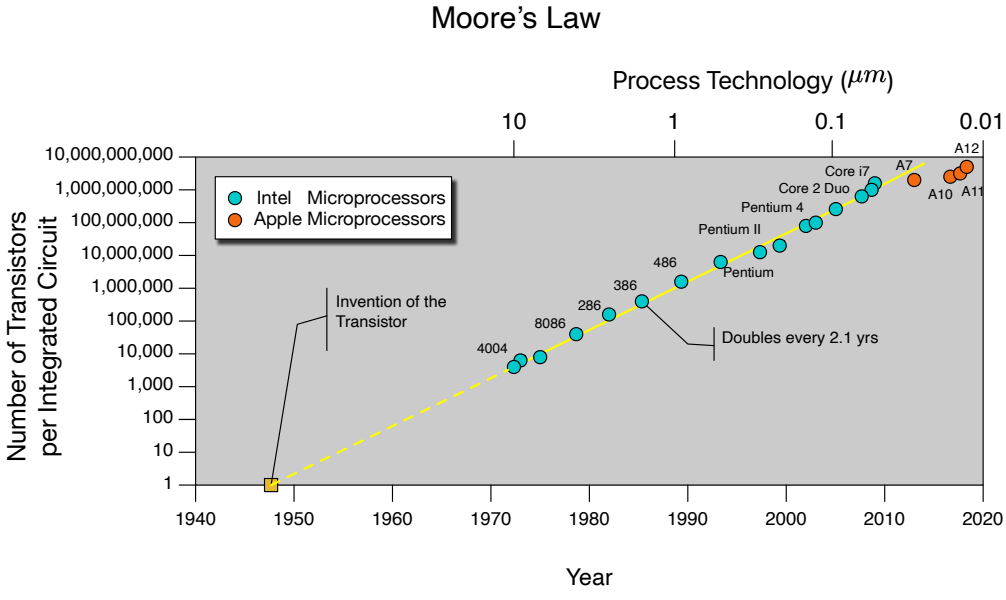
香港中文大學
The Chinese University of Hong Kong

Design for Manufacturability: From Layout To Chip

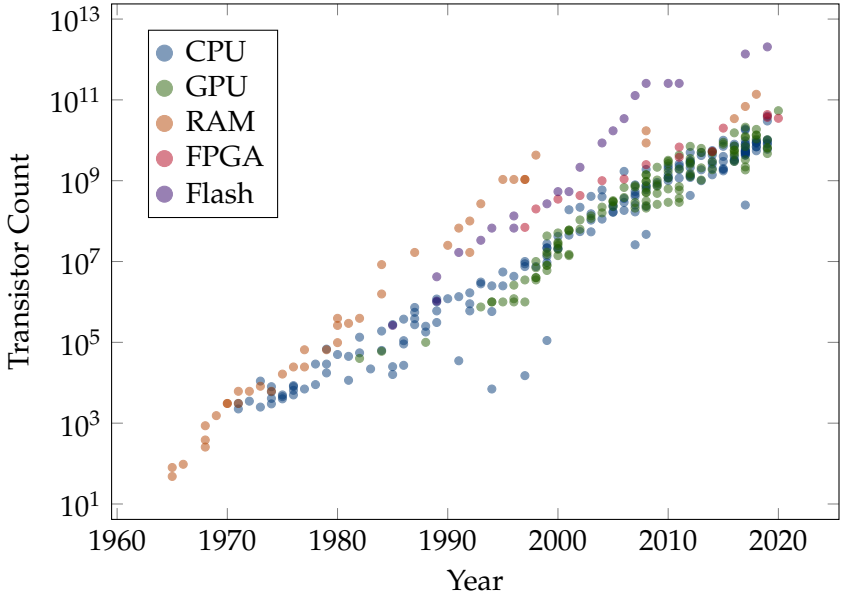
Bei Yu
Department of Computer Science & Engineering
Chinese University of Hong Kong
byu@cse.cuhk.edu.hk

November 19, 2022

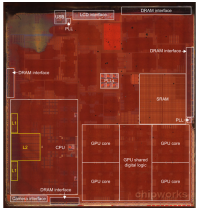
Moore's Law to Extreme Scaling



Moore's Law to Extreme Scaling

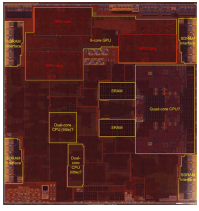


Scaling of Apple SOC



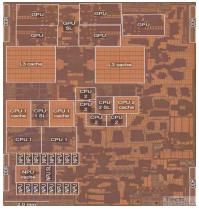
Apple A7 (2013)

- 1,000,000 K Transistors
- $102mm^2$ die size
- 1.3GHz



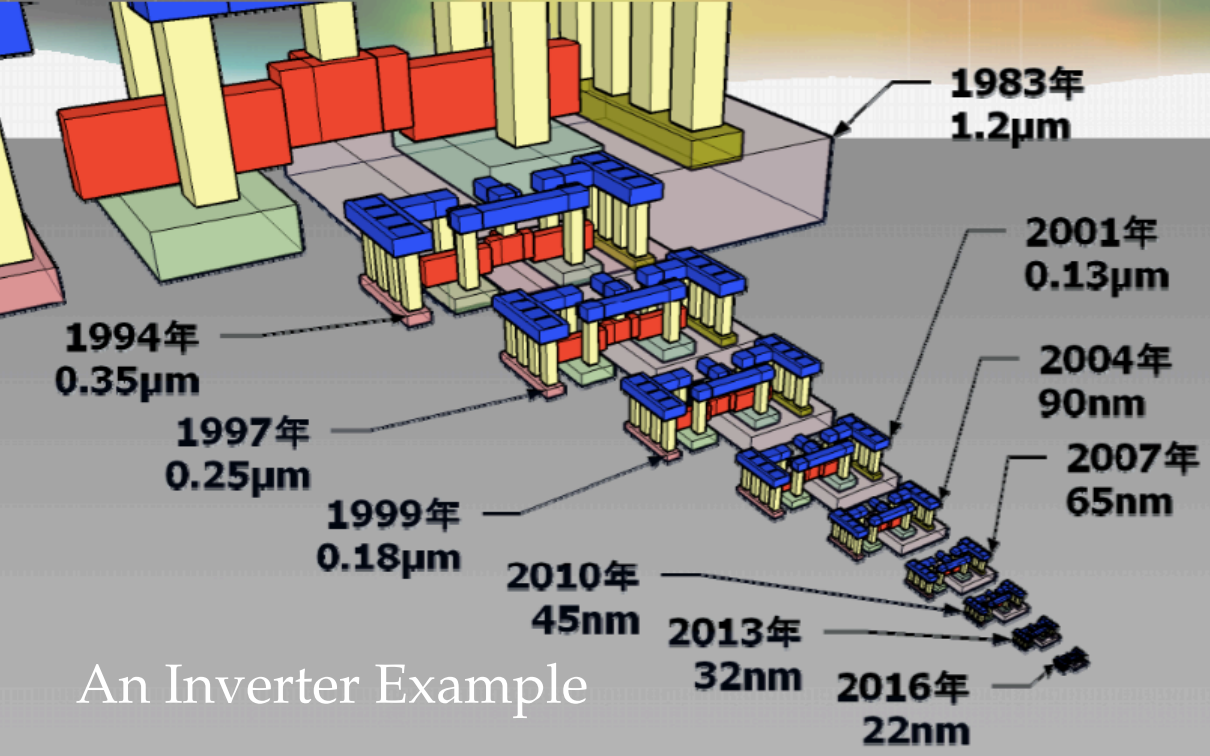
Apple A10 (2016)

- 3,300,000 K Transistors
- $125mm^2$ die size
- 2.34GHz

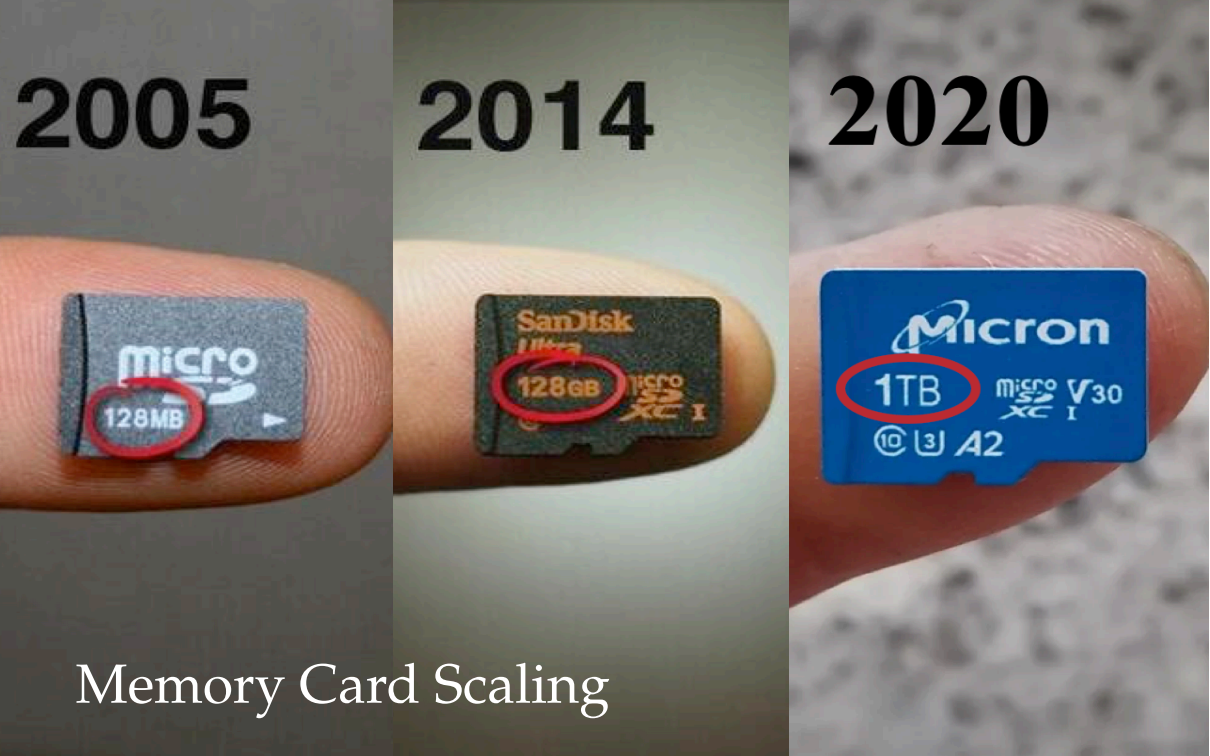


Apple A16 (2022)

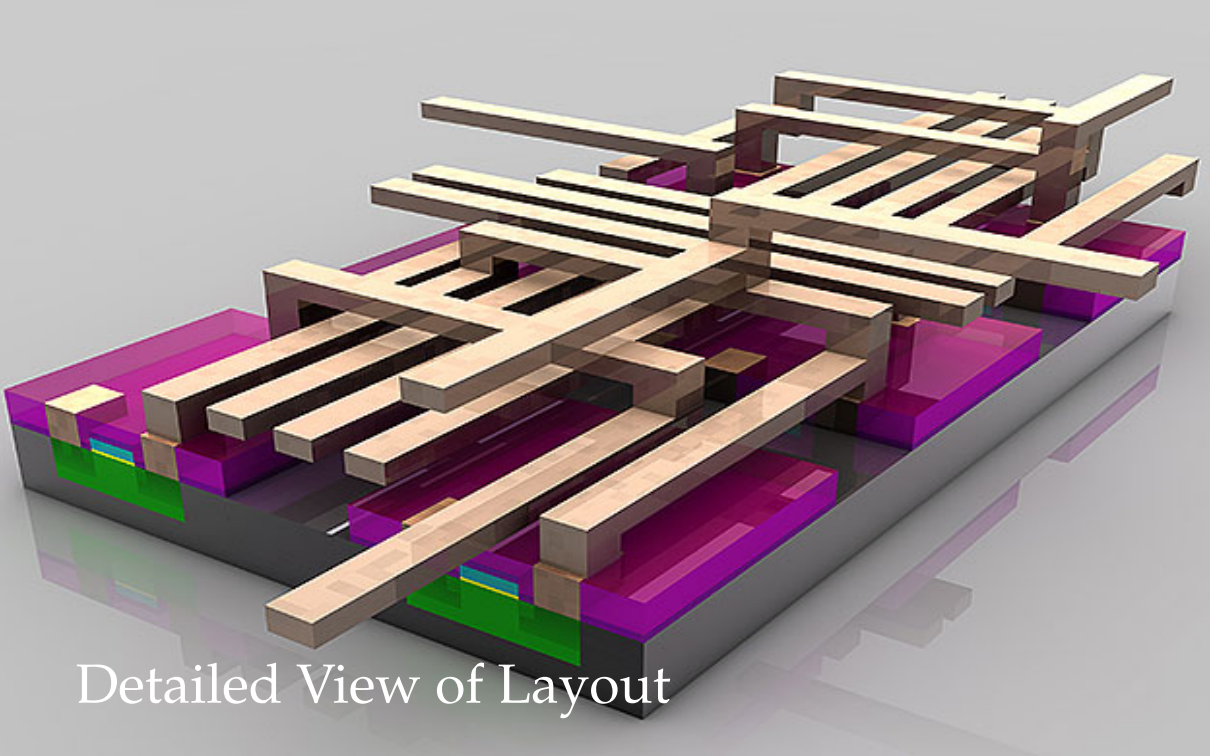
- 16,000,000 K Transistors
- $108mm^2$ die size
- 3.46GHz



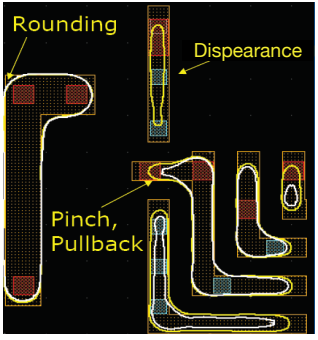
An Inverter Example

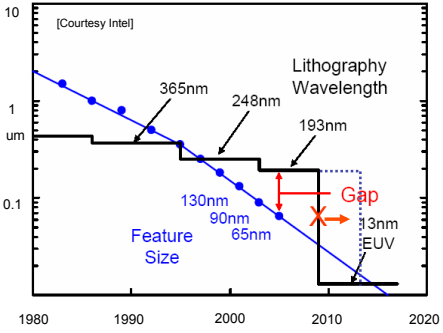


Memory Card Scaling


Detailed View of Layout

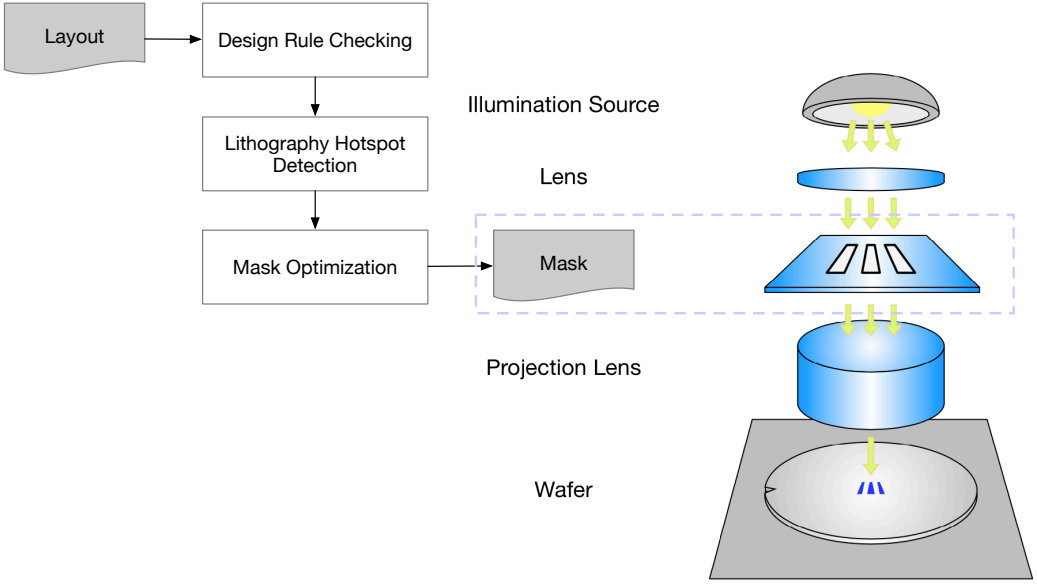
Manufacturing Issues of Layout





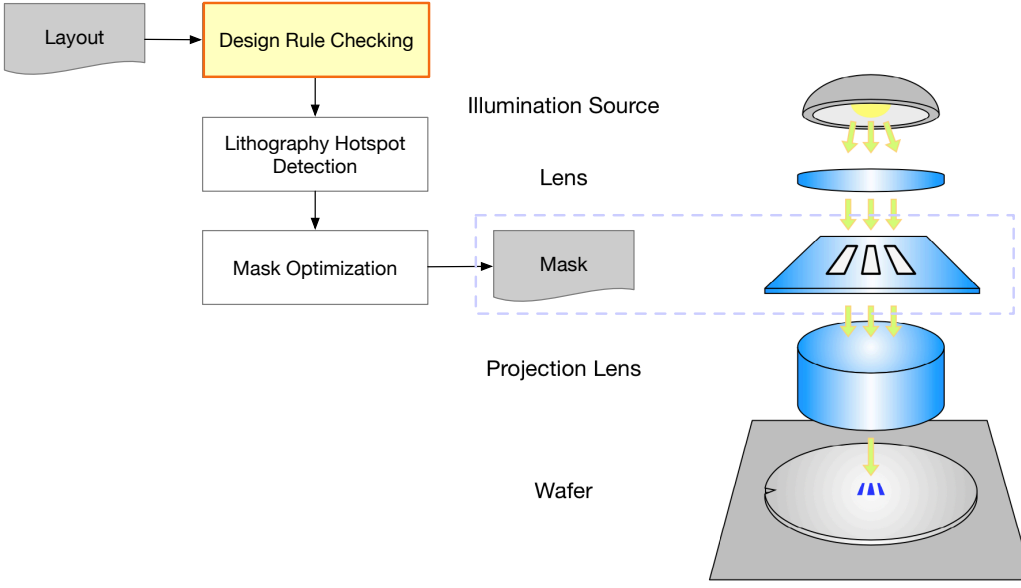
Manufacturability Status & Challenges

From Layout to Chip

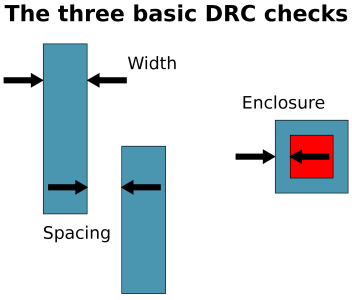


Outline

Design Rule Checking (DRC)

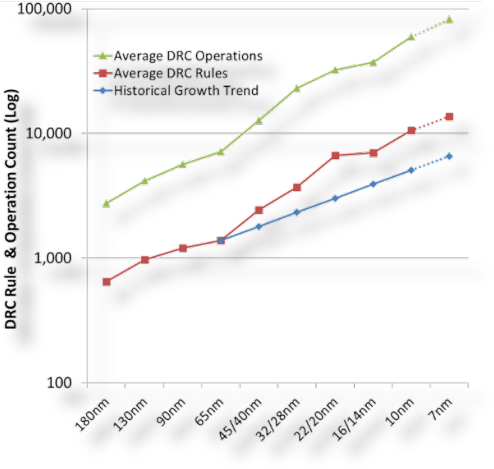

Layout
Design Rule Checking
Lithography Hotspot Detection
Mask Optimization
Mask
Illumination Source
Lens
Projection Lens
Wafer

DRC: Introduction



- Minimum area
- Shorts violation
- End of Line spacing
- ...

- DRC count increases by 20-30% every node.



DRC: Research

- Improving DRC rules: "large" as well as "small".
  - Conducted in foundaries.
- Reduces to DL tasks: classification, detection...
- Improving DRC efficiency: current single full-chip DRC takes hours to days.
  - Conducted in design companies and EDA vendors.

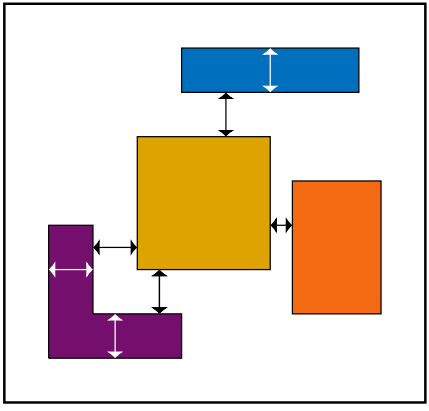
DRC: Research

- Improving DRC rules: "large" as well as "small".
  - Conducted in foundaries.
- Reduces to DL tasks: classification, detection...
- Improving DRC efficiency: current single full-chip DRC takes hours to days.
  - Conducted in design companies and EDA vendors.
  - e.g. GPU acceleration

DRC Problem Example: Distance Check

Problem (Distance Check (informal))

- *Layout: a set of axis-parallel polygonal objects*
- *Distance rule: any two edges must not be closer than a predefined minimal distance*
- *Distance violation: a pair of edges in the layout that violate the distance rule*
- *Our task: report all the distance violations*



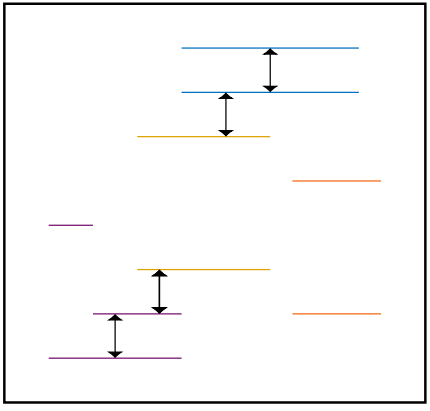
DRC Problem Example: Distance Check

(We only consider horizontal edges.)

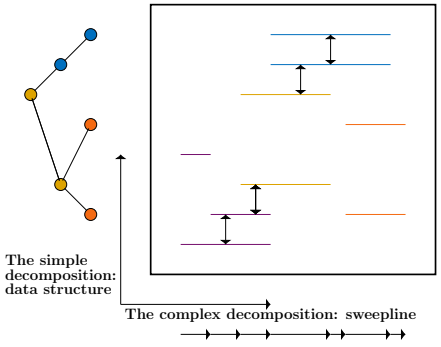
Problem (Distance Check)

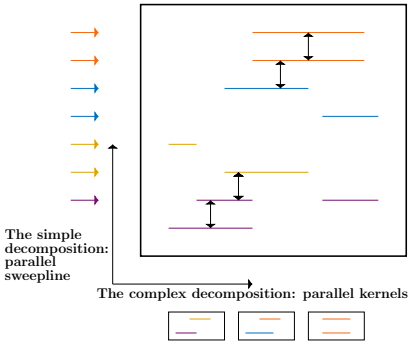
Given a set $\mathcal{H}$ of horizontal segments in $\mathbb{R}^2$, report the segment pairs from $\mathcal{H}^2$ whose horizontal projection is nonempty, and vertical distance is smaller than δ. Formally, we want to report:

$$\{([l_1, r_1] \times y_1, [l_2, r_2] \times y_2) \in \mathcal{H}^2\}$$
$$s.t.\ [l_1, r_1] \cap [l_2, r_2] \neq \emptyset, |y_1 - y_2| < \delta$$



Insight: Sequential vs Parallel





Main Idea:

Decompose a problem by the 'simple' direction for parallelism, and leave the 'complex' work to each individual processor.

Stats and Width Check [ICCAD'22][1]

Design	Layer	#Tiles	#Polygons	#Edges	#Edge/Polygon	Width Check Time (s) KLayout	X-Check	Speedup
gcd	Metal1	1	391	24440	62.5	<0.1	0.1	-
	Metal2	1	1229	4916	4.0	<0.1	<0.1	-
aes	Metal1	16	17739	2059906	116.1	2.9	3.0	0.97×
	Metal2	16	76007	304028	4.0	0.2	0.1	-
bp_be	Metal1	56	34747	27245522	784.1	21.9	19.3	1.13×
	Metal2	56	393834	1575336	4.0	0.4	0.4	-
bp	Metal1	144	107706	52595418	488.3	38.9	33.0	1.18×
	Metal2	144	833588	3334352	4.0	0.9	0.9	-
Average								1.09×

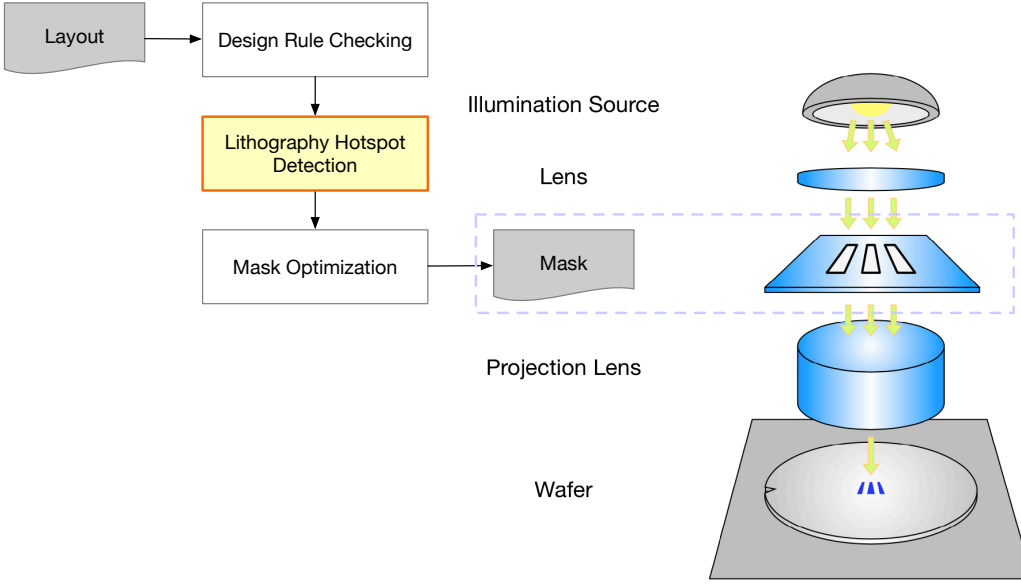
[1] Zhuolun He, Yuzhe Ma, and Bei Yu (2022). "X-Check: GPU-Accelerated Design Rule Checking via Parallel Sweepline Algorithms". In: *Proc. ICCAD*.

Enclosing Check and Space Check [ICCAD'22][2]

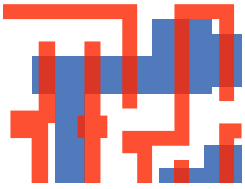
Design	Layer	Enclosing Check			Space Check		
		KLayout	X-Check	Speedup	KLayout	X-Check	Speedup
gcd	Metal1	38.4	2.4	16.00×	12.6	2.4	5.25×
	Metal2	2.5	2.5	1.00×	6.4	2.4	2.67×
aes	Metal1	15470.4	12.3	1257.76×	4493.8	67.5	66.57×
	Metal2	2227.0	14.5	153.59×	2778.5	9.9	280.66×
bp_be	Metal1	66194.6	128.6	514.73×	6718.7	123.7	54.31×
	Metal2	3089.2	147.4	20.96×	4171.5	16.6	251.30×
bp	Metal1	98370.4	235.3	418.06×	14019.7	233.4	60.07×
	Metal2	3958.7	276.6	14.41×	5164.4	65.9	78.37×
Average				61.36×			45.00×

[2] Zhuolun He, Yuzhe Ma, and Bei Yu (2022). "X-Check: GPU-Accelerated Design Rule Checking via Parallel Sweepline Algorithms". In: *Proc. ICCAD*.

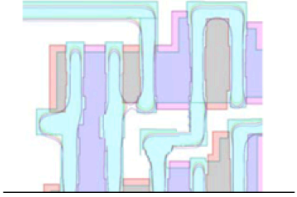
Hotspot Detection


Layout
Design Rule Checking
Lithography Hotspot Detection
Mask Optimization
Mask
Illumination Source
Lens
Projection Lens
Wafer

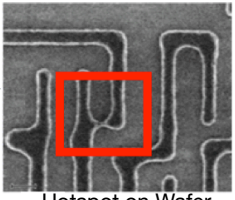
Challenge: Failure (Hotspot) Detection



Pre-OPC Layout

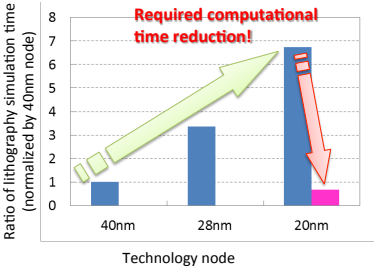


Post-OPC Mask

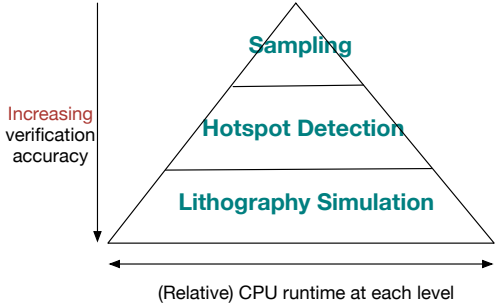


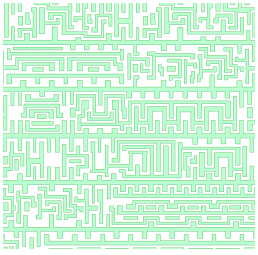
Hotspot on Wafer

- RET: OPC, SRAF, MPL
- Still hotspot: low fidelity patterns
- Simulations: extremely CPU intensive



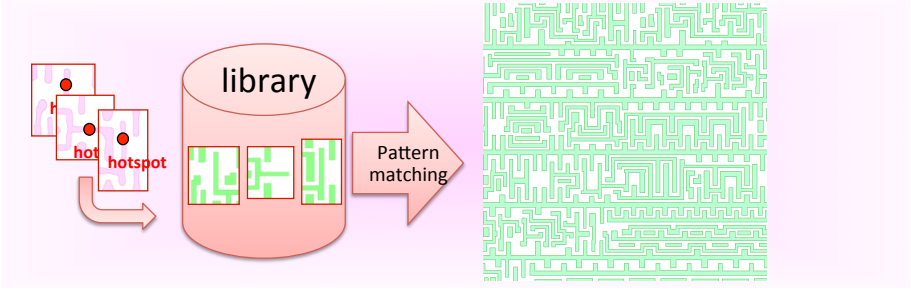
Hotspot Detection Hierarchy



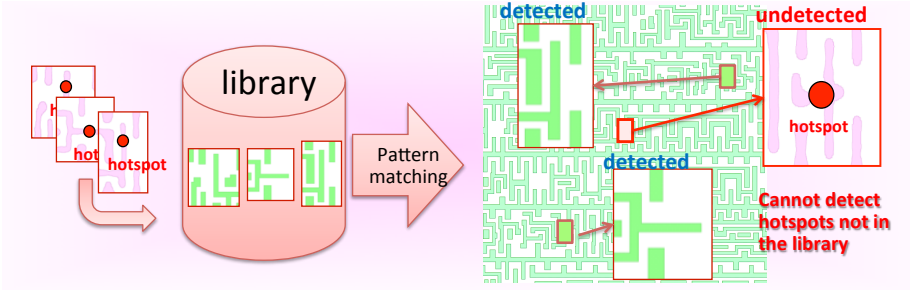


- **Sampling** (DRC Checking):
  scan and rule check each region
- **Hotspot Detection**:
  verify the sampled regions and report potential hotspots
- **Lithography Simulation**:
  final verification on the reported hotspots

Pattern Matching based Hotspot Detection

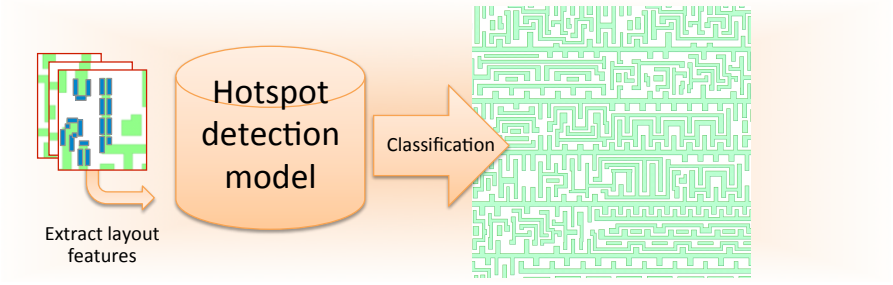


Pattern Matching based Hotspot Detection

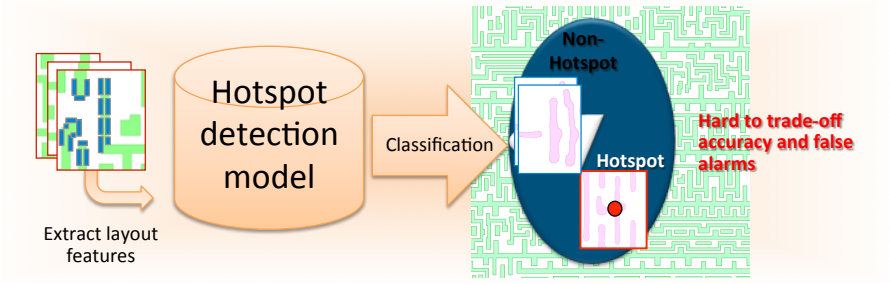


- Fast and accurate
- [Yu+,ICCAD'14] [Nosato+,JM3'14] [Su+,TCAD'15]
- Fuzzy pattern matching [Wen+,TCAD'14]
- Hard to detect non-seen pattern

Classification based Hotspot Detection



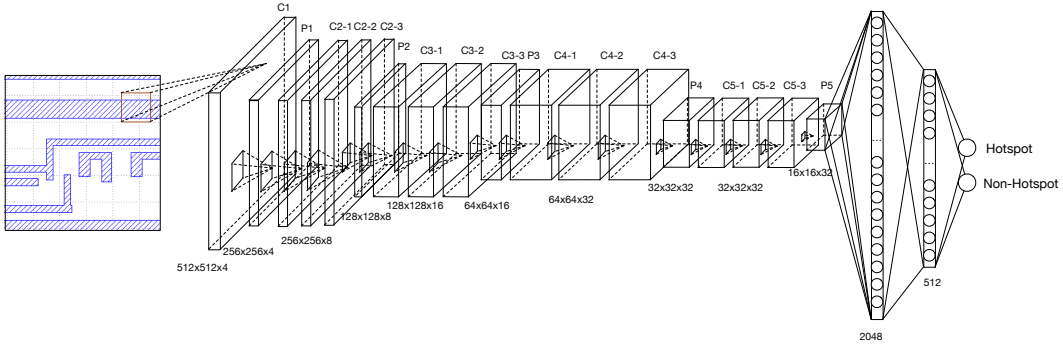
Classification based Hotspot Detection



- Predict new patterns
- Decision-tree, ANN, SVM, Boosting ...
- [Drmanac+,DAC'09] [Ding+,TCAD'12] [Yu+,JM3'15] [Matsunawa+,SPIE'15] [Yu+,TCAD'15]
- Hard to balance accuracy and false-alarm

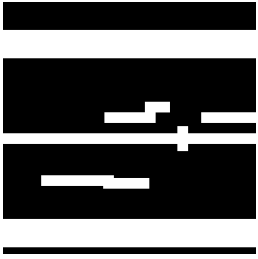
First DNN Hotspot Detection Architecture[3]

- Total 21 layers with 13 convolution layers and 5 pooling layers.
- A ReLU is applied after each convolution layer.

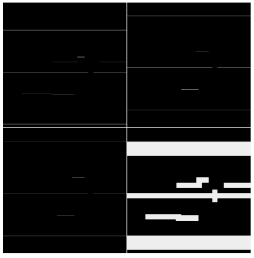


[3]Haoyu Yang, Luyang Luo, et al. (2017). "Imbalance aware lithography hotspot detection: a deep learning approach". In: *JM3* 16.3, p. 033504.

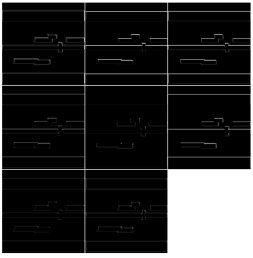
Layer Visualization



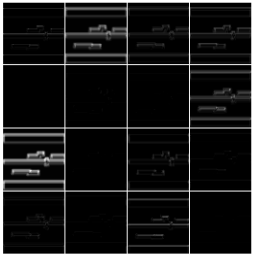
Origin



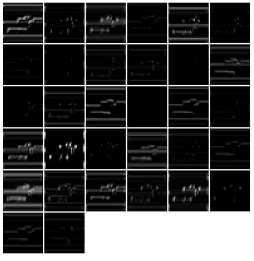
Pool1



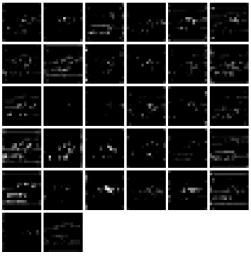
Pool2



Pool3

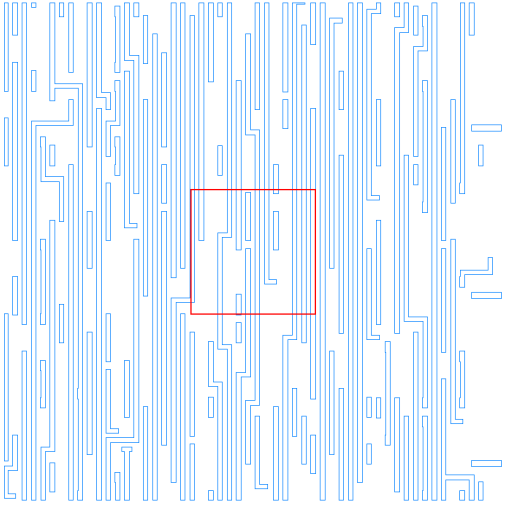


Pool4

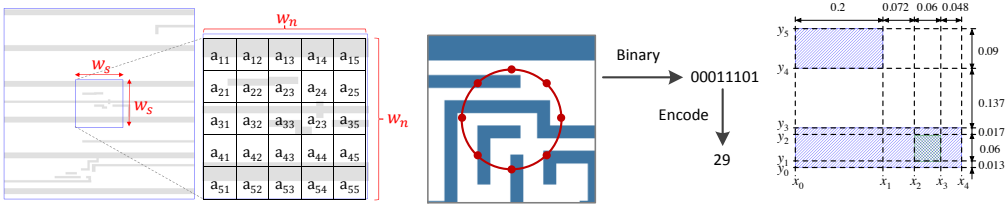


Pool5

Rethinking: “ImageHotspot” v.s. ImageNet



HSD-Research: New Representation



(a) (b) (c)

- (a) Density-based encoding [SPIE'15][4]
- (b) Concentric circle sampling [ICCAD'16][5]
- (c) Squish pattern [ASPDAC'19][6]

[4] Tetsuaki Matsunawa et al. (2015). "A new lithography hotspot detection framework based on AdaBoost classifier and simplified feature extraction". In: *Proc. SPIE*. vol. 9427.

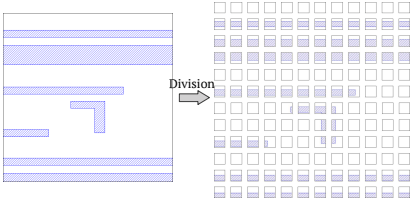
[5] Hang Zhang, Bei Yu, and Evangeline F. Y. Young (2016). "Enabling Online Learning in Lithography Hotspot Detection with Information-Theoretic Feature Optimization". In: *Proc. ICCAD*, 47:1–47:8.

[6] Haoyu Yang, Piyush Pathak, et al. (2019). "Detecting multi-layer layout hotspots with adaptive squish patterns". In: *Proc. ASPDAC*, pp. 299–304.

Simplified CNN Architecture [DAC'17][7]

Feature Tensor Generation:

- Clip Partition
- Discrete Cosine Transform
- Discarding High Frequency Components
- Feature Tensor

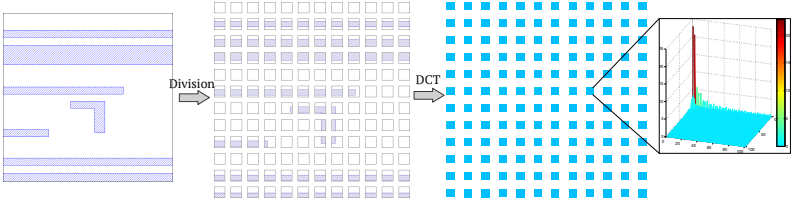


[7] Haoyu Yang, Jing Su, et al. (2017). "Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning". In: *Proc. DAC*, 62:1–62:6.

Simplified CNN Architecture [DAC'17][7]

Feature Tensor Generation:

- Clip Partition
- Discrete Cosine Transform
- Discarding High Frequency Components
- Feature Tensor

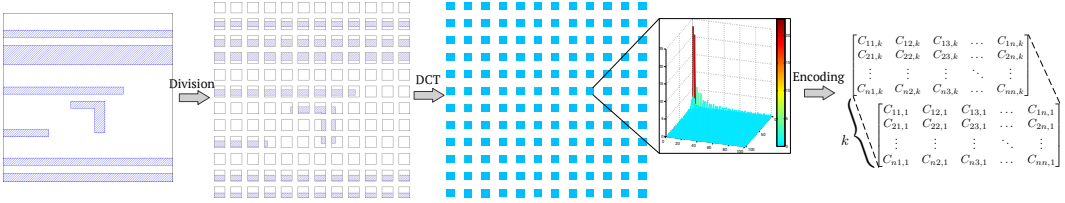


[7] Haoyu Yang, Jing Su, et al. (2017). "Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning". In: *Proc. DAC*, 62:1–62:6.

Simplified CNN Architecture [DAC'17][7]

Feature Tensor Generation:

- Clip Partition
- Discrete Cosine Transform
- Discarding High Frequency Components
- Feature Tensor



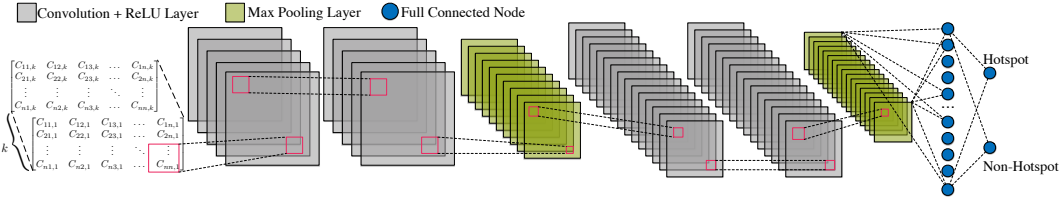
[7] Haoyu Yang, Jing Su, et al. (2017). "Layout Hotspot Detection with Feature Tensor Generation and Deep Biased Learning". In: *Proc. DAC*, 62:1–62:6.

Simplified CNN Architecture [DAC'17]

Feature Tensor

- k-channel hyper-image
- Compatible with CNN
- Storage and computional efficiency

Layer	Kernel Size	Stride	Output Node #
conv1-1	3	1	$12 \times 12 \times 16$
conv1-2	3	1	$12 \times 12 \times 16$
maxpooling1	2	2	$6 \times 6 \times 16$
conv2-1	3	1	$6 \times 6 \times 32$
conv2-2	3	1	$6 \times 6 \times 32$
maxpooling2	2	2	$3 \times 3 \times 32$
fc1	N/A	N/A	250
fc2	N/A	N/A	2



The Biased Learning Algorithm

- Minimize difference with ground truths

$$\boldsymbol{y}_n^* = [1, 0],\ \boldsymbol{y}_h^* = [0, 1].$$

$$\boldsymbol{F} \in \begin{cases} \mathcal{N}, & \text{if } \boldsymbol{y}(0) > 0.5, \\ \mathcal{H}, & \text{if } \boldsymbol{y}(1) > 0.5. \end{cases}$$

- Naive: Shifting decision boundary

$$\mathbf{F} \in \begin{cases} \mathcal{N}, & \text{if } \boldsymbol{y}(0) > 0.5 + \lambda, \\ \mathcal{H}, & \text{if } \boldsymbol{y}(1) > 0.5 - \lambda. \end{cases}$$

The Biased Learning Algorithm

- Minimize difference with ground truths

$$\boldsymbol{y}_n^* = [1, 0],\ \boldsymbol{y}_h^* = [0, 1].$$

$$\boldsymbol{F} \in \begin{cases} \mathcal{N}, & \text{if } \boldsymbol{y}(0) > 0.5, \\ \mathcal{H}, & \text{if } \boldsymbol{y}(1) > 0.5. \end{cases}$$

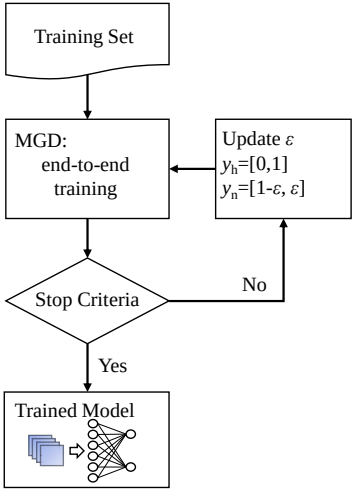
- Naive: Shifting decision boundary (✘)

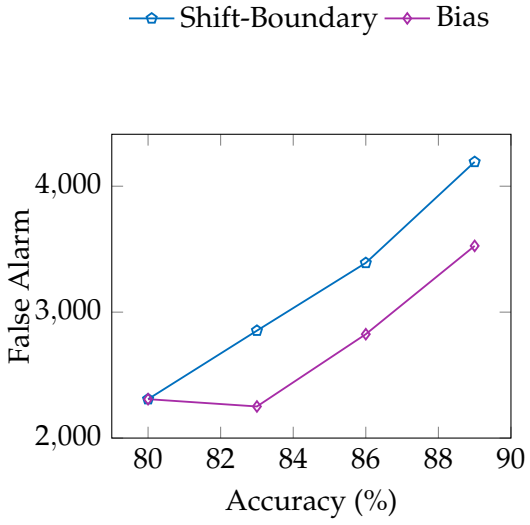
$$\mathbf{F} \in \begin{cases} \mathcal{N}, & \text{if } \boldsymbol{y}(0) > 0.5 + \lambda, \\ \mathcal{H}, & \text{if } \boldsymbol{y}(1) > 0.5 - \lambda. \end{cases}$$

- Biased ground truth:

$$\mathbf{y}_n^* = [1 - \epsilon, \epsilon].$$

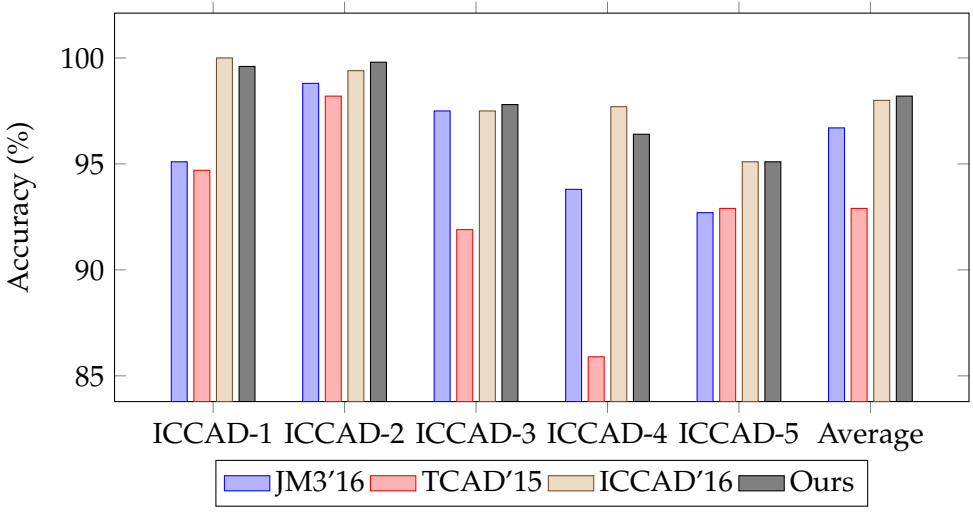
The Biased Learning Algorithm





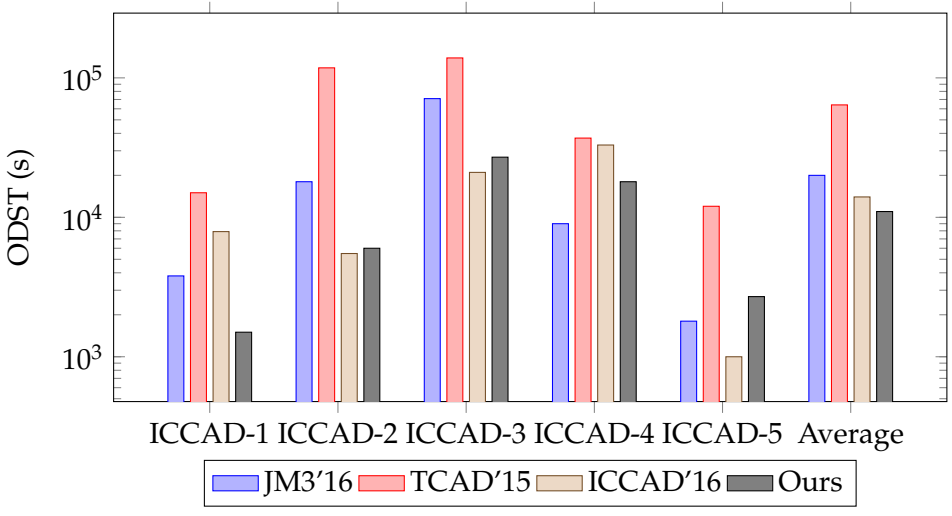
Comparison with Previous Hotspot Detectors

- Detection accuracy improved from 89.6% to 95.5%



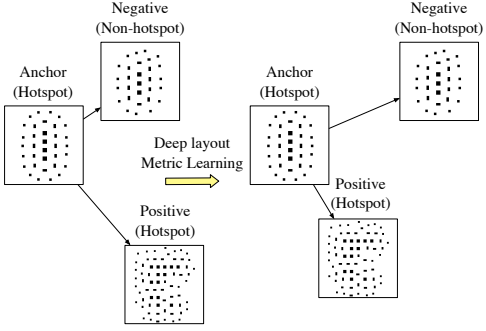
Comparison with Previous Hotspot Detectors

- Comparable false alarm penalty



Motivation

- In original space, the anchor is much similar to the negative
- After deep layout metric learning, in a new manifold, the two hotspot layout clips are kept apart from the non-hotspot clip



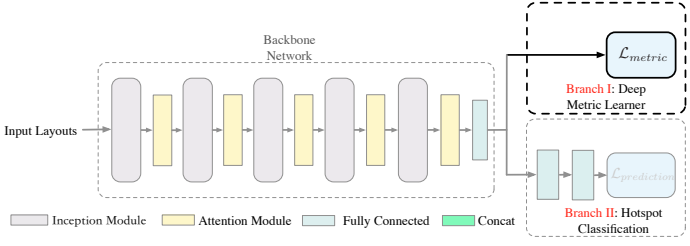
[8]Hao Geng, Haoyu Yang, et al. (2020). "Hotspot Detection via Attention-based Deep Layout Metric Learning". In: *Proc. ICCAD*.

Metric Feature Learning

- A triplet: $f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-)$
- $f_w(\boldsymbol{x}_i)$: an anchor layout clip
- $f_w(\boldsymbol{x}_i{}^+)$: sharing the same label with the anchor
- $f_w(\boldsymbol{x}_i{}^-)$: having the opposite label to the anchor

$$\min_{w} \frac{1}{n} \sum_{i=1}^{N} \max(0, M + ||f_w(\boldsymbol{x}_i) - f_w(\boldsymbol{x}_i{}^+)||_2^2 - ||f_w(\boldsymbol{x}_i) - f_w(\boldsymbol{x}_i{}^-)||_2^2) \tag{1}$$

$$\text{s.t. } ||f_w(\boldsymbol{x}_i)||_2^2 = 1, \ \forall (f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-)) \in \mathcal{T}. \tag{2}$$



Gradients Calculation:

$$\frac{\partial \mathcal{L}_{metric}(f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-))}{\partial f_w(\boldsymbol{x}_i{}^+)} = \frac{2}{n}\left(f_w(\boldsymbol{x}_i{}^+) - f_w(\boldsymbol{x}_i)\right) \cdot \mathbf{1}\left(\mathcal{L}_{metric}(f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-)) > 0\right), \tag{3a}$$

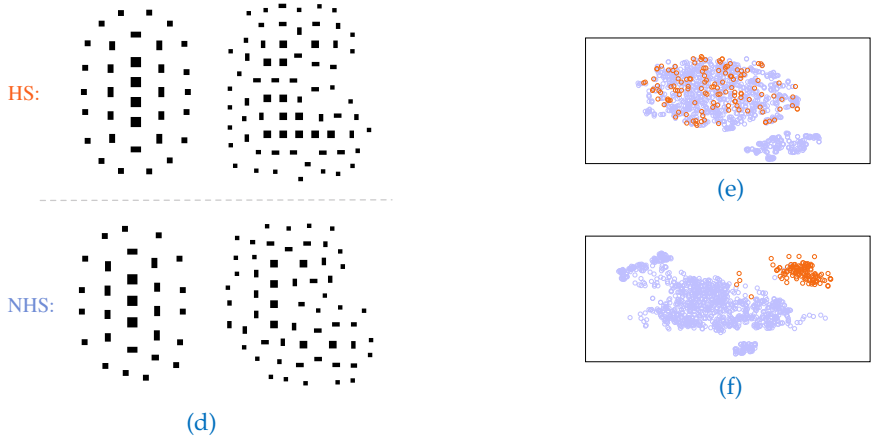
$$\frac{\partial \mathcal{L}_{metric}(f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-))}{\partial f_w(\boldsymbol{x}_i{}^-)} = \frac{2}{n}\left(f_w(\boldsymbol{x}_i) - f_w(\boldsymbol{x}_i{}^-)\right) \cdot \mathbf{1}\left(\mathcal{L}_{metric}(f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-)) > 0\right), \tag{3b}$$

$$\frac{\partial \mathcal{L}_{metric}(f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-))}{\partial f_w(\boldsymbol{x}_i)} = \frac{2}{n}\left(f_w(\boldsymbol{x}_i{}^-) - f_w(\boldsymbol{x}_i{}^+)\right) \cdot \mathbf{1}\left(\mathcal{L}_{metric}(f_w(\boldsymbol{x}_i), f_w(\boldsymbol{x}_i{}^+), f_w(\boldsymbol{x}_i{}^-)) > 0\right), \tag{3c}$$

where $\mathbf{1}$ is the indicator function which is defined as:

$$\mathbf{1}(x) = \begin{cases} 1 & \text{if x is true,} \\ 0 & \text{otherwise.} \end{cases} \tag{3d}$$

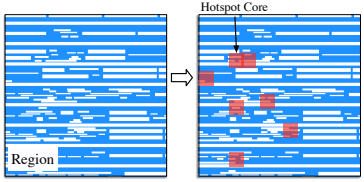
The t-SNE visualizations of feature embeddings on VIA



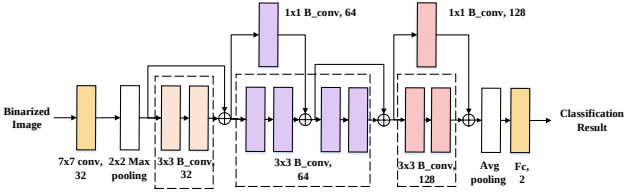
(a) The exemplars of hotspots and non-hotspots; (b) The DCT feature embeddings of TCAD'19; (c) The feature embeddings of our proposed framework.

HSD-Research: New Network Architecture

- Region-based HSD [DAC'19][9]



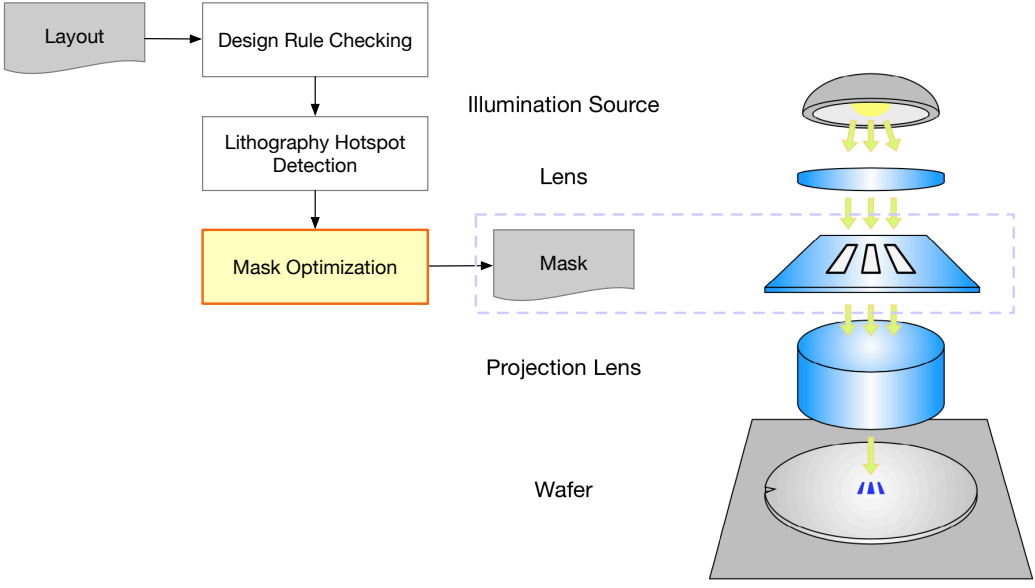
- Binarized residual neural network [DAC'19][10]



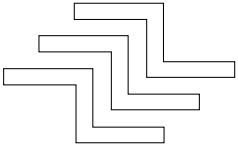
[9]Ran Chen et al. (2019). "Faster Region-based Hotspot Detection". In: *Proc. DAC*, 146:1–146:6.

[10]Yiyang Jiang et al. (2019). "Efficient Layout Hotspot Detection via Binarized Residual Neural Network". In: *Proc. DAC*, 147:1–147:6.

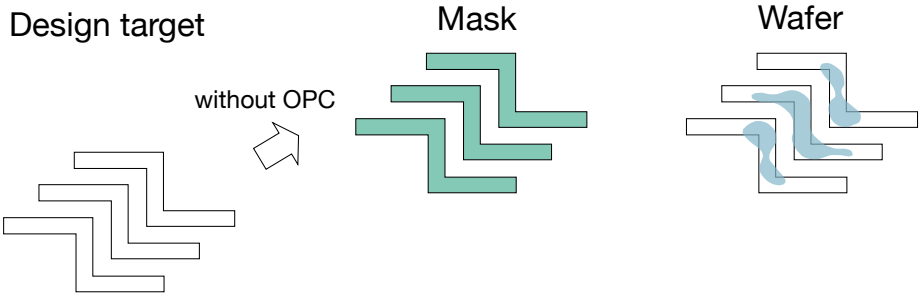
Mask Optimization


Layout
Design Rule Checking
Lithography Hotspot Detection
Mask Optimization
Mask
Illumination Source
Lens
Projection Lens
Wafer

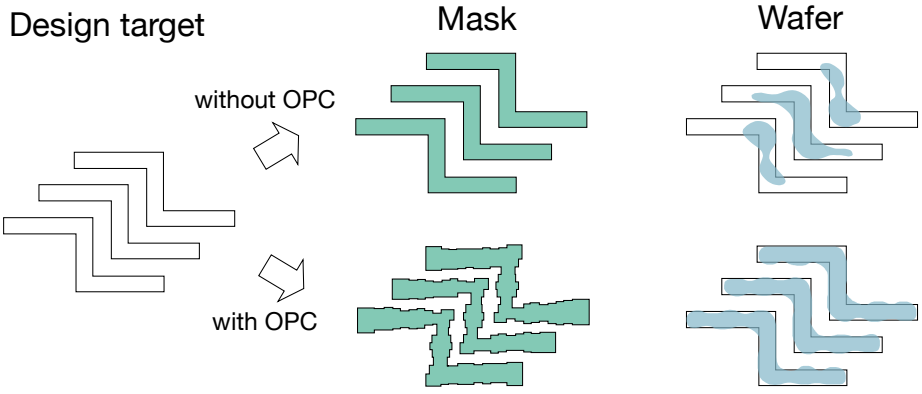
Design target



Mask Optimization



Mask Optimization



Market Anually: USD 100M!

1. `Calibre` by Mentor Graphics
2. `Brion Tool` by ASML
3. `IC Validator` by Synopsys
4. `Pegasus` by Cadence

- SVD Approximation of Partial Coherent System [Cobb,1998]

$$\mathbf{I} = \sum_{k=1}^{N^2} w_k |\mathbf{M} \otimes \mathbf{h}_k|^2. \tag{4}$$

- Reduced Model [Gao+,DAC'14]

$$\mathbf{I} = \sum_{k=1}^{N_h} w_k |\mathbf{M} \otimes \mathbf{h}_k|^2. \tag{5}$$

- Etch Model

$$\mathbf{Z}(x,y) = \begin{cases} 1, & \text{if } \mathbf{I}(x,y) \geq I_{th}, \\ 0, & \text{if } \mathbf{I}(x,y) < I_{th}. \end{cases} \tag{6}$$

Inverse Lithography Technique (ILT)

The main objective in ILT is minimizing the lithography error through gradient descent.

$$E = ||\mathbf{Z}_t - \mathbf{Z}||_2^2, \tag{7}$$

where $\mathbf{Z}_t$ is the target and $\mathbf{Z}$ is the wafer image of a given mask.
Apply translated sigmoid functions to make the pixel values close to either 0 or 1.

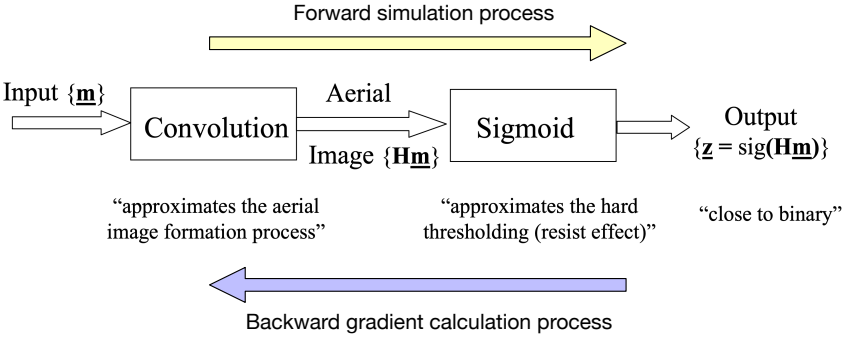
$$\mathbf{Z} = \frac{1}{1 + \exp[-\alpha \times (\mathbf{I} - \mathbf{I}_{th})]}, \tag{8}$$

$$\mathbf{M}_b = \frac{1}{1 + \exp(-\beta \times \mathbf{M})}. \tag{9}$$

Combine Equations (4)–(9) and the analysis in [Poonawala,TIP'07],

$$\begin{aligned}\frac{\partial E}{\partial \mathbf{M}} =& 2\alpha\beta \times \mathbf{M}_b \odot (1 - \mathbf{M}_b) \odot \\ & (((\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_b \otimes \mathbf{H}^*)) \otimes \mathbf{H} + \\ & ((\mathbf{Z} - \mathbf{Z}_t) \odot \mathbf{Z} \odot (1 - \mathbf{Z}) \odot (\mathbf{M}_b \otimes \mathbf{H})) \otimes \mathbf{H}^*).\end{aligned} \tag{10}$$

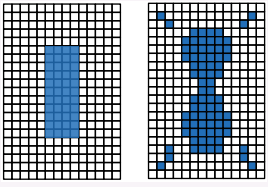
ILT flow


Forward simulation process
Input {m}
Convolution
Aerial
Image {Hm}
Sigmoid
Output
{z = sig(Hm)}
"approximates the aerial image formation process"
"approximates the hard thresholding (resist effect)"
"close to binary"
Backward gradient calculation process

ILT Methodologies [DAC'14][1] [ICCAD'17][2] [DATE'21][3]

Typical ILT

- Mask $\rightarrow$ Image $\rightarrow$ Matrix
- Calculate gradient on each pixel.



Level-set method

- Boundary-based update
- Implicit representation; focus on boundaries

$$\begin{cases} \phi(t,\boldsymbol{x}) < 0 & \text{if } \boldsymbol{x} \in \Omega(t) \\ \phi(t,\boldsymbol{x}) = 0 & \text{if } \boldsymbol{x} \in \Gamma(t) \\ \phi(t,\boldsymbol{x}) > 0 & \text{if } \boldsymbol{x} \in \overline{\Omega(t)} \end{cases}$$

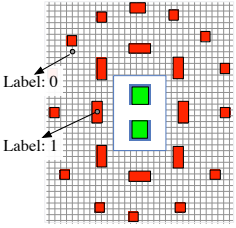
[1] Jhih-Rong Gao et al. (2014). "MOSAIC: Mask Optimizing Solution With Process Window Aware Inverse Correction". In: *Proc. DAC*. San Jose, California, 52:1–52:6.

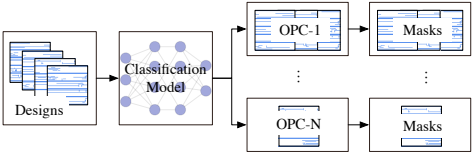
[2] Yuzhe Ma et al. (2017). "A Unified Framework for Simultaneous Layout Decomposition and Mask Optimization". In: *Proc. ICCAD*, pp. 81–88.

[3] Ziyang Yu et al. (2021). "A GPU-enabled Level Set Method for Mask Optimization". In: *Proc. DATE*.

Discriminative models [TCAD'20][4] [ASPDAC'20][5]

- Pixel-wise classification
- Printed image estimation/quality estimation



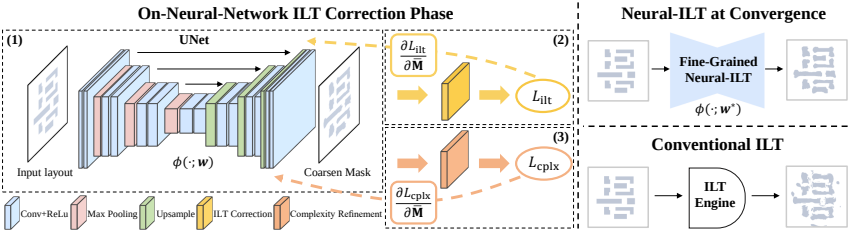


[4] Hao Geng, Wei Zhong, et al. (2020). "SRAF Insertion via Supervised Dictionary Learning". In: *IEEE TCAD*.

[5] Haoyu Yang, Wei Zhong, et al. (2020). "VLSI Mask Optimization: From Shallow To Deep Learning". In: *Proc. ASPDAC*, pp. 434–439.

Generative model [DAC'18][6] [ICCAD'20][7] [ICCAD'20][8]

- Image generation

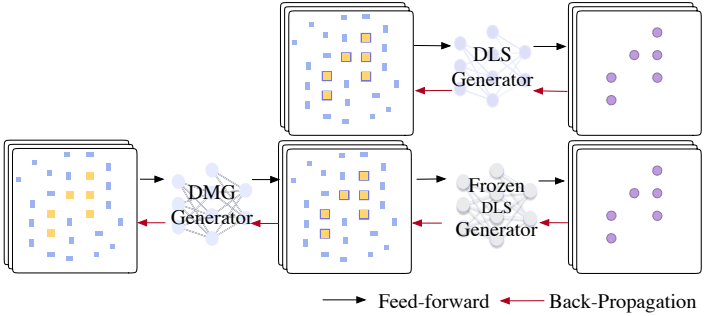


[6]Haoyu Yang, Shuhe Li, et al. (2018). "GAN-OPC: Mask Optimization with Lithography-guided Generative Adversarial Nets". In: *Proc. DAC*, 131:1–131:6.

[7]Bentian Jiang et al. (2020). "Neural-ILT: Migrating ILT to Nerual Networks for Mask Printability and Complexity Co-optimizaton"". In: *Proc. ICCAD*.

[8]Guojin Chen et al. (2020). "DAMO: Deep Agile Mask Optimization for Full Chip Scale". In: *Proc. ICCAD*.

Deep Mask Optimization [ICCAD'20][8]

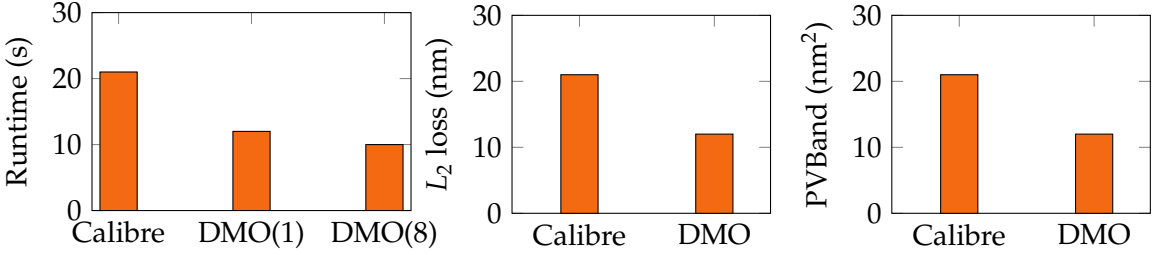


	GAN-OPC			Calibre			**DMO**		
	L_2 (nm)	PV Band (nm^2)	Runtime (s)	L_2 (nm)	PV Band (nm^2)	Runtime (s)	L_2 (nm)	PV Band (nm^2)	Runtime (s)
case 1	7456	11424	284	5159	11671	1417	**4631**	**11166**	352
case 2	7321	11215	281	4987	11463	1406	**4432**	**10955**	336
case 3	7102	11265	285	5420	11516	1435	**4802**	**11032**	367
case 4	8032	11642	322	5382	11910	1606	**4835**	**11265**	399
Average	7478	11386	293	5237	11640	1466	**4675**	**11104**	363
Ratio	1.60	1.03	0.80	1.12	1.05	4.04	**1.00**	**1.00**	1.00

[8] Guojin Chen et al. (2020). "DAMO: Deep Agile Mask Optimization for Full Chip Scale". In: *Proc. ICCAD*.

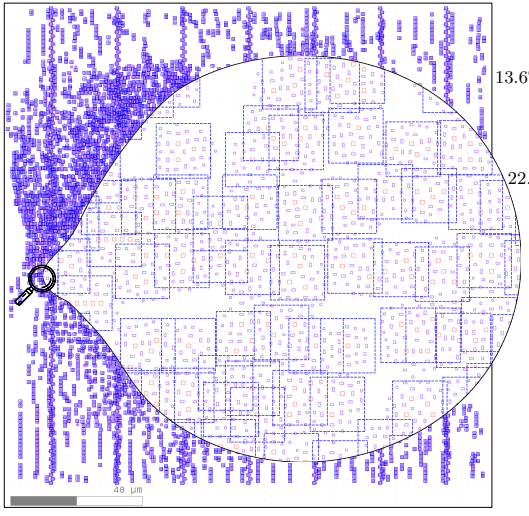
Scalability on Full-chip Layout

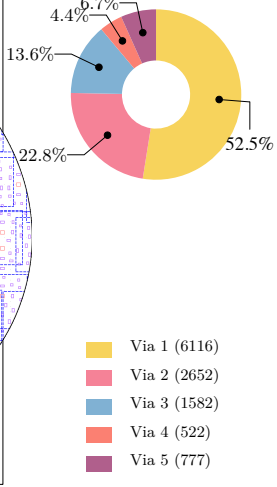
- Tested on $10\mu\text{m} \times 10\mu\text{m}$ layout.



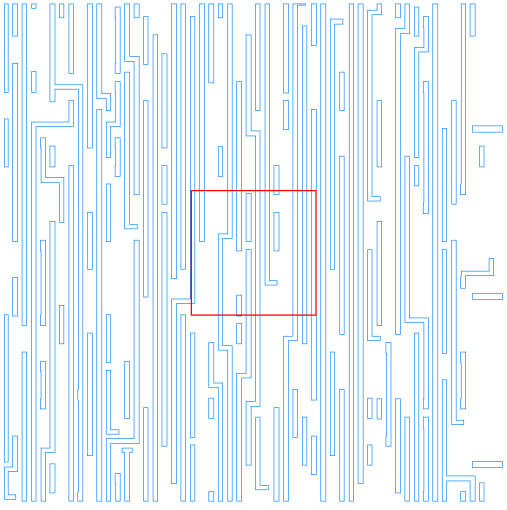
- DMO(1) – Single GPU card;
- DMO(8) – 8 GPU cards

Results on ISPD 2019 datasets





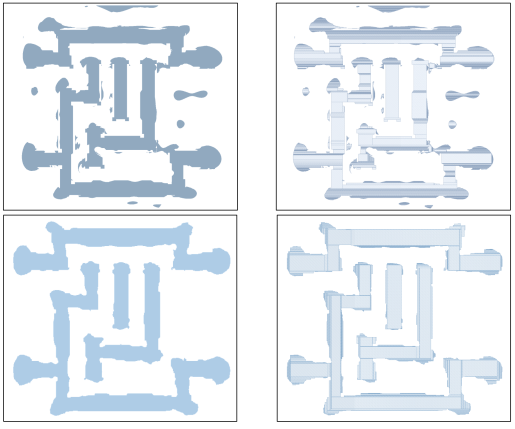
Challenges



Main issues in full chip layout

- Scalability
- Stitch error

Challenges



Main issues in mask manufacturing

- Non-desired "noisy" patterns
- High mask writing runtime

THANK YOU!