A Unified Framework for Simultaneous Layout Decomposition and Mask Optimization

Yuzhe Ma[1], Jhih-Rong Gao[2], Jian Kuang[2], Jin Miao[2], and Bei Yu[1]

[1]CSE Department, The Chinese University of Hong Kong, NT, Hong Kong
[2]Cadence Design Systems, USA
{yzma,byu}@cse.cuhk.edu.hk,{jgao,jkuang,jmiao}@cadence.com

***Abstract*—In advanced technology nodes, layout decomposition and mask optimization are two key stages in integrated circuit design. Due to the inconsistency of the objectives of these two stages, the performance of conventional layout and mask optimization may be suboptimal. To tackle this problem, in this paper we propose a unified framework, which seamlessly integrates layout decomposition and mask optimization. We propose a gradient based approach to solve the unified mathematical formulation, as well as a set of discrete optimization techniques to avoid being stuck in local optimum. The conventional optimization process can be accelerated as some inferior decompositions can be smartly pruned in early stages. The experimental results show that the proposed unified framework can achieve more than 17× speed-up compared with the conventional two-stage flow, meanwhile it can reduce EPE violations by 18%, and thus maintain better design quality.**

I. Introduction

In accordance with the Moore's law, through extreme scaling, the transistor number on a chip has increased exponentially in the last five decades. However, the continued scaling of the transistor feature size has pushed the conventional $193nm$ wavelength lithography system into its resolution limit, thus the whole semiconductor industry is facing severe manufacturing challenges [1]. To overcome these issues, resolution enhancement techniques (RETs) on layout and mask levels toward better printability and yield are of great importance [2].

Two of the most critical RET stages are layout decomposition and mask optimization. In the first stage, layout decomposition divides target image into several masks so that the coarser pitches on every mask can be manufactured through $193nm$ wavelength lithography. According to different processes, layout decomposition can be classified into litho-etch-litho-etch (LELE)-type or spacer-type [1]. Since LELE-type manufacturing process can support complex and flexible design patterns, in this work we concentrate on LELE-type layout decomposition. Depending on the total mask number available, the problem is also called double patterning layout decomposition (two masks) or triple patterning layout decomposition (three masks). In the second stage, each decomposed mask needs to be further refined by mask optimization, e.g., optical proximity correction (OPC), to reduce edge placement error (EPE). Finally, all optimized masks go through lithography process separately, then all printed images are combined together to generate target.

In emerging technology nodes, the conventional two-stage flow (i.e., layout decomposition followed by mask optimization) cannot achieve good printability on their own. The reasons are two-fold. (1) The layout decomposition and mask optimization are separated from each other and each problem is solved independently, which may lose a global view. (2) Due to the inconsistency between the objectives of the two stages, decomposed results with identical quality may cause diverse printed image qualities after mask optimization. That is, the layout decomposition is based on simple design or coloring rules, which are just coarse regression of complicated lithography model; while the mask optimization is verified by accurate and sophisticated lithography simulation. Fig. 1 gives an example on such inconsistency. Given the identical target, two different layout decomposition results are found (LD stage in the figures), and both of them satisfy all design rules and coloring rules. After the mask optimization (MO stage in the figures) on each mask, however, it can be observed that the qualities of the printed images are diverse: Fig. 1(a) has three EPE violations, while Fig. 1(b) has only one EPE violation. Therefore, there is an increasing need to bridge the gap between layout decomposition and mask optimization by a unified design framework.

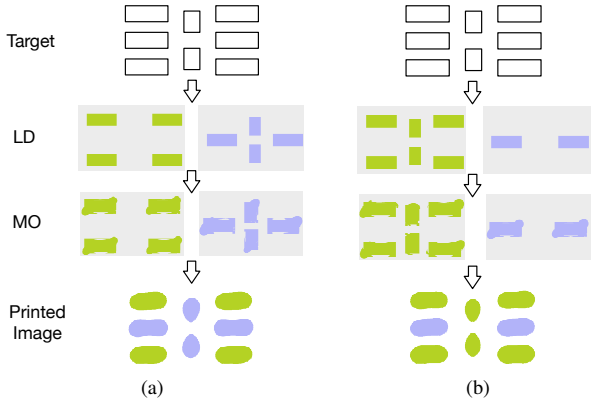


Fig. 1: Same quality layout decompositions (LD) can achieve different EPE violation number after mask optimization (MO): (a) Solution 1 with #EPE violation = 3; (b) Solution 2 with #EPE violation = 1.

There is a wealth of literature on LELE-type layout decomposition. The general layout decomposition problem minimizing both conflict and stitch can be optimally solved through integer linear programming (ILP) [3]–[5]. Due to the computationally intractable of ILP solutions, there are several speed-up techniques under different scenarios. For the double patterning scenario, Xu *et al*. [6] formulated the problem into a maximum-cut problem and proposed an ILP formulation on stitch minimization, while Tang *et al*. [7] computed a stitch graph and proved that the stitch graph is planar, and proposed min-cut-based approaches to minimize the stitch number optimally. For the triple patterning or general multiple patterning scenarios, Yu *et al*. proposed semidefinite programming (SDP) formulation to achieve reducing the ILP formulation [5], [8]; Fang *et al*. [9] discussed several graph division techniques and

This work is supported in part by The Research Grants Council of Hong Kong SAR (Project No. CUHK24209017).

978-1-5386-3093-8/17/$31.00 ©2017 IEEE

proposed a stitch-aware mask assignment algorithm which can find the mask assignment such that the conflicts in the same mask are more likely to be resolved by inserting stitches; Tian *et al*. [10] proposed a polynomial time algorithm for row based design and the mask assignment was found by searching for a shortest path; Lin *et al*. [11] developed a linear programming based relaxation; Kuang *et al*. [12] applied efficient graph matching. Chang *et al*. [13] considered more complex coloring rules for triple patterning layout decomposition, where the graph coloring problem was reduced to an exact cover problem. Kuang *et al*. [14] applied fixed-parameter tractable (FPT) algorithm to layout decomposition. There are few studies considering the quality of printed images in the early layout decomposition stage. Yu *et al*. [15] optimized the local balanced density, while Chen *et al*. [16] introduced the concept of spacing based density. However, both density measurements are just coarse estimate of complicated lithography model, thus their fidelities to the mask optimization are not guaranteed.

There are also investigations on mask optimization or OPC, which can be classified into three types: rule-based OPC, model-based OPC, and inverse lithography technique (ILT). Rule-based OPC requires comprehensive experiments determining design rules to compensate non-desired patterns, thus can only be applied to less aggressive designs [17]. Model-based OPC segments pattern edges into small parts and moves them slightly to make corrections for printed images. However, it is heavily based on lithography simulation so usually is time-consuming [18]–[20]. ILT has become a promising OPC solution, which handles the mask optimization as an inverse problem of the lithography system. Poonawala *et al*. [21] proposed a systematic formulation; Jia *et al*. [22] applied a stochastic gradient descent method to update the mask in each iteration; Gao *et al*. [23] proposed a speed-up technique for lithography simulation as well as a formulation for accurate EPE calculation. Note that some previous literatures [24]–[26] studied multiple exposure effects in ILT framework, but none of them addressed the layout decomposition problem because they only considered the multiple exposures on a single mask.

In this paper, we propose a unified optimization framework which aims at solving layout decomposition and mask optimization simultaneously. Combining the two processes together leads to a larger solution space, which has potential to search for a higher quality mask design. In addition, through effective pruning techniques, our framework can avoid exhaustive mask optimization on all layout decomposition solutions, therefore the overall efficiency can be significantly improved. The main contributions of this work are listed as follows.

- To the best of our knowledge, this is the first work handling multiple patterning layout decomposition and mask optimization simultaneously.
- We propose a unified mathematical formulation and then develop a gradient-based optimization approach.
- We further apply a set of discrete optimization techniques (e.g., semidefinite programming, randomized rounding, and pruning) to avoid being stuck in local optimum.
- The experimental results verify the effectiveness of the proposed framework.

The rest of the paper is organized as follows. Section II introduces the lithography models and gives the problem formulation. Section III describes the algorithmic details in our framework. Section IV lists the experimental results to support our ideas and methodologies, followed by conclusion in Section V.

TABLE I: Notations used in this paper.

$\mathbf{Z}_t$	Target image
$\mathbf{M}_1$, $\mathbf{M}_2$	Output masks
$\mathbf{P}_1$, $\mathbf{P}_2$	Unconstrained variables
$\mathbf{I}_1$, $\mathbf{I}_2$	Aerial images
$\mathbf{Z}_1$, $\mathbf{Z}_2$	Binary images
$\mathbf{Z}$	Printed image
$\mathbf{H}$	A set of optical kernels $\{\mathbf{h}_1, \ldots, \mathbf{h}_K\}$
$\mathbf{H}^*$	The conjugate transpose of $\mathbf{H}$
$\odot$	Element-wise matrix multiplication operator
$\otimes$	Convolution operator

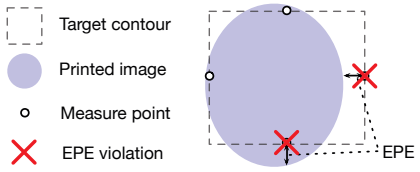


Fig. 2: Illustration of EPE measurement.

II. Preliminaries

In this section, we provide some preliminaries on lithography models, and then introduce the problem formulation. For convenience, some notations used in this paper are listed in TABLE I. Note that in this paper we focus on double patterning scenario, but the problem formulation and the corresponding methodologies can be extended to triple patterning counterpart.

A. Forward Lithography Models

Two models are needed to transform mask patterns into printed image: optical lithography model and photo resist model. First, an aerial image is generated by convolving the mask with a set of optical kernels [27], which is represented as

$$\mathbf{I} = f_{optical}(\mathbf{M}) = \sum_{k=1}^{K} w_k \cdot |\mathbf{M} \otimes \mathbf{h}_k|^2, \tag{1}$$

where $\mathbf{h}_k$ is the k-th optical kernel, w_k is the weight of $\mathbf{h}_k$, and K is the total kernel number.

Then a resist model is applied to the aerial image. In our work a constant threshold resist model is used, which sets an intensity threshold I_{th} to binarize the aerial image, denoted by $\mathbf{Z}$ in the following equation.

$$\mathbf{Z}(x,y) = f_{resist}(\mathbf{I}) = \begin{cases} 1, & \text{if } \mathbf{I}(x,y) \geq I_{th}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Finally, binary images $\mathbf{Z}_1 = f_{resist}(\mathbf{I}_1)$ and $\mathbf{Z}_2 = f_{resist}(\mathbf{I}_2)$ are combined to form the printed image. Considering that the printed image is binary as well, the process can be represented by performing logical OR operation as follows:

$$\mathbf{Z}(x,y) = \mathbf{Z}_1(x,y) \vee \mathbf{Z}_2(x,y). \tag{3}$$

B. Problem Formulation

Given a target layout and the printed image, the edge placement error (EPE) is defined as follows.

Definition 1 (EPE)**.** *EPE is defined as the geometric displacement of the image contour from the edge of target image on the layout. A violation is introduced if the perpendicular displacement is greater than an EPE threshold value.*

In our implementation, the EPE threshold value is set to $10nm$. An example of our EPE measurement is given in Fig. 2: to facilitate

the computation, a set of measure points are sampled on each edge and EPE violation will be checked at the measure points.

Based on the above notations, the problem of layout decomposition and mask optimization (LDMO) is defined as follows.

Problem 1 (LDMO). *Given target image $\mathbf{Z}_t$, two optimized masks, $\mathbf{M}_1$ and $\mathbf{M}_2$, are generated. The objectives are to minimize the difference between the final printed image $\mathbf{Z}$ and the target image $\mathbf{Z}_t$, meanwhile minimize the number of EPE violations.*

III. Algorithms

A. Mathematical Formulation

Since we are seeking a pair of masks which can form printed image with high fidelity, the LDMO problem can be formulated as an optimization problem as follows.

$$\min_{\mathbf{M}_1,\mathbf{M}_2} \quad F = \|\mathbf{Z}_t - \mathbf{Z}\|_2^2 \tag{4}$$

$$\text{s.t.} \quad \mathbf{M}_1(x,y) \in \{0,1\}, \quad \forall x,y, \tag{4a}$$

$$\mathbf{M}_2(x,y) \in \{0,1\}, \quad \forall x,y, \tag{4b}$$

$$\mathbf{I}_1 = \sum_{k=1}^{K} w_k \cdot |\mathbf{M}_1 \otimes \mathbf{h}_k|^2, \tag{4c}$$

$$\mathbf{I}_2 = \sum_{k=1}^{K} w_k \cdot |\mathbf{M}_2 \otimes \mathbf{h}_k|^2, \tag{4d}$$

$$\mathbf{Z} = f_{resist}(\mathbf{I}_1) \vee f_{resist}(\mathbf{I}_2). \tag{4e}$$

The problem is strongly non-convex with discrete constraints, thus is hard to be solved directly. In this section, we propose a unified flow for solving the LDMO problem.

B. Numerical Optimization

Gradient-based method has been widely adopted in solving numerical optimization problems. However, there are non-differentiable discrete constraints in our formulation. Therefore, it is necessary to do relaxation before deriving the gradient.

In the formulation of LDMO, the variables $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{Z}_1$, and $\mathbf{Z}_2$ are binary, which are non-differentiable. One possible method is to relax them into floating values with a feasible region of [0,1], which cannot be solved by gradient-based method directly. Alternatively, the binary constraints can be replaced with *sigmoid* function for relaxation so that the variables become unconstrained, and hence convenient to derive the gradient. To apply *sigmoid* function, we need to introduce new variables $\mathbf{P}_1$ and $\mathbf{P}_2$. Then $\mathbf{M}_1$ and $\mathbf{M}_2$ can be relaxed by applying *sigmoid* function on $\mathbf{P}_1$ and $\mathbf{P}_2$, respectively (see Equations (5) and (6)).

$$\mathbf{M}_1(x,y) = \text{sig}(\mathbf{P}_1(x,y)) = \frac{1}{1+\exp[-\theta_M \mathbf{P}_1(x,y)]}, \tag{5}$$

$$\mathbf{M}_2(x,y) = \text{sig}(\mathbf{P}_2(x,y)) = \frac{1}{1+\exp[-\theta_M \mathbf{P}_2(x,y)]}. \tag{6}$$

Similarly, the *sigmoid* function can be applied to $\mathbf{I}_1$ and $\mathbf{I}_2$ to relax $\mathbf{Z}_1$ and $\mathbf{Z}_2$ as shown in Equations (7) and (8). θ_M and θ_Z are user-defined parameters which represent the steepness of *sigmoid* functions, and I_{th} is the threshold in the resist model.

$$\mathbf{Z}_1(x,y) = \text{sig}(\mathbf{I}_1(x,y)) = \frac{1}{1+\exp[-\theta_Z(\mathbf{I}_1(x,y) - I_{th})]}, \tag{7}$$

$$\mathbf{Z}_2(x,y) = \text{sig}(\mathbf{I}_2(x,y)) = \frac{1}{1+\exp[-\theta_Z(\mathbf{I}_2(x,y) - I_{th})]}. \tag{8}$$

Note that $\mathbf{Z}$ is also a binary value, but different from $\mathbf{Z}_1$ and $\mathbf{Z}_2$, it is calculated by logical OR. We relax constraint (4e) to

$$\mathbf{Z}(x,y) = \min\{\mathbf{Z}_1(x,y) + \mathbf{Z}_2(x,y), 1\}. \tag{9}$$

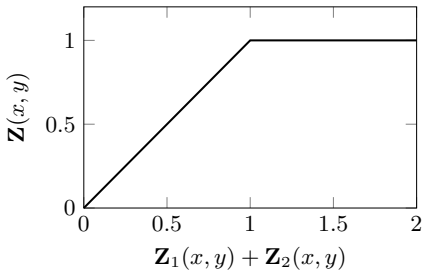


Fig. 3: Relation between $\mathbf{Z}(x,y)$ and $\mathbf{Z}_1(x,y) + \mathbf{Z}_2(x,y)$.

Considering that the maximum value of $\mathbf{Z}$ before relaxation is 1, here we set an upper bound to 1, which may reduce the error when calculating the objective value. The relation between $\mathbf{Z}(x,y)$ and $(\mathbf{Z}_1(x,y) + \mathbf{Z}_2(x,y))$ is shown in Fig. 3. Then it is easy to derive the gradient formulation of $\mathbf{Z}$ with respect to $\mathbf{Z}_1$ and $\mathbf{Z}_2$, denoted by $\mathbf{B}$, which is given by

$$\frac{\partial \mathbf{Z}(x,y)}{\partial \mathbf{Z}_1(x,y)} = \frac{\partial \mathbf{Z}(x,y)}{\partial \mathbf{Z}_2(x,y)} = \mathbf{B}(x,y) = \begin{cases} 1, & \text{if } \mathbf{Z}(x,y) \le 1, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

After relaxation, we can formulate the relaxed LDMO problem as follows.

$$\min_{\mathbf{P}_1,\mathbf{P}_2} \quad F = \|\mathbf{Z}_t - \mathbf{Z}\|_2^2 \tag{11}$$
$$\text{s.t. (4c)} - \text{(4d)}, \text{(5)} - \text{(9)}.$$

Now variables $\mathbf{P}_1$ and $\mathbf{P}_2$ are unconstrained, and functions in Equations (5)–(9) are differentiable. We obtain the gradient according to the chain rule.

$$\begin{aligned} \frac{\partial F}{\partial \mathbf{P}_1(x,y)} &= \frac{\partial \sum_{i,j}(\mathbf{Z}_t(i,j) - \mathbf{Z}(i,j))^2}{\partial \mathbf{P}_1(x,y)} \\ &= 2\sum_{i,j}(\mathbf{Z}(i,j) - \mathbf{Z}_t(i,j)) \cdot \frac{\partial \mathbf{Z}(i,j)}{\partial \mathbf{Z}_1(i,j)} \cdot \frac{\partial \mathbf{Z}_1(i,j)}{\partial \mathbf{P}_1(x,y)}, \end{aligned} \tag{12}$$

where

$$\begin{aligned} \frac{\partial \mathbf{Z}_1(i,j)}{\partial \mathbf{P}_1(x,y)} &= \theta_M \theta_Z \mathbf{Z}(i,j)(1 - \mathbf{Z}(i,j)) \\ &\times \{[\mathbf{M}_1(i,j) \otimes \mathbf{H}^*(i,j)]\mathbf{H}(i-x, j-y) \\ &+ [\mathbf{M}_1(i,j) \otimes \mathbf{H}(i,j)]\mathbf{H}^*(i-x, j-y)\} \\ &\times \mathbf{M}_1(i,j)[1 - \mathbf{M}_1(i,j)]. \end{aligned} \tag{13}$$

Then we can compute the gradient of F with respect to $\mathbf{P}_1$ and $\mathbf{P}_2$ as follows.

$$\begin{aligned} \nabla_{\mathbf{P}_1} F = 2\theta_M \theta_Z &\times \mathbf{M}_1 \odot (1 - \mathbf{M}_1) \odot \\ &\{\mathbf{H} \otimes [(\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{B} \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_1 \otimes \mathbf{H}^*)] + \\ &\mathbf{H}^* \otimes [(\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{B} \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_1 \otimes \mathbf{H})]\}, \end{aligned} \tag{14}$$

$$\begin{aligned} \nabla_{\mathbf{P}_2} F = 2\theta_M \theta_Z &\times \mathbf{M}_2 \odot (1 - \mathbf{M}_2) \odot \\ &\{\mathbf{H} \otimes [(\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{B} \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_2 \otimes \mathbf{H}^*)] + \\ &\mathbf{H}^* \otimes [(\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{B} \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_2 \otimes \mathbf{H})]\}. \end{aligned} \tag{15}$$

The numerical optimization algorithm is described in Algorithm 1. First we initialize $\mathbf{P}_1$ and $\mathbf{P}_2$, the maximum iteration number N and the tolerance ϵ (line 1). An intuitive initial solution is that $\mathbf{P}_1$ and $\mathbf{P}_2$ are initialized so that the corresponding $\mathbf{M}_1$ and $\mathbf{M}_2$ are identical to the target image. In each iteration, the printed image is obtained

Algorithm 1 Numerical Optimization Flow

1: Initialize $\mathbf{P}_1$, $\mathbf{P}_2$, maximum iteration N, tolerance ϵ;
2: **for** $i = 1, \cdots, N$ **do**
3: Compute the printed image according to current $\mathbf{P}_1$ and $\mathbf{P}_2$;
4: **if** $mod(i, w) == 0$ and printed image is illegal **then**
5: Discrete optimization ; ▷ Section III-D
6: **else**
7: `MaskUpdate`($\mathbf{P}_1$, $\mathbf{P}_2$); ▷ Algorithm 2
8: **if** $RMS(\nabla_{\mathbf{P}_1} F) + RMS(\nabla_{\mathbf{P}_2} F) \leq \epsilon$ **then**
9: **break**;
10: **end if**
11: **end if**
12: **end for**

Algorithm 2 Gradient-Based Mask Update

1: **function** `MaskUpdate`($\mathbf{P}_1$, $\mathbf{P}_2$)
2: Initialize stepsize t;
3: Compute the relaxed masks $\mathbf{M}_1, \mathbf{M}_2$;
4: Compute $\mathbf{Z}$ according to current $\mathbf{P}_1$ and $\mathbf{P}_2$;
5: Compute the gradient $\nabla_{\mathbf{P}_1} F$, $\nabla_{\mathbf{P}_2} F$ through Equations (4c)–(4e), (14)–(15);
6: $\mathbf{P}_1 \leftarrow \mathbf{P}_1 - t \times \nabla_{\mathbf{P}_1} F$;
7: $\mathbf{P}_2 \leftarrow \mathbf{P}_2 - t \times \nabla_{\mathbf{P}_2} F$;
8: **return** $\mathbf{P}_1$,$\mathbf{P}_2$, $\nabla_{\mathbf{P}_1} F$, $\nabla_{\mathbf{P}_2} F$;
9: **end function**

based on variables $\mathbf{P}_1$ and $\mathbf{P}_2$ (line 3). The violation checking will be carried out by every w iterations, which will be introduced in Section III-C. If the printed image is illegal, a discrete optimization step will be executed, which will be introduced in Section III-D. Otherwise, the function `MaskUpdate` will be called to update the masks. To save the runtime, the loop will exit early if the sum of the root mean square (RMS) of the gradient is less than a tolerance ϵ, which indicates that the objective function may be very close to the optimal value. The loop will finally terminate when the maximum number of iteration N is achieved. In our implementation, w is set to 3. N is set to 20. θ_M and θ_Z are set to 85 and 4, respectively.

The procedure for mask update is described in Algorithm 2. To derive the gradient, the lithography simulation is conducted first to compute the corresponding printed image based on current masks (line 4). Next, gradient of objective F with respect to $\mathbf{P}_1$ and $\mathbf{P}_2$ are computed (line 5), followed by variables update (lines 6–7). In our implementation, the tolerance ϵ is set to 0.01 and stepsize t is set to 0.4.

C. Violation Detection

Since there may exist conflicting patterns in the mask, the printed image must be illegal and it is difficult to be legalized only through gradient-based optimization. Intuitively, the violation can be resolved if the violated patterns are assigned to different masks. However, we need to locate where the violations occur. To do so, a Hanan-like grid is built based on the geometry of the bounding box of each target image. Each bounding box shares the same centroid with the pattern inside. Considering that the further assignment is also based on the grid and extra pattern may be generated around the original pattern during the mask optimization, each bounding box is set to be a bit larger than the pattern. In our implementation, the extra width and extra height are both set to $20nm$. All the grids are then categorized into *pattern grid* and *spacing grid* depending on their positions on the target image. Different from conventional Hanan grid in which all the grids are aligned horizontally and vertically,

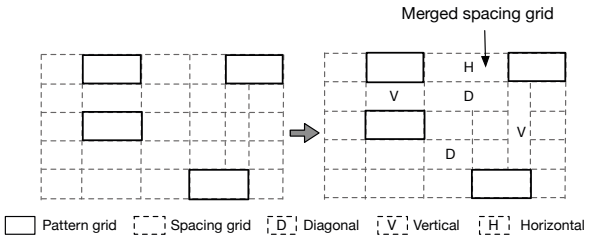


Fig. 4: Pattern grids and spacing grids.

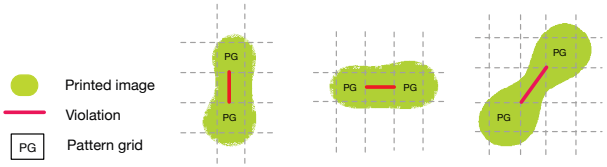


Fig. 5: Different kinds of violation.

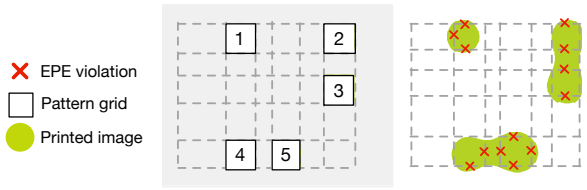
the adjacent pattern grids in our Hanan-like grid will be merged so that a single pattern will not be split by grids. In addition, the spacing grids between two patterns are also merged into one grid. The orientation of each merged spacing grid is set according to its relative position to the two pattern grids, as shown in Fig. 4. The H, V and D in Fig. 4 represent the orientation of horizontal, vertical and diagonal, respectively.

As mentioned before, the violation checking is conducted by every w iterations rather than by each iteration such that the efficiency of the whole flow is maintained. The violation detection is performed through the following way. The printed image is first mapped to the Hanan-like grids. Since all the violations happen at the region between patterns, i.e., the spacing grids, we only check the spacing grids between two patterns (see Fig. 5). Next, we initialize a matrix $\mathbf{A}$ with the same size of the grid so that each pixel in the grid corresponds to an entry of $\mathbf{A}$. Then we find all printable pixels in the grid and set the corresponding entries of the matrix to 1 and set the rest of the entries to 0. An intuitive observation is that a horizontal or vertical violation is caused by a printable line in spacing grids which connects an edge to the opposite side. Diagonal violation is due to the printable lines diagonally connecting two corners. For vertical or horizontal violation, we can check the sum of each row and each column of the matrix $\mathbf{A}$. Combined with the orientation, the violation can be determined. If the sum of one row is equal to the width W or the sum of one column is equal to the height H, there exists a violation. For diagonal violation, we compute the diagonal length of diagonal spacing grid. If the diagonal length is less than a spacing threshold, there is a diagonal violation. In our implementation, the threshold is set as $110nm$.

D. Discrete Optimization

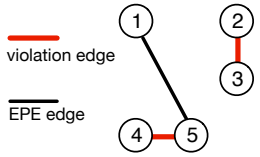
The masks $\mathbf{M}_1$ and $\mathbf{M}_2$ are updated in each iteration to reduce the objective value. Since LDMO is a highly non-convex problem, the gradient-based method can only achieve local optimum with poor quality. One reason is that the gradient-based method actually performs a greedy search and it is hard to escape from local optimum. To tackle this problem, we further propose a discrete optimization method to collaborate with the numerical optimization flow.

The main goal of LDMO problem is to resolve violations in the printed image and reduce EPE violation number. Generally, resolving violations can be achieved by assigning violated pattern



(a) (b)

Fig. 6: Graph construction based on printed image; (a) Pattern on the mask and lithography printed image with EPE violations; (b) Corresponding weighted graph.



$$\mathbf{W} = \begin{bmatrix} 0 & 0 & 0 & 0 & \beta \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \beta & 0 & 0 & 1 & 0 \end{bmatrix}$$

Fig. 7: The conflict graph and the corresponding weighted matrix $\mathbf{W}$.

grids to different masks. However, since the correlation between EPE violation and the distribution of patterns is unknown, it is difficult to derive a mathematical formulation to bridge the gap.

To overcome this issue, in this work we develop a discrete optimization approach seeking a two-way partitioning of the pattern grids considering image violation and EPE violation. Note that the proposed approach here can be easily extended to handle triple patterning lithography, where a three-way partitioning is adopted. With the position of printed image violation and the position of EPE violation, a weighted graph $G(V, E)$ can be constructed, where the vertex v_i represents i-th pattern grid and the edges with weight 1 connecting two vertices are conflict with each other. In addition, we add edges with weight β between the vertices which have large EPE, where $0 < \beta < 1$. Therefore, the objective of discrete optimization is to find a cut of the graph so that total weight of the edges between the cut and its complement is maximized. We use a vector $\mathbf{x}$ to denote the assignment of pattern grids, where $x_i = 1$ means v_i is assigned to mask 1 and $x_i = -1$ means v_i is assigned to mask 2. Then the two-way partitioning problem can be formulated as follows.

$$\max_{x} \quad \sum_{(i,j)\in E} w_{ij}(1 - x_i x_j) \tag{16}$$

$$\text{s.t.} \quad x_i \in \{-1, 1\}, \quad \forall v_i \in V, \tag{16a}$$

where w_{ij} defines the edge weight as follows.

$$w_{ij} = \begin{cases} 1, & \text{if } v_i \text{ and } v_j \text{ have conflict,} \\ \beta, & \text{if } v_i \text{ and } v_j \text{ both have large EPE,} \\ 0, & \text{otherwise.} \end{cases} \tag{17}$$

In our implementation, if the sum of EPE violations of two grids is greater than seven and they are not violated patterns, they will be connected by an edge of weight β, and β is set as 0.1.

Formulation in (16) can be approximated to a semidefinite programming (SDP) with below formulation, which can be solved efficiently while maintaining high accuracy.

$$\min_{\mathbf{X}} \quad \mathbf{W} \bullet \mathbf{X} \tag{18}$$

$$\text{s.t.} \quad \text{diag}(\mathbf{X}) = \mathbf{e}, \tag{18a}$$

$$\mathbf{X} \succeq \mathbf{0}, \tag{18b}$$

where $\mathbf{e} = [1, 1 \ldots, 1]^\intercal$.

The optimal solution $\mathbf{X}^*$ of Problem (18) need not to be in the form of $\mathbf{x}\mathbf{x}^\intercal$, and hence it does not yield a feasible solution to Problem (16) immediately. However, we can extract from $\mathbf{X}^*$ a solution via randomized rounding [28]. First, we compute Cholesky factorization $\mathbf{X}^* = \mathbf{U}^\intercal\mathbf{U}$ of $\mathbf{X}^*$. The i-th column of $\mathbf{U}$, denoted by $\mathbf{u}_i$, corresponds to the assignment of grid i. Let $\mathbf{r}$ be a vector uniformly distributed over the unit sphere (i.e., $\|\mathbf{r}\|_2 = 1$). Then we can set x_i as follows.

$$x_i = \text{sgn}(\mathbf{u}_i^\intercal \mathbf{r}) = \begin{cases} 1, & \text{if } \mathbf{u}_i^\intercal \mathbf{r} \geq 0, \\ -1, & \text{otherwise.} \end{cases} \tag{19}$$

In other words, we partition the grids according to whether their corresponding vectors lie "above" or "below" the hyperplane. The girds are therefore assigned to different masks according to the value of x_i.

An example of graph construction is given in Fig. 6 and Fig. 7. After solving the SDP, the solution $\mathbf{X}^*$ and $\mathbf{U}$ are given by

$$\mathbf{X}^* = \begin{bmatrix} 1 & 0 & 0 & 1 & -1 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & -1 & 1 \end{bmatrix}, \mathbf{U} = \begin{bmatrix} 1 & 0 & 0 & 1 & -1 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

With a random vector $\mathbf{r} = [r_1, r_2 \cdots r_5]^\intercal$ and Equation (19), vector $\mathbf{x}$ is obtained by

$$\mathbf{x} = [\text{sgn}(r_1), \text{sgn}(r_2), \text{sgn}(-r_2), \text{sgn}(r_1), \text{sgn}(-r_1)]^\intercal. \tag{20}$$

By solving SDP we can obtain multiple solutions which are useful to avoid being stuck in local optimum during the succeeding numerical optimization process. Furthermore, the runtime cost of solving SDP is much smaller than lithography simulation. Therefore, SDP is an efficient method for our discrete optimization. Once the SDP is solved, we can obtain multiple solutions for $\mathbf{x}$. Then all these obtained solutions are further optimized through numerical optimization flow without violation checking, where these solutions will go through a pruning process and suboptimal ones will be removed. The detailed procedure is shown in Algorithm 3. Firstly, an empty set P is initialized to store the potential solutions (line 1). Then S solutions are obtained by randomized rounding, from which we can get the corresponding assignment solution for two masks, i.e., $\mathbf{P}_1$ and $\mathbf{P}_2$. Each pair of $\mathbf{P}_1$ and $\mathbf{P}_2$ is treated as a 2-tuple which is stored in P (lines 2–6). Next, gradient-based mask update illustrated in Algorithm 3 is performed for T iterations for each solution in P. After that, the number of EPE violations of all the solutions will be compared and half of the solutions with larger EPE violation will be discarded (lines 8–14). The pruning process will be repeated until only one element in P is left. In order to balance the runtime and performance, T and S are both set to 5 in our implementation.

E. Overall Flow

From the problem formulation, it is clear that the LDMO is a strongly non-convex problem without analytic structure, which makes LDMO numerically hard to solve. In order to solve the problem, we design a unified optimization flow which includes a numerical optimization flow and a discrete optimization flow. These two engines are collaborative with each other. Basically, the masks are optimized numerically with gradient-based optimization. Once the violation is detected in the printed image, another SDP-based discrete optimization engine is triggered to resolve the violations. Multiple solutions are obtained from the solution of SDP, which can help to jump out of local optimum and act as a guidance of numerical

Algorithm 3 Pruning

Require: SDP solution $\mathbf{X}^*$.
Ensure: $\mathbf{P}_1$, $\mathbf{P}_2$.
1: Initialize set P;
2: **for** $i \leftarrow 1, \cdots, S$ **do**
3: Randomized rounding; ▷ Equation (19)
4: Get corresponding $\mathbf{P}_1^i$, $\mathbf{P}_2^i$; ▷ $\{\mathbf{P}_1, \mathbf{P}_2\}$ is a 2-tuple
5: Save $\{\mathbf{P}_1^i, \mathbf{P}_2^i\}$ in P;
6: **end for**
7: **while** P.`size()`> 1 **do**
8: **for all** $\{\mathbf{P}_1^i, \mathbf{P}_2^i\} \in$ P **do**
9: **for** $j \leftarrow 1, \cdots, T$ **do**
10: `MaskUpdate`$(\mathbf{P}_1^i, \mathbf{P}_2^i)$; ▷ Algorithm 2
11: **end for**
12: Get the number of EPE violations;
13: **end for**
14: Remove half of solutions with larger EPE violations;
15: **end while**
16: **return** the remaining $\{\mathbf{P}_1, \mathbf{P}_2\}$;

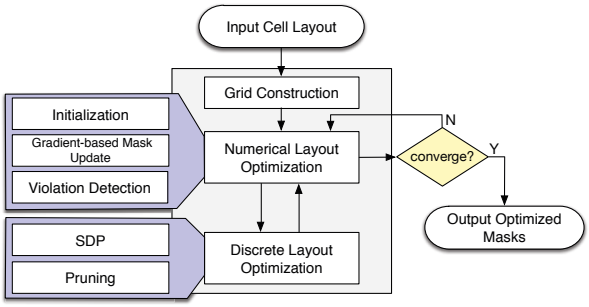


Fig. 8: Overall flow to solve LDMO problem.

optimization. The solutions returned by SDP will be numerically optimized until a pair of masks with highest quality is selected, which will be further optimized by numerical optimization flow. The overall flow is presented in Fig. 8.

IV. Experimental Results

We implement our algorithms with MATLAB on an Intel Core 2.6 GHz Linux machine with 48 GB RAM. To solve SDP we use CVX, a package for specifying and solving convex programs [29]. We use an open source lithography simulator and EPE checker [30], where the intensity threshold is set to 0.039. The EPE violation threshold value is set to $10nm$, which is more strict than the $15nm$ used in [30]. The experiments are conducted using NanGate, an open-source standard cell library [31]. We test the proposed unified framework on contact layers where stitches are forbidden. Fig. 9 gives an example of our output masks and the printed images for cell `OR2_X1`. We implement a layout decomposition engine, where branch-&-bound methodology is applied to search for all legal coloring solutions. In layout decomposition, the coloring distance is set to $110nm$, so all contact layers are double patterning friendly. We obtain a modified binary of mask optimization engine [23], where only EPE violation is optimized while the process variation (PV) band is ignored.

In the first experiment, we compare the proposed framework with an exhaustive optimization flow, where all legal layout decomposition solutions are enumerated, and all the solutions are fed into the mask optimization engine [23]. The results of the exhaustive optimization are shown in the merged column "ENUM + [23]" of TABLE II. The column "#LD" represents the total number of

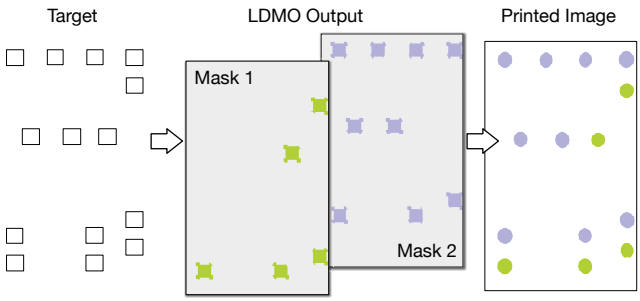


Fig. 9: Example of our layout decomposition and mask optimization on cell `OR2_X1`.

enumerated layout decomposition solutions. Considering that it will take extremely long time if we run mask optimization on all layout decomposition solutions, we set an upper bound of runtime, which is 36000 seconds (i.e., 10 hours). The column "#Complete" lists the total number of solutions that have been finished within the runtime limit. Then we can obtain the best decomposed layout with the least EPE violations. The columns "#EPEV" and "RT (s)" list the best EPE violation number and the total mask optimization runtime in seconds. Note that compared to expensive mask optimization, the runtime of layout decomposition is usually ignorable. The corresponding results of our unified framework are shown in the merged column "Ours". From the table we can see that, compared with the exhaustive optimization flow, our unified framework can effectively reduce the EPE violations number by 15%, meanwhile it can achieve more than 17× speed-up on average.

In order to avoid the unrealistic runtime cost of the exhaustive optimization, heuristic selection methods were proposed in previous work by Yu *et al.* [15] and Chen *et al.* [16]. In the second experiment, we use these two strategies to select from the exhaustive layout decomposition solutions, and feed the selected solutions to mask optimization engine of [23]. Then we compare the quality of corresponding printed patterns with ours. The corresponding results are shown in merged columns " [15] + [23]" and " [16] + [23]" in TABLE II. Here columns "#EPEV" and "RT (s)" represent the EPE violation number and the runtime of mask optimization on the selected layout decomposition solutions. From the table we can see that our proposed framework can achieve around 65% and 66% EPE violation reduction compared to the heuristic selection in [15] and [16], respectively. The experimental results show that the density based layout decomposition strategy may not promise an optimal printed image quality after mask optimization.

For comprehensive comparison among the three flows listed in TABLE II, we plot the distribution of EPE violations of all enumerated solutions of a cell according to the results of "ENUM+ [23]" (see Fig. 10). We can find that different solutions of layout decomposition result in diverse EPE cost after mask optimization. The solution obtained by different methods are marked in the figure. It can also be seen that among all the potential solutions, most coloring solutions are actually sub-optimal, while methods proposed in [15] and [16] do not select the optimal ones which correspond to the leftmost bar in each chart. Take Fig. 10(a) as an example. There are 22 DPLD solutions, while the number of EPE violations of these solutions ranges from 1 to 9, among which half of the solutions have 5 EPE violations, including the solutions selected by methods proposed in [15] and [16]. The masks generated by our method can achieve only 1 EPE violation, which is the same as the optimal solution of two-stage exhaustive optimization. From Fig. 10(c), we can see that the quality of the masks obtained by

TABLE II: Comparison between previous flow and unified flow.

Cell	ENUM + [23]				[15] + [23]		[16] + [23]		Ours	
	#LD	#Complete	#EPEV	RT (s)	#EPEV	RT (s)	#EPEV	RT (s)	#EPEV	RT (s)
INV_X1	10	10	**1**	11814	**1**	1183	**1**	1183	**1**	1995
NOR2_X1	18	18	2	25117	4	1405	8	1421	**1**	1996
BUF_X1	22	22	**1**	23072	5	1068	5	867	**1**	1990
CLKBUF_X1	30	30	1	30486	1	737	1	1046	**0**	1996
OAI211_X1	34	34	**3**	>36000	7	1207	5	1213	6	1996
AOI211_X1	42	31	3	>36000	8	1080	5	1048	**1**	1989
AND2_X1	52	23	3	>36000	13	1061	13	1080	**1**	1993
OR2_X1	82	33	**0**	>36000	3	1220	7	1046	**0**	1997
NAND4_X1	86	31	3	>36000	6	1176	5	1173	**1**	1997
NAND3_X2	252	35	6	>36000	8	774	7	1171	**3**	1998
OR4_X1	1282	33	1	>36000	2	1233	5	1188	**0**	1987
NOR3_X2	3016	37	**3**	>36000	4	1168	5	773	7	1999
OAI33_X1	6178	40	**4**	>36000	10	922	11	1017	6	1998
Average			2.55	>34595.35	6.09	1058.73	6.27	**1056.55**	**2.15**	1994.62
Ratio			1.18	>17.35	2.83	0.53	2.91	**0.53**	**1.00**	1.00

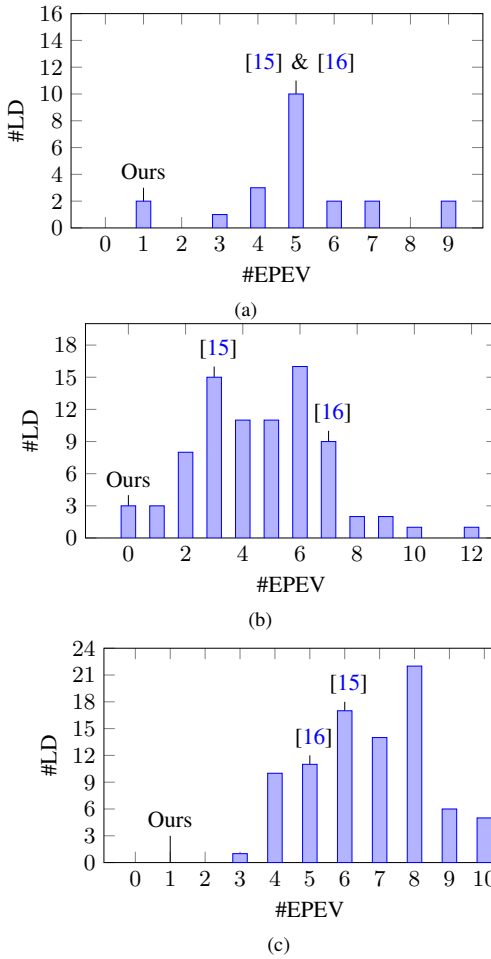


Fig. 10: (a) Distribution of #EPE violations of BUF_X1; (b) Distribution of #EPE violations of OR2_X1; (c) Distribution of #EPE violations of NAND4_X1.

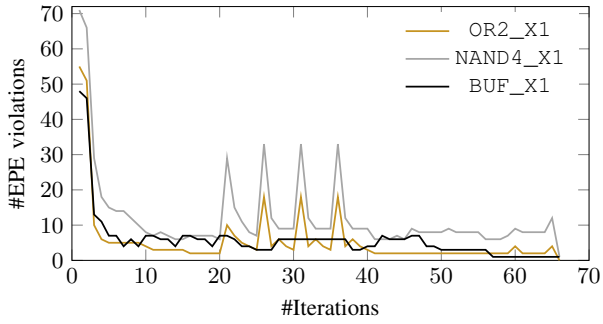


Fig. 11: Convergence of #EPE violations.

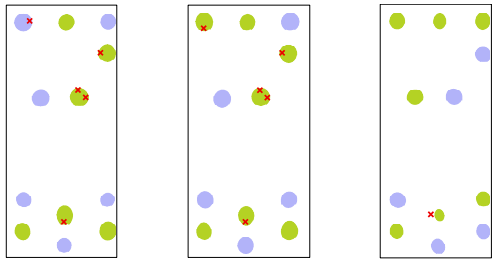


Fig. 12: Printed image of cell BUF_X1; (a) [15] + [23], #EPEV=5; (b) [16] + [23], #EPEV=5; (c) Ours, #EPEV=1.

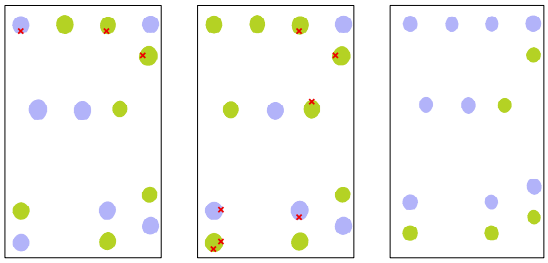
unified optimization can even outperform all the solutions obtained by conventional two-stage flow.

Fig. 11 demonstrates the convergence of the EPE violation number. Since the optimization process will not get stuck in local optimum as it will jump out, it can be seen that the number of EPE violations goes up on some iterations. Eventually, it will converge to a solution with fewer EPE violations.

Examples of printed image of the contact layer of standard cells are given in Figs. 12, 13 and 14 in which the EPE violations are marked with red cross on the pattern. It can be seen more explicitly that the proposed algorithm can find masks with higher quality, which have fewer EPE violations on printed image.

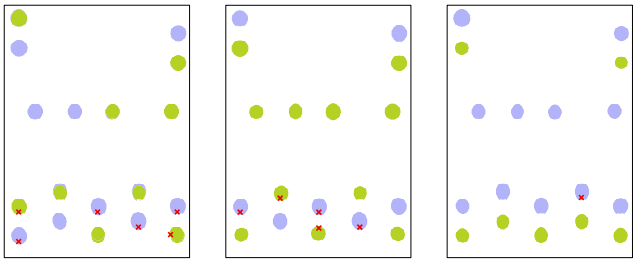
V. Conclusion

In this paper we have proposed a unified framework solving LDMO problem. In this framework, we designed two collaborative flows for optimization: a gradient based numerical optimization, as well as a set of discrete optimizations to jump out local optimum. The experimental results show that our proposed framework outperforms conventional flow in terms of both runtime and EPE



(a) (b) (c)

Fig. 13: Printed image of cell `OR2_X1`; (a) [15] + [23], #EPEV=3; (b) [16] + [23], #EPEV=7; (c) Ours, #EPEV=0.



(a) (b) (c)

Fig. 14: Printed image of cell `NAND4_X1`; (a) [15] + [23], #EPEV=6; (b) [16] + [23], #EPEV=5; (c) Ours, #EPEV=1.

violation number. Note that our framework is general and it can be extended for handling triple or quadruple patterning lithography coloring rules, as well as other mask optimization targets, such as PV band minimization.

Acknowledgment

The authors would like to thank Jing Su, Chenxi Lin, and Yi Zou from ASML for helpful comments.

References

[1] 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 (TCAD)*, vol. 32, no. 10, pp. 1453–1472, 2013.

[2] B. Yu, X. Xu, S. Roy, Y. Lin, J. Ou, and D. Z. Pan, "Design for manufacturability and reliability in extreme-scaling VLSI," *Science China Information Sciences*, pp. 1–23, 2016.

[3] A. B. Kahng, C.-H. Park, X. Xu, and H. Yao, "Layout decomposition approaches for double patterning lithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 29, pp. 939–952, June 2010.

[4] K. Yuan, J.-S. Yang, and D. Z. Pan, "Double patterning layout decomposition for simultaneous conflict and stitch minimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 29, no. 2, pp. 185–196, Feb. 2010.

[5] B. Yu, K. Yuan, D. Ding, and D. Z. Pan, "Layout decomposition for triple patterning lithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 34, no. 3, pp. 433–446, March 2015.

[6] Y. Xu and C. Chu, "GREMA: graph reduction based efficient mask assignment for double patterning technology," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2009, pp. 601–606.

[7] X. Tang and M. Cho, "Optimal layout decomposition for double patterning technology," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2011, pp. 9–13.

[8] B. Yu and D. Z. Pan, "Layout decomposition for quadruple patterning lithography and beyond," in *ACM/IEEE Design Automation Conference (DAC)*, 2014, pp. 53:1–53:6.

[9] 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 (TCAD)*, vol. 33, no. 3, pp. 397–408, March 2014.

[10] H. Tian, H. Zhang, Q. Ma, Z. Xiao, and M. D. F. Wong, "A polynomial time triple patterning algorithm for cell based row-structure layout," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2012, pp. 57–64.

[11] Y. Lin, X. Xu, B. Yu, R. Baldick, and D. Z. Pan, "Triple/quadruple patterning layout decomposition via novel linear programming and iterative rounding," in *SPIE Advanced Lithography*, vol. 9781, 2016.

[12] J. Kuang and E. F. Y. Young, "An efficient layout decomposition approach for triple patterning lithography," in *ACM/IEEE Design Automation Conference (DAC)*, 2013, pp. 69:1–69:6.

[13] H.-Y. Chang and I. H.-R. Jiang, "Multiple patterning layout decomposition considering complex coloring rules," in *ACM/IEEE Design Automation Conference (DAC)*, 2016, pp. 40:1–40:6.

[14] J. Kuang and E. F. Y. Young, "Fixed-parameter tractable algorithms for optimal layout decomposition and beyond," in *ACM/IEEE Design Automation Conference (DAC)*, 2017, pp. 61:1–61:6.

[15] 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 *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2013, pp. 163–169.

[16] Z. Chen, H. Yao, and Y. Cai, "SUALD: Spacing uniformity-aware layout decomposition in triple patterning lithography." in *IEEE International Symposium on Quality Electronic Design (ISQED)*, 2013, pp. 566–571.

[17] J.-S. Park, C.-H. Park, S.-U. Rhie, Y.-H. Kim, M.-H. Yoo, J.-T. Kong, H.-W. Kim, and S.-I. Yoo, "An efficient rule-based OPC approach using a DRC tool for 0.18 μm ASIC," in *IEEE International Symposium on Quality Electronic Design (ISQED)*, 2000, pp. 81–85.

[18] A. Awad, A. Takahashi, S. Tanaka, and C. Kodama, "A fast process variation and pattern fidelity aware mask optimization algorithm," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 238–245.

[19] J. Kuang, W.-K. Chow, and E. F. Y. Young, "A robust approach for process variation aware mask optimization," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 2015, pp. 1591–1594.

[20] Y.-H. Su, Y.-C. Huang, L.-C. Tsai, Y.-W. Chang, and S. Banerjee, "Fast lithographic mask optimization considering process variation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 35, no. 8, pp. 1345–1357, 2016.

[21] A. Poonawala and P. Milanfar, "Mask design for optical microlithography–an inverse imaging problem," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 774–788, 2007.

[22] N. Jia and E. Y. Lam, "Machine learning for inverse lithography: using stochastic gradient descent for robust photomask synthesis," *Journal of Optics*, vol. 12, no. 4, pp. 045 601:1–045 601:9, 2010.

[23] J.-R. Gao, X. Xu, B. Yu, and D. Z. Pan, "MOSAIC: Mask optimizing solution with process window aware inverse correction," in *ACM/IEEE Design Automation Conference (DAC)*, 2014, pp. 52:1–52:6.

[24] A. Poonawala and P. Milanfar, "Double-exposure mask synthesis using inverse lithography," *Journal of Micro/Nanolithography, MEMS, and MOEMS (JM3)*, vol. 6, no. 4, pp. 043 001–043 001, 2007.

[25] X. Li, G. Luk-Pat, C. Cork, L. Barnes, and K. Lucas, "Double-patterning-friendly OPC," in *Proceedings of SPIE*, vol. 7274, 2009.

[26] S. Banerjee, K. B. Agarwal, and M. Orshansky, "Simultaneous OPC and decomposition for double exposure lithography," in *Proceedings of SPIE*, vol. 7973, 2011.

[27] N. B. Cobb, "Fast optical and process proximity correction algorithms for integrated circuit manufacturing," Ph.D. dissertation, University of California at Berkeley, 1998.

[28] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM*, vol. 42, no. 6, pp. 1115–1145, 1995.

[29] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[30] S. Banerjee, Z. Li, and S. R. Nassif, "ICCAD-2013 CAD contest in mask optimization and benchmark suite," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2013, pp. 271–274.

[31] "NanGate FreePDK45 Generic Open Cell Library," http://www.si2.org/openeda.si2.org/projects/nangatelib, 2008.