of local routing and pin access during VLSI global routing, April 4 2013. US Patent App. 13/252,067.

[8] Brent A Anderson, Jeanne P Bickford, Markus Buehler, Jason D Hibbeler, Juergen Koehl, and Edward J Nowak. Redundant micro-loop

structure for use in an integrated circuit physical design process and method of forming the same, July 31 2012. US Patent 8,234,594.

[9] John Christopher Arnold, Sean D Burns, Sivananda K Kanakasabapathy, and Yunpeng Yin. Self aligning via patterning, October 30 2012. US Patent 8,298,943.

[10] Ahmed Awad, Atsushi Takahashi, Satoshi Tanaka, and Chikaaki Kodama. A fast process variation and pattern fidelity aware mask optimization algorithm. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 238–245, 2014.

[11] Yongchan Ban, Kevin Lucas, and David Z. Pan. Flexible 2D Layout Decomposition Framework for Spacer-type Double Pattering Lithography. In *ACM/IEEE Design Automation Conference (DAC)*, pages 789–794, 2011.

[12] Sean Banerjee, Zhuo Li, and Sani R Nassif. ICCAD-2013 CAD contest in mask optimization and benchmark suite. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 271–274, 2013.

[13] Shabbir Batterywala, Narendra Shenoy, William Nicholls, and Hai Zhou. Track Assignment: A Desirable Intermediate Step Between Global Routing and Detailed Routing. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 59–66, 2002.

[14] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D.S. Seljebotn, and K. Smith. Cython: The best of both worlds. *Computing in Science Engineering*, 13(2):31 –39, 2011.

[15] Jeanne Bickford, Markus Bühler, Jason Hibbeler, Jürgen Koehl, Dirk Müller, Sven Peyer, and Christian Schulte. Yield improvement by local wiring redundancy. In *IEEE Proc. International Symposium on Quality Electronic Design (ISQED)*, pages 6 pp.–478, 2006.

[16] Cadence. Cadence SOC Encounter. `http://www.cadence.com/`, 2012.

[17] Fong-Yuan Chang, Ren-Song Tsay, and Wai-Kei Mak. How to consider shorts and guarantee yield rate improvement for redundant wire insertion. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 33–38, 2009.

[18] Bharani Chava, David Rio, Yasser Sherazi, Darko Trivkovic, Werner Gillijns, Peter Debacker, Praveen Raghavan, Ahmad Elsaid, Mircea Dusa, Abdelkarim Mercha, et al. Standard cell design in N7: EUV vs. immersion. In *Proc. of SPIE*, pages 94270E–94270E–9, 2015.

[19] Geir Dahl and Njål Foldnes. LP based heuristics for the multiple knapsack problem with assignment restrictions. *Annals of Operations Research*, 146(1):91–104, 2006.

[20] William A Dees Jr and Patrick G Karger. Automated rip-up and reroute techniques. In *ACM/IEEE Design Automation Conference (DAC)*,

pages 432–439, 1982.

[21] William A Dees Jr and Robert J Smith II. Performance of interconnection rip-up and reroute strategies. In *ACM/IEEE Design Automation Conference (DAC)*, pages 382–390, 1981.

[22] Duo Ding, Xiang Wu, Joydeep Ghosh, and David Z. Pan. Machine learning based lithographic hotspot detection with critical-feature extraction and classification. In *IEEE International Conference on IC Design and Technology (ICICDT)*, 2009.

[23] Yixiao Ding, Chris Chu, and Wai-Kei Mak. Detailed routing for spacer-is-metal type self-aligned double/quadruple patterning lithography. In *ACM/IEEE Design Automation Conference (DAC)*, pages 69:1–69:6, 2015.

[24] Antun Domic. Some observations on the physical design of the next decade. In *ACM International Symposium on Physical Design (ISPD)*, pages 61–61, 2016.

[25] Dragoljub Gagi Drmanac, Frank Liu, and Li-C. Wang. Predicting variability in nanoscale lithography processes. In *ACM/IEEE Design Automation Conference (DAC)*, pages 545–550, 2009.

[26] Yuelin Du, Qiang Ma, Hua Song, James Shiely, Gerard Luk-Pat, Alexander Miloslavsky, and Martin DF Wong. Spacer-is-dielectric-compliant

detailed routing for self-aligned double patterning lithography. In *ACM/IEEE Design Automation Conference (DAC)*, pages 93:1–93:6, 2013.

[27] Shao-Yun Fang. Cut mask optimization with wire planning in self-aligned multiple patterning full-chip routing. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 396–401, 2015.

[28] Shao-Yun Fang, Wei-Yu Chen, and Yao-Wen Chang. A novel layout decomposition algorithm for triple patterning lithography. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1185–1190, 2012.

[29] Jhih-Rong Gao and David Z. Pan. Flexible Self-aligned Double Patterning Aware Detailed Routing with Prescribed Layout Planning. In *ACM International Symposium on Physical Design (ISPD)*, pages 25–32, 2012.

[30] Jhih-Rong Gao, Xiaoqing Xu, Yu Bei, and David Z Pan. MOSAIC: Mask optimizing solution with process window aware inverse correction. In *ACM/IEEE Design Automation Conference (DAC)*, pages 52:1–52:6, 2014.

[31] Jhih-Rong Gao, Bei Yu, and David Z. Pan. Self-aligned Double Patterning Layout Decomposition with Complementary E-Beam Lithography. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2014.

[32] Rani S Ghaida, Kanak B Agarwal, Sani R Nassif, Xin Yuan, Lars W Liebmann, and Puneet Gupta. Layout Decomposition and Legalization for Double-Patterning Technology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 32(2):202–215, 2013.

[33] W Gillijns, SMY Sherazi, D Trivkovic, B Chava, B Vandewalle, V Gerousis, P Raghavan, J Ryckaert, K Mercha, D Verkest, et al. Impact of a SADP flow on the design and process for N10/N7 metal layers. In *Proc. of SPIE*, pages 942709–942709–9, 2015.

[34] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *Advances in information retrieval*, pages 345–359. Springer, 2005.

[35] Allan Gu and Avideh Zakhor. Optical proximity correction with linear regression. *IEEE Transactions on Semiconductor Manufacturing*, 21(2):263–271, 2008.

[36] Puneet Gupta. What is process window? *SIGDA Newsl.*, 40(8):1–1, August 2010.

[37] Gurobi. GUROBI. `http://www.gurobi.com/html/academic.html`, 2014.

[38] Antonin Guttman. *R-trees: a dynamic index structure for spatial searching*, volume 14. ACM, 1984.

[39] Kwangsoo Han, Andrew B Kahng, and Hyein Lee. Scalable detailed placement legalization for complex sub-14nm constraints. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 867–873, 2015.

[40] Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.

[41] Michael Held, Philip Wolfe, and Harlan P Crowder. Validation of subgradient optimization. *Mathematical programming*, 6(1):62–88, 1974.

[42] Fook-Luen Heng, Zhan Chen, and Gustavo E Tellez. A VLSI artwork legalization technique based on a new criterion of minimum layout perturbation. In *ACM International Symposium on Physical Design (ISPD)*, pages 116–121, 1997.

[43] Meng-Kai Hsu, Nitesh Katta, Homer Yen-Hung Lin, Keny Tzu-Hen Lin, King Ho Tam, and Ken Chung-Hsing Wang. Design and manufacturing process co-optimization in nano-technology. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 574–581, 2014.

[44] Wenbin Huang, Daniel Morris, Neal Lafferty, Lars Liebmann, Kaushik Vaidyanathan, Kafai Lai, Larry Pileggi, and Andrzej J Strojwas. Local loops for robust inter-layer routing at sub-20 nm nodes. In *Proc. of SPIE*, pages 83270D–83270D–9, 2012.

[45] Xiaotao Jia, Yici Cai, Qiang Zhou, Gang Chen, Zhuoyuan Li, and Zuowei Li. MCFRoute: a detailed router based on multi-commodity flow method. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 397–404, 2014.

[46] Jin-Hyuck Jun, Minwoo Park, Chanha Park, Hyunjo Yang, Donggyu Yim, Munhoe Do, Dongchan Lee, Taehoon Kim, Junghoe Choi, Gerard Luk-Pat, et al. Layout optimization with assist features placement by model based rule tables for 2x node random contact. In *Proc. of SPIE*, pages 94270D–94270D–10, 2015.

[47] Andrew B Kahng. Research directions for coevolution of rules and routers. In *ACM International Symposium on Physical Design (ISPD)*, pages 122–125, 2003.

[48] Andrew B Kahng, Bao Liu, and Ion I Mandoiu. Non-tree routing for reliability and yield improvement. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 260–266, 2002.

[49] Andrew B. Kahng, Chul-Hong Park, Xu Xu, and Hailong Yao. Layout decomposition for double patterning lithography. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 465–472, 2008.

[50] Rony Kay and Rob A Rutenbar. Wire packing: a strong formulation of crosstalk-aware chip-level track/layer assignment with an efficient integer

programming solution. In *ACM International Symposium on Physical Design (ISPD)*, pages 61–68, 2000.

[51] Byung-Sung Kim, Yoo-Hyun Kim, Sung-Ho Lee, Sung-Il Kim, Sang-Rok Ha, Juhwan Kim, and Alexander Tritchkov. Pixel-based SRAF implementation for 32nm lithography process. In *Proc. of SPIE*, pages 71220T–71220T–11, 2008.

[52] Chikaaki Kodama, Hirotaka Ichikawa, Koichi Nakayama, Toshiya Kotani, Shigeki Nojima, Shoji Mimotogi, Shinji Miyamoto, and Atsushi Takahashi. Self-Aligned Double and Quadruple Patterning-Aware Grid Routing with Hotspots Control. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 267–272, 2013.

[53] Chikaaki Kodama, Toshiya Kotani, Shigeki Nojima, and Shoji Mimotogi. Sub-resolution assist feature arranging method and computer program product and manufacturing method of semiconductor device, August 19 2014. US Patent 8,809,072.

[54] Jian Kuang, Wing-Kai Chow, and Evangeline FY Young. A robust approach for process variation aware mask optimization. In *Proc. Design, Automation and Test in Eurpoe (DATE)*, pages 1591–1594, 2015.

[55] Jian Kuang and Evangeline F.Y. Young. An efficient layout decomposition approach for triple patterning lithography. In *ACM/IEEE Design Automation Conference (DAC)*, pages 69:1–19:6, 2013.

[56] Kuang-Yao Lee, Cheng-Kok Koh, Ting-Chi Wang, and Kai-Yuan Chao. Optimal post-routing redundant via insertion. In *ACM International Symposium on Physical Design (ISPD)*, pages 111–117, 2008.

[57] Kuang-Yao Lee, Shing-Tung Lin, and Ting-Chi Wang. Redundant via insertion with wire bending. In *ACM International Symposium on Physical Design (ISPD)*, pages 123–130, 2009.

[58] Kuang-Yao Lee and Ting-Chi Wang. Post-routing redundant via insertion for yield/reliability improvement. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 303–308, 2006.

[59] Kuang-Yao Lee, Ting-Chi Wang, and Kai-Yuan Chao. Post-routing redundant via insertion and line end extension with via density consideration. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 633–640, 2006.

[60] Cheok-Kei Lei, Po-Yi Chiang, and Yu-Min Lee. Post-routing redundant via insertion with wire spreading capability. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 468–473, 2009.

[61] Lars Liebmann, Albert Chu, and Paul Gutwin. The daunting complexity of scaling to 7nm without EUV: Pushing DTCO to the extreme. In *Proc. of SPIE*, pages 942702–942702–12, 2015.

[62] Shing-Tung Lin, Kuang-Yao Lee, Ting-Chi Wang, Cheng-Kok Koh, and Kai-Yuan Chao. Simultaneous redundant via insertion and line end extension for yield optimization. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 633–638, 2011.

[63] Tzu-Mu Lin and Carver A Mead. Signal delay in general rc networks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 3(4):331–349, 1984.

[64] Iou-Jen Liu, Shao-Yun Fang, and Yao-Wen Chang. Overlay-aware detailed routing for self-aligned double patterning lithography using the cut process. In *ACM/IEEE Design Automation Conference (DAC)*, pages 50:1–50:6, 2014.

[65] Kevin Lucas, Chris Cork, Bei Yu, Gerry Luk-Pat, Ben Painter, and David Z. Pan. Implications of triple patterning for 14 nm node design and patterning. In *Proc. of SPIE*, volume 8327, 2012.

[66] Gerard Luk-Pat, Alex Miloslavsky, Ben Painter, Li Lin, Peter De Bisschop, and Kevin Lucas. Design compliance for spacer is dielectric (SID) patterning. In *Proc. of SPIE*, pages 83260D–83260D–13, 2012.

[67] Gerard Luk-Pat, Ben Painter, Alex Miloslavsky, Peter De Bisschop, Adam Beacham, and Kevin Lucas. Avoiding wafer-print artifacts in spacer is dielectric (SID) patterning. In *Proc. of SPIE*, pages 868312–868312–12, 2013.

[68] Fangyi Luo, Yongbo Jia, and Wayne Wei-Ming Dai. Yield-preferred via insertion based on novel geotopological technology. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 730–735, 2006.

[69] Rui Luo. Optical proximity correction using a multilayer perceptron neural network. *Journal of Optics*, 15(7):075708, 2013.

[70] Xu Ma, Bingliang Wu, Zhiyang Song, Shangliang Jiang, and Yanqiu Li. Fast pixel-based optical proximity correction based on nonparametric kernel regression. *Journal of Microlithography, Microfabrication and Microsystems*, 13(4):043007–043007, 2014.

[71] Yuangsheng Ma, Jason Sweis, Hidekazu Yoshida, Yan Wang, Jongwook Kye, and Harry J Levinson. Self-aligned double patterning (SADP) compliant design flow. In *Proc. of SPIE*, pages 832706–832706–13, 2012.

[72] Chris A Mack. *Field guide to optical lithography*, volume 6. SPIE Press Bellingham, Washington, USA, 2006.

[73] Tetsuaki Matsunawa, Jhih-Rong Gao, Bei Yu, and David Z Pan. A new lithography hotspot detection framework based on adaboost classifier and simplified feature extraction. In *Proc. of SPIE*, pages 94270S–94270S–11, 2015.

[74] Tetsuaki Matsunawa, Bei Yu, and David Z Pan. Optical proximity correction with hierarchical bayes model. In *Proc. of SPIE*, pages 94260X–94260X–10, 2015.

[75] Larry McMurchie and Carl Ebeling. PathFinder: A Negotiation-based Performance-driven Router for FPGAs. In *ACM Symposium on FPGAs*, pages 111–117, 1995.

[76] Minoo Mirsaeedi, J. Andres Torres, and Mohab Anis. Self-aligned double-patterning (SADP) friendly detailed routing. In *Proc. of SPIE*, pages 79740O–79740O–9, 2011.

[77] Fumiharu Nakajima, Chikaaki Kodama, Hirotaka Ichikawa, Koichi Nakayama, Shigeki Nojima, Toshiya Kotani, Shoji Mimotogi, and Shinji Miyamoto. Detailed routing with advanced flexibility and in compliance with self-aligned double patterning constraints. In *Proc. of SPIE*, pages 86840A–86840A–10, 2013.

[78] NanGate. NanGate FreePDK45 Generic Open Cell Library. `http://www.si2.org/openeda.si2.org/projects/nangatelib`, 2012.

[79] NanGate. NanGate FreePDK15 Open Cell Library. `http://www.nangate.com/?page_id=2328`, 2014.

[80] Tim Nieberg. Gridless pin access in detailed routing. In *ACM/IEEE Design Automation Conference (DAC)*, pages 170–175, 2011.

[81] Muhammet Mustafa Ozdal. Detailed-routing algorithms for dense pin clusters in integrated circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 28(3):340–349, 2009.

[82] David Z. Pan, Lars Liebmann, Bei Yu, Xiaoqing Xu, and Yibo Lin. Pushing multiple patterning in sub-10nm: are we ready? In *ACM/IEEE Design Automation Conference (DAC)*, pages 197:1–197:6, 2015.

[83] David Z Pan, Bei Yu, and J-R Gao. Design for Manufacturing With Emerging Nanolithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 32(10):1453–1472, 2013.

[84] David Z Pan, Bei Yu, and Jhih-Rong Gao. Design for manufacturing with emerging nanolithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 32(10):1453–1472, 2013.

[85] Linyong Pang, Yong Liu, and Dan Abrams. Inverse lithography technology (ILT): a natural solution for model-based SRAF at 45nm and 32nm. In *Proc. of SPIE*, pages 660739–660739–10, 2007.

[86] Yang Ping, Sarah McGowan, Ying Gong, Yee Mei Foong, Jian Liu, Jianhong Qiu, Vincent Shu, Bo Yan, Jun Ye, Pengcheng Li, et al. Process window enhancement using advanced RET techniques for 20nm contact layer. In *Proc. of SPIE*, pages 90521N–90521N–10, 2014.

[87] Zhongdong Qi, Yici Cai, and Qiang Zhou. Accurate prediction of detailed routing congestion using supervised data learning. In *Proc. IEEE Int. Conf. on Computer Design (ICCD)*, pages 97–103. IEEE, 2014.

[88] Michael L Rieger and Victor Moroz. Self-aligned via interconnect using relaxed patterning exposure, August 19 2014. US Patent 8,813,012.

[89] Julien Ryckaert, Praveen Raghavan, Pieter Schuddinck, Huynh Bao Trong, Arindam Mallik, Sushil S Sakhare, Bharani Chava, Yasser Sherazi, Philippe Leray, Abdelkarim Mercha, et al. DTCO at N7 and beyond: patterning and electrical compromises and opportunities. In *Proc. of SPIE*, pages 94270C–94270C–8, 2015.

[90] Kyohei Sakajiri, Alexander Tritchkov, and Yuri Granik. Model-based SRAF insertion through pixel-based mask optimization at 32nm and beyond. In *Proc. of SPIE*, pages 702811–702811–12, 2008.

[91] Shumay Dou Shang, Lisa Swallow, and Yuri Granik. Model-based SRAF insertion, October 11 2011. US Patent 8,037,429.

[92] Syed Muhammad Yasser Sherazi, Bharani Chava, Peter Debacker, Marie Garcia Bardon, Pieter Schuddinck, Farshad Firouzi, Praveen Raghavan, Abdelkarim Mercha, Diederik Verkest, and Julien Ryckaert. Architectural strategies in standard-cell design for the 7 nm and beyond technology node. *Journal of Microlithography, Microfabrication and Microsystems*, 15(1):013507, 2016.

[93] Michael C Smayling, Koichiro Tsujita, Hidetami Yaegashi, Valery Axelrad, Tadashi Arai, Kenichi Oyama, and Arisa Hara. Sub-12nm optical lithography with 4x pitch division and SMO-lite. In *Proc. of SPIE*, pages 868305–868305–8, 2013.

[94] Yu-Hsuan Su and Yao-Wen Chang. Nanowire-aware routing considering high cut mask complexity. In *ACM/IEEE Design Automation Conference (DAC)*, pages 138:1–138:6, 2015.

[95] Yu-Hsuan Su, Yu-Chen Huang, Liang-Chun Tsai, Yao-Wen Chang, and Shayak Banerjee. Fast lithographic mask optimization considering process variation. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 230–237, 2014.

[96] Synopsys. Synopsys Design Compiler. `http://www.synopsys.com`, 2012.

[97] Taraneh Taghavi, Charles Alpert, Andrew Huber, Zhuo Li, Gi-Joon Nam, and Shyam Ramji. New placement prediction and mitigation techniques for local routing congestion. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 621–624, 2010.

[98] Xiaoping Tang and Minsik Cho. Optimal layout decomposition for double patterning technology. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 9–13, 2011.

[99] Haitong Tian, Hongbo Zhang, Qiang Ma, Zigang Xiao, and M.D.F. Wong. A polynomial time triple patterning algorithm for cell based row-structure layout. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 57–64, 2012.

[100] J. Andres Torres. ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2012.

[101] Ramya Viswanathan, Jaione Tirapu Azpiroz, and Punitha Selvam. Process optimization through model based SRAF printing prediction. In *Proc. of SPIE*, pages 83261A–83261A–10, 2012.

[102] Yan Wang, Ryoung-Han Kim, Lei Yuan, Anindarupa Chunder, Chenchen Wang, Jia Zeng, Youngtag Woo, and Jongwook Kye. Study of cut mask lithography options for sub-20nm metal routing. In *Proc. of SPIE*, pages 94260J–94260J–8, 2015.

[103] Zigang Xiao, Hongbo Zhang, Yuelin Du, and M. D. Wong. A Polynomial Time Exact Algorithm for Self-Aligned Double Patterning Layout Decomposition. In *ACM International Symposium on Physical Design (ISPD)*, 2012.

[104] Xiaoqing Xu, Brian Cline, Greg Yeric, and David Z Pan. Standard cell pin access and physical design in advanced lithography. In *Proc. of SPIE*, pages 97800P–97800P–13, 2016.

[105] Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Z Pan. Self-aligned double patterning aware pin access and standard cell layout co-optimization. In *ACM International Symposium on Physical Design (ISPD)*, pages 101–108, 2014.

[106] Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Z Pan. Self-aligned double patterning aware pin access and standard cell layout co-optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 34(5):699–712, 2015.

[107] Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Z Pan. A systematic framework for evaluating standard cell middle-of-line (mol) robustness for multiple patterning. In *Proc. of SPIE*, pages 942707–942707–14, 2015.

[108] Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Z Pan. Systematic framework for evaluating standard cell middle-of-line robustness for multiple patterning lithography. *Journal of Microlithography, Microfabrication and Microsystems*, 15(2):021202–021202, 2016.

[109] Xiaoqing Xu, Yibo Lin, Meng Li, Jiaojiao Ou, Brian Cline, and David Z. Pan. Redundant local-loop insertion for unidirectional routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2017.

[110] Xiaoqing Xu and David Z Pan. Toward unidirectional routing closure in

advanced technology nodes. *IPSJ Transactions on System LSI Design Methodology*, 10:2–12, 2017.

[111] Xiaoqing Xu, Bei Yu, Jhih-Rong Gao, Che-Lun Hsu, and David Z Pan. PARR: Pin access planning and regular routing for self-aligned double patterning. In *ACM/IEEE Design Automation Conference (DAC)*, pages 28:1–28:6, 2015.

[112] Xiaoqing Xu, Bei Yu, Jhih-Rong Gao, Che-Lun Hsu, and David Z Pan. PARR: Pin-access planning and regular routing for self-aligned double patterning. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 21(3):42, 2016.

[113] Yue Xu and Chris Chu. GREMA: graph reduction based efficient mask assignment for double patterning technology. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 601–606, 2009.

[114] Jae-Seok Yang, Katrina Lu, Minsik Cho, Kun Yuan, and David Z. Pan. A new graph-theoretic, multi-objective layout decomposition framework for double patterning lithography. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2010.

[115] Jun Ye, Yu Cao, and Hanying Feng. System and method for model-based sub-resolution assist feature generation, February 1 2011. US Patent 7,882,480.

[116] Wei Ye, Bei Yu, David Z. Pan, Yong-Chan Ban, and Lars Liebmann. Standard cell layout regularity and pin access optimization considering middle-of-line. In *ACM Great Lakes Symposium on VLSI (GLSVLSI)*, pages 289–294, 2015.

[117] Bei Yu, Jhih-Rong Gao, Duo Ding, Yongchan Ban, Jae-seok Yang, Kun Yuan, Minsik Cho, and David Z Pan. Dealing with IC Manufacturability in Extreme Scaling. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 240–242. ACM, 2012.

[118] Bei Yu, Yen-Hung Lin, Gerard Luk-Pat, Duo Ding, Kevin Lucas, and David Z. Pan. A high-performance triple patterning layout decomposer with balanced density. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 163–169, 2013.

[119] Bei Yu, Xiaoqing Xu, Subhendu Roy, Yibo Lin, Jiaojiao Ou, and David Z Pan. Design for manufacturability and reliability in extreme-scaling VLSI. *Science China Information Sciences*, pages 1–23, 2016.

[120] Bei Yu, Kun Yuan, Boyang Zhang, Duo Ding, and David Z. Pan. Layout decomposition for triple patterning lithography. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8, 2011.

[121] Y Yu, G Lin, IH Jiang, and Charles Chiang. Machine-learning-based hotspot detection using topological classification and critical feature ex-

traction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 34(3):460–470, 2015.

[122] Kun Yuan, Jae-Seok Yang, and David Z. Pan. Double patterning layout decomposition for simultaneous conflict and stitch minimization. In *ACM International Symposium on Physical Design (ISPD)*, pages 185–196, 2009.

[123] Hongbo Zhang, Yuelin Du, M. D. Wong, and Rasit Topaloglu. Self-Aligned Double Patterning Decomposition for Overlay Minimization and Hot Spot Detection. In *ACM/IEEE Design Automation Conference (DAC)*, 2011.

[124] Hongbo Zhang, Yuelin Du, Martin DF Wong, and Kai-Yuan Chao. Mask cost reduction with circuit performance consideration for self-aligned double patterning. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 787–792, 2011.

[125] Hongbo Zhang, Martin DF Wong, and Kai-Yuan Chao. On process-aware 1-D standard cell design. In *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pages 838–842, 2010.

[126] Yanheng Zhang and Chris Chu. RegularRoute: An efficient detailed router with regular routing patterns. In *ACM International Symposium on Physical Design (ISPD)*, pages 45–52, 2011.

Vita

Xiaoqing Xu received his Bachelor's degree in Microelectronics from Peking University, Beijing, China, in 2012. He started his Ph.D. program at the Department of Electrical and Computer Engineering, University of Texas at Austin in 2012, under the supervision of Professor David Z. Pan. He has interned at ARM Inc. in summer 2013 and 2014, Toshiba Corporation in summer 2015, IBM T. J. Watson Research Center in summer 2016 and Cadence Design Systems Inc. in fall 2016.

His research interests include robust standard cell design, design for manufacturability and physical design. His research has been recognized with numerous awards including Golden Medal at ACM Student Research Competition at ICCAD 2016, University Graduate Continuing Fellowship in 2016, SPIE BACUS Fellowship in 2016, Best in Session Award at SRC TECHCON 2015, William J. McCalla Best Paper Award at ICCAD 2013, CAD Contest Award at ICCAD 2013 and Microelectronics and Computer Development (MCD) Fellowship in 2012.

Permanent address: XiaoqingXu.pku@gmail.com

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.