AI-Enabled Agile IC Design and Manufacturing

David Z. Pan
The University of Texas at Austin
http://www.ece.utexas.edu/~dpan
IC Design/Manufacturing Complexity

Divide a chip into small partitions e.g., 1~2M cells per partition

Turn-around time for 1 iteration of backend flow: 3~6 days for one partition

What you see (at design) is not (necessarily) what you get (at fab)

Design target

without OPC

with OPC

Mask

Wafer
IC Design/Manufacturing Flow

System Specification
Architectural Design
Functional/Logic Design
Logic Synthesis
Physical Design
Mask Synthesis & Verification
Fabrication
Packaging and Testing

Floorplanning
Placement
Clock Tree Synthesis
Signal Routing
Design Closure
DFM Closure

Process Modeling
Test Patterns
Lithography Experiment
Lithography Modeling
Layout/Mask
Mask Optimization Recipe
SRAF & OPC
Lithography Rule Check

Mask Optimization
Nanometer Design/Manufacturing Challenges

- Performance/Power/Area PPA
- Manufacturability/Yield
- Reliability
- Security
- Turn-around time

......

2018 DARPA ERI ($1.5B) IDEA/POSH “Hardware Compiler” ($100M)

A. Olofsson, DARPA, ISPD-2018 Keynote

DARPA Grand Challenge: No-human-in-the-loop, 24-hour turn around time!
How AI (ML/DL) Can Help?

- Lots of work for various stages of physical design and DFM
- For example on lithography hotspot detection
  - Our work [Ding+, ICICDT 2009 BPA] among the first to use ML (SVM) for litho-hotspot detection
  - Very active research in last 10 years, ICCAD 2012 CAD Contest
  - Meta-classification combining ML and PM [Ding+, ASPDAC’12 BPA]
  - Deep neural network [Yang+, DAC’17]
  - Big data vs. small data; transfer/active/semi-supervised learning [Lin+, ISPD’18], [Chen+, ASPDAC’19], Litho-GPA [Ye+, DATE 2019]…

- ML and PD Tutorial in July 2019 (ACM/IEEE Seasonal School)
- My talk today will cover some recent ideas/results for discussion
DREAMPlace: Deep Learning Toolkit-Enabled GPU Acceleration for Modern VLSI Placement [Lin+, DAC’19, Best Paper Award]

Source code release: https://github.com/limbo018/DREAMPlace
Typical Nonlinear Placement Algorithm

\[ \min_{x,y} \sum_{e \in E} WL(e; x, y), \]
\[ s.t. \quad D(x, y) \leq t_d \]

Objective of nonlinear placement

\[ \min (\sum_{e \in E} WL(e; x, y)) + \lambda D(x, y) \]

- Many papers on how to model WL, density, parameter tuning, etc.
- Huge development effort on a high-quality placement engine (e.g., > 1 year for RePIAce)
- CPU>3h to get good quality placement of 10M-cell design
- Clustering/acceleration limited \(\Rightarrow\) quality degradation
What is your Dream Placement Engine?

✓ **Best quality**: wirelength ➔ congestion, timing, power, …
✓ **Ultrafast**: placement is at the center of entire design flow ➔ faster design turn-around-time
✓ **Low development overhead**: ➔ from 1 year to a month or two?
✓ **Extensible**: easy to try new algorithms and acceleration techniques

10M-cell design finishes within 5min
Over 60x speedup in neural network training since 2013
DREAMPlace Strategies

♦ We propose a novel **analogy** by casting the nonlinear placement optimization into a neural network training problem.

♦ Greatly leverage deep learning hardware (GPU) and software toolkit (e.g., PyTorch).

♦ Enable ultra-high parallelism and acceleration while getting state-of-the-art results.
Analogy Between NN Training and Placement

\[
\min_{\mathbf{w}} \sum_{i} f(\phi(x_i; \mathbf{w}), y_i) + \lambda R(\mathbf{w})
\]

**Forward Propagation** (Compute obj)

Data Instance \((x_i, y_i)\) \(\Rightarrow\) Neural Network \(\phi(\cdot; \mathbf{w})\) \(\Rightarrow\) Error Function \(f(\phi(x_i; \mathbf{w}), y_i)\)

**Backward Propagation** (Compute Gradient \(\frac{\partial \text{obj}}{\partial \mathbf{w}}\))

Train a neural network

\[
\min_{\mathbf{w}} \sum_{i} \text{WL}(e_i; \mathbf{w}) + \lambda D(\mathbf{w})
\]

**Forward Propagation** (Compute obj)

Net Instance \((e_i, 0)\) \(\Rightarrow\) Neural Network \(\text{WL}(\cdot; \mathbf{w})\) \(\Rightarrow\) Error Function \(\text{WL}(e_i; \mathbf{w})\)

**Backward Propagation** (Compute Gradient \(\frac{\partial \text{obj}}{\partial \mathbf{w}}\))

Solve a placement
Leverage mature/highly optimized deep learning toolkit
Global Placement Result Comparison

**RePlAce** [Cheng+, TCAD’18]
- CPU: 24-core 3GHz Intel Xeon
- 64GB memory allocated
- Current state-of-the-art DREAMPlace

**DREAMPlace** [Lin+, DAC’19]
- CPU: Intel E5-2698 v4 @2.20GHz
- GPU: 1 NVIDIA Tesla V100
- Single CPU thread was used

34x speedup by DREAMPlace
43x speedup by DREAMPlace

Same placement quality of results!

10M-cell design finishes within 5min, instead of 3+ hrs
Leverage AI hardware and software development recently

Decouple core algorithm innovations with implementation

- Algorithm innovation written in high-level language, e.g. Python
- Highly extensible: new solver options, new design objectives, ...
- Implementations just focus on certain low-level kernel OPs as needed

Development effort: 1 year ➔ 2 months

The paradigm can be extended to other DA areas

- Significantly enhance IC design productivity and quality
MAGICAL: Machine Generated Analog IC Layout

Open source MAGICAL 0.2
https://github.com/magical-eda/MAGICAL
Analog IC Layout

- DREAMPlace focus on digital IC
- Analog IC to interface with outside world
- Analog IC layout design still mostly manual
  - Very tedious and error-prone
  - Prior DA not successful as that in digital IC

- Our mission is to develop a full-automated analog layout system, leveraging recent AI advancement
- Project started in 08/2018
- [ISPD’19; DAC’19; ICCAD’19; ASPDAC’20]
**MAGICAL Layout System Framework**

- **Input**: unannotated netlist
- **Output**: GDSII Layout
- **Key Components:**
  - Constraint Extraction
  - Device Generation
  - Placement
  - Routing

- **Fully-automated (no-human-in-the-loop)**
- **Guided by analytical, heuristic, and machine learning algorithms (not everything is machine learning or deep learning!)**
MAGICAL Preliminary Results

- A comparator design in 40nm TSMC

Post extraction simulation results

<table>
<thead>
<tr>
<th></th>
<th>Manual</th>
<th>MAGICAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power (uW)</td>
<td>16.8</td>
<td>18.7</td>
</tr>
<tr>
<td>Output Delay (ps)</td>
<td>150</td>
<td>152</td>
</tr>
<tr>
<td>Input-referred Noise (uVrms)</td>
<td>380</td>
<td>334</td>
</tr>
<tr>
<td>Input-referred Offset (mV)</td>
<td>0.15</td>
<td>0.50</td>
</tr>
</tbody>
</table>
MAGICAL Preliminary Results

A 2-stage miller-compensated OTA design in 40nm TSMC

Post extraction simulation results

<table>
<thead>
<tr>
<th></th>
<th>Manual</th>
<th>MAGICAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC Gain (dB)</td>
<td>37.7</td>
<td>38.0</td>
</tr>
<tr>
<td>Unity-gain Bandwidth (MHz)</td>
<td>110</td>
<td>107.5</td>
</tr>
<tr>
<td>Phase Margin (degree)</td>
<td>67.8</td>
<td>62.3</td>
</tr>
<tr>
<td>Input-referred Noise (uVrms)</td>
<td>219</td>
<td>221.5</td>
</tr>
<tr>
<td>CMRR (dB)</td>
<td>103</td>
<td>92.5</td>
</tr>
<tr>
<td>Input-referred Offset (mV)</td>
<td>0.2</td>
<td>0.48</td>
</tr>
</tbody>
</table>
LithoGAN: End-to-End Lithography Modeling with Generative Adversarial Networks [Ye+, DAC’19 BPA Candidate]

Harder question cf. lithography hotspot detection: Without going through litho-simulations, can we directly get printed images?
GAN and CGAN

- Generative Adversarial Network (GAN) [Goodfellow et al, 2014]
  - Two neural networks contest (Generator and Discriminator)
  - Produce images similar to those in the training data set
- Conditional GAN (CGAN) can take a picture in one domain and translate it to another one [Isola et al, CVPR’17]

"Generative Adversarial Networks is the most interesting idea in the last ten years in machine learning."
Yann LeCun, Director, Facebook AI
Image Translation for Litho Modeling

[Ye+, DAC’19]

- Different elements encoded on different image channels
- Resist pattern zoomed in for high-resolution/accuracy

Expensive Litho Simulation

This is now a (modified) **image translation** task

Encode into RGB channels

1µm

1µm

256

256

128nm

128nm

256

256

256

256

[Ye+, DAC’19]
CGAN for Lithography Modeling

CGAN architecture

- Generator $G$ generates a fake resist pattern $G(x)$ with input mask pattern $x$
- Discriminator $D$ needs to classify the image pair $(x, G(x))$ as fake and to predict the image pair $(x, y)$ as real

Sample pairs for training CGAN

[Ye+, DAC’19]
Overall LithoGAN Algorithm

- Dual learning framework
  - CGAN to predict the shape of the resist pattern
  - CNN to predict the resist shape center

[Ye+, DAC’19]
LithoGAN Results

[Ye+, DAC’19]

LithoGAN is $1800x$ faster than rigorous simulations, with acceptable error (in consultation with industry).
Sub-Resolution Assist Feature Generation using Conditional Generative Adversarial Networks

- Directly generate sub-resolution assist feature (SRAF)
- 144x faster than model based approach with similar QoR

GAN-SRAF [Alawieh+, DAC’19]
To bridge design and manufacturing, we propose Lithography Aware Physical Design (LAPD) ➔
  › Litho Hotspot Detection
  › Litho Hotspot Correction

My group has made many key contributions in LAPD 🡞

LithoGAN opens new directions with tremendous potential
Design/Manufacturing for Hardware Security

Global IC supply chain of design, manufacture, test, package...

Design/Manufacturing for Hardware Security

- Arm race between attacking and protection
- Hardware IP reverse engineering using learning techniques
- Intelligent IC camouflaging [Li+, ICCAD’16, TCAD’17, HOST’17 BPA]
- Former PhD Meng Li won ACM SRC Grand Finals First Place in 2018

Fabrication Level

Cell Level

Netlist Level

Camouflaging Cells

Possible dummy via

Layout Modification

Real

Dummy

N+ N+

P-type Substrate
To Recap

No Human in the Loop

MAGICAL

1800x faster

140x faster
Conclusion

- Some recent results in AI-enabled agile IC design & DFM
  - DREAMPlace
  - MAGICAL: GeniusRoute…
  - LithoGAN

- Tremendous potentials to leverage both AI hardware and software advancements

- BIG data, small data, or no (training) data at all (by recasting problems e.g., placement into DREAMPlace)

- Synergistic AI-IC co-design
Acknowledgment

- Funding support / collaborations from NSF, DARPA, Intel, Nvidia, Toshiba Memory, Xilinx, Synopsys
- Many students/post-docs who do the real work
- Many collaborators in academia and industry
  - Dr. Ren and Dr. Khailany from NVIDIA for DREAMPlace
  - Prof. Nan Sun at UT for MAGICAL
  - Dr. Nojima et al. from Toshiba Memory on DFM
Thanks!

Q & A?