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

Haoyu Yang, Shuhe Li, Yuzhe Ma, Bei Yu and Evangeline F. Y. Young
Department of Computer Science and Engineering
The Chinese University of Hong Kong

GAN Basis
- \( x \): Sample from the distribution of target dataset; \( z \): Input of \( G \)
- \( G(z; \theta_G) \): Differentiable function represented by a multi-layer perceptron with parameters \( \theta_G \)
- Discriminator \( D(x; \theta_D) \): Represents the probability that \( x \) came from the data rather than \( G(z) \).

1. Train \( D \) to maximize the probability of assigning the correct label to both training examples and samples from \( G \).
2. Train \( G \) to minimize \( \log(1 - D(G(z))) \), i.e., generate fake samples that are drawn from similar distributions as \( \text{data}(x) \).

\[
\min_{\theta_G} \mathbb{E}_{z \sim \text{data}(z)} [\log(D(z))] + \mathbb{E}_{x \sim \text{data}(x)} [\log(1 - D(G(z)))]
\]

GAN Architecture

Generator Design
- Auto-encoder based generator which consists of an encoder and a decoder subnets.
- An encoder is a stacked convolutional architecture that performs hierarchical layout feature abstraction.
- A decoder operates in an opposite way that predicts the pixel-based mask correction with respect to the target.

Discriminator Design
- Take target-mask tuple as inputs: \((Z, G(Z))\) or \((Z, M')\).

GAN-OPC Architecture

GAN-OPC Training

Based on the OPC-oriented GAN architecture in our framework, we tweak the objectives of \( G \) and \( D \) accordingly.

\[
\max_{\theta_G} \mathbb{E}_{Z \sim \text{data}(z)} [\log(D(z, G(Z)))] + \mathbb{E}_{x \sim \text{data}(x)} [\log(1 - D(G(z), G(z)))]
\]

In addition to facilitate the training procedure, we minimize the differences between generated masks and reference masks when updating the generator as in Equation (12).

\[
\min_{\theta_G} \mathbb{E}_{z \sim \text{data}(z)} [\log(D(z, G(z))) + ||M' - G(Z)||_1]
\]

where \( || \cdot ||_1 \) denotes the \( l_1 \) norm. Combining (10), (11) and (12), the objective of our GAN model becomes

\[
\max_{\theta_G} \mathbb{E}_{Z \sim \text{data}(z)} [1 - \log(D(z, G(Z)))] + \mathbb{E}_{x \sim \text{data}(x)} [1 - \log(D(G(z), G(z)))]
\]

\[
+ \mathbb{E}_{z \sim \text{data}(z)} [||M' - G(Z)||_1]
\]

The generator and the discriminator are trained alternatively as follows.

The GAN-OPC Training Algorithm

1. for number of training iterations do
2. Sample \( m \) target clips \( z \leftarrow \{Z_1, Z_2, \ldots, Z_m\} \); 3. \( \Delta W_{G} \leftarrow 0 \), \( \Delta W_{D} \leftarrow 0 \); 4. for each \( Z_i \in Z \) do 5. \( M = G(Z_i, W_G) \); 6. \( M' = \) Groundtruth mask of \( Z_i \); 7. \( \Delta M = G(Z_i, W_G) - M' \); 8. \( \Delta W_{G} \leftarrow \Delta W_{G} + \frac{\partial E \mathcal{L}_{GAN}}{\partial W_{G}} \); 9. \( \Delta W_{D} \leftarrow \Delta W_{D} + \frac{\partial E \mathcal{L}_{GAN}}{\partial W_{D}} \); 10. end for 11. \( W_{G} \leftarrow W_{G} + \frac{\lambda}{m} \Delta W_{G} \); 12. end for

Experimental Results

The Dataset
- The lithography engine is based on the 11thobsia_v4 package from ICCAD 2013 CAD Contest.
- Manually generated 4000 instances based on the design specification from existing 32nm M1 layouts.

Mask Optimization Results

Visualizing PGAN-OPC and ILM:
(a) masks of GAN (DAC14); (b) masks of PGAN-OPC; (c) wafer images by masks of GAN (DAC14); (d) wafer images by masks of PGAN-OPC; (e) target patterns.