IEEE/ACM

ICCAD

2022 INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN

41st Edition

AdaOPC: A Self-Adaptive Mask Optimization Framework For Real Design Patterns

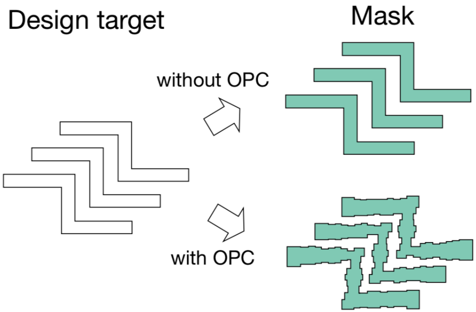
Wenqian Zhao[1], Xufeng Yao[1] Ziyang Yu[1], Guojin Chen[1], Yuzhe Ma[2], Bei Yu[1], Martin D.F. Wong[1]

[1] The Chinese University of Hong Kong
[2] HKUST(GZ)

Sept. 15, 2022

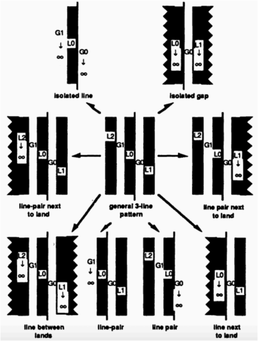
Background and Motivation

Background of Mask Optimization

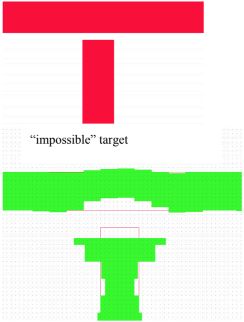


Background of Mask Optimization

- Rule-based OPC



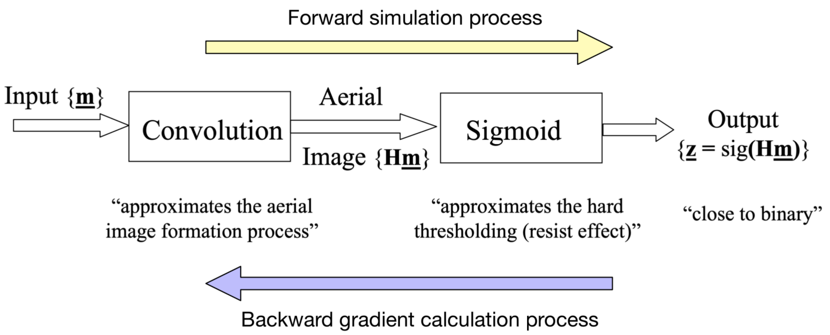
- Model-based OPC



Full image OPC corrections

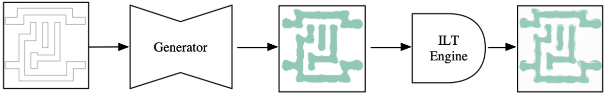
Background of Mask Optimization

- ILT-based method



1

- ML-based method



2

Deep learning model generate mask or initial mask for iterations

[2] Haoyu Yang et al. (2019). "GAN-OPC: Mask optimization with lithography-guided generative adversarial nets". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.10, pp. 2822–2834.

Background of Mask Optimization

All methods have certain drawbacks:

- Rule-based methods lack local fidelity
- Both model-based restricted by the solutions space in advanced technology nodes.
- ILT-based methods iteratively call the imaging system while optimizing an objective function which is time-consuming.
- ML-based OPCs have shown remarkable speed-up in the OPC flows, however not guaranteed to work for some critical patterns.

Lithography Simulation Model

- Hopkins diffraction model[3] decomposed into a sum of coherent systems:

$$\mathbf{I}(x,y) = \sum_{k=1}^{N^2} w_k |\mathbf{M}(x,y) \otimes h_k(x,y)|^2, \quad x,y = 1,2,...N \tag{1}$$

- h_k: k-th kernel, w_k:corresponding weight. "$\otimes$": convolution.

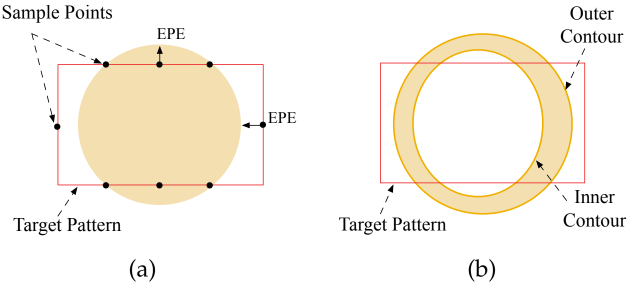
$$\mathbf{I}(x,y) \approx \sum_{k=1}^{K} w_k |\mathbf{M}(x,y) \otimes h_k(x,y)|^2, \tag{2}$$

- lithography intensity $\mathbf{I}$ sent to photoresist model to generate the final binary pattern $\mathbf{Z}$ with exposure resist threshold I_{th}:

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

[3] Harold Horace Hopkins (1951). "The concept of partial coherence in optics". In: *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 208.1093, pp. 263–277.

OPC Evaluation Criteria

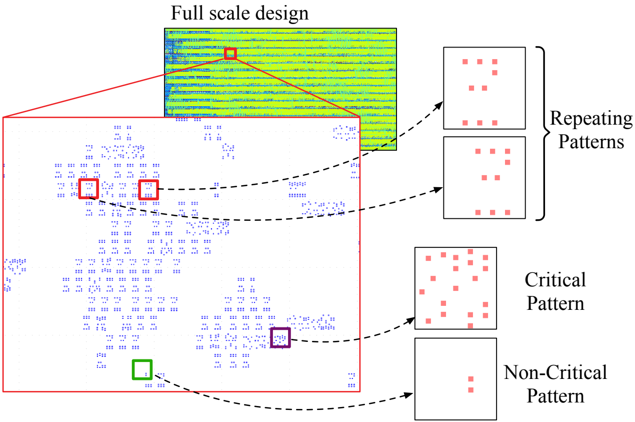


(a) Visualization of EPE measurement (b) Visualization of PVBand.

$$EPE\_violation(x,y) = \begin{cases} 1, & D(x,y) \geq th_{EPE}, \\ 0, & D(x,y) \leq th_{EPE}, \end{cases} \tag{4}$$

$$PVBand = \sum_{x,y}^{N^2} |\mathbf{Z}_{out} - \mathbf{Z}_{in}|, \tag{5}$$

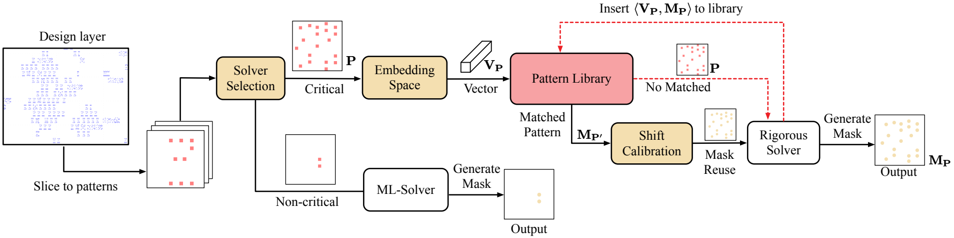
Motivations



- Patterns scattered unevenly with different complexity. → Solver selection
- Patterns have large ratio of repetition on a full layout. → Mask Reuse

Adaptive Framework

Workflow Overview



Main Contributions:

- Adaptive solver selection
- Mask reuse ← Critical Patterns
- Dynamic Pattern Library ← Fast Pattern Matching

Extensible OPC Solver Selection

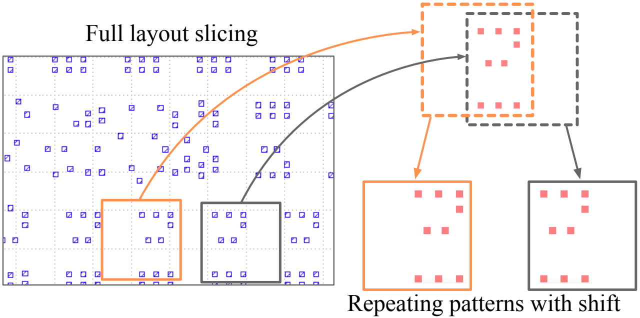
- Simple and Intuitive: Binary classification with cross-entropy loss L:

$$L = -\frac{1}{N}\sum_{i}^{N} y_i \log(p_i) + (1 - y_i)\log(1 - p_i), \tag{6}$$

- Solver pool extensible, modify loss by adding num of class:

$$L = -\frac{1}{N}\sum_{i}^{N}\sum_{c=1}^{C} y_{ic}\log(p_{ic}). \tag{7}$$

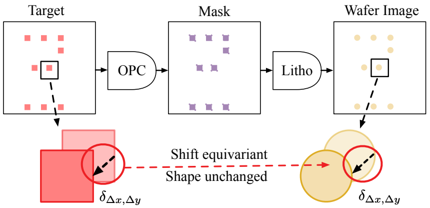
Mask Reuse



- Whether and how can an optimized mask with location shift be reused?
- How to match a same pattern accurately within an acceptable time?
- How to measure the geometric similarity of patterns with location shift?

Mask Reuse

- **Whether** and how can an optimized mask with location shift be reused?



- Shift Equivariance:

$$\delta_{\Delta x,\Delta y}(\mathbf{P}) = Litho(\delta_{\Delta x,\Delta y}(\mathbf{M}_{\mathbf{P}})). \tag{8}$$

We only need to calculate pattern shift since printed masks share same shift as corresponding patterns.

- Whether and **how** can an optimized mask with location shift be reused?
- Pattern shift calibration
  - Pixel-wise **cross-correlation** of $\mathbf{P}$ and $\mathbf{P}'$ reflects the pixel-wise similarity
  - Cross-correlation computation of two large 2-D pattern is time-comsuming.
  - Equal to convolution of $\mathbf{P}$ and $Rotate_{180^\circ}(\mathbf{P}')$.
  - Accelerated with Fast Fourier Transform (FFT)[4]:

$$\begin{aligned} x^*, y^* &= \underset{x,y}{\operatorname{argmax}}\ Conv\_FFT(\mathbf{P}, Rotate(\mathbf{P}')), \\ \Delta x = x^* - x_{ctr}&,\ \Delta y = y^* - y_{ctr}, \end{aligned} \tag{9}$$

[4]Nicolas Vasilache et al. (2014). "Fast convolutional nets with fbfft: A GPU performance evaluation". In: *arXiv preprint arXiv:1412.7580*.

Pattern Matching

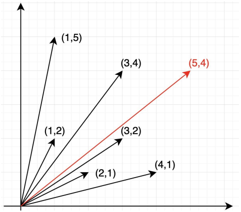
- How to match a same pattern accurately within an acceptable time?

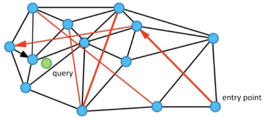
To tackle the problem of computation intensity of the rigorous method, we maintain a pattern library.

- store the features and optimized masks of previously encountered critical patterns
- use the result of saved masks with samiliar geometric structure as initial mask, hence reduce the iteration time

Pattern Matching

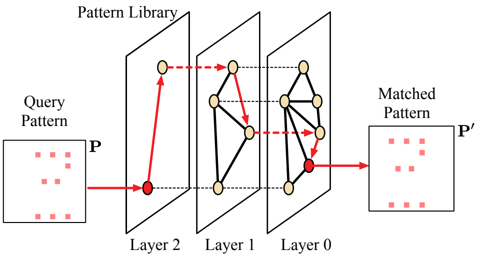
- How to match the patterns? - Pattern Library
  - Sparse neighborhood graph structure
  - Graph is divided into hierarchical layers





Hierarchical Navigable Small World (HNSW)[5]

- the overall number of distance computations is roughly proportional to a product of the average number of greedy algorithm hops by the average degree of the nodes on the greedy path



Algorithm 1

```
INSERT(hnsw, q, M, Mmax, efConstruction, mL)
Input: multilayer graph hnsw, new element q, number of established
connections M, maximum number of connections for each element
per layer Mmax, size of the dynamic candidate list efConstruction, nor-
malization factor for level generation mL
Output: update hnsw inserting element q
W ← ∅   // list for the currently found nearest elements
ep ← get enter point for hnsw
L ← level of ep   // top layer for hnsw
l ← ⌊-ln(unif(0..1))·mL⌋  // new element's level
for lc ← L … l+1
  W ← SEARCH-LAYER(q, ep, ef=1, lc)
  ep ← get the nearest element from W to q
for lc ← min(L, l) … 0
  W ← SEARCH-LAYER(q, ep, efConstruction, lc)
  neighbors ← SELECT-NEIGHBORS(q, W, M, lc) // alg. 3 or alg. 4
  add bidirectionall connections from neighbors to q at layer lc
  for each e ∈ neighbors   // shrink connections if needed
    eConn ← neighbourhood(e) at layer lc
    if │eConn│ > Mmax // shrink connections of e
                     // if lc = 0 then Mmax = Mmax0
      eNewConn ← SELECT-NEIGHBORS(e, eConn, Mmax, lc)
                                        // alg. 3 or alg. 4
      set neighbourhood(e) at layer lc to eNewConn
  ep ← W
if l > L
  set enter point for hnsw to q
```

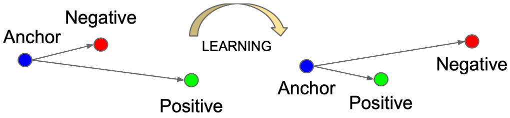

[5] Yu A Malkov and Dmitry A Yashunin (2018). "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs". In: *IEEE transactions on pattern analysis and machine intelligence* 42.4, pp. 824–836.

Pattern Embedding

- How to measure the geometric similarity of patterns with location shift?
- embedding for all critial patterns
  - positive samples are patterns that are same or similar
  - negative samples are patterns that are different
  - In learned embedding space, nearest neighbor tend to share similar geometric pattern
- similarity measurement: Euclidean Distance

$$d_{Euclid}(\mathbf{V}_{\mathbf{P}_1}, \mathbf{V}_{\mathbf{P}_2}) = \|\mathbf{V}_{\mathbf{P}_1} - \mathbf{V}_{\mathbf{P}_2}\|_2^2 = \sqrt{\sum_{i=0}^{k}(V_{\mathbf{P}_1,i} - V_{\mathbf{P}_2,i})^2}. \tag{10}$$

- Recap on Contrastive Learning



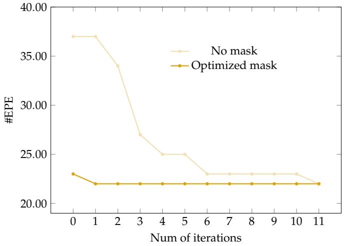
- Data Augmentation: Cropping and shifting
- Supervised Contrastive Loss[6]:

$$\mathcal{L}_{supCon} = -\sum_{i \in I} \frac{1}{|J(i)|} \sum_{j \in J(i)} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{\sum_{a \in A(i)} \exp(\mathbf{z}_i \cdot \mathbf{z}_a / \tau)}, \quad (11)$$

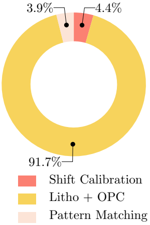
[6] Prannay Khosla et al. (2020). "Supervised contrastive learning". In: *Advances in Neural Information Processing Systems* 33, pp. 18661–18673.

Experimental Results

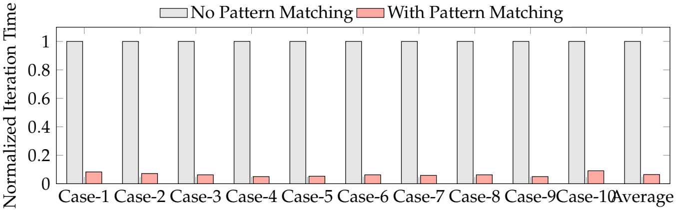
Results



(a) EPE convergence comparison



(b) Runtime breakdown



Mask convergence speed comparison with/without Pattern Matching.

Results

Table: Comparisons of baseline approaches

Test Case ID	DAMO-DGS[7] #EPE	DAMO-DGS[7] PVB (nm^2)	DAMO-DGS[7] RT (s)	ILT-GPU[8] #EPE	ILT-GPU[8] PVB (nm^2)	ILT-GPU[8] RT (s)	AdaOPC #EPE	AdaOPC PVB (nm^2)	AdaOPC RT (s)
1	22	23323	5.20	23	23329	41.15	22	23232	5.50
2	26	26729	5.26	25	26762	48.5	24	26580	5.41
3	27	26938	5.22	24	26720	55.92	24	26718	5.37
4	36	27975	5.18	29	28127	70.57	25	27934	5.40
5	35	28805	5.32	30	28925	66.89	30	28927	5.44
6	30	26960	5.31	25	26762	55.81	24	26775	5.38
7	33	26382	5.23	28	26453	59.47	28	26281	5.43
8	32	30646	5.38	25	29450	54.88	27	29341	5.42
9	25	24054	5.25	24	24053	70.62	23	24022	5.43
10	24	21939	5.29	23	21701	37.59	22	21644	5.53
Avg.	29.0	26375	**5.26**	25.6	26228	56.14	**24.9**	**26145**	5.43
Ratio	1.165	1.009	**0.970**	1.028	1.003	10.340	**1.000**	**1.000**	1.000

[7]Guojin Chen et al. (2021). “Damo: Deep agile mask optimization for full chip scale”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

[8]Jhih-Rong Gao et al. (2014). “MOSAIC: Mask optimizing solution with process window aware inverse correction”. In: *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, pp. 1–6.

THANK YOU!