GAN-OPC: Mask Optimization with Lithography-guided Generative Adversarial Nets

Haoyu Yang
CSE Department, CUHK
hyyang@cse.cuhk.edu.hk

Shuhe Li
CSE Department, CUHK
shli@cse.cuhk.edu.hk

Yuzhe Ma
CSE Department, CUHK
yzma@cse.cuhk.edu.hk

Bei Yu
CSE Department, CUHK
byu@cse.cuhk.edu.hk

Evangeline F. Y. Young
CSE Department, CUHK
fyyoung@cse.cuhk.edu.hk

ABSTRACT

Mask optimization has been a critical problem in the VLSI design flow due to the mismatch between the lithography system and the continuously shrinking feature sizes. Optical proximity correction (OPC) is one of the prevailing resolution enhancement techniques (RETs) that can significantly improve mask printability. However, in advanced technology nodes, the mask optimization process consumes more and more computational resources. In this paper, we develop a generative adversarial network (GAN) model to achieve better mask optimization performance. We first develop an OPC-oriented GAN flow that can learn target-mask mapping from the improved architecture and objectives, which leads to satisfactory mask optimization results. To facilitate the training process and ensure better convergence, we also propose a pre-training procedure that jointly trains the neural network with inverse lithography technique (ILT). At convergence, the generative network is able to create quasi-optimal masks for given target circuit patterns and fewer normal OPC steps are required to generate high quality masks. Experimental results show that our flow can facilitate the mask optimization process as well as ensure a better printability.

1 INTRODUCTION

With the VLSI technology node continuously shrinking down, the mask optimization process becomes a great challenge for designers [1, 2]. Conventional mask optimization process is illustrated in Figure 1, where OPC aims at compensating lithography proximity effects through correcting mask pattern shapes and inserting assist features. OPC methodologies include model-based techniques [3–5] and inverse lithography-based technique (ILT) [6–8].

In model-based OPC flows, pattern edges are fractured into segments which are then shifted/corrected according to mathematical models. A high printability mask can then be obtained with sub-resolution assist features (SRAF) [9]. Awad *et al.* [3] propose a pattern fidelity aware mask optimization algorithm that optimizes core polygons by simultaneously shifting adjacent segmentations. Su *et al.* [5] significantly accelerate the OPC flow by extracting representative process corners while maintaining a good wafer image quality. However, model-based OPC flows are highly restricted by their solution space and hence lacking in reliability for complicated designs. On the other hand, ILTs minimize the error between the wafer image and the target with lithography constraints. Because ILTs conduct pixel-based optimization on layout masks, they are expected to offer better lithography contour quality. Recently, Ma *et al.* [8] adopt ILT to simultaneously perform mask optimization and layout decomposition that brings a better solution of multiple patterning mask design. Although the model-based method and the ILT-based method behave well on a variety of designs, they take the wafer image as a mask update criterion in each iteration of the OPC process. In other works, multiple rounds of lithography simulation are indispensable in the optimization flow which is drastically time consuming.

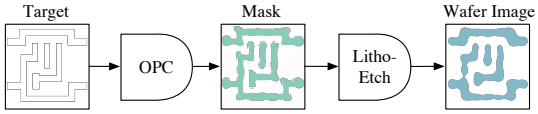


Figure 1: Conventional OPC flow and lithography process, where OPC is very time consuming.

The explosion of machine learning techniques have dramatically changed the way to solve design for manufacturability problems. Recently, both shallow and deep learning models have been successfully utilized to estimate mask printability accurately and efficiently (e.g. [10–13]). There are also several attempts on mask optimization problems that contain more complex regression or classification procedures. Matsunawa *et al.* [14] conduct segment based pattern correction with hierarchical Bayes model. Gu *et al.* [15] introduce discrete cosine transform (DCT) features and linear regression to predict fragment movement. [16, 17] incorporate artificial neural networks to estimate mask patterns. However, existing machine learning models can only perform pixel-wise or segment-wise mask calibration that is not computationally efficient.

Generative adversarial networks (GAN) has shown powerful generality when learning the distribution of a given dataset [18]. The basic optimization flow of GAN contains two networks interacting with each other. The first one is called **generator** that takes random vectors as input and generates samples which are as much closer to the true dataset distribution as possible. The second one is called **discriminator** that tries to distinguish the true dataset from the generated samples. At convergence, ideally, the generator is expected to generate samples that have the same distribution as true dataset. Inspired by the generative architecture and the adversarial training strategy, in this work we propose a lithography-guided generative framework that can synthesis quasi-optimal mask with single round forwarding calculation. The quasi-optimal mask can be further refined by few steps of normal OPC engine. It should be noted conventional GAN cannot be directly applied here, due to the following two reasons. (1) Traditional GANs are trained to mimic a dataset distribution which is not enough for the target-mask mapping procedure. (2) Compensation patterns or segment movements in the mask are derived based upon a large area of local patterns (e.g. $1000 \times 1000nm^2$) that brings much training pressure on the generator.

In accordance with these problems, we develop customized GAN training strategies for the purpose of mask optimization. Besides, since

DAC '18, June 24–29, 2018, San Francisco, CA, USA

ACM ISBN 978-1-4503-5700-5/18/06…$15.00
https://doi.org/10.1145/3195970.3196056

layout topology types are limited within specific area, we automatically synthesis local topology patterns based on size and spacing rules. The benefits of the artificial patterns are twofold: (1) we avoid to train the neural network with large images and facilitate the training procedure significantly; (2) automatically designed patterns are distributed uniformly and to some extent alleviate the over-fitting problem. Observing that most ILTs update the mask through steepest descent that resembles the training procedure in neural networks, we connect an ILT structure with the generative networks and pre-train the generator through back-propagating the lithography error to neuron weights. With the above pre-training phase, the generative model converges faster than training from random initialized neuron weights. The main contributions of this paper are listed as follows:

- We synthesis training topologies to enhance the computational efficiency and alleviate the over-fitting problem.
- We propose an ILT-guided pre-training flow to initialize the generator which can effectively facilitate the training procedure.
- We design new objectives of the discriminator to make sure the model is trained toward a target-mask mapping instead of a distribution.
- Experimental results show that our framework can significantly facilitate the mask optimization procedure as well as generating mask that has better printability under nominal condition.

The rest of the paper is organized as follows. Section 2 lists basic concepts and problem formulation. Section 3 discusses the details of the framework and training strategies. Section 4 presents experimental results, followed by conclusion in section Section 5.

2 PRELIMINARIES

In this section, we will discuss some preliminaries of mask optimization and the generative adversarial nets. Throughout this paper, we use $\mathbf{Z}_t$ to represent the target layout, $\mathbf{M}$ for the mask, $\mathbf{I}$ for the aerial image, $\mathbf{Z}$ for the wafer image, $\mathbf{G}$ for the generator output, $\mathbf{D}$ for the discriminator output and p_x for some distribution. We also denote operations "$\otimes$" and "$\odot$" as convolution and element-wise product, respectively. In order to avoid confusion, all the norms $||\cdot||$ are calculated with respect to flattened vectors.

Hopkins theory of the partially coherence imaging system has been widely applied to mathematically analyze the mask behavior of lithography [19]. Because the Hopkins diffraction model is complex and not computational-friendly, [20] adopts the singular value decomposition (SVD) to approximate the original model with a weighted summation of coherent systems.

$$\mathbf{I} = \sum_{k=1}^{N^2} w_k |\mathbf{M} \otimes \mathbf{h}_k|^2, \tag{1}$$

where $\mathbf{h}_k$ and w_k are the k^{th} kernel and its weight. As suggested in [7], we pick the N_h^{th} order approximation to the system. Equation (1) becomes,

$$\mathbf{I} = \sum_{k=1}^{N_h} w_k |\mathbf{M} \otimes \mathbf{h}_k|^2. \tag{2}$$

We pick $N_h = 24$ in our experiments. The lithography intensity corresponds to the exposure level on the photo resist, which controls the final wafer image with a photo resit model (Equation (3)).

$$\mathbf{Z}(x,y) = \begin{cases} 1, & \text{if } \mathbf{I}(x,y) \geq I_{th}, \\ 0, & \text{if } \mathbf{I}(x,y) < I_{th}. \end{cases} \tag{3}$$

Mask quality is evaluated through the fidelity of its wafer image with respect to the target image. Edge placement error (EPE), bridge and neck are three main types of defect detectors that are adopted in a layout printability estimation flow. As shown in Figure 2, EPE measures horizontal or vertical distances from given points (i.e. OPC control points) on target edges to lithography contours. Neck detector checks the error of critical dimensions of lithography contours compared to target patterns, while bridge detector aims to find unexpected short of wires. Note that unlike EPE violations, bridge and neck defects can appear in any directions. Because EPE violations could happen with good critical dimension and neck or bridge occurs with small EPE, none of these defect types individually can be an ideal representation of mask printability. Considering the objective of mask optimization is to make sure the remaining patterns after lithography process are as close as target patterns, we pick the squared L_2 error as the metric of lithography quality since a smaller L_2 indicates a better wafer image quality.

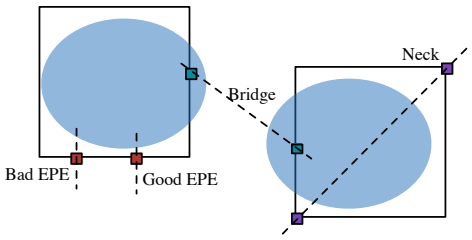


Figure 2: Different types of defects. Same lithography images result in different EPE violation counts due to different choices of measurement points. Some defects are not detectable through merely checking edge placement errors.

Definition 1 (Squared L_2 Error)**.** Let $\mathbf{Z}_t$ and $\mathbf{Z}$ as target image and wafer image respectively, the squared L_2 error of $\mathbf{Z}$ is given by $||\mathbf{Z}_t - \mathbf{Z}||_2^2$.

Following above terminologies, we define the mask optimization problem as follows.

Problem 1 (Mask Optimization)**.** Given a target image $\mathbf{Z}_t$, the objective of the problem is generating the corresponding mask $\mathbf{M}$ such that remaining patterns $\mathbf{Z}$ after lithography process is as close as $\mathbf{Z}_t$ or, in other word, minimizing the squared L_2 error of lithography images.

3 GAN-OPC FRAMEWORK

A classical GAN architecture comprises a generator and a discriminator. The generator accepts random vectors $\mathbf{z} \sim p_z$ as the input and generates samples $\mathbf{G}(\mathbf{z}; \mathbf{W}_g)$ that follows some distribution p_g, where $\mathbf{G}$ is a convolutional neural networks parameterized by $\mathbf{W}_g$. The discriminator acts as a classifier that distinguishes $\mathbf{G}(\mathbf{z}; \mathbf{W}_g)$ and the instance drawn from a data distribution p_d. The output $\mathbf{D}(\mathbf{x}; \mathbf{W}_d)$ represents the probabilities of $\mathbf{x}$ drawn from p_d and p_g. It should be noted that the original settings are not well suitable for the mask optimization problem. In this section, we will introduce the details of our framework including OPC-oriented GAN architecture and advanced training strategies.

3.1 Generator Design

From the previous discussion we can notice that the generator learns a distribution of a given dataset, which is originally designed as a mapping function $\mathbf{G} : p_z \rightarrow p_g$, where p_z is a distribution that input vectors are drawn and p_g denotes the distribution of the training set. The objective of the generator is to generate samples that deceive the discriminator as much as possible, as in Equation (4):

$$\max \mathbb{E}_{\mathbf{z} \sim p_z}[\log(\mathbf{D}(\mathbf{G}(\mathbf{z})))], \tag{4}$$

which maximizes the log-likelihood of the discriminator giving predictions that generated samples are real. Correspondingly, the generator

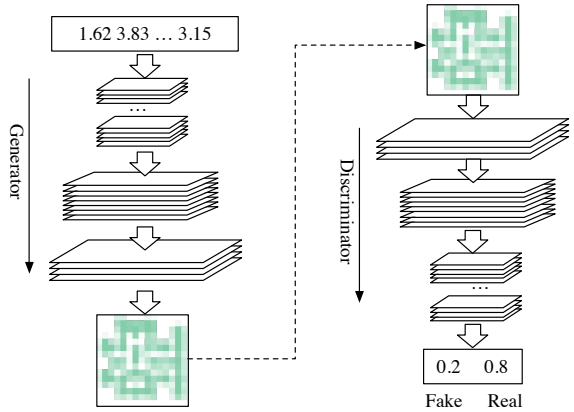


Figure 3: Conventional GAN architecture.

comprises a deconvolutional architecture that casts 1D vectors back to 2D images through stacked deconvolution operations, as shown in Figure 3.

Our framework, however, is expected to perform mask optimization on given target circuit patterns and obviously violates the deconvolutional architecture. To resolve this problem, we design a generator based on auto-encoder [21] which consists of an encoder and a decoder subnets. As depicted in Figure 4, the encoder is a stacked convolutional architecture that performs hierarchical layout feature abstractions and the decoder operates in an opposite way that predicts the pixel-based mask correction with respect to the target based on key features obtained from the encoder.

3.2 Discriminator Design

The discriminator is usually an ordinary convolutional neural networks that perform classification to distinguish the generated samples from the given data samples as shown in Equation (5):

$$\max \mathbb{E}_{\mathbf{x}\sim p_d}[\log(\mathbf{D}(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim p_z}[\log(1-\mathbf{D}(\mathbf{G}(\mathbf{z})))]. \tag{5}$$

In this work, the discriminator predicts whether an input instance is the generated mask $\mathbf{M}$ or the reference mask $\mathbf{M}^*$. However, the discriminator in Equation (5) is necessary but not sufficient to ensure generator to obtain a high quality mask (Figure 3). Consider a set of target patterns $\mathcal{Z} = \{\mathbf{Z}_{t,i}, i = 1, 2, \ldots, N\}$ and a corresponding reference mask set $\mathcal{M} = \{\mathbf{M}_i^*, i = 1, 2, \ldots, N\}$. Without loss of generality, we use $\mathbf{Z}_{t,1}$ in the following analysis. Suppose the above GAN structure has enough capacity to be well trained, the generator outputs an mask $\mathbf{G}(\mathbf{Z}_{t,1})$ that optimizes the objective function as in Equation (4). Observe that $\log(\mathbf{D}(\mathbf{G}(\mathbf{Z}_{t,1})))$ reaches its maximum value as long as

$$\mathbf{G}(\mathbf{Z}_{t,1}) = \mathbf{M}_i^*, \forall i = 1, 2, \ldots, N. \tag{6}$$

Therefore, an one-one mapping between the target and the reference mask cannot be guaranteed with current objectives. To address above concerns, we propose a new classification scheme that predicts positive or negative labels on target-mask pairs that inputs of the discriminator will be either $(\mathbf{Z}_t, \mathbf{G}(\mathbf{Z}_t))$ or $(\mathbf{Z}_t, \mathbf{M}^*)$, as illustrated in Figure 4. Claim that $\mathbf{G}(\mathbf{Z}_t) \approx \mathbf{M}^*$ at convergence with new discriminator. We still assume enough model capacity and training time for convergence. The discriminator now performs prediction on target-mask pairs instead of masks. Because only pairs $\{\mathbf{Z}_{t,i}, \mathbf{M}_i^*\}$ are labeled as data, the generator can deceive the discriminator if and only if $\mathbf{G}(\mathbf{Z}_{t,i}) \approx \mathbf{M}_i^*, \forall i = 1, 2, \ldots, N$, where N is the total number of training instances.

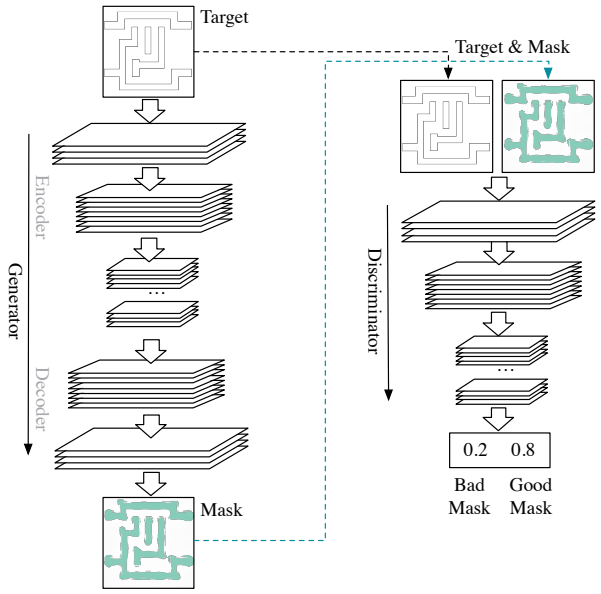


Figure 4: The proposed GAN-OPC architecture.

3.3 GAN-OPC Training

Based on the OPC-oriented GAN architecture in our framework, we tweak the objectives of $\mathbf{G}$ and $\mathbf{D}$ accordingly,

$$\max \mathbb{E}_{\mathbf{Z}_t\sim\mathcal{Z}}[\log(\mathbf{D}(\mathbf{Z}_t, \mathbf{G}(\mathbf{Z}_t)))], \tag{7}$$

$$\max \mathbb{E}_{\mathbf{Z}_t\sim\mathcal{Z}}[\log(\mathbf{D}(\mathbf{Z}_t, \mathbf{M}^*))] + \mathbb{E}_{\mathbf{Z}_t\sim\mathcal{Z}}[1-\log(\mathbf{D}(\mathbf{Z}_t, \mathbf{G}(\mathbf{Z}_t)))]. \tag{8}$$

In addition to facilitate the training procedure, we minimize the differences between generated masks and reference masks when updating the generator as in Equation (9).

$$\min \mathbb{E}_{\mathbf{Z}_t\sim\mathcal{Z}} ||\mathbf{M}^* - \mathbf{G}(\mathbf{Z}_t)||_n, \tag{9}$$

where $||\cdot||_n$ denotes the l_n norm. Combining (7), (8) and (9), the objective of our GAN model becomes

$$\min_{\mathbf{G}} \max_{\mathbf{D}} \mathbb{E}_{\mathbf{Z}_t\sim\mathcal{Z}}[1-\log(\mathbf{D}(\mathbf{Z}_t, \mathbf{G}(\mathbf{Z}_t))) + ||\mathbf{M}^* - \mathbf{G}(\mathbf{Z}_t)||_n^n] + \mathbb{E}_{\mathbf{Z}_t\sim\mathcal{Z}}[\log(\mathbf{D}(\mathbf{Z}_t, \mathbf{M}^*))]. \tag{10}$$

Previous analysis shows that the generator and the discriminator have different objectives, therefore the two sub-networks are trained alternatively, as shown in Figure 5(a) and algorithm 1. In each training iteration, we sample a mini-batch of target images (line 2); Gradients of both the generator and the discriminator are initialized to zero (line 3); A feed forward calculation is performed on each sampled instances (lines 4–5); The groundtruth mask of each sampled target image is obtained from OPC tools (line 6); We calculate the loss of the generator and the discriminator on each instance in the mini-batch (lines 7–8); We obtain the accumulated gradient of losses with respect to neuron parameters (lines 9–10); Finally the generator and the discriminator are updated by descending their mini-batch gradients (lines 11–12). Note that in Algorithm 1 we convert the min-max problem in Equation (10) into two minimization problems such that gradient ascending operations are no longer required to update neuron weights.

Algorithm 1 differs from traditional GAN optimization flow on the following aspects. (1) The generator plays as a mapping function from target to mask instead of merely a distribution, therefore the gradient of L_2 loss is back-propagated along with the information from the discriminator. (2) The discriminator functions as an alternative of ILT engine that determines only the quality of generated masks without any calibration operations. Besides, our combined input ensures that the discriminator will make positive prediction if and only if the generated

Algorithm 1 GAN-OPC Training

for number of training iterations **do**
  Sample m target clips $\mathcal{Z} \leftarrow \{\mathbf{Z}_{t,1}, \mathbf{Z}_{t,2}, \ldots, \mathbf{Z}_{t,m}\}$;
  $\Delta\mathbf{W}_g \leftarrow \mathbf{0}, \Delta\mathbf{W}_d \leftarrow \mathbf{0}$;
  for each $\mathbf{Z}_t \in \mathcal{Z}$ **do**
    $\mathbf{M} \leftarrow \mathbf{G}(\mathbf{Z}_t; \mathbf{W}_g)$;
    $\mathbf{M}^* \leftarrow$ Groundtruth mask of $\mathbf{Z}_t$;
    $l_g \leftarrow -\log(\mathbf{D}(\mathbf{Z}_t, \mathbf{M})) + \alpha||\mathbf{M}^* - \mathbf{M}||_2^2$;
    $l_d \leftarrow \log(\mathbf{D}(\mathbf{Z}_t, \mathbf{M})) - \log(\mathbf{D}(\mathbf{Z}_t, \mathbf{M}^*))$;
    $\Delta\mathbf{W}_g \leftarrow \Delta\mathbf{W}_g + \frac{\partial l_g}{\partial \mathbf{W}_g}$; $\Delta\mathbf{W}_d \leftarrow \Delta\mathbf{W}_d + \frac{\partial l_d}{\partial \mathbf{W}_g}$;
  end for
  $\mathbf{W}_g \leftarrow \mathbf{W}_g - \frac{\lambda}{m}\Delta\mathbf{W}_g$; $\mathbf{W}_d \leftarrow \mathbf{W}_d - \frac{\lambda}{m}\Delta\mathbf{W}_d$;
end for

mask is much close to the ground truth, which also helps train the generator better.

3.4 ILT-guided Pre-training

Although with OPC-oriented techniques, GAN is able to obtain a fairly good performance and training behavior, it is still a great challenge to train the complicated GAN model with satisfactory convergence. Observing that ILT and neural network training stage share similar gradient descent techniques, we develop an ILT-guided pre-training method to initialize the generator, after which the alternative mini-batch gradient descent is discussed as a training strategy of GAN optimization. The main objective in ILT is minimizing the lithography error through gradient descent.

$$E = ||\mathbf{Z}_t - \mathbf{Z}||_2^2, \tag{11}$$

where $\mathbf{Z}_t$ is the target and $\mathbf{Z}$ is the wafer image of a given mask. Because mask and wafer images are regarded as continuously valued matrices in the ILT-based optimization flow, we apply translated sigmoid functions to make the pixel values close to either 0 or 1.

$$\mathbf{Z} = \frac{1}{1 + \exp[-\alpha \times (\mathbf{I} - \mathbf{I}_{th})]}, \tag{12}$$

$$\mathbf{M}_b = \frac{1}{1 + \exp(-\beta \times \mathbf{M})}, \tag{13}$$

where $\mathbf{I}_{th}$ is the binarization threshold, $\mathbf{M}_b$ is the incompletely binarized mask, while α and β control the steepness of relaxed images.

Combine Equations (1)–(3), Equations (11)–(13) and the analysis in [6], we can derive the gradient representation as follows,

$$\begin{aligned}\frac{\partial E}{\partial \mathbf{M}} =& 2\alpha\beta \times \mathbf{M}_b \odot (1 - \mathbf{M}_b) \odot \\ & (((\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_b \otimes \mathbf{H}^*)) \otimes \mathbf{H} + \\ & ((\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_b \otimes \mathbf{H})) \otimes \mathbf{H}^*),\end{aligned} \tag{14}$$

where $\mathbf{H}^*$ is the conjugate matrix of the original lithography kernel $\mathbf{H}$. In traditional ILT flow, the mask can be optimized through iteratively descending the gradient until E is below a threshold.

The objective of mask optimization problem indicates the generator is the most critical component in GAN. Observing that both ILT and neural network optimization share similar gradient descent procedure, we propose a jointed training algorithm that takes advantages of ILT engine, as depicted in Figure 5(b). We initialize the generator with lithography-guided pre-training to make it converge well in the GAN optimization flow thereafter. The key step of neural network training is back-propagating the training error from the output layer to the input

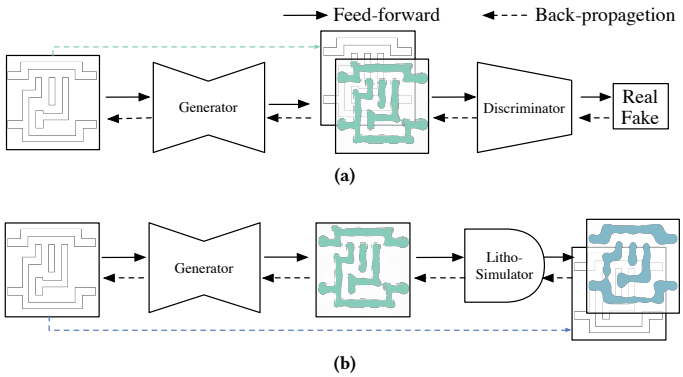


Figure 5: (a) GAN-OPC training and (b) ILT-guided pre-training.

layer while neural weights are updated as follows,

$$\mathbf{W}_g = \mathbf{W}_g - \frac{\lambda}{m}\Delta\mathbf{W}_g, \tag{15}$$

where $\Delta\mathbf{W}_g$ is accumulated gradient of a mini-batch of instances and m is the mini-batch instance count. Because Equation (15) is naturally compatible with ILT, if we create a link between the generator and ILT engine, the wafer image error can be back-propagated directly to the generator as presented in Figure 5.

The generator pre-training phase is detailed in Algorithm 2. In each pre-training iteration, we sample a mini-batch of target layouts (line 2) and initialize the gradients of the generator $\Delta\mathbf{W}_g$ to zero (line 3); The mini-batch is fed into the generator to obtain generated masks (lines 5). Each generated mask is loaded into the lithography engine to obtain a wafer image (line 6); The quality of wafer image is estimated by Equation (11) (lines 7); We calculate the gradient of lithography error E with respect to the neural networks parameter $\mathbf{W}_g$ through the chain rule, i.e., $\frac{\partial E}{\partial \mathbf{M}}\frac{\partial \mathbf{M}}{\partial \mathbf{W}_g}$ (line 8) ; Finally, $\mathbf{W}_g$ is updated following the gradient descent procedure (line 10).

Algorithm 2 ILT-guided Pre-training

1: **for** number of pre-training iterations **do**
2: Sample m target clips $\mathcal{Z} \leftarrow \{\mathbf{Z}_{t,1}, \mathbf{Z}_{t,2}, \ldots, \mathbf{Z}_{t,m}\}$;
3: $\Delta\mathbf{W}_g \leftarrow 0$;
4: **for** each $\mathbf{Z}_t \in \mathcal{Z}$ **do**
5: $\mathbf{M} \leftarrow \mathbf{G}(\mathbf{Z}_t; \mathbf{W}_g)$;
6: $\mathbf{Z} \leftarrow$ `LithoSim`$(\mathbf{M})$ ▷ Equations (2)–(3)
7: $E \leftarrow ||\mathbf{Z} - \mathbf{Z}_t||_2^2$;
8: $\Delta\mathbf{W}_g \leftarrow \Delta\mathbf{W}_g + \frac{\partial E}{\partial \mathbf{M}}\frac{\partial \mathbf{M}}{\partial \mathbf{W}_g}$; ▷ Equation (14)
9: **end for**
10: $\mathbf{W}_g \leftarrow \mathbf{W}_g - \frac{\lambda}{m}\Delta\mathbf{W}_g$; ▷ Equation (15)
11: **end for**

Compared to the training towards ground truth (i.e., directly back-propagate the mask error to neuron weights), ILT-guided pre-training provides step-by-step guidance when searching for a solution with high quality, which reduces the possibility of the generator being stuck at local minimum region in an early training stage. Because ILT contains complicated convolutions and matrix multiplications that are computational expensive, we approximate the pre-training stage through back-propagating errors of intermediate masks, which "guides" the generator towards optimality. We only adopt the ILT engine in the pre-training stages and replace it with the discriminator in the main training stage where the generator is optimized in an adversarial style.

Table 1: The design rules used.

Item	Min Size (*nm*)
M1 Critical Dimension	80
Pitch	140
Tip to tip distance	60

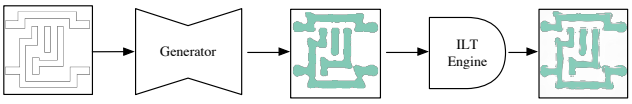


Figure 6: GAN-OPC flow: generator inference and ILT refinement.

4 EXPERIMENTAL RESULTS

The generative adversarial network for mask optimization is implemented based on `Tensorflow` [22] library and tested on single Nvidia Titan X. The lithography engine is based on the `lithosim_v4` package from ICCAD 2013 CAD Contest [23], which also provides ten industrial M1 designs on 32*nm* design node.

As a type of deep neural networks, GAN can be hardly well trained with only ten instances. To verify our framework, we synthesize a training layout library with 4000 instances based on the design specifications from existing 32*nm* M1 layout topologies. We adjust the wire sizes to make sure the shapes in synthesized layouts are similar to those in the given benchmark. To generate experimental cells, all the shapes are randomly placed together based on simple design rules, as detailed in Table 1. In addition, most generative models have shown obvious weakness in image details, which makes it extremely hard to optimize images with size 2048×2048. Therefore, we perform 8×8 average pooling on layout images before feeding them into the neural networks. In the generation stage we adopt simple linear interpolation to convert the layout images back to their original resolution.

The proposed GAN-OPC flow is illustrated in Figure 6, where we first feed target patterns into the generator and obtain the quasi-optimal masks, followed by refinement through an ILT engine. To verify the effectiveness of ILT-guided pre-training algorithm, we record training behaviors of two GANs which are denoted by GAN-OPC and PGAN-OPC. Here "GAN-OPC" and "PGAN-OPC" denote GAN-OPC flow without generator pre-training and GAN-OPC flow with ILT-guided pre-training, respectively. The training procedure is depicted in Figure 7, where x-axis indicates training steps and y-axis is L_2 loss between generator outputs and ground truth masks, as in Equation (9).

The training time for both GAN and PGAN are around 10 hours on our platform. Although L_2 loss of GAN-OPC drops slightly faster before 3000 iterations, the training curve shows that PGAN-OPC is a more stable training procedure and converges to a lower loss. Besides, it takes much more efforts for GAN-OPC to search a direction to descending the gradient fast, while the training loss of PGAN-OPC drops smoothly and converges at a lower L_2 loss than GAN-OPC, which indicates ILT-guided pre-training indeed facilitates mask-optimization-oriented GAN training flow. We will also show that PGAN-OPC exhibits better mask optimization results in the following section.

In the second experiment, we optimize the ten layout masks in ICCAD 2013 contest benchmark [23] and compare the results with previous work. Figure 8 depicts mask optimization results in comparison with an ILT engine [7], and the quantitative results are listed in Table 2. Here the wafer images are calculated from the simulation tool (`lithosim_v4`) in the contest [23]. Note that all the GAN-OPC and PGAN-OPC results are refined by an ILT engine which generates final masks to obtain wafer images. Column "L_2" is the squared L_2 error

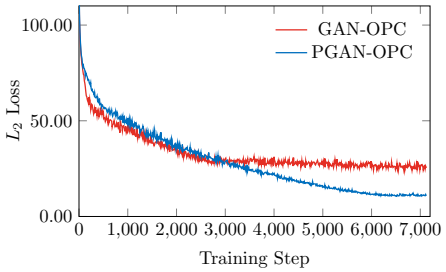


Figure 7: Training curves of GAN-OPC and PGAN-OPC.

between the wafer image and the target image under nominal condition. Column "PVB" denotes the contour area variations under ± 2% dose error. It is notable that GAN-OPC significantly reduces squared L_2 error of wafer images under the nominal condition by 9% and with the ILT-guided pre-training, squared L_2 error is slightly improved and PVB is further reduced by 1%. Because we only focus on the optimization flow under the nominal condition and no PVB factors are considered, our method only achieves comparable PVB areas among ten test cases. Additionally, feed-forward computation of GAN only takes 0.2*s* for each image which is ignorable, therefore runtime of our flow is almost determined by ILT refinements. Columns "RT (s)" lists the total mask optimization time of [7], GAN-OPC and PGAN-OPC. For most benchmark cases, GAN-OPC and PGAN-OPC show a earlier stop at a smaller L_2 error and, on average, reduce the optimization runtime by more than 50%. For most test cases, [7] exhibits a smaller PV band area possibly because the printed images are more likely to have large wafer image CD and shorter wire length, which makes masks suffer less proximity effects while inducing bridge or line-end pull back defects, as shown in Figure 9.

5 CONCLUSION

In this paper, we have proposed a GAN based mask optimization flow that takes target circuit patterns as input and generates quasi-optimal masks for further ILT refinement. We analyze the specialty of mask optimization problem and design OPC-oriented training objectives of GAN. Inspired by the observation that ILT procedure resembles gradient descent in back-propagation, we also develop an ILT-guided pre-training algorithm that initializes the generator with intermediate ILT results, which significantly facilitates the training procedure. Experimental results show that our framework not only accelerates ILT but also has the potential to generate better masks through offering better starting points in ILT flow.

ACKNOWLEDGMENTS

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

REFERENCES

[1] D. Z. Pan, B. Yu, and J.-R. Gao, "Design for manufacturing with emerging nanolithography," *IEEE 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. Awad, A. Takahashi, S. Tanaka, and C. Kodama, "A fast process variation and pattern fidelity aware mask optimization algorithm," in *Proc. ICCAD*, 2014, pp. 238–245.
[4] J. Kuang, W.-K. Chow, and E. F. Y. Young, "A robust approach for process variation aware mask optimization," in *Proc. DATE*, 2015, pp. 1591–1594.
[5] Y.-H. Su, Y.-C. Huang, L.-C. Tsai, Y.-W. Chang, and S. Banerjee, "Fast lithographic mask optimization considering process variation," *IEEE TCAD*, vol. 35, no. 8, pp. 1345–1357, 2016.
[6] 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.
[7] J.-R. Gao, X. Xu, B. Yu, and D. Z. Pan, "MOSAIC: Mask optimizing solution with process window aware inverse correction," in *Proc. DAC*, 2014, pp. 52:1–52:6.

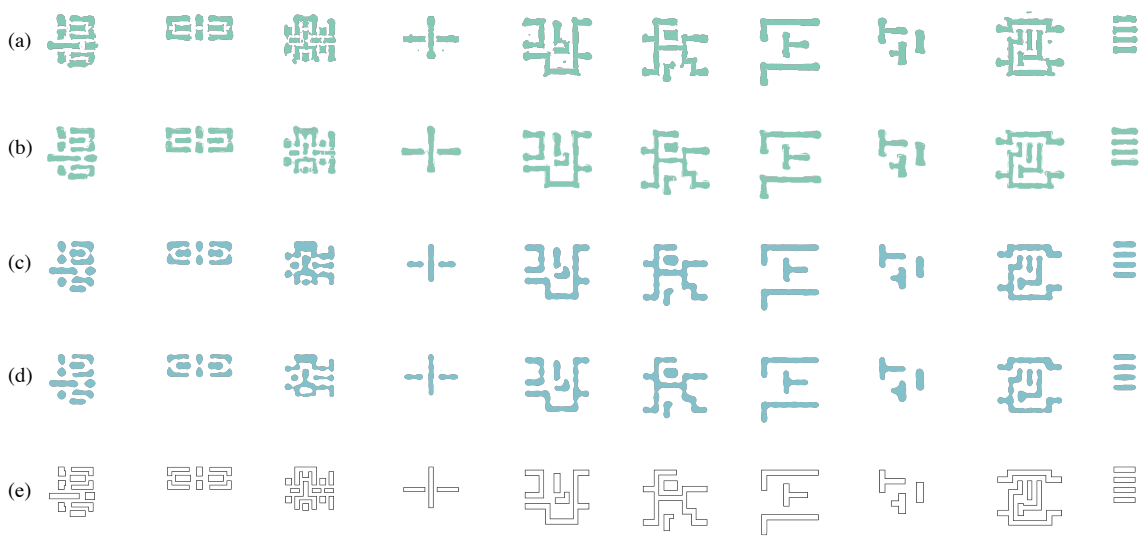
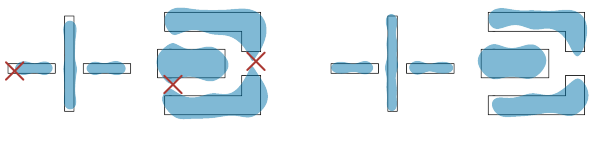


Figure 8: Result visualization of PGAN-OPC and ILT. Columns correspond to ten test cases from ICCAD 2013 CAD contest. Rows from top to bottom are: (a) masks of [7]; (b) masks of PGAN-OPC; (c) wafer images by masks of [7]; (d) wafer images by masks of PGAN-OPC; (e) target patterns.

Table 2: Comparison with state-of-the-art ILT solver.

Benchmarks		ILT [7]			GAN-OPC			PGAN-OPC		
ID	Area (nm^2)	L_2	PVB (nm^2)	RT (s)	L_2	PVB (nm^2)	RT (s)	L_2	PVB (nm^2)	RT (s)
1	215344	49893	65534	1280	54970	64163	380	52570	56267	358
2	169280	50369	48230	381	46445	56731	374	42253	50822	368
3	213504	81007	108608	1123	88899	84308	379	83663	94498	368
4	82560	20044	28285	1271	18290	29245	376	19965	28957	377
5	281958	44656	58835	1120	42835	59727	378	44733	59328	369
6	286234	57375	48739	391	44313	52627	367	46062	52845	364
7	229149	37221	43490	406	24481	47652	377	26438	47981	377
8	128544	19782	22846	388	17399	23769	394	17690	23564	383
9	317581	55399	66331	1138	53637	66766	427	56125	65417	383
10	102400	24381	18097	387	9677	20693	395	9990	19893	366
Average		44012.7	50899.5	788.5	40094.6	50568.1	384.7	39948.9	49957.2	371.3
Ratio		1.000	1.000	1.000	0.911	0.993	0.488	**0.908**	**0.981**	**0.471**



(a) (b)

Figure 9: Some wafer image details of (a) [7] and (b) PGAN-OPC.

[8] Y. Ma, J.-R. Gao, J. Kuang, J. Miao, and B. Yu, "A unified framework for simultaneous layout decomposition and mask optimization," in *Proc. ICCAD*, 2017, pp. 81–88.

[9] R. Viswanathan, J. T. Azpiroz, and P. Selvam, "Process optimization through model based SRAF printing prediction," in *SPIE Advanced Lithography*, vol. 8326, 2012.

[10] T. Matsunawa, J.-R. Gao, B. Yu, and D. Z. Pan, "A new lithography hotspot detection framework based on AdaBoost classifier and simplified feature extraction," in *Proc. SPIE*, vol. 9427, 2015.

[11] H. Zhang, B. Yu, and E. F. Y. Young, "Enabling online learning in lithography hotspot detection with information-theoretic feature optimization," in *Proc. ICCAD*, 2016, pp. 47:1–47:8.

[12] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: a deep learning approach," *JM3*, vol. 16, no. 3, p. 033504, 2017.

[13] H. Yang, J. Su, Y. Zou, B. Yu, and E. F. Y. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," in *Proc. DAC*, 2017, pp. 62:1–62:6.

[14] T. Matsunawa, B. Yu, and D. Z. Pan, "Optical proximity correction with hierarchical bayes model," *JM3*, vol. 15, no. 2, p. 021009, 2016.

[15] A. Gu and A. Zakhor, "Optical proximity correction with linear regression," *IEEE TSM*, vol. 21, no. 2, pp. 263–271, 2008.

[16] R. Luo, "Optical proximity correction using a multilayer perceptron neural network," *Journal of Optics*, vol. 15, no. 7, p. 075708, 2013.

[17] S. Choi, S. Shim, and Y. Shin, "Machine learning (ML)-guided OPC using basis functions of polar fourier transform," in *Proc. SPIE*, vol. 9780, 2016.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NIPS*, 2014, pp. 2672–2680.

[19] H. Hopkins, "The concept of partial coherence in optics," in *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 208, no. 1093. The Royal Society, 1951, pp. 263–277.

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

[21] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 52–59, 2011.

[22] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. OSDI*, 2016, pp. 265–283.

[23] S. Banerjee, Z. Li, and S. R. Nassif, "ICCAD-2013 CAD contest in mask optimization and benchmark suite," in *Proc. ICCAD*, 2013, pp. 271–274.