USING MACHINE LEARNING FOR VLSI TESTABILITY AND RELIABILITY

Mark Ren, Miloni Mehta
TAKE-HOME MESSAGES

• Machine learning can improve approximate solutions for hard problems.

• Machine learning can accurately predict and replace brute force methods for computational expensive problems.
VLSI TESTABILITY AND RELIABILITY

Design → Manufacturing → Wafer → Chip

Reliability → Testability

Years → Testing → Pass/Fail
PART 1

Testability Prediction and Test Point Insertion with Graph Convolutional Network (GCN)

Mark Ren, Brucek Khailany, Harbinder Sikka, Lijuan Luo, Karthikeyan Natarajan

Yuzhe Ma, Bei Yu

“High Performance Graph Convolutional Networks with Applications in Testability Analysis”, to appear in Proceedings of Design Automation Conference, 2019
PART 2

Full Chip FinFET Self-heat Prediction using Machine Learning

Miloni Mehta, Chi Keung Lee, Chintan Shah, Kirk Twardowski
PART 1 OUTLINE

- Introduction
- Learning model for testability analysis and enhancement
- Practical issues
  - Scalability
  - Data imbalance
HOW DO WE TEST A CHIP

Input patterns
100010
000101
100111
011101

Output patterns
010101
111111
001111
110101

Golden patterns
010101
101111
001011
110101

GND
Stuck-at-0 fault
TESTABILITY PROBLEM

B’s faults unobservable $\rightarrow$ Difficult-to-test (DT)

B’s faults are observable with an inserted register

Almost always 0
MOTIVATION

- Test Point Insertion Problem:
  - Pick the smallest number of test points to achieve the largest testability enhancement
  - Number of test points $\rightarrow$ chip area cost
  - Number of test patterns $\rightarrow$ test time
- Hard problem, only approximate solutions exist
  - Commercial solution: Synopsys TetraMax
- Can we improve it with Machine Learning?
  - Predict testability
  - Select test points
Given a circuit, predict which gate outputs are difficult-to-test (DT)

- Gate Features: [logic level, SCOAP_C0, SCOAP_C1, SCOAP_OB]
- Gate Label: DT (0 or 1) generated by TetraMax

<table>
<thead>
<tr>
<th>Input Features</th>
<th>Output classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1: 0,0,1,1</td>
<td>N1: 0</td>
</tr>
<tr>
<td>N2: 1,0,1,0</td>
<td>N2: 1</td>
</tr>
<tr>
<td>N3: 2,0,1,1</td>
<td>N3: 0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
BASIC MACHINE LEARNING MODELING

Did not fully leverage the inductive bias of circuit structure

\[ F(a) = [F_a, F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9, F_{10}] \]

ML Models
- LR
- RF
- SVM
- MLP

fanin

fanout

a is DT

a is not DT
GRAPH CONVOLUTIONAL NETWORK (GCN)

Aggregation (mean, sum)
Encoding ($\mathbb{R}^4 \rightarrow \mathbb{R}^{32}$, Relu)
GCN BASED TESTABILITY PREDICTION

Layer 1

Weighted sum & Relu($\mathbb{R}^4 \rightarrow \mathbb{R}^{32}$)

Layer 2

Weighted sum & Relu($\mathbb{R}^{32} \rightarrow \mathbb{R}^{64}$)

Layer 3

Weighted sum & Relu($\mathbb{R}^{64} \rightarrow \mathbb{R}^{128}$)

Fully Connected Layers

(64, 64, 128, 2)
ACCURACY IMPACT OF GCN LAYERS (K)

Training Accuracy (%)

Testing Accuracy(%)

Epochs

K=1

K=2

K=3
EMBEDDING VISUALIZATION

• Embeddings looks more discriminative as stage increase;
MODEL COMPARISON ON BALANCED DATASET

- Compare with basic ML modeling: LR, RF, MLP, SVM
  - N=500 nodes in fanin cone and 500 nodes in fanout cone, a total of 1000 nodes
- Compare to 3-layer GCN
  - Less than 1000 nodes influence each node, comparable with the baseline
- GCN has the best accuracy (93%).
**TEST POINT INSERTION WITH GCN MODEL**

- An iterative process to select TPs enabled by GCN model
- Select TP candidate based on predicted impact
  - Number of reduced DTs in the fanin cone of TP

Diagram:
- Circuit → Graph → GCN Model → TP Candidates
  - Graph Modification
  - GCN Model
  - Impact Estimation
  - Point Selection
  - Done?
    - Y: Final TPs
    - N: new TP → Graph Modification

Diagram Key:
- Graph Modification
- GCN Model
- Impact Estimation
- Point Selection
- Done?
TEST POINT INSERTION RESULTS COMPARISON
Machine learning can improve approximate solutions for hard problems

- 11% less test points with 6% less test pattern under same coverage vs TetraMax.

![Test point reduction and test pattern reduction chart]

- Test point reduction
- Test pattern reduction
MODEL SCALABILITY

- Choices of model implementation
  - Batch processing: Recursion
  - Full graph: Sparse matrix multiplication
    \[ E_k = ReLU((A \ast E_{k-1}) \ast W_k) \]
- Tradeoff
  - Memory vs speed
- 1M nodes/second on Volta GPU
MULTI GPU TRAINING

- Training dataset has multiple million gates designs that cannot fit on one GPU
- Data parallelism, each GPU computes one design/graph
- Replicate models across multiple GPUs
- Leverage PyTorch DataParallel module
- Trained with 4 Tesla V100 GPUs on DGX1
IMBALANCE ISSUE

- It is very common to have much more non-DTs (negative class) than DTs (positive class), imbalance ratio more than 100X

<table>
<thead>
<tr>
<th></th>
<th>Predict: 0</th>
<th>Predict: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact: 0</td>
<td>133576</td>
<td>290</td>
</tr>
<tr>
<td>Fact: 1</td>
<td>3681</td>
<td>432</td>
</tr>
</tbody>
</table>

Classifier 1: ok precision, low recall

Recall: 10.5%
Precision: 59.8%

<table>
<thead>
<tr>
<th></th>
<th>Predict: 0</th>
<th>Predict: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact: 0</td>
<td>100919</td>
<td>32927</td>
</tr>
<tr>
<td>Fact: 1</td>
<td>114</td>
<td>4069</td>
</tr>
</tbody>
</table>

Classifier 2: high recall, low precision

Recall: 97.3%
Precision: 11.0%
MULTI-STAGE CLASSIFICATION

- The networks on initial stages only filter out negative data points with high confidence
  - High recall, low precision
- Positive predictions are sent to the network on the next stage
## Multi-Stage Classification Result

**Balanced Recall and Precision**

<table>
<thead>
<tr>
<th>Stage 1</th>
<th>Pred: 0</th>
<th>Pred: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact: 0</td>
<td>100919</td>
<td>32927</td>
</tr>
<tr>
<td>Fact: 1</td>
<td>114</td>
<td>4069</td>
</tr>
</tbody>
</table>

Recall: 97.3%
Precision: 11.0%

<table>
<thead>
<tr>
<th>Stage 2</th>
<th>Pred: 0</th>
<th>Pred: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact: 0</td>
<td>26935</td>
<td>5992</td>
</tr>
<tr>
<td>Fact: 1</td>
<td>221</td>
<td>3848</td>
</tr>
</tbody>
</table>

Recall: 94.6%
Precision: 39.1%

<table>
<thead>
<tr>
<th>Stage 3</th>
<th>Pred: 0</th>
<th>Pred: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact: 0</td>
<td>5207</td>
<td>785</td>
</tr>
<tr>
<td>Fact: 1</td>
<td>309</td>
<td>3539</td>
</tr>
</tbody>
</table>

Recall: 92.05
Precision: 81.8%

<table>
<thead>
<tr>
<th>Overall</th>
<th>Pred: 0</th>
<th>Pred: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact: 0</td>
<td>133061</td>
<td>785</td>
</tr>
<tr>
<td>Fact: 1</td>
<td>574</td>
<td>3539</td>
</tr>
</tbody>
</table>

Recall: 86.0%
Precision: 81.8%
PART 1 - SUMMARY

- Machine learning can improve VLSI design testability beyond the existing solution
  - Predictive power of ML model
- Graph based model is suitable for VLSI problems
- Practical issues such as scalability and data imbalance need to be dealt with
PART 2

Full Chip FinFET Self-heat Prediction using Machine Learning

Miloni Mehta, Chi Keung Lee, Chintan Shah, Kirk Twardowski
VLSI TESTABILITY AND RELIABILITY

Design → Manufacturing → Wafer → Chip

Reliability

Testability

Years

Pass → Fail

Testing
SEMICONDUCTOR RELIABILITY

Evolving Reliability Needs for Semiconductors

New application trends push requirements in system reliability

Source: https://semiengineering.com/improving-automotive-reliability/
RELIABILITY

DEVICE SELF-HEAT (SH)

- Active power in transistors dissipated as heat to the surroundings
- FinFETs are more sensitive to SH than planar devices
- Why do we care?
  - Exacerbates Electro-migration (EM) on interconnects
  - Transistor threshold voltage ($V_t$) shifts
  - Time dependent dielectric breakdown (TDDB)
SH METHODOLOGIES SO FAR

- No sign-off tool that can handle full chip SH analysis
- Limitations using Spice simulations
  - Impractical to run on billions of transistors
  - Teams review high power density cells
- 2D Look-up Table approach
  - Based on frequency and capacitive loading for different clock drivers
  - Reduced run time by more than 90% over full Spice simulations
  - Pessimistic wrt Spice
SELF-HEAT TRENDS

- Frequency $\propto$ SH
- Capacitive loading $\propto$ SH
- Cell size $\propto$ $1/SH$
- Resistance $\propto$ $1/SH$ (non-linear)
MOTIVATION TO USE ML

- Identify problematic cells in the design without exhaustive Spice simulations
  - Complex relationship between design and SH
  - Design database available for several projects
  - Reusability across projects
- Focus
  - Clock inverters and buffers
  - Quick, easy, light-weight
  - Rank cells above certain SH threshold for thorough analysis
MACHINE LEARNING MODEL

Select Training Data

Get Attributes from PrimeTime

Simulate in HSPICE

Generate ML Model

Equation: $Y^* = ?$

Select Test Data

Get Attributes from PrimeTime

Simulate in HSPICE

Prediction on Test Set

(Predicted == Spice)?

Ready for Deployment

No

Yes

Validation

$X_{training}$

$Y_{training}$

$X_{test}$

$Y_{pred-test}$

$Y_{test}$
DATASET SELECTION

- Cover a wide range of frequencies
- Cover different types of standard cell sizes
- Prevent duplication in training data due to replicated partitions/chiplets
- Outliers in the design chosen
- Labels obtained through Spice simulations (supported from foundry spice models)
- TSMC 16nm FinFET training model used 4300 training samples with 9 features
DNN REGRESSOR MODEL

Features:
- Output Capacitance
- Frequency
- Cell size
- Net resistance
- Input slew
- Output slew
- # of output loads
- Input Capacitance of loads
- Avg transition on load

\[
\text{Cost} = \frac{\sum (Y_{\text{pred}} - Y)^2}{N}
\]
MINIMIZING COST FUNCTION

- Gradient descent
- Adam optimizer which has adaptive learning rate
- Exponential Linear Unit (ELU) used as activation function
- 300,000 training steps
RESULTS

- Xavier CPU 2000 validation samples

- Good correlation between DNN prediction and Spice SH

- Average err % wrt Spice = 6.5%

- MSE = 0.05
QUANTITATIVE BENEFITS

- Trained model is deployed for inference on millions of clock cells
  - Training time: 37 minutes (DGX1 used)
  - Inference time: <1min
- >99% cells filtered from Spice simulations!
- Top 1000 prediction results simulated and verified
- Found small clock tree cells had highest SH
- Outlier detection improved inference by 2.65% in Turing
COMPARISON TO PRIOR WORK

![Comparison to Prior Work Diagram](image-url)
PART 2 - SUMMARY

- FinFET Self-Heat is a growing reliability concern
- Proposed supervised ML model using DNN
  - Accurately predict Self-heat
  - 100x runtime improvement
- Displayed techniques to select representative dataset for training
- Model deployed for Xavier and Turing projects
- Use ML techniques to improve productivity and solve challenging problems in VLSI