分 类 号 O157　　　　　　　　　　　　　　　　　密级 公开
收藏编号　　　　　　　　　　　　　　　　　　　　学号 155410004
学校代码 10386　　　　　　　　　　　　　　　　　编号

福州大学

博士研究生学位（毕业）论文

集成电路制造设计中的布局分解问题研究

学 科 专 业：应用数学

研 究 方 向：优化理论与算法

研 究 生 姓 名：李兴权

指 导 教 师、职 称：朱文兴　教授

协 助 导 师、职 称：

所 在 学 院：离散数学与理论计算机科学研究中心

答辩委员会主席签名：

二〇一八 年 六 月

一　遵守学术行为规范承诺

本人已熟知并愿意自觉遵守《福州大学研究生和导师学术行为规范实施办法》和《福州大学关于加强研究生毕业与学位论文质量管理的规定》的所有内容，承诺所提交的毕业和学位论文是终稿，不存在学术造假或学术不端行为，且论文的纸质版与电子版内容完全一致。

二　独创性声明

本人声明所提交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得福州大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。本人完全意识到本声明的法律结果由本人承担。

三　关于论文使用授权的说明

本人完全了解福州大学有关保留使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。（保密的论文在解密后应遵守此规定）

本学位论文属于（必须在以下相应方框内打“√”，否则一律按“ 非保密论文”处理）：

1、保密论文：　□本学位论文属于保密，在____ 年解密后适用本授权书。

2、非保密论文：□本学位论文不属于保密范围，适用本授权书。

研究生本人签名：__________________　　　签字日期：20　　年　　月　　日

研究生导师签名：__________________　　　签字日期：20　　年　　月　　日

集成电路制造设计中的布局分解问题研究

摘　　要

对于半导体工业，光刻技术一直是用于制造更可靠，更小巧，更快速的集成电路芯片的关键技术。近年来，晶体管特征尺寸的不断缩小和芯片集成度的迅速提高给芯片的制造带来了巨大的挑战。在16纳米技术节点，主流的193纳米级ArF浸入式光刻技术达到了它制造能力的极限。于是，为了弥合光刻技术制造能力和芯片性能预期之间的差距，从业人员提出了许多新兴制造技术来维持半导体工业界继续前进。本文重点讨论三种最有前景的新兴制造技术，即多重图样光刻技术，电子束光刻技术和嵌段共聚物定向自组装技术，及其在物理设计（包括布局和布线）阶段的友好设计，对这些技术所面临的关键问题进行深入研究，并针对性的提出计算机辅助设计算法的解决方案。主要内容如下：

第一章阐述几种新兴的制造技术和面临的关键挑战，以及本文的主要贡献。

第二章考虑集成电路制造设计中的三重光刻版图分解问题。三重光刻技术是目前应用最广泛的技术，版图分解是其中最核心的问题。已有研究证明了三重光刻版图分解是强NP 完全问题，这使得在超大规模情况下该问题的解决是极具挑战性的。已有的研究多数采用启发式或半定规划松弛方法，这些方法要么得到的结果质量不高，要么求解时间太长。为了能够在可接受的时间内获得较好的版图分解结果，我们针对三重光刻版图分解提出一套离散松弛理论和框架。首先，我们提出一个面投影技术来识别版图中的局部冲突，并且构造出对应的冲突图。在离散松弛理论的支持下，我们采用一系列的图缩减技术来减小冲突图的规模。然后，通过对冲突边赋权来考虑缝合插入，版图分解问题被松弛成一个较简单的整数规划问题。我们采用分支定界算法来求解这个整数规划，可以得到一个原问题的松弛解。最后，我们提出了一个有效的缝合插入算法，从而将松弛解合法化成正确的版图分解结果。为了得到更好的解，我们还设计了一个回溯染色算法。在ISCAS-85&89实例上的测试和实验比较表明，我们的算法可以得到更好的结果。另外，在离散松弛理论保证下，我们可以从计算上衡量计算结果的质量。特别地，对许多测试实例，我们的方法可以得到最优解。本文所提出的离散松弛可以用于解决许多其他NP困难的组合优化问题，与传统的连续松弛不一样，它具有更广泛的适用性和更专门的针对性。

混合电子束和三重光刻技术是未来集成电路制造工业的关键技术，它们有效地结合了三重光刻的高产量和电子束光刻的高分辨率优势，其中最核心的问题是考虑高产量和高分辨率的版图分解问题。针对一般的电路，已有的研究首先将版图中每个图样分割成多个子图样，然后采用局部搜索算法来得到启发式的结果。显然，这种方法解空间非常大，并且局部搜索算法得到结果的质量非常依赖于算法执行时间。在第三章中，首先，我们证明了最大化产量和分辨率的版图分解问题是NP 困难的。然后，我们提出一个两阶段方法来快速求解混合电子束和三重光刻图分解问题。在第一阶段，我们构造两个组合优化问题：1）考虑电子束和缝合的三重光刻掩膜版分配问题和扩展的最小权集支配剩余集合掩膜版分配问题。同时，我们相应地构造了这两个问题的0-1 规划问题，并且采

用割平面算法求解0-1规划问题。在第二阶段，通过缝合插入和电子束引入，我们把第一阶段得到的结果合法化成原问题的解。这种两阶段操作可以大大减少求解时间。另外，为了进一步加速我们的分解方法，在分解之前，我们移除一些点和非关键边。实验结果比较说明了我们方法的效率和有效性。

作为多重图样光刻技术的辅助技术，嵌段共聚物自组装技术是低成本的，它特别适合用于通孔层的制造。在第四章中，我们采用自组装和三重图样光刻技术制造通孔层掩膜版和引导槽的分配问题。掩膜版和引导槽的合理安排对提高结合自组装和三重图样光刻技术的产量至关重要。目前的算法主要是基于查找表的技术。由于查找表的规模没办法枚举所有可能情况（特别是对于大规模例子），因此基于查找表的方法所得到的结果的质量是很有限的，且无法评估计算结果的质量。为了解决这些问题，首先，通过引入负权边，我们构造一个冲突组合图。然后，根据冲突组合图，我们设计一个离散松弛问题，并给出对应的0-1规划问题。通过分支定界算法求解这个0-1规划，我们可以得到原问题解的一个下界。为了提高这个下界，我们引入一些有效不等式来剪掉一些不好的松弛解。最后，通过在布局图上求解一个引导槽分配问题，我们把得到的松弛解转化成原问题的合法解。这个合法的解是原问题的一个上界。实验结果比较说明了我们方法的有效性。另外，在离散松弛理论下，我们可以根据上下界来衡量实验结果的质量。平均地，我们得到的上界和下界之间的差距只有0.4%，说明我们的结果很接近最优解。特别地，对很多例子，我们得到的上界和下界之间的差距是0，说明我们的算法得到了最优解。

为了保证通孔连接层的稳定性，为每个通孔插入一个额外的通孔是必要的。已有的研究表明对于通孔层的制造，混合嵌段共聚物定向自组装和多重图样光刻技术是最合适的。在第五章，我们研究定向自组装和多重图样光刻技术下的额外通孔插入和引导槽分配问题。由于问题的复杂性，目前的研究主要是将问题描述成多变量，多约束的整数规划问题。显然，对于大规模的实例，求解很多变量和约束的整数线性规划问题是不实际的。为了快速得到高质量的结果，我们提出一个基于图论的算法框架：首先，我们通过研究通孔和额外通孔的特性，采用某些特定的多元组来表示可能的解，而不是直接用可能的引导槽分配来表示。把几种多元组看成点，我们可以构造一个非常紧致的冲突图；然后，基于冲突图，我们把单重图样光刻问题描述成一个约束最大权独立集问题；最后，为了快速求解大规模最大权独立集问题，我们提出一个多项式时间局部收敛的数值迭代算法。另外，对于二重，三重图样光刻情形，通过最大K割算法，它们可以归约到单重框架可处理的问题。实验结果验证了我们方法的有效性，特别地，我们的方法用于求解单重，二重，三重图样光刻问题的速度分别比当前最好的工作快2.38倍，5.79倍和27.82倍。

在第六章中，为了进一步提高额外通孔插入率和通孔可制造率，我们在后布线阶段考虑带虚拟通孔的额外通孔插入问题。首先，通过分析引导槽的结构性质，我们提出一个基于字典的解表示方法。根据这个紧致的解表示，我们构造一个带虚拟通孔的冲突图。然后，我们把这个问题描述成一个最大权独立集

问题。为了在运行时间和解的质量之间获得一个更好的权衡，我们将最大权独立集问题松弛成一个无约束非线性规划问题。最后，本文提出一个基于线搜索的求解算法，并证明该算法可以在多项式时间内得到局部最优解。另外，为了让这个算法获得更好的效果，我们提出一个有效的初始解生成算法。实验比较验证了我们的方法是有效的。与当前最好的工作相比，我们的方法在保证解的质量基本相同的情况下可以节约94% 的运行时间。

为了满足低功耗和高效率电路设计的需要，先进设计技术下的电路通常包含多种不同高度的单元，这种设计给布局合法化问题带来了新的挑战。另外，为了满足设计和制造的需要，一个好的合法化工具要考虑火线/地线对齐，边缘间距，引脚接入/短路等等要求。已有的混合不同高度单元合法化问题的研究只考虑单元总移动量最小，并未考虑单元最大移动量最小。另外，已有的研究在处理单元散开时都是按照竖直和水平方向分开进行的，这显然不是一个全局的单元散开操作。在第七章中，我们提出一个同时考虑最大移动量和总移动量最小的合法化算法。首先，我们将问题描述成一个混合整数二次规划问题，此模型允许单元同时在竖直和水平方向散开。然后，通过松弛，把问题转化成二次规划问题。接着，根据KKT条件，把二次规划描述成等价的互补松弛问题，它可以通过基于模的矩阵分裂迭代方法进行求解。最后，为了获得最小单元移动的合法解，我们针对性地设计线性规划算法和Kuhn-Munkres算法。实验结果表明我们的方法在满足所有约束的情况下，可以同时减小单元最大移动量和平均移动量。

关键词： VLSI**设计自动化，新兴制造技术，版图分解，计算机辅助设计，优化算法。**

Layout Decomposition Design for Manufacturability

Abstract

Lithography has been and will continue to be the backbone of the semiconductor industry to design reliable, smaller, and faster integrated circuit chips. In recent years, the continuous shrinkage of the transistor feature size and the rapid increase of integration density have imposed severe challenges on design and manufacturing closures. At the 16nm technology node, the mainstream 193nm ArF immersion lithography has reached its limit of manufacturing capability. Thus, in order to bridge the gap between the manufacturing capability of lithography and expected chip performance, a variety of emerging technologies have been proposed to enable the industry to keep the pace of Moore's law. In this thesis, we will focus on multiple patterning lithography, e-beam lithography and block copolymer directed self-assembly lithography, which are among the most promising technologies.

As all these emerging lithography technologies are facing different challenges, more effort in research is required. In this thesis, various problems faced by these technologies are investigated and corresponding solutions with computer-aided design (CAD) algorithm are presented. The content of this thesis is organized as follows.

In Chapter 1, several promising manufacture technologies and the key challenges, and the corresponding challenges are introduced.

In Chapter 2, we consider the triple patterning lithography layout decomposition problem in IC manufacturing design. Triple patterning lithography is currently the most widely used technology, and in which layout decomposition is the most crucial challenge. Previous work proved that the triple patterning lithography layout decomposition is a strong NP complete problem, which is extremely challenging for the very large scale case. Most of the existing studies

use heuristic or semi-definite programming relaxation methods, these approaches are either low quality or time-consuming. To obtain better layout decomposition results in desirable runtime, we design a discrete relaxation theory and framework for triple patterning lithography layout decomposition. First, we propose a surface projection method for identifying native conflicts, and then construct a conflict graph. Guided by the theory, the conflict graph is reduced to small size subgraphs by several graph reductions, which is a discrete relaxation. Furthermore, by ignoring stitch insertions and assigning weights to features, the layout decomposition problem on the subgraphs is further relaxed to a integer programming, which is solved by the Branch-and-Bound method. To obtain a feasible solution of the original problem, legalization methods are introduced to legalize a relaxation solution. At the legalization stage, we prior utilize one-stitch insertion to eliminate conflicts, and use a backtrack coloring algorithm to obtain a better solution. We test our decomposition approach on the ISCAS-85 & 89 benchmarks. Experimental comparisons show that our approach achieves better results than those by the state-of-the-art decomposers. Especially, according to our discrete relaxation theory, some optimal decompositions are obtained. It must be noted that the discrete relaxation framework proposed in this thesis can be used to address many other NP-hard discrete problems.

Hybrid e-beam lithography (EBL) and triple patterning lithography (TPL) is an advanced technology for manufacture of integrated circuit. This technology combines the advantages of EBL and TPL, which is more promising for further pattern product industry. Yield and resolution aware layout decomposition is the most crucial step in this technology. For the general layout, existing research first splits each feature into several sub-features, and then uses a local research algorithm to achieve heuristic results. Apparently, the solution space in existing work is very large, and the quality of the results by local search algorithm is quite dependent on the runtime. In Chapter 3, first, we prove that the yield and resolution aware layout decomposition problem is NP-hard. And then,

we propose a two stage decomposition flow for the hybrid e-beam and triple patterning lithography of general layout decomposition (HETLD) problem. At the first stage, we formulate two optimization problems: the e-beam and stitch aware TPL mask assignment (ESTMA) problem and the extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) problem. Binary linear program formulations of the two problems are formulated and solved by the cutting plane approach. At the second stage, solutions of the first stage problems are legalized to feasible solutions of the HETLD problem by stitch insertion and e-beam shot. This two stage operation can greatly reduce the runtime. And to further speed up decomposition, we reduce the problem size by removing some vertices and some minor conflict edges before decomposition. Experimental results show the effectiveness of our decomposition methods based on ESTMA and MDSR_4MA.

Block copolymer directed self-assembly (DSA) is a simple and promising candidate next-generation chip manufacturing technology. AS a complement technology of multi-patterning, DSA is low-cost and suitable for patterning contact layer. In Chapter 4, we consider the contact layer mask and template assignment problem of DSA with triple patterning lithography. A desirable mask and template assignment is significant for the high yield of DSA with multi-patterning technology. Existing algorithm is based on look-up-table (LUT). For large scale cases, since LUT can not list all the possible mask and template assignments, the quality of obtained results is quite limited. To address this issue, we design a discrete relaxation method for this mask and template assignment problem. First, we construct a weighted conflict grouping graph, in which edges with negative weights are introduced, then a discrete relaxation based mask assignment problem is proposed. The integer linear program formulation of the discrete relaxation problem is solved by Branch-and-Bound method for obtaining a lower bound on the optimal value of this problem. In order to improve the lower bound, some valid inequalities are introduced to prune some poor relax-

ation solutions. At last, the obtained discrete relaxation solution is transformed to a legal solution of the original problem by solving a template assignment problem on the layout graph, which provides an upper bound on the optimal value of the original problem. Experimental results and comparisons show the effectiveness and efficiency of our method. In addition, under the discrete relaxation theory, the quality of our experimental results can be evaluated by the obtained upper and lower bounds. Specifically, the gap between the obtained upper and lower bounds is 0 for most of the sparse benchmarks, and the average gap is 0.4% for dense benchmarks.

To improve via yield in circuit designs, inserting a redundant via for every via is necessary. Block copolymer directed self-assembly (DSA) is an emerging and promising lithography technology for manufacture of vias and redundant vias, in which guiding templates are used to enhance the resolution. Considering manufacturability of via layer, multiple patterning lithography is also needed in advanced designs. In Chapter 5, we study the redundant via insertion and guiding template assignment for DSA with multiple patterning problem at the post-routing stage. Due to the complexity of the problem, the current studies focus on formulating the problem as integer linear programming (ILP) with many variables and constraints. Obviously, for large scale instances, it is not practical to obtain a result by solving ILP. In order to fast achieve a high quality result, we propose a graph methodology based solution framework: Firstly, a conflict graph on the grid model is constructed. Secondly, the problem with single patterning is formulated as a constrained maximum weight independent set problem, for which a fast algorithm is introduced to obtain a local optimal solution. To further improve the performance, a greedy method is proposed to search for a good initial solution. Our framework is general and can be further extended to solve the problem with double patterning or triple patterning in a two stage manner. Experimental results validate the efficiency and the effectiveness of our method. Specifically, our method is 2.38×, 5.79× and 27.82× faster

than state-of-the-art for the problem with single, double and triple patterning, respectively.

For better reliability and manufacturability, in Chapter 6, we consider the redundant via insertion and DSA guiding template assignment with dummy via insertion at the post-routing stage. Firstly, by analyzing the structure property of guiding templates, we propose a dictionary-based solution expression to discard redundant solutions. Then, honoring the compact solution expression, we construct a conflict graph with dummy via insertion, and then formulate the problem as a constrained maximum weight independent set problem (CMWIS). Furthermore, the CMWIS problem is reformulated as an integer linear programming (ILP). To make a good tradeoff between solution quality and runtime, we relax the ILP to an unconstrained nonlinear programming (UNP) by a three dimensional tensor. Finally, a line search optimization algorithm is proposed to solve the UNP. Experimental results verify the efficiency and effectiveness of our proposed algorithm. Specifically, our algorithm achieves experimental results comparable with a state-of-the-art work, and saves 90% runtime.

Modern circuit designs often contain standard cells of different row heights to meet various design requirements such as low power and high performance. Due to the higher interference among heterogenous cell structures, the legalization problem for mixed-cell-height standard cells becomes more challenging. In addition, to meet the needs of design and manufacturibility, an ideal legalization tool should concern total cell movement, maximum cell movement, VDD/VSS alignment, edge spacing, pin access/short, etc. The existing mixed-cell-height standard cell legalization algorithms ignore the maximum cell movement. In addition, the existing studies eliminate cell overlap in the horizontal direction, which lacks a global view. In Chapter 7, we present an analytical legalization algorithm for mixed-cell-height standard cells to simultaneously minimize the average and the maximum cell movements. By analyzing and remodeling the legalization problem, we first formulate it as a mixed integer quadratic pro-

gramming problem (MIQP), which allows cell spreading concurrently in both the horizontal and vertical directions. By relaxing its discrete constraints to linear ones, we convert the MIQP into a quadratic programming problem (QP). To solve the QP efficiently, we further reformulate it as a linear complementary problem (LCP), and solve the LCP by a modulus-based matrix splitting iteration method (MMSIM). To guarantee the convergence of the MMSIM, we use a series of operations to ensure that its induced objective matrix is symmetric positive definite and its constraint matrix is of full row rank. Finally, a linear programming based method and the Kuhn-Munkres algorithm are used to legalize cells with least movements. Experimental results demonstrate the effectiveness of our algorithm in reducing both the average and the maximum cell movements for mixed-cell-height legalization.

Keywords: VLSI design automation, emerging manufacture technology, layout decomposition, computer-aided design, optimization algorithm.

Contents

中文摘要 i

Abstract v

Contents xvi

List of Figures xxi

List of Tables xxiv

Chapter 1 Introduction 1

1.1 Introduction to Lithography Technology 1

1.2 Challenges for Emerging Lithography Technology 8

1.2.1 Challenge for Multiple Patterning Lithography 8

1.2.2 Challenge for E-beam Lithography 9

1.2.3 Challenge for Block Copolymer Directed Self-Assembly . 10

1.2.4 Challenge for Redundant Via Insertion by DSA with Multiple Patterning 10

1.2.5 Challenge for Multi-Deck Standard Cell Legalization . . . 11

1.3 Contribution of This Thesis 12

1.4 Organization of This Thesis 16

Chapter 2 Discrete Relaxation Method for Triple Patterning Lithography Layout Decomposition 19

2.1 Introduction 19

2.2 Problem Formulation, Discrete Relaxation Theory, and Layout Decomposition Framework 23

2.2.1 Problem Formulation 23
2.2.2 Discrete Relaxation Theory 24
2.2.3 Overview of Layout Decomposition Framework 26
2.3 Surface Projection and K_4 Conflict Structure 27
2.3.1 Surface Projection 27
2.3.2 Conflict Feature and K_4 Conflict Structure 29
2.4 Discrete Relaxation Method for TPL Layout Decomposition . . 31
2.4.1 Equivalent Formulation of Problem (P_0) to (P_1) 32
2.4.2 Relaxation of Problem (P_1) to (P_2) 33
2.5 Legalization 40
2.5.1 Stitch Insertion 41
2.5.2 Backtrack Coloring 43
2.6 Relaxation Via Graph Reduction 45
2.7 Experimental Results 48
2.7.1 First Experiment 49
2.7.2 Second Experiment 55
2.8 Summary 56

Chapter 3 Two-Stage Decomposition for Hybrid E-Beam and Triple Patterning Lithography **59**

3.1 Introduction 59
3.2 Preliminaries 63
3.2.1 Hybrid E-Beam and TPL Layout Decomposition Problem 63
3.2.2 Conflict Pattern and Native Conflict 65
3.3 Hybrid E-Beam and TPL Mask Assignment Methods 68
3.3.1 E-Beam and Stitch Aware TPL Mask Assignment (ESTMA) 69

3.3.2 Extended Minimum Weight Dominating Set for R_4 Mask Assignment (MDSR_4MA) 71
3.4 Legalization by Stitch Insertion, E-Beam Shot and Backtrack Coloring 76
3.4.1 Conflict Elimination 76
3.4.2 Assignment of Patterns in R_4 79
3.4.3 Backtrack Coloring 79
3.5 Graph Reduction and Decomposition Flow 81
3.5.1 Graph Reduction 81
3.5.2 Flow for Hybrid EBL and TPL Layout Decomposition . . 84
3.5.3 An Example of The Two Stage Decomposition Algorithm 84
3.6 Experimental Results 87
3.6.1 Statistics and Analysis 87
3.6.2 Comparisons 92
3.7 Summary 95

Chapter 4 Discrete Relaxation Method for Contact Layer Decomposition of DSA with Triple Patterning 97

4.1 Introduction 97
4.2 Preliminaries 101
4.2.1 DSA Guiding Template 101
4.2.2 Problem Formulation 104
4.3 Discrete Relaxation Method for Mask and Template Assignment of DSA with TPL 105
4.3.1 Conflict Grouping Graph Construction 106
4.3.2 Discrete Relaxation Based Mask Assignment 107
4.3.3 Improving the Lower Bound by Adding Valid Inequalities 113

4.4 Template Assignment 119
4.5 Experimental Results 121
4.5.1 First Experiment 122
4.5.2 Second Experiment 127
4.6 Summary 128

Chapter 5 Graph Based Redundant Via Insertion and Guiding Template Assignment for DSA-MP 131

5.1 Introduction 131
5.2 Preliminaries 135
5.2.1 Problem Formulation 135
5.2.2 Solution Flow 137
5.3 Conflict Graph Construction on Grid 138
5.4 Algorithms for RGDS Problem 143
5.4.1 Constrained Maximum Weight Independent Set Problem 143
5.4.2 A Fast Algorithm for the MWIS Problem 147
5.4.3 Reduction of CMWIS to MWIS 151
5.5 Algorithms for RGDD/RGDT Problems 154
5.5.1 Mask Assignment for RGDD/RGDT 154
5.5.2 Solving the RGDS Problem on Every Mask and Legalization 158
5.6 Experimental Results 159
5.6.1 First Experiment 160
5.6.2 Second Experiment 164
5.7 Summary 169

Chapter 6 Redundant Via Insertion and DSA Guiding Template Assignment with Dummy Via 171

6.1 Introduction 171
6.2 Preliminaries 174
6.2.1 Redundant Via Insertion 174
6.2.2 Guiding Template Assignment 175
6.2.3 Dummy Via Insertion 175
6.2.4 Problem Formulation 177
6.3 Conflict Graph Construction 178
6.4 Our Algorithms 180
6.4.1 Constrained Maximum Weight Independent Set Problem 181
6.4.2 A Fast Algorithm for The CMWIS Problem 182
6.5 Experimental Results 189
6.5.1 Effectiveness of ILP 189
6.5.2 Effectiveness of Local Optimal Algorithm 191
6.6 Summary 192

Chapter 7 Analytical Mixed-Cell-Height Legalization Considering Average and Maximum Movement Minimization 195

7.1 Introduction 195
7.2 Problem Statement 198
7.3 Problem Reformulations 200
7.3.1 Mixed Integer Quadratic Programming 200
7.3.2 Quadratic Programming 202
7.3.3 Linear Complementarity Problem 207
7.4 Our Legalization Framework 211
7.5 Experimental Results 216
7.6 Summary 219

Chapter 8 Conclusions and Future Works **221**

8.1 Conclusions 221

8.2 Future Works 225

Bibliography **227**

致　　谢 **239**

攻读博士学位期间发表的论文和成果 **243**

List of Figures

1.1 Conventional optical lithography system. 2

1.2 Lithography wavelength. 2

1.3 193nm ArF lithography technology up to its limit. 3

1.4 Roadmap for transistor, interconnect and patterning. 4

1.5 Triple patterning lithography process. 5

1.6 E-beam system and process. 6

1.7 Directed self-assembly process and guiding template. 7

1.8 Extreme ultraviolet lithography system. 8

1.9 Design flow and challenges. 9

2.1 TPL layout decomposition. (a) An example of three masks assignment. (b) An example of layout with a conflict. (c) An example of layout with inserting a stitch to eliminate conflict. 20

2.2 A sample of legal stitch insertion. 24

2.3 Geometrical representation of discrete relaxation. 25

2.4 Our TPL layout decomposition framework. 26

2.5 Surface projection and line projection. (a) Line projection. (b) Surface projection. (c) An example shows that line projection cannot find inside *TCRT*. (d) Surface projection is used to find *TCRT*. 28

2.6 Conflict features and a K_4 conflict structure. 30

2.7 Two samples of backtrack coloring. (a)(c) Initial illegal coloring solutions. (b)(d) Feasible solutions after backtracking. 46

2.8 An example of vertex removal. (a) The initial graph. (b) The remainder graph after removing contained vertex g. 47

2.9 Running time vs the number of vertices. 55

3.1 Hybrid e-beam and triple patterning lithography layout decomposition. (a) A mask assignment for triple patterning lithography layout decomposition with a conflict. (b) An example of layout with stitch insertion for eliminating the conflict. (c) A mask assignment for TPL layout decomposition with a conflict. (d) A mask assignment for HETLD. 60

3.2 An illegal mask assignment for HETLD. 64

3.3 Conflict region and conflict patterns. (a) Conflict region. (b) Conflict pattern. (c) An example of non-conflict pattern. (d) K_4 conflict structure. (e)(f) Native conflict structure. 67

3.4 A comparison of the ESTMA problem and the MDSR_4MA problem. (a) A layout where all patterns are CP. (b) A feasible solution of the ESTMA problem. (c) The decomposition result of (b)(g). (d) A feasible solution of the ESTMA problem. (e) The decomposition result of (d)(h). (f) The set of conflict adjacent edges E_{ca}. (g)(h) Two feasible solutions of the MDSR_4MA problem. 72

3.5 Our HETLD decomposition flow. 85

3.6 A sample of HETLD flow. (a) Initial layout. (b) Conflict graph. (c) Conflict pattern identification. (d) Relaxed conflict graph. (e) Solution of problem (3.2) or (3.4). (f) Feasible solution of HETLD. 86

3.7 Decomposed layout for benchmark C880 with $\min_{cs}$=160nm. . . 91

4.1 An example of contact layer decomposition for DSA with TPL. (a) A template assignment for the sparse layout. (b) A dense layout. (c) A mask and template assignment for the dense layout. 98

4.2 Template types. (a)-(d) Available vertical and horizontal templates. (e)-(g) Illegal templates. 101

4.3 Comparison of the costs of different templates. (a) Rule 1). (b) Rule 2). (c) Rule 3). 102

4.4 Conflict spacing and grouping spacing. 105

4.5 Conflict grouping graph construction. 107

4.6 Total edge costs of different templates. 109

4.7 Corner incompatibility and triangle edges. (a) Weighted conflict grouping graph. (b) A solution obtained by solving problem P_2. (c) New weighted conflict grouping graph. (d) A solution obtained by solving problem P_2^+. 113

4.8 Structures with corner incompatibility. (a) "L" shape. (b) "T" shape. (c) "Z" shape. (d) "+" shape. 115

4.9 Template assignment for layout graph. (a) A layout graph. (b)(c) Two template assignment results. 120

4.10 T_2 template only assignment for the layout graph in Figure 4.9(a). 123

4.11 Mask and template assignment for benchmark dp1_Via1 with $d_c = 51nm$ and $d_{g_{max}} = 40nm$. 127

5.1 An example of redundant via insertion. (a) Four locations of redundant via candidates. (b) A feasible redundant via insertion result. 131

5.2 Four usable types of guiding templates. (a) t_1. (b) t_2. (c) t_3. (d) t_4. 132

5.3 Example for the RGDS and the RGDM problems. (a) A layout with a via layer and two metal layers. (b) A result of the RGDS problem. (c) A result of the RGDD problem. 137

5.4 Our flow of graph based method. 138

5.5 All redundant via candidates of the layout in Figure 5.3(a). . . . 139

5.6 (a) The S_1 and D_1s of via v_4. (b) D_2. 140

5.7 All possible combinations of *multiplet*s to form four guiding template types. 141

5.8 The conflict graph CG for vias v_1, v_2, v_3 and v_4 of the layout in Figure 5.3(a). 142

5.9 The graph constructed based on GTA for vias v_1 and v_2 of the layout in Figure 5.3(a). 142

5.10 Two kinds of incompatibility structures. 146

5.11 An optimal solution of the MWIS problem on the conflict graph in Figure 5.8. 151

5.12 (a) A CG with seven *multiplet*s. (b) An IG with two INCs $I_i(i,\ j,\ k)$ and $I_j(j,\ k,\ l)$. (c) A TG and its maximum weight matching result. 153

5.13 (a) Process of contraction graph construction. (b) Contraction graph of the conflict graph in Figure 5.8. (c) Result of the max-2-cut problem on the contraction graph in (b). 155

5.14 Three more usable types of guiding templates. (a) t_5. (b) t_6. (c) t_7. 164

5.15 (a) D_3. (b) (c) All possible combinations of *multiplet*s to form guiding templates t_5 and t_6. (d) sextet. 165

5.16 A result of the RGDS problem on benchmark `s9234`. 169

6.1 (a) A via and four possible positions of its RVCs. (b) Legal RVCs. (c) A redundant via insertion and guiding template assignment result. 172

6.2 Seven usable types of guiding templates. 174

6.3 (a) A layout. (b) Redundant via insertion and guiding template assignment. (c) An irregular guiding template. (d) A result with dummy via insertion. 176

6.4 *Multiplet*s. 178

6.5 All possible combinations of *multiplet*s to form the seven types of guiding templates. 179

6.6 (a) A layout after finding RVC and DVC. (b) All *multiplet*s of the layout in (a). (c) Conflict graph. 180

6.7 Three kinds of incompatibility structures. 183

6.8 Sigmoid function $\sigma(y_i)$ with different γ. 184

6.9 The result of benchmark s9234 and its a partial layout on vias between metal 0 and metal 1. 192

6.10 The result of benchmark primary2 and its a partial layout on vias between metal 0 and metal 1. 192

7.1 Example of the VDD/VSS alignment constraints. 196

7.2 Comparisons on legalization with average cell movement and that with simultaneous average and maximum cell movements. 197

7.3 Cells are aligned to the nearest correct rows. 202

7.4 (a). Vertical moving intervals of sub-cells. (b). The moving ranges of bottom-left y-coordinate of sub-cells. 204

7.5 Four possible structures for two adjacent sub-cells i and j with $VMI_i \cap VMI_j \neq \emptyset$. 205

7.6 Our legalization framework. 212

7.7 (a) Legalization result of the benchmark "fft_2_md2" from our algorithm. Cells are in blue, and movement in green. (b) A partial layout of (a). 219

List of Tables

2.1 Test results of our decomposition method with $\min_{cs} = 120 / 100nm$ 49

2.2 Experimental result comparisons with $\min_{cs} = 120 / 100nm$ . . 50

2.3 Experimental result comparisons on dense layouts with $\min_{cs} = 120nm$ 53

2.4 Experimental result comparisons on dense layouts with $\min_{cs} = 160nm$ 54

3.1 Statistics of hybrid e-beam and TPL layout decomposition benchmarks with $\min_{cs} = 160nm$ 88

3.2 Comparison of decomposition results with less gap and larger gap for the ILP of the ESTMA based decomposition, $\min_{cs} = 160nm$ 90

3.3 Comparison results of the four HETLD decomposers, $\min_{cs} = 160nm$ 93

4.1 Statistics of benchmarks for mask and template assignment for DSA with TPL 123

4.2 Comparison results of mask and template assignment for DSA with TPL, $d_c = 41nm$, $d_{g_{min}} = 10nm$, $d_{g_{max}} = 30nm$ 124

4.3 Comparison results of mask and template assignment for DSA with TPL, $d_c = 51nm$, $d_{g_{min}} = 10nm$, $d_{g_{max}} = 40nm$ 125

5.1 Comparison with the ISPD 2016 work on the RGDS problem . . 162

5.2 Comparison with the ISPD 2016 work on the RGDD problem . 163

5.3 Comparison with the ISPD 2016 work on the RGDT problem . 163

5.4 Comparison with the ILP in TCAD 2017 work on the RGDS problem 166

5.5 Comparison with the graph method in TCAD 2017 work on the RGDS problem 167

5.6 Comparison with the TCAD 2017 work on the RGDS problem . 168

6.1 Comparison of computational results of three methods for the RDD problem 190

7.1 Experimental results. 218

Chapter 1 Introduction

In semiconductor industry, 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 lithography tool, i.e., $193nm$ ArF immersion lithography. To keep the high growth of very large scale integrated (VLSI) circuit, a variety of emerging technologies have been designed to enable the manufacture of VLSI circuit. Computer-aided design (CAD) optimization is a crucial step for driving these emerging technologies to fit the manufacture of advanced VLSI circuit. The most existing problems in VLSI design for manufacturibility are large-scale and NP-hard complexity. More optimization theories and effective algorithms are required to achieve desirable performance and throughput of chips.

1.1 Introduction to Lithography Technology

The fabrication of ICs involves a series of physical and chemical processes, among which lithography is one of the most important step. Servers, computers, mobile phones, and other digital appliances, which are now indispensable parts of modern societies, are all impossible without the continuous advancement of the lithography technologies.

Conventional lithography relies on light for manufacturing, thus it is called photolithography or optical lithography. It is similar to photographic printing. A conventional optical lithography system is sketchily shown in Figure 1.1. There are four major components: light source, mask, projection lens and photoresist-coated wafer consisting of a number of dies. Through the mask and the projection lens, light images of extremely small features (down to several

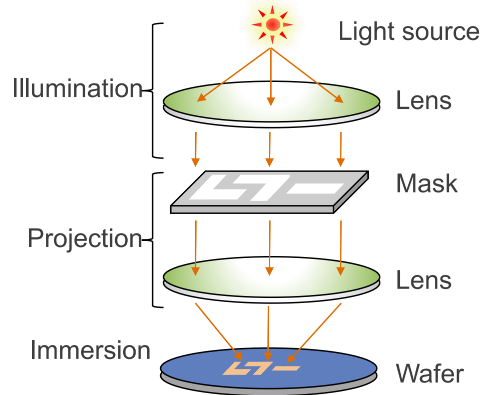


Figure 1.1: Conventional optical lithography system.

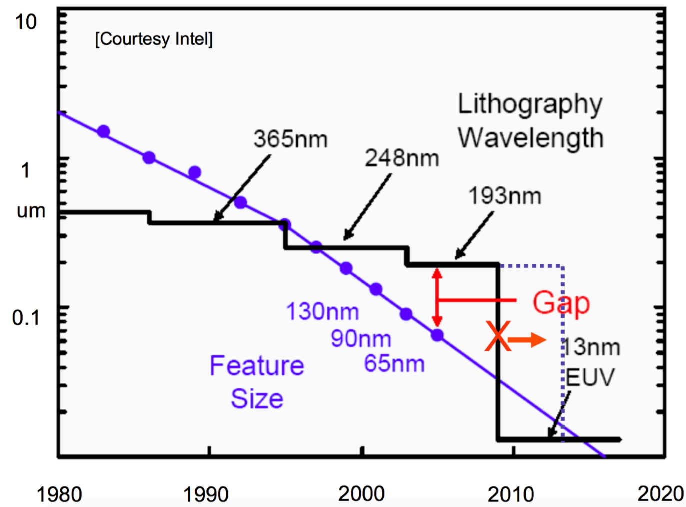


Figure 1.2: Lithography wavelength.

nanometers in size) will be created on the photoresist on the wafer surface. The step after lithography is etching, which is to use chemicals to create features on the wafer based on the light images on the photoresist. The lithography process and the etching process, denoted as litho-etch process, are the key steps for IC manufacturing.

In the current mainstream optical lithography technology, the light source is ArF deep ultraviolet with a wavelength of 193nm. The minimum resolution achievable with optical lithography is linear to the light wavelength. According

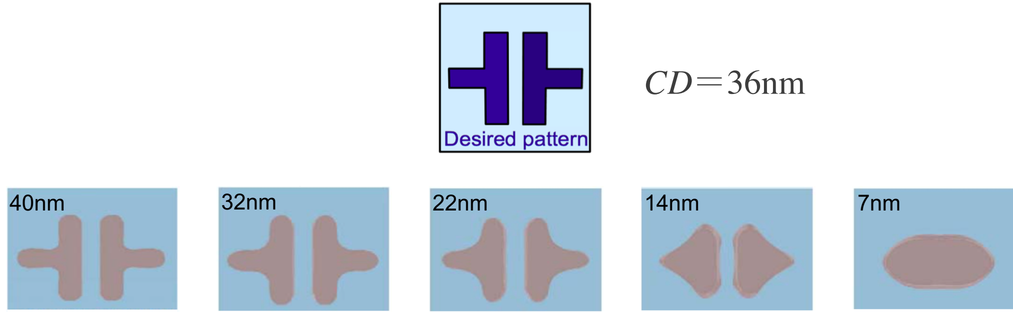


Figure 1.3: 193nm ArF lithography technology up to its limit.

to [65], the critical dimension (the minimum feature size) CD equals to

$$CD = \frac{k_1 \cdot \lambda}{NA}, \tag{1.1}$$

where k_1 is the process factor, λ is the wavelength of used light, and NA is the numerical aperture of lens. As illustrated in Figure 1.2, as the light wavelength remains the same (193nm) but the transistor feature size keeps shrinking following Moore' s law, the gap between the desired resolution (CD) and the light wavelength becomes increasingly large [100].

The continuous shrinkage of the transistor feature size and the rapid increasing of integration density have imposed severe challenges on design and manufacturing closures. The widening gap between limited manufacturing capability with sub-wavelength lithography technology and the high expected design performance has pushed the 193nm ArF lithography technology to its limit [73, 83, 102]. As shown in Figure 1.3, given an optical lithography system with $CD = 36nm$, if the required CD of advanced technology node designs are 40nm, 32nm, 22nm, 14nm and 7nm, respectively, then the patterned features are simulated. From the simulated results, it can be seen that the smaller size of features suffer from the more severe feature overlap.

To keep the pace of Moore's law, a variety of emerging technologies have been proposed to enable the manufacture of IC in semiconductor industry. A-

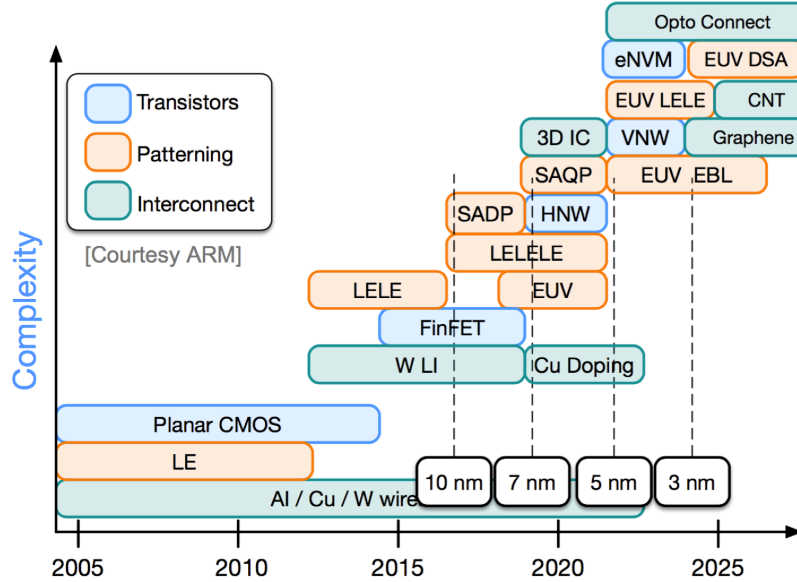


Figure 1.4: Roadmap for transistor, interconnect and patterning.

mong them, the most promising ones are multiple patterning lithography (MPL), e-beam lithography (EBL), block copolymer directed self-assembly (DSA) and extreme ultraviolet (EUV). Figure 1.4 shows the roadmap of lithography technologies. In the near term, MPL has become the most viable lithography technique. In the longer term, next-generation emerging lithography technologies, including EUV, EBL and DSA, are under intensive research and development.

Multiple patterning lithography (MPL) technology is an extension of the conventional optical lithography. It divides a layout into multiple masks and repeats the litho-etch process. As illustrated in Figure 1.5(a), MPL splits target features into several masks such that the coarser pitches on each mask can be single patterned using the 193nm wavelength lithography. Then features on different masks are combined to obtain finer pitches. According to different processes, MPL can be classified into LELE-type MPL and spacer-type MPL. LELE-type MPL includes double patterning lithography (DPL), triple patterning lithography (TPL) and quadruple patterning lithography (QPL), while spacer-type MPL includes self-aligned double patterning (SADP) and self-aligned quadruple

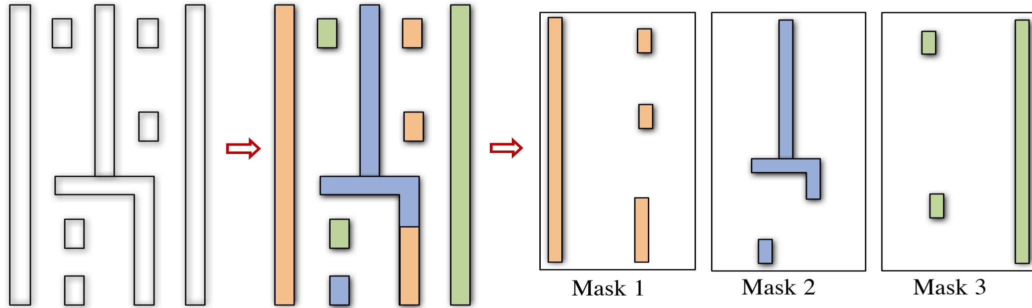
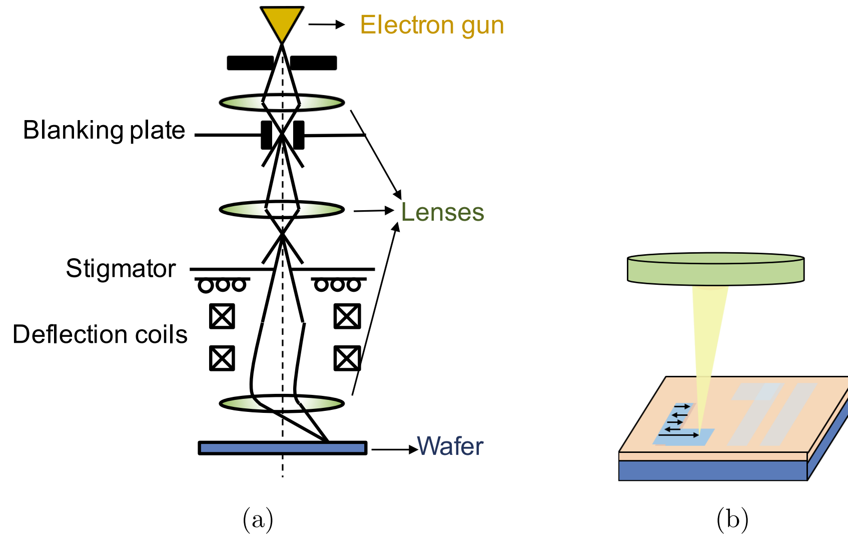


Figure 1.5: Triple patterning lithography process.

patterning (SAQP) [102]. As shown in Figure 1.4. DPL has been widely used in industry for years to produce sub-22nm devices. However, DPL gets to its limit for 14nm technology nodes and beyond. TPL, as a natural extension of DPL, is considered as a substitute of DPL for the 14nm, 10nm or even 7nm technology nodes. Figure 1.5 shows the layout decomposition process of TPL. Given a layout as the left figure, it is decomposed into three sparser parts corresponding to three masks.

EBL has been successfully deployed in many applications, especially for critical layers printing, photomask manufacturing, and prototyping. Besides, in the longer future, e.g., for the 7nm and 5nm technology nodes, it is a promising manufacturing solution for mass productions [72, 74]. EBL is a maskless lithography technology that shoots a beam of electrons onto a wafer to directly create features of desired shapes, as shown in Figure 1.6. Benefiting from this, EBL can achieve quite high resolution due to avoiding the light diffraction from the mask. Conventional EBL uses variable shaped beam (VSB), by which every shot can only create one rectangle, hence the total number of shots will be unacceptable for high-volume manufacturing [76].

As MPL is an extension of optical lithography, its throughput is high, but the resolution is still limited comparing with EBL. Besides, the mask cost will increase rapidly with the increasing number of masks and the design rules will

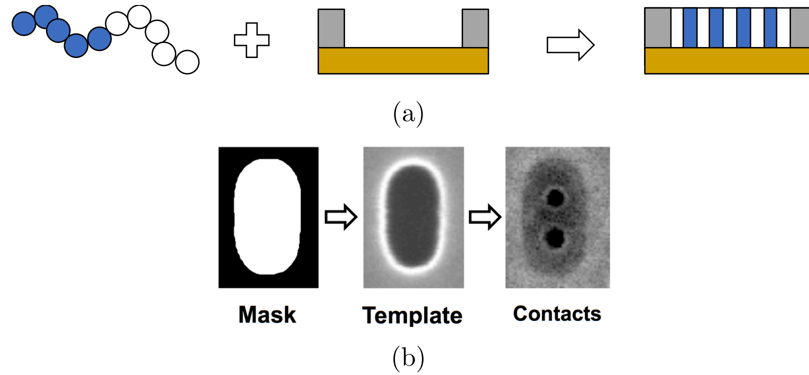


(a) (b)

Figure 1.6: E-beam system and process.

become even more complicated with more constraints. The resolution of EBL is excellent, but after decades of development, it still suffers from the bottleneck of low throughput. Therefore, a novel concept of hybrid lithography was recently proposed, which is to use multiple types of lithography technologies together to fabricate a layout [14]. For example, by combining MPL with high throughput and EBL with high resolution, more powerful manufacturing capability can be achieved and the manufacturing cost, including the mask cost, can be reduced.

In the DSA process, when the proportion between two block copolymers is appropriate, the block copolymers would form many desirable cylinders. And by removing these cylinders, the remaining material can be used to fabricate vias/contacts. To generate irregularly distributed vias using DSA technology, several neighbouring vias should be surrounded by guiding templates [34, 55]. As illustrated in Figure 1.7(a), two block copolymers and a guiding template produce a material with many cylinders. The guiding templates are first printed using the 193nm wavelength lithography, and then they are filled with special chemical material such as block copolymer. After the annealing process, cylinders will be formed and transferred to substrate patterns with sub-lithographic



(a)

(b)

Figure 1.7: Directed self-assembly process and guiding template.

pitches. DSA is particularly suitable for contact/via layer manufacturing [88] because of its capability of printing dense features with uniform dimensions. As shown in Figure 1.7(b), patterns of guide templates are first generated using optical lithography with masks, which will be used to guide the self-assembly of block copolymers. After that, contact holes will be formed by etching. When the contacts/vias are very dense, we can use MPL (i.e., multiple masks) to print the guide templates. In this thesis, we will present algorithms to incorporate DSA with MPL to fabricate contact/via layers in a more cost-effective way.

EUV has very short wavelength (13.5nm) to provide finer printing resolution compared to the 193nm wavelength lithography as shown in Figure 1.8. However, tremendous challenges, such as power sources, resists and defect-free masks, have notably delayed the adoption of EUV lithography for volume production. With the wavelength of only 13.5nm, EUV lithography greatly improves the problem of feature distortions caused by light diffraction. However, the light is absorbed by most materials, and thus only reflective optics (mirrors) and masks can be used; further, the scanning processing must be performed in a vacuum environment. For the process using clear-field masks, layout features on the masks are made by light absorbers, forming the dark regions on a wafer. In contrast, regions not covered by features are bright. As a result, the flare effect due to the surface roughness of the reflective optical components could

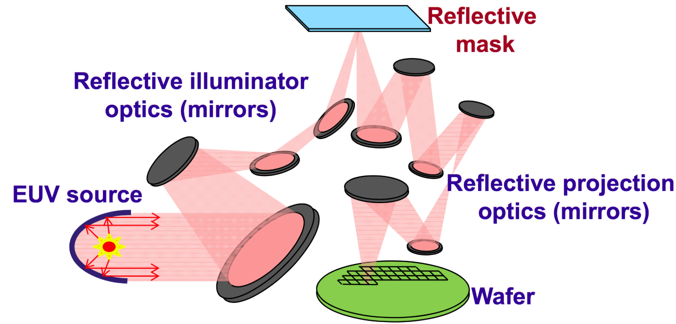


Figure 1.8: Extreme ultraviolet lithography system.

reduce the contrast between bright regions and dark ones. In particular, flare is inversely proportional to squared wavelength; the employed small wavelength makes EUV lithography suffer from high flare level. In addition, the thickness of absorbers and the incident angle of light cause the shadowing effect. Both process effects could result in significant critical dimension distortions or shape variations.

1.2 Challenges for Emerging Lithography Technology

As mentioned above, with the shrinking feature size, various of lithography technologies were proposed. However, each of these technologies suffers from many challenges. In this thesis, we consider some of crucial issues about computer-aided design algorithm as in Figure 1.9.

1.2.1 Challenge for Multiple Patterning Lithography

The key challenge of MPL is the new design problem, called layout decomposition, where input layout is divided into several masks (colors). When the distance between two input features is less than the minimum coloring distance, they need to be assigned to different masks to avoid a coloring conflict. Sometimes coloring conflict can be also resolved by inserting stitch to split a pattern into two touching parts. However, these introduced stitches lead to yield loss

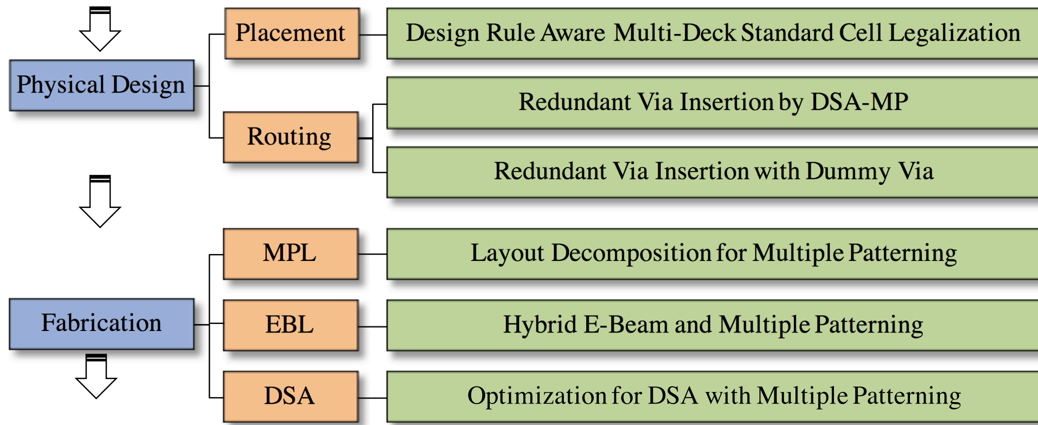


Figure 1.9: Design flow and challenges.

because of overlay error. Therefore, two of the main objectives in layout decomposition are conflict minimization and stitch minimization. An example of triple patterning layout decomposition is shown in Figure 1.5, where all features in input layout are divided into three masks, in which one of the features is split into two parts, and they are assigned to two masks.

1.2.2 Challenge for E-beam Lithography

The conventional type of EBL system is variable shaped beam (VSB). As illustrated in Figure 1.6(b), in VSB mode the layout is decomposed into a set of rectangles, and each rectangle would be shot into resist by dose of electron sequentially. The whole processing time of EBL system increases with numbers of beam shots. Even with decades of development, the key limitation of the EBL system has been and still is the low throughput [76]. Nowadays, EBL is more likely used as a complementary technology of MPL in manufacture process. Since the low throughput, we should assign as less as possible features to EBL process. Yield and resolution aware layout decomposition problem is the most crucial challenge.

1.2.3 Challenge for Block Copolymer Directed Self-Assembly

To pattern via/contact holes by DSA, guiding templates are usually used to form contacts [44]. For sparse structure, a number of single-hole templates are used to form contacts. For dense structure, too close templates would generate conflicts. To reduce the conflicts, some of the contacts within a short distance would be grouped together in a multi-hole template. However, grouping more than one contacts in a multi-hole template may introduce overlays. For different guiding templates with different shapes or sizes, the overlays are different. Specifically, complex (irregular shape) guiding templates may introduce large overlays and the contained contacts may not be patterned correctly [93]. Hence, during template assignment, the cost of a guiding template should be considered.

Furthermore, for a very dense contact layer layout, the contact layer fabricated by single patterning is unqualified due to a number of conflict errors. Hence the DSA with multiple patterning (DSA-MP) technology is a solid choice, and a crucial problem in DSA-MP is the mask and template assignment.

1.2.4 Challenge for Redundant Via Insertion by DSA with Multiple Patterning

In an IC layout, a via provides the connection between two net segments from adjacent metal layers. A single via may fail partially or completely because of various reasons, such as random defects, cut misalignment and electro migration or thermal stress [110]. A partial via failure may induce timing problems due to the increase of contact resistance and parasitic capacitance, while a complete via failure will produce a broken net in a circuit [85]. These failures may heavily hinder the functionality and yield of a circuit. Therefore, reducing yield loss due to via failure is one of the most important problems in the IC design flow. A promising method for improving via yield and reliability is adding a redundant via adjacent to every via [110], enabling via failure to be tolerated.

Conventionally, the redundant via insertion problem and the guiding template assignment for DSA-MP problem are considered at two separate stages. After obtaining a redundant via insertion, the guiding template assignment is considered. There exists an apparent issue for this separate manner. That is, if the via distribution is locally very dense, assigning vias to regular shaped DSA guiding templates is very difficult without violating design rules. Hence, consideration of the two stages simultaneously is necessary. In this chapter, we focus on redundant via insertion and guiding template assignment for DSA-MP problem, considering three scenarios: single patterning (SP), double patterning (DP), and triple patterning (TP).

1.2.5 Challenge for Multi-Deck Standard Cell Legalization

Modern circuit designs often contain (tens of) millions of standard cells located at placement sites on rows. To meet various design requirements such as low power and high performance, multi-deck standard cells occupying multi-rows (e.g., flip-flops) are often used in advanced technologies [8, 32]. Such multi-row height standard cells bring up challenging issues for placement, especially the mixed-cell-height legalization, due to the heterogenous cell structures and additional power-rail constraints, as pointed out in [26, 89].

In traditional single-row height standard-cell legalization, cell overlapping is independent among rows. In contrast, with multi-row height cells, shifting a cell in one row may cause cell overlaps in another row. The heterogenous cell structures could incur substantial global cell interferences among all cells in a circuit. Due to the global cell interference, existing single-row height standard-cell legalizers [16, 25, 28, 30, 50] cannot directly be extended to handle mixed-cell-height standard cells effectively. As a result, a mixed-cell-height legalization method needs to consider the heterogenous cell structures, with more global cell interferences and larger solution spaces. Moreover, the alignment of power

(VDD) or ground (VSS) lines must be considered in mixed-cell-height standard-cell legalization. In addition, to preserve the quality of a given global placement, an ideal legalization method should minimize not only the average cell movement but also the maximum one [29].

1.3 Contribution of This Thesis

In this thesis, firstly, some promising emerging lithography technologies for VLSI circuit manufacture are investigated. We propose theoretical frameworks and corresponding algorithms for challenges listed in the above subsections. Numerous experiments verify the efficiency and effectiveness of all proposed algorithms.

In Chapter 2, by considering the advantage of the relaxation method, we propose in this chapter a discrete relaxation theory and the theory based layout decomposition framework for triple patterning lithography (TPL). In the framework, conflicts and stitch insertions are considered separately. Our decomposition framework obtains a decomposition solution in two steps. The first step focuses on finding a discrete relaxation solution. Due to our relaxation by graph reduction tricks and stitch insertions not considered at this step, the solution space is dramatically reduced, and we can quickly find an optimal solution of the relaxation problem. At the second step, the relaxation solution is legalized to a feasible solution of the TPL layout decomposition problem. Experimental results and comparisons indicate that our discrete relaxation based method is fast and effective. Specifically, for some test benchmarks, optimal solutions are obtained according to our discrete relaxation theory.

In Chapter 3, we consider the hybrid e-beam and TPL of general layout decomposition problem. We propose a two-stage decomposition flow for the problem. At the first stage, we consider an e-beam and stitch aware TPL mask

assignment (ESTMA) problem, and then the problem is formulated as a binary linear program and solved by the cutting plane approach. At the second stage, the solution is legalized to a feasible solution of the HETLD problem by stitch insertion and e-beam shot. In addition, some graph reduction techniques proposed by previous TPL layout decomposers [36, 53, 59, 105] are used to reduce the problem size. Moreover, a new graph reduction which deletes some minor conflict edges is proposed to further speed up the decomposition flow. Furthermore, in order to obtain a better solution with less VSB number, we propose an extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) problem, which is also formulated as an ILP. In the first stage, if we solve the MDSR_4MA problem instead of the ESTMA problem, then more patterns can be assigned to TPL masks by inserting stitches. Experimental results show the effectiveness of the ESTMA and the MDSR_4MA based decomposition methods. In addition, it must be noted that the two issues in [96] are avoided in this chapter.

In Chapter 4, we focus on the mask and template assignment for DSA with MP (MTADT) problem of general layout. We sum up general rules for the costs of vertical or horizontal templates with different sizes, and construct a weighted conflict grouping graph, in which the cost of used templates can be calculated by the weights of conflict edges. Basing on the weighted conflict grouping graph, we propose a novel integer linear program for the MTADT problem, which is not equivalent to the MTADT problem but provides a lower bound on the optimal value of the MTADT problem. Moreover, some valid inequalities are introduced for cutting some no good solutions, and obtaining a better lower bound. We propose a template assignment approach to transform a relaxation solution to a feasible solution of the MTADT problem, which provides an upper bound on the optimal value of the MTADT problem. According to the obtained lower bound and upper bound, we can evaluate the quality of our

experimental results. Specially, if the upper bound is equal to the lower bound, then we obtain an optimal solution of the MTADT problem. Comparisons of experimental results show that our decomposition method is effective. More specifically, the gap between the obtained upper and lower bounds is 0.0% for most of the sparse benchmarks, which shows the optimality of the obtained results. And the average gap is 0.4% for the dense benchmarks, which shows the goodness of the obtained results for dense layouts.

In Chapter 5, we focus on redundant via insertion and guiding template assignment for DSA-MP problem, considering three scenarios: single patterning (SP), double patterning (DP), and triple patterning (TP). Previous works [39, 70] constructed ILP formulation basing on all the possible GTAs, which needs a large number of variables and constraints. Unlike [39, 70], we construct ILP formulation by introducing *multiplet*s, which can greatly reduce the numbers of variables and constraints. For single patterning, we construct a new conflict graph basing on *multiplet*s and formulate the problem as a constrained maximum weight independent set (CMWIS) problem. Under the assumption that a redundant via cannot be inserted if its related via is not manufacturable, we prove that the CMWIS problem is equivalent to the initial problem. We reduce the CMWIS problem to a maximum weight independent set problem such that it can be tackled by a fast algorithm, which can obtain a local optimal solution. For improving the solution quality, we propose a greedy method to obtain an initial solution for the fast algorithm. For double/triple patterning, we propose a new solution flow, which is a two-stage method. At the first stage, a contraction graph is constructed, and the contracted vertices are assigned to 2 or 3 masks. At the second stage, the solver for the single patterning is called to achieve a desirable RVI and GTA for every mask. Experimental results show that our algorithm for the problem with single patterning is faster than the methods in [39, 70], and our two-stage method for the problem with

double/triple patterning is much faster than the method in [70]. Moreover, the obtained results are better than those of the compared methods.

In Chapter 6, we investigate the redundant via insertion and DSA guiding template assignment problem with dummy via insertion simultaneously. We construct a new conflict graph for the redundant via insertion and DSA guiding template assignment problem with dummy via insertion, and describe it as an ILP formulation. We develop a novel fast algorithm to solve the ILP problem, which can obtain a local optimal solution. Experimental results indicate efficiency and effectiveness of our algorithm, and demonstrate that considering dummy via insertion for the problem is better than without the help of dummy via.

In Chapter 7, we present an analytical mixed-cell-height standard-cell legalization algorithm to simultaneously minimize the average and the maximum cell movements. By analyzing and remodeling the objective function and constraints, we formulate the mixed-cell-height standard-cell legalization problem as a mixed integer quadratic program (MIQP), which considers not only the average cell movement, but also the maximum cell movement, the sub-maximum movement, and the third maximum movement, etc. We convert the MIQP to a quadratic programming problem (QP). Unlike the work in [21] which minimizes only the average cell movement in the horizontal direction, we consider cells spreading continuously in both the horizontal and vertical directions. The QP is further reformulated as a linear complementarity problem (LCP), and solved by a modulus-based matrix splitting iteration method (MMSIM). The equivalence between the QP and the LCP is proved. We prove that the objective matrix is symmetric positive definite and the constraint matrix is of full row rank, therefore, the convergence of MMSIM is guaranteed. We propose a linear programming (LP) based method to further minimize the maximum cell movement in the horizontal direction. Experimental results demonstrate that our legalization model and method are effective for minimizing both the av-

erage and the maximum cell movements. Compared with the state-of-the-art work [21], for example, our algorithm reduces the average and maximum cell movements by 16% and 64%, respectively. In particular, our legalization model can be easily extended to consider the other complicated design rules (like fence region, edge type spacing, and pin access/shorts issues [29, 30], etc.).

1.4 Organization of This Thesis

In this thesis, we present our research results on design for manufacturability. The goal of this thesis is to resolve five DFM challenges in advanced lithography: layout decomposition, hybrid e-beam and MPL, optimization for DSA-MP, redundant via insertion by DSA-MP, manufacturability aware multi-deck cell legalization. The remaining chapters of this thesis are organized as follows.

In Chapter 2, we propose a discrete relaxation framework for triple patterning lithography layout decomposition. In Section 2.1, we introduce the double and triple patterning lithography layout decomposition, and corresponding previous methodologies. Problem formulation, the discrete relaxation theory, and the overview of the TPL layout decomposition framework are stated in Section 2.2. Section 2.3 shows the surface projection technique for identifying conflict features. Section 2.4 introduces the discrete relaxation method. Legalization process is explained in Section 2.5. Section 2.6 details the graph reduction methods used in our decomposition framework. The graph reduction is also a kind of discrete relaxation of the TPL layout decomposition problem. Experimental results will be given in Section 2.7, and we conclude our work finally.

In Chapter 3, we resolve the layout decomposition for hybrid e-beam and multiple patterning lithography by designing a two stage method. Section 3.1 describes the e-beam shot system and hybrid e-beam with MP layout decompo-

sition. In Section 3.2, we state the problem, introduce the concepts of conflict pattern and native conflict structure. In Section 3.3, we propose two problems: e-beam and stitch aware TPL mask assignment problem and extended minimum weight dominating set for R_4 mask assignment problem, and formulate them as integer linear programming problems. In Section 3.4, we detail the stitch and e-beam assignment for the patterns that are not resolved in the first decomposition stage. Section 3.5 introduces some graph reduction techniques and shows our hybrid decomposition flow. Experimental results are presented in Section 3.6, and summary is made in Section 3.7.

In Chapter 4, we focus on the mask and template assignment for DSA with multiple patterning problem of general layout. The existing works relative to DSA are presented in Section 4.1. Section 4.2 shows the template types and problem formulation. Section 4.3 introduces the discrete relaxation decomposition method, and the feasible solution generation method is introduced in Section 4.4. Experimental results are presented in Section 4.5, and summary of our work is made in Section 4.6.

In Chapter 5, we insert redundant via for each via in contact layer, and fabricate them by DSA-MP technology. The function and structure of redundant via and the hybrid DSA with MP technology are shown in Secntion 5.1. In Section 5.2, we describe the redundant via insertion and guiding template assignment for DSA-MP problem, and present our solution flow. In Section 5.3, we construct a conflict graph on the grid model. In Sections 5.4 and 5.5, we detail our solution methods for the problem with single patterning and double/triple patterning, respectively. In Section 5.6, we present the experimental results, followed by the summary in Section 5.7.

In Chapter 6, we consider redundant via insertion by DSA-MP with dummy via. Dummy via insertion is first introduced in Section 6.1. In Section 6.2, we introduce the related concepts and the problem. In Section 6.3, we introduce

the conflict graph construction. In Section 6.4, we detail our algorithms for the problem. Experimental results are presented in Section 6.5, and summary is drawn in Section 6.6.

In Chapter 7, manufacturability aware mixed-cell-height standard cell legalization is handled. Previous single-row standard cell legalizers and total movement aware mixed-cell-height standard cell legalizers are listed in Section 7.1. Section 7.2 gives the problem statement. Section 7.3 shows our legalization model, which allows cell movements in both the horizontal and vertical directions simultaneously. Section 7.4 details our legalization algorithms. Experimental results are given in Section 7.5, and finally summary is made in Section 7.6.

Finally, in Chapter 8, conclusions of this thesis are made and several feasible future works are listed.

Chapter 2 Discrete Relaxation Method for Triple Patterning Lithography Layout Decomposition

2.1 Introduction

Relaxation is an important approach for NP-hard combinatorial optimization problems. The minimum value of a relaxation problem provides a lower bound on the minimum value of an NP-hard combinatorial optimization problem. Moreover, by legalizing a minimum solution of the relaxation problem to a feasible one of the original problem, we can get a possibly least upper bound on the minimum value of the original problem. There are two categories of relaxation methods, i.e., continuous relaxation and discrete relaxation. Continuous relaxation includes convex relaxations [24], e.g., linear programming relaxation and semidefinite programming relaxation. Discrete relaxation refers to some discrete optimization problem based relaxation [12]. The discrete relaxation problem should be much easier to solve. In this chapter, we propose a discrete relaxation method for layout decomposition for triple patterning lithography, which is a 0-1 program relaxation.

With the need of industry for more and more smaller cells, the current 193nm ArF lithography technology cannot meet the need of the IC industry. Extreme Ultra-Violet (EUV) lithography is considered as a promising technology for next-generation lithography. However, due to mask material, light source and other device problems, EUV still cannot be put into IC manufacture. Consequently, multiple patterning lithography (MPL) technology has become the preferred, which decomposes an initial layout into multiple masks for use in multiple exposure lithography. MPL is considered of great values in research and industrial applications [4, 13, 51]. In MPL, layout decomposition is a key

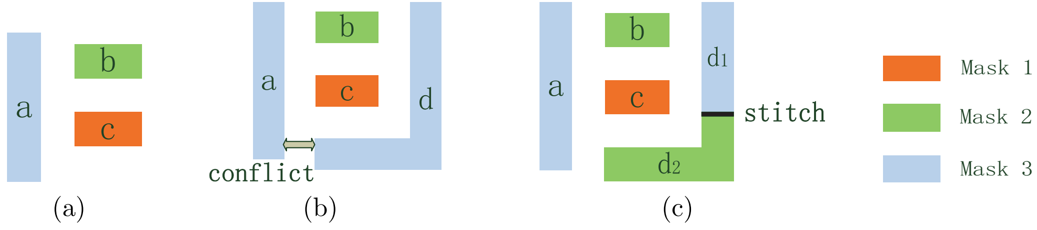


Figure 2.1: TPL layout decomposition. (a) An example of three masks assignment. (b) An example of layout with a conflict. (c) An example of layout with inserting a stitch to eliminate conflict.

step. In this chapter, we focus on the layout decomposition for triple patterning lithography.

Layout decomposition for triple patterning lithography (TPL) is that, an initial layout is decomposed into three masks. Figure 2.1(a) shows an example of decomposing a layout to three masks. For TPL, the rule of minimum coloring spacing is that, if two features are within the minimum coloring spacing min_{cs}, then they should be assigned to different masks; otherwise a conflict occurs between the two features. As shown in Figure 1(b), according to the rule of minimum coloring spacing, a conflict occurs between features a and d. Conflicts can be eliminated by inserting stitches. That is, a feature may be split into several touching sub-features by inserting stitches. As shown in Figure 2.1(c), a stitch is inserted into feature d. Since both conflict and stitch will affect the effect of lithography, especially the conflict, a crucial issue in TPL is to achieve the minimum numbers of conflicts and stitches.

Before introducing previous works on TPL layout decomposition, it is necessary to introduce some researches on double pattering lithography (DPL). Generally, DPL layout decomposer converts the layout decomposition problem to the 2-coloring problem. Kahng et al. [51] and Yuan et al. [107] constructed integer linear programming (ILP) models, where the number of conflicts and the number of stitches are minimized simultaneously. Xu et al. [95] and Tang et al. [79] used many graph division methods in DPL for reducing the scale of the problem. A method for identifying the native conflicts was proposed by Fang et

al. [37]. The method by Tang et. al [79] achieved the most up-to-date results, which are based on a polynomial time minimum cut algorithm.

The TPL layout decomposition problem and the problem with balanced density are NP-complete, which were proved by Yu et al. [101, 105]. Hence most of algorithms for the TPL layout decomposition problem belong to heuristic methods. A special case of the TPL layout decomposition problem is the problem where only row structure and standard cells are considered. A polynomial time heuristic algorithm was proposed by Tian et al. [81] for this problem. This was improved by Chien et al. [23].

For the general TPL layout decomposition problem, there are two kinds of common approaches. The first is the methods operating in the feasible solution space only, and trying to find a least possible upper bound on the minimum value of the TPL layout decomposition problem. Among this kind of methods, Cork et al. [27] proposed an algorithm based on the 3-SAT problem. The heuristic method by Fang et al. [36] uses the line projection method to identify conflicts, and then all edges of the conflict graph are weighted for designing algorithms. At present, the heuristic graph matching method proposed by Kuang et al. [53] achieves the fastest running time, in which a number of graph division methods are introduced for reducing the graph vertex number.

The second approach is the semidefinite program relaxations of the TPL layout decomposition problem by Yu et al. [101, 105], which are continuous relaxations. In [105], Yu et al. formulated the TPL problem as a linear binary program and a vector program. The vector program was further transformed to a semidefinite program for finding a relaxation solution. Generally, a solution of the relaxation problem is infeasible for the TPL problem, and is rounded to a feasible one. An advantage of this method is that, a solution of the relaxation problem provides a lower bound, and the feasible solution by rounding provides an upper bound on the minimum value of the TPL problem. Hence, the solution quality may be estimated. However, since conflicts and stitches are

considered simultaneously, the size of the relaxation problem is much larger, and the solution time is much more.

Discrete relaxation has not been proposed for the TPL layout decomposition problem. By considering the advantage of the relaxation method, we propose in this chapter a discrete relaxation theory and the theory based layout decomposition framework for TPL. In the framework, conflicts and stitch insertions are considered separately. Our decomposition framework obtains a decomposition solution in two steps. The first step focuses on finding a discrete relaxation solution. Due to our relaxation by graph reduction tricks and stitch insertions not considered at this step, the solution space is dramatically reduced, and we can quickly find an optimal solution of the relaxation problem. At the second step, the relaxation solution is legalized to a feasible solution of the TPL layout decomposition problem. Experimental results and comparisons indicate that our discrete relaxation based method is fast and effective. Specifically, for some test benchmarks, optimal solutions are obtained according to our discrete relaxation theory.

If stitch insertions are not allowed, then the TPL layout decomposition problem is the 3-coloring problem, and our decomposition method still works by deleting stitch insertion operations. Although the techniques used in our method look like heuristic methods, they are carefully adopted or designed for the discrete relaxation purpose. For example, the surface projection method developed is used for finding some features in a layout which will be colored prior. We believe our discrete relaxation idea and techniques could be applied to other problems.

2.2 Problem Formulation, Discrete Relaxation Theory, and Layout Decomposition Framework

In this section, we present firstly the TPL layout decomposition problem. Then we propose the discrete relaxation theory based on which our algorithm for the TPL problem is developed. Finally, we give an overview of our TPL layout decomposition framework.

2.2.1 Problem Formulation

We introduce two definitions as follows.

Definition 2.2.1 (conflict graph, CG)**.** The conflict graph is defined as an undirected graph $G(V, E)$, where V represents the set of vertices, and $v \in V$ represents a feature. E is the set of edges, and $e_{ij} \in E$ exists between two features i and j if the distance between them is less than $\min_{cs}$.

Definition 2.2.2 (sub-feature)**.** One or more stitches are inserted into a feature, and the feature is split into several parts, namely sub-features.

The TPL layout decomposition problem is to decompose an initial layout into three masks via possible stitch insertions. The objective is minimizing the numbers of conflicts and inserted stitches. Formally, it can be described as follows.

Problem (P_0): TPL layout decomposition problem

Given: Layout L, the minimum coloring spacing $\min_{cs}$, the minimum feature size $\min_{fs}$, the minimum overlap margin $\min_{om}$, a constant α ($0 \leq \alpha < 1$).

Find: Stitch insertions in features and assignment of features and sub-features to three masks, such that $|C| + \alpha|S|$ is minimized, where $|C|$ is the number of conflicts occurring and $|S|$ is the number of stitches inserted. The locations of stitches should be legal.

Inserting a stitch legally into a feature means that: (i) the size of the

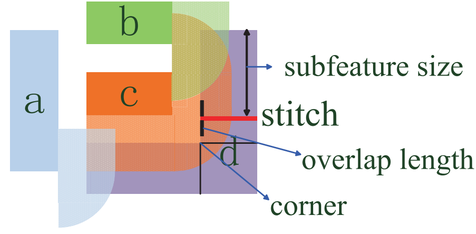


Figure 2.2: A sample of legal stitch insertion.

generated sub-features should be larger than the minimum feature size $\min_{fs}$; (ii) an inserted stitch is not near any corner of the feature; and (iii) the overlap length is greater than the minimum overlap margin $\min_{om}$. Here overlap length means that, stitch insertion position can be moved vertically or horizontally without causing any new conflict, and the move length is called the overlap length [53]. An example of a legal stitch is shown in Figure 2.2.

The above constraints are due to that, a very small sub-feature is easy to generate overlay error [36,51,53]; a corner stitch may cause significant side effect on printability; and similarly, if the overlap length of a stitch is less than $\min_{om}$, then the stitch may cause side effect too [36, 51, 53].

In the problem, assigning features to three masks is equivalent to 3-coloring of the conflict graph. Hence in the following, when saying coloring a vertex of the conflict graph, it means assignment of the corresponding feature to a mask, and vice versa.

2.2.2 Discrete Relaxation Theory

Definition 2.2.3 (discrete relaxation)**.** Problem (RP):

$$z^R = \min\{f^R(x) : x \in X^R\}$$

is a discrete relaxation of problem (P):

$$z = \min\{f(x) : x \in X\},$$

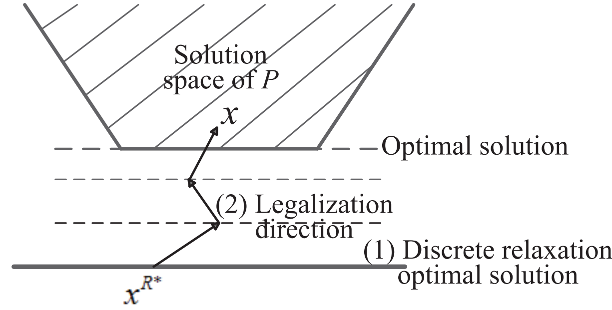


Figure 2.3: Geometrical representation of discrete relaxation.

if there exists an optimal solution $x^{R\star}$ of problem (RP), and there exists an optimal solution x^* of problem (P) such that $f^R(x^{R\star}) \leq f(x^*)$.

In the above definition, the function f^R and the feasible set X^R must be selected carefully. Generally, we should select an X^R such that an optimal solution of the relaxation problem (RP) could be transformed easily to a feasible solution of the original problem (P).

Although Definition 2.2.3 can be applied without any assumption on the feasible sets X^R and X, we restrict them to be discrete sets for our usage and call the relaxation as discrete relaxation. By Definition 2.2.3, we immediately have:

Proposition 2.2.4. If problem (RP) is a discrete relaxation of problem (P), then $z^R \leq z$.

Proposition 2.2.4 means that, by discrete relaxation we will obtain a lower bound on the minimum value of the original problem. Specifically, we have:

Proposition 2.2.5. Suppose that problem (RP) is a discrete relaxation of problem (P). Let $x^{R\star}$ be an optimal solution of problem (RP). If $x^{R\star}$ can be transformed to a feasible solution x of problem (P), such that $f^R(x^{R\star}) = f(x)$, then x is an optimal solution of problem (P).

Proof. By Proposition 2.2.4 and the assumptions of this proposition, $z^R = f^R(x^{R\star}) = f(x) \leq z$. Since $x \in X$, we have $z \leq f(x)$. So $f(x) = z$, which

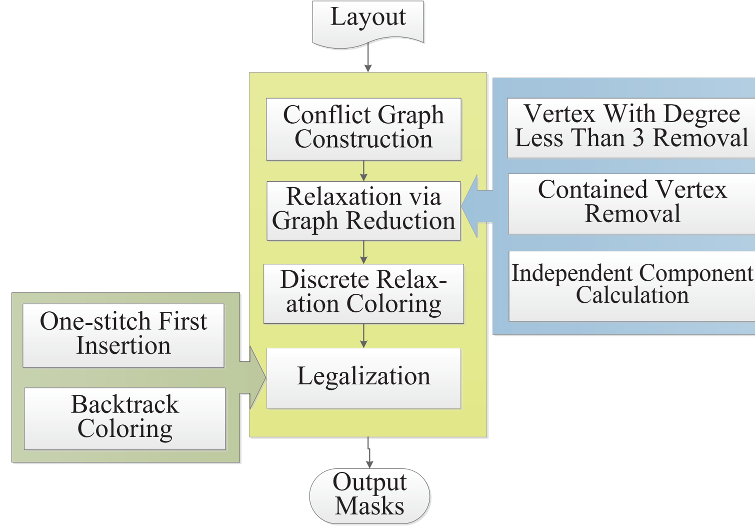


Figure 2.4: Our TPL layout decomposition framework.

means that x is an optimal solution of problem (P). □

The principle of using the above discrete relaxation theory is as Figure 2.3 shows. Firstly, we formulate a discrete relaxation problem (RP) for problem (P) to obtain an optimal solution $x^{R\star}$ of (RP). And then, $x^{R\star}$ is legalized to a feasible solution x of the original problem (P), and we have $f^R(x^{R\star}) \leq f(x)$ by Proposition 2.2.4. If $f^R(x^{R\star}) = f(x)$, then by Proposition 2.2.5 we obtain an optimal solution x of problem (P).

Although the above discrete relaxation theory looks simple, we carefully design relaxation techniques for the TPL layout decomposition problem, and the computational results are promising.

2.2.3 Overview of Layout Decomposition Framework

In Figure 2.4, we illustrate our decomposition framework for the TPL layout decomposition problem. Firstly, according to the rule of minimum coloring spacing, we transform an initial layout to a conflict graph using our new projection method, i.e., surface projection method. And then, we adopt two techniques to reduce the conflict graph, which is a discrete relaxation. After that, the large scale TPL layout decomposition problem is reduced to numerous small scale

TPL layout decomposition subproblems.

At the discrete relaxation coloring step, a 0-1 program is constructed for each reduced TPL problem, which is a discrete relaxation. Then, we solve the 0-1 program to obtain a discrete relaxation solution for each reduced TPL problem. At the discrete relaxation step, stitch insertions are ignored. At the legalization step, stitch insertion and backtrack coloring algorithms are respectively used to legalize the discrete relaxation solutions. Finally, we color the removed features, which are removed at the relaxation by graph reduction step. Details of the decomposition framework are described in the following sections.

2.3 Surface Projection and K_4 Conflict Structure

In order to eliminate conflict edges in a layout, stitches might be introduced to split a feature into several sub-features. But, conflict edges between some features might not be totally eliminated by inserting stitches. We call these conflict edges as non-resolvable conflicts, and call the corresponding conflict sub-graph as non-resolvable conflict structure.

In this section, we develop a surface projection method for finding the conflict features in a layout. The conflict features will be prior considered at the discrete relaxation coloring step. Using the conflict features, the K_4 conflict structures in a layout can be identified, which have an important property that the conflict edges in the structures cannot be totally eliminated by inserting stitches.

2.3.1 Surface Projection

Line projection method [51] has been popularly utilized for constructing a conflict graph and finding stitch insertions for the DPL or TPL layout decompositions. Figure 2.5(a) shows an example of line projection of feature b on feature c. However, we want to identify features which are critical and should

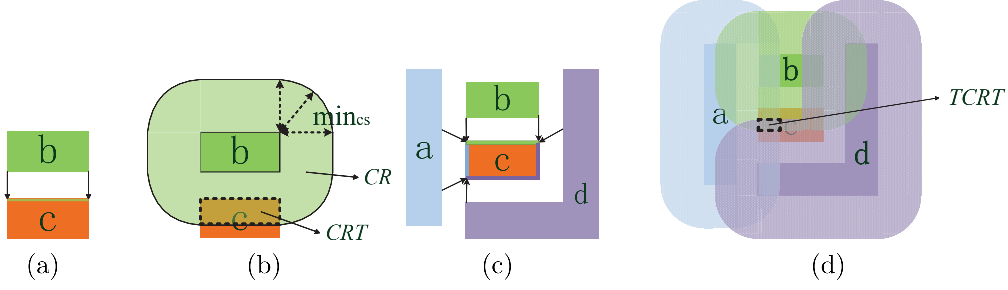


Figure 2.5: Surface projection and line projection. (a) Line projection. (b) Surface projection. (c) An example shows that line projection cannot find inside $TCRT$. (d) Surface projection is used to find $TCRT$.

be colored prior, and this forms a basis of our discrete relaxation method. For this purpose, we develop the surface projection method.

Note that, the relation between two neighboring features is more than their boundaries. Some useful relation may be inside the features. Using the information, we want to identify the features at which non-resolvable conflicts are more likely to exist, which might be colored prior. We develop a surface projection method to mine the information. In order to present the method accurately, some definitions are introduced as follows.

Definition 2.3.1 (conflict region, CR)**.** The conflict region of a feature is defined as a 2D region around the feature but within the minimum coloring spacing $\min_{cs}$.

Definition 2.3.2 (conflict rectangle, CRT)**.** The conflict rectangle on feature a by feature b is defined as the minimum rectangle enclosing the intersection of feature a and the conflict region of feature b.

As illustrated in Figure 2.5(b), the shaded round region around feature b is called the conflict region (CR) of b, and the dashed box on feature c is the conflict rectangle CRT resulted by b.

Definition 2.3.3 (triple conflict rectangle, $TCRT$)**.** The triple conflict rectangle on feature a is defined as a rectangle, which is the intersection of three or more

conflict rectangles on feature a. Some adjacent features of feature a create these conflict rectangles, and these adjacent features are called **conflict adjacent features (**CAF**)** of feature a.

For any feature a, the line projection method can find all possible conflict line segments on the peripheries of a, which are projections of the neighbouring features. However, for some features, line projection may not find all possible conflict regions inside the features, and then, it may not find the triple conflict rectangles inside the features. An example is illustrated as Figures 2.5(c) and 2.5(d).

In Figure 2.5(c), line projection finds three 1D conflict line segments on the four peripheries of feature c. In Figure 2.5(d), surface projection finds three 2D conflict regions on feature c, and a triple conflict rectangle ($TCRT$), i.e., dashed box, can be found inside c, which cannot be found by line projection. Detecting triple conflict rectangle is crucial for finding conflict features and $K4$ conflict structures, which will be presented in the next subsection.

2.3.2 Conflict Feature and K_4 Conflict Structure

Definition 2.3.4 (conflict feature, CF)**.** Feature a is called a conflict feature if it satisfies the two conditions: (1) there exists a triple conflict rectangle $TCRT$ on feature a; (2) the graph composed of feature a and its conflict adjacent features (CAF) is not 3-colorable.

Definition 2.3.5 (K_4 conflict structure, K_4CS)**.** A graph structure is a K_4 conflict structure, if it is a K_4 structure, and all the four features are conflict features CF and they are CAF each other.

By Definition 2.3.4, the feature c in Figure 2.5(d) is a CF, since there is a $TCRT$ on it; and c and its conflict adjacent features a, b and d compose a K_4. Similarly, features a, b, c and d in Figure 2.6(a) are CF too. In addition, the structure composed of features a, b, c and d in Figure 2.6(a) is a K_4 conflict

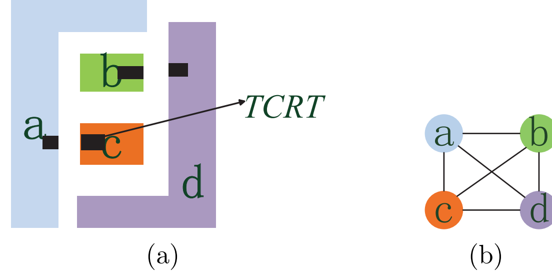


Figure 2.6: Conflict features and a K_4 conflict structure.

structure K_4CS, in which the $TCRT$s in features b and c cannot be identified by the line projection method. Hence, our surface projection method can be used to locate more unresolvable conflicts, since some conflict structures K_4CS cannot be identified by the line projection method.

For a conflict feature, it has a property as follows.

Lemma 2.3.6. Suppose that feature a is a conflict feature, and features a_1, a_2, $\ldots$, a_{k+1} are the sub-features of feature a, where k is the number of stitches inserted into feature a. Then at least one of features a_1, a_2, $\ldots$, a_{k+1} is still a conflict feature.

Proof. If feature a is a conflict feature, then it has at least a $TCRT$. We insert k stitches into feature a and split it into sub-features $a_1, a_2, ..., a_{k+1}$. Suppose that we first insert stitch s_1 into feature a and split it into two parts a_1 and a_2. Then there are two possible results: 1) if stitch s_1 cuts across the $TCRT$ on feature a, we have that both of the sub-features a_1 and a_2 contain a new $TCRT$; 2) otherwise, only one of a_1 and a_2 contains $TCRT$. Next, stitches are inserted into feature a one by one. At last, feature a is split into $k+1$ sub-features $a_1, a_2, ..., a_{k+1}$. Obviously, the number of sub-features containing $TCRT$ is nondecreasing. Hence, at least one of the sub-features a_i contains $TCRT$, and the sub-graph consisting of the sub-feature a_i and the conflict adjacent features CAF of feature a is not 3-colorable. By Definition 2.3.3, this completes the proof. □

Lemma 2.3.6 shows that, inserting stitches into a conflict feature cannot make it be a non-conflict feature. Hence, when coloring, conflict features should be colored prior. Especially, for a K_4 conflict structure, it has the following property.

Theorem 2.3.7. Every K_4 conflict structure exists at least a non-resolvable conflict.

Proof. Since a K_4CS is a K_4 structure, it is not 3-colorable, which means that for any color assignment, at least a conflict exists in the K_4CS. Furthermore, since all the four features in a K_4 structure are CFs, and by Lemma 2.3.6, stitch insertions cannot eliminate the conflict in the K_4CS. Hence a K_4 conflict structure exists at least a non-resolvable conflict. □

Theorem 2.3.7 suggests that a K_4 conflict structure exists a conflict which cannot be eliminated by inserting stitches. To find potential conflicts in a layout, it is important to identify the conflict features. Identifying the conflict features can be done by the BFS algorithm, which traverses the conflict graph. Furthermore, for every vertex v with $TCRT$, the algorithm will find all K_4 subgraphs containing v. Suppose that the maximum degree of vertices in a layout is d. Then the runtime of determining whether there exists a subgraph containing v is K_4 is in time $O(d^3)$. Hence, the total runtime complexity for identifying CFs is in time $O(nd^3)$, where n is the number of features in a layout.

2.4 Discrete Relaxation Method for TPL Layout Decomposition

Since it is hard to solve directly the TPL layout decomposition problem, we propose a discrete relaxation method for finding a lower bound on the minimum value of the problem, basing on the discrete relaxation theory.

In this part, we describe our discrete relaxation method for the TPL layout

decomposition problem, which is to obtain a lower bound on the optimal value of the problem.

2.4.1 Equivalent Formulation of Problem (P_0) to (P_1)

First, we formulate the TPL layout decomposition problem (P_0) to an equivalent problem (P_1). Here, we consider the TPL layout decomposition problem by assigning features to three masks first, and then consider stitch insertions into some features.

Suppose that $G(V, E)$ is decomposed into four sub-graphs $G_1(C_1,\ E_1)$, $G_2(C_2,\ E_2)$, $G_3(C_3,\ E_3)$, $G_4(R_4,\ E_4)$, where E_i is the set of edges between the features in C_i ($i = 1, 2, 3, 4$), respectively, and E_i ($i = 1, 2, 3$) is an empty set. After obtaining the decomposition C_1, C_2, C_3 and R_4, we consider stitch insertions for the features in R_4. Then, the generated sub-features and the unsplitted features in R_4 are assigned to classes C_1, C_2 and C_3 with the minimum total number of conflicts in C_1, C_2 and C_3.

Let $\mathcal{S}(R_4)$ be the set of all possible plans of inserting stitches into the features in R_4. Given C_1, C_2, C_3, $S \in \mathcal{S}(R_4)$, there are many ways of assigning the generated sub-features and the unsplitted features in R_4 to classes C_1, C_2 and C_3. And different way of assignment will produce different conflicts. Here we let $T(C_1, C_2, C_3, S)$ be the minimum total number of conflicts in C_1, C_2 and C_3 among these ways of assignment.

Mathematically, problem (P_0) can be formulated equivalently as the following problem (P_1):

$$\min \quad T(C_1, C_2, C_3, S) + \alpha|S| \tag{2.1}$$

$$\text{s.t.} \quad \text{if } i, j \in C_k,\ \text{then } e_{ij} \notin E,\ k = 1, 2, 3; \tag{2.1a}$$

$$S \in \mathcal{S}(R_4), \tag{2.1b}$$

where constant α is a weight parameter. In this chapter, we take $\alpha = 0.1$, which is as that in Yu et al. [105].

Let X_{P_0}, X_{P_1} be the solution spaces of problems (P_0) and (P_1), and let $f_0(x_{P_0})$, $f_1(x_{P_1})$ be the objective functions of problems (P_0) and (P_1), respectively.

Claim 2.4.1. Problem (P_1) is an equivalent formulation of problem (P_0).

Proof. Suppose that $x^*_{P_0}$=(M^*_1, M^*_2, M^*_3)$\in X_{P_0}$ is an optimal solution of problem (P_0), where M^*_k $(k = 1, 2, 3)$ includes the stitch free features and the sub-features produced by stitch insertion plan S. Move all sub-features in M^*_1, M^*_2, M^*_3 to R_4. After that, if there exists a conflict in M^*_1, M^*_2 or M^*_3, move a feature at which the conflict occurs to R_4. Repeating this operation will finally produce a feasible solution (C_1, C_2, C_3, S) of problem (P_1). Obviously

$$f_0(x^*_{P_0}) \geq T(C_1, C_2, C_3, S) + \alpha|S|.$$

Hence the optimal value of problem (P_1) is a lower bound on the optimal value of problem (P_0).

Conversely, an optimal solution of problem (P_1) is a feasible solution of problem (P_0), which means that the optimal value of problem (P_0) is also a lower bound on the optimal value of problem (P_1). Combining the two cases implies that Claim 2.4.1 holds. □

2.4.2 Relaxation of Problem (P_1) to (P_2)

Next, we relax problem (P_1) to a problem (P_2) by discrete relaxation, which is a 0-1 program whose optimal value provides a lower bound on the optimal value of problems (P_1) and (P_0). Problem size of the 0-1 program is significantly less than that of the TPL layout decomposition problem, due to ignoring stitch insertions.

Let (x_{i1}, x_{i2}) be a two dimensional binary variable, which is used to express

the color of vertex i. When $(x_{i1}, x_{i2}) = (0,1)$, it means $i \in C_1$; similarly, when $(x_{i1}, x_{i2}) = (1,0)$, it means $i \in C_2$; and when $(x_{i1}, x_{i2}) = (1,1)$, it means $i \in C_3$. However, when $(x_{i1}, x_{i2}) = (0,0)$, it means that vertex i is uncolored, i.e., $i \in R_4$.

Let w_i be the weight of vertex i. By Theorem 2.3.7, we know that a non-resolvable conflict is more likely to exist at a conflict feature CF. Hence a conflict feature CF should be assigned a greater weight. Specifically, let

$$w_i = \begin{cases} 1, & \text{if } i \text{ is conflict feature;} \\ \alpha, & \text{if } i \text{ is not conflict feature.} \end{cases}$$

Then we relax problem (P_1) to the 0-1 program (P_2):

$$\begin{aligned}
\min \quad & \sum_{i \in V} w_i (1 - x_{i1})(1 - x_{i2}) && && (2.2) \\
\text{s.t.} \quad & x_{i2} - x_{i1} + x_{j2} - x_{j1} \le 1, && \forall e_{ij} \in E; && (2.2a) \\
& x_{i1} - x_{i2} + x_{j1} - x_{j2} \le 1, && \forall e_{ij} \in E; && (2.2b) \\
& x_{i1} + x_{i2} + x_{j1} + x_{j2} \le 3, && \forall e_{ij} \in E; && (2.2c) \\
& (x_{i1}, x_{i2}) \in \{0,1\} \times \{0,1\}, && \forall i \in V. && (2.2d)
\end{aligned}$$

In the objective function,

$$(1 - x_{i1})(1 - x_{i2}) = \begin{cases} 0, & \text{if } i \notin R_4; \\ 1, & \text{if } i \in R_4. \end{cases}$$

Hence the objective is minimizing the total weight of the vertices in R_4. Constraints (2.2a)-(2.2c) are equivalent to Constraint (1a), which are used to force that, if $e_{ij} \in E$, then vertices i and j should not be in the same class C_1, C_2 or C_3, respectively. Specifically, for (2.2a), if vertices i and j are in C_1, then $e_{ij} \notin E$; otherwise, $(x_{i1}, x_{i2}) = (x_{j1}, x_{j2}) = (0,1)$ violate Constraint (2.2a).

Furthermore, for an optimal solution (C_1, C_2, C_3, R_4) of problem (P_2), the following observation is obvious. Any feature in R_4 conflicts with a feature in

C_i, $i = 1, 2, 3$, respectively; otherwise it will be moved to C_i, and we get a better solution.

From the formulation of program (P_2), it can be seen that the difficulty of resolving conflicts is considered by assigning weights to vertices. This difficulty has also been addressed by Fang et al. [36], in which weights are assigned to edges.

Next, we discuss the relationship between problems (P_1) and (P_2). For any optimal solution $x^*_{P_1}$=(C^*_1, C^*_2, C^*_3, S^*) of problem (P_1), we suppose without loss of generality that, if a feature $i \in R^*_4 = V - C^*_1 \cup C^*_2 \cup C^*_3$ can be moved to C^*_1, C^*_2 or C^*_3 without causing conflicts, then it has been moved.

For the convenience of description, we divide the features in R^*_4 into two classes, UCF and $UNCF$, where UCF is the set of uncolored conflict features, and $UNCF$ is the set of uncolored non-conflict features.

Lemma 2.4.2. For any optimal solution $x^*_{P_1}$=(C^*_1, C^*_2, C^*_3, S^*)$\in X_{P_1}$ of problem (P_1), if a feature $i \in UCF$, then there exists at least a conflict occurring at i which cannot be eliminated by inserting stitches.

Proof. Suppose that feature i belongs to UCF. Then i is a conflict feature, $i \in R^*_4$, and at least a conflict will occur at feature i. By Lemma 2.3.6, if stitches are inserted into feature i, then one or more sub-features of feature i are conflict features, and there still exists at least a conflict. Hence, at least a conflict occurs at feature i which cannot be eliminated by inserting stitches. □

Note that, given an optimal solution $x^*_{P_1}$=(C^*_1, C^*_2, C^*_3, S^*) of problem (P_1), when calculating $T(x^*_{P_1})$, the features and sub-features resulted by stitch insertions S^* of features in R^*_4 are assigned optimally to C^*_1, C^*_2 or C^*_3. In this case, some features in R^*_4 will contribute conflicts, we call them Occurring Conflict Features (OCF); and some features in R^*_4 will not contribute conflicts, we call them Not Occurring Conflict Features ($NOCF$). Obviously, $OCF \cup NOCF = R^*_4$, and $|OCF| + |NOCF| = |R^*_4|$.

Let $f_1(x_{P_1})$ be the objective function of problem (P_1), and let $f_2(x_{P_2})$ be the objective function of problem (P_2). By the above assumptions, the following result can be deduced.

Theorem 2.4.3. Given an optimal solution $x^*_{P_1}$=(C^*_1, C^*_2, C^*_3, S^*) of problem (P_1), there exists an optimal solution $x^*_{P_2}$ of problem (P_2) such that $f_2(x^*_{P_2}) \leq f_1(x^*_{P_1})$.

Proof. Suppose that $x^*_{P_1} = (C^*_1,\ C^*_2,\ C^*_3,\ S^*) \in X_{P_1}$ is an optimal solution of problem (P_1). It is obvious that $|UNCF| + |UCF| = |R^*_4|$. By Lemma 2.4.2, we know that every feature in UCF contributes at least a conflict number. Hence $|UCF| \leq |OCF| \leq T(x^*_{P_1})$. Let $|Y| = T(x^*_{P_1}) - |UCF|$. Obviously, $|Y| \geq 0$.

Moreover, every feature $i \in NOCF$ must be inserted $s_i \geq 1$ stitches to eliminate conflicts. So $|NOCF| \leq |S^*|$, and

$$\begin{aligned}
&|UNCF| + |UCF| - T(x^*_{P_1}) = |R^*_4| - T(x^*_{P_1}) \\
&\leq |R_4|^* - |OCF| = |NOCF| \leq |S^*|.
\end{aligned}$$

Furthermore, since $0 \leq \alpha = 0.1 < 1$, we have

$$\begin{aligned}
&|UCF| + \alpha|UNCF| \\
&\leq |UCF| + \alpha|UNCF| + (1-\alpha)|Y| \\
&= |UCF| + |Y| + \alpha(|UNCF| - |Y|) \\
&= T(x^*_{P_1}) + \alpha(|UNCF| + |UCF| - T(x^*_{P_1})) \\
&\leq T(x^*_{P_1}) + \alpha|S^*| = f_1(x^*_{P_1}).
\end{aligned}$$

According to the value of w_i, for the solution $x_{P_2} = (C^*_1,\ C^*_2,\ C^*_3,\ R^*_4)$ of problem (P_2), which is transformed from $x^*_{P_1}$, it is evidently that

$$f_2(x_{P_2}) = \sum_{i \in V} w_i(1 - x_{i1})(1 - x_{i2}) = |UCF| + \alpha|UNCF|.$$

Obviously, there exists an optimal solution $x^*_{P_2}$ of problem (P_2) with

$$f_2(x^*_{P_2}) \leq f_2(x_{P_2}) \leq f_1(x^*_{P_1}).$$

Hence, Theorem 2.4.3 holds. $\square$

By Claim 2.4.1, Theorem 2.4.3 and the definition of discrete relaxation, it is obvious that problem (P_2) is a discrete relaxation of the TPL layout decomposition problem (P_0). Specifically, we have

Claim 2.4.4. Suppose $x^*_{P_2} \in X_{P_2}$ is an optimal solution of problem (P_2). If $x^*_{P_2}$ is legalized to a feasible solution x_{P_0} of problem (P_0), then

$$f_2(x^*_{P_2}) \leq f_0(x_{P_0}).$$

Specially, if $f_2(x^*_{P_2}) = f_0(x_{P_0})$, then x_{P_0} is an optimal solution of the TPL layout decomposition problem (P_0).

Furthermore, we have two special cases in which optimality of the TPL layout decomposition problem can be verified.

Claim 2.4.5. Suppose N is the number of K_4 conflict structures in a layout. If there exists a solution x_{P_2} of problem (P_2) with conflict number N, then x_{P_2} is a minimum conflict solution of problem (P_0).

Proof. Since every K_4 conflict structure contributes at least a conflict, the layout with N K_4 conflict structures exists at least N conflicts. Hence the solution x_{P_2} with conflict number N is a minimum conflict solution of problem (P_0). $\square$

Claim 2.4.6. Let $x^*_{P_2} \in X_{P_2}$ be an optimal solution of problem (P_2). If every feature $i \in UNCF$ can be inserted at most $s_i = 1$ stitch to eliminate conflicts, and every feature $i \in UCF$ contributes at most a conflict number, then $x^*_{P_2}$ together with the stitch insertion into every feature $i \in UNCF$ is an optimal solution of problem (P_0).

Proof. For the optimal solution $x^*_{P_2}$ of problem (P_2), if every feature $i \in UNCF$ can be inserted at most $s_i = 1$ stitch to eliminate conflicts, and every feature $i \in UCF$ contributes at most a conflict number, then $UCF = OCF$ and $UNCF = NOCF$. Thus the total conflict numbers $|C| = |OCF| = |UCF|$, and

$$|S| = \sum_{i \in NOCF} s_i = |NOCF| = |UNCF|.$$

So

$$|UCF| + \alpha|UNCF| = |C| + \alpha|S|,$$

and

$$|UCF| + \alpha|UNCF| = \sum_{i \in V} w_i(1 - x_{i1})(1 - x_{i2}).$$

Hence, $x^*_{P_2}$ together with the stitch insertion into every feature $i \in UNCF$ is an optimal solution of problem (P_0). □

Problem (P_2) is a nonlinear 0-1 program, which is generally difficult to solve in large scale cases. However, our relaxation via graph reduction techniques proposed in Section 2.6 can reduce the conflict graph to many small size independent components. Thus problem (P_2) on the small size independent components can be solved easily. Specifically, we adopted the Branch-and-Bound method in the software package GUROBI [3] to solve the problem on the small size independent components.

Actually, our 0-1 non-linear program (P_2) can be linearized equivalently to the 0-1 linear program (P_3) as follows by introducing new variables and constraints:

$$
\begin{aligned}
\min \quad & \sum_{i\in V} w_i x_{i3} && && (2.3)\\
\text{s.t.} \quad & x_{i2} - x_{i1} + x_{j2} - x_{j1} \leq 1, && \forall e_{ij} \in E; && (2.3a)\\
& x_{i1} - x_{i2} + x_{j1} - x_{j2} \leq 1, && \forall e_{ij} \in E; && (2.3b)\\
& x_{i1} + x_{i2} + x_{j1} + x_{j2} \leq 3, && \forall e_{ij} \in E; && (2.3c)\\
& 1 - x_{i1} - x_{i2} \leq x_{i3}, && \forall i \in V; && (2.3d)\\
& (x_{i1}, x_{i2}) \in \{0,1\}^2, x_{i3} \in \{0,1\}, && \forall i \in V. && (2.3e)
\end{aligned}
$$

In the above formulation, Constraint (2.3d) is used to force that, if $x_{i1} = 0$ and $x_{i2} = 0$ then $x_{i3} = 1$; otherwise $x_{i3} = 0$, since the objective is minimization and $w_i > 0$. However, we did not adopt the 0-1 linear program (P_3) in our decomposition flow, since it uses more variables and constraints than the 0-1 non-linear program (P_2), and the software package GUROBI [3] is faster on the small scale program (P_2) than on program (P_3).

The 0-1 linear program (P_3) is a formulation of the TPL layout decomposition problem (P_0) without stitch insertions. In [105], Yu et al. proposed an exact 0-1 linear program formulation of the TPL layout decomposition problem (P_0). If forbidding stitch insertions, their formulation is reduced to the following program (P_4),

$$
\begin{aligned}
\min \quad & \sum_{e_{ij} \in E} c_{ij} && && (2.4)\\
\text{s.t.} \quad & x_{i1} + x_{i2} \leq 1, && \forall i \in V; && (2.4a)\\
& x_{i1} + x_{j1} \leq 1 + c_{ij1}, && \forall e_{ij} \in E; && (2.4b)\\
& (1 - x_{i1}) + (1 - x_{j1}) \leq 1 + c_{ij1}, && \forall e_{ij} \in E; && (2.4c)\\
& x_{i2} + x_{j2} \leq 1 + c_{ij2}, && \forall e_{ij} \in E; && (2.4d)\\
& (1 - x_{i2}) + (1 - x_{j2}) \leq 1 + c_{ij2}, && \forall e_{ij} \in E; && (2.4e)\\
& c_{ij1} + c_{ij2} \leq 1 + c_{ij}, && \forall e_{ij} \in E; && (2.4f)\\
& x_{i1},\ x_{i2},\ c_{ij1},\ c_{ij2},\ c_{ij} \in \{0, 1\}, && \forall i, j \in V. && (2.4g)
\end{aligned}
$$

However, it can be seen that program (P_3) uses $3|V|$ variables and $|V| + 3|E|$ constraints, while program (P_4) uses $2|V| + 3|E|$ variables and $|V| + 5|E|$ constraints. Hence, if forbidding stitch insertions, our 0-1 linear program (P_3) uses less variables and constraints than program (P_4).

We solve the discrete relaxation problem (P_2) to obtain a relaxation solution. And the relaxation solution will be legalized to a feasible solution (M_1, M_2, M_3) of problem (P_0) with stitch insertions. The feasible solution (M_1, M_2, M_3) may not be an optimal solution of problem (P_0). However, by Claim 2.4.4 or Claim 2.4.6, in some cases it will be.

2.5 Legalization

For every independent component G of the conflict graph got after graph reduction proposed in Section 2.6, we obtain a relaxation coloring solution from program (P_2). Then, in order to obtain a final mask assignment of TPL layout decomposition, the discrete relaxation solution should be legalized. For an independent component G, the legalization proceeds as follows.

If in the coloring solution (C_1, C_2, C_3, R_4) of problem (P_2), $R_4 \neq \emptyset$, then to eliminate conflicts, we introduce one-stitch first insertions on the features in R_4. Details of the optimal one-stitch insertion are described in Section 2.5.1. Furthermore, if there exits a feature $i \in UNCF$ which cannot be inserted at most $s_i = 1$ stitch to eliminate conflicts, then the solution obtained from the relaxation coloring solution may not be optimal for problem (P_0). In this case, we propose a backtrack coloring method to obtain another better relaxation solution $(C_1^*, C_2^*, C_3^*, R_4^*)$. And then we consider dichotomy stitch insertion on the new solution. The details are shown in Section 2.5.2.

2.5.1 Stitch Insertion

Due to conflicts, the features in the set R_4 of a relaxation solution are uncolored. And the features might be inserted stitches to eliminate conflicts. Since our objective is that, these features should be colored with the minimum $|C|+\alpha|S|$, we should find stitch insertions into the features in R_4, and then assign their sub-features to the three masks. An algorithm for finding all candidate one-stitch insertions on a feature a is shown as Algorithm 2.1. Furthermore, if we need to insert multiple stitches into feature a, then we execute Algorithm 2.1 on the sub-features of feature a iteratively.

Algorithm 2.1 Candidate one-stitches finding

Input: feature a in R_4, its adjacent features and their colors;
Output: all candidate one-stitch insertions on a;

1: **for** every adjacent feature of feature a **do**
2:     calculate the CRTs on feature a caused by its adjacent features, and store CRTs in $CRTSet$;
3: **end for**
4: **for** every $CRT \in CRTSet$ **do**
5:     CSEs=$genCSE(CRT)$;
6:     store CSEs in $CSESet$;
7: **end for**
8: **for** $CSEs \in CSESet$ **do**
9:     if splitting feature a into two sub-features along CSE satisfies **Conditions 1, 2 and 3**, then store the CSE into candidate stitch set $COSSet$;
10: **end for**

In the algorithm, firstly we find all conflict rectangles (CRTs) on feature a (Line 2). A CRT consists of four edges, each edge of CRT generates a Checking Stitch Edge (CSE). A CSE of feature a is a line segment whose two end-points are on the boundary of feature a, and feature a could be split into two sub-features as long as the split along the CSE satisfies the following three conditions:

Condition 1. Sizes of the generated sub-features should be larger than the minimum feature size $\min_{fs}$;

Condition 2. A candidate stitch insertion is not near a corner of feature a. Overlap length should not be less than the overlap margin $\min_{om}$.

Condition 3. The number of conflict edges connected to generated sub-features is less than the number of conflict edges connected to feature a.

Conditions 1 and 2 are used to make the locations of stitches legal. Condition 3 indicates that inserting a stitch along the CSE will eliminate some conflicts, and then generate a better layout decomposition.

In the algorithm, the function *genCSE(CRT)* refers to generating four CSEs of a conflict rectangle CRT (Lines 4-7). If along a CSE the feature a could be split into two sub-features satisfying the three conditions, then the CSE is a candidate stitch insertion (Lines 8-10).

After obtaining all candidate stitch insertions in $COSSet$, we only keep those candidate stitches which may generate sub-features with the minimum conflict number, and the others are deleted. Then we find an optimal stitch in $COSSet$ for feature a, and split feature a along the optimal stitch into sub-features a_1 and a_2. The optimal stitch is based on a criterion *cut_cost* for evaluating every stitch, which is the increased number of edges of the conflict graph after splitting feature a along the stitch.

Note that, the sub-features a_1 and a_2 of feature a got by splitting feature a along stitch s will be assigned different colors. Suppose that b_i is an uncolored

feature and adjacent to feature a, i.e., $e_{ab_i} \in E$. If $e_{a_1b_i} \in E$ and $e_{a_2b_i} \in E$, then splitting feature a along stitch s will increase the degree of feature b_i by 1, and we let $cut_cost_{b_i} = 1$. So the function $cut_cost(s)$ of stitch s is formulated as

$$cut_cost(s) = \sum_{e_{ab_i} \in E} cut_cost_{b_i},$$

where

$$cut_cost_{b_i} = \begin{cases} 1, & e_{a_1b_i} \in E \text{ and } e_{a_2b_i} \in E; \\ 0, & \text{otherwise.} \end{cases}$$

Obviously, $cut_cost(s) \geq 0$. An optimal one-stitch is with the minimum $cut_cost(s)$. Our empirical experiments indicate that most of features in $UNCF$ have optimal one-stitches with $cut_cost(s) = 0$.

For every feature in R_4, Algorithm 2.1 is used to find all candidate one-stitch insertions, and all optimal one-stitch insertions are found based on the above criterion. After that, the features in R_4 with optimal one-stitch insertions are split along their optimal one-stitches respectively, and the obtained sub-features are colored legally. Then the conflict graph G, the sets C_1, C_2, C_3 and R_4 are updated accordingly. If conditions of Claim 2.4.6 are satisfied, then the obtained coloring solution is optimal for the TPL layout decomposition problem; otherwise, we perform the process of backtrack coloring in Section 2.5.2.

2.5.2 Backtrack Coloring

To obtain a better legal solution, a backtrack coloring algorithm is proposed in this subsection. Similar backtrack method was introduced by Yu et al. [103] to address the TPL aware detailed placement problem, which is fast and effective on small size graphs. Let $x = (C_1, C_2, C_3, R_4)$ be a relaxation coloring solution of independent component G got by program (P_2). We introduce a set $WOSF$ to represent the features in $UNCF$ which cannot be inserted at most $s_i = 1$ stitch to eliminate conflicts. We try to find a better relaxation coloring solution

Algorithm 2.2 Backtrack coloring

Input: independent component G, coloring solution $x = (C_1, C_2, C_3, R_4)$ by program (2), subsets X' and X'' of coloring solution space of G;
Output: another coloring solution x^* of G;

```
1: for every x' = (C'_1, C'_2, C'_3, R_4) ∈ X' do
2:   if there exist one-stitch insertions such that WOSF = ∅ then
3:     x* = x';
4:     break;
5:   end if
6: end for
7: if WOSF = ∅ then
8:   return x*;
9: else
10:  for every x'' = (C''_1, C''_2, C''_3, R''_4) ∈ X'' do
11:    calculate the decomposition cost of x'': cost(x'') = |C| + α|S|;
12:    if find a less cost solution x'' then
13:      if cost(x'') == Σ_{i∈R_4} w_i then
14:        x* = x''; Break;
15:      end if
16:      x* = x'';
17:    end if
18:  end for
19:  return x*;
20: end if
```

$x^* = (C_1^*, C_2^*, C_3^*, R_4^*)$ of G by Algorithm 2.2, and then insert stitches into the features in R_4^*.

In Algorithm 2.2, inputs X' and X'' are two subsets of coloring solutions of program (2.2), which are obtained and stored at the solution stage of program (2.2) by the Branch-and-Bound method. Every solution $x' = (C_1', C_2', C_3', R_4') \in X'$ satisfies that $R_4' = R_4$, here R_4 is a part of the coloring solution x of program (2.2), which is an input of Algorithm 2.2. And every solution $x'' = (C_1'', C_2'', C_3'', R_4'') \in X''$ satisfies that $R_4'' \neq R_4$ and there are no conflict features in R_4''.

Theoretically, the number of solutions in X'' generated by the Branch-and-Bound method may be exponential. However, there are few non-conflict features in a dense layout. Moreover, for a sparse layout, after graph reduction, the size

of every resulting connected component is small. These means that $|X''|$ is not too large and the storage of X'' is tolerable. The situation for X' is similar. To make Algorithm 2.2 faster, we might only store solutions nearing to the coloring solution x of program (2.2) in the Branch-and-Bound process, to X' and X'' respectively.

Algorithm 2.2 firstly scan all solutions in X' of G (Lines 1-6). For every new coloring solution (C_1', C_2', C_3', R_4), we enumerate all candidate one-stitch insertions on features in R_4 by Algorithm 2.1, and check whether the criterion $WOSF = \emptyset$ is satisfied (Line 2). If a solution x^* with $WOSF = \emptyset$ is found, then a condition of Claim 2.4.6 is satisfied, and we stop the backtrack coloring; otherwise, all solutions in X'' will be scanned to find another better one (Lines 10-19), and we have another break criterion to end the scan (Line 13). This criterion means that, if the algorithm has found a solution with $cost(x'') = \sum_{i \in R_4} w_i$, which is the optimal value of Problem (2.2), then it is optimal for problem (1) and it does not need to scan further.

Note that stitch insertions should be found before calculating $cost(x'') = |C| + \alpha|S|$ (Line 11). Here, we consider multiple stitch insertions more than one-stitch insertions for eliminating more conflicts. Inserting multiple stitches into a feature a means that we execute Algorithm 2.1 on the sub-features of feature a iteratively. Main steps for finding candidate multiple stitch insertions are similar to those in Algorithm 2.1, and the details are omitted here. Figure 2.7 shows two examples of backtrack coloring.

2.6 Relaxation Via Graph Reduction

In this section we introduce several graph reduction techniques to reduce the conflict graph to small size subgraphs, and thus achieve a relaxation of the TPL layout decomposition problem.

Using some rule, we remove some vertices in the initial conflict graph

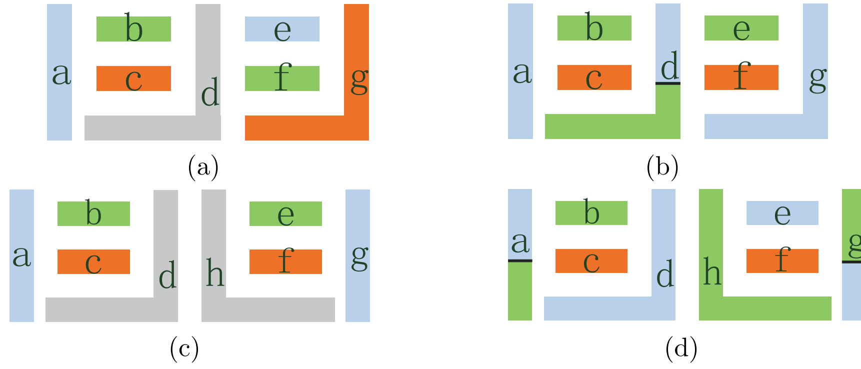


Figure 2.7: Two samples of backtrack coloring. (a)(c) Initial illegal coloring solutions. (b)(d) Feasible solutions after backtracking.

$G(V, E)$, and finally get some small size disconnected subgraphs. After that, we solve the TPL problem on the subgraphs to obtain a mask assignment. Suppose that x is a mask assignment of $G(V, E)$. Obviously, the optimal value of the TPL problem on the subgraphs is not more than the optimal value of the original problem. Hence the mask assignment problem on the subgraphs is a relaxation problem of the mask assignment problem of $G(V, E)$. Theoretical results of this relaxation are similar to those in Section 2.4.1, and are omitted here.

Removed vertices should be selected carefully from the initial conflict graph. A rule is that, after mask assignment of the subgraphs, the removed vertices can be colored easily. The vertex removing techniques in this chapter are listed as follows.

- Vertex with degree less than three removal [36, 53, 105];
- Contained vertex removal.

After vertex removal from the initial conflict graph, independent components must be calculated for constructing subgraphs [36, 53, 105]. Independent components calculation and vertex with degree less than three removal have been used popularly in the previous TPL layout decomposition works. Here,

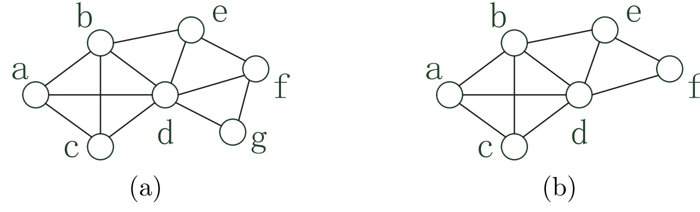


Figure 2.8: An example of vertex removal. (a) The initial graph. (b) The remainder graph after removing contained vertex g.

we focus on the contained vertex removal technique. Firstly we introduce the definition of contained vertex [63].

Definition 2.6.1 (contained vertex)**.** Given a graph $G(V, E)$, for a pair of vertices $i, j \in V$, suppose that $e_{ij} \notin E$ and $A(i) \subseteq A(j)$, where $A(i)$ and $A(j)$ are the set of adjacent vertices of vertices i and j, respectively. We call that vertex i is contained in vertex j, vertex i is called a contained vertex, and j is called a containing vertex.

In Figure 2.8(a), vertex g is a contained vertex, since $e_{ge} \notin E$, and $A(g) = \{d,\ f\} \subseteq A(e) = \{b,\ d,\ f\}$.

We find a contained vertex and its containing vertices in the graph after the vertex with degree less than three removal, and remove it from the graph. This process is repeated until no contained vertex found. The time complexity of removing all contained vertices is $O(d^3|V|)$, where d is the maximum vertex degree of the graph.

The vertex removals are performed before the discrete relaxation coloring stage. After obtaining the mask assignment of the remainder graph, these removed vertices are still uncolored, and need to be assigned to masks.

A contained vertex is prior assigned to the same mask as one of its containing vertices. For the other removed vertices, we color them in the reverse order of removing them at the vertex removal stage. Since the features in the remainder graph are well colored, it is easy to assign colors to the removed vertices. To achieve final coloring, if needed, we will insert stitches into them or perform

backtrack coloring as in Section 2.5.2 again. In all, the removed features will be assigned to appropriate masks with the minimum number of conflicts or stitches.

Note that a removed vertex might have several optional colors during coloring. That is, no matter which color is assigned to the feature, no conflict will be generated. We call this feature as a color-optional-feature. Since a well balanced decomposition is benefit for manufacturing [62, 101], here we consider global density balance for layout decomposition. We define

$$\mathrm{den} = \frac{\max\{A_1, A_2, A_3\}}{\min\{A_1, A_2, A_3\}}$$

as our density balance measurement, where A_k $(k = 1, 2, 3)$ is the total area of features on the $k-$th mask. During the process of removed feature coloring, a color-optional-feature will be assigned a color which may keep the minimal density balance measurement. This strategy benefits to the global density balance of the three masks.

2.7 Experimental Results

In order to evaluate our TPL decomposition method, we tested it on the ISCAS-85 & 89 benchmarks provided by Yu et al. [105]. The circuit sizes of the benchmarks range from 1109 to 168K. The algorithms were programmed in C++ and run on a personal computer with 2.4GHz CPU, 16GB memory and the Linux operating system. For problem (P_2), we adopted the Branch-and-Bound code for nonlinear integer programming in the package GUROBI [3] as our nonlinear 0-1 program solver. For comparing with previous TPL decomposers, we set the parameter values the same as those in previous works. More precisely, the minimum feature size $\min_{fs}$ and the overlap margin $\min_{om}$ were set as 10nm, and the weight parameter α was set as 0.1.

We performed two experiments to adequately demonstrate the effectiveness of our discrete relaxation based decomposition method. Test results of the first

experiment are compared with those of the state-of-the-art TPL decomposers [36, 53, 101]. Test results of the second experiment are compared with those of the state-of-the-art TPL decomposers [36, 105, 109]. Since the binaries of the compared decomposers are not available to us, the test results of the state-of-the-art TPL decomposers are quoted from the respective publications directly.

2.7.1 First Experiment

In this experiment, we tested our decomposition method on the benchmarks with smaller minimum coloring spacing $\min_{cs}$. More precisely, the minimum coloring spacing $\min_{cs}$ was set as $120nm$ for benchmarks C432-C7552, and as $100nm$ for benchmarks S1488-S15850. The test results are listed in Table 2.1.

Table 2.1: Test results of our decomposition method with $\min_{cs} = 120 / 100nm$

Circuits	#IC	#RIC	Ratio	#UCF	#UNCF	#WOSF	Den
C432	123	4	1.50	0	4	0	1.02
C499	175	0	-	0	0	0	1.37
C880	270	7	1.62	0	7	1	1.09
C1355	467	3	1.50	0	3	0	1.13
C1908	507	1	1.50	0	1	0	1.21
C2670	614	7	1.54	0	6	0	1.25
C3540	827	9	1.50	1	8	0	1.16
C5315	1154	9	1.50	0	9	0	1.23
C6288	2325	171	1.55	0	191	16	1.01
C7552	1783	22	1.51	0	22	1	1.18
S1488	274	2	1.50	0	2	0	1.03
S38417	5298	74	1.51	19	54	0	1.00
S35932	15804	84	1.56	44	40	1	1.00
S38584	16235	152	1.51	36	116	1	1.00
S15850	13226	131	1.52	34	97	1	1.00

2.7.1.1 Analysis of Our Test Results

In Table 2.1, the data in the columns "#IC" and "#RIC" are the numbers of independent components of the initial conflict graphs, and the numbers of independent components of the remainder graphs after graph reduction, respectively. The data in the column "ratio" are the ratios between the numbers of

Table 2.2: Experimental result comparisons with $\min_{cs} = 120 / 100nm$

Circuits	From [36]				From [53]				From [101]				Ours			
	#C	#S	Cost	CPU (s)	#C	#S	Cost	CPU (s)	#C	#S	Cost	CPU (s)	#C	#S	Cost	CPU (s)
C432	0	6	0.6	0.01	0	4	0.4	0.01	0	4	0.4	0.2	0	4	0.4	0.01
C499	0	0	0	0.01	0	0	0	0.01	0	0	0	0.2	0	0	0	0.01
C880	1	15	2.5	0.01	0	7	0.7	0.01	0	7	0.7	0.3	0	7	0.7	0.02
C1355	1	7	0.7	0.02	0	3	0.3	0.01	0	3	0.3	0.3	0	3	0.3	0.02
C1908	1	0	0	0.04	0	1	0.1	0.01	0	1	0.1	0.3	0	1	0.1	0.04
C2670	2	14	3.4	0.06	0	6	0.6	0.04	0	6	0.6	0.4	0	6	0.6	0.06
C3540	2	15	3.5	0.08	1	8	1.8	0.05	1	8	1.8	0.5	1	8	1.8	0.07
C5315	3	11	4.1	0.11	0	9	0.9	0.05	0	9	0.9	0.7	0	9	0.9	0.11
C6288	19	341	53.1	0.13	14	191	33.1	0.25	1	213	22.3	2.7	0	204	20.4	0.16
C7552	3	46	7.6	0.17	0	22	2.2	0.1	0	22	2.2	1.1	0	22	2.2	0.17
S1488	0	4	0.4	0.03	0	2	0.2	0.01	0	2	0.2	0.3	0	2	0.2	0.03
S38417	20	122	32.2	0.62	19	55	24.5	0.42	19	55	24.5	7.9	19	54	24.4	0.60
S35932	46	103	56.3	2.13	44	41	48.1	0.82	44	48	48.8	21.4	44	40	48.0	1.70
S38584	36	280	64.0	2.26	36	116	47.6	0.77	37	118	48.8	22.2	36	116	47.6	1.81
S15850	36	201	56.1	2.14	36	97	45.7	0.76	34	101	44.1	20	34	97	43.7	1.73
Avg.	11.3	77.7	19.0	0.52	10.1	37.4	13.75	0.22	9.07	39.8	13.05	5.23	8.9	38.2	12.75	0.44
Ratio			1.49	1.19			1.08	0.50			1.02	10.06			1.00	1.00

edges and the numbers of vertices in the remainder independent components, respectively. From Table 2.1, the data in the column "#RIC" are small, compared with the data in the column "#IC". Specifically, the average #RIC is only 1.1% of #IC. Hence we can conclude that our graph reduction methods are effective.

The data in the column "#UCF" of Table 2.1 are the numbers of uncolored conflict features by our algorithm on the remainder graphs of the benchmarks. By Lemma 2.4.2, we know that #UCF is a lower bound on the minimum conflict number of TPL. The data in the column "#C" of Table 2.2 are the total conflict numbers found by our method on the benchmarks. Comparing #UCF with #C, it is obvious that both of them are equal for every benchmark. Hence, our decomposition method found results with the minimum conflict numbers for these benchmarks. Moreover, if #UCF=#C, then every feature in UCF contributes at most a conflict number.

The data in the columns "#UNCF" and "#WOSF" of Table 2.1 are the numbers of uncolored non-conflict features, and the numbers of uncolored non-conflict features without one-stitch insertion, respectively. Table 2.1 shows that all benchmarks C432-C5315 are with #WOSF=0 except C880, and benchmarks S1488 and S38417 are with #WOSF=0. Since #UCF=#C for benchmarks C432-C5315 except C880 and #UCF=#C for benchmarks S1488 and S38417, every feature in UCF contributes at most a conflict number for these benchmarks. Hence according to Claim 2.4.6, we have obtained a minimum cost solution, i.e., we have achieved optimal decomposition costs for these benchmarks.

From the column "Den" in Table 2.1, it can be seen that most of benchmarks have Den close to 1, i.e., the total areas of the obtained three masks are almost equal. That is, the densities of our results are well balanced.

2.7.1.2 Comparing with Other TPL Decomposers

In Table 2.2, we list the test results of our decomposition method and the state-of-the-art TPL decomposers [36, 53, 101], on the benchmarks C432-C7552

and S1488-S15850 with $\min_{cs} = 120/100nm$. In the table, the data in the columns “#C” and “#S” denote the conflict numbers and the stitch numbers of the final results, respectively. And the data in the column “Cost” are calculated in the same way as that in problem (P_1) and references [36, 53, 101]. The data in the columns “CPU(s)” are the running times of our decomposition method and the decomposers [36, 53, 101], respectively.

We compare in Table 2.2 our test results with those of decomposers in [36, 53, 101]. From the table, we can see that our method achieves the best TPL decomposition result for every benchmark among the compared decomposers. The last row in Table 2.2 lists the average #C, #S, Cost and runtime; and lists the Cost, runtime ratios based on the results of our TPL layout decomposition method. Comparing with the results achieved by Kuang et al. [53], which is the fastest decomposer, we reduce the average conflict number by 12%, and the average cost by 8%. For the state-of-the-art decomposer proposed by Yu et al. [101], the average cost is 2% more than ours, and the running time is 9 times more than ours. These comparisons validate the effectiveness of our decomposition method.

It must be remarked that the compared decomposers were run on different platforms. In [101], the platform is a personal computer with 3.0GHz CPU and 32GB RAM; in [36], the platform is a personal computer with 2.93GHz CPU and 48GB RAM; in [53], the platform is a personal computer with 2.39GHz CPU and 48GB RAM. But our platform is a personal computer with 2.40GHz CPU and 16GB RAM, which is almost the worst.

2.7.1.3 Scalability of Our Method

To analyze scalability of our decomposition method, Figure 2.9 presents the relationship between the number of features and the running time of our decomposition method. The figure is based on our computational results on the ISCAS-85 & 89 benchmarks with $\min_{cs} = 120nm$. In the figure, the bottom

Table 2.3: Experimental result comparisons on dense layouts with $\min_{cs} = 120nm$

Circuits	From [36]				From [105]				From [109]					Ours			
	#C	#S	Cost	Im%	#C	#S	Cost	Im%	#C	#S	Cost	Im%	CPU(s)	#C	#S	Cost	CPU(s)
S1488	0	4	0.4	50.0	1	1	1.1	81.8	0	30	3.0	93.3	2.15	0	2	0.2	0.04
S38417	43	112	54.2	32.3	61	54	66.4	44.7	47	271	74.1	50.5	45.1	30	67	36.7	0.77
S35932	108	117	119.7	26.8	127	67	133.7	72.5	119	608	179.8	51.3	122	79	86	87.6	2.19
S38584	120	251	145.1	31.1	146	113	157.3	36.4	124	689	192.9	48.2	114	85	150	100.0	2.28
S15850	85	254	110.4	29.3	123	128	135.8	42.6	96	772	173.2	55.0	127	63	150	78.0	2.12
Avg.				33.9				55.6				59.6					

Table 2.4: Experimental result comparisons on dense layouts with $\min_{cs} = 160nm$

Circuits	From [36]				From [105]				From [109]					Ours			
	#C	#S	Cost	Im(%)	#C	#S	Cost	Im(%)	#C	#S	Cost	Im(%)	CPU(s)	#C	#S	Cost	CPU(s)
C432	94	12	95.2	19.9	89	11	90.4	15.3	79	20	81.0	5.8	2.18	73	33	76.3	0.90
C499	350	17	351.7	19.5	324	29	326.9	13.4	289	49	293.9	3.7	6.15	274	90	283.0	5.49
C880	230	36	233.6	45.0	193	16	194.6	34.1	136	90	145.0	11.5	4.50	116	123	128.3	3.87
C1355	227	50	232.0	41.7	193	29	195.9	30.8	135	87	143.7	5.8	7.13	123	123	135.3	3.55
C1908	287	51	292.1	40.9	210	30	213.0	19.0	176	93	185.3	6.9	10.7	163	95	172.5	2.38
C2670	810	122	822.2	40.1	633	57	638.7	22.9	473	238	496.8	0.8	21.4	463	297	492.7	12.3
C3540	810	156	825.6	46.4	732	62	738.2	40.0	489	349	523.9	15.5	25.2	394	487	442.7	5.85
C5315	1313	201	1333.1	33.7	1181	94	1190.4	25.7	933	404	973.4	9.2	38.1	830	529	883.9	10.5
C6288	879	219	900.9	29.2	816	40	820.0	22.3	731	293	760.3	16.2	40.6	603	344	637.4	11.1
C7552	1585	380	162.3	30.8	1275	506	1325.6	15.2	1100	642	1164.2	3.5	55.9	1054	697	1123.7	17.3
S1488	615	108	625.8	24.3	602	34	605.4	21.8	443	215	464.5	-2.0	10.4	446	277	473.7	27.8
S38417	7748	534	7801.4	40.4	4908	952	5003.2	7.1	4473	2408	4713.8	1.4	197	4394	2524	4646.4	66.5
S35932	23767	1404	23907.4	39.6	14412	3120	14724.0	1.9	13101	7137	13814.7	-3.8	722	13751	6921	14443.1	813
S38584	20106	1392	20235.2	44.7	11622	2639	11885.9	5.8	11187	6097	11796.7	5.1	654	10564	6301	11194.1	102
S15850	22561	1915	22752.5	37.8	15196	2613	15457.3	8.4	13405	6997	14104.7	-0.3	713	13425	7282	14153.2	591
Avg.				35.6				18.9				5.3					

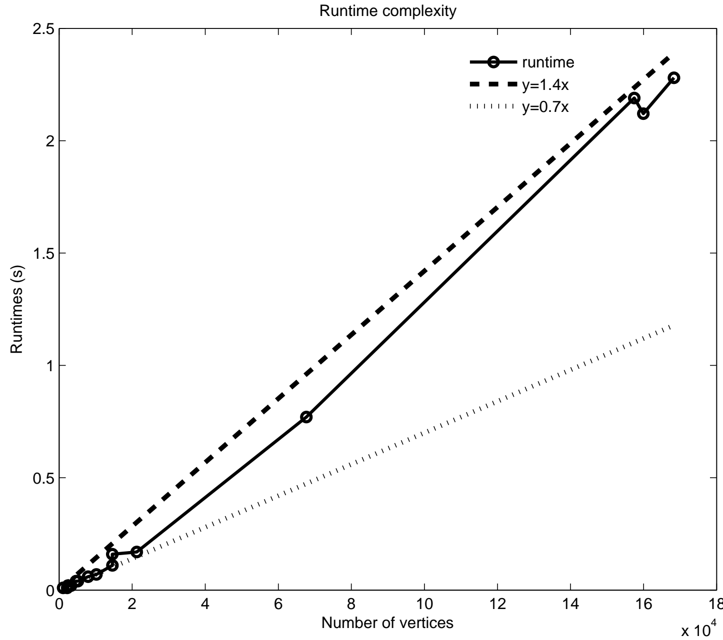


Figure 2.9: Running time vs the number of vertices.

dotted line is the plot of function $y = 0.7x$; the up dotted line is the plot of function $y = 1.4x$; and our runtime picture is the middle solid line. This fully illustrates that our discrete relaxation based decomposition method is almost a linear-time decomposer for the tested benchmarks.

2.7.2 Second Experiment

To further evaluate effectiveness and scalability of our discrete relaxation based decomposition method, we performed additional experiments of testing our method on the benchmarks S1488-S15850 with $\min_{cs} = 120nm$, and on the ISCAS-85 & 89 benchmarks with minimum coloring spacing $\min_{cs} = 160nm$. It is obvious that the numbers of conflict edges of these graphs are more than those in the first experiment, and the conflict graphs of these benchmarks are more dense. The test results of our decomposition method and the compared decomposers are listed in Table 2.3 and Table 2.4 respectively, where "Im(%)" denotes the cost reduction percentage by our method comparing with the corresponding method. The data in the columns "CPU(s)" are the runtimes of the algorithm

in [109] and our method, respectively. Since the binaries of [105] and [36] are not available to us, we do not list their runtimes.

From Table 2.3, comparing with [36], [105] and [109], our method achieves considerably cost reduction for every benchmark. Averagely, the cost reduction percentages by our method comparing with the corresponding methods in [36], [105] and [109] are 33.9%, 55.6% and 59.6%, respectively. From Table 2.4, the total costs of benchmarks S1488-S15850 by our decomposition method are larger than those in Table 2.4 respectively, due to the dense structures. The last row of Table 2.4 lists the average improvements of the cost reduction percentage by our method comparing with the corresponding methods in [36], [105] and [109], which are 35.6%, 18.9% and 5.3%, respectively. Comparing runtime between [109] and our approach, it can be found that, our approach is faster than the method in [109] on most of the benchmarks, especially on the sparse layouts. It must be remarked that the platform in [109] is a personal computer with 2.66GHz and 4GB RAM, which is better than ours.

Finally, for the TPL layout decomposition problem, it is obvious that $\min_{cs}$ is a main parameter to control the density of the conflict graph. In this experiment, we have achieved greater improvement than the first experiment, in which the conflict graph is sparser. This demonstrates that our discrete relaxation based decomposition method is more effective on the dense layouts.

2.8 Summary

In this chapter, we have proposed a discrete relaxation theory, and have developed a discrete relaxation based decomposition framework for the TPL layout decomposition problem. Although the line projection method can construct the conflict graph of the problem, we have developed a surface projection method for identifying features which are critical and should be colored prior, and this forms a basis of our discrete relaxation method.

To solve the TPL layout decomposition problem, our discrete relaxation based decomposition method relaxes the problem in two steps. Firstly, the conflict graph is reduced to small size subgraphs by two graph reduction techniques, which is a discrete relaxation of the TPL problem. After that, the TPL problem on the small subgraphs is relaxed to a nonlinear 0-1 programming problem by ignoring stitch insertions and assigning weights to features. To legalize an optimal solution of the relaxation problem to a feasible one of the TPL layout decomposition problem, some techniques have been carefully adopted, e.g., the one-stitch first insertion, backtrack coloring. Experiments on the tested benchmarks show that our decomposition method is efficient and effective, compared with the state-of-the-art decomposers. Moreover, by our theoretical results, we have obtained optimal decompositions for some benchmarks. The developed discrete relaxation based decomposition method for TPL is successful. We believe the idea can be applied to other problems.

Chapter 3 Two-Stage Decomposition for Hybrid E-Beam and Triple Patterning Lithography

3.1 Introduction

The major bottleneck that hinder faster and more powerful processor development is the design and manufacture technologies of integrated circuit (IC). At present, many manufacture technologies have been developed [86], such as the 193nm ArF immersion optical lithography (ArF-IOL) and the related multiple patterning lithography (MPL), electron beam lithography (EBL), directed self-assembly (DSA), and extreme ultraviolet lithography (EUVL). EUVL is considered as a promising technology for next-generation lithography. However, due to various obstacles, EUVL still cannot be put into mass IC manufacture [18]. Currently, MPL is among the most popular, which is high throughput and low optimal exposure [13]. On the contrary, E-Beam lithography has lower throughput but good for random complex patterns [19]. Since a solo lithography technology cannot achieve the measures (cost, throughput, timing et al.) very well simultaneously, hybrid lithography is introduced recently. One of the most promising technologies is the combination of MPL with EBL, which is a novel and practical choice for manufacture of IC [78]. Comparing with a solo lithography technology, hybrid lithography can produce better quality circuit board [82, 96].

LELELE style lithography, i.e., triple patterning lithography (TPL), is one of MPL that has been proposed for quite a few years. In order to obtain high resolution, many methods have been proposed for TPL layout decomposition (TPLLD). For general layout, Ref. [105] described the TPLLD problem and proved that it is NP-hard. To solve the problem, they introduced a semi-

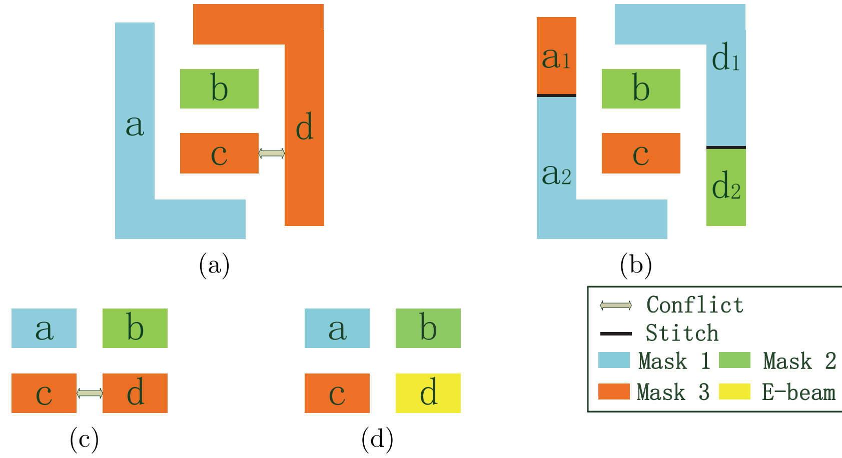


Figure 3.1: Hybrid e-beam and triple patterning lithography layout decomposition. (a) A mask assignment for triple patterning lithography layout decomposition with a conflict. (b) An example of layout with stitch insertion for eliminating the conflict. (c) A mask assignment for TPL layout decomposition with a conflict. (d) A mask assignment for HETLD.

definite programming relaxation based decomposition method. In [36, 53, 59], the authors proposed different heuristic methods to obtain decomposition solutions fast. Specially, for standard cells with a row structure layout, decomposers in [22, 80, 103] have considered layout decomposability in physical design stages.

Compared with MPL, EBL is a flexible lithography technique, which prints patterns by mass electron beams. The conventional electron beam is a variable-shaped beam (VSB), which is a rectangle based e-beam, and can only print rectangular patterns. As a result, EBL is low throughput. To raise throughput, many methods [66, 104, 108] have been proposed for the character-shaped beam (CSB). However, the character-shaped beam technique cannot substantially resolve the throughput issue of EBL. Hence the use of EBL should be kept as less as possible.

Figure 3.1 shows an example of hybrid e-beam and triple patterning lithography layout decomposition. Suppose patterns a, b, c and d in Figure 3.1(a) and Figure 3.1(c) are too close to each other. In Figure 3.1(a), patterns are assigned to three masks, but a conflict occurs between patterns c and d. To

eliminate the conflict, two stitches are inserted in patterns a and d to split them into two sub-patterns respectively, as shown in Figure 3.1(b). However, for some dense layouts, stitch insertion may not eliminate conflicts. As shown in Figure 3.1(c), there is a conflict between patterns c and d, and the conflict cannot be eliminated by inserting stitches. A manufacture plan is shown as Figure 3.1(d), in which pattern d is printed by e-beam.

Over the years, some works have been done for hybrid lithography layout decomposition. For 1D layout structure, Du *et al.* [35] constructed a mathematical formulation and proposed an iterative ILP algorithm to assign cuts for hybrid lithography with e-beam and 193nm immersion (193i) single exposure. For 2D layout, Ding *et al.* [31] investigated the layout decomposition for hybrid self-aligned double patterning lithography (SADP) and EBL by solving an elegant ILP formulation; Gao *et al.* [42] considered simultaneously the e-beam cut cost and the trim cut cost, and introduced a matching method for hybrid SADP and EBL layout decomposition.

Hybrid EBL and TPL was firstly investigated by Tian *et al.* [82]. Their method is only for layouts with standard cells on rows. Recently, Yang *et al.* [96] considered the hybrid EBL and TPL of general layout decomposition problem (HETLD), and proposed a random-initialized improvement local search method basing on their hybrid EBL and double patterning lithography decomposer. Before decomposition, their method divides every pattern into several rectangles using candidate stitches. This operation leads to the following two issues: i) it would increase the size of the decomposition problem; ii) since candidate stitches are inserted at the corners of the patterns, the locations of the stitches are illegal due to causing side-effect [36, 53, 59, 105] and increasing the manufacturing cost.

The HETLD problem looks like the TPLLD problem, however, there are many differences between the HETLD problem and the TPLLD problem. First, the TPLLD problem uses stitch insertion for eliminating conflicts, while the HETLD problem uses e-beam and stitch insertion. Second, the primary objec-

tive of the HETLD problem is minimizing the sum of VSB numbers, while the TPLLD problem is minimizing the total number of conflict edges. Furthermore, when decomposing a layout for the HETLD problem, if a pattern is assigned to e-beam, then stitch cannot be used for reducing the VSB number of the pattern. While for the TPLLD problem, if a pattern conflicts with some other patterns, then stitch still can be used to reduce the number of conflicts. Finally, for the HETLD problem, if a pattern is assigned to e-beam, then the pattern has no conflicts with other patterns. Hence, the discrete relaxation based decomposition method for the TPLLD problem [59] cannot be used directly for the HETLD problem.

In this chapter, we consider the hybrid e-beam and TPL of general layout decomposition problem. We propose a two-stage decomposition flow for the problem. At the first stage, we consider an e-beam and stitch aware TPL mask assignment (ESTMA) problem, and then the problem is formulated as a binary linear program and solved by the cutting plane approach. At the second stage, the solution is legalized to a feasible solution of the HETLD problem by stitch insertion and e-beam shot. In addition, some graph reduction techniques proposed by previous TPL layout decomposers [36, 53, 59, 105] are used to reduce the problem size. Moreover, a new graph reduction which deletes some minor conflict edges is proposed to further speed up the decomposition flow. Furthermore, in order to obtain a better solution with less VSB number, we propose an extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) problem, which is also formulated as an ILP. In the first stage, if we solve the MDSR_4MA problem instead of the ESTMA problem, then more patterns can be assigned to TPL masks by inserting stitches. Experimental results show the effectiveness of the ESTMA and the MDSR_4MA based decomposition methods. In addition, it must be noted that the two issues in [96] are avoided in this chapter.

3.2 Preliminaries

In this section, first we describe the hybrid EBL and TPL of general layout decomposition (HETLD) problem, and then we introduce some concepts and analyze properties of the HETLD problem.

3.2.1 Hybrid E-Beam and TPL Layout Decomposition Problem

Given a layout L, the minimum coloring spacing $\min_{cs}$ rule is that, if the distance between two patterns is less than $\min_{cs}$, then there exists a conflict between the two patterns. The hybrid e-beam and TPL layout decomposition (HETLD) problem is that, the patterns in L are assigned to three TPL masks with stitch insertions to eliminate most of the conflicts, and to totally eliminate conflicts, e-beam shots are used to print some patterns.

The HETLD problem can be seen as that, all patterns in L are assigned to four different colors, i.e., three colors for TPL and one color for EBL. For the three colors for TPL, stitch insertion can be used to split patterns into several sub-patterns, and the distance between any two patterns or sub-patterns in the same TPL color should be greater than $\min_{cs}$.

Since stitch will lead to potential functional errors of a chip, and increase the manufacture cost [82], the number of stitches should be minimized. In addition, EBL system mainly applies the variable shaped beam (VSB) technique to print patterns, which is low throughput due to one-by-one print process [67, 104]. Hence, to maximize the throughput, the number of VSB shots should be minimized. It must be noted that VSB is a rectangle-shaped electron beam, hence a pattern should be split into a set of rectangles for being printed by VSB shots [31, 35, 41, 48]. That is, the number of rectangles of patterns printed by e-beam is equal to the number of VSBs [82]. Thus, in the hybrid TPL and EBL layout decomposition, the number of VSBs and the number of stitches should be minimized for low cost and high throughput of manufacture. Formally, the

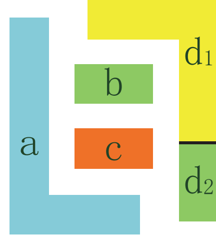


Figure 3.2: An illegal mask assignment for HETLD.

HETLD problem is described as follows.

Hybrid e-beam and triple patterning lithography layout decomposition problem P_0:

Given: Layout L, the minimum coloring spacing $\min_{cs}$, the minimum pattern size $\min_{ps}$, and the minimum overlap margin $\min_{ov}$.

Find: A color assignment of patterns in layout L to three colors for TPL with stitch insertions and one color for EBL, subject to: i) any two patterns or sub-patterns within $\min_{cs}$ should not be assigned to the same TPL color; ii) a pattern in L should be printed only by TPL or EBL; and iii) the location of stitch insertion should be legal.

Objective: Mainly minimize the number of VSBs, i.e., $|VSB|$, and secondly minimize the number of stitches, i.e., $|S|$.

In the problem, the second constraint means that, sub-patterns of a pattern should be produced by the same manufacture technique, i.e., either by TPL or by EBL. This is due to that, producing a pattern using two different manufacture techniques will induce greater manufacture cost [96]. Here, we show an illegal mask assignment as Figure 3.2. In the figure, sub-pattern d_1 is printed by e-beam shot, but d_2 is printed by TPL exposure.

For the location of an inserted stitch, it should satisfy that [59]: i) a generated sub-pattern must be larger than the minimum pattern size $\min_{ps}$; ii) the location of an inserted stitch should not be near any corner of a pattern; and iii) the overlap length should be greater than the minimum overlap margin $\min_{om}$.

Here the overlap length means that, an inserted stitch can be moved within some range without causing any new conflict, and the length of the range is called the overlap length [53].

To show complexity of the HETLD problem, first we consider the minimum weight vertex removal 3-coloring (MWVR3C) problem. The decision problem of the MWVR3C problem is that, given a vertex-weighted undirected graph $G(V, E)$ and a constant K, we ask if there exists a subset V' of V, such that the sum of weights of vertices in V' is less than or equal to K, and the induced subgraph $G[V - V']$ of G is 3-colorable. Since deciding whether a graph is 3-colorable is NP-complete [43], the MWVR3C problem is NP-hard. Furthermore, the MWVR3C problem can be reduced easily to the HETLD problem. The detail of reduction is the same as that in [105], in which the planar graph 3-coloring problem is reduced to the triple patterning layout decomposition problem. Hence, the HETLD problem is NP-hard, which means that it cannot be solved in polynomial time unless P=NP.

3.2.2 Conflict Pattern and Native Conflict

Given a layout L, according to the minimum coloring spacing $\min_{cs}$ rule, we transform the geometric layout structure to a conflict graph $CG(V, E)$, where V is the set of patterns, and E is the set of conflict edges between any two patterns. In the conflict graph CG, two patterns where a conflict edge exists should be assigned to different masks. However, there might be some patterns which cannot be assigned to different masks to eliminate conflicts. They should be inserted stitches or assigned to e-beam. Furthermore, there might be conflicts at some patterns which cannot be totally eliminated by inserting stitches. Some of these patterns should be assigned to e-beam.

We present a graph structure where conflicts between patterns cannot be resolved by inserting stitches. Before that, some definitions from [59] are introduced as follows.

Definition 3.2.1 (conflict region, CR)**.** The conflict region of a pattern is defined as a 2D region around the pattern, which is within the minimum coloring spacing $\min_{cs}$ of the pattern.

Figure 3.3(a) shows an example of conflict region, where the round rectangle region of pattern c is the conflict region of c. And the red dashed box in Figure 3.3(a) is the intersection of pattern d and CR of pattern c.

As we know, K_4 is the smallest 3-uncolorable structure, and a 3-uncolorable graph would generate conflicts for TPL layout decomposition. Actually, for TPL, most of conflicts would be generated from K_4. The K_4 structure is introduced as follows.

Definition 3.2.2 (conflict pattern, CP)**.** Pattern v is called a conflict pattern if it satisfies the two conditions: i) on pattern v there is an intersection of conflict regions of three other different patterns; ii) the sub-graph induced by pattern v and the three patterns is a K_4 graph. The three patterns are called conflict adjacent patterns CAP of pattern v.

Definition 3.2.3 (K_4 conflict structure, K_4CS)**.** A graph structure is a K_4 conflict structure if: i) it is a K_4 structure; ii) all of the four patterns are conflict patterns CP; and iii) the four patterns are CAP each other.

As Figure 3.3(b) shows, pattern d is a CP, since there is a red box on pattern d, which is the intersection of conflict regions of patterns a, b and c. Actually, patterns a, b, and c also are CP, and the four patterns compose a K_4. Furthermore, the four patterns are CAP each other. Thus, the structure composed by patterns a, b, c and d is a K_4CS. For comparison, we show a non-conflict pattern NCP as the pattern a in Figure 3.3(f), in which pattern a belongs to a K_4 graph, but on pattern a there does not exist an intersection of conflict regions of patterns b, c and d.

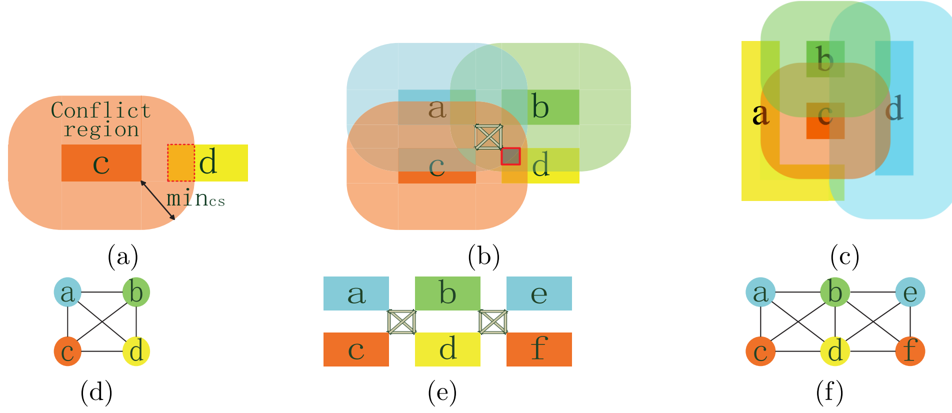


Figure 3.3: Conflict region and conflict patterns. (a) Conflict region. (b) Conflict pattern. (c) An example of non-conflict pattern. (d) K_4 conflict structure. (e)(f) Native conflict structure.

Definition 3.2.4 (native conflict structure, NCS)**.** A native conflict structure NCS consists of one or more connected K_4 conflict structures. A K_4CS is the smallest NCS.

By Definition 3.2.4, the structures in Figure 3.3(b) and Figure 3.3(d) are native conflict structures NCS. Figure 3.3(b) consists of one K_4CS $\{a, b, c, d\}$, and Figure 3.3(d) consists of two K_4CSs $\{a, b, c, d\}$ and $\{b, e, d, f\}$, since $\{a, b, c, d\}$ and $\{b, e, d, f\}$ are connected at vertices b and d.

Next, we show a relationship between NCS and e-beam as follows.

Theorem 3.2.5. At least a pattern in K_4CS should be printed by e-beam.

Proof. Since K_4CS is a K_4 structure, it is not 3-colorable, and there is at least a conflict between two of the four patterns after coloring. Moreover, from [59], we know that the conflicts between the four patterns cannot be totally eliminated by stitch insertions. Hence at least a pattern in K_4CS cannot be printed by TPL, which must be printed by e-beam. □

According to the definition of NCS and Theorem 3.2.5, we have a corollary as follows.

Corollary 3.2.6. At least a pattern in NCS should be printed by e-beam.

Given a layout, we cannot distinguish which conflict can be eliminated by stitch insertion, but Theorem 3.2.5 and its corollary provide sufficient conditions for finding patterns which should be printed by e-beam. The two sufficient conditions are critical for our e-beam and stitch aware TPL mask assignment in the next section. Hence, before coloring, it is significant to check the conflict patterns and native conflict structures, by which we can find potential unresolvable conflicts in a layout.

Checking the native conflict structures in a layout can be done by the BFS algorithm, which traverses all vertices, and finds all conflict patterns. For a conflict graph $CG(V, E)$, suppose $D = \max\{d_v, v \in V\}$ is the maximum degree of vertices, then the computing time of determining whether there exists a structure containing a pattern v is K_4 is in time $O(D^3)$. It is easy to know that the runtime complexity of checking all NCS in $CG(V, E)$ is $O(D^3 * |V|)$.

3.3 Hybrid E-Beam and TPL Mask Assignment Methods

In this section, we introduce the first layout decomposition stage. For the hybrid e-beam and TPL layout decomposition (HETLD) problem, the minimum number of VSBs is the primary objective, and the minimum number of stitches is the secondary objective. For a large scale case of the HETLD problem, it is not good that every pattern in a layout is split into several sub-patterns by candidate stitches before solving the problem, since this would increase the size of the problem. Hence, we introduce two mask assignment methods: (1) the e-beam and stitch aware TPL mask assignment (ESTMA) method; (2) the extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) method. The two methods consider implicitly the e-beam and stitch insertion in the first decomposition stage, and the concrete e-beam and stitch insertion will be considered in the second decomposition stage. Some involved notations

are introduced as follows:

- V, the set of patterns in a conflict graph;
- E, the set of conflict edges in a conflict graph;
- W, the set of weights of patterns in a conflict graph;
- VSB_i, the number of VSBs (or the number of rectangles) of pattern i;
- C_1, C_2, C_3, the colors (masks) of TPL;
- R_4, the set of uncolored patterns;
- β, the weighting parameter between VSB and stitch numbers, which is set as $\beta = 0.01$ as in [96].

3.3.1 E-Beam and Stitch Aware TPL Mask Assignment (ESTMA)

For the HETLD problem, e-beam and stitch insertion can be seen as two kinds of conflict eliminating techniques, where the cost of e-beam is higher than that of stitch insertion. Hence, if we can distinguish which pattern will use stitch insertion to eliminate conflicts, and which pattern must use e-beam to eliminate conflicts, then the HETLD problem can be well addressed. However, we do not know about that before coloring. In this subsection, we introduce the e-beam and stitch aware TPL mask assignment (ESTMA) problem by assigning weights to all patterns. The objective of the ESTMA problem is minimizing the sum of weights of uncolored patterns, which implies minimizing the numbers of VSBs and stitches.

According to the analysis in Section 3.2.2 for conflict pattern CP and K_4 conflict structure K_4CS, we know that for a CP, stitch insertion is almost useless for eliminating all conflicts, unless some of its conflict adjacent patterns CAP are assigned to e-beam or stitches are inserted into its CAP. However, if

a pattern is not a CP, stitch insertion is more likely to eliminate all conflicts. Thus, the weights of patterns are set as

$$w_i = \begin{cases} VSB_i, & \text{if } i \text{ is a conflict pattern;} \\ \beta VSB_i, & \text{if } i \text{ is not a conflict pattern.} \end{cases}$$

We divide patterns into four color classes without considering e-beam and stitch insertion directly. The objective is minimizing the sum of weights of patterns in R_4, i.e., $\sum_{i \in R_4} w_i$. Thus, the e-beam and stitch aware TPL mask assignment (ESTMA) problem is formulated as

$$\begin{aligned} \min \quad & \sum_{i \in R_4} w_i && (3.1) \\ \text{s.t.} \quad & \text{if } i,\ j \in C_k,\ \text{then } i \notin A(j),\ k = 1, 2, 3; && (3.1a) \\ & i \in C_1 \cup C_2 \cup C_3 \cup R_4. && (3.1b) \end{aligned}$$

In the above formulation, $A(j)$ is the set of adjacent patterns of j in the conflict graph CG; $i \in C_k$ means pattern i is assigned to color k, $k = 1, 2, 3$; and $i \in R_4$ means pattern i is uncolored. Constraint (3.1a) is used to force that any two touch patterns should be assigned different colors.

In order to solve Problem (3.1) and obtain a solution, we formulate it as a binary linear program (BLP). Let (x_{i1}, x_{i2}) be a two dimensional binary variable, which is used to represent the color of vertex i. When $(x_{i1}, x_{i2}) = (0, 1)$, it means $i \in C_1$; similarly, when $(x_{i1}, x_{i2}) = (1, 0)$, it means $i \in C_2$; and when $(x_{i1}, x_{i2}) = (1, 1)$, it means $i \in C_3$. However, when $(x_{i1}, x_{i2}) = (0, 0)$, it means $i \in R_4$. Then Problem (3.1) is equivalent to the following binary problem (3.2).

$$
\begin{aligned}
\min \quad & \sum_{i \in V} w_i y_i && && (3.2)\\
\text{s.t.} \quad & x_{i2} - x_{i1} + x_{j2} - x_{j1} \le 1, && \forall e_{ij} \in E; && (3.2a)\\
& x_{i1} - x_{i2} + x_{j1} - x_{j2} \le 1, && \forall e_{ij} \in E; && (3.2b)\\
& x_{i1} + x_{i2} + x_{j1} + x_{j2} \le 3, && \forall e_{ij} \in E; && (3.2c)\\
& 1 - x_{i1} - x_{i2} \le y_i, && \forall i \in V; && (3.2d)\\
& (x_{i1}, x_{i2}) \in \{0,1\}^2, y_i, \in \{0,1\}, && \forall i \in V. && (3.2e)
\end{aligned}
$$

In the above equation, Constraints (3.2a)-(3.2c) is equivalent to constraint (3.1a). That is, any two patterns within the same color class C_k are not conflicting, $k = 1, 2, 3$. Constraint (3.2d) is used to force that, if $x_{i1} = 0$ and $x_{i2} = 0$ then $y_i = 1$; otherwise $y_i = 0$, since the objective is minimization and $w_i > 0$.

3.3.2 Extended Minimum Weight Dominating Set for R_4 Mask Assignment (MDSR_4MA)

According to the weighting rule for the ESTMA Problem (3.1), it can be seen that for a solution of Problem (3.1), if a conflict pattern i is assigned to R_4, then VSB_i will be added to the objective value. Actually, the CPs in R_4 might be assigned to TPL masks using stitch insertion, and then the total number of VSBs will decrease. We take an example to show this as follows.

For the layout given in Figure 3.4(a), all patterns are CP, and an optimal solution of the ESTMA problem (3.1) is shown in Figure 3.4b, where conflict patterns c and g are assigned to R_4. In the second decomposition stage, since patterns c and g cannot be inserted stitches to eliminate conflicts, they are assigned to e-beam as in Figure 3.4(c). Then the total decomposition cost of the HETLD problem is the number of VSBs, i.e., $|VSB| = 2$. However, there exists a feasible solution of Problem (3.1) as shown in Figure 3.4(d), where conflict patterns a and g are assigned to R_4. For this solution, if pattern g is

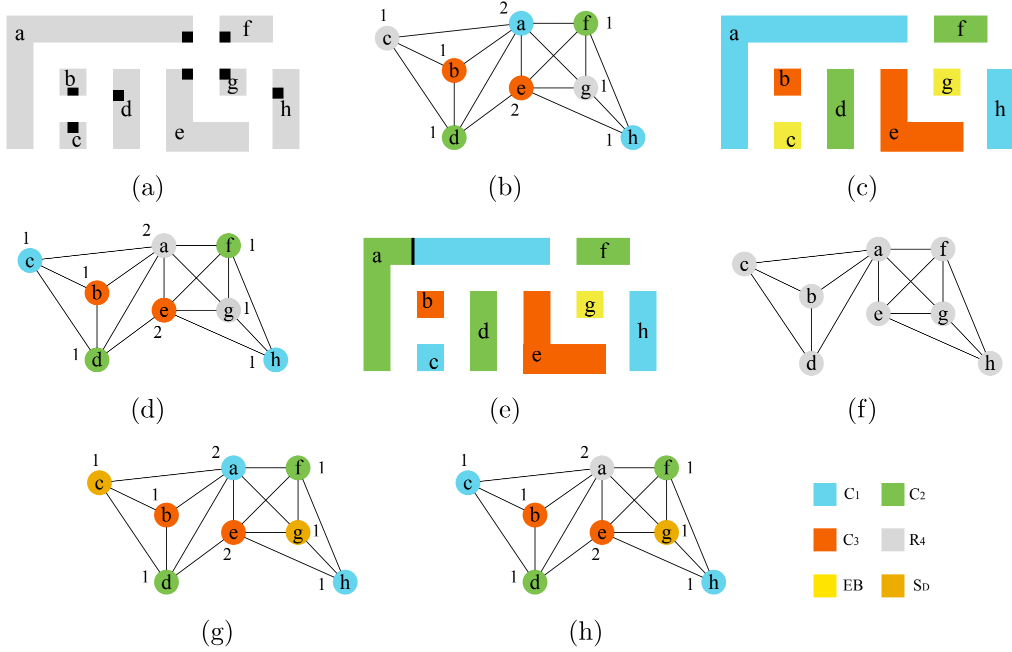


Figure 3.4: A comparison of the ESTMA problem and the MDSR_4MA problem. (a) A layout where all patterns are CP. (b) A feasible solution of the ESTMA problem. (c) The decomposition result of (b)(g). (d) A feasible solution of the ESTMA problem. (e) The decomposition result of (d)(h). (f) The set of conflict adjacent edges E_{ca}. (g)(h) Two feasible solutions of the MDSR_4MA problem.

assigned to e-beam, and pattern a is inserted one stitch for coloring in the second stage, then $|VSB| = 1$, $|S| = 1$, and the total cost of the HETLD problem is $1 + \beta$, which is smaller than the solution shown in Figure 3.4(b).

Note that, for a feasible solution of problem (3.1), if a conflict pattern i is in R_4, then it cannot be inserted stitches for TPL color assignment, unless some of its conflict adjacent patterns CAPs are assigned to e-beam. The reason is that, if at least one of its CAPs are assigned to e-beam, then stitches might be inserted into i for assigning the sub-patterns of i to TPL colors. Thus, comparing with assigning some irrelevant CPs to R_4, like c and g in Figure 3.4(b), it is better to assign some of the CPs and their CAPs to R_4 simultaneously at the first decomposition stage, like a and g in Figure 3.4(d), if needed. And the second decomposition stage will deal with the patterns in R_4 by stitch insertion or e-beam shot.

For two patterns i and j, if i is a CAP of j or j is a CAP of i, then we call the edge between i and j as a conflict adjacent edge and denote it by cae_{ij}. For example, every edge in Figure 3.4(f) is a conflict adjacent edge. Let E_{ca} be the set of conflict adjacent edges. We introduce a graph $G_{R_4}(R_4, E_{R_4})$, which is induced by R_4 from graph $G(V, E_{ca})$, where $E_{R_4} \subset E_{ca}$. Since if a pattern $i \in R_4$ is assigned to e-beam, then some other patterns in R_4 connected to i by cae_{ij} may be assigned to TPL masks using stitch insertion. Hence we hope to assign the conflict patterns connected by conflict adjacent edges to R_4 at the first decomposition stage, if needed.

Following this motivation, we propose the extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) problem. A dominating set S_D of a graph $G_{R_4}(R_4, E_{R_4})$ is a subset of R_4 such that every vertex not in S_D is adjacent to at least one member of S_D. For the MDSR_4MA problem, every pattern in a layout would be assigned to one of the sets C_1, C_2, C_3, S_D and $R_4 - S_D$, such that the conflict spacing rule is satisfied. The main objective is minimizing the total VSB number of patterns in S_D, and the secondary objective is minimizing the size of the set $R_4 - S_D$. The MDSR_4MA problem can be seen as a hybrid of the minimum weight dominating set problem and the 3-coloring problem, which can be formulated as

$$\min \quad \sum_{i \in S_D} w_i + \beta(|R_4| - |S_D|) \tag{3.3}$$

$$\text{s.t.} \quad \text{if } i,\ j \in C_k,\ \text{then } i \notin A(j),\ k = 1, 2, 3; \tag{3.3a}$$

$$i \in C_1 \cup C_2 \cup C_3 \cup R_4; \tag{3.3b}$$

$$S_D \text{ is a dominating set of } G_{R_4}(R_4, E_{R_4}). \tag{3.3c}$$

Obviously, any vertex in R_4 is either in S_D, or adjacent to at least a vertex in S_D. The objective of the problem is minimizing the sum of weights of vertices in the dominating set S_D and $\beta(|R_4| - |S_D|)$, where the weight w_i of vertex i is

the same as that in problem (3.1).

Consider the MDSR_4MA problem on the graph in Figure 3.4(g), where the coloring schemes of vertices in Figures 3.4(g) and 3.4(h) are two feasible solutions respectively. For Figure 3.4(g), $G_{R_4}(R_4, E_{R_4})$ is the graph with $R_4 = \{c,\ g\}$ and $E_{R_4} = \emptyset$. $S_D = \{c,\ g\}$ is a dominating set of G_{R_4}, and the objective value of the MDSR_4MA problem is $\sum_{i \in S_D} w_i + \beta(|R_4| - |S_D|) = w_c + w_g = 2$. For Figure 3.4(h), $G_{R_4}(R_4, E_{R_4})$ is the graph with $R_4 = \{a,\ g\}$ and $E_{R_4} = \{(a,\ g)\}$. $S_D = \{g\}$ is a dominating set of G_{R_4}, and the objective value the MDSR_4MA problem is $\sum_{i \in S_D} w_i + \beta(|R_4| - |S_D|) = w_g + \beta(|R_4| - |S_D|) = 1 + \beta$. In fact, the solution as Figure 3.4(h) is an optimal solution of the MDSR_4MA problem.

Problem (3.3) is not in numerical form. In order to solve the problem, we formulate it as a binary linear program (BLP):

$$\min \quad \sum_{i \in V} w_i z_i + \beta \sum_{i \in V} (y_i - z_i) \tag{3.4}$$

$$\text{s.t.} \quad x_{i2} - x_{i1} + x_{j2} - x_{j1} \leq 1, \qquad \forall e_{ij} \in E; \tag{3.4a}$$

$$x_{i1} - x_{i2} + x_{j1} - x_{j2} \leq 1, \qquad \forall e_{ij} \in E; \tag{3.4b}$$

$$x_{i1} + x_{i2} + x_{j1} + x_{j2} \leq 3, \qquad \forall e_{ij} \in E; \tag{3.4c}$$

$$1 - x_{i1} - x_{i2} \leq y_i, \qquad \forall i \in V; \tag{3.4d}$$

$$x_{i1} + x_{i2} - 2 \leq -2z_i, \qquad \forall i \in V; \tag{3.4e}$$

$$1 - x_{i1} - x_{i2} - x_{j1} - x_{j2} \leq \sum_{m \in A_{ca}(i) \cup \{i\}} z_m, \qquad \forall i \in V, j \in A_{ca}(i) \cup \{i\}; \tag{3.4f}$$

$$(x_{i1}, x_{i2}) \in \{0,1\}^2, y_i, z_i \in \{0,1\}, \qquad \forall i \in V. \tag{3.4g}$$

In the above formulation, (x_{i1}, x_{i2}) is used to denote a color as in problem (3.2). $A_{ca}(i)$ is the set of vertices connected to i by conflict adjacent edges. Constraints (3.4a)-(3.4c) is an equivalent formulation of Constraint (3.3a). That is, any two patterns within the same color class C_k are not conflicting, $k = 1, 2, 3$.

Constraint (3.4d) is used to force that, if $(x_{i1}, x_{i2}) = (0{,}0)$, i.e., $i \in R_4$, then $y_i = 1$; otherwise $y_i = 0$, since the objective is minimization and $w_i > 0$. Constraint (3.4e) is used to force that, only if pattern $i \in R_4$, then z_i may be equal to 1; otherwise, $z_i = 0$. Constraint (3.4f) is used to find a dominating set S_D of $G_{R_4}(R_4, E_{R_4})$. If $z_i = 1$, then $i \in S_D$. If all adjacent patterns of pattern $i \in R_4$ are not in R_4, i.e., all $j \in A_{ca}(i)$ are not in R_4, then $z_i = 1$. If there exists $j \in A_{ca}(i) \cap R_4$ such that $z_j = 1$, then $z_i = 0$ since the objective is minimization. If for all $j \in A_{ca}(i) \cap R_4$, $z_j = 0$, then $z_i = 1$, which means i is a vertex in the dominating set.

We use the cutting plane approach in the software package GUROBI [3] to solve Problems (3.2) and (3.4). Problems (3.2) and (3.4) are hard to solve in the large scale case, especially for problem (3.4), since it has more variables and constraints. However, the graph reduction techniques in Section 3.5 can cut down the size of the problem such that it is easy to solve using the cutting plane approach. Here, we have the following result for Problems (3.2) and (3.4).

Theorem 3.3.1. Suppose M is the number of native conflict structures NCS in a layout L. We have

i) for the HETLD problem, its VSB number is at least M;

ii) suppose x^{R*} is an optimal solution of problem (3.2) or (3.4), then it holds that $VSB(x^{R*}) \geq M$.

Proof. i) For the initial layout L, by Theorem 3.2.5, at least M patterns should be printed by e-beam, since there are M native conflict structures NCS in the layout L. Moreover, every one of these patterns should be printed by at least a VSB shot. Hence at least M VSB shots should be used to eliminate the conflicts. So the total VSB number for the HETLD problem is not less than M.

ii) For problem (3.2) or (3.4), all vertices of a K_4CS in E^R satisfy Constraints (3.2a)-(3.2c) or (3.4a)-(3.4c). Since an NCS is not 3-colorable, an optimal solution x^{R*} of problem (3.2) or (3.4) includes at least M components with

$x_i^{R*} \in R_4$, and $VSB_i \geq 1$ $(\forall i \in M)$ holds. Thus $VSB(x^{R*}) \geq M$. □

3.4 Legalization by Stitch Insertion, E-Beam Shot and Backtrack Coloring

In the second decomposition stage, a solution $x^R = (x_1^R, x_2^R, \cdots, x_n^R)$ of problem (3.2) and (3.4) may be infeasible for the HETLD problem. This is caused by that: First, there may exist some conflicts between patterns due to the edge deletion of the relaxed conflict graph $RG(V, E^R, W)$ (introduced in Subsection 3.5.1.2). Second, the patterns in R_4 are uncolored which should be assigned to TPL masks by stitch insertion or printed by e-beam shot.

Hence an infeasible solution x^R must be legalized to a feasible solution $x^H = (x_1^H, x_2^H, \cdots, x_n^H)$ of the HETLD problem. First, we deal with the conflicts between patterns by stitch insertion. If a conflict cannot be eliminated by stitch insertion, then for the two patterns causing the conflict edge, we assign one of them with less VSB number to e-beam shot. After all conflicts having been eliminated, we assign the patterns in R_4 to TPL masks by stitch insertion or assign to e-beam. In addition, in order to achieve a better solution, a backtrack coloring method is used, which is a local swap based method.

3.4.1 Conflict Elimination

In an optimal solution x^R of problem (3.2) or (3.4), most of the components satisfy Constraint (3.1a) or (3.3a), but there still exist some components which violate the constraint. Suppose that components p and q of x^R satisfy $x_p^R = x_q^R \in C_k$ $(k \in \{1, 2, 3\})$, and $(p, q) \in E$. That means (p, q) is a conflict edge and p and q are assigned to the same TPL color, and there is a conflict between p and q. We introduce stitch insertion or e-beam shot to patterns p or q to eliminate the conflict. There are three cases:

1. Stitches are inserted into one of the patterns p or q, say p. Pattern p is

split into several sub-patterns $p_1, p_2, \cdots, p_m$. These sub-patterns p_i, $i = 1, 2, \cdots, m$, can be divided into two classes according to the distance between p_i and q, $i = 1, 2, \cdots, m$: i) one class of patterns is that the distance is less than or equal to $\min_{cs}$, then the TPL colors of these sub-patterns should be different from q; ii) another one is that the distance is greater than $\min_{cs}$, then the TPL colors of these sub-patterns may be the same as that of pattern q.

2. Both patterns p and q are split into several sub-patterns using stitches. Too close sub-patterns (within $\min_{cs}$) should be assigned different TPL colors.

3. Stitch insertion cannot eliminate the conflict between the two patterns. That means one of p and q should be printed by e-beam. We choose the pattern with less VSB number for e-beam shot.

Note that, if one of patterns p and q, say p, is a CP, and p and three patterns from its CAP compose a K_4, then inserting stitches into p cannot eliminate conflicts. Motivated by this fact, for patterns p and q that satisfy $(p, q) \in E$ and p and q are assigned to the same TPL color in the infeasible solution, we propose a conflict elimination algorithm as Algorithm 3.1 to insert stitches to patterns p or q or assign e-beam to p or q.

In Algorithm 3.1, the function $genCSI()$ (line 2) is used to generate potential candidate stitch insertions for every pair of patterns p and q which satisfy that $(p, q) \in E$ and p, q are assigned to the same TPL color. In [59], the authors proposed an algorithm to generate all possible candidate stitch insertions CSI for a pattern i, we call it $genCSI(i)$. The algorithm first finds all conflict regions on pattern i, and then finds the horizontal or vertical line segments on pattern i, which are tangent to a conflict region. A line segment at pattern side is called a Checking Stitch Edge (CSE). The combination of one or more CSEs on pattern i is called a Candidate Stitch Insertion (CSI). A Candidate Stitch Insertion on a pattern will produce sub-patterns with the minimum number of

Algorithm 3.1 Relaxation solution legalization 1

Input: Relaxation solution x^R;
Output: Solution $x^{R'}$ without conflict;

```
1: for every (p, q) ∈ E and x_p^R = x_q^R ∈ C_k do
2:    CSIs = genCSI(p, q) for patterns p and q;
3:    NS_0 ← +∞;
4:    for every CSI of p and q do
5:       if conflict between p and q is eliminated and there is no new conflict
         generated by inserting CSI then
6:          if NS_CSI < NS_0 then
7:             NS_0 ← NS_CSI, and store current CSI;
8:          end if
9:       end if
10:   end for
11:   if NS_0 < +∞ then
12:      insert CSI with stitch number NS_CSI = NS_0 to patterns p or q, and
         the generated sub-patterns are assigned TPL colors C_k, k = 1, 2, 3;
13:   else
14:      if VSB_p < VSB_q then
15:         pattern p is assigned to e-beam;
16:      else
17:         q is assigned to e-beam;
18:      end if
19:   end if
20: end for
```

conflicts.

In this chapter, we use the corresponding algorithm in [59] as the function $genCSI()$ to generate all candidate stitch insertions of patterns p and q, i.e., $CSIs = genCSI(p, q)$ (line 2). A candidate stitch insertion CSI is a stitch insertion plan which may include some stitches on patterns p and q. NS_{CSI} (line 6) is the stitch number of a certain CSI. NS_0 is an intermediate variable. Of course, stitches generated by $genCSI()$ should satisfy the stitch location condition: a candidate stitch insertion is not near the periphery or the corner of the pattern, and should be in the overlap margin of the pattern which is greater than $\min_{om}$.

In Algorithm 3.1, first we check stitch insertions for a pair of patterns p and q (lines 2-10). If there are some insertion plans which can totally eliminate the conflict between p and q, then we choose the insertion plan with the minimum stitch number to eliminate the conflict (lines 11-12); otherwise, we consider e-beam shot to print one of p and q which has less VSB number (lines 13-19), and let another one printed by TPL.

3.4.2 Assignment of Patterns in R_4

After removing all conflicts of the solution x^R, in order to obtain a feasible solution x^H of the HETLD problem, we assign the patterns in R_4 to TPL colors or e-beam shot. The patterns in R_4 are dealt with one by one. We check whether every pattern $i \in R_4$ can be divided into several sub-patterns by stitch insertion such that these sub-patterns can be assigned to TPL masks without generating conflicts. If not, i is assigned to e-beam. The details are as Algorithm 3.2 shows, which is similar to Algorithm 3.1, where we only consider a pattern i instead of a pair of patterns p and q. The explanations are the same as those of Algorithm 3.1, and are skipped here. It must be remarked that, Algorithm 3.2 shows the assignment of patterns in R_4 for the ESTMA problem (3.2). For the MDSR_4MA problem (3.4), the statement "every pattern $i \in R_4$" (line 1) in Algorithm 3.2 would be replaced by "every pattern $i \in R_4 - S_D$, then every pattern $i \in S_D$".

3.4.3 Backtrack Coloring

In the above legalization algorithms, the quality of an obtained solution depends on the legalization order. Moreover, due to stitch insertion, some removed patterns at the graph reduction stage introduced in the next section may have to be assigned to e-beam. However, the VSB number caused from the above two cases could be reduced by a backtrack coloring method, which tries to further reduce the VSB number of the solution x^H obtained from Sections 3.4.1 and 3.4.2. The method is detailed as Algorithm 3.3.

Algorithm 3.2 Relaxation solution legalization 2

Input: Legalized solution $x^{R'}$ from Algorithm 3.1;
Output: Feasible solution x^H;

```
1: for every pattern i ∈ R_4 do
2:    CSIs = genCSI(i) for pattern i;
3:    NS_0 ← +∞;
4:    for every CSI of i do
5:       if there is no conflict generated from i by inserting CSI then
6:          if NS_CSI < NS_0 then
7:             NS_0 ← NS_CSI, and store current CSI;
8:          end if
9:       end if
10:   end for
11:   if NS_0 < +∞ then
12:      insert CSI with stitch number NS_CSI = NS_0 to pattern i, and the
         generated sub-patterns are assigned to TPL colors C_k, k = 1, 2, 3;
13:   else
14:      pattern i is assigned to e-beam;
15:   end if
16: end for
```

Algorithm 3.3 Backtrack coloring

Input: Solution x^H obtained from Algorithm 3.2;
Output: Another feasible solution $x^{H'}$;

```
1: for every pattern i assigned to EB in x^H do
2:    calculate totalVSB(C_k, i) of all patterns in C_k connected to i, k = 1, 2, 3;
3:    k_0 ← argmin_{k=1,2,3} {totalVSB(C_k, i)};
4:    if totalVSB(C_{k_0}, i) < VSB_i then
5:       C_{k_0} = C_{k_0} ∪ {i}, EB = EB − {i};
6:       for every j connected to i and j ∈ C_{k_0} do
7:          C_{k_0} = C_{k_0} − {j};
8:          Use lines 2-15 of Algorithm 3.2 for pattern j to perform stitch inser-
            tions or e-beam assignment;
9:       end for
10:   end if
11: end for
```

Algorithm 3.3 aims at finding a better solution with less total VSB number by searching the adjacent patterns of pattern $i \in EB$ of x^H. In this algorithm, $totalVSB(C_k, i)$ is the total VSB (rectangle) number of patterns in color class C_k connected to i, VSB_i is the VSB (rectangle) number of pattern i. Line 8

is used to perform stitch insertions or e-beam assignment for pattern $j \in C_{k_0}$ which is connected to i.

3.5 Graph Reduction and Decomposition Flow

In this section, firstly, we introduce some vertex removal techniques, and propose a new graph reduction technique which removes some edges. And then, we show our decomposition flow.

3.5.1 Graph Reduction

Generally, it is hard to solve directly the large scale HETLD problem of general layout, due to the complexity of the problem. For tackling the large scale problem, some techniques should be utilized first to preprocess the conflict graph for reducing the size of the problem.

3.5.1.1 Vertex Removal

We introduce some tricks to delete some easily colored vertices from the conflict graph, which have been popularly used to reduce the size of the TPL layout decomposition problem [36, 53, 59, 105]:

- Vertex with degree less than three removal [36, 53, 59, 105];
- Contained vertex removal [59];
- Connected component calculation [36, 53, 59, 105].

These techniques are highly effective for reducing the problem size of the HETLD problem. Since vertices with degree less three are easily colored for the 3-coloring problem, the operation *Vertex with degree less than three removal* will not lose the solution quality of the HETLD problem, and is used repeatedly in our decomposition flow. The operation *Contained vertex removal* was introduced in [59], which aims at deleting contained vertices. The definition of *Contained vertex* is as follows.

Definition 3.5.1 (contained vertex)**.** Given a graph $G(V, E)$, for a pair of vertices $i, j \in V$, suppose that $(i, j) \notin E$ and $A(i) \subseteq A(j)$, where $A(i)$ and $A(j)$ are the set of adjacent vertices of vertices i and j, respectively. Then we call that vertex i is contained in vertex j, vertex i is called a contained vertex, and j is called a containing vertex.

A contained vertex can be prior assigned the color of its containing vertex. The vertices with degree less than three and the contained vertices are deleted from the conflict graph before the TPL mask assignment stage. And they are colored as soon as the TPL mask assignment stage finishes. The order of coloring these vertices is in the reverse order of deleting them.

Another graph reduction technique is *connected component calculation*. S-ince the HETLD problem in different connected components is independent, we need to calculate the connected components for solving the problem easier. Actually, the conflict graph can be divided into a number of connected components after the operations *Vertex with degree less than three removal* and *Contained vertex removal*. Furthermore, our algorithm deals with the HETLD problem on the connected components one by one.

3.5.1.2 Edge Deletion

For the HETLD problem, e-beam and stitch are used to eliminate conflicts. To minimize VSB and stitch numbers, it is necessary to figure out the relationship among the conflict, stitch and VSB. That is, we must decide which conflict edge can be eliminated by inserting stitches in a pattern, and which conflict edge must be eliminated by VSB. According to our analysis in Section 3.2.2 and our empirical experiments, we have three meaningful observations:

1. Conflicts are mainly generated due to K_4 structure, which is the smallest 3-uncolorable graph;
2. For the native conflict structure NCS (including K_4CS), the conflicts in it

cannot be totally eliminated by stitch insertions;

3. Suppose that there are conflicts at pattern i. If i is a conflict pattern (CP), then the conflicts at pattern i cannot be totally eliminated by inserting stitches into this pattern; if i is a non-conflict pattern (NCP), then the conflicts at pattern i might be eliminated by inserting stitches into this pattern.

Inspired by the above observations, we construct a relaxed conflict graph $RG(V, E^R, W)$ by deleting some minor conflict edges of the weighted conflict graph $CG(V, E, W)$. In the weighted conflict graph, if a K_4 structure is not a K_4CS, and a conflict edge (i, j) in the K_4 structure satisfies one of the following conditions, then (i, j) is considered minor:

i) at least one of i and j is not CP;

ii) both i and j are CP, but at least one of them has that its CAPs are not all in the K_4 structure.

For a K_4 structure, there may be more than one conflict edges satisfying the above conditions, but we only delete the minor conflict edge (i, j) with the sum of weights of vertices $w_i + w_j$ less than those of the other minor conflict edges.

According to our statistics, many conflict edges could be deleted at this step. The relaxed conflict graph is constructed at the conflict pattern CP and K_4 conflict structure identifying step (Section 3.2.2). After that, a relaxed conflict graph $RG(V, E^R, W)$ is generated, where $E^R \subseteq E$. Then our ESTMA and MDSR_4MA problems are solved on $RG(V, E^R, W)$, respectively. The relaxed conflict graph is sparser than the original conflict graph by deleting some conflict edges. Thus, solving the ESTMA and MDSR_4MA problems on $RG(V, E^R, W)$ are easier than on $CG(V, E, W)$.

3.5.2 Flow for Hybrid EBL and TPL Layout Decomposition

For the HETLD problem of general layout, our decomposition flow is summed up as Figure 3.5. Given a layout L, according to the minimum coloring spacing $\min_{cs}$ rule, we transform the geometric layout structure to a conflict graph $CG(V, E)$, where V is the set of patterns, and E is the set of conflict edges between any two patterns. Some graph reduction techniques are used to reduce the size of the conflict graph. After that, the surface projection based method [59] is introduced to calculate conflict patterns and detect native conflict structures. Then we introduce an e-beam and stitch aware TPL mask assignment problem for the reduced graph.

Based on the native conflict structure, a relaxed conflict graph is constructed by removing some conflict edges. On the relaxed conflict graph, the e-beam and stitch aware TPL mask assignment (ESTMA) problem and the extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) problem are formulated. Furthermore, the 0-1 linear program of one of the above two problems are formulated for obtaining a solution. At last, stitch insertion and e-beam assignment are introduced to eliminate all conflicts, and obtain a higher resolution for the layout decomposition problem. The details of the decomposition flow are illustrated in the above sections.

3.5.3 An Example of The Two Stage Decomposition Algorithm

In this section, for the hybrid e-beam and TPL layout decomposition (HETLD) problem, we give an example to illustrate the ESTMA / MDSR_4MA based decomposition methods described in Sections 3.2, 3.3 and 3.4.

Figure 3.6(a) is a layout with patterns $\{a,\ b,\ c,\ d,\ e,\ f,\ g\}$. According to the minimum coloring spacing rule, a conflict graph $CG(V, E)$ is constructed as in Figure 3.6(b), where

$V = \{a,\ b,\ c,\ d,\ e,\ f,\ g\}$;

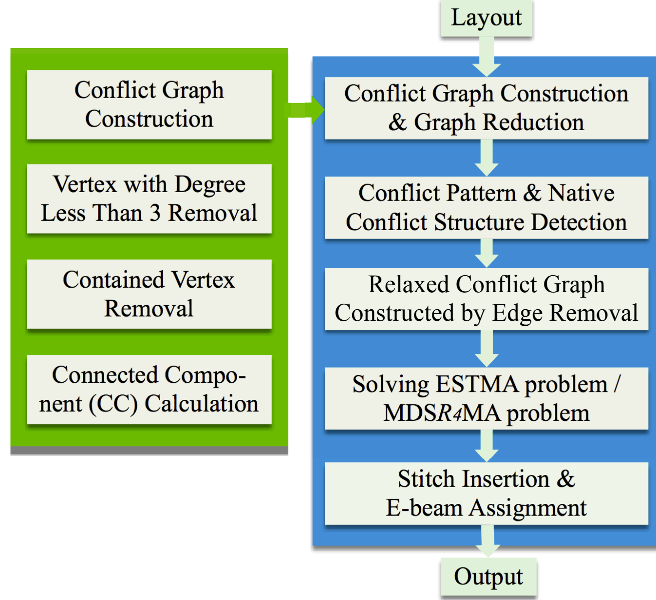


Figure 3.5: Our HETLD decomposition flow.

$$E = \{(a,b),\ (a,c),\ (a,d),\ (b,c),\ (b,d),\ (b,e),\ (c,d),$$
$$(c,e),\ (c,f),\ (c,g),\ (d,e),\ (d,f),\ (e,f),\ (f,g)\}.$$

Here, we skip the graph reduction techniques, focusing on the two decomposition stages.

3.5.3.1 Decomposition Stage

Before coloring, using the conflict identification method, we can find all K_4 subgraphs: $\{a, b, c, d\}$, $\{b, c, d, e\}$, $\{c, d, e, f\}$; all conflict patterns CP: b, c, d, e, f; and all K_4CSs: $\{b, c, d, e\}$, $\{c, d, e, f\}$, which are shown in Figure 3.6(c). Then all patterns are weighted by our weighting rule: $w(a) = 0.01$, $w(b) = 1$, $w(c) = 2$, $w(d) = 3$, $w(e) = 1$, $w(f) = 1$, $w(g) = 0.01$.

For the K_4 subgraph $\{a, b, c, d\}$, according to the relaxed conflict graph construction method, since pattern a is not a CP, edge (a, b) satisfies the conflict edge deletion condition, and the weights of a and b are $w(a) = 0.01$ and $w(b) = 1$, respectively, (a, b) will be deleted. Then we obtain the relaxed conflict graph $RG(V, E^R, W)$ as Figure 3.6(d), where $E^R = E - \{(a, b)\}$.

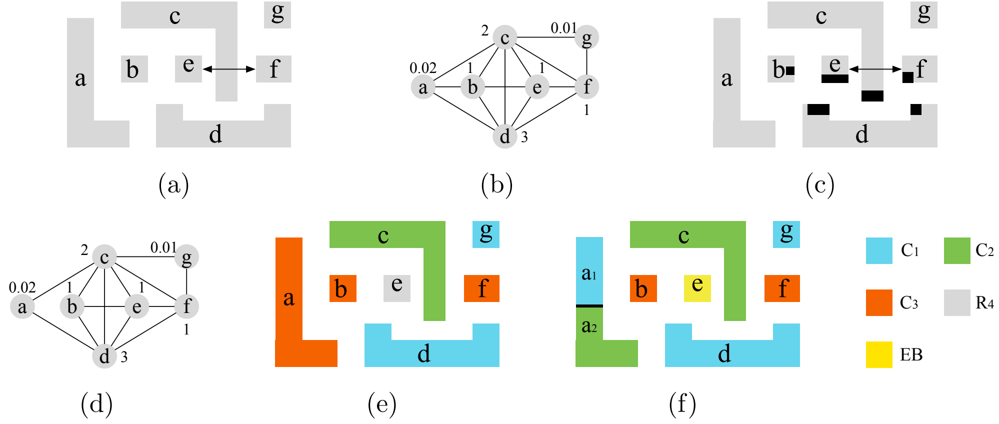


Figure 3.6: A sample of HETLD flow. (a) Initial layout. (b) Conflict graph. (c) Conflict pattern identification. (d) Relaxed conflict graph. (e) Solution of problem (3.2) or (3.4). (f) Feasible solution of HETLD.

By solving problem (3.2) or (3.4) (the two problems have the same solution in this case), we obtain a solution x^R with the minimum sum of weights: $x^R = (c_3,\ c_3,\ c_2,\ c_1,\ r_4,\ c_3,\ c_1)$, where c_k means the corresponding vertex is in the color class C_k, $k = 1, 2, 3$, and r_4 means the corresponding vertex is not colored. Hence $C_1 = \{d,\ g\},\ C_2 = \{c\},\ C_3 = \{a,\ b,\ f\},\ R_4 = \{e\}$.

As Figure 3.6(e) shows, color C_1 is blue, C_2 is green, C_3 is orange, and R_4 is uncolored.

3.5.3.2 Legalization stage

However, patterns a and b in C_3 are infeasible for the HETLD problem. Hence we first consider inserting stitches to patterns a and b. By Algorithm 3.1, we find a stitch insertion plan as Figure 3.6(f), where a stitch is inserted into pattern a, and a is split into two sub-patterns a_1 and a_2. Then a_1 is assigned to color class C_1, and a_2 is assigned to color class C_2. Furthermore, stitch insertion is invalid for pattern e by calling Algorithm 3.2. Thus e is assigned to e-beam. Finally, we obtain a feasible solution x^H of the HETLD problem:

$x^H = (\{c_1,\ c_2\},\ c_3,\ c_2,\ c_1,\ eb,\ c_3,\ c_1)$;

$C_1 = \{a_1,\ d,\ g\},\ C_2 = \{a_2,\ c\},\ C_3 = \{b,\ f\},\ EB = \{e\}$.

For the solution x^H, the VSB number is $\sum_{i \in EB} VSB_i = VSB_e = 1$, and the stitch number is 1. Actually, it can be seen that the solution is an optimal solution of the HETLD problem for this layout.

3.6 Experimental Results

Our decomposition methods for the hybrid e-beam and triple patterning lithography of general layout is programmed in C++ and run on a personal computer with 2.7GHz CPU, 8GB memory and the Unix operating system. We test our method on the ISCAS-85 & 89 benchmarks provided by Yu *et al.* [105]. In this chapter, the minimum coloring spacing is set as $160nm$, the minimum pattern size $\min_{ps}$ and the overlap margin $\min_{om}$ are set as $10nm$.

Since this chapter aims at hybrid decomposition for general layout, we ignore the row structures of the test benchmarks. Note that EBL is low throughput, and the general e-beam is a variable-shaped beam (VSB), which means that if a pattern is printed by e-beam shot, then it would be printed by several VSBs. Therefore, for the purpose of throughput, we use the number of VSBs mainly to evaluate the performance of the compared methods. Moreover, stitch may lead to potential functional errors of a chip during manufacture, hence the number of stitches is another comparison criterion.

3.6.1 Statistics and Analysis

Since one concern of this chapter is reducing the size of the problem, we compare some statistics of the initial conflict graphs and the relaxed conflict graphs. The statistics on all benchmarks are listed in Table 3.1. The data in the columns "#P" and "#E" are the numbers of patterns and conflict edges in the conflict graphs, respectively. The data in the column "Ratio" of "initial conflict graph" and "after graph reduction" are the ratios between the numbers of edges and patterns. Every data in the column "#ANC" is the average number of patterns on the number of connected components. Every data in the column

Table 3.1: Statistics of hybrid e-beam and TPL layout decomposition benchmarks with $\min_{cs} = 160nm$

Circuits	Initial conflict graph				After graph reduction				After Relaxing	
	#P	#E	Ratio	#ANC	#P	#E	Ratio	#ANC	$\#E^R$	Ratio^R
C432	1109	2160	1.95	65	478	966	2.02	19	890	1.86
C499	2126	4590	2.16	41	1342	2827	2.11	21	2549	1.9
C880	2411	4434	1.84	35	1187	2282	1.92	17	2162	1.82
C1355	3262	5906	1.81	36	1323	2583	1.95	17	2453	1.85
C1908	5125	8846	1.73	36	1580	3049	1.93	14	2914	1.84
C2670	7933	14480	1.83	36	3896	7687	1.97	18	7302	1.87
C3540	10189	17798	1.75	38	4389	8171	1.86	13	7758	1.77
C5315	14603	26467	1.81	40	6686	12768	1.91	17	11940	1.79
C6288	14575	26038	1.79	38	5251	10033	1.91	16	9476	1.8
C7552	21253	37930	1.78	43	9033	17336	1.92	17	16365	1.81
S1488	4611	8769	1.9	35	3020	5877	1.95	26	5461	1.81
S38417	67696	126215	1.86	105	28978	61522	2.12	17	57507	1.98
S35932	157455	317832	2.02	133	88188	192619	2.18	27	179621	2.04
S38584	168319	314785	1.87	83	70121	151141	2.16	16	141199	2.01
S15850	159952	309753	1.94	77	87216	184948	2.12	22	172449	1.98
Avg.	42708	81733	1.87	56	20846	44254	2.00	18	41336	1.88
Ratio	1.00	1.00	1.00	1.00	0.49	0.54	1.07	0.33	0.50	1.01

"#E^R" is the number of conflict edges in the relaxed conflict graph, and every data in the column "Ratio^R" is the ratio between the number of edges and patterns in the relaxed conflict graph.

From Table 3.1, comparing with the initial conflict graph, the number of patterns and the number of conflict edges are only half left after graph reduction, which shows that the graph reduction techniques used are effective. Note that, conflict edges are removed at two stages, one is at the graph reduction stage and another one is at the relaxed conflict graph (RG) construction stage. From the column "Ratio" of "initial conflict graph", the average value is 1.87, which means the number of total conflict edges is nearly 1.87 times as many as the number of total patterns for every benchmark. From the column "Ratio" of "after graph reduction", the average value is 2.00, which means that after graph reduction the conflict graph is denser than the initial conflict graph.

The graph reduction stage mainly removes vertices with degree less than three and contained vertices and remove the incident edges, while the RG con-

struction stage aims at removing conflict edges from dense structures. Comparing the column #E^R with the column #E in "after graph reduction", it can be found that the number of edges of #E is reduced to #E^R by 0.54/0.50=8%. This implies that removing edges at the RG construction stage is effective for dense graph structures. Furthermore, comparing the data in the two columns "#ANC" indicates that, the average number of patterns in every connected component in the relaxed conflict graph is only one-third of the initial graph. Thus, it is less enough for solving 0-1 linear programs (3.2) and (3.4) on every connected component.

The most time consuming computation in our two stage decomposition method is solving the binary linear program (3.2) or (3.4) by the cutting-plane approach in the software package GUROBI. In order to speed up the computation, we set the parameter *gap* in GUROBI [3] as a larger value for larger connected components. The parameter *gap* is used to control the termination criterion in the cutting plane approach, which is

$$\frac{|f(x^c) - LB|}{|f(x^c)|} \leq gap,$$

where $f(x^c)$ is the currently minimal value, and LB is the lower bound obtained by linear program relaxation of the binary linear program (3.2) or (3.4).

For every relaxed conflict graph, we count the number of connected components with vertex number between 60 and 100, and the number of connected components with vertex number not less than 100, respectively, and put them in columns "#$\geq$ 60" and "#$\geq$ 100" in Table 3.2, respectively. We test our hybrid decomposition method with different *gap*s in the cutting plane approach for the ILP of the ESTMA problem: i) less gap, $gap = 10^{-4}$ for all connected components; ii) larger gap, $gap = 0.4$ for connected components with vertex number less than 60, $gap = 0.5$ for connected components with vertex number between 60 and 100, and $gap = 0.6$ for connected components with vertex number not

Table 3.2: Comparison of decomposition results with less gap and larger gap for the ILP of the ESTMA based decomposition, $\min_{cs} = 160nm$

Circuits	CC number and size			#CP	Less gap			Larger gap		
	#CC	#≥60	#≥100		#VSB	#S	CPU(s)	#VSB	#S	CPU(s)
C432	25	1	0	275	85	7	9.77	86	6	2.24
C499	63	4	1	1042	279	33	109.06	281	31	7.89
C880	68	4	0	525	110	89	26.08	110	89	7.44
C1355	76	3	0	549	146	62	21.30	146	62	7.98
C1908	116	3	0	561	152	86	25.42	152	86	11.28
C2670	214	2	2	1549	423	271	200.10	430	264	21.09
C3540	347	5	0	2080	431	362	55.51	431	362	30.27
C5315	399	8	1	3363	858	344	176.45	862	341	37.32
C6288	338	5	2	2673	738	236	273.80	740	234	32.98
C7552	534	8	3	3840	1007	524	365.33	1019	515	51.85
S1488	116	6	2	1488	416	182	256.75	419	180	24.60
S38417	1723	26	5	17218	4048	1331	966.22	4069	1314	177.65
S35932	3243	297	74	52982	N/A	N/A	>3600	13258	2688	718.53
S38584	4463	54	3	46445	10021	3270	1321.14	10063	3233	269.90
S15850	4012	164	42	50086	N/A	N/A	>3600	13011	3689	487.90
Avg.	1049	39	9	12312	1439	523	292.84	1447	517	52.50
Ratio					0.99	1.01	5.58	1.00	1.00	1.00

less than 100. The test results of the ESTMA based decomposition method are listed in Table 3.2.

In Table 3.2, the data in the column "#CP" are the numbers of conflict patterns in the conflict graphs. These patterns should be prior assigned to T-PL masks, since conflicts at these patterns can hardly be eliminated by stitch insertions. In columns "#VSB", "#S" and "CPU(s)", we list the total VSB numbers, the total stitch numbers and the runtimes by our ESTMA based decomposition method, respectively. On the one hand, the average runtime of the ESTMA based method with less gap is 5.61× more than with that with lager gap. Moreover, the ESTMA based method with less gap is very slow for the connected components with many vertices, especially for benchmarks S35932 and S15850, which cannot be solved in one hour.

On the other hand, it can be seen that our ESTMA based decomposition method with larger gap produces slightly more VSB numbers than that with less gap for almost all benchmarks. More precisely, the ESTMA based method

Figure 3.7: Decomposed layout for benchmark C880 with $\min_{cs}$=160nm.

with larger gap achieves nearly the same good solutions as that with less gap for the benchmarks. Actually, this is due to the cutting plane method and our backtrack coloring algorithm. First, although the gap is not small enough, the cutting plane method may still obtain an sub-optimal solution of the binary programming problem. Second, the backtrack coloring algorithm can reduce the number of VSBs by local swapping.

Figure 3.7 presents our decomposition result for benchmark C880 with $\min_{cs} = 160nm$, which is obtained by the ESTMA based method with larger gap. In the figure, green, red and blue colors denote Mask 1, Mask 2 and

Mask 3, respectively, and yellow denotes E-beam.

3.6.2 Comparisons

In this subsection, we compare four hybrid e-beam and TPL decomposers: "TCAD'16" is the decomposer of [96], "Extended TOC'17" is the decomposer extended from [59], "ESTMA" and "MDSR_4MA" are the two decomposition methods proposed in this chapter based on problems ESTMA and MDSR_4MA respectively.

Yang *et al.* first focused on hybrid e-beam and multiple patterning lithography for general layout decomposition, and considered two different objectives to evaluate the throughput of EBL: minimum total VSB number and minimum total area for e-beam shot. Since for the e-beam direct writing strategy, total area of e-beam shot may not be an essential factor [66,104], and the writing time is the main factor for evaluating the throughput of EBL [31,35,42,82]. Furthermore, for VSB e-beam shot, the writing time mainly depends on the number of VSBs (rectangles). Hence we mainly compare our method with [96] on this aspect. Ref. [96] only lists results of ten benchmarks. We cite them directly for comparisons, since the code of their method is not available to us.

In Table 3.3, the results in the column "Extended TOC'17" are adapted from [59]. Since the objectives in [59] are conflict number and stitch number, we directly use e-beam shot to eliminate conflicts in the decomposition results by [59]. For fair comparison, we use VSB as few as possible to eliminate conflicts. The data in columns "ESTMA" and "MDSR_4MA" are the results of the two methods with larger gap.

The results of the four decomposers are reported in Table 3.3. Data in the columns "#VSB" and "#S" are the total VSB numbers and the total stitch numbers by the decomposers on the tested benchmarks, respectively. Data in the columns "CPU(s)" are the runtimes by the respective decomposers. In row "Avg.1", we list the average results on all the benchmarks. Since Ref. [96] did

Table 3.3: Comparison results of the four HETLD decomposers, $\min_{cs} = 160nm$

Circuits	TCAD'16 [96]			Extended TOC'17 [59]			ESTMA			MDSR_4MA		
	#VSB	#S	CPU(s)	#VSB	#S	CPU(s)	#VSB	#S	CPU(s)	#VSB	#S	CPU(s)
C432	108	17	5	162	9	0.89	86	6	2.24	81	11	3.01
C499	333	46	17.19	634	25	4.99	281	31	7.89	274	42	160.29
C880	N/A	N/A	N/A	297	73	3.11	110	89	7.44	118	94	13.48
C1355	207	99	10.74	315	86	3.56	146	62	7.98	145	78	20.14
C1908	N/A	N/A	N/A	383	56	2.42	152	86	11.28	156	98	172.47
C2670	N/A	N/A	N/A	876	172	12.14	430	264	21.09	411	294	90.96
C3540	574	380	45.21	768	368	4.89	431	362	30.27	429	449	166.32
C5315	1013	415	69.83	1589	262	11.68	921	341	37.32	909	386	176.45
C6288	N/A	N/A	N/A	1299	271	9.42	740	234	32.98	722	264	119.27
C7552	1321	640	94.06	2439	420	17.52	1119	515	51.85	1110	604	266.03
S1488	N/A	N/A	N/A	785	170	27.49	419	180	24.6	411	196	115.19
S38417	4634	2403	315.44	8395	1386	62.84	4269	1314	177.65	4235	1457	298.39
S35932	13963	7184	1117.9	26998	3516	1019.8	13258	2688	718.53	12235	3258	2768.38
S38584	10957	6132	773.7	20459	3779	90.12	10063	3233	269.9	9936	3709	684.51
S15850	14292	6983	800	25841	3897	547.17	13011	3689	487.9	12322	4370	1140.39
Avg.1				6083	966	121.2	3029	873	125.93	2900	1021	413.02
Avg.2	4740	2430	324.91				4359	1224	179.15			
Ratio	1.09	1.99	1.81	2.01	1.11	0.96	1.00	1.00	1.00	0.96	1.17	3.28

not report the test results on the benchmarks C880, C1908, C2670, C6288 and S1488, we do not list the average results of "TCAD'16" on all the benchmarks. However, we list in row "Avg.2" the average results of "TCAD'16" on all the benchmarks except the five benchmarks. For fair comparison, we also list in row "Avg.2" the average test results of "ESTMA" on all the benchmarks except the five benchmarks. In the last row "Ratio" of Table 3.3, we list the ratios of the average results of "TCAD'16", "Extended TOC'17" and "MDSR_4MA" based on the results of "ESTMA". It must be remarked that, the data in the last row "Ratio" of "TCAD'16" are calculated based on the data in row "Avg.2".

From Table 3.3, it can be seen that the average VSB number by [96] is 9% more than that by our ESTMA based method, and the average stitch number is twice more than that by our ESTMA based method. Moreover, it can be seen that the runtime of the method in [96] is 1.81 times more than that of our ESTMA based method. Furthermore, it must be noted that their decomposer was run on a workstation with 3GHz CPU and 4GB memory, which is better than ours. This together with the comparison results demonstrates that our ESTMA based method works better than the decomposer in [96] on the test benchmarks.

Comparing the data of columns "Extended TOC'17" and "ESTMA" in the row "Ratio", it can be seen that the average VSB number of "Extended TOC'17" is twice more than that of the ESTMA based method. This is due to that, the method in TOC'17 focuses on the minimum conflict number instead of the VSB number, while a pattern may be printed by more than one VSB shot. Furthermore, the average stitch number is 11% more than that by the ESTMA based method. Hence, extending the TPL layout decomposition method in [59] directly to solve the HETLD problem is not a good choice.

At last, we compare the data of columns "ESTMA" and "MDSR_4MA" in the row "Ratio". As expected by theory, the average number of VSB in "MDSR_4MA" is 4% less than that in "ESTMA", and the average number of

stitch in "MDSR_4MA" is 17% more than that in "ESTMA". This demonstrates that the MDSR_4MA based method assigns more pairs of a CP and its CAPs to R_4 simultaneously at the first decomposition stage, and then there are more patterns in R_4 are assigned to TPL masks by inserting stitches. Finally, the average runtime of MDSR_4MA is 3.28× of ESTMA. This is due to that the ILP formulation in the MDSR_4MA based method has more variables and constraints than the ILP formulation in the ESTMA based method.

3.7 Summary

Hybrid e-beam and triple patterning lithography is a new technology for manufacture of VLSI circuit, which combines the advantages of e-beam and T-PL. Layout decomposition is a core problem in the hybrid lithography, which is NP-hard on the general layout. In this chapter, we propose a two stage layout decomposition flow for the HETLD problem, which achieves decomposition by two steps. First, we consider the e-beam and stitch aware TPL mask assignment (ESTMA) problem, and then the problem is relaxed by deleting some conflict edges, which is used for fast obtaining a solution with some conflicts. Second, the infeasible solution with conflicts is legalized to a feasible one of the HET-LD problem by stitch insertion and e-beam shot. To speed up decomposition, we reduce the problem size by removing some vertices and some edges before decomposition.

Furthermore, in order to obtain a better solution with less VSB number, we propose the extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) problem. By solving the MDSR_4MA problem in the first decomposition stage, we can obtain a solution with the patterns in R_4 more likely being assigned to TPL masks by stitch insertion. However, the ILP formulation of the MDSR_4MA problem has many more variables and constraints than the ILP formulation of the ESTMA problem.

In the decomposition process, our objective is maximizing e-beam throughput (minimizing VSB number) and minimizing stitch number. Experimental results show the effectiveness of the ESTMA and the MDSR_4MA based decomposition methods, comparing with the state-of-the-art decomposer.

Chapter 4 Discrete Relaxation Method for Contact Layer Decomposition of DSA with Triple Patterning

4.1 Introduction

As the pitch size between features shrinking and the number of nodes increasing, manufacture of integrated circuit (IC) layout is more and more difficult. This urges on series of manufacture technologies, such as 193nm ArF immersion optical lithography and the related multiple patterning lithography, electron beam lithography, block copolymer directed self-assembly, and extreme ultra violet lithography [11, 18, 73]. IC layouts consist of patterned lines and holes. The lines define the active device regions, gate electrodes, and the wirings between the devices. The holes define the electrical contacts between the wires and the transistors [10, 99]. Some of the above manufacture technologies are popularly used to pattern line features in a layout [105], but the DSA technology is fit for patterning the dense hole features [75]. Especially, in 7nm nodes distribution of the features on contact/via layer is dense and aligned [91], hence the DSA technique is necessary.

To pattern contact holes by DSA, guiding templates are usually used to form contacts [44,77]. For sparse structure, a number of single-hole templates are used to form contacts. For dense structure, too close templates would generate conflicts [7, 52, 93]. To reduce the conflicts, some of the contacts within a short distance would be grouped together in a multi-hole template [7,52,93]. As shown in Figure 4.1(a), the left contact is contained in a single-hole template, and the right two close contacts are grouped in a two-hole template.

However, grouping more than one contacts in a multi-hole template may introduce overlays. For different guiding templates with different shapes or

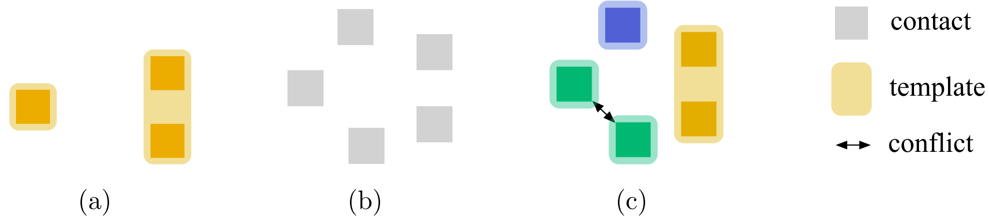


Figure 4.1: An example of contact layer decomposition for DSA with TPL. (a) A template assignment for the sparse layout. (b) A dense layout. (c) A mask and template assignment for the dense layout.

sizes, the overlays are different. Specifically, complex (irregular shape) guiding templates may introduce large overlays and the contained contacts may not be patterned correctly [93]. Hence, during template assignment, the cost of a guiding template should be considered.

Furthermore, for a very dense contact layer layout, the contact layer fabricated by single patterning is unqualified due to a number of conflict errors. Hence the DSA with multiple patterning (DSA-MP) technology is a solid choice, and a crucial problem in DSA-MP is the mask and template assignment. An example of mask and template assignment for DSA with triple patterning (DSA-TP) is shown in Figures 4.1(b) and 4.1(c). Figure 4.1(c) is a template assignment of the layout in Figure 4.1(b), where the three colors represent three masks, and the right two contacts are contained in a vertical two-hole template, and a conflict is generated between the two green one-hole templates due to the small pitch between them.

Recently, some works concerned the mask and template assignment problem of DSA-MP [7, 52], including the mask and template assignment problem of DSA with double patterning (MTADD) and with triple patterning (MTADT). For the MTADD problem, Ref. [52] has obtained good enough solutions for the tested benchmarks comparing with the solutions of the exact integer linear program formulation. However, the solutions still have many unresolved conflicts, although under their conflict spacing setting, the distributions of contacts in the tested layouts are sparse. Therefore, it is necessary to use triple masks for

the contact layer with $7nm$ nodes. In this chapter, we consider the mask and template assignment problem of DSA-TP (MTADT).

For the row structure layout, Xiao et al. [93] proposed three methods and compared their effectiveness. These methods are: 1) color first iterative; 2) group first iterative; 3) shortest path based optimal decomposition. By comparisons, the shortest path based method achieved the best decomposition. For the general layout, Badr et al. [7] first considered the MTADM problem and formulated it as an integer linear programming problem, and proposed a maximum cardinality matching (MCM) based method to quickly obtain a result. However, the method in [7] has some issues. In the aspect of problem formulation, the method in [7] does not consider the template cost, which is different for different types of templates. In the aspect of solution method, Ref. [7] proposed two methods for the MTADT problem. However, since many variables and constraints of template grouping are introduced, the ILP formulation is too complex to fast solve. Moreover, the MCM based method is a grouping first method, and the solution quality is unknown.

In order to improve the quality of decomposition results of the MTADT problem, Kuang et al. [52] considered the simultaneous template optimization and mask assignment problem of DSA with triple patterning. They proposed a look-up table (LUT) based assignment method, which finds all the possible 3-colorable sub-graphs by removing some edges. The method is fast and effective for sparse and small graph. However, since the number of 3-colorable sub-graphs of a graph is exponential, the storage size of LUT would not be scalable for very dense or large graph, and it is time consuming to check the LUT. In order to reduce runtime, the method in [52] does not store and check all 3-colorable sub-graphs. This will lose optimality of the results, and the gap between an obtained solution and the optimal solution is still unknown.

In this chapter, we propose a discrete relaxation based decomposition method to solve the MTADT problem of general layout. The discrete relaxation method

is a general scheme for dealing with hard discrete optimization problems, which relaxes a hard problem to an easier one, and then the relaxation solution is legalized to a solution of the initial problem [59]. An advantage of the method is that the solution quality can be evaluated in the experiment. The evaluation of a solution is significant for an NP-hard problem. If we know the gap between an obtained result and the optimal value of an instance, then we will know whether the solution is good or not. This scheme has been proposed and used to address the triple patterning layout decomposition problem [59]. However, the discrete relaxation method should be designed carefully according to the feature of an addressed problem.

For the MTADT problem of general layout, our main contributions are listed as follows.

- We sum up general rules for the costs of vertical or horizontal templates with different sizes, and construct a weighted conflict grouping graph.

- Basing on the weighted conflict grouping graph, we propose a novel integer linear program for the MTADT problem, which is not equivalent to the MTADT problem but provides a lower bound on the optimal value of the MTADT problem. Moreover, some valid inequalities are introduced for cutting some no good solutions, and obtaining a better lower bound.

- We propose a template assignment approach to transform a relaxation solution to a feasible solution of the MTADT problem, which provides an upper bound on the optimal value of the MTADT problem. According to the obtained lower bound and upper bound, we can evaluate the quality of our experimental results. Specially, if the upper bound is equal to the lower bound, then we obtain an optimal solution of the MTADT problem.

- Comparisons of experimental results show that our decomposition method is effective. More specifically, the gap between the obtained upper and

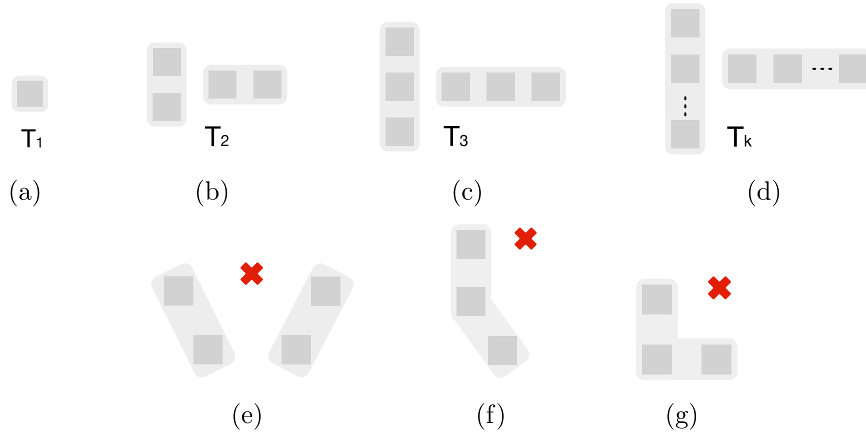


Figure 4.2: Template types. (a)-(d) Available vertical and horizontal templates. (e)-(g) Illegal templates.

lower bounds is 0.0% for most of the sparse benchmarks, which shows the optimality of the obtained results. And the average gap is 0.4% for the dense benchmarks, which shows the goodness of the obtained results for dense layouts.

4.2 Preliminaries

In this section, first we introduce the types of the DSA guiding templates, and then we describe the mask and template assignment problem of DSA-TP.

4.2.1 DSA Guiding Template

To print contact holes by DSA, guiding templates are needed, which are usually fabricated by conventional optical lithography technology [99]. Thus the resolution is limited by the pitch of guiding templates. For sparse structure, the contact pitch is big enough, hence the contacts can be contained in a series of single-hole templates. But for dense structure, the contact pitch is too small to satisfy the resolution for numerous single-hole templates, and multi-hole template would be used to guide a group of contacts for improving the resolution.

Theoretically, the type of multi-hole template could be of any shape [33].

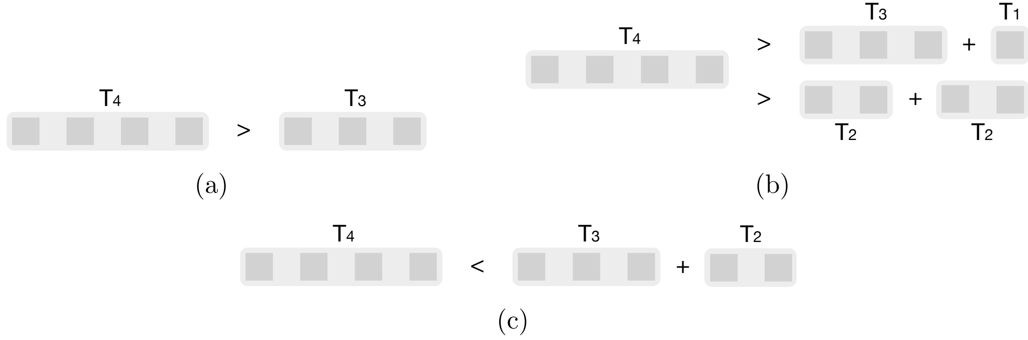


Figure 4.3: Comparison of the costs of different templates. (a) Rule 1). (b) Rule 2). (c) Rule 3).

However, complex guiding template may introduce large overlay and the intended contacts may not be patterned correctly [93]. Such as the diagonal templates (Figure 4.2(e)), the local diagonal templates (Figure 4.2(f)), or the "L" shape templates (Figure 4.2(g)), they cannot be printed reliably, hence the results after printed should be verified by the optical proximity correction process, which is of high cost [52, 92]. Furthermore, in order to consider the complex templates for the DSA technology, the grid model should be modeled for the contact layer layout [39, 70]. However, for some special contact layer layout, under the given conflict spacing and grouping spacing, some contacts do not align to the gird line. This may lead to that some templates cannot properly guide the matching contacts. In this chapter, we consider the MTADT problem without grid model. Hence under the gridless assumption and for the reason of avoiding high correction cost, we only consider the vertical and horizontal templates as in [52, 93]. That is, only a group of contacts in a vertical or horizontal line can be grouped into a template.

For vertical or horizontal templates, different sizes of templates have different costs [99]. The main factors deciding the cost of a template are the number of holes in the template and the size of the template. It must be remarked that, the hole pitches in a template may not be uniform since the distribution of contacts in a layout may not be regular. Thus, for two templates with the

same number of holes, their costs may be different. But for simplicity, we only consider in this chapter the number of holes as the evaluation of cost of a vertical or horizontal template as in [52].

We sum up three rules on the cost of a template, which are important but not unique:

1. A template with more holes will have higher cost.
2. A template will have higher cost than two or more templates for grouping the same number of holes.
3. Suppose that any two neighboring holes in a template is regarded as a pair. A template will have less cost than two or more templates if the latter templates have the same number of pairs as the former one.

Rule 1) is due to that, a template with more holes is more difficult to control the lithographic variations [33, 92]. Rule 2) is due to that, a template containing several holes is more difficult for lithographic variations than several other templates containing these holes. As for rule 3), the latter templates involve more contacts, which may generate more manufacture errors [92]. Figures 4.3(a)-4.3(c) show examples for rules 1-3, respectively.

Let T_k be a template with k holes. Figures 4.2(a)-4.2(d) show the vertical and horizontal templates, T_1, T_2, T_3, $\cdots$, and T_K, respectively, where K is the maximum number of holes in a template. Let $cost_{T_k}$ be the cost of template T_k, and we suppose that $cost_{T_k}$ is an integer in this chapter. According to the above three rules, the costs of templates are set as: i) $cost_{T_1} = 0$. This is because every contact can be guided by a single-hole template, while the use of multi-hole templates is for eliminating conflicts which needs extra cost; ii) $cost_{T_2} = 3$, as a baseline; iii) $2cost_{T_{k-1}} \geq cost_{T_k} > \frac{k}{k-1}cost_{T_{k-1}}$, $k = 3, 4, \cdots K$. This inequality indicates that the average cost of holes in T_k should be greater than that of holes in T_{k-1}.

The above setting rules for the cost of template is compatible with the settings in previous works [52,92,93]. In [92], Xiao et al. formulated an equation for calculating the cost of template: $c_i = \lambda \times p_i$, where c_i denotes the cost of the ith multiple template, and p_i is the number of templates pairs in the multiple template, i.e., $p_i = k - 1$ for template T_k. Suppose the cost of template T_2 is $cost_{T_2}$, then $cost_{T_3} = 2cost_{T_2}$, $cost_{T_4} = 3cost_{T_2} = \frac{3}{2}cost_{T_3}$, $\cdots$. It is easy to show that $2cost_{T_{k-1}} \geq cost_{T_k} > \frac{k}{k-1}cost_{T_{k-1}}$ includes the above equalities. Furthermore, as an example of setting in [93], Xiao et al. set the costs of templates $cost_{T_1}$, $cost_{T_2}$, and $cost_{T_3}$ as 0, 5, 8, respectively. This setting still satisfies $2cost_{T_{k-1}} \geq cost_{T_k} > \frac{k}{k-1}cost_{T_{k-1}}$. In another work [52], Kuang et al. assumed that the cost of a template with more than 2 holes is always larger than the summation of the costs of the constituent templates, e.g., $cost_{T_3} > cost_{T_1} + cost_{T_2}$ and $cost_{T_4} > 2cost_{T2}$. This assumption is also compatible with our assumption.

Note that, when $cost_{T_3} = 3$ and $cost_{T_k}$ is an integer, it is easy to show that $cost_{T_k} = 2k - 1$ is the tightest setting for satisfying $2cost_{T_{k-1}} \geq cost_{T_k} > \frac{k}{k-1}cost_{T_{k-1}}$, $k = 3, 4, \cdots K$. That is, when $cost_{T_2} = 3$ and $cost_{T_k}$ is an integer, it holds that $cost_{T_k} \geq 2k - 1$.

4.2.2 Problem Formulation

Some involved notations are introduced as follows:

- d_c, the minimum conflict spacing;
- $d_{g_{min}}$, the minimum grouping spacing;
- $d_{g_{max}}$, the maximum grouping spacing;
- C_1, C_2, C_3, the colors of TPL;
- β, the weighting parameter between the conflict number and the total cost of templates, which is set as $\beta = 0.01$.

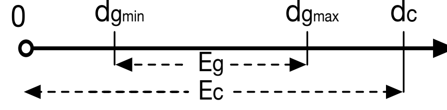


Figure 4.4: Conflict spacing and grouping spacing.

For the above notations, $d_{g_{min}} < d_{g_{max}} < d_c$. If the distance between two contacts is less than the minimum conflict spacing d_c, and the two contacts are assigned to the same mask without grouping, then a conflict is generated between the two contacts. In order to reduce the number of conflicts, we group some contacts together according to the template types defined in Section 4.2.1. Then the MTADT problem is defined as follows:

The mask and template assignment problem of DSA-TP P_0.

Given: Contact layer layout, the set of vertical and horizontal templates, three masks, parameter β.

Find: A mask assignment for all contacts, and groups of some of the contacts by available multi-hole templates.

Subject to: Every contact is assigned to only one of the three masks, and is assigned to only one of the templates. Moreover, all contacts in a template must be assigned to the same mask.

Objective: $|C|+\beta \cdot T_Cost$ is minimized, where $c_{ij} \in C$ denotes the conflict between contacts i and j, and $|C|$ is the number of conflicts, and T_Cost is the total cost of used templates.

4.3 Discrete Relaxation Method for Mask and Template Assignment of DSA with TPL

In this section, we construct the conflict grouping graph (CGG) for a layout, and propose a discrete relaxation of the MTADT problem using an ILP formulation. In order to obtain a better relaxation solution, we introduce some

valid inequalities for the ILP problem.

Before showing the discrete relaxation method for the MTADT problem, we introduce the definition of discrete relaxation and its propositions as follows.

Definition 4.3.1 (discrete relaxation)**.** Problem RP: $z^R = \min\{f^R(x) : x \in X^R\}$ is a discrete relaxation of problem P: $z = \min\{f(x) : x \in X\}$, if there exists an optimal solution $x^{R\star}$ of problem RP, and there exists an optimal solution $x^\star$ of problem P such that $f^R(x^{R\star}) \leq f(x^\star)$.

Proposition 4.3.2. If problem RP is a discrete relaxation of problem P, then $z^R \leq z$.

Proposition 4.3.2 means that, we will obtain a lower bound on the minimum value of the original problem by solving the discrete relaxation problem. Specially, we have

Proposition 4.3.3. Suppose that problem RP is a discrete relaxation of problem P. Let $x^{R\star}$ be an optimal solution of problem RP. If $x^{R\star}$ can be transformed to a feasible solution x of problem P, such that $f^R(x^{R\star}) = f(x)$, then x is an optimal solution of problem P.

For the discrete relaxation method, the function $f^R(x)$ and the solution set X^R must be carefully selected. Generally, we should select an X^R such that an optimal solution of problem RP can be transformed easily to a feasible solution of problem P, and the gap between the minimum values of problems P and RP is not too large.

4.3.1 Conflict Grouping Graph Construction

First, we define the conflict grouping graph as follows.

Definition 4.3.4 (conflict grouping graph, CGG)**.** The conflict grouping graph is defined as an undirected graph $CGG(V,\ E_c)$, where V is the set of vertices, E_c is the set of conflict edges.

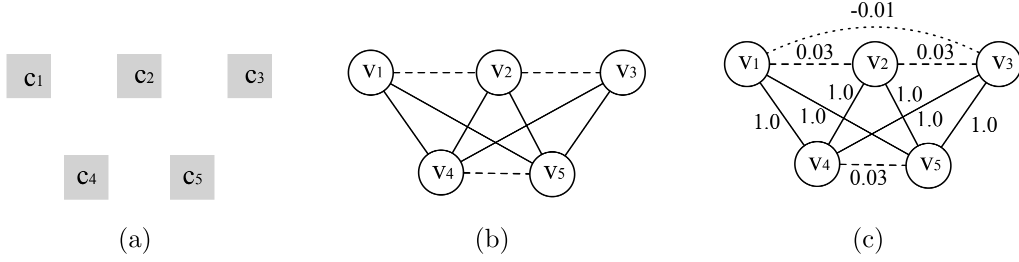


Figure 4.5: Conflict grouping graph construction.

If the distance between two contacts i and $j \in V$ is less than d_c, then there exists a conflict edge $e_{ij} \in E_c$ between them; if the distance between two contacts i and $j \in V$ is between $d_{g_{min}}$ and $d_{g_{max}}$, and i and j are in the vertical or horizontal line, then there exists a grouping edge $e_{ij} \in E_g$ between them. Obviously, $E_g \subseteq E_c$.

According to the distances between contacts, a layout with contacts is transformed to a conflict grouping graph. This can be achieved in $O(kn)$ runtime, where n is the number of contacts, and k is the maximum number of contacts within the minimum conflict spacing d_c of contacts. Figure 4.4 illustrates the conflict spacing and grouping spacing, and an example of conflict grouping graph construction is shown as Figure 4.5(b), where all lines are the conflict edges and dotted lines are the grouping edges.

4.3.2 Discrete Relaxation Based Mask Assignment

Discrete relaxation is an optimization method, which relaxes a hard minimization problem to an easier one by some relaxation techniques [59]. An optimal solution of the relaxation problem provides a lower bound on the minimal value of the original problem. In this section, we propose a way of discrete relaxation for the MTADT problem.

4.3.2.1 Weighted Conflict Grouping Graph Construction

In the conflict grouping graph CGG, two contacts connected by a conflict edge should be assigned to different masks or grouped by a template for MTADT.

In order to reduce the conflicts and the total cost of used templates, we need to decide which contacts should be grouped first, and which conflict edge could not be eliminated by grouping. We distinguish the conflict edges by weighting them, and then construct the weighted conflict grouping graph (WCGG). In order to handle the grouping, we introduce a definition of negative edge as follows.

Definition 4.3.5 (negative edge, ne)**.** A negative edge is an undirected edge with negative weight in a graph. If there exist two vertices i and j connected to the same vertex k by grouping edges, i.e., $ge_{ik} \in E_g$, $ge_{jk} \in E_g$, and contacts i, j are in the same vertical or horizontal line, then e_{ij} is added to the graph and called a negative edge.

Let E_n be the set of negative edges. We define the $WCGG$ as follows:

Definition 4.3.6 (weighted conflict grouping graph, $WCGG$)**.** The weighted conflict grouping graph is an undirected edge-weighted graph $WCGG(V,\ E,\ W)$, where V is the set of vertices, E is the set of edges, $E = E_c \cup E_n$, and W is the set of weights of edges in E.

The weighting rule for edge $e_{ij} \in E$ between contacts i and j is set as

$$w_{ij} = \begin{cases} 1.0, & \text{if } e_{ij} \in E_c - E_g; \\ 0.03, & \text{if } e_{ij} \in E_g; \\ -0.01, & \text{if } e_{ij} \in E_n. \end{cases}$$

Figure 4.5(c) shows an example of weighted conflict grouping graph.

According to the above weighting rule, we can see that:

1. If contacts i_1, i_2 are assigned to the same mask, and $e_{i_1 i_2} \in E_c - E_g$, then there exists a conflict between i and j, and the edge cost is 1.0;

2. If contacts i_1, i_2 are assigned to the same mask, and $e_{i_1 i_2} \in E_g$, then the edge cost is 0.03;

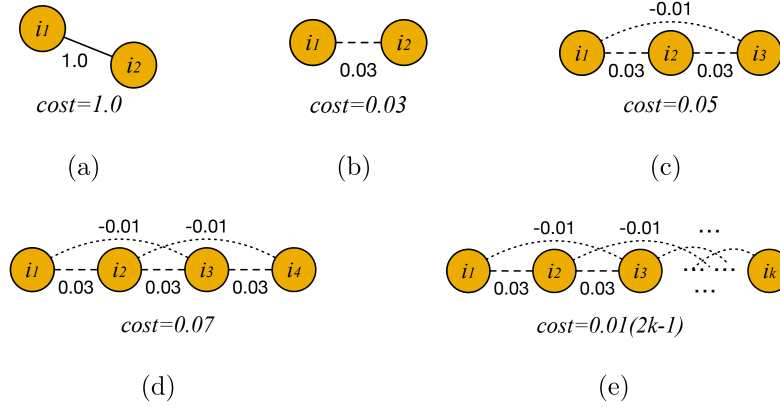


Figure 4.6: Total edge costs of different templates.

3. If contacts i_1, i_2, i_3 are assigned to the same mask, and $e_{i_1i_2} \in E_g$, $e_{i_2i_3} \in E_g$, $e_{i_1i_3} \in E_n$, then the edge cost is 0.05;

4. If contacts i_1, i_2, $\cdots$, i_k, $k = 3, 4$, are assigned to the same mask, and i_1, i_2, $\cdots$, i_k satisfy the T_k template condition. It can be deduced that there are $k-1$ grouping edges and $k-2$ negative edges, and then the edge cost is $0.03(k-1) - 0.01(k-2) = 0.01(2k-1)$;

Figures 4.6(a)-4.6(c) and Figure 4.6(e) show examples for the above cases 1-4.

4.3.2.2 Discrete Relaxation

We consider the following problem P_1 on the weighted conflict grouping graph:

$$\min \quad \sum_{e_{ij} \in E} w_{ij}(x_i == x_j) \tag{4.1}$$

$$\text{s.t.} \quad x_i \in \{1, 2, 3\}, \qquad \forall i \in V, \tag{4.1a}$$

where x_i denotes the assigned mask of vertex i.

Suppose X_0 and X_1 are the solution spaces of problems P_0 and P_1, respectively. For any solution x^0 of problem P_0, every contact has been assigned to a

mask and a template. However, for any solution x^1 of problem P_1, every contact has been assigned to a mask, but has not been assigned to a template. That means, for any $x^0 \in X_0$, we can get a solution $x^1 \in X_1$ from x^0 by omitting the template assignment.

Lemma 4.3.7. Suppose $x^0 \in X_0$ is transformed to $x^1 \in X_1$ by omitting the template assignment, X_0 and X_1 are the solution spaces of problems P_0 and P_1, respectively, and $f_0(x^0)$ and $f_1(x^1)$ are the objective functions of problems P_0 and P_1, respectively. Then $f_1(x^1) \leq f_0(x^0)$.

Proof. For any $x^0 \in X_0$, the total cost of problem P_0 is

$$f_0(x^0) = |C| + \beta \cdot T_Cost = |C| + 0.01 \sum_{k=2}^{K} cost_{T_k} |T_k|,$$

where K is the number of template types, $|C|$ is the number of total conflicts. If contacts i and j are assigned to the same mask, $e_{ij} \in E_c$, and i and j are not assigned to the same template, then a conflict is generated between i and j, i.e., $c_{ij} = 1$. $|T_k|$ is the number of used k-hole templates in x^0, where the contacts in the same k-hole template should be in the same mask, and satisfy the vertical or horizontal k-hole template conditions.

The cost of x^0 consists of conflict cost and template cost. x^1 is obtained from x^0 by omitting the template assignment. Let $E = E_T \cup E_D$, where E_T is the set of edges between contacts i and j in the same template of x^0. E_D is the set of edges between contacts i and j in different templates of x^0. Then

$$\begin{aligned} f_1(x^1) &= \sum_{e_{ij} \in E} w_{ij}(x_i^1 == x_j^1) \\ &= \sum_{e_{ij} \in E_D} w_{ij}(x_i^1 == x_j^1) + \sum_{e_{ij} \in E_T} w_{ij}(x_i^1 == x_j^1). \end{aligned}$$

First, for the contacts i and j that are in the same template of x^0, according to

the weight setting of problem P_1, it holds that

$$\begin{aligned}\sum_{e_{ij}\in E_T} w_{ij}(x_i^1 == x_j^1) &= \sum_{e_{ij}\in E_T} w_{ij}\\ &= \sum_{k=2}^{K}\{[0.03(k-1)-0.01(k-2)]|T_k|\}\\ &= 0.01\sum_{k=2}^{K}(2k-1)|T_k|\\ &\leq 0.01\sum_{k=2}^{K} cost_{T_k}|T_k|.\end{aligned}$$

The last inequality is due to $cost_{T_k} \geq 2k-1$ for $k \geq 2$.

Second, we consider the cost between the contacts i and j that are in different templates of x^0. Here, we only consider the cost between contacts i and j that are in the same mask, since if i and j are in different masks, then the cost is 0. Let E'_D be the set of edges $e_{ij} \in E_D$ and i and j are in the same mask. Let $E_{D_1} = E'_D \cap (E_c - E_g)$, $E_{D_2} = E'_D \cap E_g$, and $E_{D_3} = E'_D \cap E_n$. Then $E'_D = E_{D_1} \cup E_{D_2} \cup E_{D_3}$. Moreover, $|E'_D \cap E_c| = |E_{D_1} \cup E_{D_2}| = |E_{D_1}| + |E_{D_2}|$ which is the number of conflicts $|C|$. Then

$$\begin{aligned}\sum_{e_{ij}\in E_D} w_{ij}(x_i^1 == x_j^1) &= \sum_{e_{ij}\in E'_D} w_{ij}\\ &= \sum_{e_{ij}\in E_{D_1}} w_{ij} + \sum_{e_{ij}\in E_{D_2}} w_{ij} + \sum_{e_{ij}\in E_{D_3}} w_{ij}\\ &= |E_{D_1}| + 0.03|E_{D_2}| - 0.01|E_{D_3}|\\ &\leq |E_{D_1}| + |E_{D_2}| = |C|.\end{aligned}$$

Therefore, for any $x^0 \in X_0$, and for $x^1 \in X_1$ which is obtained from x^0 by

omitting the template assignment, we have

$$\begin{aligned} f_1(x^1) &= \sum_{e_{ij}\in E} w_{ij}(x_i^1 == x_j^1) \le |C| + 0.01\sum_{k=2}^{K} cost_{T_k}|T_k| \\ &= f_0(x^0). \end{aligned}$$

□

Theorem 4.3.8. Problem P_1 is a discrete relaxation of problem P_0.

Proof. Suppose x^{0*} and x^{1*} are optimal solutions of problems P_0 and P_1, respectively. By Lemma 4.3.7, we have

$$f_1(x^{1*}) \le f_1(x^{1'}) \le f_0(x^{0*}),$$

where $x^{1'}$ is obtained from x^{0*} by omitting the template assignment. Hence, problem P_1 is a discrete relaxation of problem P_0. □

Corollary 4.3.9. Suppose an optimal solution $x^{1*} \in X_1$ of problem P_1 is transformed to $x^{0*} \in X_0$ by some template assignment. If $f_0(x^{0*}) = f_1(x^{1*})$, then x^{0*} is an optimal solution of problem P_0.

Corollary 4.3.9 holds obviously from Theorem 4.3.8. By Theorem 4.3.8, the optimal value LB of problem P_1 is a lower bound on the optimal value OPT of problem P_0. By legalizing an optimal solution of problem P_1 to a feasible solution of problem P_0, we can obtain an upper bound UB on the optimal value OPT of problem P_0, and it holds $UB - OPT \le UB - LB$. Thus Theorem 4.3.8 can be used to evaluate the quality of our experimental results, i.e., the gap between our experimental result and the optimal value of problem P_0.

In order to solve the discrete relaxation problem P_1, we transform P_1 to an Integer Linear Programming (ILP) problem P_2 equivalently as follows:

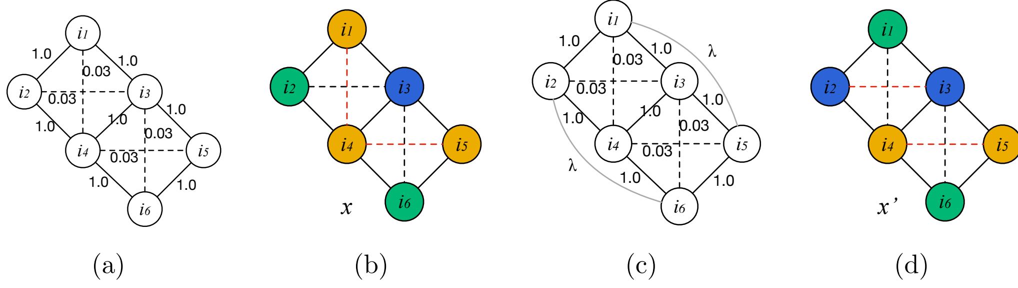


Figure 4.7: Corner incompatibility and triangle edges. (a) Weighted conflict grouping graph. (b) A solution obtained by solving problem P_2. (c) New weighted conflict grouping graph. (d) A solution obtained by solving problem P_2^+.

$$
\begin{aligned}
\min \quad & \sum_{e_{ij}\in E} w_{ij}c_{ij} && && (4.2)\\
\text{s.t.} \quad & x_{im} + x_{jm} \le 1 + c_{ij}, && \forall e_{ij} \in E,\ m = 1,2,3; && (4.2a)\\
& \sum_{m=1}^{3} x_{im} = 1, && \forall i \in V; && (4.2b)\\
& x_{im}, c_{ij} \in \{0,1\}, && \forall i \in V,\ \forall e_{ij} \in E,\ m = 1,2,3.
\end{aligned}
$$

In the above formulation, x_{im} is a binary variable, which denotes the assigned mask for contact i. If $x_{im} = 1$, then i is assigned to mask m. c_{ij} is a binary variable, which is used for indicating whether a conflict is generated between contacts i and j.

In problem P_2, Constraint (4.2a) is used to decide whether a conflict c_{ij} is generated between contacts i and j. That is, for $e_{ij} \in E$, if contacts i and j are in the same mask, then $c_{ij} = 1$. Constraint (4.2b) is used to select one of the three masks for contact i.

4.3.3 Improving the Lower Bound by Adding Valid Inequalities

The discrete relaxation problem P_2 can be solved for obtaining a lower bound of problem P_0. However, the gap between the lower bound and the

optimal value of problem P_0 may be large, and the obtained decomposition result might be of poor quality. The proposed discrete relaxation method is not only for finding a lower bound of the MTADT problem, but also for obtaining a good solution of the MTADT problem. Hence in this subsection we improve the lower bound provided by problem P_2 by adding some valid inequalities

In the weighted conflict grouping graph $WCGG$ of Figure 4.7(a), the solid lines are conflict edges, and their weights are $w = 1.0$ respectively; while the dotted lines are grouping edges, and their weights are $w = 0.03$ respectively. Figure 4.7(b) shows an optimal solution x of problem P_2, which has two conflict variables $c_{i_1 i_4}$ and $c_{i_4 i_5}$ equal to 1, and the objective value of problem P_2 is 0.06. However, contacts i_1, i_4 and i_5 cannot be assigned to the same template at the template assignment stage. Only one of the two conflict grouping edges $e_{i_1 i_4}$ and $e_{i_4 i_5}$ can be grouped by a T_2 template, another one would cause a conflict, and the total cost of P_0 is $0.01 \times cost_{T_2} + |C| = 1 + 0.01 \times cost_{T_2}$. Thus, the gap between the objective values of the discrete relaxation problems P_2 and the problem P_0 is $0.94 + 0.01 \times cost_{T_2}$. In the following, we let $\lambda = 0.94 + 0.01 \times cost_{T_2}$.

In order to reduce the gap, and further improve the quality of an obtained solution of problem P_0, we should handle the case as in Figure 4.7(b). We call the case as corner incompatibility (CI). More formally, corner incompatibility is that, there exist at least two grouping edges incident to a contact k called corner contact, and at least two grouping edges containing k are orthogonal. Since corner incompatibility is not allowed, we preclude in this section corner incompatibility by introducing some valid inequalities. The technique is adding some extra edges, and these edges are called triangle edges, which is defined as:

Definition 4.3.10 (triangle edge, te)**.** A triangle edge is an undirected edge in a graph. If there exist two vertices i and j connected to the same vertex k by grouping edges, i.e., $e_{ik} \in E_g$, $e_{jk} \in E_g$, and if $e_{ij} \notin E_c \cup E_n$, then e_{ij} is added to the graph and is called a triangle edge. Let E_t be the set of triangle edges.

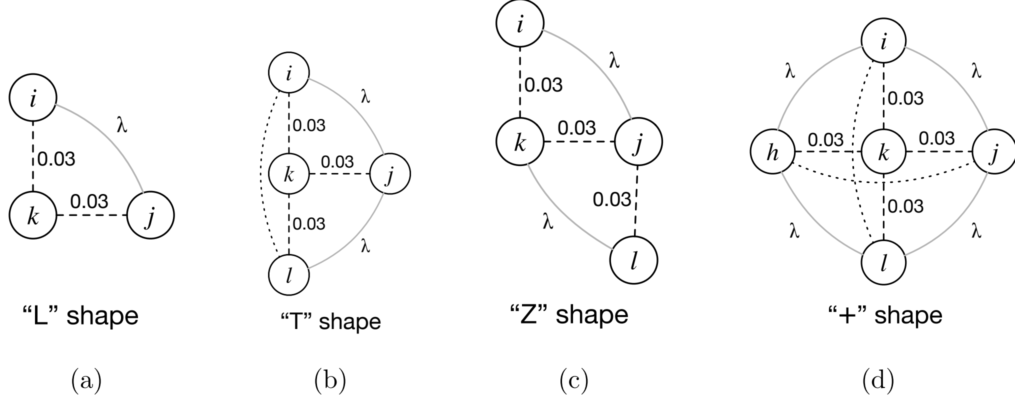


Figure 4.8: Structures with corner incompatibility. (a) "L" shape. (b) "T" shape. (c) "Z" shape. (d) "+" shape.

In Figure 4.8, there are four types of structures with corner incompatibility structure (CIS): i) "L" shape, $\{e_{ik},\ e_{jk}\} \subseteq E_g$ and $\{e_{ij}\} \subseteq E_t$ as shown in Figure 4.8(a); ii) "T" shape, $\{e_{ik},\ e_{jk},\ e_{lk}\} \subseteq E_g$, $\{e_{il}\} \subseteq E_n$ and $\{e_{ij},\ e_{jl}\} \subseteq E_t$ as shown in Figure 4.8(b); iii) "Z" shape, $\{e_{ik},\ e_{jk},\ e_{jl}\} \subseteq E_g$, and $\{e_{ij},\ e_{kl}\} \subseteq E_t$ as shown in Figure 4.8(c); and iv) "+" shape, $\{e_{ik},\ e_{jk},\ e_{lk},\ e_{hk}\} \subseteq E_g$, $\{e_{il},\ e_{hl}\} \subseteq E_n$ and $\{e_{ij},\ e_{jl},\ e_{lh},\ e_{hi}\} \subseteq E_t$ as shown in Figure 4.8(d).

In order to avoid the contacts in a CIS being assigned to the same mask, we modify the $WCGG$ as the $newWCGG$ by adding some triangle edges, and assign the weight of every triangle edge as $w = \lambda$. Then we add four kinds of new constraints to problem P_2, and call the new problem as P_2^+.

For the "L" shape structure, if the three contacts i, j and k are assigned to the same mask, then there exists at least a conflict due to corner incompatibility. In order to obtain a better solution, we add some new constraints for contacts i, j and k of the "L" shape structure to avoid the three contacts i, j and k being assigned to the same mask. These constraints are

$$x_{im} + x_{km} + x_{jm} \leq 2 + c_{ij}, \qquad m = 1, 2, 3, \tag{4.3a}$$

where contact k is a corner contact. Constraint (4.3a) is used to restrict that, if the contacts i, j and k are assigned to the same mask, then the conflict variable of triangle edge $e_{ij} \in E_t$ has $c_{ij} = 1$.

For the "T" shape structure, if the four contacts i, j, l and k are assigned to the same mask, then there exists at least a conflict due to corner incompatibility. Similarly, we add some valid inequalities for the four contacts of the "T" shape structure. These constraints are

$$x_{im} + x_{km} + x_{jm} \leq 2 + c_{ij} + c_{jl}, \qquad m = 1, 2, 3; \tag{4.4a}$$

$$x_{jm} + x_{km} + x_{lm} \leq 2 + c_{ij} + c_{jl}, \qquad m = 1, 2, 3, \tag{4.4b}$$

where contact k is a corner contact. Constraints 4.4(a)-4.4(b) are used to restrict that, if the three contacts i, j and k (or j, l and k) or the four contacts are assigned to the same mask, then at least one of the conflict variables of the triangle edges e_{ij} and e_{jl} is equal to 1.

For the "Z" shape structure, if the four contacts i, j, l and k are assigned to the same mask, then there exists at least a conflict due to corner incompatibility. Similarly, we add some valid inequalities for the four contacts of the "Z" shape structure. These constraints are

$$x_{im} + x_{km} + x_{jm} \leq 2 + c_{ij} + c_{kl}, \qquad m = 1, 2, 3; \tag{4.5a}$$

$$x_{jm} + x_{km} + x_{lm} \leq 2 + c_{ij} + c_{kl}, \qquad m = 1, 2, 3, \tag{4.5b}$$

where contacts k and j are corner contact. Constraints 4.5(a)-4.5(b) are used to restrict that, if the three contacts i, j and k (or j, l and k) or the four contacts are assigned to the same mask, then at least one of the conflict variables of the triangle edges e_{ij} and e_{kl} is equal to 1.

For the "+" shape structure, if the five contacts i, j, l, h and k are assigned to the same mask, then there exists at least two conflicts due to corner incompatibility. Similarly, we add some valid inequalities for the five contacts of the "+" shape structure. These constraints are

$$x_{im} + x_{km} + x_{jm} \leq 2 + c_{ij} + c_{jl}, \qquad m = 1, 2, 3; \tag{4.6a}$$

$$x_{jm} + x_{km} + x_{lm} \leq 2 + c_{jl} + c_{lh}, \qquad m = 1, 2, 3; \tag{4.6b}$$

$$x_{lm} + x_{km} + x_{hm} \leq 2 + c_{lh} + c_{hi}, \qquad m = 1, 2, 3; \tag{4.6c}$$

$$x_{hm} + x_{km} + x_{im} \leq 2 + c_{hi} + c_{ij}, \qquad m = 1, 2, 3, \tag{4.6d}$$

where contact k is a corner contact. Constraints 4.6(a)-4.6(d) are used to restrict that, if the three contacts in an "L" shape CIS or the four contacts in a "T" shape CIS are assigned to the same mask, then at least one of the conflict variables of the triangle edges is equal to 1. If the five contacts in a "+" shape CIS are assigned to the same mask, then two conflict variables among the triangle edges e_{ij}, e_{jl}, e_{lh} and e_{hi} are equal to 1.

Actually, according to our experiments, the number of "T" shapes, "Z" shapes and "+" shapes CIS is very small. After adding the CIS constraints, the $newWCGG$ of the structure in Figure 4.7(b) is constructed as Figure 4.7(c). Then by solving problem P_2^+, we can obtain a discrete relaxation solution as Figure 4.7(d) instead of Figure 4.7(b).

Theorem 4.3.11. Problem P_2^+ is still a discrete relaxation of problem P_0.

Proof. Suppose that f_2^+, f_2 and f_0 are the objective functions of problems P_2^+, P_2 and P_0, respectively. Suppose x^0 is an optimal solution of problem P_0, and x is obtained from x^0 by omitting the template assignment, then x is a solution of both problems P_2^+ and P_2, respectively. The difference between problems P_2^+ and P_2 is the triangle edges of the three types of CIS. We analyze the cost of

the three types of CIS in problem P_2^+.

Suppose S_L is an "L" shape CIS, i, j and k are the three contacts of S_L, k is a corner contact and te_{ij} is the triangle edge. Let x_L ={x_{i1}, x_{i2}, x_{i3}, x_{j1}, x_{j2}, x_{j3}, x_{k1}, x_{k2}, x_{k3}}. Apparently, x_L is a part of x. Let x_L^0 be the part of the solution x^0 of problem P_0 for S_L. If i, j are assigned to different masks, then the conflict variable of the triangle edge te_{ij} has $c_{ij} = 0$ for minimizing the objective. If the mask of contact k is different from both of the masks of i and j, then $f_2^+(x_L) = f_0(x_L^0) = 0$; if the mask of k is the same as one of the masks of i and j, say i, then $f_2^+(x_L) = 0.03$. And for x_L^0, if the grouping edge ge_{ki} is guided by a multi-hole template, then $f_0(x_L^0) = 0.01 \times cost_{T_2} \geq 0.03$; otherwise a conflict is generated between k and i, and $f_0(x_L^0) = 1$. Anyway, it holds that $f_2^+(x_L) \leq f_0(x_L^0)$.

If all contacts i, j and k of S_L are assigned to the same mask, then constraint 4.3(a) will force c_{ij} to be 1. On the one hand, $f_2^+(x_L) = 0.06 + \lambda = 1 + 0.01 \times cost_{T_2}$. On the other hand, since S_L is an "L" shape CIS, (1) if in x_L^0, S_L needs a conflict and a multi-hole template to contain one of the grouping edges in S_L, then the decomposition cost for S_L is $f_0(x_L^0) = |C| + T_Cost = 1 + 0.01 \times cost_{T_2}$; (2) if in x_L^0, S_L needs two conflicts, then $f_0(x_L^0) = |C| + T_Cost = 2$. Thus $f_2^+(x_L) \leq f_0(x_L^0)$.

Hence it holds that $f_2^+(x_L) \leq f_0(x_L^0)$ for S_L. For the "T" shape, "Z" shape and "+" shape CIS, we can prove similarly that the statement still holds.

Therefore, for an optimal solution x^0 of P_0, and x obtained from x^0 by omitting the template assignment, it holds that $f_2^+(x) \leq f_0(x^0)$. Thus P_2^+ is still a discrete relaxation of problem P_0. □

Similarly, Corollary 4.3.9 still holds for problem P_2^+.

We use the Branch-and-Bound approach in the software package CPLEX [2] to solve problem P_2^+. Since problem P_2^+ is hard to solve in the large scale case, we introduce some graph reduction techniques to reduce the size of the prob-

lem, such that it is easy to solve using the Branch-and-Bound approach. In this chapter, the used graph reduction techniques include: connected components calculation, vertices with degree less than 3 removal and 2-edge connected components calculation [36, 53, 106]. The vertices with degree less than 3 removal technique will remove some contacts. Since these removed contacts can be easily assigned to templates without any cost of assignment, they would be handled after the template assignment stage in Section 4.4. Furthermore, for the 2-edge connected components calculation, if vertices i and j are connected by a bridge and are assigned to the same mask at the mask assignment stage, then a conflict is generated between i and j. We can eliminate the conflict by rotating the colors of one of the 2-edge connected components such that i and j are in different masks. Moreover, we do not need to consider template assignment for i and j.

4.4 Template Assignment

After obtaining a discrete relaxation solution of problem P_0 by solving problem P_2^+, we must decide the template assignment for the discrete relaxation solution. The solution of problem P_2^+ divides the initial layout (except removed contacts) into three masks, and we obtain three decomposed layouts L_1, L_2, L_3. Then we should consider the template assignment for every decomposed layout L_m $(m = 1, 2, 3)$, which is described as follows:

Template assignment for each decomposed mask

Given: A decomposed contact layout, a set of vertical and horizontal templates.

Find: A template assignment of contacts, which groups some of the contacts by available multi-hole templates.

Subject to: Every contact is assigned to only one of the templates.

Objective: $|C| + \beta \cdot T_Cost$ is minimized, where $|C|$ is the number of conflicts, and T_Cost is the total cost of used templates.

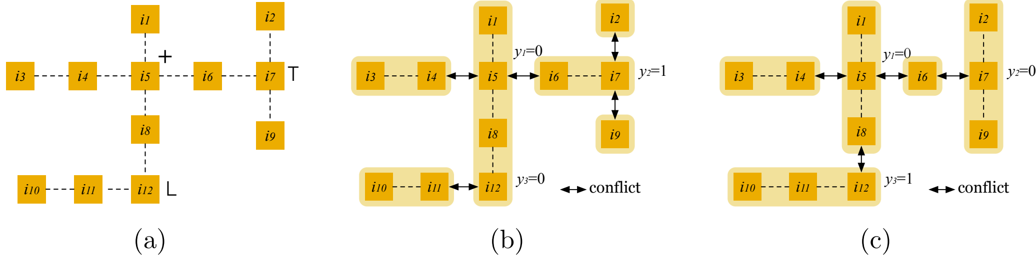


Figure 4.9: Template assignment for layout graph. (a) A layout graph. (b)(c) Two template assignment results.

For every decomposed layout L_m $(m = 1, 2, 3)$, we generate a layout graph LG_m, and consider the template assignment problem on LG_m. The definition of LG_m is:

Definition 4.4.1 (layout graph)**.** The layout graph of a layout L_m is a graph $LG_m(C_m, E_{cm})$, where C_m is the set of contacts in the decomposed layout L_m, E_{cm} is the set of conflict edges of the conflict grouping graph $CGG(V, E_c)$ which are vertical or horizontal and connect only contacts in L_m.

Let E_{gm} be the set of grouping edges between any two contacts which are in L_m. Obviously, $E_{gm} \subset E_{cm}$. First we compute all connected components CC of layout graph LG_m $(m = 1, 2, 3)$, and then deal with every connected component one by one. Since the edges in E_{cm} are either vertical or horizontal, the considered contacts in every connected component of $LG_m(C_m, E_{cm})$ are lined up vertically or horizontally. Figure 4.9(a) shows a connected component with 12 contacts of a layout graph LG_m, where the dotted lines are grouping edges E_{gm}.

There exists some corner contacts in a CC. A corner contact may be one of the three types: i) it is a corner contact belonging to a "+" shape structure; ii) it is a corner contact belonging to a "T" shape structure; iii) it is a corner contact belonging to an "L" shape structure. In Figure 4.9(a), i_5 is a corner contact belonging to a "+" shape structure, i_7 is a corner contact belonging to a "T" shape structure, and i_{12} is a corner contact belonging to an "L" shape

structure.

For every isolated vertex in LG_m, it is assigned to a single-hole template. For every CC without corner contact, all contacts in CC are lined up vertically or horizontally, and we assign these contacts greedily to a template with the most holes first. Otherwise, for the other complicated CC of LG_m, we consider a heuristic assigning method as follows.

Note that, once all corner contacts in a CC have been assigned to vertical or horizontal templates, then the other contacts can be assigned optimally using the method for a CC without corner contact. Hence, to obtain an optimal template assignment for a complicated CC, we only need to decide whether a corner contact i is assigned to a vertical or a horizontal template. Binary variable y_i is used to indicate if i is assigned to a vertical template or not. That is, $y_i = 0$ denotes that i is assigned to a vertical template; $y_i = 1$ denotes that i is assigned to a horizontal template.

In the experiments, the size of every CC is very small. Hence we check all possible solutions of y to find an optimal template assignment. It must be noted that, if the size of CC is large, then we may find a good solution by some greedy tricks or local search algorithms.

Figures 4.9(b) and 4.9(c) show two template assignment results of Figure 4.9(a). When y is (0, 1, 0), then the result is shown as Figure 4.9(b), and $|C| = 5$, $\beta \cdot T_Cost = 0.01 \times (cost_{T_4} + 3cost_{T_2} + 2cost_{T_1})$. When y is (0, 0, 1), then the result is shown as Figure 4.9(c), and $|C| = 4$, $\beta \cdot T_Cost = 0.01 \times (3cost_{T_3} + cost_{T_2} + cost_{T_1})$

4.5 Experimental Results

Our discrete relaxation based mask and template assignment method for DSA with TPL of general layout is programmed in C++, and run on a personal computer with 2.4GHz CPU, 4GB memory and the Linux operating system. We

test our method on the benchmarks provided by Kuang et al. [52]. The width of contacts and the minimum conflict spacing are scaled to $10nm$ to reflect the pitch in advanced technology nodes.

In order to evaluate our method, we design two experiments with different minimum conflict spacings. In the first experiment, the minimum conflict spacing d_c is set as 41nm, the minimum grouping spacing $d_{g_{min}}$ and the maximum grouping spacing $d_{g_{max}}$ are set as $10nm$ and $30nm$, respectively, as in [52]. Note that, increasing d_c has the same effect as shrinking the sizes of nodes. In the second experiment, the minimum conflict spacing d_c is set as $51nm$, the minimum grouping spacing $d_{g_{min}}$ and the maximum grouping spacing $d_{g_{max}}$ are set as $10nm$ and $40nm$, respectively. For simplification, the costs of templates are set as $cost_{T_1} = 0$, $cost_{T_2} = 3$, $cost_{T_k} = 2k - 1$, $k = 3, 4, \cdots, K$, in our experiments.

Statistics of the two experiments are listed in Table 4.1. In the table, for each benchmark, every data in the column $|V|$ is the number of contacts, and every data in the columns $|E_c|$ or $|E_g|$ is the number of conflict edges or grouping edges in the conflict grouping graph CGG, respectively. Moreover, every data in the column "Ratio" is the ratio of the number of conflict edges to the number of contacts for every benchmark. From the row "Ratio", it can be seen that the number of conflict edges in the second experiment is almost 1.87× the number in the first experiment, while the numbers of grouping edges of the two experiments are almost the same.

4.5.1 First Experiment

In [52], the listed experimental results only use T_2 template to guide contacts for experimental comparisons. Hence, in this experiment, we also only use T_2 template for fair comparisons. In the mask assignment stage, we delete all negative edges he_{ij} in the set E_n. And in the template assignment stage, regardless of the techniques in Section 4.4, given a layout graph, we generate guiding templates T_2 from left to right and up to down of the positions of the contacts.

Table 4.1: Statistics of benchmarks for mask and template assignment for DSA with TPL

Circuits	$\|V\|$	First experiment			Second experiment		
		$\|E_c\|$	Ratio	$\|E_g\|$	$\|E_c\|$	Ratio	$\|E_g\|$
dp1_Via1	307739	203073	0.66	53120	373415	1.21	53228
dp1_Via2	256885	174502	0.68	33473	333247	1.30	33473
ed1_Via1	400123	186480	0.47	56450	370029	0.92	56450
ed1_Via2	301607	119797	0.40	24587	228241	0.76	24587
fft_Via1	99509	61926	0.62	16306	113993	1.15	16308
fft_Via2	90114	62944	0.70	12456	117854	1.31	12456
mm_Via1	429664	267546	0.62	65426	487013	1.13	65585
mm_Via2	341789	218668	0.64	39882	409226	1.20	39887
pb1_Via1	79635	44668	0.56	11684	82017	1.03	11719
pb1_Via2	59110	30518	0.52	6752	58036	0.98	6752
Avg.	236617	137012	0.59	32013	257307	1.10	32044
Ratio		1.00	1.00	1.00	1.87	1.87	1.001

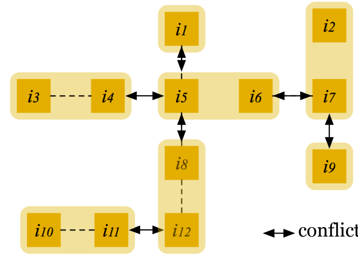


Figure 4.10: T_2 template only assignment for the layout graph in Figure 4.9(a).

We group as many as possible the contacts by T_2 templates, and then the rest contacts are guided by single-hole templates. This trick is a greedy approach which would find an optimal assignment. Figure 4.10 shows a T_2 template only assignment for the layout graph in Figure 4.9(a).

We run the binary files of DAC'15 [7], ILP [7] and ASP-DAC'16 [52] provided by Dr. Kuang. The comparison results of DAC'15 [7], ILP [7], ASP-DAC'16 [52] and ours are listed in Table 4.2. In Table 4.2, every data in the column "$|C|$" is the number of conflicts, and every data in the column "$|T_2|$" is the number of used T_2 templates for every benchmark. Moreover, the data in the columns "COST" are the total cost of decomposition for DSA with TPL, which

Table 4.2: Comparison results of mask and template assignment for DSA with TPL, $d_c = 41nm$, $d_{g_{min}} = 10nm$, $d_{g_{max}} = 30nm$

Circuits	DAC'15 [7]				ILP [7]				ASP-DAC'16 [52]				Ours				Lower
	$\|C\|$	$\|T_2\|$	COST	CPU(s)	$\|C\|$	$\|T_2\|$	COST	CPU(s)	$\|C\|$	$\|T_2\|$	COST	CPU(s)	$\|C\|$	$\|T_2\|$	COST	CPU(s)	DRS
dp1_Via1	13	29757	905.71	1.80	10	24	10.72	164.99	10	24	10.72	1.64	10	24	10.72	3.74	10.72
dp1_Via2	24	18191	569.73	1.59	0	400	12	498.37	0	400	12	2.91	0	400	12	3.30	12
ed1_Via1	0	34233	1026.99	2.17	0	1	0.03	215.98	0	1	0.03	1.77	0	1	0.03	5.26	0.03
ed1_Via2	3	16395	494.85	1.60	0	90	2.7	168.42	0	90	2.7	2.61	0	90	2.7	3.92	2.7
fft_Via1	0	9183	275.49	0.76	0	0	0	53.65	0	0	0	0.59	0	0	0	1.28	0
fft_Via2	11	6621	209.63	0.70	0	175	5.25	515.29	0	175	5.25	1.43	0	175	5.25	1.15	5.25
mm_Via1	14	37897	1150.91	2.55	11	25	11.75	220.92	11	25	11.75	1.94	11	25	11.75	5.56	11.75
mm_Via2	18	22361	688.83	2.01	0	384	11.52	297.92	0	384	11.52	3.00	0	384	11.52	4.43	11.52
pb1_Via1	1	7191	216.73	0.63	1	9	1.27	38.8	1	9	1.27	0.49	1	9	1.27	1.03	1.27
pb1_Via2	4	40044	1205.32	0.53	0	49	1.47	49.3	0	49	1.47	1.03	0	49	1.47	0.76	1.44
Avg.	9	22187	674.42	1.43	2	115	5.67	222.36	2	115	5.67	1.74	2	115	5.67	3.05	5.67
Ratio	4.19	191.60	120.99	0.47	1.00	1.00	1.00	73.02	1.00	1.00	1.00	0.57	1.00	1.00	1.00	1.00	1.00

Table 4.3: Comparison results of mask and template assignment for DSA with TPL, $d_c = 51nm$, $d_{g_{min}} = 10nm$, $d_{g_{max}} = 40nm$

Circuits	ASP-DAC'16 [52]						Ours								Lower
	$\|C\|$	$\|T_2\|$	$\|T_3\|$	T_Cost	COST	CPU(s)	$\|C\|$	$\|T_2\|$	$\|T_3\|$	$\|T_4\|$	T_Cost	$COST_1$	COST	CPU(s)	DRS
dp1_Via1	3690	5219	42	158.67	3848.67	67.5	3413	5100	137	6	160.27	3579.15	3573.27	31.62	3557.61
dp1_Via2	7422	6189	74	189.37	7611.37	378.5	6943	6128	178	11	193.51	7147.29	7136.51	61.36	7108.42
ed1_Via1	2297	3395	18	102.75	2399.75	40.46	2159	3479	52	3	107.18	2269.12	2266.18	29.01	2258.25
ed1_Via2	1667	1998	6	60.24	1727.24	39.74	1572	1993	24	0	60.99	1632.99	1632.99	22.60	1631.69
fft_Via1	743	1277	8	38.71	781.71	11.18	694	1281	30	1	40.00	734.98	734.00	8.57	730.86
fft_Via2	2544	2453	40	75.59	2619.59	110.3	2380	2429	67	2	76.36	2458.32	2456.36	28.43	2447.43
mm_Via1	3729	5464	22	165.02	3894.02	61.36	3576	5398	94	2	166.78	3744.74	3742.78	37.18	3732.44
mm_Via2	8165	6745	84	206.55	8371.55	381.3	7638	6678	202	13	211.35	7862.09	7849.35	86.10	7811.86
pb1_Via1	752	866	2	26.08	778.08	10.53	727	851	11	0	26.08	753.08	753.08	5.72	751.58
pb1_Via2	833	729	8	22.27	855.27	46.77	737	699	10	0	21.47	758.47	758.47	7.39	755.97
Avg.	3184	3433	30	104.5	3288.7	114.7	2984	3403	80	4	106.4	3094.0	3090.3	31.8	3078.6
Ratio	1.07	1.01	0.38	0.98	1.06	3.61	1.00	1.00	1.00		1.00	1.00	1.004	1.00	1.00

are calculated by COST=$|C| + 0.01 \times cost_{T_2} \times |T_2|$. From the four columns "COST", we can see that, comparing with DAC'15, the results of ILP, ASP-DAC'16 and Ours are significantly better for the mask and template assignment problem of DSA-TP.

For all benchmarks with $d_c = 41nm$ and $d_{g_{max}} = 30nm$, ILP, ASP-DAC'16 and our method get optimal solutions. In fact, every data in the column "DRS" is the optimal value of problem P_2^+ for every benchmark, which is a lower bound on the optimal value of problem P_0, since P_2^+ is a discrete relaxation of problem P_0. By the similar claim as Corollary 4.3.9, if the assignment cost (column "COST") is equal to the objective value of P_2^+ (column "DRS"), then we obtain an optimal mask and template assignment. Comparing columns "COST" in "ILP", "ASP-DAC'16" and "Ours" with column "DRS", it can be seen that most of the results of the four columns are equal, which means that all the three methods obtain optimal solutions for most of the benchmarks with $d_c = 41nm$ and $d_{g_{max}} = 30nm$.

Although the method in ILP [7] can evaluate the quality of the experimental results as our method, it can be found that the computing time of the ILP is 73× more than ours. In addition, although the method in ASP-DAC'16 [52] is about twice faster than our method, it cannot evaluate the quality of the results. This is due to that, the simultaneous assignment method for DSA with TPL in [52] achieves decomposition basing on an off-line Look-Up Table for matching 3-colorable sub-graphs. When the number of nodes is small, the method can quickly match all possible 3-colorable sub-graphs. However, if the number of nodes is large, the Look-Up Table would be too large to match all sub-graphs and cannot find all possible 3-colorable sub-graphs. In order to show the issue of the simultaneous assignment method [52], we design another experiment in the following subsection.

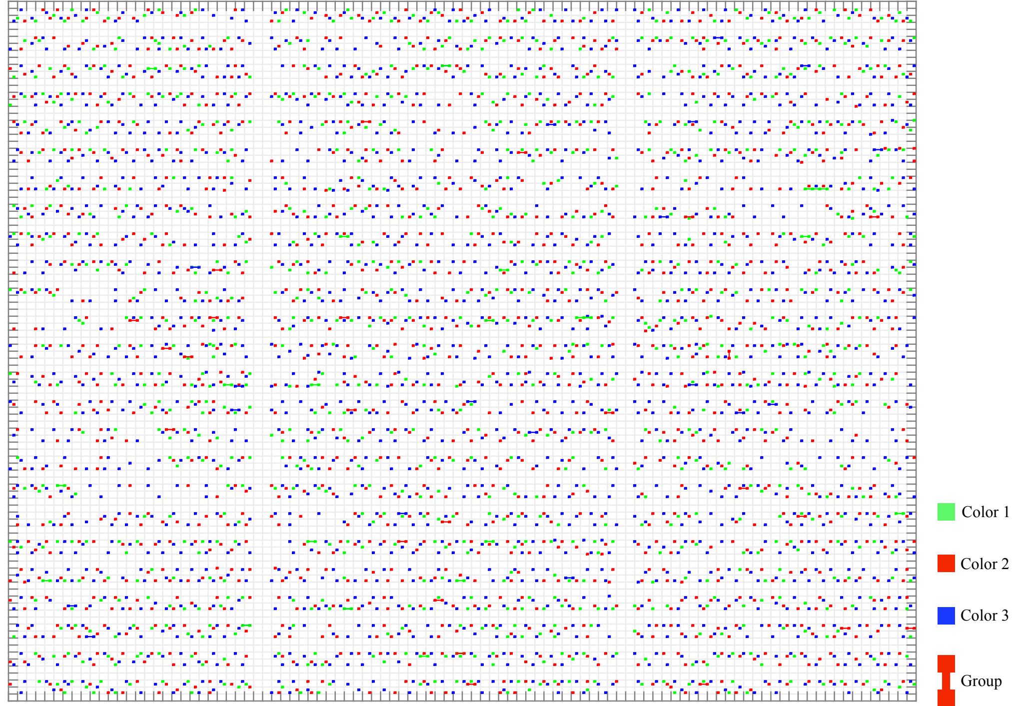


Figure 4.11: Mask and template assignment for benchmark dp1_Via1 with $d_c = 51nm$ and $d_{g_{max}} = 40nm$.

4.5.2 Second Experiment

To further evaluate the effectiveness of our assignment method on more dense layout, we perform another experiment on the benchmarks with $d_c = 51nm$ and $d_{g_{max}} = 40nm$. The comparison results are listed in Table 4.3. In the table, $|T_3|$ and $|T_4|$ are the numbers of used T_3 templates and T_4 templates, respectively, and T_Cost is the total cost of used templates, which is calculated by T_Cost=$0.01 \times (cost_{T_2} \times |T_2| + cost_{T_3} \times |T_3| + cost_{T_4} \times |T_4|)$. The notation COST is the total cost by the respective method for the assignment problem, and is calculated by COST=$|C|$+T_Cost. The other notations are the same as those in the first experiment.

The binary files of [52] only use T_2 and T_3 templates to guide the group contacts, while our method does not restrict which type of template is used. Since the experimental results of our method only contain templates T_k, $k = 2, 3, 4$, for fair comparisons, every T_4 template is further split into a T_3 template

and a single-hole template, like Figure 4.3(b). It is obvious that a conflict would be generated by this split. Correspondingly, for our method the cost of used T_4 templates, i.e., $0.01 \times cost_{T_4} \times |T_4|$, is replaced by the cost of used T_3 templates and the cost of generated conflicts, i.e., $(0.01 \times cost_{T_3} + 1)|T_4|$. Then COST_1 of our method is calculated by COST_1=$|C| + T_Cost - 0.01 \times cost_{T_4} \times |T_4| + (0.01 \times cost_{T_3} + 1)|T_4| = |C| + T_Cost + 0.98|T_4|$.

First, we compare the column "COST" in "ASP-DAC'16" with the column "COST_1" in "Ours". Assuming that only T_2 and T_3 templates are used, we can reduce the total cost of decomposition by 6%, and for every benchmark, we achieve a better result. Moreover, we can reduce the total number of conflicts by 7%. In addition, the CPU time of ASP-DAC'16 is 3.61× more than ours. Then, we compare the column "COST" in "Ours" with the column "DRS", our total cost averagely is only 0.4% greater than the lower bound. Hence, the gap between our total cost and the optimal value should be less than 0.4%. The experimental results indicate that our method almost obtains optimal solutions for all benchmarks. At last, a part of the experimental result for the benchmark dp1_Via1 with $d_c = 51nm$ and $d_{gmax} = 40nm$ is shown as Figure 4.11.

It must be noted that the quality evaluation of a solution is significant for an NP-hard problem which has real applications. If we know the gap between the obtained result and the optimal value, then we know whether an instance of the problem is solved or not. Typically, for this experiment with denser setting, our method shows that the gap between our result and the optimal value is very tiny, but the number of conflicts is still large. This indicates that one or more masks might be needed for further eliminating the conflicts.

4.6 Summary

In this chapter, we consider the contact layer mask and template assignment problem of DSA with TPL for general layout, and propose a discrete relaxation

method. First, we introduce negative edges in the conflict grouping graph, and weight the edges of the conflict grouping graph. Then we formulate a discrete relaxation problem of the contact layer assignment problem of DSA with TPL. For obtaining better results, we introduce triangle edges in the weighted conflict grouping graph, and thus introduce some valid inequalities in the discrete relaxation problem. We transform the discrete relaxation solution to a legal solution of the initial problem by addressing the template assignment problem on the layout graph. Our discrete relaxation based method can estimate the gap between the obtained solution and the optimal solution in the experiment, which is meaningful for the NP-hard problem. Furthermore, our experimental results show that the gaps between the obtained solutions and the optimal solutions are very small. Specially, the discrete relaxation approach verifies the optimality of our experimental results of sparse benchmarks since the gaps are 0. Finally, it must be remarked that we only consider the 1-D templates in this chapter. However, the proposed method can be extended to handle more general templates like 2×2, which needs further careful investigation.

Chapter 5 Graph Based Redundant Via Insertion and Guiding Template Assignment for DSA-MP

5.1 Introduction

In an integrated circuit (IC) layout, a via provides the connection between two net segments from adjacent metal layers. A single via may fail partially or completely because of various reasons, such as random defects, cut misalignment and electro migration or thermal stress [85, 110]. A partial via failure may induce timing problems due to the increase of contact resistance and parasitic capacitance, while a complete via failure will produce a broken net in a circuit [85]. These failures may heavily hinder the functionality and yield of a circuit. Therefore, reducing yield loss due to via failure is one of the most important problems in the IC design flow.

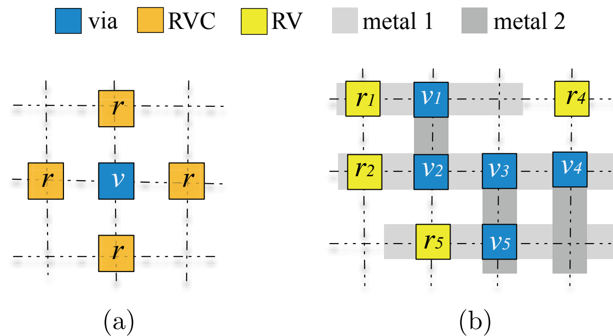


Figure 5.1: An example of redundant via insertion. (a) Four locations of redundant via candidates. (b) A feasible redundant via insertion result.

A promising method for improving via yield and reliability is adding a redundant via adjacent to every via [85,110], enabling via failure to be tolerated. For each via, the position of a redundant via should meet two conditions: first, the redundant via should be aligned with and be close to the via, as shown in Figure 5.1(a), where four redundant via candidates (RVC) r are next to the via

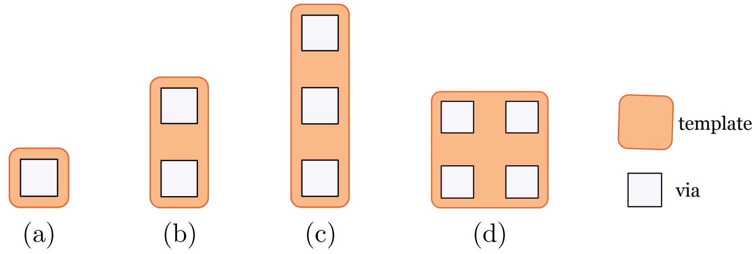


Figure 5.2: Four usable types of guiding templates. (a) t_1. (b) t_2. (c) t_3. (d) t_4.

v; second, the inserted redundant via should not cause any short circuit, i.e., it can only be inserted at a free space not occupied by any metal wire, or at a space occupied by only one metal wire which is on the same net as the via [20, 94, 97]. A via is an alive via if it has free spaces for inserting a redundant via, otherwise it is called a dead via [85]. As shown in Figure 5.1(b), vias v_1, v_2, v_4 and v_5 are alive vias, while via v_3 is a dead via since it has no free space for redundant via insertion. Some works have concerned the redundant via insertion (RVI) problem at the routing stage [60, 71], or at the post routing stage [56, 57, 61]. In order to improve yield and reliability, considering the problem is necessary at both the during routing and the post routing. In this chapter we consider redundant via insertion at the post routing stage.

The recent proposed block copolymer directed self-assembly (DSA) lithography technology is considered as a promising fit for via layers in the $7nm$ technology node and beyond [98, 99]. Many significant improvements have been made on manufacturing, modeling and simulation of DSA, especially on the graphoepitaxy DSA [64, 87]. The block copolymers form cylinders, and by removing cylinders, the material can be used to fabricate vias. To generate irregularly distributed vias using DSA, guiding templates surrounding vias are required [34, 55]. These guiding templates are manufactured by conventional optical lithography, and thus the resolution is limited. To improve the resolution, some closed vias may be put in a multi-hole guiding template [64, 90].

Theoretically, guiding templates can be of any shape. However, overlay accuracy varies with different guiding template shapes. Empirically, a highly

oriented and aligned via arrangement could have better overlay accuracy [70]. Hence, to guarantee a reasonable overlay accuracy, only some regular guiding templates with few holes are used to surround vias. In this chapter, the usable four types of guiding templates t_1, t_2, t_3 and t_4 are shown in Figures 5.2(a)–5.2(d) as of [70].

Usually, assigning vias to guiding templates is important in the DSA based designs. With the feature size decreasing, the density of vias dramatically increases in the via layer correspondingly, and single patterning only cannot obtain the required resolution of a guiding template. Hence multiple patterning (MP) lithography technologies are necessary for higher resolution [7, 52, 77, 93]. The guiding template assignment (GTA) for DSA-MP problem has been well investigated in recent years [7, 52, 77, 93].

Fang *et al.* [39] first considered the redundant via insertion and guiding template assignment for DSA with single patterning problem, where they introduced two ILP formulations. In the better ILP, firstly, all redundant via candidates are found, and then all feasible guiding templates (via patterns in [39]) for every via or RVC are detected. Furthermore, at most one of the guiding templates would be selected for a via or RVC. Since the better ILP in [39] considers all RVCs for every via and all the feasible guiding templates for every via or RVC, there are too many variables and constraints, even though some ILP reduction techniques are introduced. In addition, they proposed a graph based method to obtain a heuristic solution by solving the maximum independent set problem. In order to improve the insertion rate and the manufacture rate, Fang *et al.* [38] introduced metal wire perturbation to the RVI and GTA for DSA-SP problem. By perturbing some metal wires, it becomes more free for the insertion of redundant vias, but in the cost of increasing the wirelength. Hung *et al.* [47] introduced dummy vias for further improving the insertion rate and the manufacture rate. By inserting some dummy via, more guiding templates can be used for patterning vias.

For a dense layout, it may not be easy to obtain a good enough insertion rate and manufacture rate under single patterning, hence multiple patterning is needed. Ou *et al.* [70] first investigated the redundant via insertion and guiding template assignment for DSA with multiple patterning problem, and gave an ILP formulation based on a constrained weighted matching with MP. Similarly, the ILP in [70] considers all possible guiding template assignments (GTA in [70]) for every via. Consequently, the ILP has too many variables and constraints. Apparently, it is time consuming to solve the ILP problem, hence a linear program relaxation based approximate algorithm was proposed for fast obtaining a relaxation solution. However, a solution obtained by randomized rounding of the relaxation solution may be far away from an optimal solution.

Conventionally, the redundant via insertion problem and the guiding template assignment for DSA-MP problem are considered at two separate stages. After obtaining a redundant via insertion, the guiding template assignment is considered. There exists an apparent issue for this separate manner. That is, if the via distribution is locally very dense, assigning vias to regular shaped DSA guiding templates is very difficult without violating design rules. Hence,consideration of the two stages simultaneously is necessary [38, 39, 70]. In this chapter, we focus on redundant via insertion and guiding template assignment for DSA-MP problem, considering three scenarios: single patterning (SP), double patterning (DP), and triple patterning (TP).

Our main contributions are summarized as follows.

- Under single patterning, we construct a new conflict graph by introducing *multiplet*s, and formulate the redundant via insertion and guiding template assignment for DSA with single patterning problem as a constrained maximum weight independent set (CMWIS) problem on the conflict graph. Under the assumption that a redundant via cannot be inserted if its related via is not manufacturable, we prove that the CMWIS problem is equivalent to the initial problem. The ILP formulation of CMWIS is based on *multiplet*s

instead of all possible guiding templates, whose variables and constraints are much less than those of the existing ILPs.

- We reduce the CMWIS problem to the maximum weight independent set problem such that it can be tackled by a fast algorithm, which can obtain a local optimal solution. For improving the solution quality, we propose a greedy method to obtain an initial solution for the fast algorithm.

- Under double/triple patterning, we propose a new solution flow, which is a two-stage method. At the first stage, a contraction graph is constructed, and the contracted vertices are assigned to 2 or 3 masks. At the second stage, the solver for the single patterning is called to achieve redundant via insertion and guiding template assignment for every mask.

- Experimental results show that our algorithm for the problem with single patterning is faster than the methods in [39, 70], and our two-stage method for the problem with double/triple patterning is much faster than the method in [70]. Moreover, the obtained results are better than those of the compared methods.

5.2 Preliminaries

In this section, we first describe the problem formulation handled in this chapter, then we summarize our solution flow.

5.2.1 Problem Formulation

In this chapter, we consider the problem on the grid graph. For a via v_i, the position of a redundant via candidate (RVC) of v_i should satisfy one of the following conditions: i) no via and metal wire occupy the position; ii) only one metal wire occupies the position, and the metal wire is on the same net as via v_i. We only need insert one redundant via for a via. For high resolution and focal

depth of guiding templates, the spacing between any two neighboring guiding templates should not be less than the optical resolution limit spacing d_s.

The problem aims at finding redundant vias for vias, and manufacturing all vias and their redundant vias by the DSA-MP technique. The manufacture rate (MR) and the insertion rate (IR) [39, 70] are introduced to evaluate the effectiveness of previous methods, whose definitions are described as follows.

Definition 5.2.1 (manufacture rate)**.** The manufacture rate is the ratio of the number of manufacturable vias to the number of vias.

Definition 5.2.2 (insertion rate)**.** The insertion rate is the ratio of the number of inserted redundant vias to the number of vias.

Since via manufacturability is generally the first consideration for yield, an inserted redundant via should not cause generation of an infeasible via pattern [39]. Hence, we make the following assumption:

Assumption 5.2.3. A redundant via cannot be inserted, if its related via is not manufacturable.

Under Assumption 5.2.3, the redundant via insertion and guiding template assignment for DSA with multiple patterning (RGDM) problem is formulated as follows.

Problem 5.2.4. (RGDM). Given a post-routing layout with via/redundant via layers, the usable guiding templates, and M masks, insert redundant vias for vias, assign vias and inserted redundant vias to guiding templates, and assign these guiding templates to M masks, such that: i) the inserted redundant vias are legal; ii) the spacing between any two neighboring guiding templates should not be less than the optical resolution limit spacing d_s. The objective is maximizing MR + β·IR, here β is a weighting parameter.

Depending on the number of masks, i.e., the value of M, the RGDM problem has different versions, such as single patterning (RGDS), double patterning

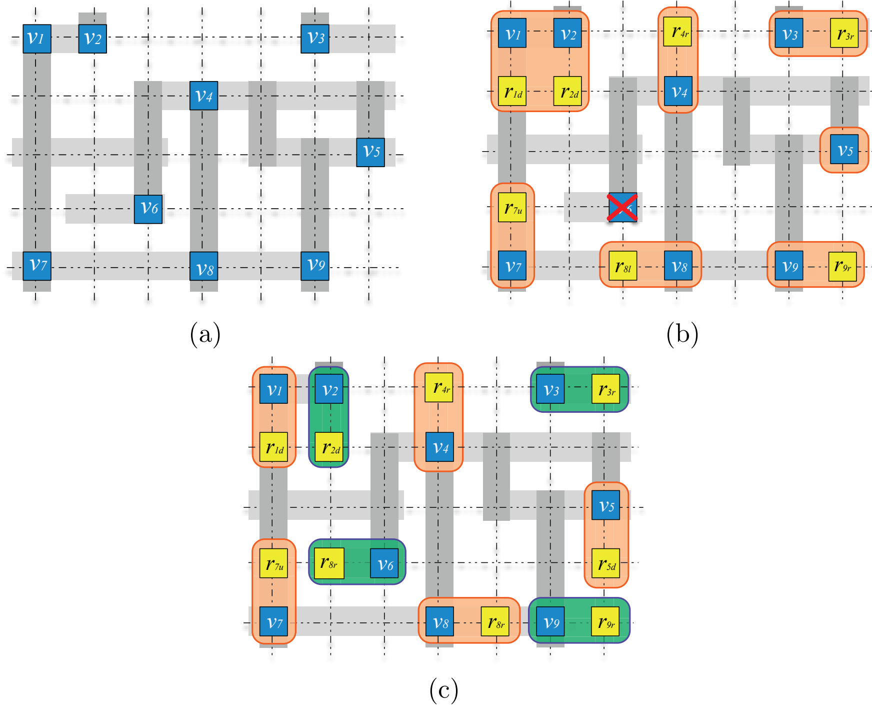


Figure 5.3: Example for the RGDS and the RGDM problems. (a) A layout with a via layer and two metal layers. (b) A result of the RGDS problem. (c) A result of the RGDD problem.

(RGDD), and triple patterning (RGDT) problems. Figure 5.3 gives an example of the RGDS and the RGDM problems. It must be remarked that all the three problems are NP-hard, since they contain the maximum independent set problem as a special case [57], which is also NP-hard.

5.2.2 Solution Flow

Our solution flow for the RGDM problem is depicted in Figure 5.4. There are three parts in the flow, i.e., preprocessing, the RGDS solver, and the RGDD/RGDT solver.

In the preprocessing, firstly, we find all redundant via candidates for every via. Based on these vias and redundant via candidates, we construct some *multiplets*. Then we consider every *multiplet* as a vertex, and construct a conflict graph $CG(V, E)$ on the grid model. Detailed definitions are described in Section 5.3. Based on the conflict graph, we propose a fast assignment algorithm

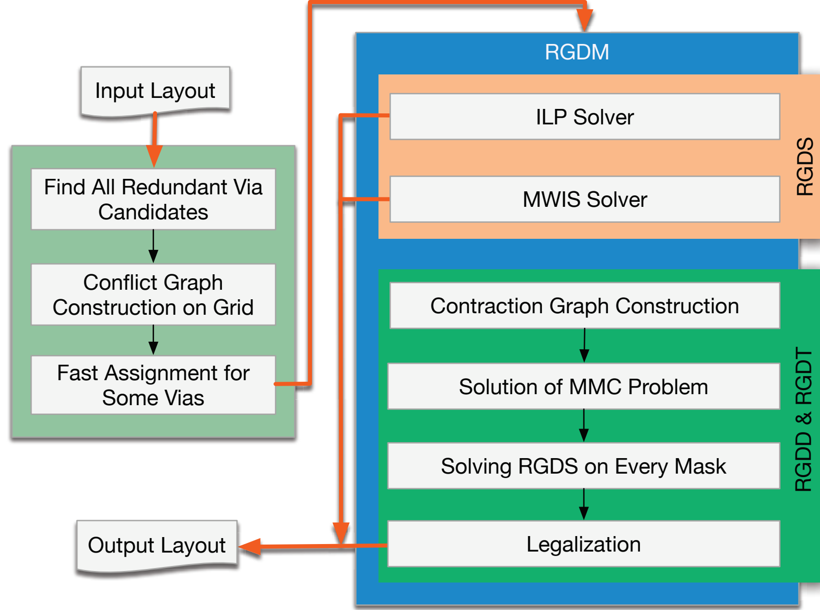


Figure 5.4: Our flow of graph based method.

for fast obtaining an optimal assignment of some vias to some guiding templates. After that, we obtain a number of connected components of the conflict graph.

For the RGDS problem on every connected component, we formulate it as a constrained maximum weight independent set problem which is an ILP. We have two approaches to solve the problem: i) use an ILP solver to obtain an optimal solution; ii) use an MWIS solver to obtain a good-enough solution, which is a very fast algorithm.

For the RGDD/RGDT problems, we propose a two-stage method. At the first stage, we construct a contraction graph $CoG(C, E^c, W^c)$, and the max-M-cut problems are formulated for obtaining the mask assignment results. At the second stage, we call our RGDS solver to obtain a redundant via insertion and guiding template assignment for every mask.

5.3 Conflict Graph Construction on Grid

In this section, firstly, we find all redundant via candidates (RVC) for every via. Then we introduce some *multiplets* based on these RVCs. At last, we

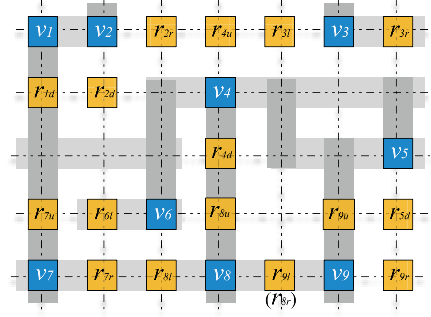


Figure 5.5: All redundant via candidates of the layout in Figure 5.3(a).

construct a conflict graph, in which vertices are *multiplets*.

According to the introduction of RVC in Section 5.2.1, we can easily find all RVCs in time $\mathcal{O}(n)$, where n is the number of vias. All RVCs of the layout in Figure 5.3(a) are presented in Figure 5.5. For example, r_{1d} in Figure 5.5 is the RVC of via v_1, here the index "*d*" denotes that r_{1d} is under v_1. Moreover, we use the indices "*u*", "*l*" and "*r*" to denote that an RVC is on the top, left and right of a via, respectively.

According to Assumption 5.2.3 in Section 5.2.1, we make following assumption as in [70]:

Assumption 5.3.1. A via and its redundant via (if existing) should be assigned to the same guiding template.

After finding all RVCs for every via, we construct a conflict graph for the via layer layout. Suppose that the RVCs of via v_i are r_{iu}, r_{id}, r_{il} and r_{ir}. We group every RVC with v_i, and denote it as *doublet*. Then we have four *doublet*s: $\{v_i, r_{iu}\}$, $\{v_i, r_{id}\}$, $\{v_i, r_{il}\}$ and $\{v_i, r_{ir}\}$. Via v_i is called a *single*. The four *doublet*s and the *single* $\{v_i\}$ denote the five possible cases of inserting a redundant via for v_i. If *doublet* $\{v_i, r_{iu}\}$ is chosen, then r_{iu} is inserted on the upper side of v_i; if *single* $\{v_i\}$ is chosen, then v_i does not have any redundant via insertion. The RVCs, the *doublet*s and the *single* of v_2 in Figure 5.5 are shown in Figure 5.6(a). In addition, we introduce another type of *doublet*, which is composed of two close and aligned vias. As shown in Figure 5.6(b), the *doublet* is composed

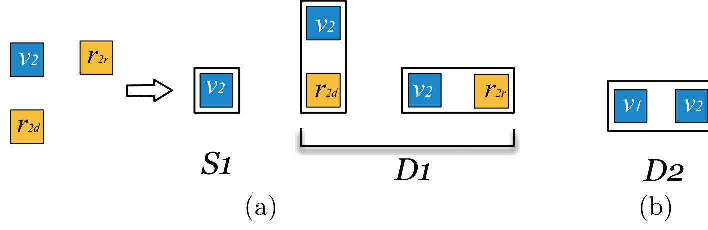


Figure 5.6: (a) The S_1 and D_1s of via v_4. (b) D_2.

of vias v_1 and v_2. Let D_1 denote the *doublet* composed of a via and a redundant via, let D_2 denote the *doublet* composed of two vias, and let S_1 denote the *single*. Every one of these *doublets* and *single* is called collectively a *multiplet*.

We regard the above every *multiplet* as a vertex in the conflict graph. Based on these vertices, we introduce some edges between them.

Definition 5.3.2 (overlap edge)**.** If *multiplets* i and j are overlapped with each other, then there exists an overlap edge e_{ij} between them. Let E_O be the set of overlap edges.

Definition 5.3.3 (conflict edge)**.** If the distance between two *multiplets* i and j is within the optical resolution limit spacing, and there does not exist an overlap edge between them, then there exists a conflict edge e_{ij} between i and j. Let E_C be the set of conflict edges.

According to Assumptions 5.2.3 and 5.3.1, we have some observations: i) a guiding template may include vias only, but it may not include only redundant vias; ii) the number of redundant vias in a guiding template must not be larger than the number of vias in the guiding template; iii) for every redundant via in a guiding template, its via must also be in the same guiding template. Then the usable guiding template types in Figure 5.2 are of the following combinations: A t_1 includes a via, i.e., an S_1; A t_2 includes a via and a redundant via, i.e., a D_1, or includes two vias, i.e., a D_2; A t_3 includes two vias and a redundant via, i.e., a D_1 and an S_1, or includes three vias, i.e., a D_2 and an S_1; A t_4 includes two vias and two redundant vias, i.e., two D_2s, or includes three vias and a redundant

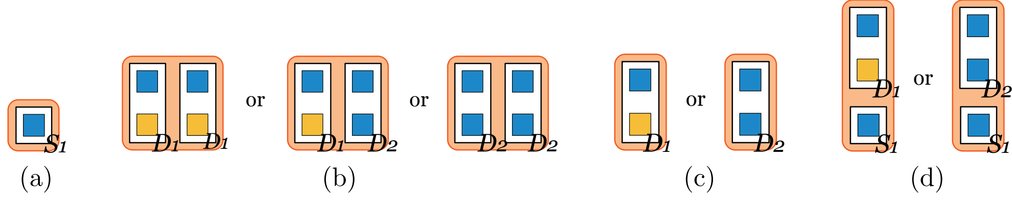


Figure 5.7: All possible combinations of *multiplet*s to form four guiding template types.

via, i.e., a D_1 and a D_2, or it includes four vias, i.e., two D_2s. All possible combinations of *multiplet*s are shown in Figure 5.7. It must be remarked that, although we only consider the above four guiding template types, combinations of *multiplet*s to form other guiding template shapes still can be figured out.

Definition 5.3.4 (template edge)**.** For two *multiplet*s i and j, suppose that at least one of them is not S_1. If i and j can be assigned to the same guiding template without any design rule violation, and between i and j there exists a conflict edge $e_{ij} \in E_C$, then e_{ij} is also called a template edge. Let E_T be the set of template edges. Obviously, $E_T \subseteq E_C$.

Definition 5.3.5 (conflict graph)**.** The conflict graph is an undirected graph $CG(V, E)$, where vertex $v \in V$ denotes a *multiplet*, $e_{ij} \in E$ is an edge and $E = (E_C - E_T) \cup E_O$. E_C, E_T and E_O are the sets of conflict edges, template edges and overlap edges, respectively.

The conflict graph of vias v_1, v_2, v_3 and v_4 of the layout in Figure 5.3(a) is constructed as shown in Figure 5.8 (for clearness, we omit the conflict graph of other vias in Figure 5.3(a)). In Figure 5.8, nodes a, c, g and l are S_1, and nodes b, d, e, h, i, j and k are D_1 and node f is D_2. The red lines are the conflict edges, the black edges are the overlap edges, and the green dotted edges are the template edges. The *multiplet*s corresponding to vertices a, b, c, d and f are shown in the box. In Figure 5.8, vertices a and b are overlapped with each other at v_1, thus there is an overlap edge between them. The distance between vertices b and c is within the optical resolution limit spacing (one grid), thus a

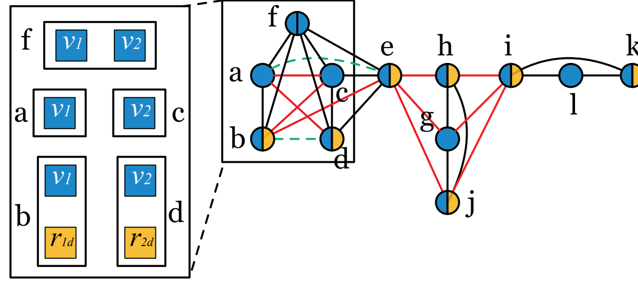


Figure 5.8: The conflict graph CG for vias v_1, v_2, v_3 and v_4 of the layout in Figure 5.3(a).

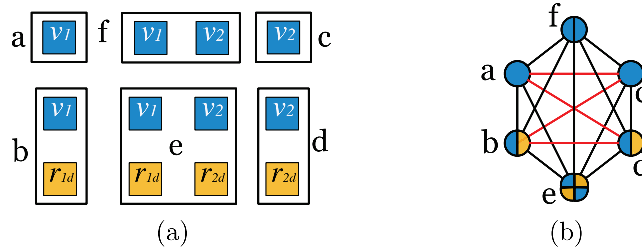


Figure 5.9: The graph constructed based on GTA for vias v_1 and v_2 of the layout in Figure 5.3(a).

conflict edge is generated between them. Since vertices b and d can be assigned to a quadruple guiding template as in Figure 5.3(b), there is a template edge between b and d.

It must be remarked that introducing *multiplets* may reduce the size of a conflict graph. In the previous works [39, 70], the constructed graphs are based on all the possible guiding template assignments (GTA) instead of *multiplets*. In a layout, the number of GTAs is greater than the number of *multiplets*, so do the sizes of the corresponding graphs. For example, all the possible GTAs of vias v_1 and v_2 in Figure 5.3(a) are shown in Figure 5.9(a), while the graph based on these GTAs is shown in Figure 5.9(b). Comparing with the partial conflict graph in the box of Figure 5.8, the graph in Figure 5.9(b) is more dense.

Before solving the RGDM problem, we introduce a fast guiding template assignment and redundant via insertion for some vias. On the conflict graph $CG(V, E)$, if a vertex is D_1, and there is no conflict edge incident to it, then the via and the redundant via in the D_1 are assigned to a guiding template, and

the adjacent vertices of the vertex connected by overlap edges are removed from the conflict graph. After that, we calculate the connected components, and the RGDM problem is considered on the connected components one by one.

5.4 Algorithms for RGDS Problem

In this section, we describe the technical details of solving the guiding template assignment and redundant via insertion for DSA with single patterning (RGDS) problem. First we formulate the RGDS problem as a constrained maximum weight independent set (CMWIS) problem, and give the corresponding ILP formulation. Then we prove the equivalence between the RGDS problem and the CMWIS problem, through which the RGDS problem can be solved by ILP solver. In addition, for fast solving the CMWIS problem, we introduce a fast algorithm for the maximum weight independent set problem, and reduce the CMWIS problem to the problem.

5.4.1 Constrained Maximum Weight Independent Set Problem

According to the conflict graph of the RGDS problem, we construct the CMWIS problem. For the *multiplet*s overlapped with each other, we can only choose one of them for patterning. Thus, in the conflict graph, if between two *multiplet*s i and j there is an $e_{ij} \in E_O$, then only one of the *multiplet*s can be chosen. Furthermore, for the RGDS problem, only a mask can be used for guiding templates. Thus, if two *multiplet*s are within the optical resolution limit spacing, then only one of them can be patterned, unless the two *multiplet*s are assigned to the same guiding template. That is, if between the *multiplet*s i and j there exists $e_{ij} \in E_C$ but $e_{ij} \notin E_T$, then only one of them can be chosen. Hence, the RGDS problem is similar to the independent set problem.

In addition, the vertex set V of the conflict graph CG is composed of *multiplet*s. A D_1 includes a via and a redundant via, a D_2 includes two vias, and an S_1 only includes a via. The objective of the RGDS problem is maximizing the

weighted sum of MR and IR, i.e., maximizing the weighted sum of the number of manufacturable vias and the number of inserted redundant vias. We jointly consider MR and IR by assigning weight to every vertex as

$$w_i = \begin{cases} 1+\beta, & \text{if } i \text{ is a } D_1; \\ 2, & \text{if } i \text{ is a } D_2; \\ 1, & \text{if } i \text{ is an } S_1; \end{cases} \tag{5.1}$$

where w_i is the weight of vertex i, β is a parameter which is used to balance MR and IR, and is set as 0.5 in this chapter. Let W be the set of weights, then the conflict graph $CG(V, E)$ is weighted and is written as $CG(V, E, W)$.

Then, we formulate the RGDS problem as the maximum weight independent set (MWIS) problem

$$\max_{x} \quad \sum_{i \in V} w_i x_i \tag{5.2}$$

$$\text{s.t.} \quad x_i + x_j \leq 1, \quad \forall e_{ij} \in E; \tag{5.2a}$$

$$x_i \in \{0, 1\}, \quad \forall i \in V. \tag{5.2b}$$

In the problem, Constraint (5.2a) means that, if between vertices i and j there exists $e_{ij} \in E_O$ or $e_{ij} \in E_C - E_T$, then at most one of them can be patterned, i.e., $x_i = 1$, $x_j = 0$, or $x_i = 0$, $x_j = 1$, or $x_i = 0$, $x_j = 0$.

It must be remarked that the MWIS problem is not equivalent to the RGDS problem. Some special structures in a layout make the equivalence not hold. These structures have a feature that, every two of several close *multiplet*s can be assigned to a same guiding template, but all of these *multiplet*s cannot be included in a same guiding template. We present two examples in Figure 5.10. In Figure 5.10(a), i, j and k are three *multiplet*s which are D_1, e_{ij} and e_{jk} are template edges, and there is no edge between i and k. For this structure, i and

j can be assigned to a t_4 guiding template, and so does j and k, but all of i, j and k cannot be assigned to a same guiding template. However, by solving the MWIS problem on the structure in Figure 5.10(a), we get an optimal solution $x_i = 1$, $x_j = 1$ and $x_k = 1$, which means i, j and k can be patterned simultaneously. However, one of the three *multiplet*s cannot be patterned since this is the RGDS problem and all of the three *multiplet*s cannot be assigned to the same guiding template. For the structure in Figure 5.10(b), this situation still holds.

We call the structures in Figure 5.10 as incompatibility structure (INC), which can be defined as follows.

Definition 5.4.1 (INC)**.** For the RGDS problem, the incompatibility structure is a structure composed of three *multiplet*s i, j and k, in which e_{ij} and e_{jk} are template edges and there does not exist any edge between i and k.

It must be noted that, the definition of INC only includes three *multiplet*s, since any one of the four usable guiding templates includes at most two *multiplet*s, as shown in Figure 5.7. If there exists other shape of guiding template, we still can define a corresponding INC.

In order to exclude the solutions of the MWIS problem which are infeasible for the RGDS problem due to INC, we add the following constraint to the MWIS problem:

$$x_i + x_j + x_k \leq 2, \qquad \text{if } i,\ j \text{ and } k \text{ compose an INC.} \tag{5.3}$$

Then we call the new problem as a constrained maximum weight independent set (CMWIS) problem.

Now we show the relationship between the CMWIS problem and the RGDS problem. Before that, we show the relationship between Assumptions 5.2.3 and 5.3.1 for the RGDS problem.

Lemma 5.4.2. Under Assumption 5.2.3, Assumption 5.3.1 holds.

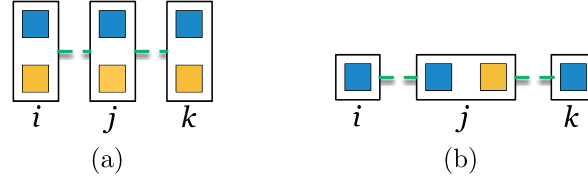


Figure 5.10: Two kinds of incompatibility structures.

Proof. Under Assumption 5.2.3, suppose Assumption 5.3.1 does not hold. Then an inserted redundant via is not in the same guiding template as its via. Obviously, there exists a conflict edge between the via and its redundant via due to the spacing rule. However, for the RGDS problem, only a mask can be used to pattern guiding templates. Thus the via cannot be patterned, which contradicts Assumption 5.2.3. Hence Assumption 5.3.1 holds. □

Theorem 5.4.3. The CMWIS problem is equivalent to the RGDS problem.

Correctness of Theorem 5.4.3 is explained as follows. In the CMWIS problem, all possible redundant via candidates and guiding template assignments (under Assumption 5.2.3) are considered. After obtaining a solution of the CMWIS problem, we only need to assign the vertices with $x_i = 1$ and connected by template edges to a guiding template, and then we obtain a solution of the RGDS problem. Since forced by Constraints (5.2a) and (5.3), the obtained guiding template assignments must be legal. Furthermore, the objective of the RGDS problem is maximizing MR + β·IR, which is equivalent to maximizing $N_M + \beta \cdot N_I$, here N_M is the number of patterned vias, N_I is the number of inserted redundant vias. According to the definition of *multiplet*s and the weighting rule of *multiplet*, we have $\sum_{i \in V} w_i x_i = N_M + \beta \cdot N_I$.

Comparing with the via layer layout, the conflict graph of the CMWIS problem has much more vertices, and has too many edges introduced for constraints. Hence solving the CMWIS problem on a large dense graph is time consuming. However, after fast assignment of some vias to some guiding templates, the conflict graph is divided into a number of connected components.

Thus, it is possible to solve the CMWIS problem by ILP solver.

5.4.2 A Fast Algorithm for the MWIS Problem

To solve the CMWIS problem faster, we introduce a fast algorithm for the maximum weight independent set (MWIS) problem. Then, we reduce the CMWIS problem to the MWIS problem in the next subsection. Many fast algorithms have been proposed to fast obtain good-enough solutions [15, 58] of the MWIS problem. Among them, we introduce the fast algorithm in [15] and improve it for our usage.

The MWIS problem can be reformulated as an integer quadratic program. Detailed derivation is as follows. The ILP formulation (5.2) of the MWIS problem is equivalent to

$$\max_{\mathbf{x}} \quad \mathbf{w}^\intercal \mathbf{x} \tag{5.4}$$

$$\text{s.t.} \quad x_i x_j = 0, \quad \forall e_{ij} \in E; \tag{5.4a}$$

$$x_i \in \{0, 1\}, \quad \forall i \in V, \tag{5.4b}$$

where $\mathbf{w} = (w_1, w_2, \cdots, w_n)^\intercal \in \mathbb{R}^n$, $\mathbf{x} = (x_1, x_2, \cdots, x_n)^\intercal \in \{0, 1\}^n$, $n = |V|$. Let $\mathbf{A} = (A_{ij})$ be the adjacency matrix with $A_{ij} = 1$ if $e_{ij} \in E$ and $A_{ij} = 0$ if $e_{ij} \notin E$. Then the IP formulation (5.4) can be rewritten as

$$\max_{\mathbf{x}} \quad \mathbf{w}^\intercal \mathbf{x} \tag{5.5}$$

$$\text{s.t.} \quad \mathbf{x}^\intercal \mathbf{A} \mathbf{x} = \mathbf{0}; \tag{5.5a}$$

$$x_i \in \{0, 1\}, \quad \forall i \in V. \tag{5.5b}$$

Putting Constraint (5.5a) into the objective function, we get

$$\max_{\mathbf{x}} \quad f(\mathbf{x}) = \mathbf{w}^{\intercal}\mathbf{x} - \frac{1}{2}\alpha\mathbf{x}^{\intercal}\mathbf{A}\mathbf{x} \tag{5.6}$$

$$\text{s.t.} \quad x_i \in \{0,1\}, \qquad \forall i \in V, \tag{5.6a}$$

where $\alpha > 0$ is a regularization parameter. Since the adjacency matrix $\mathbf{A}$ may not be positive semi-definite, $f(\mathbf{x})$ may not be a concave function.

Let $\hat{\mathbf{x}}$, $\mathbf{x}^* \in \{0,1\}^n$ be a candidate solution and a solution of Problem (5.6), respectively, let $\mathbf{y} \in [0,1]^n$ be a point in the continuous domain. Based on Problem (5.6), the Algorithm 2 in [15] works as follows. It visits a sequence of continuous points $\{\mathbf{y}^{(t)}\}$ in iterations $t = 0, 1, 2, \cdots$, where $\mathbf{y}^{(t)} \in [0,1]^n$, and finds discrete candidate solutions $\hat{\mathbf{x}}^{(t)}$ in the respective neighborhoods of $\mathbf{y}^{(t)}$, until convergence. Let $h(\mathbf{y}, \mathbf{y}^{(t)})$ be the first-order Taylor series approximation of $f(\mathbf{y})$ at $\mathbf{y}^{(t)}$, i.e.,

$$\begin{aligned} h(\mathbf{y}, \mathbf{y}^{(t)}) &= f(\mathbf{y}^{(t)}) + (\mathbf{y} - \mathbf{y}^{(t)})^{\intercal}(\mathbf{w} - \alpha\mathbf{A}\mathbf{y}^{(t)}) \\ &= \mathbf{y}^{\intercal}(\mathbf{w} - \alpha\mathbf{A}\mathbf{y}^{(t)}) + \text{constant}, \end{aligned} \tag{5.7}$$

where ‘constant’ does not depend on $\mathbf{y}$. The approximation $h(\mathbf{y}, \mathbf{y}^{(t)})$ is linear in $\mathbf{y}$. Hence, if α is large enough, we can easily obtain a discrete maximizer $\hat{\mathbf{x}}^{(t)}$ $= \text{argmax}_{\mathbf{y}\in\{0,1\}^n} h(\mathbf{y}, \mathbf{y}^{(t)})$ by assigning

$$\hat{x}_i^{(t)} = \begin{cases} 1, & \text{if } (\mathbf{w} - \alpha\mathbf{A}\mathbf{y}^{(t)})_i > 0; \\ 0, & \text{otherwise.} \end{cases} \tag{5.8}$$

Then $\mathbf{y}^{(t)}$ is mainly updated by a linear interpolation of $\mathbf{y}^{(t)}$ and $\hat{\mathbf{x}}^{(t)}$, i.e., $\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} + \eta(\hat{\mathbf{x}}^{(t)} - \mathbf{y}^{(t)})$, where η is an interpolation parameter with $\eta \in [0,1]$ and satisfying $df(\mathbf{y}^{(t+1)})/d\eta \geq 0$. This ensures that the sequence $\{f(\mathbf{y}^{(t)})\}$ is non-deceasing. Brendel *et al.* [15] provided a closed-form solution of η as

$$\eta = \min\left\{\max\left\{\frac{(\mathbf{w}-\alpha\mathbf{A}\mathbf{y}^{(t)})^\intercal(\hat{\mathbf{x}}^{(t)}-\mathbf{y}^{(t)})}{\alpha(\hat{\mathbf{x}}^{(t)}-\mathbf{y}^{(t)})^\intercal\mathbf{A}(\hat{\mathbf{x}}^{(t)}-\mathbf{y}^{(t)})},0\right\},1\right\}, \tag{5.9}$$

and proved that the sequence $\{\mathbf{y}^{(t)}\}$ converges to a local optimal solution of Problem (5.2).

However, from Eq. (5.8), we can see that the solution quality of $\hat{\mathbf{x}}^{(t)}$ is highly dependent on $\mathbf{y}^{(0)}$. If we choose a bad $\mathbf{y}^{(0)}$, then the obtained local optimal value $f(\mathbf{x}^*)$ may be far away from the global optimal value. Hence, in order to obtain a better solution, we must have a good initial solution $\hat{\mathbf{x}}$ for this algorithm.

Algorithm 5.1 Initial solution generation

Input: A connected component of $CG(V, E, W)$;
Output: Initial solution $\hat{\mathbf{x}}^{(0)}$ of the MWIS problem;
1: **repeat**
2: Compute $w_s(k)$ by Equation (5.10), $\forall k \in V$;
3: $\hat{x}_i^{(0)} \leftarrow 1$, where $i = \text{argmin}_{k\in V} w_s(k)$;
4: **for** every j in V with $e_{ji} \in E$ **do**
5: $\hat{x}_j^{(0)} \leftarrow 0$, and $V \leftarrow V - \{j\}$;
6: **end for**
7: $V \leftarrow V - \{i\}$;
8: **until** $V = \emptyset$

We propose a greedy based algorithm to obtain a good enough initial solution $\hat{\mathbf{x}}^{(0)}$ in Algorithm 5.1. In line 2 of the algorithm, the selection weight $w_s(k)$ is computed by

$$w_s(k) = \begin{cases} d_c(k) - d_t(k), & \text{if } k \text{ is } D_1; \\ N + d_c(k) - d_t(k), & \text{otherwise,} \end{cases} \tag{5.10}$$

where $d_c(k)$ is the number of conflict edges incident to vertex k, $d_t(k)$ is the number of template edges incident to vertex k. The selection weight $w_s(k)$ is used to evaluate the degree of conflict of vertex k. N is a number larger than all d_c (N is set to 50). This setting indicates that we prior select D_1s

to other *multiplets*. Since the maximal degree of vertices is a constant for the conflict graph, the runtime complexity of Algorithm 5.1 is $\mathcal{O}(|V|)$. For the MWIS problem, we improve the algorithm in [15] as outlined in Algorithm 5.2.

Algorithm 5.2 The MWIS algorithm

Input: A connected component of $CG(V, E, W)$, convergence threshold δ, and $\alpha \leftarrow 2$.

Output: Solution $\mathbf{x}^*$ of the MWIS problem.

1: Initialize $t \leftarrow 0$, and $\mathbf{y}^{(0)} \in [0,1]^n$, $\mathbf{y}^{(0)} \neq \mathbf{0}$;
2: Generate $\hat{\mathbf{x}}^{(0)}$ by Algorithm 5.1, and $\mathbf{x}^* \leftarrow \hat{\mathbf{x}}^{(0)}$;
3: **repeat**
4:   **if** $f(\hat{\mathbf{x}}^{(t)}) \geq f(\mathbf{y}^{(t)})$ **then**
5:     $\mathbf{y}^{(t+1)} \leftarrow \hat{\mathbf{x}}^{(t)}$;
6:   **else**
7:     Obtain η by Equation (5.9);
8:     $\mathbf{y}^{(t+1)} \leftarrow \mathbf{y}^{(t)} + \eta(\hat{\mathbf{x}}^{(t)} - \mathbf{y}^{(t)})$;
9:   **end if**
10:   Obtain $\hat{\mathbf{x}}^{(t+1)}$ by Equation (5.8);
11:   **if** $f(\hat{\mathbf{x}}^{(t+1)}) \geq f(\mathbf{x}^*)$ **then**
12:     $\mathbf{x}^* \leftarrow \hat{\mathbf{x}}^{(t+1)}$;
13:   **end if**
14:   $t \leftarrow t + 1$;
15: **until** $||\mathbf{y}^{(t+1)} - \mathbf{y}^{(t)}|| < \delta$;
16: $V^1 \leftarrow \{i : x_i^* = 1, i = 1, 2, \cdots, n\}$;
17: **repeat**
18:   $x_{i_0}^* \leftarrow 1$, where $i_0 = \mathrm{argmax}_{i \in V^1} w_i$;
19:   **for** every j in V^1 with $e_{ji_0} \in E$ **do**
20:     $x_j^* \leftarrow 0$, and $V^1 \leftarrow V^1 - \{j\}$;
21:   **end for**
22:   $V^1 \leftarrow V^1 - \{i_0\}$;
23: **until** $V^1 = \emptyset$

Lines 16-23 in Algorithm 5.2 are used to obtain the final feasible solution $\mathbf{x}^*$ of the MWIS problem. Algorithm 5.2 is very fast. The iteration between lines 3-15 converges in only a few number of iterations, and the complexity per iteration is only $\mathcal{O}(|E|)$ [15]. Moreover, according to our experiments, it even can obtain optimal solutions of the RGDS problem on many connected components. Especially, Algorithm 5.1 is also very effective, since it may also obtain optimal solution of the RGDS problem on many connected components. Figure 5.11

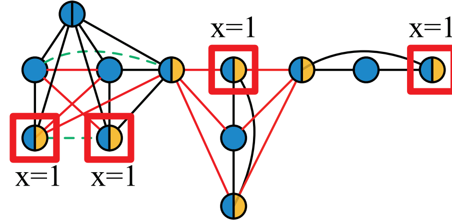


Figure 5.11: An optimal solution of the MWIS problem on the conflict graph in Figure 5.8.

shows an optimal solution of the RGDS problem on the conflict graph shown in Figure 5.8, which is obtained by Algorithm 5.1. The corresponding result is shown in Figure 5.3(b).

5.4.3 Reduction of CMWIS to MWIS

According to the analyses in Section 5.4.1, the CMWIS problem is composed of the MWIS problem and Constraint (5.3). According to our statistics, most of the connected components obtained after the fast assignment for some vias stage do not have incompatibility structure. Hence Constraint (5.3) is not active on these connected components, and we only need to solve the MWIS problem using Algorithm 5.2 instead of solving the CMWIS problem. However, for the connected components with incompatibility structure, Algorithm 5.2 cannot be directly used. On these connected components, we reduce the CMWIS problem to the MWIS problem by restricting Constraint (5.3) to Constraint (5.2a), which is by deleting a template edge in every INC from the conflict graph CG. If a template edge e_{ij} of an INC is deleted, then it appears in E as a conflict edge due to $E = E_C - E_T$, and i and j should satisfy Constraint (5.2a). Consequently, combining Constraint (5.2a) and the other vertex in the INC implies that Constraint (5.3) holds. Naturally, we use the following two template edge deletion criteria: first, we delete template edges as few as possible; second, we prior delete those template edges which connect two *multiplet*s with low-probability being assigned to the same guiding template. The incompatibility structure eliminating algorithm is outlined in Algorithm 5.3.

Algorithm 5.3 Incompatibility structure elimination

Input: A connected component (CC_CG) of CG with INC;
Output: A CC_CG of CG without INC;
1: Construct the incompatibility graph IG of CC_CG;
2: Find all connected components (CC_IGs) of IG;
3: **for** every CC_IG **do**
4: Extract all *multiplet*s in CC_IG;
5: Construct the template graph TG;
6: Solve maximum weight matching on TG;
7: Mark all un-matched template edges, and delete them from CC_CG;
8: **end for**

Definition 5.4.4 (incompatibility graph, IG)**.** The incompatibility graph $IG(I, E_I)$ is an undirected graph , where node $I_i \in I$ is an INC. An incompatibility edge $e_{ij} \in E_I$ exists between INCs I_i and I_j if they share a template edge in the conflict graph CG.

Definition 5.4.5 (template graph, TG)**.** The template graph $TG(T, E_T, W_T)$ is an induced subgraph of conflict graph CG by template edges, where node $T_i \in T$ is a *multiplet* in an INC. An edge $e_{ij} \in E_T$ exists between nodes T_i and T_j if $e_{ij} \in E_T$ in CG. $w^t_{ij} \in W^T$ is the weight of template edge e_{ij}, which is set as Equation (5.11).

For line 1 of Algorithm 5.3, the detailed definition of incompatibility graph IG is described in Definition 5.4.4. An example for a connected component CC_IG of IG is shown in Figure 5.12(b), which is composed of two INCs $I_i(i,\ j,\ k)$ and $I_j(j,\ k,\ l)$, where I_i is composed of *multiplet*s i, j and k, and I_j is composed of *multiplet*s j, k and l. Template edge e^t_{jk} is in both of I_i and I_j. Thus there is an incompatibility edge between I_i and I_j, and I_i and I_j are in the same connected component of IG. In line 5 of Algorithm 5.3, template graph TG is defined as in Definition 5.4.5, and the template graph of Figure 5.12(b) is shown in Figure 5.12(c). It must be noted that, in a TG the degree of every *multiplet* is 1 or 2 (at least two *multiplet*s are with degree 1). The detailed explanation is stated as follows. The three *multiplet*s in every INC should be

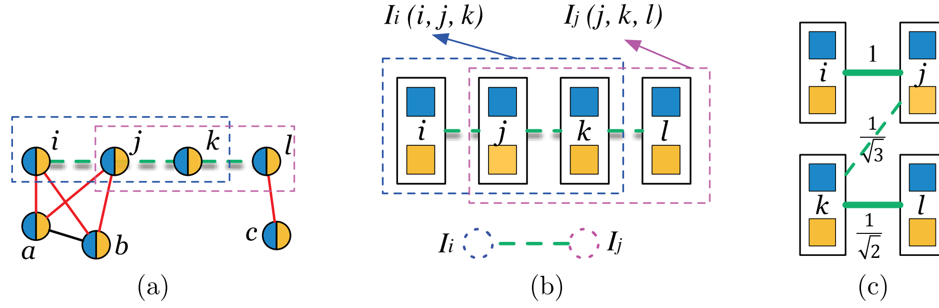


Figure 5.12: (a) A CG with seven $multiplet$s. (b) An IG with two INCs $I_i(i,\ j,\ k)$ and $I_j(j,\ k,\ l)$. (c) A TG and its maximum weight matching result.

in a line, and any two connected INCs in a connected component of IG should share a template edge. As a result, all *multiplet*s in the TG constructed by line 5 in Algorithm 5.3 are in a line. Thus the TG is a bipartite graph.

In line 5 of Algorithm 5.3, the weight of a template edge e^t_{ij} is set as

$$w^t_{ij} = \frac{1}{\sqrt{n^t_{i-j} + n^t_{j-i}}}, \tag{5.11}$$

where n^t_{i-j} is the number of *multiplet*s which satisfies that: i) these *multiplet*s are connected to i but not to j by conflict edges in the conflict graph CG; ii) if several *multiplet*s satisfy i), but they are connected to each other by overlap edges in the conflict graph CG, then these *multiplet*s are counted as one *multiplet*. n^t_{j-i} has the similar meaning, and at least one of n^t_{i-j} and n^t_{j-i} are not less than 1. w^t_{ij} in Equation (5.11) is used to reflect the similarity of connected *multiplet*s. If i and j share more connected *multiplet*s, then the value of w^t_{ij} is larger. And in this case, *multiplets* i and j are more likely assigned to the same guiding template. In Figure 5.12(a), *multiplets* i, a and b are connected to j but not to k by conflict edges, and *multiplets* a and b are connected to each other by an overlap edge, hence $n^t_{j-k} = 2$. Correspondingly, the weights of all template edges are calculated by (5.11) as shown in Figure 5.12(c).

In order to obtain the best template edge deletion, we obtain the maximum

weight matching on TG by solving a maximum flow problem. Generally, the runtime complexity of solving a maximum flow problem is $\mathcal{O}(|V|^3)$. Since the degrees of vertices in TG are 1 or 2 and the capacity of every edge in TG is 1, our TG is a unit capacity simple network. Thus, solving the maximum flow problem on our TG can be finished in $\mathcal{O}(|T|^{\frac{1}{2}} \cdot |E_T|)$. In addition, since the size of a TG is very small, we can fast obtain a maximum weight matching result, and further delete those un-matched template edges from the conflict graph CG. For example, for the TG in Figure 5.12(c), we can obtain a maximum weight matching (i, j) and (k, l), and then we delete the un-matched template edge (j, k) from the conflict graph. Thus the two INCs are eliminated.

5.5 Algorithms for RGDD/RGDT Problems

In this section, we consider redundant via insertion and guiding template assignment for DSA with double patterning (RGDD) or triple patterning (RGDT). The RGDD and the RGDT problems can also be formulated as ILPs, and then may be solved by ILP solver. However, the work in [70] indicates that, the ILP formulations have too many variables and constraints and are very hard to solve, especially on large and dense layouts.

We solve the RGDD and the RGDT problems in two stages. At the first stage, vias are assigned to M masks, such that the vias in every mask can be easily inserted redundant vias and patterned. At the second stage, the RGDS problem is called to assign vias and inserted redundant vias to templates for every mask.

5.5.1 Mask Assignment for RGDD/RGDT

At the first stage of our two stage assignment method, we assign the vias on via layer to M masks, such that the vias in every mask can be easily inserted redundant vias and patterned. In order to achieve the assignment, we construct a contraction graph $CoG(C, E^c, W^c)$.

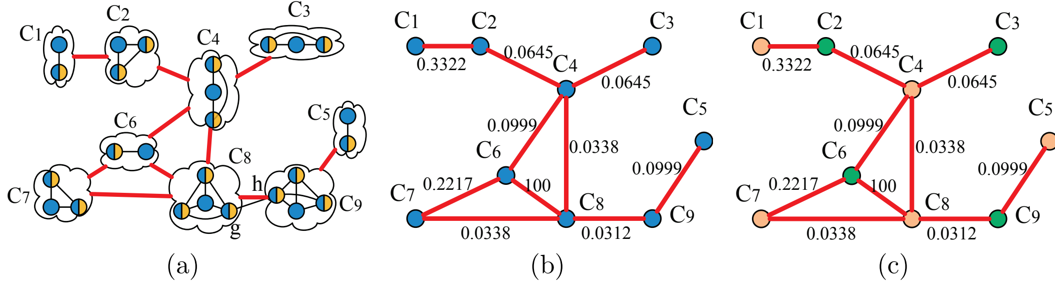


Figure 5.13: (a) Process of contraction graph construction. (b) Contraction graph of the conflict graph in Figure 5.8. (c) Result of the max-2-cut problem on the contraction graph in (b).

Definition 5.5.1 (contraction graph)**.** The contraction graph is an undirected graph $CoG(C, E^c, W^c)$, where node $C_i \in C$ is composed of all S_1 and D_1s of via i. A contraction edge $e^c_{ij} \in E^c$ exists between nodes C_i and C_j if there exists a pair of *multiplets* p and $q \in V$ of CG with $e_{pq} \in E_C$, where p is an S_1 or a D_1 in node C_i, and q is an S_1 or a D_1 in node C_j. $W^c_{ij} \in W^c$ is the weight of contraction edge e^c_{ij}.

The weight $W^c_{ij} \in W^c$ of the contraction edge e^c_{ij} indicates the degree of conflict between C_i and C_j, which is calculated by

$$W^c_{ij} = \frac{1}{\overline{W}_{ij} - \widetilde{W}_{ij} + 0.01}, \tag{5.12}$$

where

$$\overline{W}_{ij} = \sum_{p \in C_i, q \in C_j} \widetilde{w}_{pq}, \tag{5.13}$$

$$\widetilde{W}_{ij} = \sum_{\substack{p \in C_i, q \in C_j \\ e_{pq} \in E_C}} \widetilde{w}_{pq}, \tag{5.14}$$

and

$$\widetilde{w}_{pq} = \begin{cases} 2+2\beta, & \text{if both of } p \text{ and } q \text{ are } D_1; \\ 2+\beta, & \text{if one of } p \text{ and } q \text{ is } D_1; \\ 2, & \text{if both of } p \text{ and } q \text{ are } S_1. \end{cases} \tag{5.15}$$

In (5.13), $\widetilde{w}_{pq} = w_p + w_q$ is the weight sum of *multiplet*s (vertices) p and q in CG. $\overline{W}_{ij}$ is the weight sum of all pairs of *multiple*s p and q in C_i and C_j, respectively. $\widetilde{W}_{ij}$ is the weight sum of *multiplet*s $p \in C_i$ and $q \in C_j$ between which there exists a conflict edge e_{pq}. Furthermore, $\overline{W}_{ij} - \widetilde{W}_{ij}$ can be seen as the degree of freedom between C_i and C_j. The weight W_{ij}^c of the contraction edge e_{ij}^c is calculated by Eq. (5.12), which indicates the degree of conflict between C_i and C_j, where 0.01 in the denominator is used to avoid the denominator being 0.

The contraction graph CoG of the layout in Figure 5.3(a) is constructed in Figure 5.13(b) based on the conflict graph in Figure 5.8, where the bold red lines are contraction edges. Detailed process of contraction is illustrated in Figure 5.13(a), where every cloud is a node. When β is set as 0.5, the weights are calculated as in Figure 5.13(b). In Figure 5.13(b), $W_{48}^c = 0.0338$ is the degree of conflict between nodes C_4 and C_8, which is smaller than the other edge weights. This shows that we still can insert easily redundant vias and pattern vias for vias v_4 and v_8, even though they are assigned to the same mask. While $W_{68}^c = 100$ is the degree of conflict between C_6 and C_8, which is much larger than the others. In fact, if nodes C_6 and C_8 are assigned to the same mask, then at least one of vias v_6 and v_8 cannot be patterned, due to conflicts.

After constructing the contraction graph $CoG(C, E^c, W^c)$, we perform the first stage for the RGDD and RGDT problems. That is, the nodes in CoG are assigned to M masks ($M = 2$ or 3), which is a max-M-cut (MMC) problem on CoG. The MMC problem is also an NP-hard problem. Fortunately, the number of nodes and the number of edges in CoG are much less than those of the conflict graph CG. Hence, solving the MMC problem on the contraction graph by ILP

solver is possible. Before that, some graph reduction techniques are used for speeding up the solution methods.

In this chapter, we utilize three graph reduction techniques, which can keep optimality of the MMC problem:

- Vertex with degree less than M removal;
- Bridge edge removal;
- Connected component calculation.

All the three techniques have been widely used to reduce the size of a graph for a series of partition problems [36, 53, 105]. Here, we skip the details.

For the RGDD and RGDT problems, the corresponding max-2-cut problem and max-3-cut problem on $CoG(C, E^c, W^c)$ are presented in ILP formulations. In the previous works [6, 40, 49], various ILP formulations for the MMC problem have been formulated. In this chapter, the max-2-cut problem for RGDD is formulated as

$$\min_{z,c} \quad \sum_{e_{ij}^c \in E^c} W_{ij}^c c_{ij} \tag{5.16}$$

$$\text{s.t.} \quad z_i + z_j \geq 1 - c_{ij}, \quad \forall e_{ij}^c \in E^c; \tag{5.16a}$$

$$z_i + z_j \leq 1 + c_{ij}, \quad \forall e_{ij}^c \in E^c; \tag{5.16b}$$

$$z_i, c_{ij} \in \{0, 1\}, \quad \forall C_i \in C. \tag{5.16c}$$

The max-3-cut problem for RGDT is formulated as

$$\begin{aligned}
\min_{z,c} \quad & \sum_{e^c_{ij} \in E^c} W^c_{ij} c_{ij} && && (5.17)\\
\text{s.t.} \quad & z_{1i} + z_{2i} \geq 1, && \forall C_i \in C; && (5.17a)\\
& z_{2i} - z_{1i} + z_{2j} - z_{1j} \leq 1 + c_{ij}, && \forall e^c_{ij} \in E^c; && (5.17b)\\
& z_{1i} - z_{2i} + z_{1j} - z_{2j} \leq 1 + c_{ij}, && \forall e^c_{ij} \in E^c; && (5.17c)\\
& z_{1i} + z_{2i} + z_{1j} + z_{2j} \leq 3 + c_{ij}, && \forall e^c_{ij} \in E^c; && (5.17d)\\
& z_{1i}, z_{2i}, c_{ij} \in \{0, 1\}, && \forall C_i \in C. && (5.17e)
\end{aligned}$$

Since the size of every connected component of the contraction graph is not large, we directly use ILP solver to solve Problems (5.16) and (5.17). A max-2-cut result of the contraction graph in Figure 5.13(b) is shown in Figure 5.13(c). The objective value is 0.0338 + 0.0338 = 0.0676.

5.5.2 Solving the RGDS Problem on Every Mask and Legalization

In this subsection, we explain the second stage for the RGDD and the RGDT problems. After assigning nodes to M masks ($M = 2$ or 3), we need to consider the RGDS problem on every mask. First, we reconstruct the conflict graph $CG_k(V_k, E_k)$ for every mask $k = 1, 2, \cdots, M$. Then we solve the RGDS problem on every $CG_k(V_k, E_k)$ using the methods in Section 5.4.2. Here, the MWIS solver is chosen as the RGDS solver.

Since our two stage method deals with guiding template assignment and redundant via insertion on masks one by one, it may cause some illegal redundant via insertions. For example, for vertices g and h in Figure 5.13(a), if we deal with node C_8 by the RGDS solver, we may have $x_g = 1$, and when we deal with node C_9, we may also have $x_h = 1$. But there is an overlap edge between g and h due to the overlap between *multiplet*s g and h, which indicates that only one of g and h can be patterned. In order to handle this issue, we propose a trick as follows.

After guiding template assignment and redundant via insertion on a mask, the corresponding positions of vias and inserted redundant vias are marked as occupied. Then we re-find all possible redundant via candidates on the remaining unoccupied positions for the other masks, and reconstruct conflict graphs, respectively.

It must be remarked that, for the RGDD and RGDT problems, Lemma 5.4.2 does not hold. That means a via and its redundant via (if existing) may be assigned to different guiding templates. Under Assumption 5.3.1, we may miss the optimal guiding template assignments for the RGDD and RGDT problems. Actually, the proposed method can be extended to considering the s-cenario without Assumption 5.3.1 by introducing some extra *multiplet*s. But, such consideration would greatly increase the size of solution space, and such consideration cannot always bring improvement on solution quality. Hence, we still consider the RGDD and the RGDT problems under Assumption 5.3.1 by using the methods in Section 5.3. In addition, our two stage method also may lose the optimal solution. Fortunately, for the RGDD and the RGDT problems, after assigning all vias to M masks, the layout in every mask is sparse, and the guiding template assignment under Assumption 5.3.1 may be good enough. Our experimental results verify this statement.

5.6 Experimental Results

Our methods for guiding template assignment and redundant via insertion for DSA-MP were programmed in C++ and run on a personal computer with 2.7GHz CPU, 8GB memory and the Unix operating system. We tested our methods on the benchmarks based on the OpenSPARC T1 design [5], provided by Ou *et al.* [70], and on the MCNC benchmarks and the industry Faraday benchmarks, provided by Fang *et al.* [39]. Since the usable guiding templates in [70] and [39] are different, we designed two different experiments for fair

comparisons. In both experiments, the Branch-and-Bound solver in the software package CPLEX [2] was chosen as our ILP solver.

5.6.1 First Experiment

We implemented our methods for the redundant via insertion and guiding template assignment for DSA with single patterning (RGDS), double patterning (RGDD), and triple patterning (RGDT) problems on the OpenSPARC T1 benchmarks to compare with the methods in [70][1]. As [70], layouts of all benchmarks were transformed to grid models. The grid size was set to one metal pitch. The distance between a via and its redundant via was set to one metal pitch, and the optical resolution limit spacing d_s of adjacent guiding templates was set to two metal pitches. The available guiding template types were t_1, t_2, t_3 and t_4 as in Figure 5.2. The experimental results of RGDS, RGDD and RGDT are listed in Tables 5.1, 5.2 and 5.3, respectively, where the last two rows of every table are the average results and the ratios based on some corresponding average results.

In Table 5.1, four solvers for the RGDS problem are compared. The data in the column "SP_ILP [70]" are the results cited directly from [70]. The column "Our_ILP" lists our results, which were obtained by solving the ILP of the CMWIS problem using ILP solver. The results in "Our_MWIS" were obtained by using Algorithm 5.2 in Section 5.4.2. From Table 5.1, we can see that the results obtained by our ILP are slightly better than by the ILP in [70]. The difference is mainly generated by more redundant via candidates in our experiments. For fair, we add another comparison by testing the ILP formulation in [70] on our redundant via candidates, and list the obtained results into the column "Imp_ILP [70]".

In Table 5.1, "#V" is the number of the original vias, and "CPU(s)" is the runtime in seconds. "MR(%)" and "IR(%)" are respectively the manufacture

[1]The results of [70] are directly cited for comparisons. Their platform was a computer with Core i7 3.4GHz CPU and 32GB memory, and CBC was used as the ILP and LP solver.

rate and the redundant via insertion rate, which are calculated by

$$\mathrm{MR} = \frac{\#\mathrm{MV}}{\#\mathrm{V}} \times 100\%, \quad \mathrm{IR} = \frac{\#\mathrm{RV}}{\#\mathrm{V}} \times 100\%,$$

where #MV is the number of manufactured vias (excluding redundant vias), #RV is the number of inserted redundant vias. MR and IR are primary indicators for comparison. And β in Subsection 5.4.1 was set to 0.5 for our algorithms[2] for balancing MR and IR.

Compared with the ILP formulation in [70], our ILP formulation can be solved faster and can obtain better results. From the row "Ratio" in Table 5.1, it can be seen that the improvements of "Our_ILP" over "SP_ILP [70]" on the average MR and IR are 5% and 1%, respectively. And the average runtime of "SP_ILP [70]" is 1.55× more than that of "Our_ILP". Although both of the results in columns "Our_ILP" and "Imp_ILP [70]" are almost equal, the average runtime of "Imp_ILP [70]" is 2.38× more than that of "Our_ILP". These comparisons show that our ILP formulation is better than that in [70] for the RGDS problem. Furthermore, comparing "Our_MWIS" with "Our_ILP", it can be found that "Our-ILP" achieves a little better average MR and IR (the improvements are 1% and 1% respectively) than those of "Our_MWIS", but the average runtime is about 7× more than that of "Our_MWIS". This shows our MWIS algorithm is fast and effective. Actually, for most of the connected components, our MWIS solver also can achieve the optimal result.

Table 5.2 lists the comparison results of two solvers on the RGDD problem. "DP_AP [70]" is the approximation algorithm proposed by Ou *et al.* in [70]. "Our_DP" is our two stage method for the RGDD problem. "DP_AP [70]" is a linear program relaxation based method, which is much faster than ILP based method. Both of "Our_DP" and "DP_AP [70]" achieve almost the same performance. However, the average runtime of "DP_AP [70]" is 5.79× more

[2] In [70], β was set to 250, while in [39], β was set to a value much less than 1.

Table 5.1: Comparison with the ISPD 2016 work on the RGDS problem

Circuits	#V	SP_ILP [70]			Imp_ILP [70]			Our_MWIS			Our_ILP		
		MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)
efc	4983	76.33	75.15	0.85	80.59	76.09	2.36	80.35	75.49	0.18	80.75	76.28	1.33
ecc	5523	80.1	78.68	0.84	84.03	79.28	2.74	83.98	78.6	0.19	84.35	79.59	1.44
ffu	7026	78.49	76.77	1.17	82.23	77.32	3.35	81.7	76.4	0.22	82.25	77.51	1.96
alu	7046	74.65	72.79	1.33	81.64	75.21	3.62	81.31	75.61	0.23	81.88	75.67	2.11
byp	28847	75.14	70.21	6.31	75.24	69.1	12.55	74.82	68.35	0.78	75.56	69.38	6.25
mul	62988	70.23	68.59	30.98	75.21	69.89	38.94	74.54	68.69	2.03	75.33	70.01	13.64
Avg.	19402	75.82	73.70	6.91	79.82	74.48	10.59	79.45	73.86	0.61	80.02	74.74	4.45
Ratio		0.95	0.99	1.55	1.00	1.00	2.38	0.99	0.99	0.14	1.00	1.00	1.00

Table 5.2: Comparison with the ISPD 2016 work on the RGDD problem

Circuits	DP_AP [70]			Our_DP		
	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)
efc	98.63	96.84	3.36	98.58	97.09	2.56
ecc	98.55	97.44	3.80	99.19	98.17	2.28
ffu	98.83	97.35	4.99	98.52	97.44	3.12
alu	98.14	96.39	5.40	97.87	97.01	3.51
byp	97.29	91.35	41.41	97.34	92.11	15.85
mul	96.39	94.91	417.76	97.65	95.21	55.07
Avg.	97.97	95.71	79.45	98.19	96.17	13.73
Ratio	1.00	1.00	5.79	1.00	1.00	1.00

than that of “Our_DP”, which shows that “Our_DP” is a very fast method for the RGDD problem. The great improvement of runtime is due to our two stage process, in which we divide a large and dense conflict graph into many small size connected components.

Table 5.3: Comparison with the ISPD 2016 work on the RGDT problem

Circuits	TP_AP [70]			Our_TP		
	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)
efc	99.53	97.89	7.38	99.94	99.04	1.72
ecc	99.67	98.62	7.93	99.93	99.60	1.41
ffu	99.45	97.93	12.02	99.93	99.19	1.83
alu	99.16	97.44	11.55	99.88	98.95	2.11
byp	98.98	92.88	136.69	99.83	95.42	10.10
mul	98.31	96.85	1613.23	99.78	98.41	47.12
Avg.	99.18	96.94	298.13	99.88	98.44	10.72
Ratio	0.99	0.98	27.82	1.00	1.00	1.00

In Table 5.3, “TP_AP [70]” is the approximate algorithm proposed in [70]. “Our_TP” is our two stage method for the RGDT problem. Comparing “Our_TP” with “TP_AP [70]”, “Our_TP” obtains greater average MR and IR with the improvements 1% and 2%, respectively. Moreover, the average runtime of “TP_AP [70]” is 27.82× more than that of “Our_TP”, which shows that “Our_TP” is a very fast algorithm for the RGDT problem. Furthermore, from

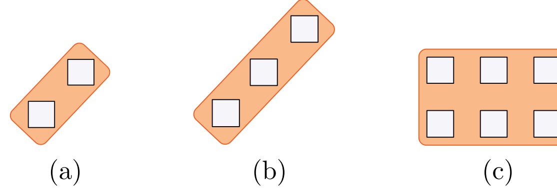


Figure 5.14: Three more usable types of guiding templates. (a) t_5. (b) t_6. (c) t_7.

the respect of average runtime, "TP_AP [70]" in Table 5.3 is much slower than "DP_AP [70]" in Table 5.2, since the numbers of variables and constraints in "TP_AP" are much more than those in "DP_AP". On the contrary, the average runtime of "Our_TP" in Table 5.3 is a little less than that of "Our_DP" in Table 5.2. The main reason is that the conflict graph is divided into more connected components in "Our_TP" in the first stage of our two stage process.

It must be noted that some vias and redundant vias cannot be fabricated even for triple patterning. If the number of un-fabricated vias and redundant vias is small, complementary electron beam lithography (CEBL) is a promising technique for further manufacturing the vias and redundant vias, which is low cost and high resolution [54]. Otherwise, if the remained un-manufacturable vias or redundant vias are numerous, then another more mask patterning might be considered.

5.6.2 Second Experiment

In the second experiment, we further compare our methods with the method in [39] on the MCNC and the industry Faraday benchmarks for the RGDS problem. Layouts of all benchmarks are also transformed to grid models, where a grid size is one metal pitch, and the optical resolution limit spacing d_s of adjacent guiding templates is set to one metal pitch too.

For this comparison, three more guiding template types t_5, t_6 and t_7 are used besides the types t_1, t_2, t_3 and t_4 in Figure 5.2. The three guiding template types are depicted in Figure 5.14. In order to form the guiding templates t_5

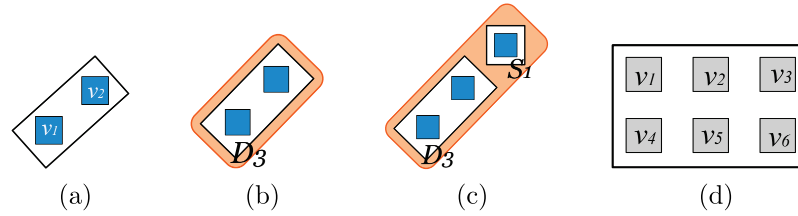


Figure 5.15: (a) D_3. (b) (c) All possible combinations of *multiplets* to form guiding templates t_5 and t_6. (d) sextet.

and t_6, correspondingly, we introduce a new *multiplet* D_3 as shown in Figure 5.15(a). Under Assumption 5.3.1, the possible combinations of *multiplets* to form the guiding templates t_5 and t_6 are listed in Figure 5.15(b) and Figure 5.15(c), respectively. Then the template edge between *multiplets* S_1 and D_3 can be calculated. There exist some incompatibility structures for t_6 templates, which are similar for t_3 in Figure 5.10(b). To exclude these incompatibility structures, Constraint (5.3) should be added into the MWIS problem (5.2).

In order to handle the template type t_7 with six holes, we introduce a new *multiplet* called *sextet* as shown in Figure 5.15(d), where v_i, $i \in \{1, 2, \cdots, 6\}$, can be a via or a redundant via. We use the detection windows in [39] to identify all *sextets*, and construct a new conflict graph by adding the *multiplets* D_3 and *sextet* and the related edges. Then we solve the RGDS problem by ILP solver and the MWIS solver on the new conflict graph respectively.

In Table 5.4, we list the results of three solvers "ILP [39]"[3], "Imp_ILP [39]", "Our_ILP" on the RGDS problem, where the last two rows of the table are the average results and the ratios based on the average results of "Our_ILP". "Imp_ILP [39]" is the same as "Imp_ILP [70]" in Table 5.1, in which the results were obtained by solving the ILP in [39] on our redundant via candidates. The results in column "Our_ILP" were obtained by using CPLEX [2] to solve our ILP formulation.

Comparing "ILP [39]" with "Our_ILP", we improve the average IR by 4%.

[3]The results of ILP [39] are directly cited from the paper, in which the platform was Core i7 3.5GHz with 72GB memory, and CPLEX was used as the ILP solver.

Table 5.4: Comparison with the ILP in TCAD 2017 work on the RGDS problem

Circuits	#V	ILP [39]			Imp_ILP [39]			Our_ILP		
		MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)
struct	12551	99.86	99.42	30.00	99.67	99.52	58.21	99.60	99.54	13.41
primary1	8764	99.80	99.21	28.00	99.68	99.63	43.56	99.53	99.48	8.95
primary2	32684	99.54	98.97	72.00	99.67	99.46	96.51	99.68	99.46	36.55
s5378	8649	81.10	61.78	11.00	84.09	68.87	13.55	85.11	68.81	2.75
s9234	6874	80.07	59.54	9.00	82.80	68.03	8.62	82.96	69.17	2.37
s13207	18780	84.36	66.93	23.00	84.22	70.99	29.98	83.86	72.24	5.51
s15850	22694	82.70	64.41	28.00	83.99	69.17	36.83	84.23	70.80	7.37
s38417	54225	84.01	65.71	65.00	83.45	70.70	82.10	83.47	71.74	18.07
s38584	74155	81.53	63.01	88.00	82.90	69.20	118.67	83.15	69.87	25.26
dma	34697	97.85	95.29	55.00	97.22	95.88	69.53	97.12	95.71	15.71
dsp1	30317	99.05	97.57	53.00	98.42	96.66	66.71	98.36	97.54	17.34
dsp2	31301	98.51	96.68	52.00	98.18	96.93	63.57	98.22	97.45	17.01
risc1	43858	98.78	96.93	75.00	98.14	96.83	112.82	98.14	96.83	24.02
risc2	44385	98.80	96.91	77.00	98.16	96.87	112.65	98.09	96.74	23.73
Avg.	30281	91.86	83.03	47.57	92.19	85.55	65.24	92.25	86.10	15.58
Ratio		1.00	0.96	3.05	1.00	0.99	4.19	1.00	1.00	1.00

Table 5.5: Comparison with the graph method in TCAD 2017 work on the RGDS problem

Circuits	Graph [39]			Our_MWIS		
	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)
struct	99.86	99.06	2.60	99.72	99.32	9.33
primary1	99.80	98.71	3.20	99.53	99.28	8.42
primary2	99.55	97.73	8.00	99.64	99.01	35.31
s5378	81.11	56.89	0.30	83.95	67.91	0.27
s9234	80.07	55.34	0.30	82.87	68.32	0.25
s13207	84.35	62.35	0.80	83.09	70.54	0.90
s15850	82.71	59.65	0.90	84.03	70.08	1.20
s38417	84.00	61.08	2.40	83.05	71.27	4.42
s38584	81.52	58.23	3.10	82.82	69.04	7.38
dma	97.86	92.89	3.10	96.57	94.19	1.94
dsp1	99.06	96.14	3.70	98.05	96.35	2.25
dsp2	98.50	95.20	3.40	97.77	96.19	2.45
risc1	98.77	95.18	6.10	97.79	95.89	5.08
risc2	98.81	95.09	5.80	97.71	95.75	5.27
Avg.	91.85	80.25	3.12	91.90	85.22	6.03
Ratio	1.00	0.94	0.52	1.00	1.00	1.00

Furthermore, the average runtime of “ILP [39]” is about 3× more than that of “Our_ILP”. Both of the results in columns “Imp_ILP [39]” and “Our_ILP” are almost the same, but the average runtime of “Imp_ILP [39]” is about 4× more than that of “Our_ILP”. These verify that our ILP formulation is better than that in [39] for RGDS problem, which has less variables and constraints.

The results in “Graph [39]” and “Our_MWIS” of Table 5.5 were obtained by the graph method in [39] and our MWIS solver, respectively. The average runtime of “Graph [39]” is half of “Our_MWIS”, but the average IR of “Graph [39]” is 6% less than of “Our_MWIS”.

To demonstrate the scalability of our ILP formulation and the proposed fast MWIS algorithm, we further tested them on nine much larger benchmarks from the ISPD 2015 Placement Contest [17], which were processed by NTUPlace4dr [45] and Cadence SoC Encounter [1]. The experimental results are listed in Table 5.6. Compared with “ILP [39]”, “our_ILP” achieves 2.54× shorter runtime.

Table 5.6: Comparison with the TCAD 2017 work on the RGDS problem

Circuits	#V	ILP [39]			Graph [39]			Our_MWIS			Our_ILP		
		MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)
mgc_des_perf_1	736470	94.04	76.89	962.00	94.05	72.37	52.00	92.58	82.74	138.27	95.42	83.95	478.93
mgc_des_perf_a	765166	95.98	76.41	928.00	95.98	72.58	60.00	94.23	80.41	113.83	96.70	81.75	362.21
mgc_des_perf_b	720412	96.68	83.35	951.00	96.69	79.75	61.00	94.56	85.94	128.57	96.81	88.17	376.58
mgc_fft_1	238324	94.79	78.53	317.00	94.78	73.97	17.00	92.57	84.14	13.03	95.26	85.28	112.22
mgc_fft_2	255324	95.64	81.70	336.00	95.65	77.42	18.00	93.36	85.74	14.31	95.93	87.90	115.97
mgc_fft_a	234441	96.45	84.11	328.00	96.46	79.93	24.00	94.06	87.11	15.01	96.42	88.21	105.31
mgc_fft_b	247866	95.36	79.66	334.00	95.36	74.99	24.00	93.33	84.52	14.96	95.88	85.59	123.28
mgc_pci_bridge32_a	198376	96.80	82.88	258.00	96.79	79.10	14.50	94.56	86.22	9.99	96.80	87.21	88.85
mgc_pci_bridge32_b	172483	97.55	88.74	273.00	97.54	86.06	20.10	95.08	90.20	13.59	97.21	91.36	81.38
Avg.	396540	95.92	81.36	520.78	95.92	77.35	32.29	93.81	85.22	51.28	96.27	86.60	204.97
Ratio		1.00	0.94	2.54	1.00	0.89	0.16	0.97	0.98	0.25	1.00	1.00	1.00

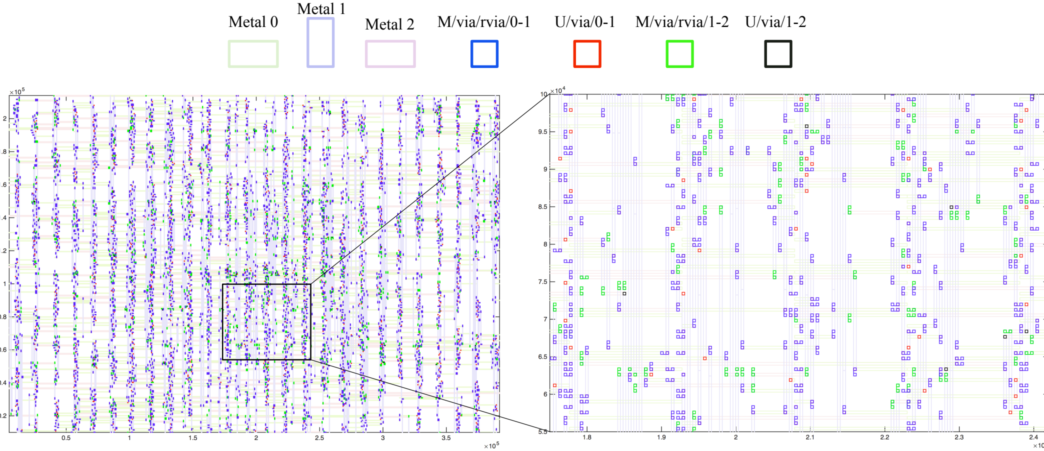


Figure 5.16: A result of the RGDS problem on benchmark `s9234`.

Compared with the graph method in [39], our ILP based method improves the average insertion rate by 11%. On these larger test cases, our MWIS based algorithm is still fast and effective. It takes only 0.25× average runtime of our ILP based method with only 3% manufacture rate loss and 2% insertion rate loss.

In the layout, there are three metal layers, Metal 0, Metal 1 and Metal 2, and two via layers V_{0-1} and V_{1-2}. In the legend of Figure 5.16, M/via/rvia/0-1 (in blue) denotes manufacturable via and redundant via on layer V_{0-1}, U/via/0-1 (in red) denotes un-manufacturable via on layer V_{0-1}, M/via/rvia/1-2 (in green) denotes manufacturable via and redundant via on layer V_{1-2}, and U/via/0-1 (in black) denotes un-manufacturable via on layer V_{1-2}.

5.7 Summary

In this chapter, we have considered the redundant via insertion and guiding template assignment for the DSA with multiple patterning (RGDM) problem, including single patterning (RGDS), double patterning (RGDD) and triple patterning (RGDT). First, for the RGDS problem, we constructed a new ILP formulation basing on our conflict graph. The vertices in the conflict graph are

*multiplet*s instead of guiding template assignments (GTAs), which can greatly reduce the size of the conflict graph. To fast solve the ILP, a local optimal MWIS solver was introduced to obtain a local optimal result. Second, for the RGDD and RGDT problems, we proposed a two stage method. At the first stage, a contraction graph is constructed, and the max-M-cut problem is formulated and solved to obtain a mask assignment. At the second stage, our MWIS solver for RGDS is used to obtain a redundant via insertion and guiding template assignment for every mask. Experimental results validate the efficiency and effectiveness of our ILP formulation and algorithms.

Chapter 6 Redundant Via Insertion and DSA Guiding Template Assignment with Dummy Via

6.1 Introduction

Due to various reasons such as random defects, cut misalignment, electro migration and thermal/mechanical stress [85], a single via may fail partially or completely. Via failure heavily impacts functionality and yield of a design [57, 94]. Up to now, redundant via (RV) insertion has been considered as the necessary step for reducing via failure, and then improving circuit reliability and yield [20, 56]. The redundant via insertion technique is that we should insert a redundant via close and align to every via. As shown in Figure 6.1(a), the four positions r of via v are called the redundant via candidates (RVC), since a redundant via of v may be inserted at one of the four positions. In addition, an inserted redundant via should not cause any circuit short, i.e., an inserted redundant via should not overlap with any metal wire from other nets of wires. In Figure 6.1(b), the only legal RVCs of vias v_1, v_2 and v_3 are r_1, r_2 and r_3, respectively.

Block copolymer directed self-assembly (DSA) is considered as a promising fit technique for via layers beyond the $7nm$ node technology [98, 99]. Specially, the previous works have made many significant improvements on manufacturing, modeling, simulation and graphoepitaxy of DSA [64, 87]. These improvements make patterning feature by DSA technology possible. In DSA, the block copolymers form cylinders, and the remainder material can be used to fabricate contacts/vias by removing cylinders. To generate irregularly distributed vias using DSA, guiding templates surrounding vias are required [55, 77]. These guiding templates are manufactured by the conventional optical lithography,

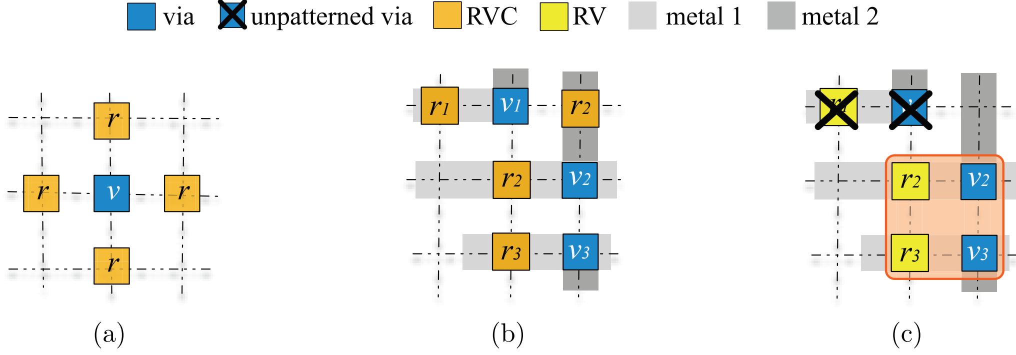


Figure 6.1: (a) A via and four possible positions of its RVCs. (b) Legal RVCs. (c) A redundant via insertion and guiding template assignment result.

and thus the resolution is limited by the pattern pitch. In addition, for improving the resolution, some close vias may be put into a multi-hole guiding template [7, 90]. Naturally, we should assign a guiding template for every via. The guiding template assignment problem is crucial, which has been well investigated in [7, 52, 55, 77, 90].

To improve the resolution, some close vias (includes redundant vias) may be put into a multi-hole guiding template [7, 90]. Naturally, we should assign a guiding template for each via and redundant via. Figure 6.1(c) shows a guiding template assignment for the layout in Figure 6.1(b), where vias v_2 and v_3, and redundant vias r_2 and r_3 are assigned to a four-hole template, however via v_1 and redundant via r_1 cannot be guided and patterned due to the resolution limit. The guiding template assignment problem is crucial since it patterns as many as possible vias and redundant vias [7, 52, 90].

In the traditional design process, the redundant via insertion and the manufacture of via layers are processed in two separate stage. Fang *et al.* in [39] first concurrently considered the redundant via insertion and DSA guiding template assignment problem. For this concurrent consideration, both the number of insertable vias (insertion rate, IR) and the number of manufacturable vias (manufacture rate, MR) are increased, especially the number of manufacturable vias. To improve both of the insertion rate and the manufacture rate, some

techniques are used, such as wire bending, dummy via insertion and multiple patterning. In [38], Fang *et al.* investigated the redundant via insertion and DSA guiding template assignment problem with wire bending. By local perturbing some metal wires, it becomes more free for inserting redundant vias at the cost of increasing the wirelength. In addition, in [70] Ou *et al.* considered the problem with multiple patterning, and gave an ILP based approach.

Although the bending wire insertion and multiple patterning techniques can be used to improve the numbers of insertion and manufacture rates, one or more following defects limit the usability of wire bending and multiple patterning for via manufacture. First, the multiple patterning process will multiply the manufacture cost; Second, wire bending will increase the wirelength; Third, for the advanced 1-D metal layer design, wire bending are unwarrantable. To avoid the above three drawbacks, Hung *et al.* [47] studied the problem with dummy via insertion, in which some dummy vias are inserted for assisting formation of guiding templates.

After using dummy via insertion, the layout is more free for inserting redundant vias and using multi-hole guiding templates, which achieves a higher insertion rate and manufacture rate. In [47], the authors generated all guiding template candidates for all the redundant via candidates, dummy via candidates, and immediate neighbor vias. The generated guiding template candidates are utilized to express solution space, which is extremely large.

It is desirable to derive a more effective and efficient solution expression and its optimization method for the guiding template assignment with redundant via and dummy via insertions problem. The main contributions of this paper are summarized as follows.

- Unlike the guiding template candidate solution expression in [47], we introduce a dictionary-based manner to express solution. The new compact expression can discard redundant solutions significantly.

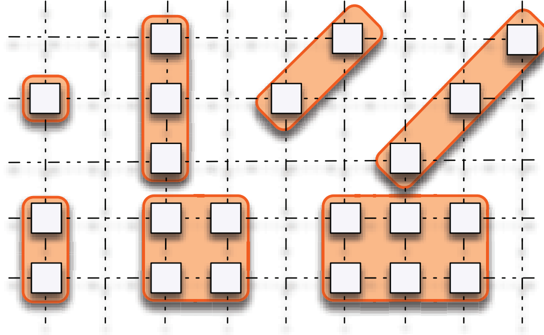


Figure 6.2: Seven usable types of guiding templates.

- Honoring the compact solution expression, we construct a conflict graph with dummy via insertion, and then formulate the problem as a constrained maximum weight independent set problem (CMWIS).
- We relax the problem to an unconstrained nonlinear programming (UNP) to make a good tradeoff between solution quality and runtime.
- We develop a line search optimization algorithm to solve the UNP. In addition, an effective initial solution generation operation is proposed to avoid involving undesirable local extremum.
- Experimental results show the efficiency and effectiveness of our solution expression and optimization method. Specifically, our algorithm achieves comparable experimental results with a state-of-the-art work, and saves 94% runtime.

6.2 Preliminaries

In this section, we introduce some related concepts and the considered problem.

6.2.1 Redundant Via Insertion

For convenience of solving the problem, we consider the redundant via insertion and DSA guiding template assignment problem on a grid graph. Suppose

the grid coordinate of a single via v_i is (x_i, y_i). For the four grid coordinates $(x_i - 1, y_i)$, $(x_i + 1, y_i)$, $(x_i, y_i - 1)$ and $(x_i, y_i + 1)$, each of them is called a redundant via candidate (RVC) of v_i, if this position is not occupied by other vias or metal wires from other nets. The objective is inserting a legal redundant via for every via.

6.2.2 Guiding Template Assignment

For the DSA technique, template is needed for guiding the holes. Since irregular guiding template has a higher chance of generating overlay error, to guarantee the overlay accuracy, we only use some regular guiding templates with few holes. In this chapter, the usable seven types of guiding templates are shown in Figure 6.2 as of [39]. Furthermore, for high resolution and focal depth of guiding templates, the spacing between neighboring guiding templates should not be less than the optical resolution limit spacing d_s. Generally, d_s is set not less than the distance between a redundant via and its related via, and not less than the hole pitch in a guiding template. We need to decide the assignment of guiding templates, such that more vias and redundant vias can be surrounded by guiding templates.

Figure 6.1(c) shows a guiding template assignment for the layout in Figure 6.1(b), where vias v_2 and v_3, and redundant vias r_2 and r_3 are assigned to a four-hole template. However, via v_1 and redundant via r_1 cannot be patterned due to the resolution limit.

6.2.3 Dummy Via Insertion

Vias or redundant vias manufactured by DSA technique must be guided by some templates. And the most popular manufacture technique for these guiding templates is conventional lithography. For some dense structures in a layout, some used guiding templates cannot be manufactured due to the limitation of optical resolution. Furthermore, the vias or redundant vias guided by these

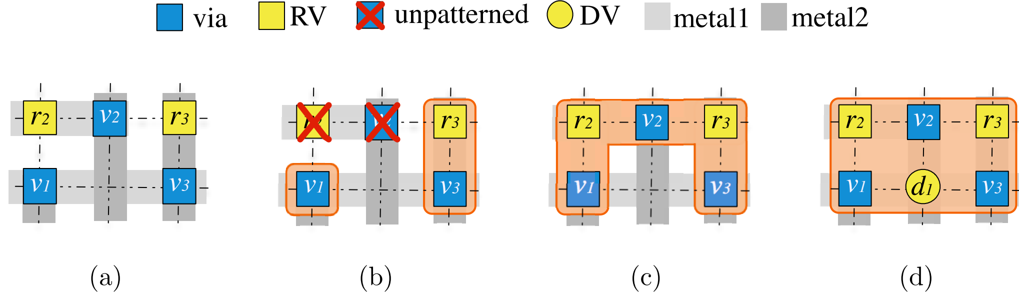


Figure 6.3: (a) A layout. (b) Redundant via insertion and guiding template assignment. (c) An irregular guiding template. (d) A result with dummy via insertion.

unmanufactured guiding templates cannot be formed by DSA block copolymer. Given a layout in Figure 6.3(a), due to the limitations of the used templates in Figure 6.2 and the small pitch between vias, via v_2 and redundant via r_2 are unpatterned in Figure 6.3(b).

We cannot change the pitch between vias, and the used guiding templates must be regular, hence the shape of guiding template in Figure 6.3(c) is undesirable for DSA block copolymer. Therefore, the only effective trick is changing the shape of a structure of vias by adding some dummy vias (DV), such that the structure of vias matches the used guiding template.

In a circuit, dummy via does not connect to any wire, which is only used for filling the guiding template. The insertion of dummy vias should satisfy the following two conditions: i) the insertion can make up a multi-hole (not less than three holes) guiding template with other vias or redundant vias; ii) it can improve the insertion rate or manufacture rate. Obviously, if an inserted dummy via does not satisfy the above two conditions, then the dummy via is useless. Figure 6.3(d) shows one result of guiding template assignment with careful insertion of dummy via d_1, wherein a six-hole template is used to guide the vias and RVs.

After finding the possible RVCs, we should find all potential guiding template assignments for every via. In a grid graph, if all grid points covered by

a multi-hole (not less than three holes) guiding template are vias or redundant vias, then the guiding template does not need dummy vias; otherwise, every empty grid point needs a dummy via for forming a complete guiding template. If these needed dummy vias on the empty grid points satisfy the above two conditions, then these empty grid points are marked as dummy via candidates (DVC).

6.2.4 Problem Formulation

The problem aims at inserting a redundant via for every via, and manufacturing all vias and their redundant vias by the DSA technique with the help of dummy via insertion. The redundant via insertion rate (IR) and the manufacture rate (MR) are considered as evaluation indicators in [39, 70]. The insertion rate is defined as the ratio of the number of inserted redundant vias to the number of vias. And the manufacture rate is the ratio of the number of manufacturable vias to the number of vias.

Since via manufacturability is generally the first consideration for yield, an inserted redundant via should not cause generation of an infeasible via pattern [39]. Hence, we assume that a redundant via cannot be inserted, if its related single via is not manufacturable. According to this assumption, we can easily obtain another equivalent assumption, i.e., a via and its redundant via (if exists) should be assigned to the same guiding template [70]. Under this assumption, the redundant via insertion and DSA guiding template assignment with dummy via insertion (RDD) problem is formulated as follows:

Problem 6.2.1. Given a post-routing via layers layout, the usable types of guiding templates, and the optical resolution limit spacing d_s, insert a redundant via for every via, assign guiding templates for vias, redundant vias and dummy vias, such that: i) the inserted redundant vias are legal; ii) the spacing between neighboring guiding templates should not be less than d_s. The objective is maximizing $MR + \beta \cdot IR$, where β is a weighting parameter.

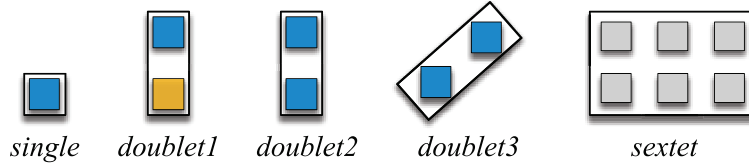


Figure 6.4: *Multiplet*s.

6.3 Conflict Graph Construction

After finding all possible redundant via candidates and dummy via candidates, we introduce a concept of *multiplet*, where a *multiplet* is composed of some vias and some candidates, and *multiplet*s can be used to compose guiding templates. Five types of *multiplet*s are shown in Figure 6.4, where a *single* includes a via; a *doublet*1 includes a via and a redundant via candidate; a *doublet*2 includes two aligned vias; a *doublet*3 includes two diagonal vias; and a *sextet* includes six vias or redundant vias, which can be covered by a six-hole guiding template.

Under our assumption that a via and its redundant via should be assigned to the same guiding template, we have some observations: i) a guiding template cannot only include redundant vias; ii) the number of redundant vias must not be larger than the number of vias in a guiding template; iii) for every redundant via in a guiding template, its via must also be in the guiding template. Then the seven types of usable guiding templates in this chapter can be formed by grouping some *multiplet*s, as shown in Figure 6.5, where we only list the vertical cases and skip the horizontal cases due to similarity.

A DVC must belong to some guiding template, and then it must belong to some *multiplet*. At the DVC finding stage, we can easily find out which *multiplet* a DVC belongs to.

Given a result of finding redundant via candidates as in Figure 6.6(a), we can identify all possible *multiplet*s as in Figure 6.6(b), and these *multiplet*s are regarded as vertices in the conflict graph. Based on these vertices, we introduce

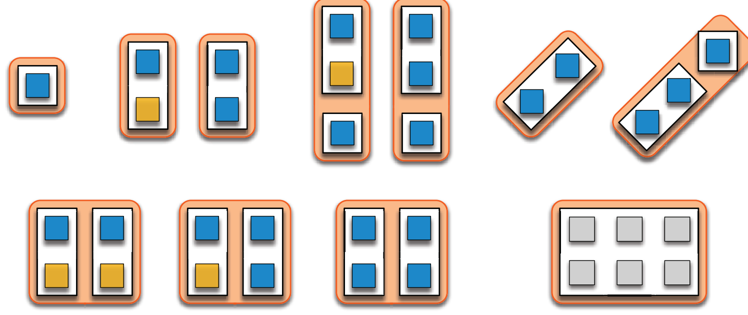


Figure 6.5: All possible combinations of *multiplets* to form the seven types of guiding templates.

some edges between them. The conflict graph of Figure 6.6(b) is shown in Figure 6.6(c). The detailed definitions of edges and conflict graph are presented as follows.

Definition 6.3.1 (overlap edge)**.** If *multiplets* i and j are overlapped with each other, then there exists an overlap edge e_{ij} between i and j. Let E_O be the set of overlap edges.

Definition 6.3.2 (conflict edge)**.** If the distance between two *multiplets* i and j is not larger than d_s, and there does not exist an overlap edge between i and j, then there exists a conflict edge e_{ij} between i and j. Let E_C be the set of conflict edges.

Definition 6.3.3 (template edge)**.** For two *multiplets* i and j, suppose that at least one of them is not S_1. If i and j can be assigned simultaneously to a guiding template without any design error, and between i and j there exists a conflict edge $e_{ij} \in E_C$, then e_{ij} is also called a template edge between i and j. Let E_T be the set of template edges. Obviously, $E_T \subseteq E_C$.

Definition 6.3.4 (conflict graph, CG)**.** The conflict graph is an undirected graph $CG(V, E)$, where vertex $v \in V$ denotes a *multiplet*, $e_{ij} \in E$ is an edge and $E = (E_C - E_T) \cup E_O$. E_C, E_T and E_O are the sets of conflict edges, template edges and overlap edges, respectively.

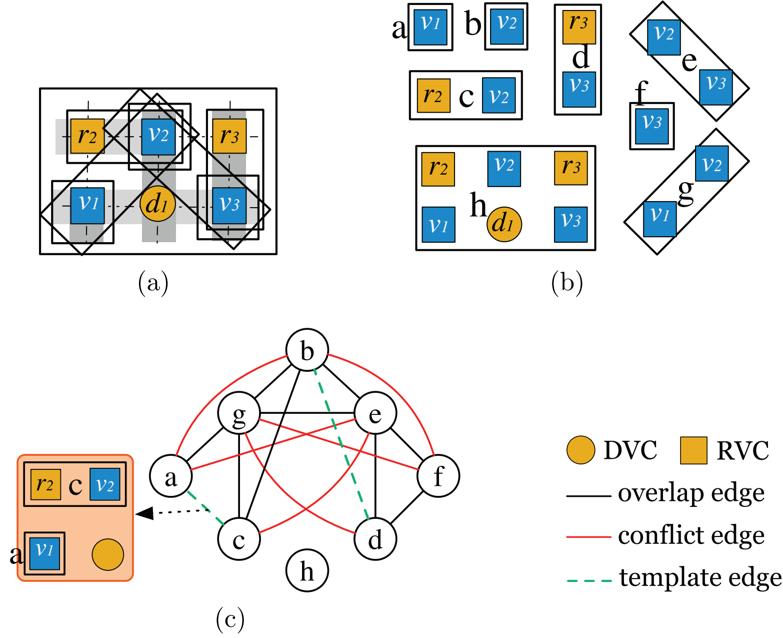


Figure 6.6: (a) A layout after finding RVC and DVC. (b) All *multiplets* of the layout in (a). (c) Conflict graph.

The conflict graph of the layout in Figure 6.6 is constructed in Figure 6.6(c), in which the vertices are the *multiplets* in Figure 6.6(c). The black edges are the overlap edges, the red edges are the conflict edges, and the green dotted edges are the template edges. In Figure 6.6(a), *multiplets* a and g are overlapped with each other at v_1, hence there is an overlap edge between them as in Figure 6.6(c). The distance between *multiplets* a and e is not larger than d_s, hence there is a conflict edge between them. Since *multiplets* a and c can be assigned to a four-hole guiding template as in Figure 6.6(c), there is a template edge between a and c. Note that for the conflict graph of Figure 6.6(b), every vertex is in fact connected to vertex h by an overlap edge, but we skip drawing these overlap edges in Figure 6.6(c) for easiness of visualization.

6.4 Our Algorithms

In this section, we first formulate the redundant via insertion and DSA guiding template assignment with dummy via insertion problem into a constrained

maximum weight independent set (CMWIS) problem. The CMWIS problem is a ILP formulation, which can directly be solved by existing commercial optimization solver. To achieve better trade-off between runtime and solution quality, we relax the CMWIS problem into an unconstrained nonlinear programming (UNP), and propose a line search based optimization algorithm to solve the UNP. Moreover, the convergence of the proposed algorithm is proved.

6.4.1 Constrained Maximum Weight Independent Set Problem

According to the conflict graph of the RDD problem, we construct the constrained maximum weight independent set problem. For the *multiplet*s overlapped with each other, only one of them can be chosen to pattern due to overlap. Thus, in the conflict graph, if between two *multiplet*s i and j with $e_{ij} \in E_O$, then only one of the *multiplet*s can be patterned. Furthermore, if two *multiplet*s are within the optical resolution limit spacing d_s, then only one of them can be patterned due to limitation of resolution, unless the two *multiplet*s are assigned into the same guiding template. That is, if between the *multiplet*s i and j there exists $e_{ij} \in E_C$, but $e_{ij} \notin E_T$, then only one of them can be patterned.

If two *multiplet*s i and j are connected by a template edge, then they may be assigned to the same guiding template, but not necessarily. Specially, for the structure shown in Figure 6.7(a), *multiplet*s i and l are connected to k by two template edges, but i, k and l cannot be simultaneously assigned to the same guiding template, since we do not have a guiding template with four holes aligned in a line. We call the triple $(i,\ k,\ l)$ in Figure 6.7 as incompatibility structure (Inc), and we let INC be the set of Incs.

Definition 6.4.1 (Inc)**.** The incompatibility structure is a structure composed of three *multiplet*s i, k and l, in which e_{ik} and e_{kl} are template edges and there does not exist any edge between i and l.

In addition, different *multiplet*s include different vias and redundant vias. The objective of the RDD problem is maximizing $MR + \beta \cdot IR$, i.e., maximizing

the weighted sum of the number of manufacturable vias and the number of inserted redundant vias. Suppose the weights of a via and a redundant via is 1 and β, respectively. We jointly consider MR and IR by assigning weight w_i to every *multiplet* i as

$$w_i = N_v + \beta \cdot N_r, \tag{6.1}$$

where N_v and N_r are the numbers of included vias and redundant vias by *multiplet* i, respectively. Let W be the set of weights, then the conflict graph $CG(V, E)$ is weighted and written as $CG(V, E, W)$.

Thus, we formulate the RDD problem as the constrained maximum weight independent set (CMWIS) problem

$$\max \quad \sum_{i \in V} w_i x_i \tag{6.2}$$

$$\text{s.t.} \quad x_i + x_j \leq 1, \qquad \forall e_{ij} \in E; \tag{6.2a}$$

$$x_i + x_k + x_l \leq 2, \qquad \forall (i, k, l) \in \text{INC}; \tag{6.2b}$$

$$x_i \in \{0, 1\}, \qquad \forall i \in V. \tag{6.2c}$$

In the CMWIS problem, Constraint (6.2a) shows that, if there exists $e_{ij} \in E_O$ or $e_{ij} \in E_C - E_T$ between vertices i and j, then at most only one of them can be patterned. Constraint (6.2b) is used to force that, if i, k and l compose an Inc, then at most two of them can be patterned.

6.4.2 A Fast Algorithm for The CMWIS Problem

The CMWIS problem is NP-hard, since it contains the maximum weight independent set problem as a special case. Hence, it is time consuming to solve the CMWIS problem by ILP solver for a large scale layout. In this subsection, we develop a fast algorithm to obtain a local optimal solution of the problem.

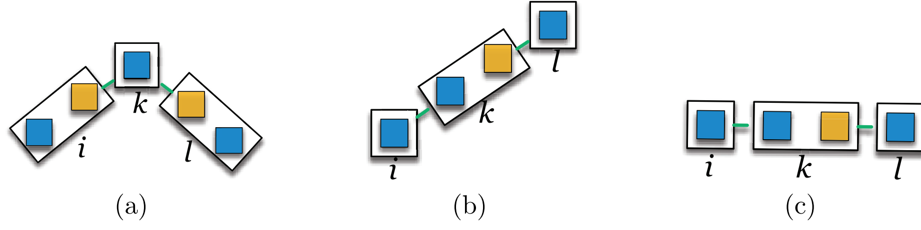


Figure 6.7: Three kinds of incompatibility structures.

It must be remarked that, the obtained conflict graph can be divided into numerous connected components, and we deal with the CMWIS problem on the connected components one by one. Note that the ILP formulation of Problem (6.2) can be relaxed into a linear programming (LP) by discarding integer constraint (6.2c). However, the relaxed linear constraints (6.2a) and (6.2b) of Problem (6.2) would become weak. For example, with $x_i = 0.5$ and $x_j = 0.5$, we have $0.5 + 0.5 \geq 1$, which still satisfies the linear constraint (6.2a). In this section, we would derive an unconstrained nonlinear programming relaxation instead of linear programming relaxation for Problem (6.2).

Firstly, the ILP formulation (6.2) of the CMWIS problem is equivalent to

$$\max_{\mathbf{x}} \quad \sum_{i \in V} w_i x_i \tag{6.3}$$

$$\text{s.t.} \quad x_i x_j = 0, \qquad \forall e_{ij} \in E; \tag{6.3a}$$

$$x_i x_k x_l = 0, \qquad \forall (i,k,l) \in \text{INC}; \tag{6.3b}$$

$$x_i \in \{0,1\}, \qquad \forall i \in V, \tag{6.3c}$$

where $\mathbf{w} = (w_1, w_2, \cdots, w_n)^T \in \mathbb{R}^n$, $\mathbf{x} = (x_1, x_2, \cdots, x_n)^T \in \{0,1\}^n$, $n = |V|$. Then Problem (6.3) can be rewritten as

$$\max_{\mathbf{x}} \sum_{i\in V}\{w_i x_i \prod_{\substack{j\in V\\ e_{ij}\in E}} (1-x_j) \prod_{\substack{k,l\in V\\ (i,k,l)\in \text{INC}}} (1-x_k x_l)\} \tag{6.4}$$

$$\text{s.t.} \quad x_i \in \{0,1\}, \qquad \forall i \in V. \tag{6.4a}$$

Let $\mathbf{B} = (B_{ij}) \in \{0,1\}^{n\times n}$ be the adjacency matrix of the conflict graph CG with $B_{ij} = 1$ if $e_{ij} \in E$ and $B_{ij} = 0$ if $e_{ij} \notin E$. And let $\mathbf{C} = (C_{ikl}) \in \{0,1\}^{n\times n\times n}$ be a three dimensional tensor with $C_{ikl} = 1$ if $(i,k,l) \in \text{INC}$, otherwise $C_{ikl} = 0$. Then, Formula (6.4) is equivalent to

$$\max_{\mathbf{x}} \sum_{i\in V}\{w_i x_i \prod_{j\in V}(1-x_j)^{B_{ij}} \prod_{k,l\in V} (1-x_k x_l)^{C_{ikl}}\} \tag{6.5}$$

$$\text{s.t.} \quad x_i \in \{0,1\}, \qquad \forall i \in V. \tag{6.5a}$$

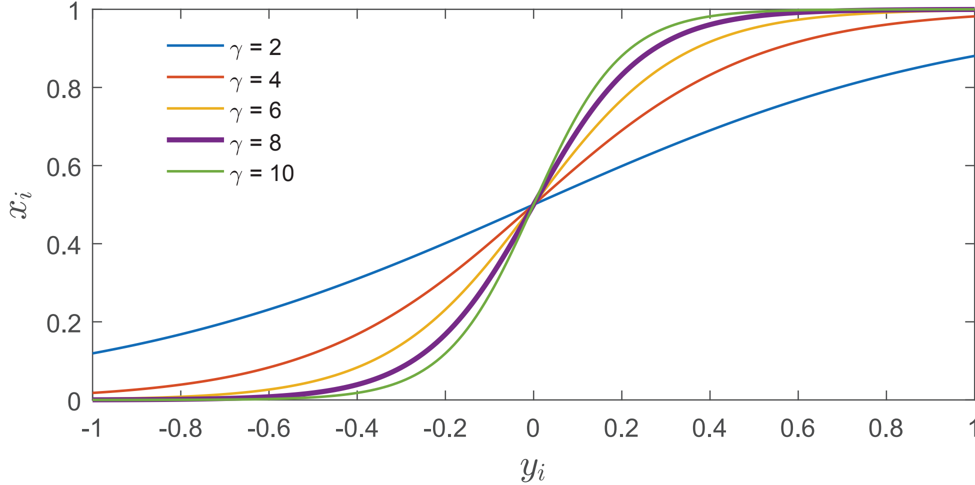


Figure 6.8: Sigmoid function $\sigma(y_i)$ with different γ.

Although Problem (6.5) is equivalent to Problem (6.2), it is still a discrete formulation. To fast solve Problem (6.5), as common in combinatorial optimization, we relax this discrete formulation to the continuous domain. First, we

introduce an auxiliary vector $\mathbf{y} = (y_i) \in \mathbb{R}^n$, and approximate the constraint $x_i \in \{0, 1\},\ \forall i \in V$ with the sigmoid function

$$x_i \approx \sigma(y_i) = (1 + e^{-\gamma y_i})^{-1}. \tag{6.6}$$

Above sigmoid function is used to approximate function

$$x_i = \begin{cases} 0, & y_i \le 0; \\ 1, & y_i > 0. \end{cases} \tag{6.7}$$

The detailed curves of sigmoid function with different γ are plotted in Figure 6.8, where γ is set as 8 in this paper for a sharper sigmoid function.

Then Problem (6.5) is approximated as

$$\max_{\mathbf{y}}\ f(\mathbf{y}) = \sum_{i \in V} \{w_i \sigma(y_i) \prod_{j \in V} (1 - \sigma(y_j))^{B_{ij}} \prod_{k,l \in V} (1 - \sigma(y_k)\sigma(y_l))^{C_{ikl}}\}. \tag{6.8}$$

If we obtain a solution $\mathbf{y}^*$ of Problem (6.8), then the final solution $\mathbf{x}^*$ is obtained by rounding the sigmoid function value $\sigma(y_i^*)$ to the nearest integer, $\forall i \in V$.

Problem (6.8) is an unconstrained nonlinear programming problem. Let

$$g_i(\mathbf{y}) = \sigma(y_i) \prod_{j \in V} (1 - \sigma(y_j))^{B_{ij}} \prod_{k,l \in V} (1 - \sigma(y_k)\sigma(y_l))^{C_{ijk}}. \tag{6.9}$$

Then the objective of (6.8) is $f(\mathbf{y}) = \sum_i w_i g_i(\mathbf{y})$.

Our Algorithm 6.2 aims at finding a maximal solution $\mathbf{y}^* \in \mathbb{R}^n$ of (6.8). At each iteration t, the solution is updated following gradient direction of $f(\mathbf{y})$ as:

$$\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} + \Delta\alpha \nabla f(\mathbf{y}^{(t)}), \tag{6.10}$$

where $\Delta\alpha$ is the steplength, which is obtained by the Wolfe-Powell inexact line search method [69]. And $[\nabla f(\mathbf{y}^{(t)})]_i = \partial f(\mathbf{y}^{(t)})/\partial y_i$ is calculated by

$$[\nabla f(\mathbf{y}^{(t)})]_i = \tag{6.11}$$
$$w_i g_i^{(t)}\{(1-\sigma(y_i^{(t)})) - \sum_j B_{ij}\sigma(y_j^{(t)}) - \sum_k \sum_l C_{ikl} \frac{\sigma(y_k^{(t)})\sigma(y_l^{(t)})(1-\sigma(y_k^{(t)}))}{1-\sigma(y_k^{(t)})\sigma(y_l^{(t)})}\},$$

where $g_i^{(t)} = g_i(\mathbf{y}^{(t)})$. It is easy to show that the first order dynamic in (6.10) increases $f(\mathbf{y}^{(t)})$ in every iteration t, and will convergence to a local optimum.

However, since the objective function (6.8) is highly non-linear and non-concave, the above iteration is highly dependent on the initial solution $\mathbf{y}^{(0)}$ and may converge to a poor local optima. Hence, in order to obtain a better solution, the iteration would be better starting from a good initial solution $\mathbf{y}^{(0)}$. We propose a greedy based algorithm to obtain a good enough initial solution in Algorithm 6.1.

In line 2 of Algorithm 6.1, $w_n(l)$ is the number of vias and redundant vias covered by *multiplet* l. In line 4, the selection weight $w_s(k)$ of *multiplet* k is calculated by

$$w_s(k) = d_c(k) - d_t(k), \tag{6.12}$$

where $d_c(k)$ is the number of conflict edges incident to *multiplet* k, and $d_t(k)$ is the number of template edges incident to *multiplet* k.

After obtaining a good initial solution, we present our Algorithm 6.2 for the CMWIS problem. Algorithm 6.2 increases the objective value at every iteration, and converges to a maximal solution. Experimentally, for many connected components, Algorithm 6.2 even can obtain optimal solutions of the CMWIS problem. According to our experiments, Algorithm 6.2 only takes few iterations

to converge from a good enough initial solution. Hence Algorithm 6.2 can fast obtain a maximal solution.

Algorithm 6.1 Initial solution generation

Input: A connected component of $CG(V, E, W)$;
Output: Initial solution $\mathbf{x}^{(0)}$ of the CMWIS problem;
1: **repeat**
2: $S \leftarrow \{k \mid k \in \text{argmin}_{l \in V} w_n(l)\}$;
3: **repeat**
4: Compute $w_s(k)$ by Equation (6.12), for all $k \in S$;
5: $x_i^{(0)} \leftarrow 1$, where $i = \text{argmin}_{k \in S} w_s(k)$;
6: **for** every j in V with $e_{ji} \in E$ **do**
7: $x_j^{(0)} \leftarrow 0$, and $V \leftarrow V - \{j\}$; if $j \in S$, $S \leftarrow S - \{j\}$;
8: **end for**
9: $S \leftarrow S - \{i\}$, and $V \leftarrow V - \{i\}$;
10: **until** $S = \emptyset$
11: **until** $V = \emptyset$

Algorithm 6.2 CMWIS solver

Input: A connected component of $CG(V, E, W)$, convergence threshold $\delta = 10^{-4}$;
Output: Solution $\mathbf{x}^{\star}$ of the CMWIS problem;
1: Initialize $t \leftarrow 0$;
2: Generate $\mathbf{x}^{(0)}$ by Algorithm 6.1;
3: If $x_i^{(0)} = 1$, let $y_i^{(0)} \leftarrow 1$; otherwise, let $y_i^{(0)} \leftarrow$ -1;
4: **repeat**
5: $\forall i \in V$, compute $g_i^{(t)}$ by Equation (6.9);
6: Obtain $\nabla f(\mathbf{y}^{(t)})$ by Equation (6.11);
7: $\Delta\alpha \leftarrow LineSearch(\mathbf{y}^{(t)})$;
8: $\mathbf{y}^{(t+1)} \leftarrow \mathbf{y}^{(t)} + \Delta\alpha \nabla f(\mathbf{y}^{(t)})$;
9: $t \leftarrow t + 1$;
10: **until** $||\nabla f(\mathbf{y}^{(t)})|| < \delta$
11: Get x_i^* by rounding $\sigma(y_i^{(t)})$ to the nearest integer, for all $i \in V$.

Lemma 6.4.2. Under Equation (6.11), $\sum_i w_i \nabla g_i \geq 0$

Proof. Let $\sigma_i = \sigma(y_i)$. By Equation (6.11),

$$\sum_i w_i \nabla g_i = \sum_i w_i \gamma g_i \{(1-\sigma(y_i))\nabla y_i - \sum_j B_{ij}\sigma(y_j)\nabla y_j \\ - \sum_k C_{ikl} \sum_l \frac{\sigma(y_k)\sigma(y_l)(1-\sigma(y_k))}{1-\sigma(y_k)\sigma(y_l)} \nabla y_k\},$$

Let $u_i = w_i g_i$, $\mathbf{u} = (u_i) \in \mathbb{R}^l$, and the auxiliary matrix $\mathbf{A} = (\mathbf{A}_{ij}) \in \mathbb{R}^{n\times n}$ has the following elements: $\mathbf{A}_{ij} = 1-\sigma(y_i)$, if $i = j$; $\mathbf{A}_{ij} = -\sigma(y_j)$, if $e_{ij} \in E$; $\mathbf{A}_{im} = -\sum_{n\in V} \frac{\sigma(y_k)\sigma(y_l)(1-\sigma(y_k))}{1-\sigma(y_k)\sigma(y_l)}$, if $i, k, l \in INC$. Then $\sum_i w_i \nabla g_i = \gamma \mathbf{u}^T \mathbf{A} \nabla \mathbf{y}$. According to Definition (6.11) of ∇y_i and $\mathbf{A}_{ij}$, we have $\nabla \mathbf{y} = \mathbf{A}^T \mathbf{u}$. Thus $\sum_i w_i \nabla g_i = \gamma \mathbf{u}^T \mathbf{A}\mathbf{A}^T \mathbf{u} \geq 0$. □

According to Lemma 6.4.2, $\sum_i w_i \nabla g_i \geq 0$ in every iteration, thus we have following Theorem 6.4.3.

Theorem 6.4.3. Under Equation (6.11), $f(\mathbf{y})$ does not decrease.

Corollary 6.4.4. Strict inequality $\sum_i w_i \nabla g_i > 0$ cannot be achieved, since $\mathbf{A}\mathbf{A}^T$ is not positive definite.

Proof. We prove that $\mathbf{A}\mathbf{A}^T$ is not positive definite. The CMWIS contains at least one node, e.g., $x_i = 1$. It follows, $\forall j \in V, e_{ij} \in E$, $x_j = 0$. Then, all the elements of ith row of $\mathbf{A}$ are zero, i.e., $\mathbf{A}$ does not have the full rank. Consequently, at least one of the eigenvalues of $\mathbf{A}\mathbf{A}^T$ is zero. □

Theorem 6.4.5. Algorithm 6.2 converges to a local maximum.

Proof. Since $x_i = \sigma(y_i) : \mathbb{R} \to [0,1]$, $\forall i$, then $g_i : \mathbb{R} \to [0,1]$, $\forall i$. Consequently, $\sum_i w_i g_i \leq \mathbf{w}^T \mathbf{1}$. And by Theorem 6.4.3, $f(\mathbf{y}) = \sum_i w_i g_i$ always increases. Thus Algorithm 6.2 converges, and which stop when the gradient $||\nabla \mathbf{y}||_2 = 0$, i.e., in a local maximum. □

With Theorem 6.4.5, we can achieve a local optimal result by performing Algorithm 6.2. In addition, if Algorithm 6.2 starts from a desirable initial

solution $\mathbf{x}^{(0)}$ by Algorithm 6.1, it likely returns a near global optimal result.

6.5 Experimental Results

Our proposed algorithms for the RDD problem are implemented in C++ and run on a personal computer with 2.7GHz CPU, 8GB memory and the Unix operating system. We test our method on MCNC benchmarks and the industry Faraday benchmarks, provided by Fang *et al.* [39]. As [38], layouts of all benchmarks are transformed as grid models, where a grid size is one metal pitch. In the experiments, the distance between a via and its redundant via is set to one metal pitch, and the optical resolution limit spacing d_s of adjacent guiding templates is set to one metal pitch too. The user-defined parameter β is set to 1. The Branch and Bound approach in the software package CPLEX [2] was chosen as our ILP solver.

6.5.1 Effectiveness of ILP

To evaluate the performance of the proposed ILP (6.2) in Section 6.4.1, we compare the obtained results by solving ILP (6.2) with the results from the ILPs in TCAD'17 [39] and in ASPDAC'17 [47]. The experimental comparisons are reported in Table 6.1. The data in the columns "TCAD'17 [39]" and "ASPDAC'17 [47]" are the results in [39] and [47], respectively[1]. The results in "Our_ILP" are obtained by solving our ILP (6.2) in Section 6.4.1. Moreover, in this table, column "#V" lists the numbers of vias, and column "CPU(s)" is the runtimes in second. "MR(%)" and "IR(%)" are, respectively, the manufacture rate and the redundant via insertion rate.

$$\mathrm{MR} = \frac{\#\mathrm{MV}}{\#\mathrm{V}} \times 100\%, \qquad \mathrm{IR} = \frac{\#\mathrm{RV}}{\#\mathrm{V}} \times 100\%.$$

[1] The results of [39, 47] are directly cited from the papers. The platform of [39] was 3.5 GHz Linux Workstation with 72GB memory, while the platform of [47] was 2.0 GHz Linux Machine with 56 GB memory.

Table 6.1: Comparison of computational results of three methods for the RDD problem

Circuits	#V	TCAD'17 [39]			ASPDAC'17 [47]			Our_ILP			Our_Fast		
		MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)	MR(%)	IR(%)	CPU(s)
struct	12551	99.86	99.42	30.00	99.96	99.79	40.47	99.96	99.81	27.44	99.95	99.69	15.31
primary1	8764	99.80	99.21	28.00	99.92	99.33	29.14	99.87	99.44	28.73	99.80	99.45	16.43
primary2	32684	99.54	98.97	72.00	99.92	99.49	111.22	99.82	99.42	92.11	99.70	99.11	47.20
s5378	8649	81.10	61.78	11.00	96.41	75.27	34.41	95.99	76.37	13.62	95.12	75.37	0.56
s9234	6874	80.07	59.54	9.00	96.21	74.51	38.63	95.47	75.98	10.25	94.03	75.40	0.43
s13207	18780	84.35	66.93	23.00	97.13	79.28	99.96	96.04	80.49	29.49	94.83	79.24	1.94
s15850	22694	82.70	64.41	28.00	96.63	77.35	121.94	95.88	79.40	38.37	94.55	77.81	2.69
s38417	54225	84.00	65.71	65.00	96.83	78.13	320.35	96.09	80.38	98.83	94.74	78.82	15.59
s38584	74155	81.53	63.01	88.00	96.37	76.83	416.11	95.41	78.58	145.35	94.23	77.35	27.55
dma	34697	97.85	95.29	55.00	99.61	97.49	208.75	99.53	97.38	42.41	99.16	97.13	8.65
dsp1	30317	99.05	97.57	53.00	99.74	98.45	176.42	99.71	98.44	45.81	99.50	98.19	13.32
dsp2	31301	98.50	96.68	52.00	99.76	98.74	179.37	99.66	98.26	43.50	99.43	98.08	10.87
risc1	43858	98.77	96.93	75.00	99.70	98.04	216.61	99.69	98.12	66.72	99.40	97.84	23.90
risc2	44385	98.79	96.91	77.00	99.70	97.98	229.22	99.65	98.00	69.80	99.41	97.82	23.87
Avg.	30281	91.85	83.03	47.57	98.42	89.33	158.76	98.06	90.01	53.74	97.42	89.38	14.88
Ratio		0.94	0.93	3.20	1.01	1.00	10.67	1.01	1.01	3.61	1.00	1.00	1.00

In the above equations, #MV is the number of manufacturable vias (excluding redundant vias), #RV is the number of inserted redundant vias.

The difference between the results in "Our_ILP" and in "TCAD'17 [39]" is that our work considers dummy via insertion but work TCAD'17 [39] does not. Compared with the computational results of "TCAD'17 [39]", from the row "Ratio", our ILP (6.2) improves MR and IR up to 6% and 7%, respectively. These improvements mainly result from the help of dummy via insertion. Naturally, considering dummy via insertion will extremely increase the size of solution space, which leads to more challenge for solving. In spite of this, our ILP (6.2) achieves less runtime than the ILP in TCAD'17 [39].

Both of the methods in ASPDAC'17 [47] and our ILP (6.2) consider the dummy via insertion as the complementary technique for improving MR and IR. From the comparison in Table 6.1, our ILP achieves almost the same results as the ILP in [47] [2]. It must be noted that, the average runtime of the ILP in [47] is 2.95× slower than ours. The improvement in runtime owes to our compact solution expression, which greatly reduces the solution space.

6.5.2 Effectiveness of Local Optimal Algorithm

In this subsection, we design another experiment to show the performance of our fast algorithm. In Table 6.1, the data in "Our_Fast" are obtained by solving our fast algorithm in Section 6.4.2. Compared with the results in columns "ASPDAC'17 [47]" and "Our_ILP", "Our_Fast" achieves almost the same quality of results. In addition, the average runtime of our fast algorithm is 10.67× and 3.61× less than the ILP based method in ASPDAC'17 [47] and our ILP based method. In order words, compared with the method in [47] and our ILP, our fast algorithm can save 90% and 72% runtime. These comparisons show that our fast algorithm is very effective and efficient.

[2] The slight difference between ASPDAC'17 [47] and Our_ILP (6.2) may be caused by the different setting of parameter β. In [47], β is set to 0.1.

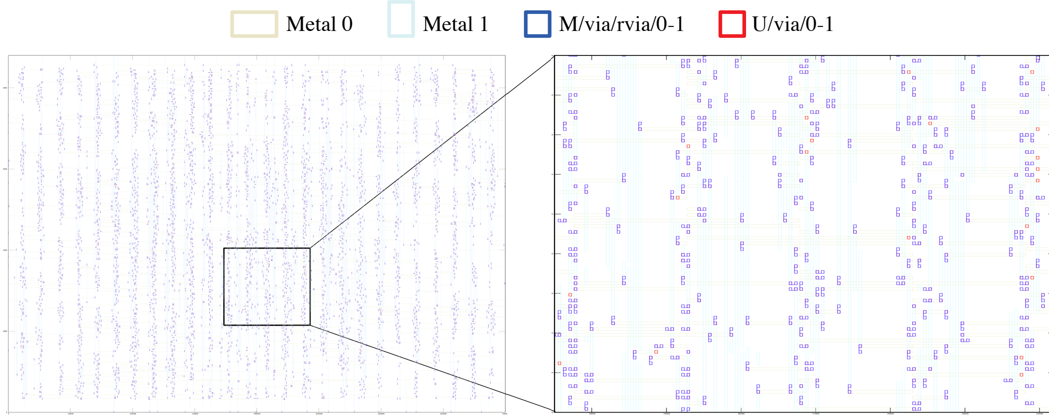


Figure 6.9: The result of benchmark s9234 and its a partial layout on vias between metal 0 and metal 1.

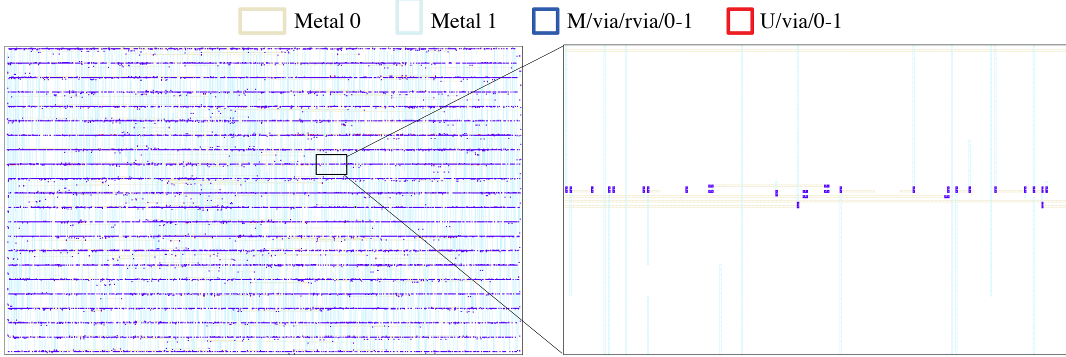


Figure 6.10: The result of benchmark primary2 and its a partial layout on vias between metal 0 and metal 1.

In order to view the result more intuitively, we plot in Figures 6.9 and 6.10 the results for layouts s9234 and primary2, which is obtained by our CMWIS based method. In the layout, we show two metal layers, Metal 0, Metal 1, and two via layers V_{0-1}. In the legend of Figure 6.9 and 6.10, M/via/rvia/0-1 (blue) denotes manufacturable via and redundant via on layer V_{0-1}, U/via/0-1 (red) denotes unmanufacturable via on layer V_{0-1}.

6.6 Summary

In this paper, we have concurrently considered dummy via insertion for the redundant via insertion and DSA guiding template assignment problem. We

constructed a conflict graph on grid model. The vertices in the conflict graph are *multiplet*s instead of guiding template candidates. This substitution can greatly reduce the size of the conflict graph. Based on the conflict graph, we modeled the problem as an ILP formulation, and relaxed it as an unconstrained nonlinear programming (UNP), and then developed a line search optimization algorithm to obtain a local optimal solution of the UNP. Since the algorithm is highly dependent on the initial solution, we used a good enough initial solution instead of random generation. Experimental results demonstrate that our solution expression and the proposed algorithm is effective and efficient.

Chapter 7 Analytical Mixed-Cell-Height Legalization Considering Average and Maximum Movement Minimization

7.1 Introduction

Modern circuit designs often contain (tens of) millions of standard cells located at placement sites on rows. To meet various design requirements such as low power and high performance, multi-deck cells occupying multi-rows (e.g., flip-flops) are often used in advanced technologies [8, 32]. Such multi-row height standard cells bring up challenging issues for placement, especially the mixed-cell-height legalization, due to the heterogenous cell structures and additional power-rail constraints, as pointed out in [26, 89].

In traditional single-row height standard-cell legalization, cell overlapping is independent among rows. In contrast, with multi-row height cells, shifting a cell in one row may cause cell overlaps in another row. The heterogenous cell structures could incur substantial global cell interferences among all cells in a circuit. Due to the global cell interference, existing single-row height standard-cell legalizers [16, 25, 28, 30, 50] cannot directly be extended to handle mixed-cell-height standard cells effectively. As a result, a mixed-cell-height legalization method needs to consider the heterogenous cell structures, with more global cell interferences and larger solution spaces. Moreover, the alignment of power (VDD) or ground (VSS) lines must be considered in mixed-cell-height standard-cell legalization. For single-row height standard-cell legalization, such VDD/VSS alignment can easily be handled by vertical cell flipping, e.g., the single-row height cell c_1 in Figure 7.1. However, the vertical cell flipping is invalid for an even-row height cell (the double-row cell c_2 in Figure 7.1). During legalization, therefore, each even-row height cell must be aligned to its correct row which

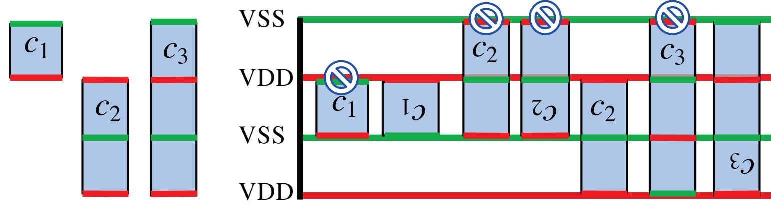


Figure 7.1: Example of the VDD/VSS alignment constraints.

meets the VDD/VSS constraint.

In addition, to preserve the quality of a given global placement, an ideal legalization method should minimize not only the average cell movement but also the maximum one [29], illustrated in Figure 7.2. In Figure 7.2(a), if we focus only on minimizing the average movement, we may obtain a legalization result as in Figure 7.2(b); in contrast, if we concurrently minimize the average and maximum cell movements, we can obtain a better legalization result as in Figure 7.2(c). Both of the results in Figures 7.2(b) and 7.2(c) have the same average cell movement, but the maximum cell movement of the result in Figure 7.2(b) is twice of that in Figure 7.2(c). In this chapter, we aim to minimize the average and maximum cell movements for mixed-cell-height legalization simultaneously.

Recent state-of-the-art works considered the mixed-cell-height standard-cell legalization problem [21, 26, 46, 84, 89]. Wu et al. [89] first investigated the standard-cell legalization with both of single- and double-row height cells. In [26], a multi-row local legalization algorithm was proposed to place cells in a local region. Wang et al. [84] extended Abacus to handle the legalization problem with mixed-cell-height standard cells. Hung et al. [46] proposed a flow-based method to spread cells and placed cells based on an integer linear program (ILP). With a modulus-based matrix splitting iteration method (MMSIM), Chen et al. [21] developed a near optimal legalization method to address mixed-cell-height standard-cell legalization. To guarantee the MMSIM convergence, the authors pointed out that the objective matrix should be symmetric positive definite, and the constraint matrix should be of full row rank. Nevertheless,

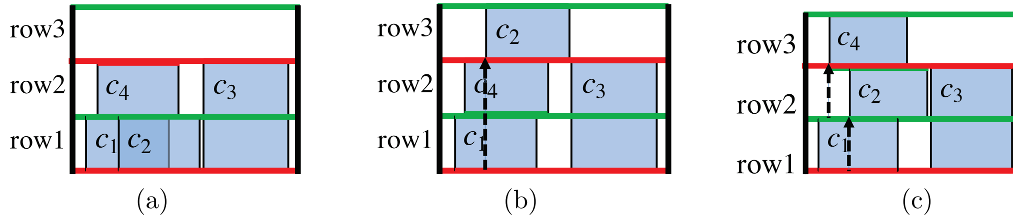


Figure 7.2: Comparisons on legalization with average cell movement and that with simultaneous average and maximum cell movements.

these legalization methods focus only on minimizing the average cell movement, and do not consider the maximum cell movement.

To minimize the maximum cell movement for single-row height standard-cell legalization, Darav et al. [28] proposed a flow-based legalization method by finding augmentation paths among bins. If the flow of a candidate augmentation path is larger than a pre-set value (named *maximum cell movement*), it would be pruned. Apparently, their method considers the maximum cell movement as a hard constraint rather than an objective. Further, in order to resolve cell overlapping, cells are moved from a dense bin to a sparse one along paths [28]. In single-row height standard-cell legalization, cell overlapping is independent among rows. With multi-row height cells, in contrast, shifting a cell in one row may cause cell overlaps in another row. What is worse, shifting a multi-row height cell in a bin to anther one may make cells illegal due a complex domino effect. Furthermore, in order to meet the VDD/VSS alignment constraints, it is much harder for the flow-based method to control cells movement. As a result, it is not easy to extend the flow-based method in [28] to handle the mixed-cell-height cell legalization problem.

In this chapter, we present an analytical mixed-cell-height standard-cell legalization algorithm to simultaneously minimize the average and the maximum cell movements. The major contributions of our work are summarized as follows:

- By analyzing and remodeling the objective function and constraints, we formulate the mixed-cell-height standard-cell legalization problem as a mixed

integer quadratic program (MIQP), which considers not only the average cell movement, but also the maximum cell movement, the sub-maximum movement, and the third maximum movement, etc.

- We convert the MIQP to a quadratic programming problem (QP). Unlike the work in [21] which minimizes only the average cell movement in the horizontal direction, we consider cells spreading continuously in both the horizontal and vertical directions.
- The QP is further reformulated as a linear complementarity problem (LCP), and solved by a modulus-based matrix splitting iteration method (MMSIM). The equivalence between the QP and the LCP is proved.
- We prove that the objective matrix of QP is symmetric positive definite and the constraint matrix of QP is of full row rank; therefore, the convergence of MMSIM is guaranteed.
- We propose a linear programming (LP) based method to further minimize the maximum cell movement in the horizontal direction.
- Experimental results demonstrate that our legalization model and method are effective for minimizing both the average and the maximum cell movements. Compared with the state-of-the-art work [21], for example, our algorithm reduces the average and maximum cell movements by 16% and 64%, respectively.

7.2 Problem Statement

The problem given are a global placement result with n' mixed-cell-height standard cells $C = \{c_1, c_2 \cdots, c_{n'}\}$ with initial bottom-left coordinates for all cells, and the height and width of a cell c_i denoted as (x_i^0, y_i^0), h_i and w_i, respectively. Suppose that each cell has a boundary power-rail type (VSS or VDD).

The chip is a rectangular sheet from $(0,0)$ to (W,H), where W and H are the chip width and height, and R_h and S_w denote the row height and placement site width, respectively. In this chapter, the mixed-cell-height standard-cell legalization problem aims at placing each cell to its best position, such that the average and maximum cell movements are minimized and the following constraints are satisfied:

1. cells should be aligned with correct VDD/VSSs;
2. cells should be non-overlapping;
3. cells should be inside the chip;
4. cells should be located at placement sites on rows.

Let (x_i, y_i) be the bottom-left coordinate of cell c_i, $i = 1, 2, \cdots, n'$. The problem of mixed-cell-height standard-cell legalization with simultaneous average and maximum movements minimization can be formulated as:

$$
\begin{aligned}
\min_{x,y} \quad & \frac{1}{n'} \sum_{c_i \in C} (|x_i - x_i^0| + |y_i - y_i^0|) + \omega \cdot \max_{c_i \in C} (|x_i - x_i^0| + |y_i - y_i^0|) && (7.1)\\
\text{s.t.} \quad & y_i = k_i' R_h, \ \forall c_i \in C, \\
& k_i' \in \begin{cases} \{0,1,2,\ldots\}, \ \text{if } c_i \text{ is odd-row height}, \\ \{0,2,4,\ldots\} \text{ or } \{1,3,5,\ldots\}, \ \text{o.w;} \end{cases} && (7.1a)\\
& x_i + w_i \le x_j, \ \forall c_i, c_j \in C, \\
& \text{if } c_i \text{ and } c_j \text{ are in the same row, and } x_i \le x_j; && (7.1b)\\
& 0 \le x_i, x_i + w_i \le W, \ 0 \le y_i, y_i + h_i \le H, \ \forall c_i \in C; && (7.1c)\\
& x_i = l_i S_w, l_i \in \{0,1,2,\ldots\}, \ \forall c_i \in C, && (7.1d)
\end{aligned}
$$

where ω is a user-defined parameter, used to weight the average and the maximum cell movements.

7.3 Problem Reformulations

In this section, we first formulate the mixed-cell-height legalization problem as a mixed integer quadratic programming problem (MIQP), then we convert the MIQP to a quadratic programming problem (QP), and further we reformulate the QP as a linear complementarity problem (LCP).

7.3.1 Mixed Integer Quadratic Programming

The objective function of Problem (7.1) is the weighted sum of the average cell movement and the maximum cell movement. We transform the objective as:

$$\min_{x,y} \sum_{c_i \in C} \frac{\alpha_i}{2}((x_i - x_i^0)^2 + (y_i - y_i^0)^2), \tag{7.2}$$

where α_i can be seen as a weight on the movement of cell c_i. In fact, Objective (7.2) includes minimizing the maximum cell movement if α_i is assigned a proper value, $i = 1, 2, \cdots, n'$.

Naturally, it would be better that a legalizer can not only reduce the maximum cell movement, but also the sub-maximum movement, the third maximum movement, etc. For example, if a cell c_i overlaps with a macro cell, in order to resolve the overlap, c_i must be moved out of the macro cell, then the maximum cell movement is likely generated. If the generated maximum cell movement is much more than the other cell movements, then minimizing the maximum cell movement is meaningless to all the rest cells. Hence, we control the above scenario by re-assigning weight α_i to the movement of each cell c_i. An intuitive weight setting rule is that, if the movement of c_i is larger than the movement of c_j, then α_i should be larger than α_j for reducing the movement of c_i. In this chapter, α_i is set as:

$$\alpha_i = \left(\frac{(x_i - x_i^0)^2 + (y_i - y_i^0)^2}{\frac{1}{n'} \sum_{c_j \in C}(|x_j - x_j^0| + |y_j - y_j^0|)} \right)^{\kappa}, \tag{7.3}$$

where $\kappa \geq -1$ and is relative to the value of ω in (7.1). The parameter κ is used to make a trade-off between the average cell movement and the maximum cell movement. If $\kappa \gg 1$, it focuses on minimizing the maximum cell movement; If $\kappa = -1$, it focuses on minimizing the average cell movement. In Equation (7.3), x_i and y_i are set as the latest iteration results (coordinates) of cell c_i in our algorithm (described in Subsection 7.4).

Next, we give detailed analysis and handling of constraints.

First, for Constraint (7.1a), cells should be aligned with correct rows to meet the VDD/VSS constraints. In order to obtain a high-quality legalization solution with the least vertical movement, a trivial but effective operation is moving each cell to the nearest VDD/VSS. Ideally, after aligning to the nearest correct row, the y-coordinate of each cell c_i is updated to y_i' from y_i^0. If the distribution of cells in a row is locally sparse, then the overlapsamong these cells in this row can be desirably solved; conversely, if the distribution of cells in a row is locally dense, then the locations of some cells in this row may not be desirable. They should be assigned to other rows for which Constraint (7.4) still should be satisfied:

$$y_i \in \{y_i' - k_i^l R_h, y_i' + k_i^u R_h\}, \ \forall c_i \in C. \tag{7.4}$$

In the above equation, if c_i is odd-row height, then $k_i^l \in \{0, 1, 2, \ldots, \lfloor \frac{y_i'}{R_h} \rfloor\}$ and $k_i^u \in \{0, 1, 2, \ldots, \lfloor \frac{H - y_i'}{R_h} \rfloor\}$; if c_i is even-row height, then $k_i^l \in \{0, 2, 4, \ldots, \lfloor \frac{y_i'}{R_h} \rfloor\}$ and $k_i^u \in \{0, 2, 4, \ldots, \lfloor \frac{H - y_i'}{R_h} \rfloor\}$. For example, in Figure 7.3, after aligning to the nearest correct row, the distribution of cells in row 1 is too dense to resolve overlaps. As a result, the cell c_2 should be aligned to row 2.

Second, for Constraint (7.1b), all cells in the same row are sorted by their initial bottom-left x^0, i.e., $x_i^0 \leq x_j^0$, if cell c_i is on the left of cell c_j according to the global placement result. Furthermore, if cells c_i and c_j are totally in different rows, then the position constraint between x_i and x_j is free. Hence, constraint

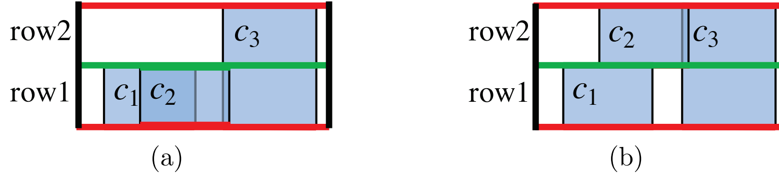


Figure 7.3: Cells are aligned to the nearest correct rows.

(7.1b) can be reformulated as:

$$x_i + w_i \leq x_j + M(1 - z_{ij}),\ z_{ij} \in \{0, 1\},\ \text{if } x_i^0 \leq x_j^0, \tag{7.5}$$

where M is a large enough number, i.e., chip width W. If cells c_i and c_j are adjacent in the same row, then $z_{ij} = 1$; otherwise, $z_{ij} = 0$.

Third, for Constraint (7.1c), for each cell c_i, $0 \leq y_i, y_i + h_i \leq H$ is satisfied under Constraint (7.1a). Thus, it can be removed. In addition, since we minimize Objective (7.2), the horizontal moving of each cell would be not far away from its original position, and there are few cells out of the boundary. We skip the right boundary constraint temporarily, and then Constraint (7.1c) is changed as:

$$x_i \geq 0,\ \forall c_i \in C. \tag{7.6}$$

Fourth, in Constraint (7.1d), if all the cells are placed at their best positions in rows, then we only need to shift each cell to their nearest placement sites. We will handle Constraint (7.1d) in Section 7.4.

Overall, Objective (7.2) and Constraints (7.4), (7.5), and (7.6) compose a mixed integer quadratic programming problem (MIQP).

7.3.2 Quadratic Programming

For the MIQP in Subsection 7.3.1, variable y_i in Constraint (7.4) and variable z_{ij} in Constraint (7.5) are integral. In addition, in order to obtain a better result, all cells in a circuit should be considered together instead of row-by-row. Consequently, the MIQP is seriously hard to solve for large-scale circuits.

In this subsection, we relax Constraints (7.4) and (7.5) to linear constraints. Correspondingly, the MIQP is relaxed to a quadratic programming problem.

In [9], Bai proposed a modulus-based matrix splitting iteration method (MMSIM), which is very efficient for solving linear complementarity problems. In addition, Chen et al. [21] modified MMSIM to the QP legalization problem. In order to use the effective and efficient MMSIM, as [21], we split all multi-row height cells into single-row height cells for the MMSIM solver. Then the mixed-cell-height standard cells $C = \{c_1, c_2 \cdots, c_{n'}\}$ are split as single-row sub-cells $SC = \{sc_1, sc_2 \cdots, sc_n\}$. These single-row height sub-cells should satisfy:

$$\begin{aligned} x_{i1} = x_{i2} = \cdots = x_{ir_i}, \\ y_{i1} + (r_i - 1)R_h = y_{i2} + (r_i - 2)R_h = \cdots = y_{ir_i}. \end{aligned} \tag{7.7}$$

where sub-cells sc_{i1}, sc_{i2}, ..., sc_{ir_i} are split from a r_i-row height cell, $r_i = h_i/R_h$.

After cell splitting, Objective (7.2) is transformed to

$$\min_{x,y} \sum_{sc_i \in SC} \frac{\alpha_i}{2r_i}((x_i - x_i^0)^2 + (y_i - y_i^0)^2). \tag{7.8}$$

Since an r_i-row height cell is split into r_i single-row height sub-cells and the movement of the r_i-row cell is counted r_i times, we divide the objective function by r_i for each cell in (7.8).

Next, we relax Constraints (7.4) and (7.5) to linear constraints. Actually, for each sub-cell sc_i, after aligning it to its nearest correct row, i.e., $y_i = y_i'$, the movement of sub-cell sc_i in the vertical direction is minimized. Since sub-cells may be re-assigned to other rows for achieving better legalization solution, we relax the range y_i of sub-cell sc_i from integers to continuous real numbers. That is

$$y_i \in [y_i' - k_i^l R_h, y_i' + k_i^u R_h], \ \forall sc_i \in SC, \tag{7.9}$$

where $k_i^l \in \{0, 1, 2, \ldots, \lfloor \frac{y_i'}{R_h} \rfloor\}$ and $k_i^u \in \{0, 1, 2, \ldots, \lfloor \frac{H - y_i'}{R_h} \rfloor\}$.

Since the moving range $y_i \in [y_i' - k_i^l R_h, y_i' + k_i^u R_h]$ of sub-cell sc_i is excessive,

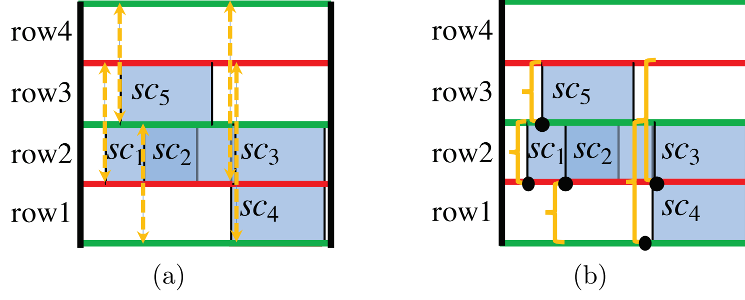


Figure 7.4: (a). Vertical moving intervals of sub-cells. (b). The moving ranges of bottom-left y-coordinate of sub-cells.

a speed-up technique is restricting the moving orientation o_i in vertical (upward or downward) of each sub-cell sc_i. Then, Constraint (7.9) of sc_i is limited as:

$$y_i \in \begin{cases} \left[y_i' - k_i^l R_h, y_i'\right], & \text{if } o_i \text{ is downward;} \\ \left[y_i', y_i' + k_i^u R_h\right], & \text{if } o_i \text{ is upward,} \end{cases} \tag{7.10}$$

where k_i^l and k_i^u are used to control the range of y_i. Correspondingly, the vertical moving interval VMI_i of sub-cell sc_i is $[y_i' - k_i^l R_h, y_i' + R_h]$ or $[y_i', y_i' + k_i^u R_h]$. The setting of sub-cell orientation is listed in detail in Algorithm 7.1 (Section 7.4). For example, as shown in Figure 7.4(a), if the orientations of sub-cells sc_1, sc_2, sc_3, sc_4 and sc_5 are upward, downward, upward, upward, and upward, respectively, and $k_1^u = k_2^l = k_5^u = 1$ and $k_3^u = k_4^u = 2$, then the VMI_i of each sub-cell is marked by double headed arrow line in Figure 7.4(a), and the corresponding range of y_i for sub-cell sc_i is marked by brace in Figure 7.4(b).

Next, the mixed integer Constraint (7.5) would be relaxed to a linear constraint. According to Constraint (7.5), if $x_i^0 \leq x_j^0$, and sc_i and sc_j are adjacent, then $x_i + w_i \leq x_j + M(1 - z_{ij})$, where $z_{ij} \in \{0, 1\}$ denotes whether two adjacent sub-cells sc_i and sc_j are in the same row. Since all the sub-cells have the same height (i.e., $h_i = R_h$), $z_{ij} R_h$ reflects the overlapping length of sub-cells sc_i and sc_j in the vertical direction. If $z_{ij} = 1$, i.e., $|y_i - y_j| = 0$, then the vertical overlapping length is R_h; if $z_{ij} = 0$, i.e., $|y_i - y_j| \geq R_h$, then the vertical overlapping length is 0. The vertical overlapping length of sub-cells sc_i and sc_j with $z_{ij} R_h$

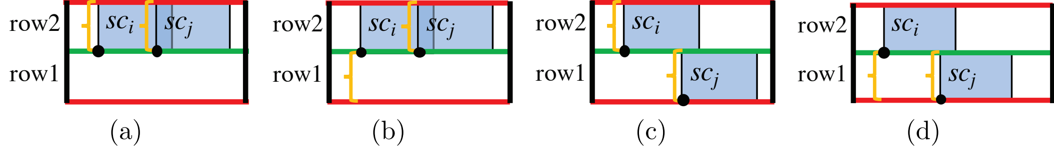


Figure 7.5: Four possible structures for two adjacent sub-cells i and j with $VMI_i \cap VMI_j \neq \emptyset$.

$= \max\{R_h - |y_i - y_j|, 0\}$. Then, for two adjacent sub-cells sc_i and sc_j, we have

$$z_{ij} = \max\left\{1 - \frac{|y_i - y_j|}{R_h}, 0\right\}, \text{ and} \tag{7.11}$$

$$x_i + w_i \leq x_j + M \cdot \min\left\{\frac{|y_i - y_j|}{R_h}, 1\right\}, \text{ if } x_i^0 \leq x_j^0. \tag{7.12}$$

By Equation (7.11), the integer variable z_{ij} is relaxed to a continuous variable. In addition, since the VMI_i of each sub-cell sc_i can be pre-calculated, if $VMI_i \cap VMI_j = \emptyset$, then we do not need to consider Constraint (7.12). For example, for sub-cells sc_2 and sc_5 in Figure 7.4(a), $VMI_2 \cap VMI_5 = \emptyset$. Otherwise, if $VMI_i \cap VMI_j \neq \emptyset$ for sub-cells sc_i and sc_j, then we would check wether sc_i and sc_j are adjacent.

Observing Constraint (7.12), it can be seen that this constraint is still hard to handle due to the absolute value function and the minimum value function. It should be further transformed to a linear constraint by eliminating the absolute and the minimum value functions.

For two adjacent sub-cells sc_i and sc_j, where $VMI_i \cap VMI_j \neq \emptyset$, there exist four possible cases: (1) sc_i and sc_j are in the same row and have the same orientation, as in Figure 7.5(a); (2) sc_i and sc_j are in the same row and have different orientations, as in Figure 7.5(b); (3) sc_i and sc_j are in different rows and have the same orientation, as in Figure 7.5(c); (4) sc_i and sc_j are in different rows and have different orientations, as in Figure 7.5(d).

For cases 2) and 3), the relationship between y_i and y_j is known. If $y_i \leq y_j$, then $|y_i - y_j| = y_j - y_i$; if $y_i > y_j$, then $|y_i - y_j| = y_i - y_j$. However, for cases

1) and 4), we do not know which is larger between y_i and y_j. For the two cases, we use the initial y-coordinates to determine the value of $|y_i - y_j|$. That is, if $y_i^0 \le y_j^0$, then $|y_i - y_j| := y_j - y_i$; otherwise, $|y_i - y_j| := y_i - y_j$. We introduce a notation $\ominus$ to denote the operator between y_i and y_j, and let

$$y_i \ominus y_j = \begin{cases} y_i - y_j, & \text{if } y_i \ge y_j \text{ for cases 2) and 3),} \\ & \text{and if } y_i^0 \ge y_j^0 \text{ for cases 1) and 4);} \\ y_j - y_i, & \text{if } y_i < y_j \text{ for cases 2) and 3),} \\ & \text{and if } y_i^0 < y_j^0 \text{ for cases 1) and 4).} \end{cases} \tag{7.13}$$

Furthermore, since the maximum value of $\frac{y_i \ominus y_j}{R_h}$ is 2, we have $\min\{\frac{y_i \ominus y_j}{2R_h}, 1\} = \frac{y_i \ominus y_j}{2R_h}$. After these transformations, for two adjacent sub-cells sc_i and sc_j, Constraint (7.12) is reduced to

$$x_i + w_i \le x_j + M \cdot \frac{y_i \ominus y_j}{2R_h}, \text{ if } x_i^0 \le x_j^0. \tag{7.14}$$

Finally, in Constraint (7.14), M is large and $\frac{y_i \ominus y_j}{2R_h}$ is a real number instead of an integer. If $y_i \ominus y_j > 0$, then $M\frac{y_i \ominus y_j}{2R_h}$ may be too large due to M, and then Constraint (7.14) may be invalid for resolving the horizontal overlap between sub-cells sc_i and sc_j. In addition, with Objective (7.2), the x_i should not be far away from x_i^0. Hence, we set M as

$$M = \beta_{ij} \cdot (w_i + w_j), \tag{7.15}$$

where β_{ij} is a user-defined parameter, which is used to control the value of M.

Thus far, the mixed-cell-height standard-cell legalization problem is reformulated to a quadratic programming problem (QP):

$$\min_{x,y} \quad \sum_{sc_i \in SC} \frac{\alpha_i}{2r_i}((x_i - x_i^0)^2 + (y_i - y_i^0)^2) \tag{7.16}$$

$$\text{s.t.} \quad x_i + w_i \leq x_j + \frac{\beta_{ij} \cdot (w_i + w_j)}{2R_h} \cdot (y_i \ominus y_j), \ \forall sc_i, sc_j \in SC,$$

$$\text{if } VMI_i \cap VMI_j \neq \emptyset, \ sc_i, \ sc_j \text{ are adjacent, and } x_i^0 \leq x_j^0; \tag{7.16a}$$

$$y_i \in \begin{cases} \left[y_i' - k_i^l R_h, y_i'\right], & \text{if } o_i \text{ is downward,} \\ \left[y_i', y_i' + k_i^u R_h\right], & \text{if } o_i \text{ is upward,} \end{cases}$$

$$x_i \geq 0, \ \forall sc_i \in SC; \tag{7.16b}$$

$$y_{i1} + (r_i - 1)R_h = y_{i2} + (r_i - 2)R_h = \cdots = y_{ir_i},$$

$$x_{i1} = x_{i2} = \cdots = x_{ir_i}, \ \ sc_{i1}, \cdots sc_{ir_i} \text{are from the same cell.} \tag{7.16c}$$

7.3.3 Linear Complementarity Problem

Since it is generally time consuming to solve a large-scale quadratic programming problem with many inequality constraints, we convert the QP (7.16) equivalently into a LCP, and solve the LCP by the modulus-based matrix splitting iteration method (MMSIM) [21]. To guarantee the convergence of MMSIM, it requires that the objective matrix is symmetric positive definite and the constraint matrix is of full row rank.

Let $x = (x_1, x_2, \cdots, x_n)^T$, and $y = (y_1, y_2, \cdots, y_n)^T$, and let $\mu = (\mu_i)_{2n} = \begin{bmatrix} x \\ y \end{bmatrix}$. Problem (7.16) can be rewritten as:

$$\min_{\mu} \quad \frac{1}{2}\mu^T Q \mu + p^T \mu \tag{7.17}$$

$$\text{s.t.} \quad A\mu \geq b; \tag{7.17a}$$

$$d \leq y \leq d^u; \tag{7.17b}$$

$$x \geq 0; \tag{7.17c}$$

$$E\mu = f, \tag{7.17d}$$

where Q is a diagonal matrix with its elements $q_{i,i} = q_{n+i,n+i} = \frac{\alpha_i}{r_i}$, $i = 1, 2, \cdots, n$; p is a vector with $p_i = -\frac{\alpha_i x_i^0}{r_i}$, $i = 1, 2, \cdots, n$; $p_i = -\frac{\alpha_i y_i^0}{r_{i-n}}$, $i = n+1, n+2, \cdots, 2n$; and A is the overlap constraint matrix with only four nonzero elements 1, -1, $\frac{\beta_{ij}(w_i+w_j)}{2R_h}$ and $-\frac{\beta_{ij}(w_i+w_j)}{2R_h}$ in each row, respectively.

In order to obtain the uniform boundary constraint $\overline{\mu} \geq 0$, let $\overline{y} = y - d$, and $\overline{\mu} = (\overline{\mu}_i)_{2n} = \begin{bmatrix} x \\ \overline{y} \end{bmatrix}$. Then we have

$$\min_{\overline{\mu}} \quad \frac{1}{2}\overline{\mu}^T Q \overline{\mu} + \overline{p}^T \overline{\mu} \tag{7.18}$$

$$\text{s.t.} \quad A\overline{\mu} \geq \overline{b}; \tag{7.18a}$$

$$-I\overline{y} \geq d - d^u; \tag{7.18b}$$

$$\overline{\mu} \geq 0; \tag{7.18c}$$

$$E\overline{\mu} = \overline{f}, \tag{7.18d}$$

where I is an identity matrix, and vectors $\overline{p}$, $\overline{b}$, $\overline{f}$ are correspondingly transformed from p, b, f.

To guarantee that the constraint matrix is of full row rank, we increase the number of variables by duplicating $\overline{y}$ to $\underline{y}$ with $\underline{y} = \overline{y}$. Then Constraint (7.18b) is replaced by $-I\underline{y} \geq d - d^u$, and $\underline{y} - \overline{y} = 0$. Let $E' \begin{bmatrix} \overline{y} \\ \underline{y} \end{bmatrix} = 0$ denote $\underline{y} - \overline{y} = 0$. In this way, Constraint (7.18a) and $-I\underline{y} \geq d - d^u$ compose a new system of inequality constraints, and Constraint (7.18d) and $\underline{y} - \overline{y} = 0$ compose the other new system of equality constraints.

Let $\widetilde{\mu} = \begin{bmatrix} x \\ \overline{y} \\ \underline{y} \end{bmatrix}$, $\widetilde{Q} = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix}$, $\widetilde{p} = \begin{bmatrix} \overline{p} \\ 0 \end{bmatrix}$, $\widetilde{A} = \begin{bmatrix} A & 0 \\ 0 & -I \end{bmatrix}$, $\widetilde{b} = \begin{bmatrix} \bar{b} \\ d - d^u \end{bmatrix}$, $\widetilde{E} = \begin{bmatrix} E \\ E' \end{bmatrix}$, $\widetilde{f} = \begin{bmatrix} \overline{f} \\ 0 \end{bmatrix}$. Then we describe the standard form of the quadratic programming problem as follows:

$$\min_{\widetilde{\mu}} \quad \frac{1}{2}\widetilde{\mu}^T\widetilde{Q}\widetilde{\mu} + \widetilde{p}^T\widetilde{\mu} \tag{7.19}$$

$$\text{s.t.} \quad \widetilde{A}\widetilde{\mu} \geq \widetilde{b}; \tag{7.19a}$$

$$\widetilde{E}\widetilde{\mu} = \widetilde{f}; \tag{7.19b}$$

$$\widetilde{\mu} \geq 0. \tag{7.19c}$$

For Problem (7.19), we relax Constraint (7.19b) by putting it into the objective with a penalty parameter $\lambda > 0$. And a new QP is formulated as:

$$\min_{\widetilde{\mu}} \quad \frac{1}{2}\widetilde{\mu}^T(\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E})\widetilde{\mu} + (\widetilde{p}^T - \lambda\widetilde{f}^T\widetilde{E})\widetilde{\mu} \tag{7.20}$$

$$\text{s.t.} \quad \widetilde{A}\widetilde{\mu} \geq \widetilde{b}; \tag{7.20a}$$

$$\widetilde{\mu} \geq 0. \tag{7.20b}$$

Proposition 7.3.1. In Problem (7.20), $\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E}$ is a symmetric positive definite matrix.

Proof. First, $\widetilde{Q}$ is a diagonal matrix with all elements not less than 0, and $\widetilde{E}^T\widetilde{E}$ is a symmetric positive semi-definite matrix. Thus $\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E}$ is a symmetric semi-definite positive matrix. Furthermore, according to the structure of $\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E}$, there exist a series of elementary matrixes $P_1, P_2, \cdots P_s$ such that $P_s^T(\cdots(P_2^T(P_1^T(\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E})P_1)P_2)\cdots)P_s = \mathit{\Lambda}$, where $\mathit{\Lambda}$ is a diagonal matrix with all nonzero elements positive. Let $P = P_1P_2\cdots P_s$, and P is an invertible matrix. Then, $\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E}$ is a congruent matrix of $\mathit{\Lambda}$. Since $\mathit{\Lambda}$ is a positive definite matrix, and by the proposition of congruent transformation, $\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E}$ is positive definite. □

Proposition 7.3.2. In Problem (7.20), if $k_i^l = k_i^u = 1$ for all sub-cells, then matrix $\widetilde{A}$ is of full row rank.

Proof. $\widetilde{A}$ is a block diagonal matrix, which is composed of A and $-I$. Since I is an identity matrix, $-I$ is of full row rank.

Next, we prove matrix A is of full row rank. First, the number of variables of μ is $2n$, which is equal to the number of columns of A. Suppose A is an $m \times 2n$ matrix. On the one hand, according to Constraint (7.16a), it can be found that every row of A has an element with value 1. On the other hand, if $k_i^l = k_i^u = 1$ for all sub-cells, then all sub-cells are moved only between two rows. And, if a sub-cell sc_i can be moved only between two rows, then there are at most two sub-cells next to sc_i and on its left. In other words, at most two elements with values 1 in the i-th column of matrix A in constraint (7.17a). This claim holds for every column. Thus, there are at most $2n$ rows in A, i.e., $m \leq 2n$. In addition, there are only four nonzero elements in each row, and at most four nonzero elements in each column. We can choose the columns with two elements 1 to form an m-order matrix. By fixing the orders of sub-cells in the x-direction and y-direction respectively, and by elementary transformations, this m-order matrix can be further transformed to an upper triangular matrix, in which all diagonal elements are positive. Thus, A is of full row rank. And further, $\widetilde{A}$ is of full row rank. □

In this chapter, k_i^l and k_i^u are set as 1 for each sub-cell. By Propositions (7.3.1) and (7.3.2), $\widetilde{\mu}$ is the global minimal solution of Problem (7.20) if and only if there exist vectors r, u, $v \geq 0$ such that the quadruple $(\widetilde{\mu}, r, u, v)$ satisfies the following KKT conditions [69]:

$$LCP(B,t): \quad w = Bz + t \geq 0,\ z \geq 0,\ \text{and}\ z^T w = 0, \tag{7.21}$$

where $w = \begin{bmatrix} u \\ v \end{bmatrix}$, $B = \begin{bmatrix} \widetilde{Q} + \lambda \widetilde{E}^T \widetilde{E} & -\widetilde{A}^T \\ \widetilde{A} & 0 \end{bmatrix}$, $t = \begin{bmatrix} (\widetilde{p} - \lambda \widetilde{f}^T \widetilde{E}) \\ -\widetilde{b} \end{bmatrix}$, and $z = \begin{bmatrix} \widetilde{\mu} \\ r \end{bmatrix}$. Problem (7.21) is a linear complementarity problem (LCP). According to Theorem 1 of [21], we obtain the following theorem:

Theorem 7.3.3. The solution of $LCP(B,t)$ (7.21) gives the optimal solution of QP (7.20), and vice versa.

Previous works [9,21] used a modulus-based matrix splitting iteration method (MMSIM) to solve the $LCP(B,t)$, which is efficient. And, if matrix B is positive definite, the global convergence of MMSIM for $LCP(B,t)$ holds. However, since $\widetilde{Q}+\lambda\widetilde{E}^T\widetilde{E}$ in $LCP(B,t)$ (7.21) is positive definite, it is easy to check that B is a positive semi-definite asymmetric matrix. Note that the non-singularity of B in $LCP(B,t)$ (7.21) is not guaranteed, and further the global convergence of MMSIM for $LCP(B,t)$ is also not guaranteed. In this work, $LCP(B,t)$ (7.21) is resolved by an asymptotic modulus-based approach. Before that, the positive semi-definite matrix B in $LCP(B,t)$ (7.21) is approximated by a positive definite matrix $B(\varepsilon)=B+\varepsilon I$, where I is an identify matrix and ε is a small constant. Then $LCP(B(\varepsilon),t)$ is closely linked to the unperturbed $LCP(B,t)$. By using MMSIM to solve $LCP(B(\varepsilon),t)$, we can obtain a solution close to an optimal solution of $LCP(B,t)$. The gap between the optimal values of $LCP(B,t)$ and $LCP(B(\varepsilon),t)$ is dependent on ε.

7.4 Our Legalization Framework

In this chapter, we simultaneously minimize the average and maximum movements by proposing a horizontal- and vertical-direction legalization method for mixed-cell-height standard-cell legalization. Our framework is summarized in Figure 7.6. There are three major steps in the framework: 1) preprocessing; 2) horizontal- and vertical-direction legalization; and 3) horizontal-direction legalization. The details are stated in the following.

In the preprocessing step, all cells are firstly aligned to the nearest correct rows (meet the VDD/VSS alignment constraints) by shifting cells in the vertical direction. In addition, each cell is split into several single-row height sub-cells. And in order to decide the vertical moving interval VMI_i of y_i for each sub-cell,

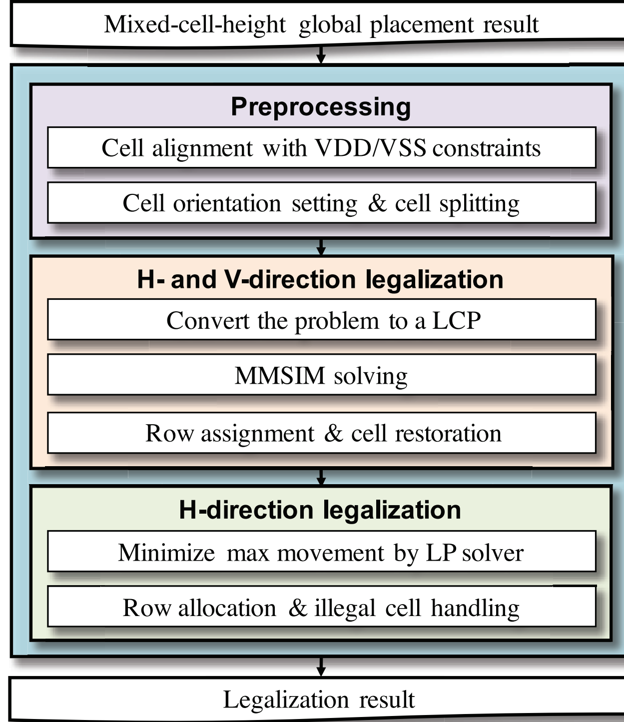


Figure 7.6: Our legalization framework.

an orientation setting algorithm is proposed in Algorithm 7.1. In Algorithm 7.1, NR is the number of rows in a circuit, and R_i is the number of sub-cells in row i. $o_{i,j}$ is the orientation of the j-th sub-cell $sc_{i,j}$ in row i. If $o_{i,j} = 1$, then the orientation of sub-cell $sc_{i,j}$ is upward, and downward otherwise. $w_{i,j}$, $y^0_{i,j}$ are the width and initial y-coordinate of sub-cell $sc_{i,j}$, respectively. In lines 2-4, the ratio of the number of upward cells r^u to the number of downward cells $1 - r^u$ are calculated. Lines 7-17 are used to decide the orientation $o_{i,j}$ of the j-th sub-cell in row R_i.

In the horizontal- and vertical-direction legalization step, we first split cells into sub-cells. Then as described in Section 7.4, we model the mixed-cell-height standard-cell legalization problem to $LCP(B, t)$ (7.21), which simultaneously considers cell moving in both of the horizontal and vertical directions. Then, we apply the MMSIM [21] to approximately solve $LCP(B, t)$. Our MMSIM solution algorithm is depicted in Algorithm 7.2. In Algorithm 7.2, the MMSIM

Algorithm 7.1 Sub-cell orientation setting

Input: a circuit with all sub-cells aligned to the correct rows;
Output: orientations o of all sub-cells in the vertical direction;
1: **for** $i = 0 : NR$ **do**
2: calculate the average density ad_i^u of densities of rows $i+1$ to NR;
3: and the average density ad_i^d of densities of rows 0 to $i-1$;
4: r^u =max$\{0$,min$\{1, 0.5 + ad_i^u - ad_i^d\}\}$;
5: $W^u = w_{i,0}$, $o_{i,0} = 1$, $W^d = w_{i,1}$, $o_{i,1} = 0$;
6: **for** $j = 1 : R_i$ **do**
7: **if** $\frac{W^u}{W^d} \leq \frac{r^u}{1-r^u}$ **then**
8: $W^u = W^u + w_{i,j}$, $o_{i,j} = 1$;
9: **else**
10: $W^d = W^d + w_{i,j}$, $o_{i,j} = 0$;
11: **end if**
12: **if** $(1 - o_{i,j-1})o_{i,j} > 0$ and $y^0_{i,j-1} > y^0_{i,j}$ **then**
13: $o_{i,j-1} = 1$, $o_{i,j} = 0$;
14: **end if**
15: **if** $o_{i,j-1}(1 - o_{i,j}) > 0$ and $y^0_{i,j-1} < y^0_{i,j}$ **then**
16: $o_{i,j-1} = 0$, $o_{i,j} = 1$;
17: **end if**
18: **end for**
19: **end for**
20: **Return** o.

iterations are stated in lines 1-8, and each sub-cell is aligned to a correct row in lines 9-21. In line 4 of Algorithm 7.2, we choose the splitting matrices $M(\varepsilon)$ and N with $B(\varepsilon) = M(\varepsilon) - N$ as

$$\begin{aligned} M &= \begin{bmatrix} \frac{1}{\beta^*}(\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E}) & 0 \\ \widetilde{A} & \frac{1}{\theta^*}D \end{bmatrix}, \quad M(\varepsilon) = M + \varepsilon I, \\ N &= \begin{bmatrix} (\frac{1}{\beta^*} - 1)(\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E}) & -\widetilde{A}^T \\ 0 & \frac{1}{\theta^*}D \end{bmatrix}, \end{aligned} \tag{7.22}$$

where $D = tridiag(\widetilde{A}(\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E})^{-1}\widetilde{A}^T)$, in which $(\widetilde{Q} + \lambda\widetilde{E}^T\widetilde{E})^{-1}$ can be fast calculated by the trick in [21], and β^*, θ^* are two positive constants as in [21]. After Algorithm 7.2, all multi-row height cells are restored. According to [21], we also can obtain the following theorem:

Theorem 7.4.1. The iteration sequence $\{z^{(l)}\}_{k=0}^{+\infty} \subset \mathbb{R}^n_+$ generated by Algorithm

Algorithm 7.2 Horizontal- and vertical-direction legalization

Input: matrices: $M(\varepsilon)$, N, $B(\varepsilon)$; vectors: t, $s^{(0)}$, $z^{(0)}$; parameters: γ, ε, σ, κ;
Output: horizontal- and vertical-direction legalization result;
1: **repeat**
2: $x_i^{(l)} = z_i^{(l)}$, $y_i^{(l)} = z_{n+i}^{(l)} + d_i$, $i = 1, 2, \cdots, n$;
3: $\alpha_i = \left(\frac{(x_i^{(l)} - x_i^0)^2 + (y_i^{(l)} - y_i^0)^2}{\frac{1}{n}\sum_{j=1}^{n}(|x_j^{(l)} - x_j^0| + |y_j^{(l)} - y_j^0|)} \right)^{\kappa}$;
4: solve $(M(\varepsilon) + I)s^{(l+1)} = Ns^{(l)} + (I - B(\varepsilon))|s^{(l)}| - \gamma t$;
5: $z^{(l+1)} = \frac{1}{\gamma}(|s^{(l+1)}| + s^{(l+1)})$;
6: $l{+}{+}$;
7: **until** $|z^{(l)} - z^{(l-1)}| < \varepsilon$
8: obtain the coordinate (x, y) of all sub-cells by $z^{(l)}$;
9: $i = 0$;
10: **repeat**
11: **if** $\frac{|y_i - y_i'|}{R_h} < \sigma$ **then**
12: $y_i = y_i'$;
13: **end if**
14: **if** $\frac{y_i - (y_i' - kR_h)}{R_h} < 1 - \sigma$ **then**
15: $y_i = y_i' - kR_h$;
16: **end if**
17: **if** $\frac{(y_i' + kR_h) - y_i}{R_h} < 1 - \sigma$ **then**
18: $y_i = y_i' + kR_h$;
19: **end if**
20: $i{+}{+}$;
21: **until** $i \geq n$
22: **Return** x, y.

7.2 converges to the unique solution $z^* \in \mathbb{R}_+^n$ of $LCP(B(\varepsilon), t)$ for any initial vector $s^{(0)} \in \mathbb{R}^n$.

After the horizontal- and vertical-direction legalization, there still exist some overlaps due to row assignment by rounding. In addition, each cell must be aligned to the placement site. In the horizontal-direction legalization step, we first align each cell to its nearest placement site. After that, if cell c_i does not overlap with other cells, then the location of c_i will be fixed; otherwise, to reduce the cells overlaps, a linear programming based method is used to minimize the cell movement.

Given a circuit with all cells realigned to the correct rows and to the place-

ment sites, the coordinate of cell c_i is (x_i'', y_i''). Let OC be the set of cells that still overlap with other cells. For the cells in OC, we consider the following problem:

$$
\begin{aligned}
\min_{x} \quad & \max_{c_i \in OC} |x_i - x_i''| && (7.23)\\
\text{s.t.} \quad & x_i + w_i \le x_j,\ \forall c_i, c_j \in OC,\ \text{if } c_i,\ c_j \text{ are adjacent in a row,} \\
& \text{and } c_i \text{ is on the left of } c_j; && (7.23a)\\
& x_r \le W - w_r,\ \forall c_r \in OC; && (7.23b)\\
& x_i \ge 0,\ \forall c_i \in OC, && (7.23c)
\end{aligned}
$$

where c_r is the most right cell in every row. Problem (7.23) can be reformulated as the following linear programming (LP):

$$
\begin{aligned}
\min_{x} \quad & x_{max} && (7.24)\\
\text{s.t.} \quad & x_i - x_j \le -w_i,\ \forall c_i, c_j \in OC,\ \text{if } c_i,\ c_j \text{ are adjacent in a row,} \\
& \text{and } c_i \text{ is on the left of } c_j; && (7.24a)\\
& x_r \le W - w_r,\ \forall c_r \in OC; && (7.24b)\\
& x_i - x_{max} \le x_i'',\ \forall c_i \in OC; && (7.24c)\\
& -x_i - x_{max} \le -x_i'',\ \forall c_i \in OC; && (7.24d)\\
& x_i, x_{max} \ge 0,\ \forall c_i \in OC. && (7.24e)
\end{aligned}
$$

Before solving the LP (7.24), many cells are overlapping free, hence the number of variables in (7.24) would not be too many. Furthermore, from a good initial solution, it only needs to perform LP solver in a few iterations. After solving the LP (7.24), each cell would be aligned to the nearest placement site. Meanwhile, a cell is marked as an illegal cell if this cell is overlapped with other cells or out of the right boundary. For every illegal cell, we search all

possible blank spaces. Then, a bipartite graph of illegal cells and blank spaces is constructed, and the Kuhn-Munkres algorithm [68] is applied to find a best matching for illegal cells and blank spaces, which runs in $O(n_{ill}^3)$, where n_{ill} is the number of illegal cells. After the above operations, all cells are placed in the chip region legally.

7.5 Experimental Results

We implemented our analytical mixed-cell-height legalization algorithm in the C++ programming language. To evaluate the effectiveness of our proposed algorithm, we compared with the method in DAC'17 [21] and the algorithm of the first-place team of the ICCAD-2017 CAD Contest [29], namely, DAC'17 and Top1, respectively. The tested benchmarks are modified from the ICCAD-2017 CAD Contest on Multi-Deck Standard-Cell Legalization [29] by omitting the fence-region constraints and the soft constraints. In our algorithm, the parameter λ in Problem (7.20) was set as 500, β^* and θ^* in the splitting matrices M and N were both set as 0.5. γ, σ, and κ in Algorithm 7.2 were set as 1, 0.4 and 1, respectively. After the preprocessing step, if sub-cells sc_i and sc_i are in the same row, then $\beta_{ij} = \frac{1}{2}$; otherwise, $\beta_{ij} = 2$. With the binaries provided by the authors of [21] and the first place team of ICCAD-2017 CAD Contest, all the experiments were run on the same PC with a 2.7GHz CPU and 16GB memory.

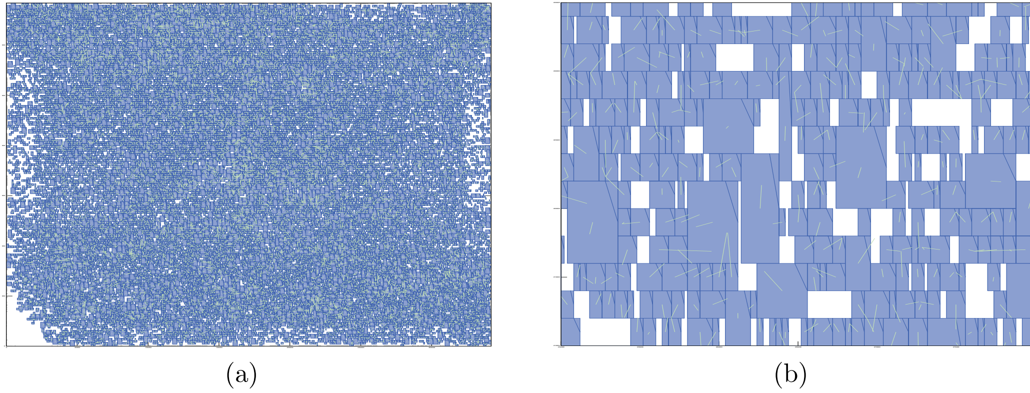
Table 7.1 lists the statistics of the benchmarks and the legalization results. In this table, for all benchmarks, "#C" gives the numbers of total standard cells, "#S" the numbers of total single-row height standard cells, "#M" the numbers of total macro cells, "D.(%)" the cell densities of circuits, "ΔHPWL(%)" the HPWL increases from the corresponding global placement results, "Avg. Move. (sites)" the average cell movements measured in the number of placement site width, "Max. Move. (sites)" the maximum cell movements measured in the

number of placement site width, and "CPU(s)" the runtimes of all compared algorithms.

The experimental results are reported in Table 7.1. The table shows that our algorithm achieves the best results among the three optimization methods. Compared with the work "DAC'17", our algorithm achieves 37% shorter ΔHPWL, 16% smaller average cell movement, and 64% smaller maximum cell movement. The reason is that the method in [21] resolves overlaps only in the horizontal direction, and if some overlaps among cells cannot be resolved in a row, they are marked and moved into blank positions of a circuit in the illegal cell handling step. In contrast, our method allows cells moving in both of the horizontal and vertical directions, and if some overlaps among cells cannot be resolved in a row, then these cells are automatically moved into other adjacent rows. This horizontal- and vertical-direction legalization model can reduce not only the average cell movement but also the maximum cell movement. Since the number of variables in our model is larger than that in [21], our average runtime is longer than that of DAC'17. Compared with the work "Top1", our legalization algorithm also achieves 9% shorter ΔHPWL, 5% smaller average cell movement, and 8% smaller maximum cell movement. For some benchmarks with macros, the maximum cell movements obtained by all the three compared works are very large. These generated maximum cell movements are actually caused by overlapping with macros, which cannot be further reduced. Since "Top1" has shown its great advantages in the contest by winning other teams with very large margins (perhaps the biggest margins in the placement contest history), this comparison further validates the effectiveness of our analytical legalization method. Figures 7.7(a) and 7.7(b) show the respective final layout and a partial layout of the benchmark "fft_2_md2" generated by our legalization algorithm.

Table 7.1: Experimental results.

Circuits	Statistics				ΔHPWL(%)			Avg. Move. (sites)			Max. Move. (sites)			CPU(s)		
	#C	#S	#M	D.(%)	DAC'17	Top1	Ours	DAC'17	Top1	Ours	DAC'17	Top1	Ours	DAC'17	Top1	Ours
des_1	112644	112644	0	90.59	16.21	6.83	6.32	10.86	7.12	6.80	200.82	49.95	49.95	11.23	49.81	27.28
des_a_md1	108288	103589	4	55.05	3.27	2.67	2.42	6.71	6.07	5.92	607.30	607.30	607.30	2.30	3.55	27.10
des_a_md2	108288	105030	4	55.86	3.35	2.77	2.52	6.77	6.17	5.90	403.86	403.86	403.86	2.19	3.98	34.91
des_b_md1	112679	106702	0	54.98	1.75	1.61	1.47	5.17	4.94	4.73	79.34	60.63	40.50	2.01	4.16	23.32
des_b_md2	112679	101908	0	64.69	2.05	1.78	1.72	5.74	5.43	5.25	198.74	45.24	39.76	2.31	4.50	6.39
edit_1_md1	130661	118005	0	67.47	1.47	1.40	1.40	6.22	5.94	5.76	109.34	83.75	74.34	3.49	4.22	8.92
edit_a_md2	127414	115066	6	59.42	1.17	1.08	0.99	6.02	5.69	5.47	164.00	164.00	164.00	2.59	4.30	16.48
edit_a_md3	127414	119616	6	56.92	2.69	1.70	1.39	9.11	7.63	6.77	233.00	233.00	233.00	5.91	10.03	29.71
fft_2_md2	32281	28930	0	83.12	11.21	9.02	8.52	8.84	7.80	7.44	102.94	66.25	62.69	0.70	1.16	7.22
fft_a_md2	30625	27431	6	32.41	0.98	0.97	0.96	5.03	4.94	4.85	345.50	345.50	345.50	0.69	1.14	2.62
fft_a_md3	30625	28609	6	31.24	1.08	1.07	1.07	4.73	4.64	4.54	109.62	109.62	109.62	0.63	1.07	1.88
pci_a_md1	29533	26680	4	49.57	3.61	3.19	3.16	6.01	5.70	5.53	72.48	63.76	63.76	0.61	0.91	9.64
pci_a_md2	29533	25239	4	57.69	8.33	5.12	3.89	9.43	7.93	6.75	186.08	122.06	121.35	0.53	1.16	15.25
pci_b_md1	28914	26134	6	26.47	2.55	2.36	2.06	6.35	6.08	5.88	322.71	332.71	332.71	0.52	1.07	5.59
pci_b_md2	28914	28038	6	18.20	2.80	2.65	2.47	5.92	5.69	5.49	640.12	640.12	430.04	0.50	0.95	3.80
pci_b_md3	28914	27452	6	22.13	3.63	3.37	3.06	6.74	6.40	6.06	398.57	398.57	398.57	0.51	0.98	6.29
N.Avg.					1.37	1.09	1.00	1.16	1.05	1.00	1.64	1.08	1.00	0.18	0.37	1.00



(a) (b)

Figure 7.7: (a) Legalization result of the benchmark “fft_2_md2” from our algorithm. Cells are in blue, and movement in green. (b) A partial layout of (a).

7.6 Summary

In this chapter, we have considered both the average and the maximum cell movements for the mixed-cell-height standard-cell legalization problem. By analyzing and remodeling the objective function and constraints, we formulated the mixed-cell-height standard-cell legalization problem as an MIQP, which considers not only the average cell movement, but also the maximum movement, the sub-maximum movement, the third maximum movement, etc. Then, the MIQP was relaxed to a QP, which allows cells spreading in both of the horizontal and vertical directions. By substituting and duplicating variables, the QP was further converted to an equivalent LCP. Then the MMSIM was used to solve the LCP. Experimental results have shown that our analytical legalization method is effective in reducing wirelength and the average and maximum cell movements.

Chapter 8 Conclusions and Future Works

In this chapter, the results obtained in this thesis will be summarized, and further works will also be suggested.

8.1 Conclusions

In this thesis, we have investigated the optimization theories and algorithms for the challenges of emerging technologies and the advanced design technologies in VLSI circuit industry.

- In Chapter 2, we proposed a discrete relaxation theory, and developed a discrete relaxation based decomposition framework for the TPL layout decomposition problem. By extending the existing line projection method, we developed a surface projection method for identifying features which are critical and should be colored prior, and this forms a basis of our discrete relaxation method. To solve the TPL layout decomposition problem, our discrete relaxation based decomposition method relaxes the problem in two steps. Firstly, the conflict graph was reduced to small size subgraphs by two graph reduction techniques, which is a discrete relaxation of the TPL problem. After that, the TPL problem on the small subgraphs was relaxed to a nonlinear 0-1 programming problem by ignoring stitch insertions and assigning weights to features. To legalize an optimal solution of the relaxation problem to a feasible one of the TPL layout decomposition problem, some techniques were carefully adopted, e.g., the one-stitch first insertion, backtrack coloring. Experiments on the tested benchmarks have shown that our decomposition method is efficient and effective, compared with the state-of-the-art decomposers. Moreover, by our theoretical results, we obtained optimal decompositions for some benchmarks. The developed

discrete relaxation based decomposition method for TPL is successful. We believe the idea can be applied to other problems.

- Hybrid e-beam and triple patterning lithography is a new technology for manufacture of VLSI circuit, which combines the advantages of e-beam and TPL. Layout decomposition is a core problem in the hybrid lithography, which is NP-hard on the general layout. In Chapter 3, we proposed a two stage layout decomposition flow for the HETLD problem, which achieves decomposition by two steps. First, we considered the e-beam and stitch aware TPL mask assignment (ESTMA) problem, and then the problem was relaxed by deleting some conflict edges, which was used for fast obtaining a solution with some conflicts. Second, the infeasible solution with conflicts was legalized to a feasible one of the HETLD problem by stitch insertion and e-beam shot. To speed up decomposition, we reduced the problem size by removing some vertices and some edges before decomposition. Furthermore, in order to obtain a better solution with less VSB number, we proposed the extended minimum weight dominating set for R_4 mask assignment (MDSR_4MA) problem. By solving the MDSR_4MA problem in the first decomposition stage, we could obtain a solution with the patterns in R_4 more likely being assigned to TPL masks by stitch insertion. However, the ILP formulation of the MDSR_4MA problem has many more variables and constraints than the ILP formulation of the ESTMA problem. In the decomposition process, our objective is maximizing e-beam throughput (minimizing VSB number) and minimizing stitch number. Experimental results have shown the effectiveness of the ESTMA and the MDSR_4MA based decomposition methods, compared with the state-of-the-art decomposer.

- In Chapter 4, we considered the contact layer mask and template assignment problem of DSA with TPL for general layout, and proposed a discrete relaxation method. First, we introduced negative edges in the conflict group-

ing graph, and weight the edges of the conflict grouping graph. Then we formulated a discrete relaxation problem of the contact layer assignment problem of DSA with TPL. For obtaining better results, we introduced triangle edges in the weighted conflict grouping graph, and thus introduced some valid inequalities in the discrete relaxation problem. We transformed the discrete relaxation solution to a legal solution of the initial problem by addressing the template assignment problem on the layout graph. Our discrete relaxation based method can estimate the gap between the obtained solution and the optimal solution in the experiment, which is meaningful for the NP-hard problem. Furthermore, our experimental results have shown that the gaps between the obtained solutions and the optimal solutions are very small. Specially, the discrete relaxation approach verifies the optimality of our experimental results of sparse benchmarks since the gaps are 0. Finally, it must be remarked that we only consider the 1-D templates in this chapter. However, the proposed method can be extended to handle more general templates like 2×2, which needs further careful investigation.

- In Chapter 5, we considered the redundant via insertion and guiding template assignment for the DSA with multiple patterning (RGDM) problem, including single patterning (RGDS), double patterning (RGDD) and triple patterning (RGDT). First, for the RGDS problem, we constructed a new ILP formulation basing on our conflict graph. The vertices in the conflict graph are *multiplet*s instead of guiding template assignments (GTAs), which can greatly reduce the size of the conflict graph. To fast solve the ILP, a local optimal MWIS solver was introduced to obtain a local optimal result. Second, for the RGDD and RGDT problems, we proposed a two stage method. At the first stage, a contraction graph was constructed, and the max-M-cut problem is formulated and solved to obtain a mask assignment. At the second stage, our MWIS solver for RGDS was used to obtain a redundant via insertion and guiding template assignment for every mask.

Experimental results have validated the efficiency and effectiveness of our ILP formulation and algorithms.

- In Chapter 6, we further considered dummy via insertion for the redundant via insertion and DSA guiding template assignment problem. We constructed a conflict graph on grid model. The vertices in the conflict graph are *multiplet*s instead of guiding template candidates. This substitution can greatly reduce the size of the conflict graph. Based on the conflict graph, we formulated the problem as an integer linear programming (ILP). To fast solve the ILP, we relaxed it as an unconstrained nonlinear programming (UNP), and then developed a line search optimization algorithm to obtain a local optimal solution of the UNP. Since the algorithm is highly dependent on the initial solution, we used a good enough initial solution instead of random generation. Experimental results have demonstrated that the proposed algorithm is effective and efficient. Experimental comparisons have indicated that consideration of dummy via insertion for the problem is better than without the help of dummy via.

- In Chapter 7, we considered both the average and the maximum cell movements for the mixed-cell-height standard-cell legalization problem. By analyzing and remodeling the objective function and constraints, we formulated the mixed-cell-height standard-cell legalization problem as an MIQP, which considers not only the average cell movement, but also the maximum movement, the sub-maximum movement, the third maximum movement, etc. Then, the MIQP was relaxed to a QP, which allows cells spreading in both of the horizontal and vertical directions. By substituting and duplicating variables, the QP was further converted to an equivalent LCP. Then the MMSIM was used to solve the LCP. Experimental results have shown that our analytical legalization method is effective in reducing wirelength and the average and maximum cell movements.

8.2 Future Works

For the development of emerging lithography technologies, besides the challenges we investigated in this thesis, more other challenges still need to be further studied and resolved. These include:

- **MPL aware mixed-row-height standard cell design:** In the near future, multiple patterning (MPL) technology would be widely used to fabricated advanced circuits. Existing works have considered MPL aware detailed placement for the single-row-height standard cell. For modern mixed-row-height standard cell circuit, the MPL friendly designs are still needed.

- **EBL friendly placement:** Besides low throughput, fogging effect and proximity effect are two crucial challenges for EBL. A friendly consideration is EBL aware placement, which achieves optimization by placing cell to avoid the fogging and proximity effects.

- **Manufacturing metal wire by 2D DSA:** At present, most of the DSA researches focus on fabricating vias/contacts. In the near future, 2D DSA may be used to fabricate wires in metal layers. Hence, to obtain a better patterning result with high resolution, the topic of 2D DSA aware routing and placement will be further touched.

- **DSA-EUV layout decomposition with Multi BCP:** Recent years, many works have concerned the DSA complementary MPL technology. However, due to the limitation of present correction technology, only a few of regular DSA structures are discussed. Under the EUV lithography technology, more DSA structures and more BCP materials will be allowed to fabricate features in layout. A crucial challenge is still layout decomposition with more complex constraints.

- **Layout-dependent effect aware physical design:** Layout-dependent effect (LDE) causes variation in device performance as well as mismatch in

model-hardware correlation (MHC) in sub-10nm nodes. In order to effectively explore the power-performance envelope for IC design, cell libraries must provide cells with different diffusion heights, leading to neighbor diffusion effect (NDE) due to inter-cell diffusion height change (diffusion steps). A desirable design should enable handling NDE. Similarly, metal boundary effect (MBE), gate line end effect (LEE), length of oxidation (LO), and well proximity effect (WPE) should also be considered during physical design.

- **Incremental design:** Most of the existing EDA design tools cannot support an incremental operation. A tiny schematic change may require to perform the existing EDA design tool again. Undoubtedly, this start-all-over-again manner is time- and cost-consuming. So, it would be better if an EDA tool can support incremental design for a tiny schematic change.

Bibliography

[1] Cadence SOC Encounter (Sep. 1, 2016). http://www.cadence.com/.

[2] CPLEX (Jul. 1, 2016). http://www-01.ibm.com/.

[3] GUROBI (Jun. 1, 2015). http://www.gurobi.com/html/academic.html.

[4] ITRS (Mar. 1, 2015). http://www.itrs.net.

[5] OpenSPARC T1 (Sep. 1, 2016). www.oracle.com/technetwork/systems/opensparc.

[6] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer, 2003.

[7] Y. Badr, A. Torres, and P. Gupta. Mask assignment and synthesis of DSA-MP hybrid lithography for sub-7nm contacts/vias. In *Proc. ACM/IEEE Design Automation Conference*, pages 1–6, 2015.

[8] S. H. Baek, H. Y. Kim, Y. K. Lee, D. Y. Jin, S. C. Park, and J. D. Cho. Ultra high density standard cell library using multi-height cell structure. In *Proc. SPIE 7268, Photomask and Next-Generation Lithography Mask Technology XV*, 2008.

[9] Z. Z. Bai. Modulus-based matrix splitting iteration methods for linear complementarity problems. *Numerical Linear Algebra with Applications*, 17(6):917–933, 2010.

[10] X. Y. Bao, H. Yi, C. Bencher, L. W. Chang, H. Dai, Y. Chen, P. T. Chen, and H. S. Wong. Sram, nand, dram contact hole patterning using block copolymer directed self-assembly guided by small topographical templates. In *Proc. IEEE International Electron Devices Meeting*, 2011.

[11] C. M. Bates, T. Seshimo, M. J. Maher, W. J. Durand, J. D. Cushen, L. M. Dean, G. Blachut, C. J. Ellison, and C. G. Willson. Polarity-switching top coats enable orientation of sub-10-nm block copolymer domains. *Science*, 338(6108):775–779, 2012.

[12] R. Borndorfer and R. Weismantel. Discrete relaxations of combinatorial programs. *Discrete Applied Mathematics*, 112:11–26, 2001.

[13] Y. Borodovsky. Lithography 2009 overview of opportunities. *Semicon West*, 2009.

[14] Y. Borodovsky. Mpprocessing for mpprocessors. In *Proc. Maskless Lithography and Multibeam Mask Writer Workshop*, 2010.

[15] W. Brendel and S. Todorovic. Segmentation as maximum-weight independent set. In *Conference on Neural Information Processing System*, pages 307–315, 2010.

[16] U. Brenner. Bonnplace legalization: Minimizing movement by iterative augmentation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(8):1215–1227, 2013.

[17] I. S. Bustany, D. Chinnery, J. R. Shinnerl, and V. Yutsis. ISPD 2015 benchmarks with fence regions and routing blockages for detailed-routing-driven placement. In *Proc. ACM International Symposium on Physical Design*, pages 157–164, 2015.

[18] Y.-W. Chang, R.-G. Liu, and S.-Y. Fang. EUV and E-Beam manufacturability: challenges and solutions. In *Proc. ACM/IEEE Design Automation Conference*, pages 198–204, 2015.

[19] J. G. Chase and B. W. Smith. Overview of modern lithography techniques and a MEMS-Based approach to high throughput rate electron beam lithography. *Journal of Intelligent Material System and Structures*, 12(12):807–817, 2001.

[20] H.-Y. Chen, M.-F. Chiang, Y.-W. Chang, L. Chen, and B. Han. Full-chip routing considering double-via insertion. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(5):844–857, 2008.

[21] J. Chen, Z. Zhu, W. Zhu, and Y.-W. Chang. Toward optimal legalization for mixed-cell-height circuit designs. In *Proc. ACM/IEEE Design Automation Conference*, 2017.

[22] H.-A. Chien, Y.-H. Chen, S.-Y. Han, H.-Y. Lai, and T.-C. Wang. On refining row-based detailed placement for triple patterning lithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(5):778–793, 2015.

[23] H.-A. Chien, S.-Y. Han, Y.-H. Chen, and T.-C. Wang. A cell-based row-structure layout decomposer for triple patterning lithography. In *Proc. ACM International Symposium on Physical Design*, pages 67–74, 2015.

[24] E. Chlamtac and M. Tulsiani. *Convex relaxations and integrality gaps*, volume 166. Handbook on Semidefinite, Conic and Polynomial Optimization, Editors: M. F. Anjos and J. B. Lasserre, International Series in Operations Research & Management Science, 2012.

[25] M. Cho, H. Ren, H. Xiang, and R. Puri. History-based VLSI legalization using network flow. In *Proc. ACM/IEEE Design Automation Conference*, pages 286–291, 2010.

[26] W.-K. Chow, C.-W. Pui, and E. F. Young. Legalization algorithm for multiple-row height standard cell design. In *Proc. ACM/IEEE Design Automation Conference*, 2016.

[27] C. Cork, J. C. Madre, and L. Barnes. Comparison of triple-patterning decomposition algorithms using aperiodic tiling patterns. In *Proc. SPIE 7028, Photomask and Next-Generation Lithography Mask Technology XV*, 2008.

[28] N. K. Darav, I. S. Bustany, A. Kennings, and L. Behjat. A fast, robust network flow-based standard-cell legalization method for minimizing maximum movement. In *Proc. ACM International Symposium on Physical Design*, 2017.

[29] N. K. Darav, I. S. Bustany, A. Kennings, and R. Mamidi. ICCAD-2017 CAD Contest in multi-deck standard cell legalization and benchmarks suite. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2017.

[30] N. K. Darav, A. Kennings, A. F. Tabrizi, D. Westwick, and L. Behjat. Eh?placer: a high-performance modern technology-driven placer. *ACM Transactions on Design Automation of Electronic Systems*, 21(3):37:1–37:27, April 2016.

[31] Y. Ding, C. Chu, and W.-K. Mak. Throughput optimization for SADP and e-beam based manufacturing of 1D layout. In *Proc. ACM/IEEE Design Automation Conference*, pages 1–6, 2014.

[32] S. Dobre, A. B. Kahng, and J. Li. Mixed cell-height implementation for improved design quality in advanced nodes. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 854–860, 2015.

[33] Y. Du, D. Guo, M. D. Wong, H. Yi, H.-S. P. Wong, H. Zhang, and Q. Ma. Block copolymer directed self-assembly (dsa) aware contact layer optimization for 10 nm 1d standard cell library. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 186–193, 2013.

[34] Y. Du, Z. Xiao, M. D. F. Wong, H. Yi, and H.-S. P. Wong. DSA-aware detailed routing for via layer optimization. In *Proc. SPIE 9049, Photomask and Next-Generation Lithography Mask Technology XV*, 2014.

[35] Y. Du, H. Zhang, M. D. F. Wong, and K.-Y. Chao. Hybrid lithography optimization with e-beam and immersion processes for 16nm 1D gridded design. In *Proc. IEEE/ACM Asia and South Pacific Design Automation Conference*, pages 707–712, 2012.

[36] S. Y. Fang, Y. W. Chang, and W. Y. Chen. A novel layout decomposition algorithm for triple patterning lithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(3):397–408, 2014.

[37] S. Y. Fang, S. Y. Chen, and Y. W. Chang. Native-conflict and stitch-aware wire perturbation for double patterning technology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(5):703–716, 2012.

[38] S.-Y. Fang and Y.-X. Hong. Design optimization considering guiding template feasibility and redundant via insertion for directed self-assembly. In *Proc. IEEE Asia Pacific Conferenc on Circuit and Systems*, 2016.

[39] S.-Y. Fang, Y.-X. Hong, and Y.-Z. Lu. Simultaneous guiding template optimization and redundant via insertion for directed self-assembly. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36(1):156–169, 2017.

[40] A. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica*, (18):67–81, 1997.

[41] A. Fujimura. Design for e-beam: design insights for direct-write maskless lithography. In *Proc. SPIE 7823, Photomask and Next-Generation Lithography Mask Technology XV*, pages 137–140, Sep. 2010.

[42] J.-R. Gao, B. Yu, and D. Z. Pan. Self-aligned double patterning layout decomposition with complementary e-beam lithography. In *Proc. IEEE/ACM Asia and South Pacific Design Automation Conference*, pages 143–148, 2014.

[43] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical computer science*, 1(3):237–267, 1976.

[44] R. A. Griffiths, A. Williams, C. Oakland, J. Roberts, A. Vijayaraghavan, and T. Thomson. Directed self-assembly of block copolymers for use in bit patterned media fabrication. *Journal of Physics D: Applied Physics*, 46(50):503001, 2013.

[45] C.-C. Huang, H.-Y. Lee, B.-Q. Lin, S.-W. Yang, C.-H. Chang, S.-T. Chen, Y.-W. Chang, T.-C. Chen, and I. Bustany. Ntuplace4dr: a detailed-routing-driven placer for mixed-size circuit designs with technology and region constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, DOI: 10.1109/TCAD.2017.2712665, 2017.

[46] C.-Y. Hung, P.-Y. Chou, and W.-K. Mak. Mixed-cell-height standard cell placement legalization. In *Proc. ACM Great Lakes Symposium on VLSI*, 2017.

[47] C.-Y. Hung, P.-Y. Chou, and W.-K. Mak. Optimizing DSA-MP decomposition and redundant via insertion with dummy vias. In *Proc. IEEE/ACM Asia South Pacific Design Automation Conference*, 2017.

[48] R. Inanami, S. Magoshi, S. Kousai, A. Ando, T. Nakasugi, I. Mori, K. Sugihara, and A. Miura. Maskless lithography: Estimation of the number of shots for each layer in a logic device with character projection-type low-energy electron-beam direct writing system. In *Proc. SPIE 5037, Photomask and Next-Generation Lithography Mask Technology XV*, pages 1043–1050, 2003.

[49] G. Jager and A. Srivastav. Improved approximation algorithms for maximum graph partitioning problems. *Foundations of Software Technology and Theoretical Computer Science*, 10(2):133–167, 2005.

[50] A. B. Kahng, I. L. Markov, and S. Reda. On legalization of row-based placements. In *Proc. ACM Great Lakes Symposium on VLSI*, pages 214–219, 2004.

[51] A. B. Kahng, C. H. Park, X. Xu, and H. Yao. Layout decomposition for double patterning lithography. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 465–472, 2008.

[52] J. Kuang, J. J. Ye, and F. Y. Young. Simultaneous template optimization and mask assignment for DSA with multiple patterning. In *Proc. IEEE/ACM Asia and South Pacific Design Automation Conference*, pages 75–82, 2016.

[53] J. Kuang and E. F. Young. An efficient layout decomposition approach for triple patterning lithography. In *Proc. ACM/IEEE Design Automation Conference*, pages 1–6, 2013.

[54] D. K. Lam. Maskless electron-beam lithography for trusted microchip production. In *Multibeam Corporation Santa Clara United States*, 2016.

[55] A. Latypov, T. H. Coskun, G. Garner, M. Preil, G. Schmid, J. Xu, and Y. Zou. Simulations of spatial DSA morphology, DSA-aware assist features and block copolymer-homopolymer blends. In *Proc. SPIE 9049, Photomask and Next-Generation Lithography Mask Technology XV*, 2014.

[56] K.-Y. Lee, C.-K. Koh, T.-C. Wang, and K.-Y. Chao. Fast and optimal redundant via insertion. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(12):2197–2208, 2008.

[57] K.-Y. Lee and T.-C. Wang. Post-routing redundant via insertion for yield/reliability improvement. In *Proc. IEEE/ACM Asia South Pacific Design Automation Conference*, pages 303–308, 2006.

[58] M. Leordeanu, M. Hebert, and R. Sukthankar. An integer projected fixed point method for graph matching and MAP inference. In *Conference on Neural Information Processing System*, pages 1114–1122, 2009.

[59] X. Li, Z. Zhu, and W. Zhu. Discrete relaxation method for triple patterning lithography layout decomposition. *IEEE Transactions on Computers*, 66(2):285–298, 2017.

[60] K.-L. Lin and S.-Y. Fang. Guiding template-aware routing considering redundant via insertion for directed self-assembly. In *Proc. IEEE/ACM Asia South Pacific Design Automation Conference*, 2017.

[61] S.-T. Lin, K.-Y. Lee, T.-C. Wang, C.-K. Koh, and K.-Y. Chao. Simultaneous redundant via insertion and line end extension for yield optimization. In *Proc. IEEE/ACM Asia South Pacific Design Automation Conference*, pages 633–638, 2011.

[62] Y. Lin, X. Xu, B. Yu, R. Baldicky, and D. Z. Pan. Triple/quadruple patterning layout decomposition via novel linear programming and iterative rounding. In *Proc. SPIE 9781, Design-Process-Technology Co-optimization for Manufacturability X*, 2016.

[63] C. Lucet, F. Mendes, and A. Moukrim. Pre-processing and linear-decomposition algorithm to solve the k-colorability problem. *Lecture Notes in Computer Science*, 3059:315–325, 2004.

[64] Y. Ma, J. Lei, J. A. Torres, L. Hong, J. Word, G. Fenger, A. Tritchkov, G. Lippincott, R. Gupta, N. Laerty, Y. He, J. Bekaert, and G. Vanderberghe. Directed self-assembly (DSA) grapho-epitaxy template generation with immersion lithography. *Journal of Micro/Nanolithography, MEMS and MOEMS*, 14(3):031216–031216, 2015.

[65] C. Mack. *Fundamental principles of optical lithography: the science of microfabrication*. John Wiley & Sons, 2008.

[66] W.-K. Mak and C. Chu. E-Beam lithography character and stencil co-optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(5):741–751, 2014.

[67] T. Maruyama, Y. Machida, S. Sugatani, H. Takita, H. Hoshino, T. Hino, M. Ito, A. Yamada, T. Iizuka, and S. Komatsu. CP element based design for 14nm node EBDW high volume manufacturing. In *Proc. SPIE 8323, Photomask and Next-Generation Lithography Mask Technology XV*, pages 8323–14, Apr. 2012.

[68] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–48, 1957.

[69] J. Nocedal and S. J. Wright. *Numerical optimization, 2nd Edition*. New York: Springer, 2006.

[70] J. Ou, B. Yu, and D. Z. Pan. Concurrent guiding template assignment and redundant via insertion for DSA-MP hybrid lithography. In *Proc. ACM International Symposium on Physical Design*, pages 39–46, 2016.

[71] J. Ou, B. Yu, X. Xu, J. Mitra, Y. Lin, and D. Z. Pan. DSAR: DSA aware routing with simultaneous DSA guiding pattern and double patterning assignment. In *Proc. ACM International Symposium on Physical Design*, pages 91–98, 2017.

[72] L. Pain, M. Jurdit, J. Todeschini, S. Manakli, B. Icard, B. Minghetti, G. Bervin, A. Beverina, F. Leverd, M. Broekaart, P. Gouraud, V. De Jonghe, Ph. Brun, S. Denorme, F. Boeuf, V. Wang, and D. Henry. Electron beam direct write lithography flexibility for asic manufacturing an opportunity for cost reduction. In *Proc. SPIE*, volume 5751, 2005.

[73] D. Z. Pan, B. Yu, and J.-R. Gao. Design for manufacturing with emerging nanolithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(10):1453–1472, 2013.

[74] H. C. Pfeiffer. New prospects for electron beams as tools for semiconductor lithography. In *Proc. SPIE*, volume 7378, 2009.

[75] R. Ruiz, H. Kang, F. A. Detcheverry, E. Dobisz, D. S. Kercher, T. R. Albrecht, J. J. de Pablo, and P. F. Nealey. Density multiplication and improved lithography by directed block copolymer assembly. *Science*, 321(5891):936–939, 2008.

[76] E. Sahouria and A. Bowhill. Generalization of shot definition for variable shaped e-beam machines for write time reduction. In *Proc. SPIE*, volume 7283, 2010.

[77] S. Shim, W. Chung, and Y. Shin. Redundant via insertion for multiple-patterning directed-self-assembly lithography. In *Proc. ACM/IEEE Design Automation Conference*, number 41, 2016.

[78] S. E. Steen, S. J. McNab, L. Sekaric, I. Babich, J. Patel, J. Bucchignano, M. Rooks, D. M. Fried, A. W. Topol, and J. R. Brancaccio. Looking into the crystal ball: future device learning using hybrid e-beam and optical lithography (keynote paper. *Microlithography 2005. International Society for Optics and Photonics*, pages 26–34, 2005.

[79] X. Tang and M. Cho. Optimal layout decomposition for double patterning technology. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 9–13, 2011.

[80] H. Tian, Y. Du, H. Zhang, Z. Xiao, and M. D. F. Wong. Triple patterning aware detailed placement with constrained pattern assignment. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 116–123, 2014.

[81] H. Tian, H. Zhang, Q. Ma, Z. Xiao, and D. F. Wong. A polynomial time triple patterning algorithm for cell based row-structure layout. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 57–64, 2012.

[82] H. Tian, H. Zhang, Z. Xiao, and M. D. F. Wong. Hybrid lithography for triple patterning decomposition and e-beam lithography. In *Proc. SPIE 9052 Advanced Lithography International Society for Optics and Photonics*, 2014.

[83] C. Wagner and N. Harned. EUV lithography: Lithography gets extreme. *Nature Photonics*, 4(1):24–26, 2010.

[84] C.-H. Wang, Y.-Y. Wu, J. Chen, Y.-W. Chang, S.-Y. Kuo, W. Zhu, and G. Fan. An effective legalization algorithm for mixed-cell-height standard cells. In *Proc. IEEE/ACM Asia and South Pacific Design Automation Conference*, 2017.

[85] L.-T. Wang, K.-T. Cheng, and Y.-W. Chang. *Electronic Design Automation: Synthesis, Verification, and Test*. Morgan Kaufmann, 2009.

[86] L. Wilson. *International technology roadmap for semiconductors*. Semiconductor Industry Association, 2013.

[87] H.-S. P. Wong, C. Bencher, X.-Y. Bao H. Yi, and L.-W. Chang. Block copolymer directed self-assembly enables sublithographic patterning for device fabrication. In *Proc. SPIE 8323, Photomask and Next-Generation Lithography Mask Technology XV*, volume 8323, San Jose, CA, USA, 2012.

[88] H.-S. P. Wong, H. Yi, M. Tung, and K. Okabe. Physical layout design of directed self-assembly guiding alphabet for ic contact hole/via patterning. In *Proc. ACM International Symposium on Physical Design*, 2015.

[89] G. Wu and C. Chu. Detailed placement algorithm for vlsi design with double-row height standard cells. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(9):1569–1573, 2015.

[90] Z. Xiao, Y. Du, H. Tian, M. D. F. Wong, H. Yi, H.-S. P. Wong, and H. Zhang. Directed self-assembly (DSA) template pattern verification. In *proc. ACM/IEEE Design Automation Conference*, pages 1–6, 2014.

[91] Z. Xiao, Y. Du, M. D. F. Wong, H. Yi, P. Wong, and H. Zhang. Contact pitch and location prediction for directed self-assembly template verification. In *Proc. IEEE/ACM Asia and South Pacific Design Automation Conference*, pages 644–651, 2015.

[92] Z. Xiao, D. Guy, M. D. Wong, H. Yi, M. C. Tung, and H. S. P. Wong. Layout optimization and template pattern verification for directed self-assembly (DSA). In *Proc. of ACM/IEEE Design Automation Conference*, number 199, 2015.

[93] Z. Xiao, C. X. Lin, M. D. F. Wong, and H. Zhang. Contact layer decomposition to enable dsa with multi-patterning technique for standard cell based layout. In *Proc. IEEE/ACM and South Pacific Design Automation Conference*, volume 95–102, 2016.

[94] G. Xu, L. D. Huang, D. Z. Pan, and M. D. F. Wong. Redundant-via enhanced maze routing for yield improvement. In *Proc. IEEE/ACM Asia South Pacific Design Automation Conference*, pages 1148–1151, 2005.

[95] Y. Xu and C. Chu. Grema: graph reduction based efficient mask assignment for double patterning technology. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 601–606, 2009.

[96] Y. Yang, W.-S. Luk, D. Z. Pan, H. Zhou, C. Yan, D. Zhou, and X. Zeng. Layout decomposition co-optimization for hybrid e-beam and multiple patterning lithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(9):1532–1545, 2016.

[97] H. Yao, Y. Cai, X. Hong, and Q. Zhou. Improved multilevel routing with redundant via placement for yield and reliability. In *Proc. ACM Great Lakes Symposium on VLSI*, pages 143–146, 2005.

[98] H. Yi, X.-Y. Bao, R. Tiberio, and H.-S. P. Wong. Design strategy of small topographical guiding templates for sub-15nm integrated circuits contact hole patterns using block copolymer directed self assembly. In *Proc. SPIE 8680, Photomask and Next-Generation Lithography Mask Technology XV*, volume 8680, San Jose, CA, USA, 2013.

[99] H. Yi, X. Y. Bao, J. Zhang, C. Bencher, L. W. Chang, X. Chen, R. Tiberio, J. Conway, H. Dai, Y. Chen, S. Mitra, and H.-S. P. Wong. Flexible control of block copolymer directed self assembly using small, topographical templates: potential lithography solution for integrated circuit contact hole patterning. *Advanced Materials*, 24(23):3107–3114, 2012.

[100] B. Yu, J.-R. Gao, D. Ding, Y. Ban, J.-S. Yang, K. Yuan, M. Cho, and David Z Pan. Dealing with ic manufacturability in extreme scaling. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 240–242, 2012.

[101] B. Yu, Y. H. Lin, G. Luk-Pat, D. Ding, K. Lucas, and D. Z. Pan. A high-performance triple patterning layout decomposer with balanced density. In

Proc. IEEE/ACM International Conference on Computer-Aided Design, pages 163–169, 2013.

[102] B. Yu and D. Z. Pan. *Design for Manufacturability with Advanced Lithography.* Springer, 2015.

[103] B. Yu, X. Xu, J. R. Gao, Y. Lin, Z. Li, C. Alpert, and D. Z. Pan. Methodology for standard cell compliance and detailed placement for triple patterning lithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(5):726–739, 2015.

[104] B. Yu, K. Yuan, J.-R. Gao, and D. Z. Pan. E-BLOW: E-Beam lithography overlapping aware stencil planning for MCC system. In *Proc. ACM/IEEE Design Automation Conference*, pages 70–76, 2013.

[105] B. Yu, K. Yuan, B. Zhang, D. Ding, and D. Z. Pan. Layout decomposition for triple patterning lithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(3):433–446, 2015.

[106] B. Yu, K. Yuan, B. Zhang, D. Ding, and D. Z. Pan. Layout decomposition for triple patterning lithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(3):433–446, 2015.

[107] K. Yuan, J. S. Yang, and D. Z. Pan. Lithography 2009 overview of opportunities. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(2):185–196, 2010.

[108] K. Yuan, B. Yu, and D. Z. Pan. E-beam lithography stencil planning and optimization with overlapped characters. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(2):167–179, 2012.

[109] Y. Zhang, W. S. Luk, H. Zhou, C. Yan, and X. Zeng. Layout decomposition with pairwise coloring for multiple patterning lithography. In *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pages 170–177, 2013.

[110] Y. Zorian, D. Gizopoulos, C. Vandenberg, and P. Magarshack. Guest editors introduction: design for yield and reliability. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(12):2197–2208, 2004.

个 人 简 历

一、个人概况

姓　　名: 李兴权　　性　　别: 男

民　　族: 汉　族　　籍　　贯: 福建龙岩

出生年月: 1990 年 05 月　　政治面貌: 中共党员

专　　业: 应用数学　　毕业院校: 福州大学

研究方向: 电子设计自动化

二、教育背景

1. 2015.09-2018.07: 福州大学离散数学与理论计算机科学研究中心
   应用数学专业　　博士研究生

2. 2013.09-2015.07: 福州大学数学与计算机科学学院
   运筹学与控制论专业　　硕士研究生

3. 2009.09-2013.07: 福州大学数学与计算机科学学院
   应用数学专业　　理学学士

攻读博士学位期间发表的学术论文及研究成果

学术论文：

[1] Xingquan Li, Ziran Zhu and Wenxing Zhu, "Discrete Relaxation Method for Triple Patterning Lithography Layout Decomposition," *IEEE Transactions on Computers*, vol. 66, no. 2, pp. 285–298, 2017.

[2] Xingquan Li and Wenxing Zhu, "Two-Stage Layout Decomposition for Hybrid E-Beam and Triple Patterning Lithography," *ACM Transactions on Design Automation of Electronic Systems*, vol. 23, no. 1, pp. 6:1–6:23, 2017.

[3] Xingquan Li, Jianli Chen and Wenxing Zhu, "Discrete Relaxation Method for Contact Layer Decomposition of DSA with Triple Patterning," *Integration, the VLSI Journal*, Vol. 61, no. 1, pp. 77–87, 2018.

[4] Ye Huang, Xinquan Li, Wenxing Zhu and Jianli Chen, "Cut Redistribution and DSA Template Assignment for Unidirectional Design," In *Proc. IEEE International Conference on ASIC*, 2017.

[5] Xingquan Li, Bei Yu, Jiaojiao Ou, Jianli Chen, David Z. Pan and Wenxing Zhu, "Graph Based Redundant Via Insertion and Guiding Template Assignment for DSA-MP," submitted to *IEEE Transactions on Very Large Scale Integration Systems.*

[6] Xingquan Li, Jianli Chen, Yuhang Chen, Ziran Zhu, Wenxing Zhu and Yao-Wen Chang, "Analytical Mixed-Cell-Height Legalization Considering Average and Maximum Movement Minimization," submitted.

[7] Jianli Chen, Peng Yang, Ye Huang, Xingquan Li, Wenxing Zhu and Yao-Wen Chang, "Mixed-Cell-Height Placement with Minimum-Implant-Area Constraints," submitted to *International Conference on Computer-Aided Design*, 2018.

[8] Xingquan Li, Bei Yu, Jianli Chen and Wenxing Zhu, "Redundant Via Insertion and DSA Guiding Template Assignment with Dummy Via," submitted.

[9] Ziran Zhu, Xingquan Li, Yuhang Chen, Zhipeng Huang, Jianli Chen and Wenxing Zhu,"Technology and Region Constraints-Aware Multi-Deck Standard Cell Legalization", submitted to *International Conference on Computer-Aided Design*, 2018.

[10] Jianli Chen, Yan Liu, Xingquan Li, Ziran Zhu and Wenxing Zhu, "An Improved Simulated Annealing Algorithm and An Excessive Length Model for Fixed-Outline Floorplanning," submitted.

[11] Jianli Chen, Tingshen Lan, Xingquan Li, Hailong Yao, Tsung-Yi Ho and Wenxing Zhu, "Unified Contamination-Aware Routing Method Considering Realistic Washing Capacity Constraint in Digital Microfluidic Biochips," submitted to *International Conference on Computer-Aided Design*, 2018.

[12] Hao Geng, Haoyu Yang, Bei Yu, Xingquan Li and Xuan Zeng, "Sparse VLSI Layout Feature Extraction: A Dictionary Learning Approach (Invited)," In *Proc. IEEE Computer Society Annual Symposium on VLSI*, 2018.

[13] Xiongfeng Chen, Xingquan Li, Xiqiong Bai and Wenxing Zhu, "An Adaptive Memetic Algorithm for VLSI Standard Cell Placement," submitted to *Kuwait Journal of Science*.

专利软件：

1. 朱文兴，李兴权，“基于分解成本最小化和合法化的三重图样光刻布局分解方法”中国发明专利，专利申请号：201510394778.X。

2. 形成基于离散松弛算法为基础的三重光刻技术版图分解算法软件一套。在ISCAS-85 & 89 测试实例集上的实验结果表明，与其他算法相比，我们的算法软件可以找到实例问题的最优解或者近优解。

3. 形成结合三重图样光刻技术和自组装技术的额外通孔插入算法软件一套。

4. 形成考虑结单元总移动量和最大移动量的混合高度标准单元合法化算法软件一套。

获奖情况：

1. ICCAD 2017 CAD Contest on Multi-Deck Standard Cell Legalization 全球第一名

2. 2017年博士研究生国家奖学金

3. 2018年福州大学博士研究生中期优秀学业奖学金壹等奖

4. 2015年第一、第二学年福州大学研究生优秀学业奖学金壹等奖

5. 2015年度福州大学博士生新生奖学金壹等奖

6. 2014年第十一届“华为杯”全国研究生数学建模竞赛贰等奖

7. 2015年福州大学文化科技创新创业奖学金

8. 2014年首届“东兴证券杯”福州大学研究生数学建模竞赛贰等奖

参与的科研项目

[1] 2017.1-2020.12，基于热传导方程的超大规模集成电路布局模型及快速算法研究, 项目编号：61672005，国家自然科学基金面上项目。