Machine Learning Techniques for Chip Design Verification and Prediction

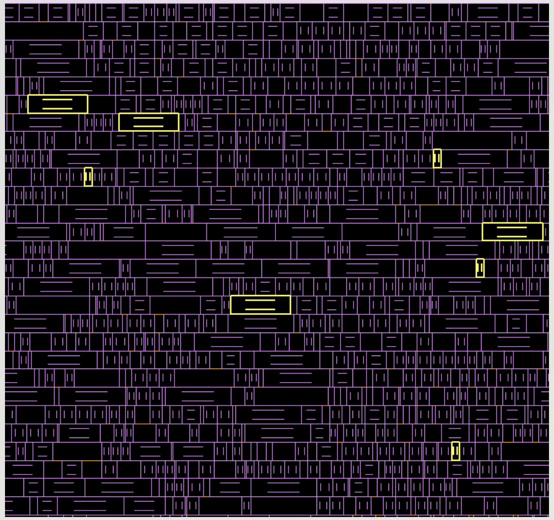
Jiang Hu

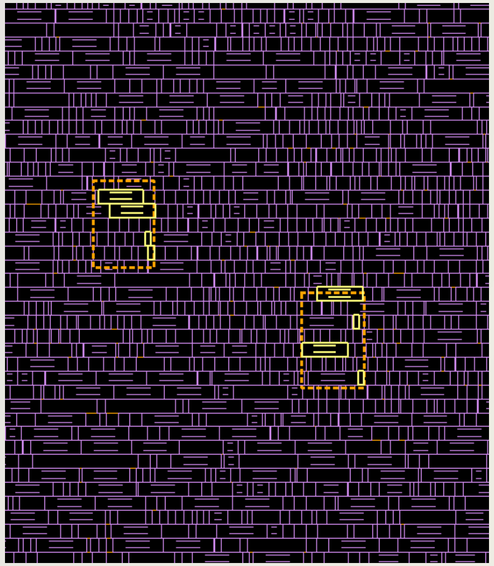
Department of Electrical and Computer Engineering

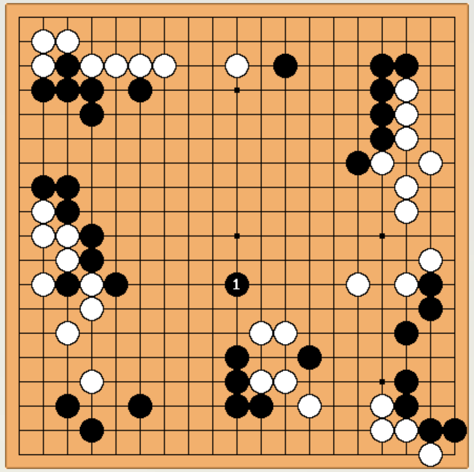
Texas A&M University

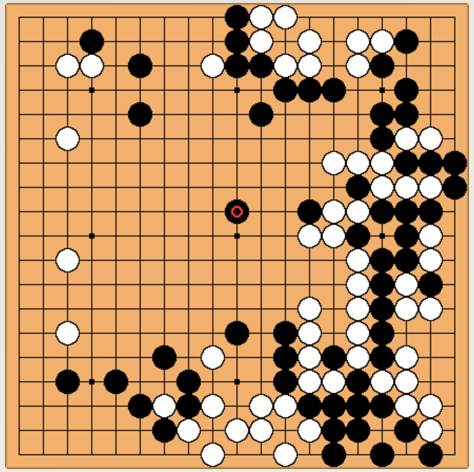
Why Machine Learning?
Optimization vs. Knowledge Reuse

- Example: cell placement
- Given netlist and cell library
- Find cell locations
- No cell overlap
- Wirelength is minimized

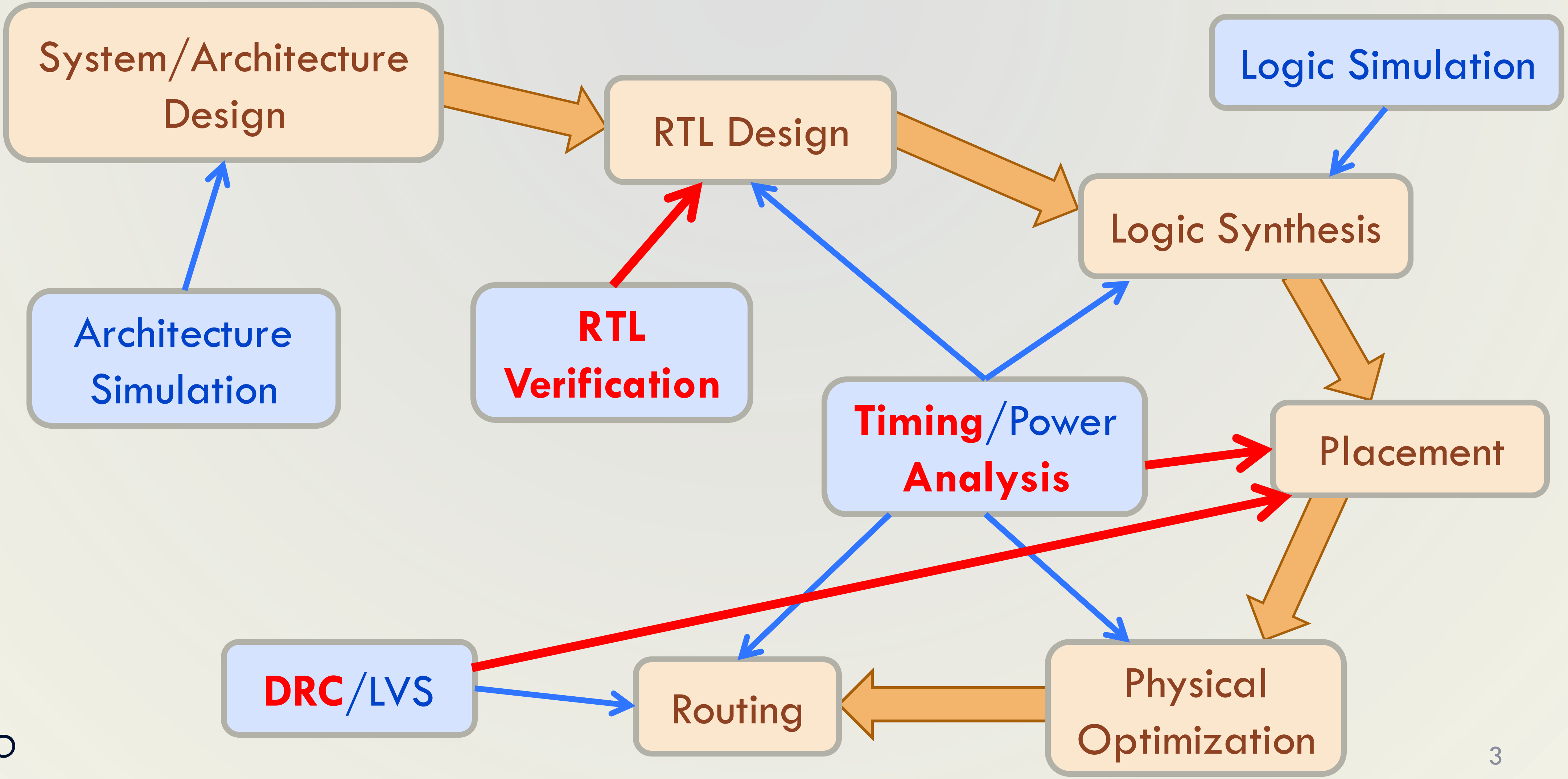








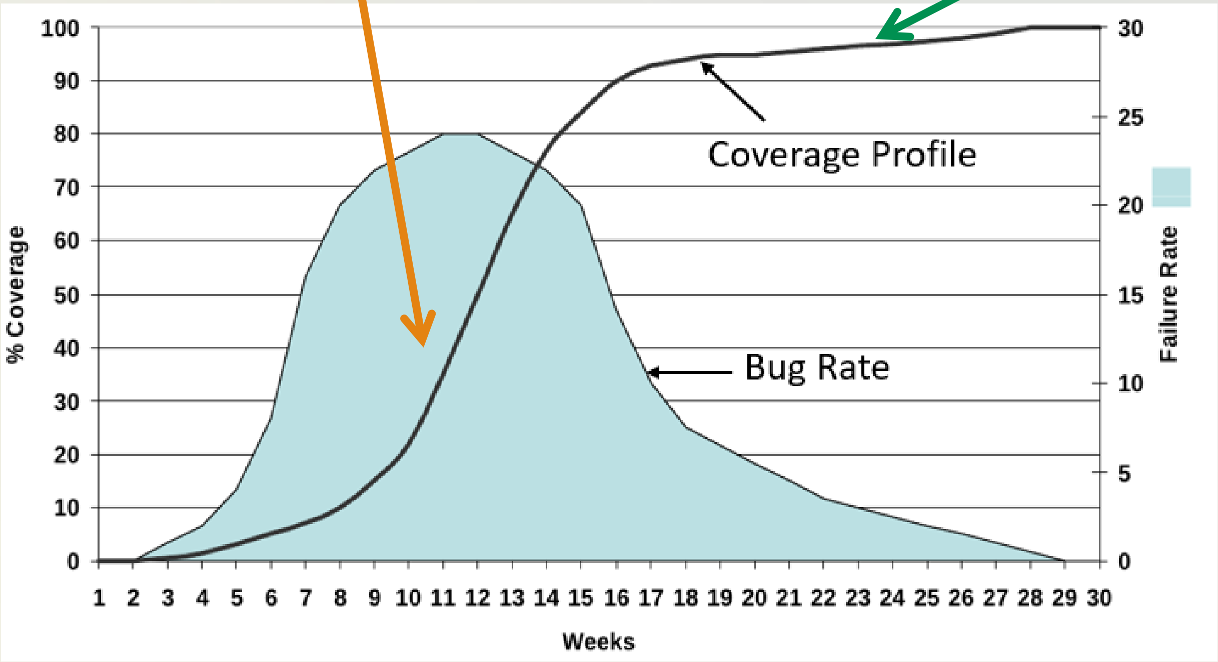
What Can We Do? Prediction



Machine Learning for Fast Functional Verification Coverage

Phase I:
- Random test generation
- ML model is trained

Phase II:
- ML model is applied for test pruning
- ML model continues to be trained



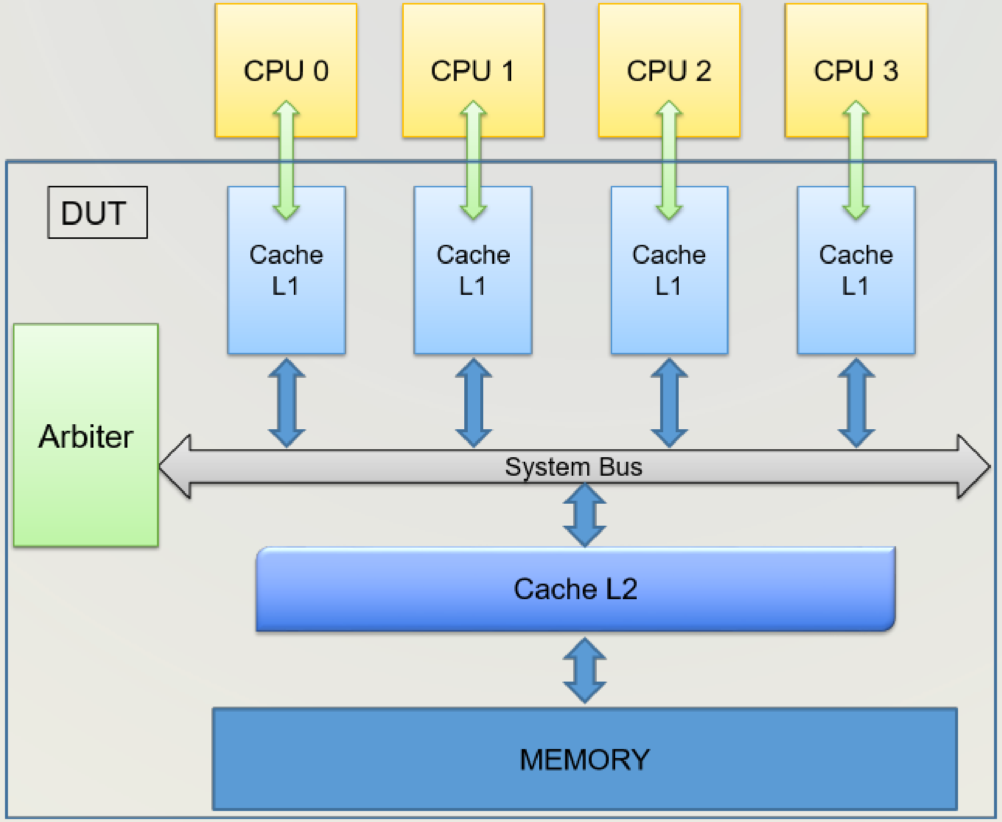
Pruning

- ML predicts if a stimulus ψ will cover an unverified point
- Simulate ψ only if the answer is yes

Machine Learning Model Setup

Features

- Seed
- #transactions
- Core selection
- $type
- Request type
- …

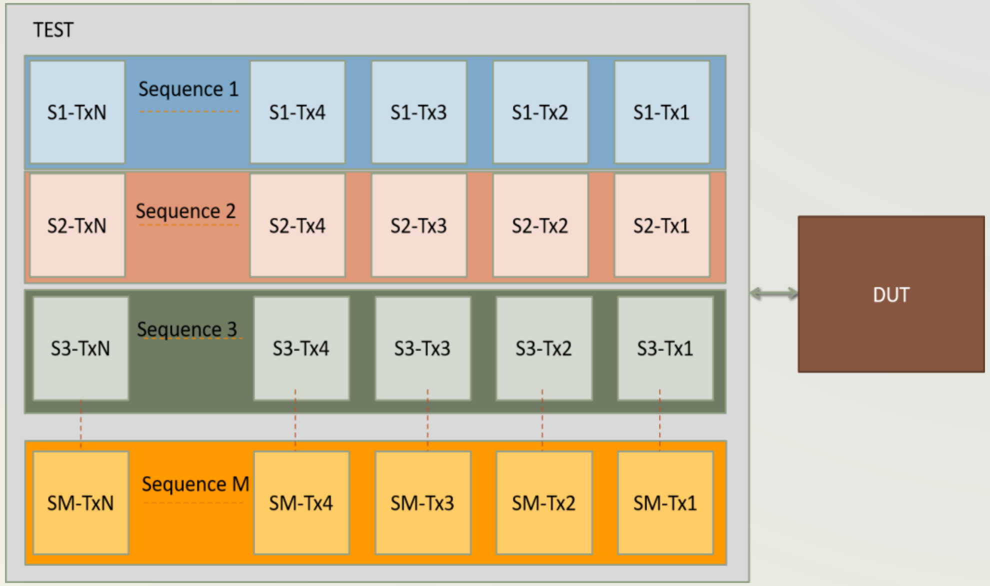


Labels

- Address X req type in bins
- Snoop request
- $protocol transitions
- $hit on each address
- …

Transaction-Level Stimulus Optimization

Finer-grained control than test-level optimization

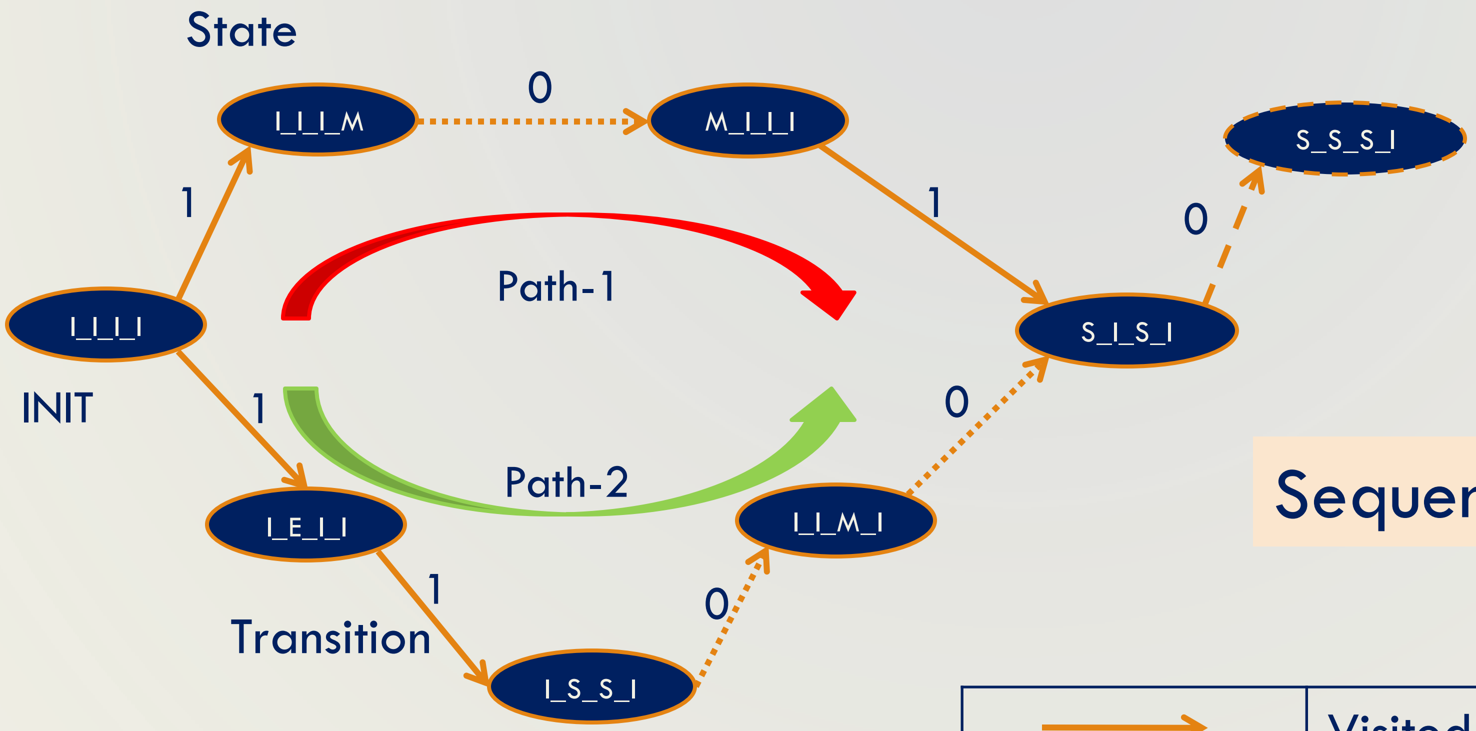


	Finite State Machine	Non-FSM
Online transaction pruning		
Offline sequence generation		

Offline Sequence Generation for FSM

- Coverage metric: state transitions
- ML model: given current state and transaction attribute, predict the next state
- Phase 1: random simulation while ML model is trained
- Phase 2: generate transaction sequences leading to new transitions

Sequence Generation by Graph

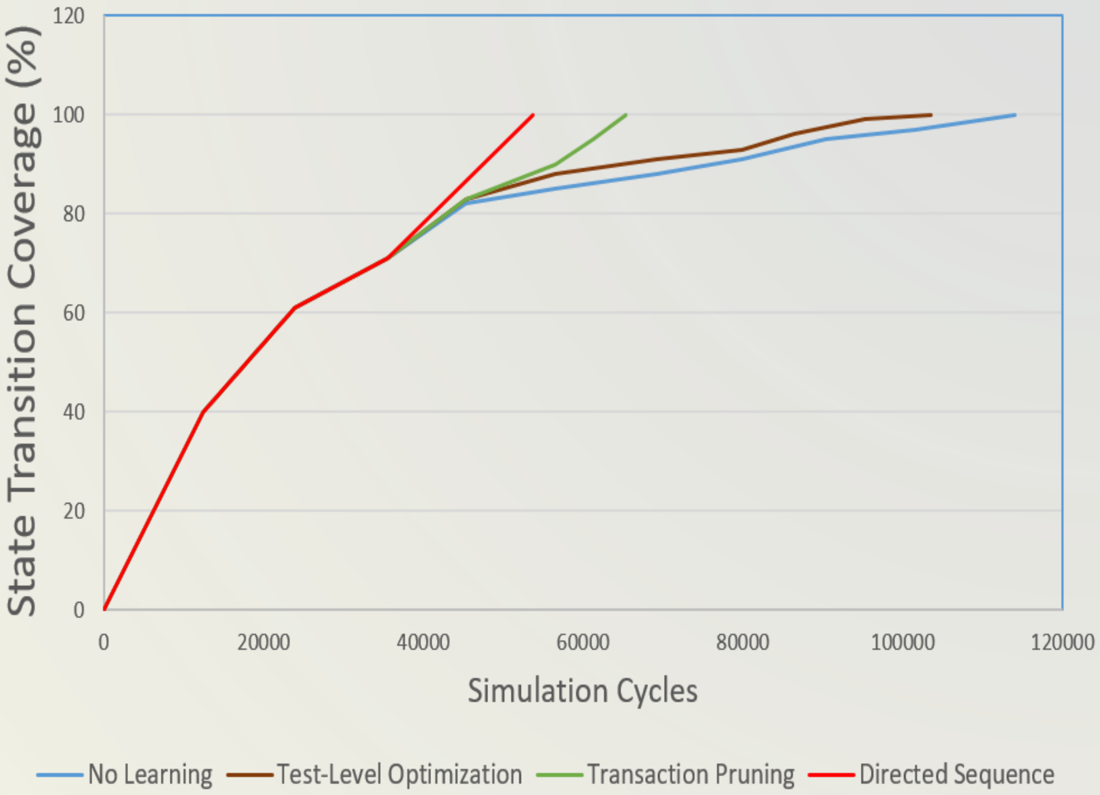


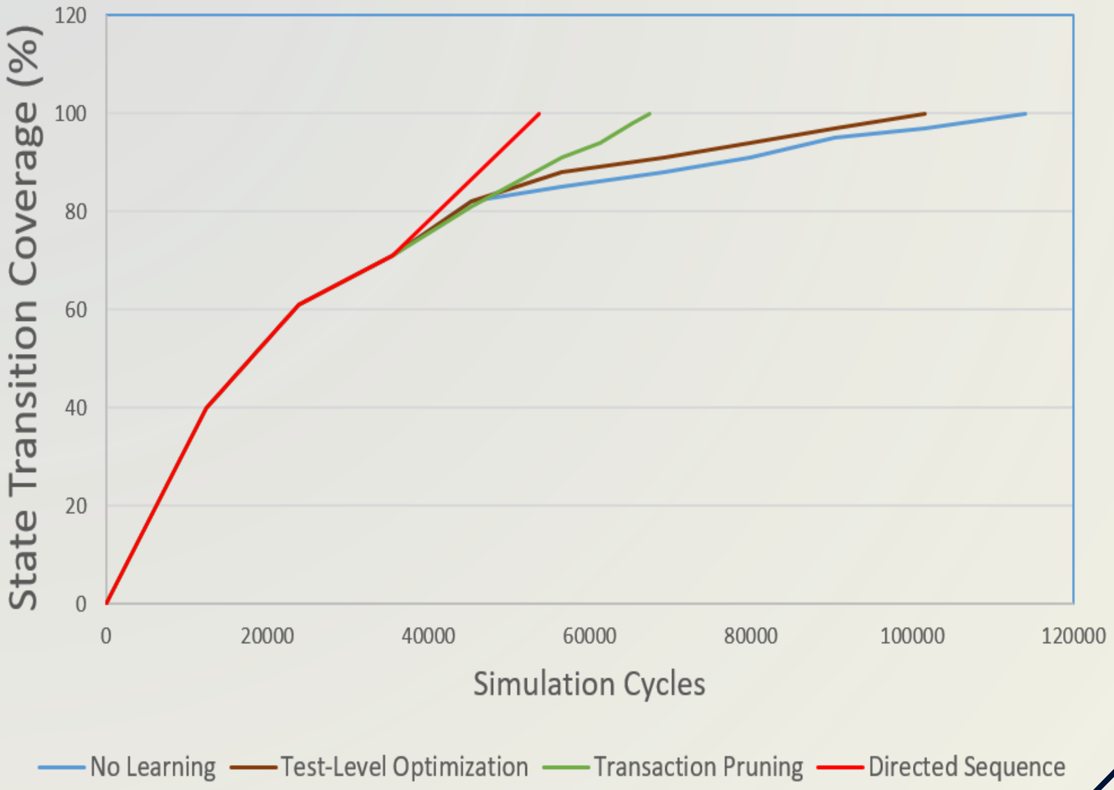
Sequence = shortest path

→	Visited edge
⇢ (dotted)	Unvisited and predicted edge
⇢ (dashed)	Unvisited and illegal prediction

FSM Transaction Optimization Results

Coverage Metric: MESI state transitions – 143 bins



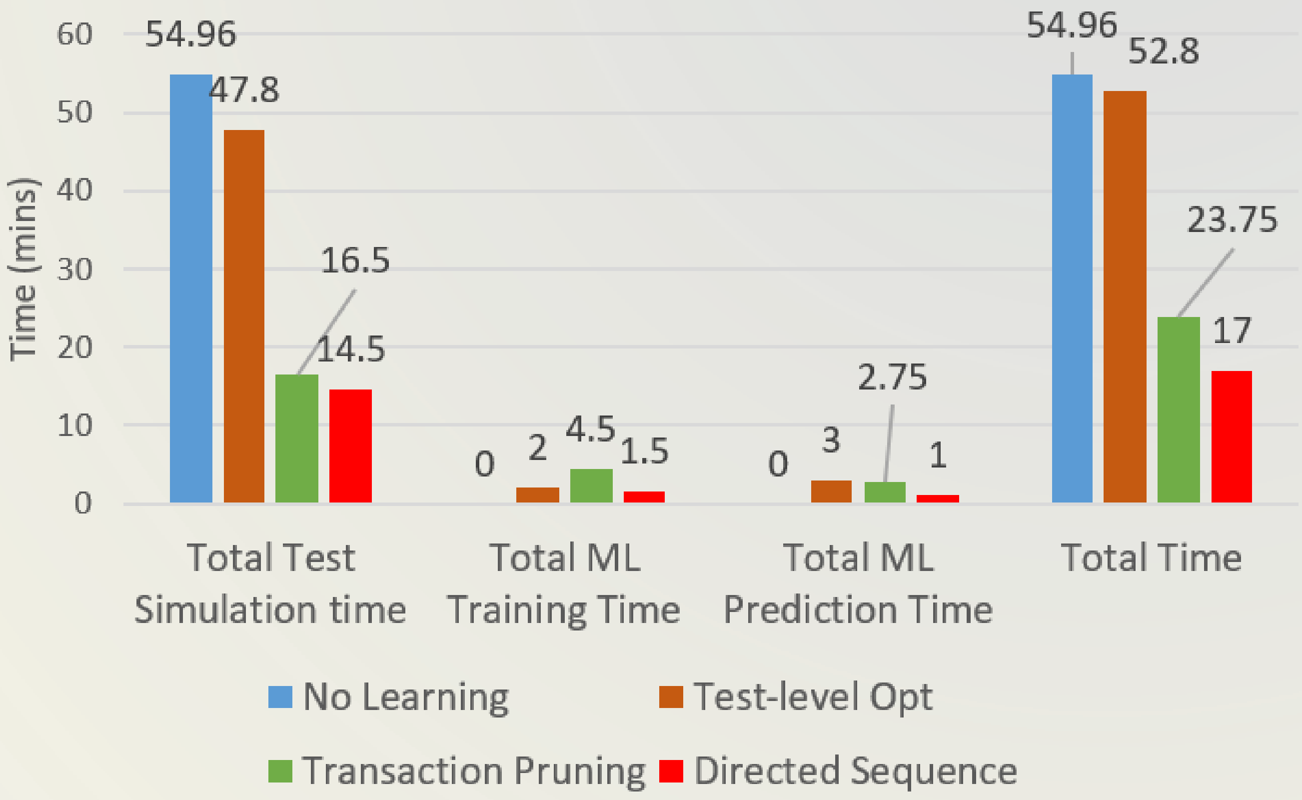


Deep Neural Network (DNN)
48% reduction in simulation cycles

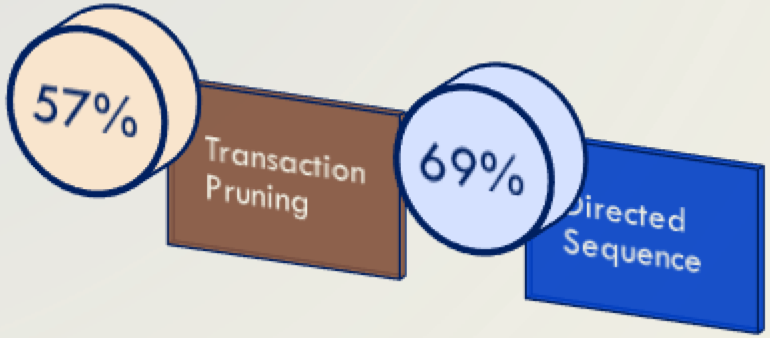
Random Forest Classifier (RF)
55% reduction in simulation cycles

FSM Verification Time

ML engine: random forest



Verification time reduction

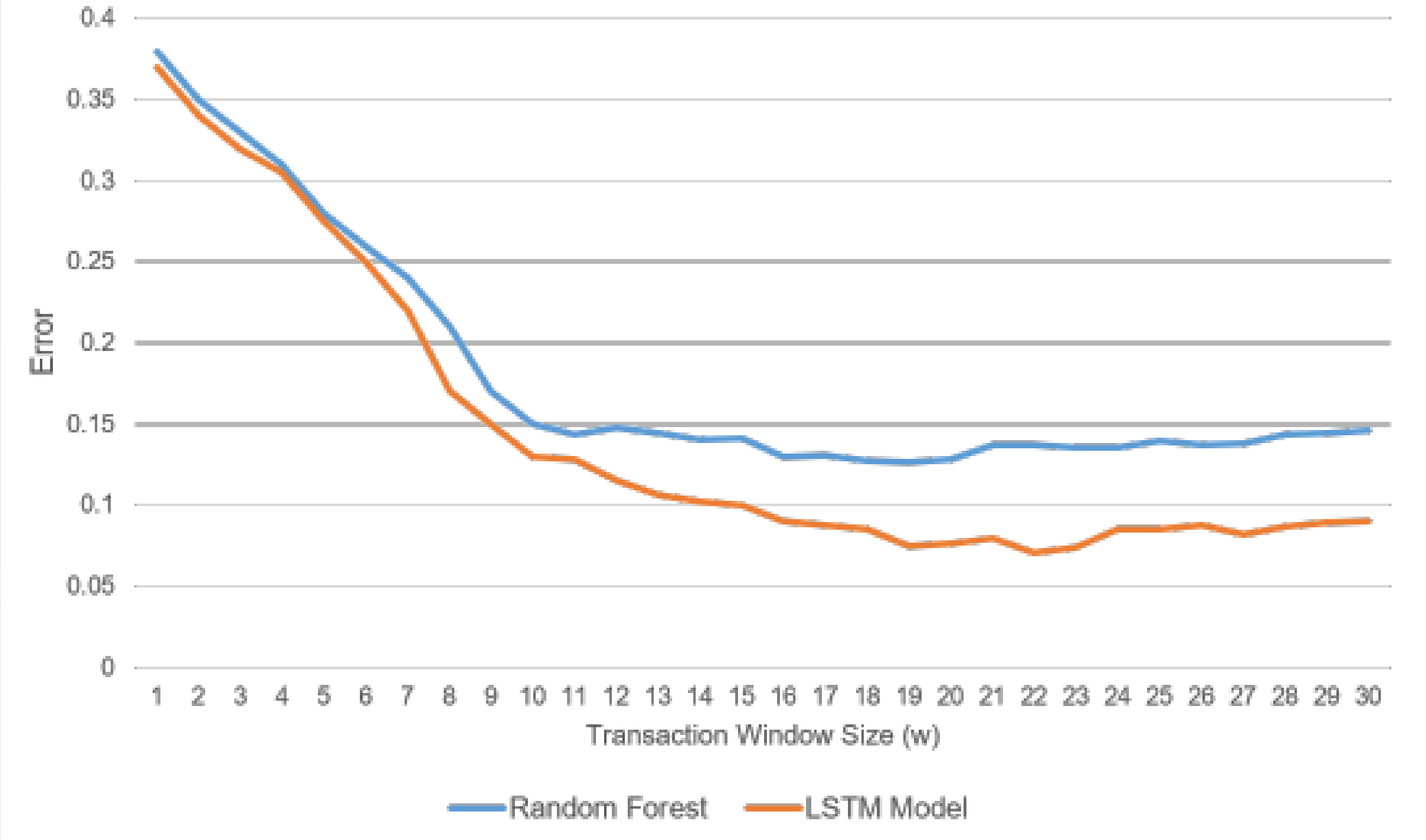


Non-FSM Event Coverage

- Events: buffer full, cache hit, etc.
- Almost impossible to deterministically cover events through test-level optimization
- Event coverage depends on transaction history

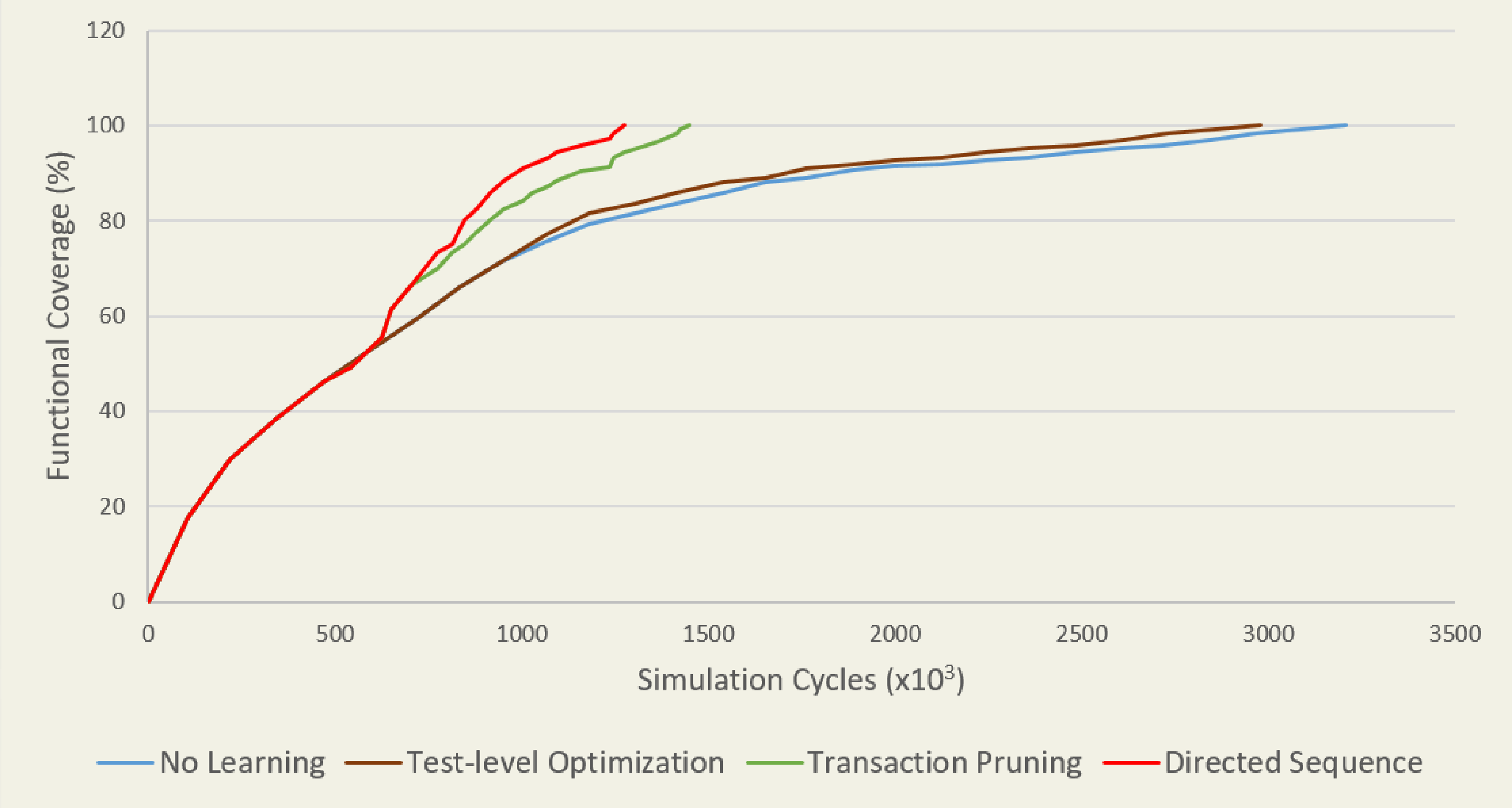
History Effect

Model error



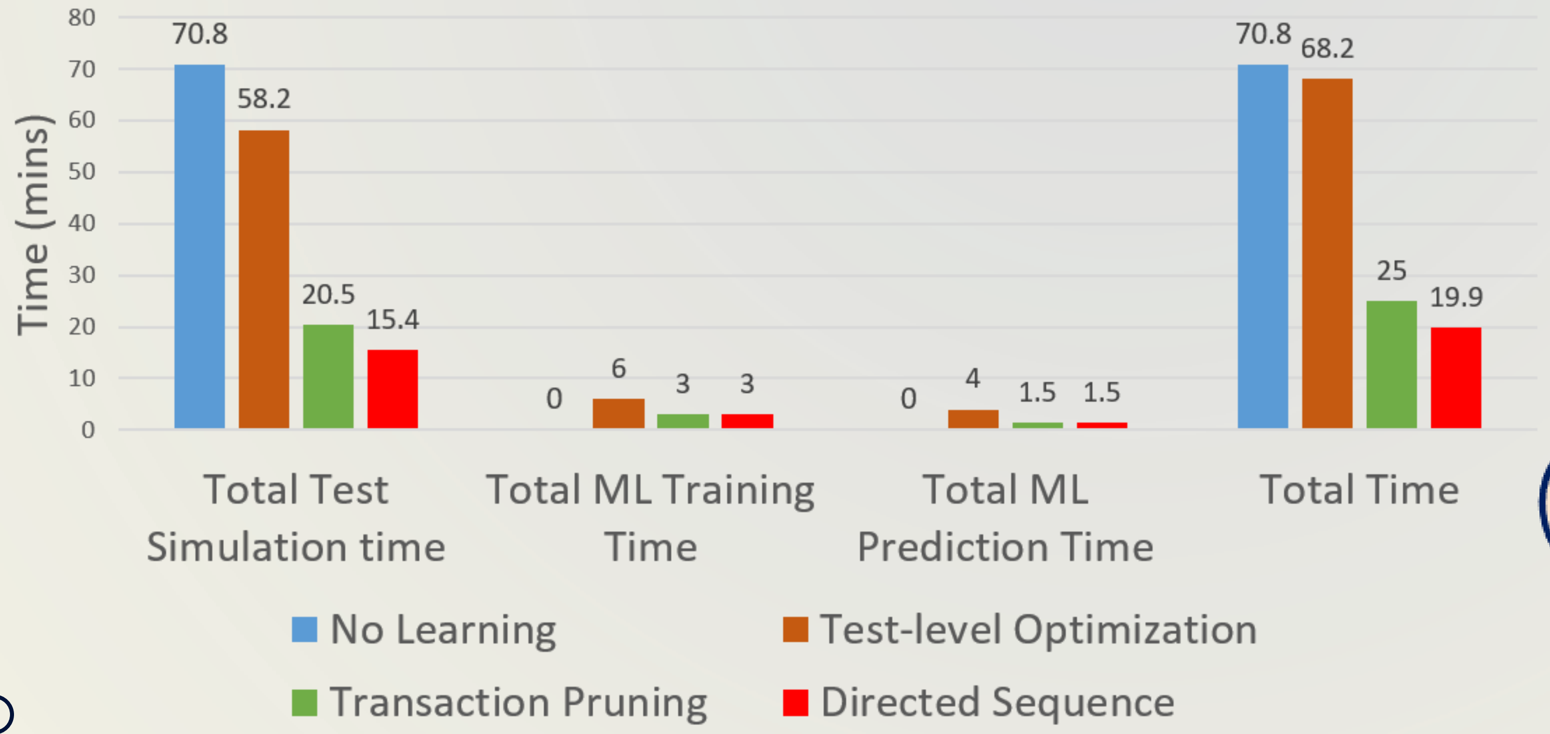
Non-FSM Event Coverage Results

Coverage Metric: cache hit on every address – 768 bins

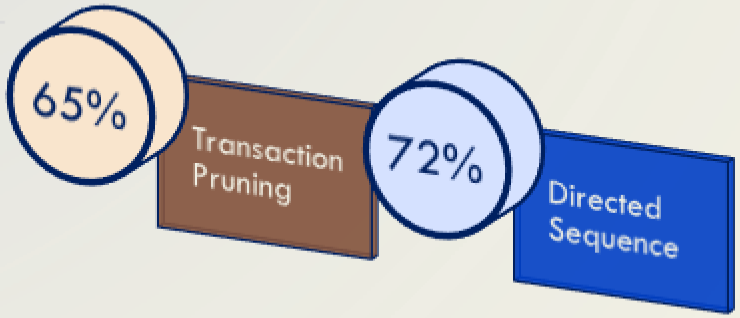


Long Short-Term Memory (LSTM) 61% reduction in simulation cycles

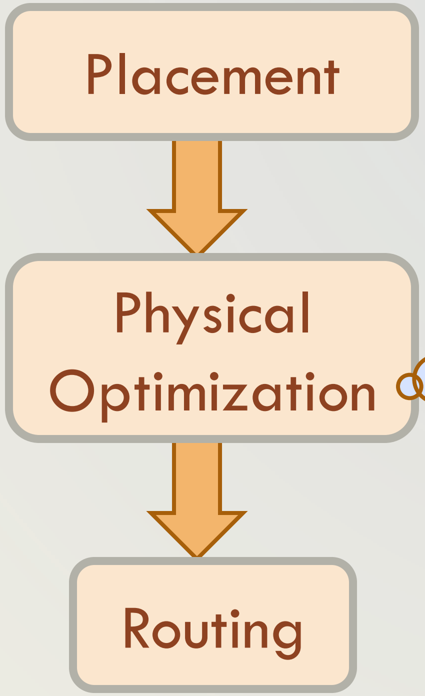
Non-FSM Verification Time

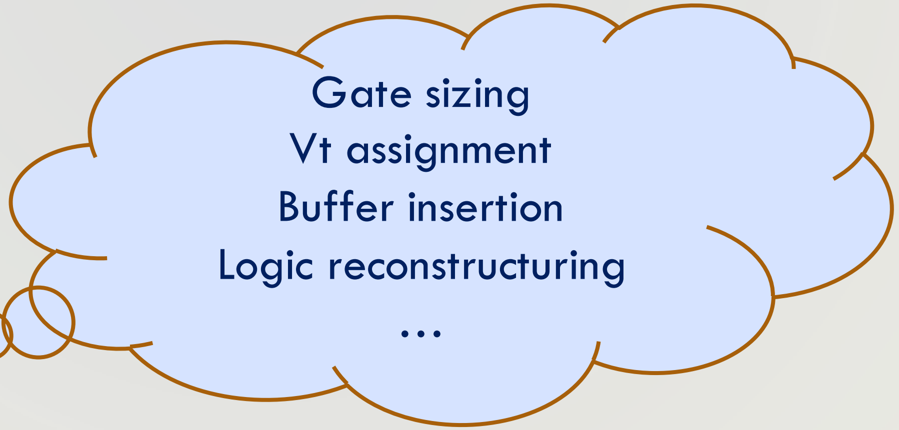


Verification time reduction



Pre-routing Timing Prediction

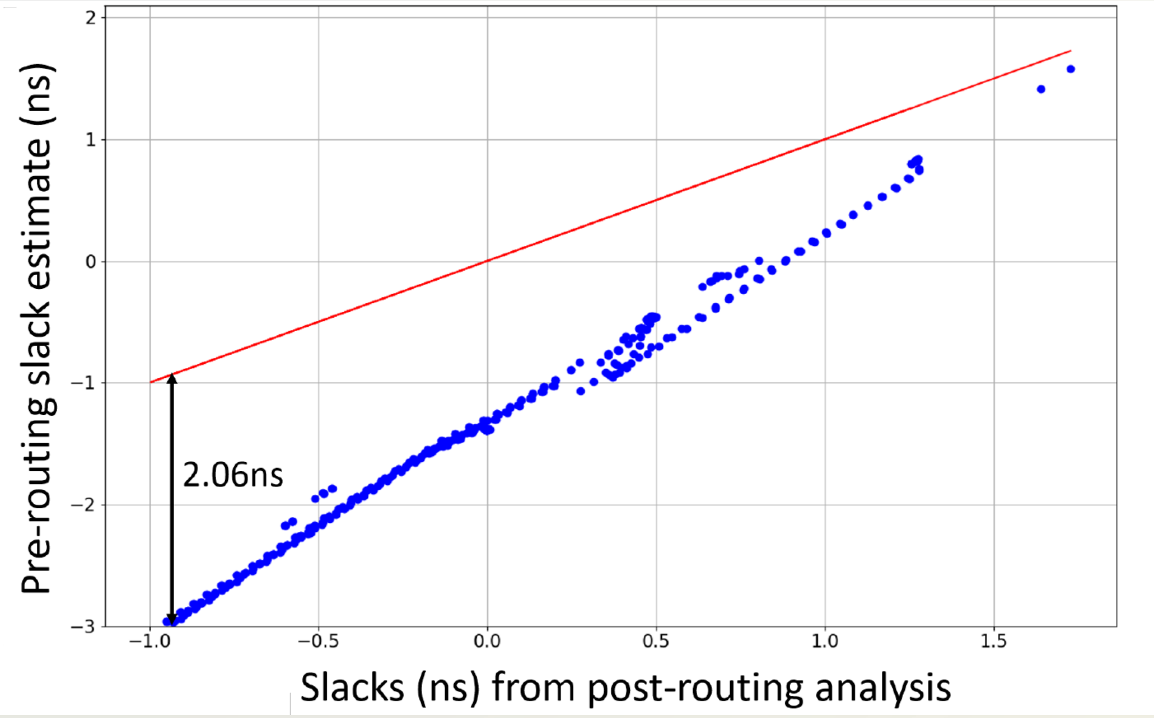




Capture routing effect before routing

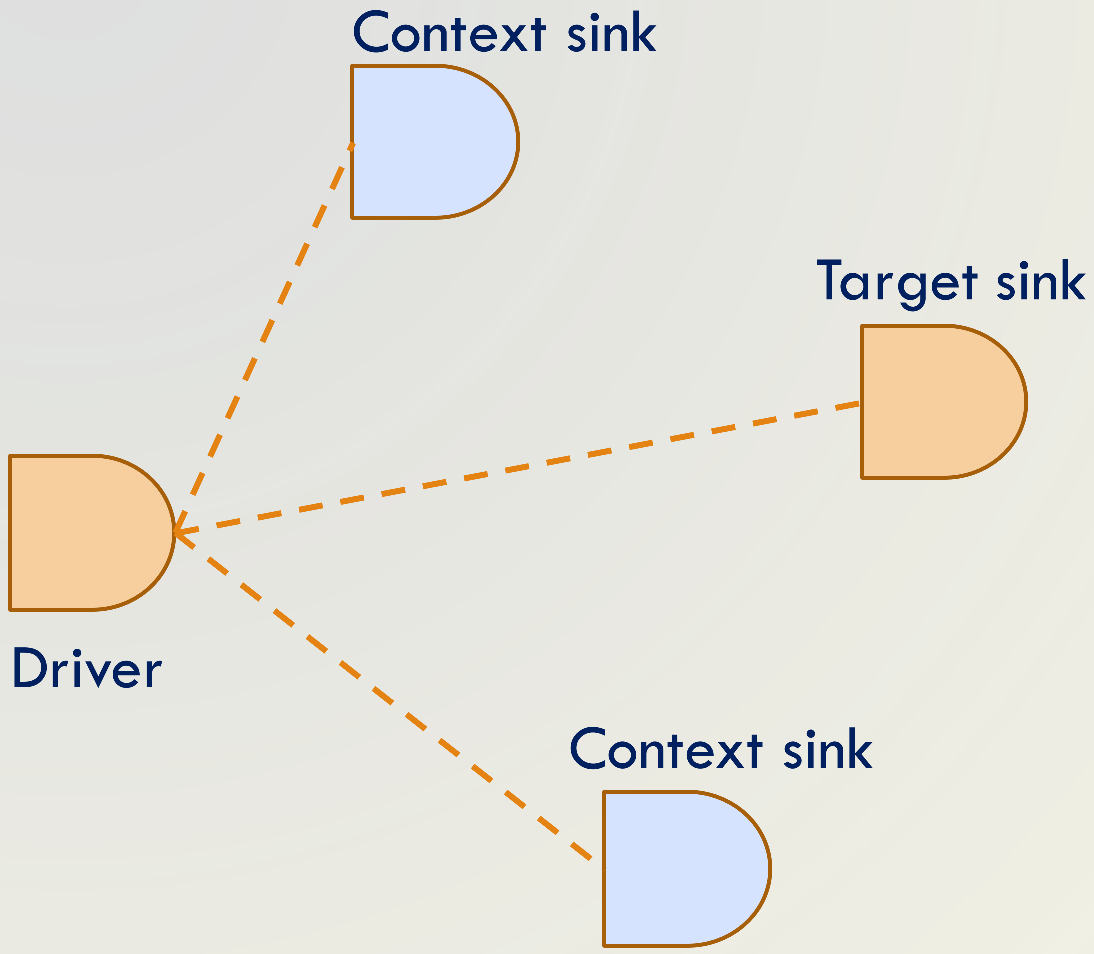
Existing Timing Prediction Techniques

- Trial routing
  - Too slow
- Empirical analytical formula
  - Inaccurate and pessimistic

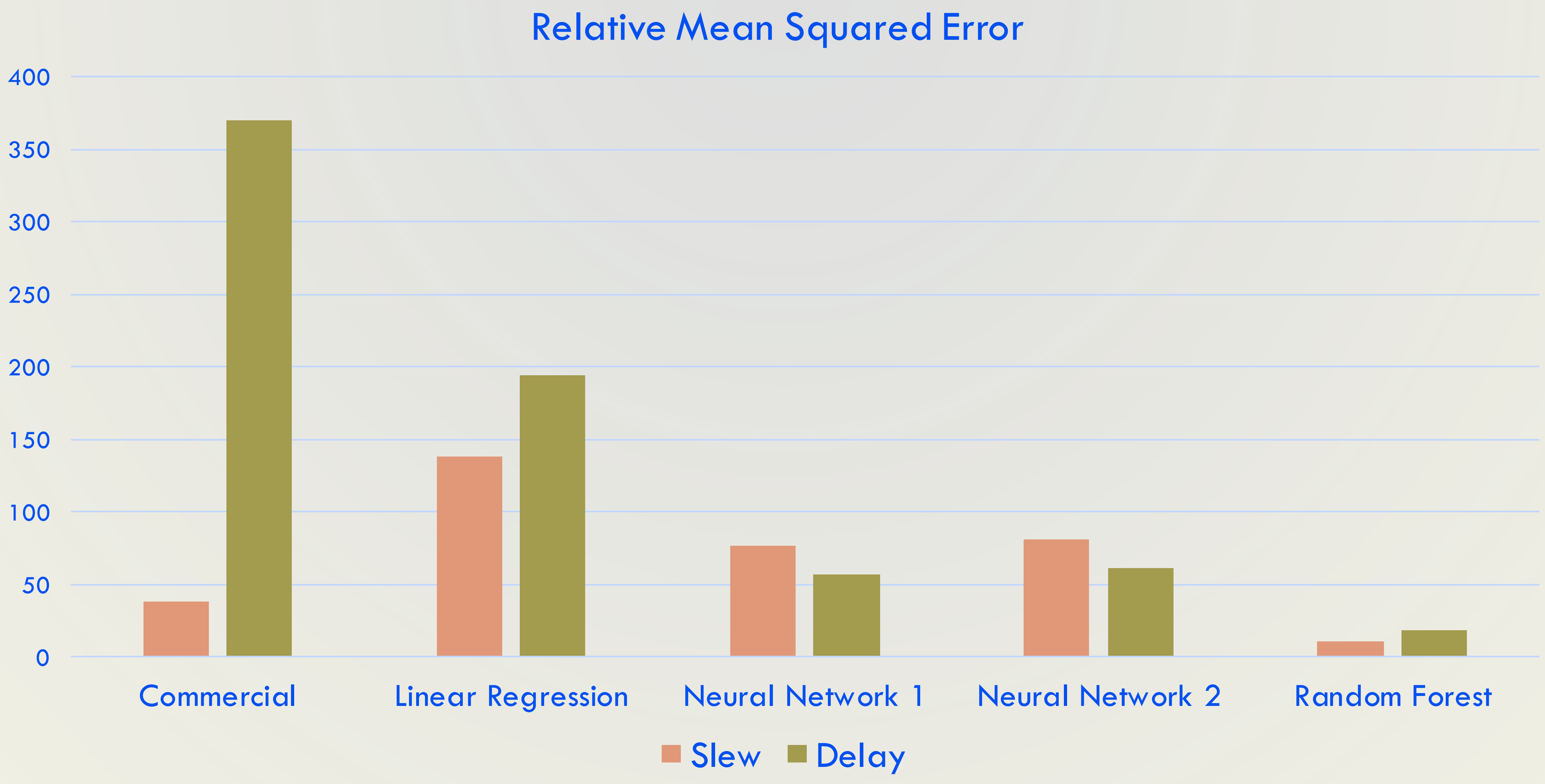


Machine Learning Driver-Sink Delay Model

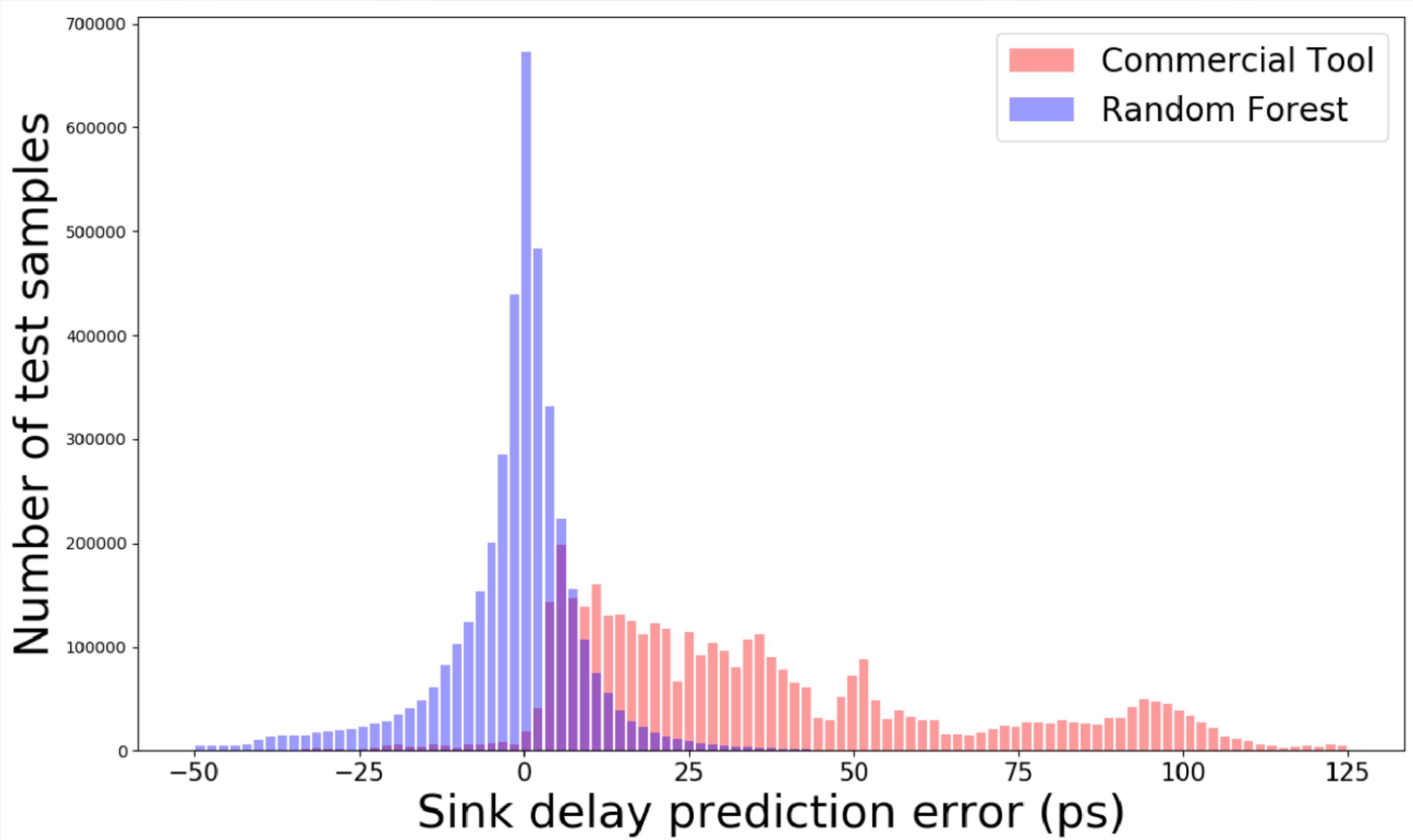
- Features
  - Driver output cap
  - Target sink cap
  - Total sink cap
  - Driver-sink distance
  - Input slew
  - Context sink locations
- Output labels
  - Driver-sink delay
  - Slew at target sink
  - Post-routing timing analysis



Sink Prediction Accuracy on ITC99 Circuits

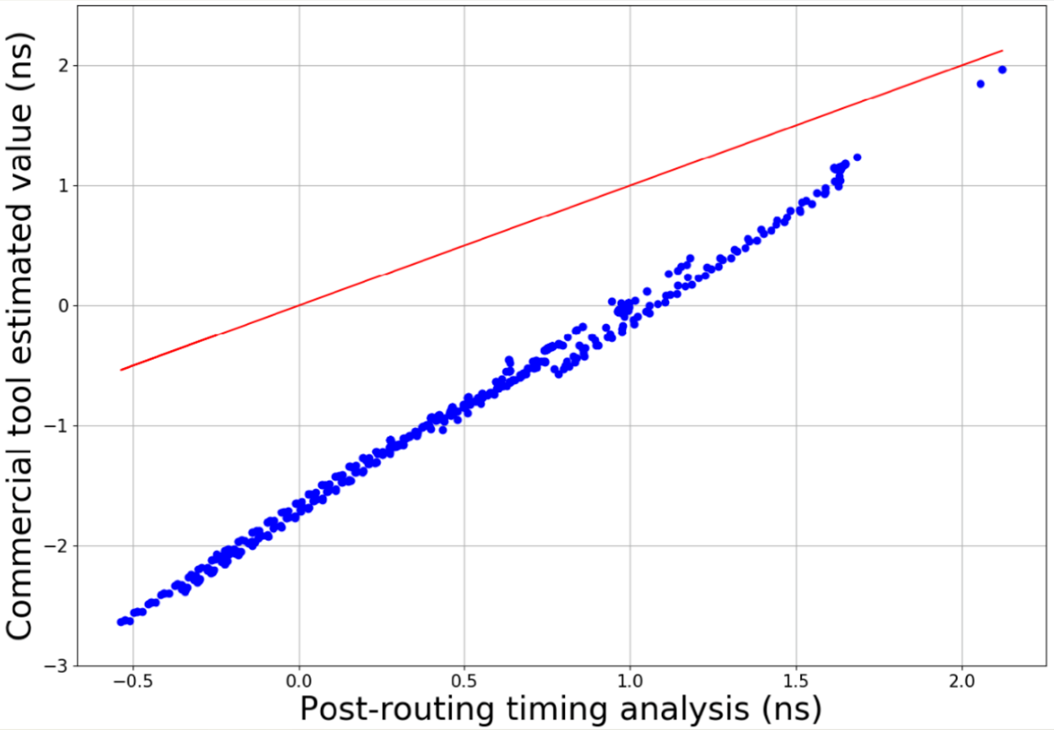


Delay Error Distribution

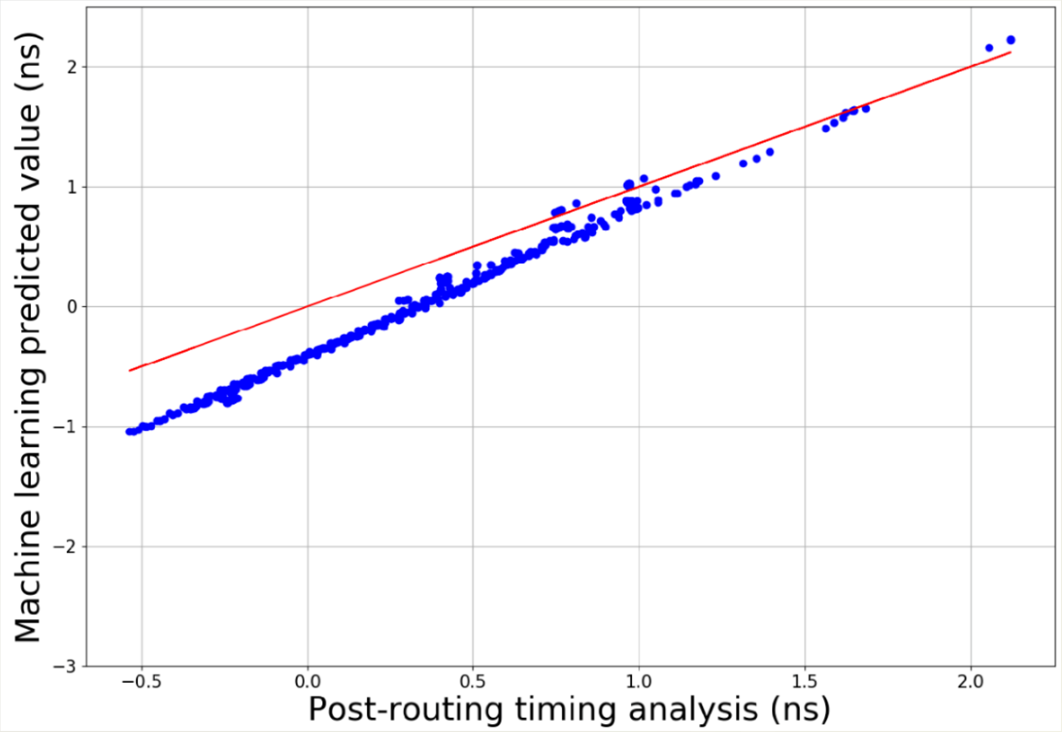


Path Slack Prediction Results

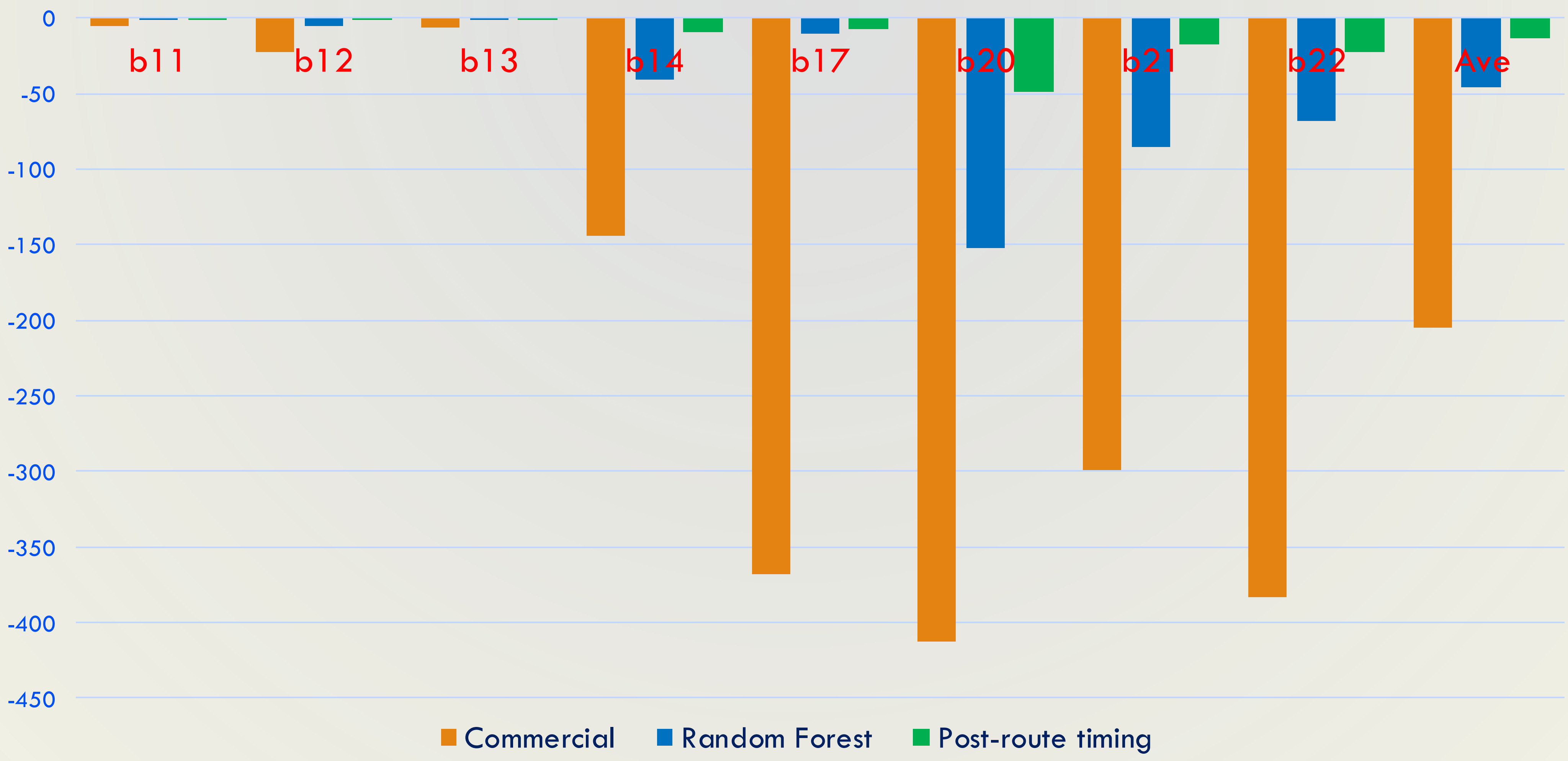
Commercial Tool



Random Forest



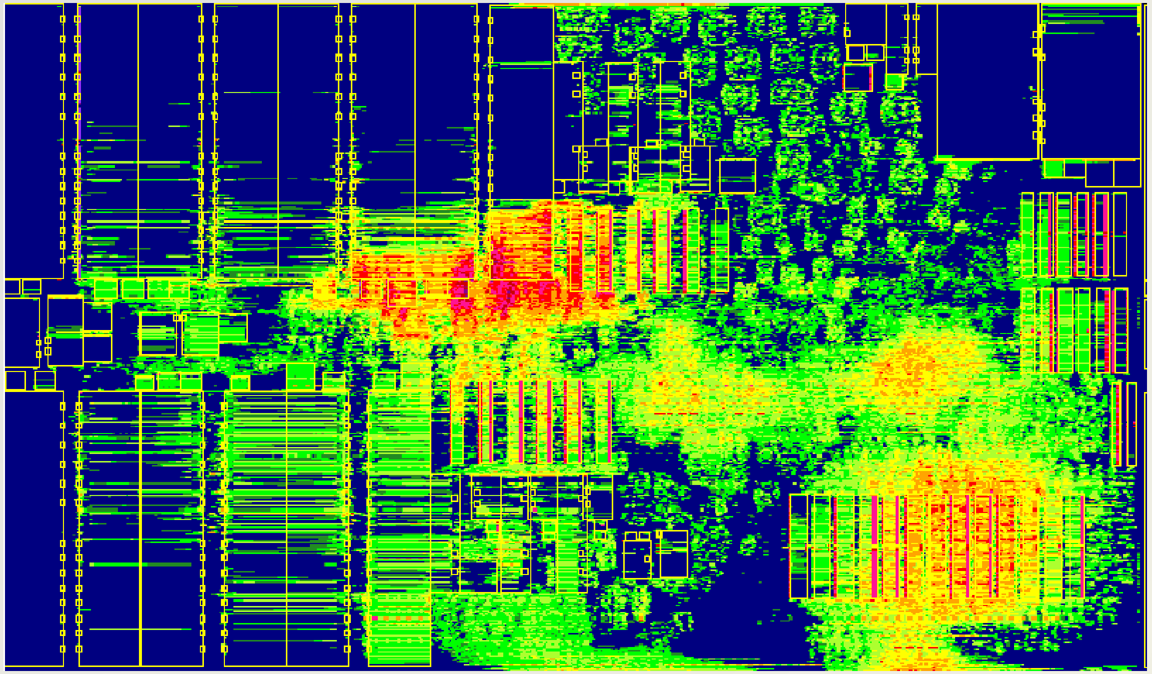
Total Negative Slack Results



Random forest prediction is 30X faster than routing + timing analysis

Early Routability Prediction

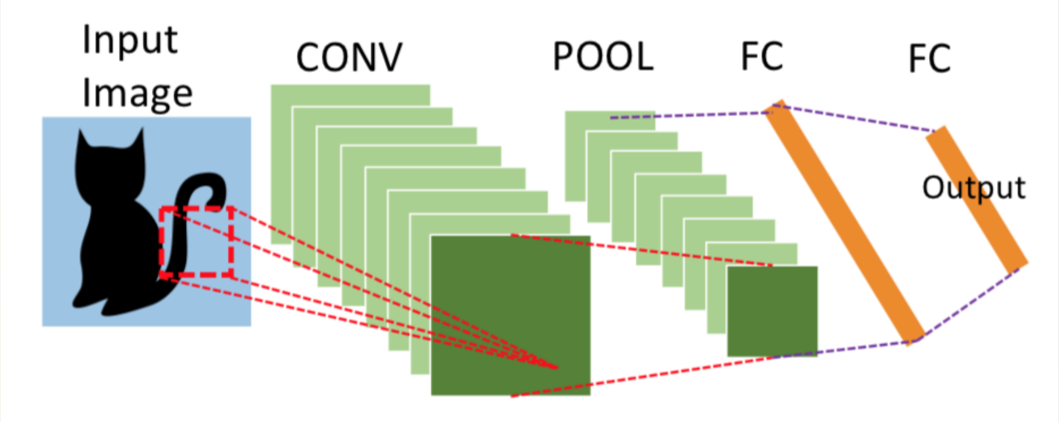
- Routability: post-routing design rule violations
- Early prediction at placement stage
- Analytical techniques
  - Very fast
  - Not enough fidelity
- Trial routing
  - Acceptable fidelity
  - Not fast enough



Problem Formulations

- Predicting overall number of design rule violations (#DRV)
  - Given two placement solutions, tell which is more routable with high fidelity
- DRV hotspot detection
  - Given a relatively routable placement solution, pinpoint DRV hotspots such that mitigation measures are well targeted

Convolutional Neural Network for #DRV Prediction



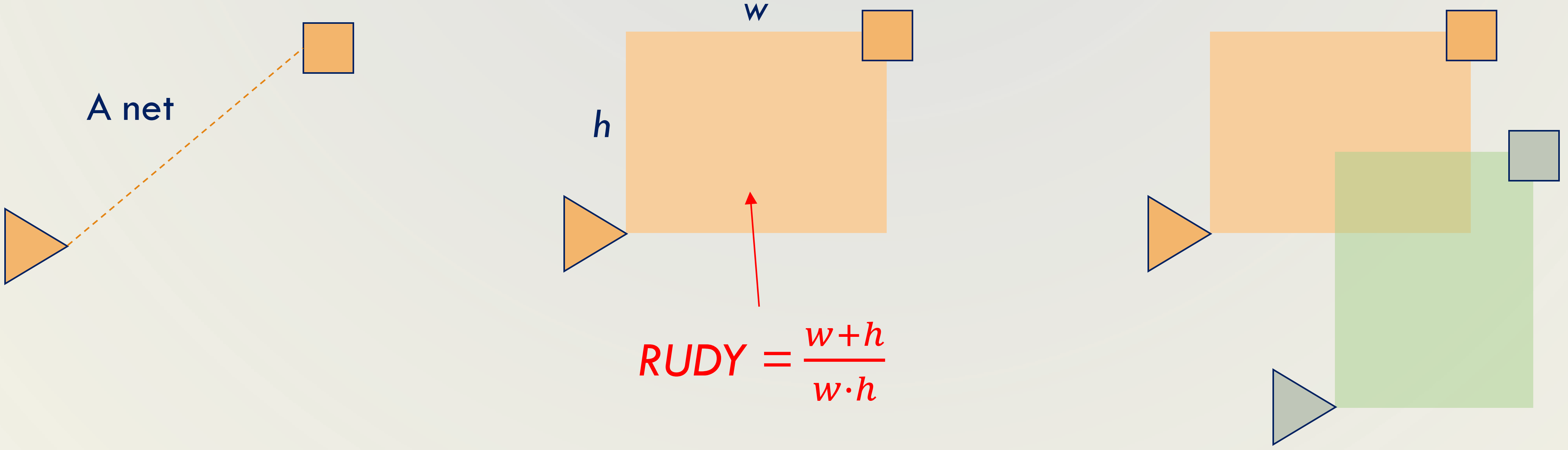
Convolutional (CONV), Pooling (POOL) and Fully Connected (FC) layers

Widely used in image classification

- Given a cell placement, classify it among four routability levels, c_0, c_1, c_2, c_3
- c_0 has the least #DRVs

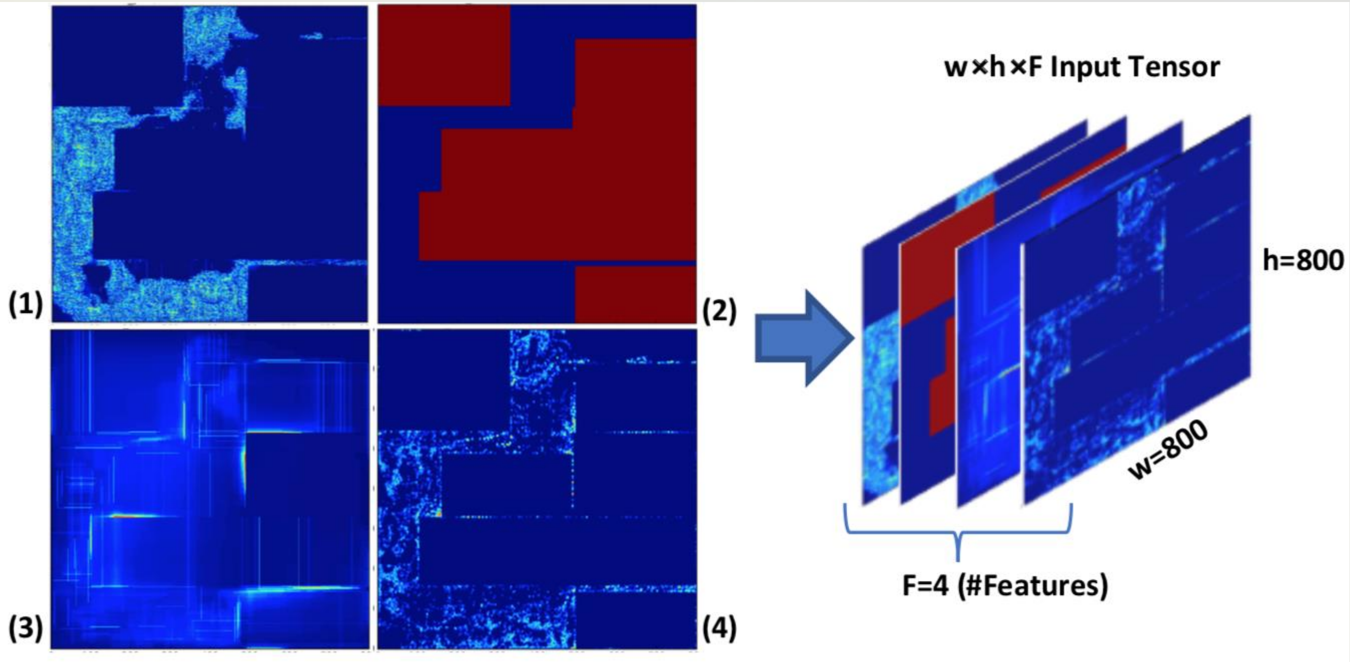
An Important Feature

- RUDY (Rectangular Uniform wire DensitY) (P. Spinder et al. DATE07)
- RUDY at a point is superposition of RUDYs of multiple nets

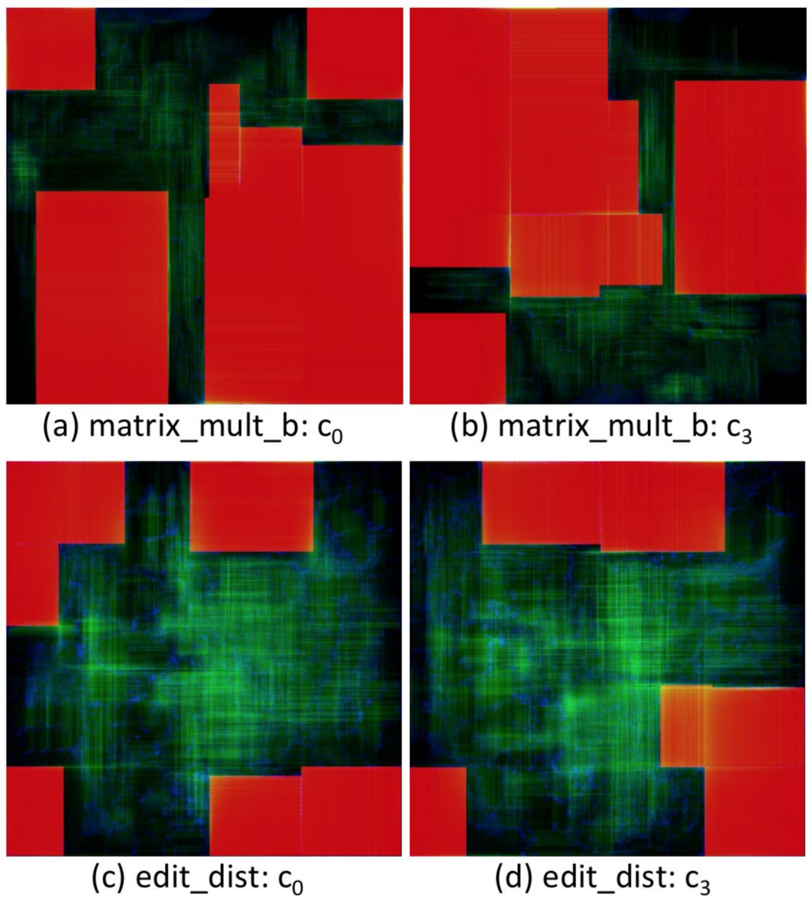


$$RUDY = \frac{w+h}{w \cdot h}$$

Feature Illustration

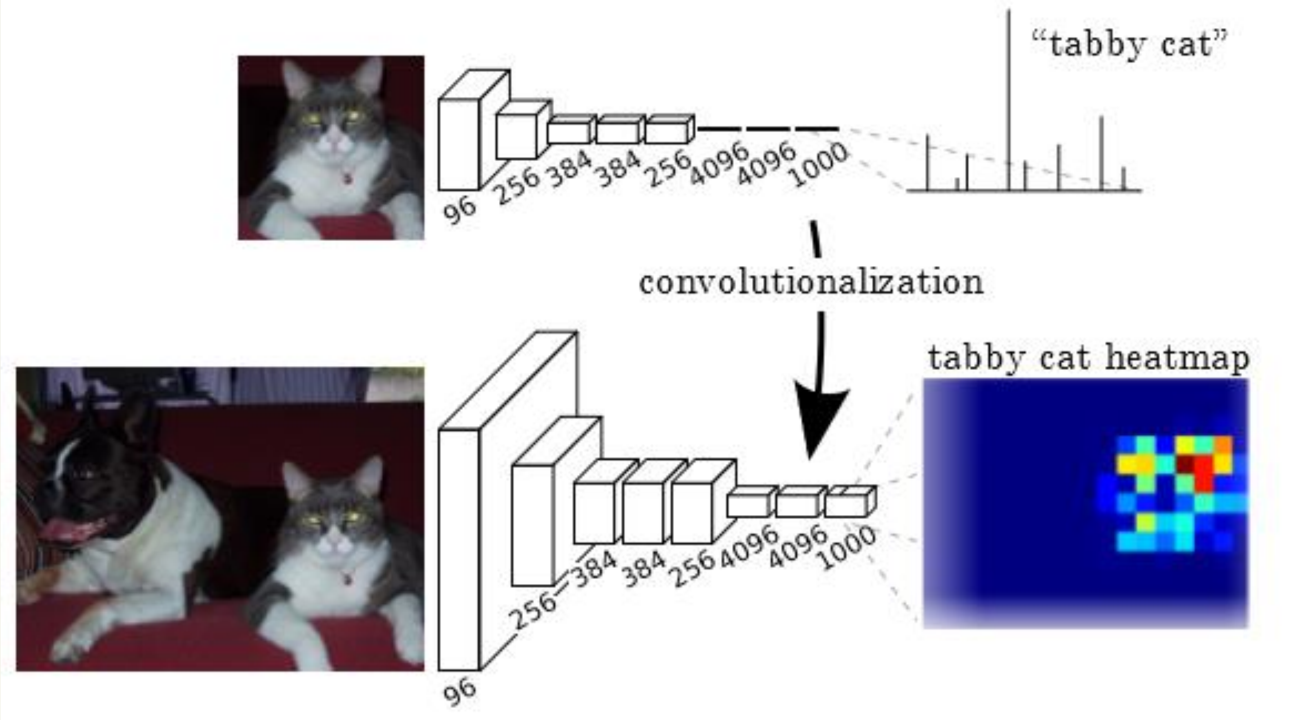


(1) Pin density, (2) macro,
(3) long-range RUDY, (4) RUDY pins



Red: macro region
Green: global long-range RUDY
Blue: global RUDY pins

Fully Convolutional Network (FCN) for Hotspot Detection



Semantic segmentation: FCN output is another image instead of classification

Image from Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2017. Fully Convolutional Networks for Semantic Segmentation. (TPAMI)

Experiment Setup

- Five designs from ISPD 2015 placement contest
- ~300 different placements by placing macros in different ways
- Placement, routing and DRC are done by Cadence tool
- When a circuit is tested, the model trained with the other circuits
- SVM and Logistic Regression (LR) methods for comparison

Circuit Name	#Macros	#Cells	#Nets	Width (μm)	#Placements
des_perf	4	108288	110283	900	600
edit_dist	6	127413	131134	800	300
fft	6	30625	32088	800	300
matrix_mult_a	5	149650	154284	1500	300
matrix_mult_b	7	146435	151614	1500	300

#DRV Prediction Fidelity

- How recognize placement with the least #DRV (c_0)
- The **best rank** among the 10 least #DRV solutions

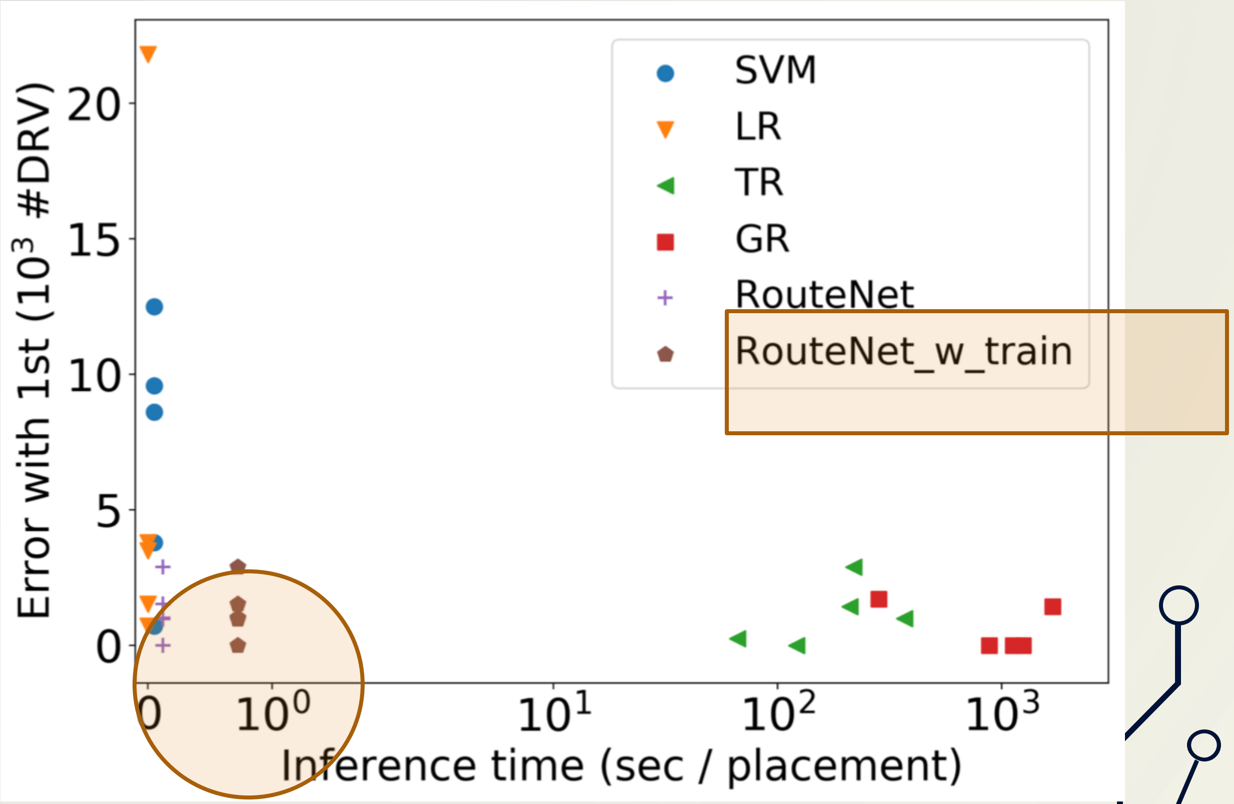
Circuit Name	c_0/c_1+c_2+c_3 accuracy (%)					Best rank in top 10				
	SVM	LR	TR	GR	Route Net	SVM	LR	TR	GR	Route Net
des_perf	63	74	80	77	80	87^{th}	15^{th}	2^{nd}	1^{st}	2^{nd}
edit_dist	69	68	78	77	76	17^{th}	17^{th}	3^{rd}	3^{rd}	2^{nd}
fft	66	62	73	70	75	6^{th}	6^{th}	2^{nd}	33^{rd}	1^{st}
matrix_mult_a	66	65	78	74	72	30^{th}	5^{th}	1^{st}	1^{st}	5^{th}
matrix_mult_b	63	62	76	73	76	22^{nd}	93^{rd}	4^{th}	1^{st}	4^{th}
Average	65	66	77	74	76	32^{nd}	27^{th}	2^{nd}	8^{th}	3^{rd}

Our method

TR: Trial Routing
GR: Global Routing

#DRV Prediction Error and Runtime

- Y: gap between the ‘best in 10’ and the actually 1st-ranked placement with least #DRV
- X: inference time
- w_train: plus training time



DRV Hotspot Detection Evaluation

- Same decision threshold is used for all designs
- Slight different FPR, but all under 1%
- RouteNet is superior to all methods and improves global routing accuracy by 50%

Circuit Name	FPR (%)	TPR (%) TR	GR	LR	SVM	RouteNet
des_perf	0.54	17	56	54	42	74
edit_dist	1.00	25	36	38	28	64
fft	0.30	21	45	54	31	71
matrix_mult_a	0.21	13	30	34	12	49
matrix_mult_b	0.24	13	37	41	20	53
Average	0.46	18	41	44	27	62

		Prediction Result: Positive	Negative
Label	Positive	TP	FN
	Negative	FP	TN

Evaluation

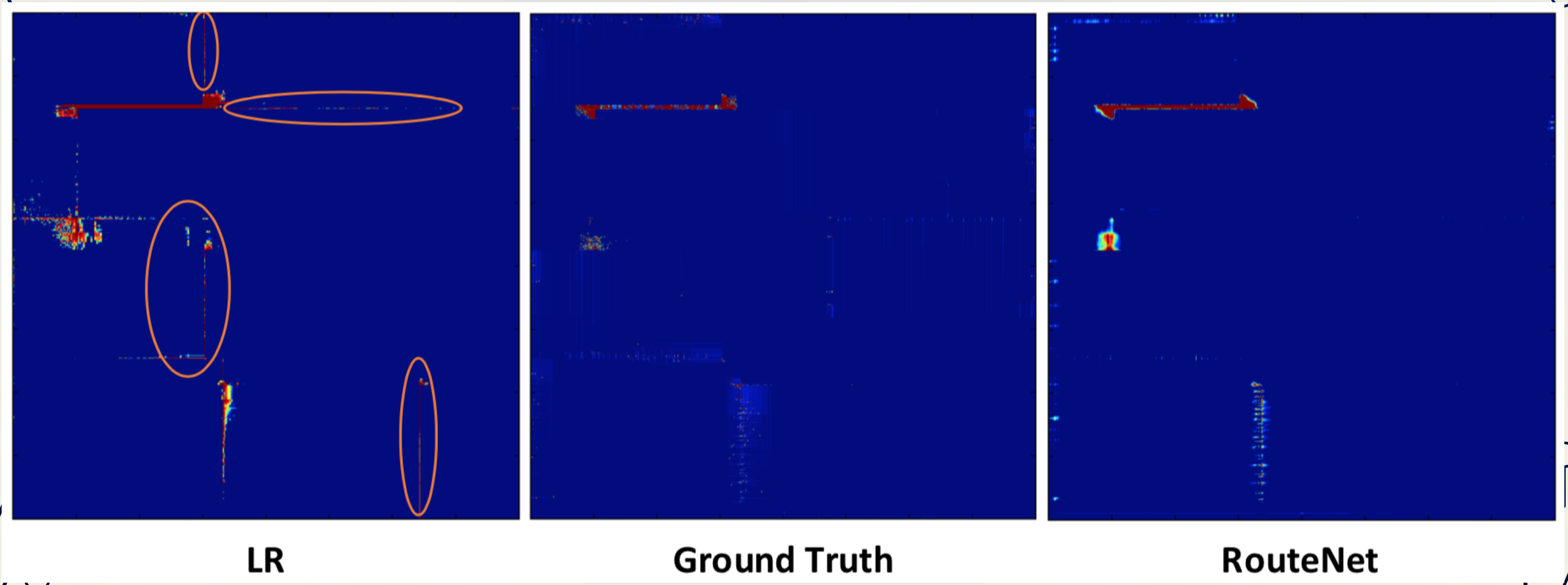
$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

TPR (True Positive Rate)

FPR (False Positive Rate)

DRC Hotspot Detection Demonstration



Conclusions and Outlook

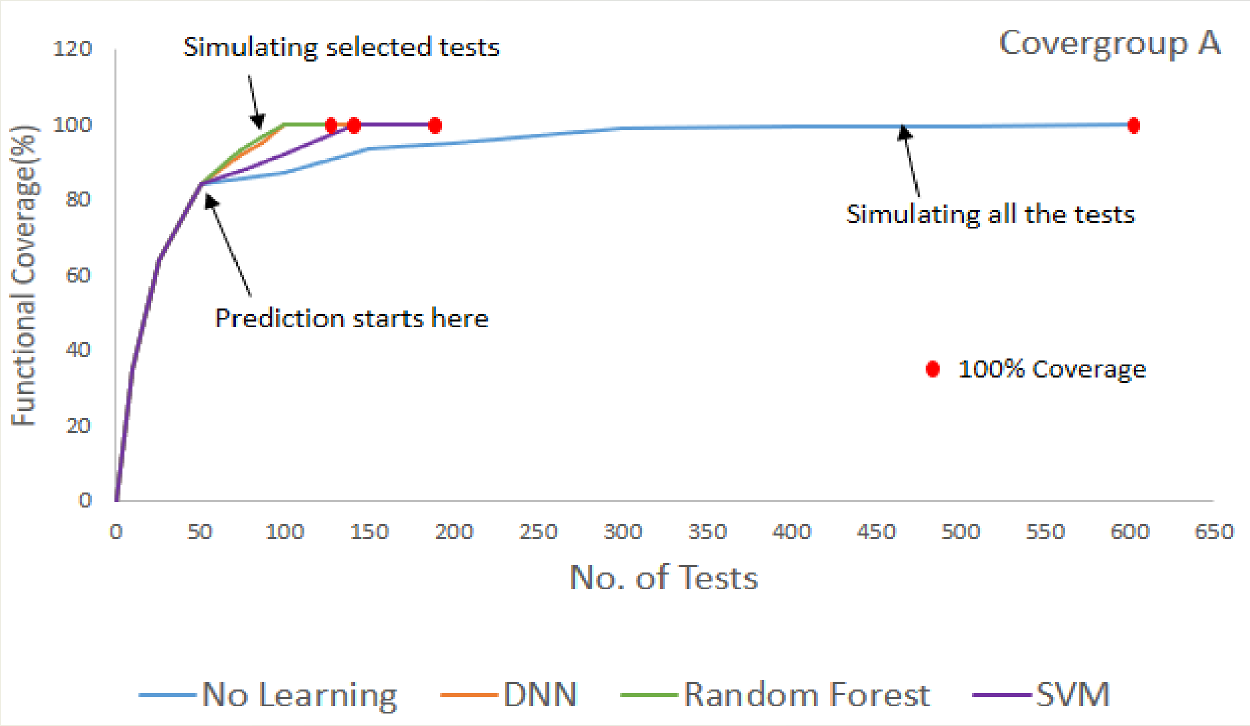
- Machine learning provides huge potential for improving existing chip design and verification
- Now is the beginning of new era
- Collaboration between academia and industry is more needed than ever

Thank You!

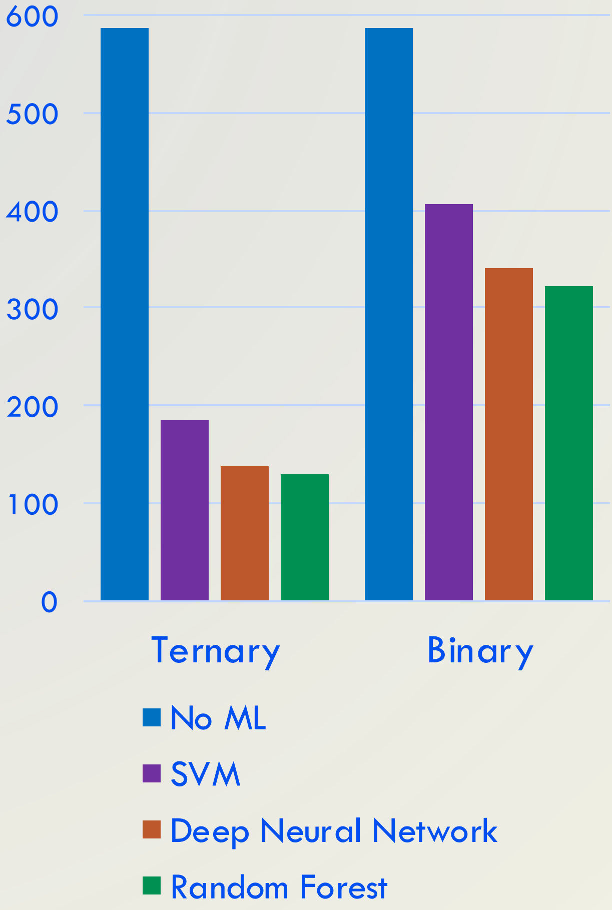
Questions?

Test-Level Results: Group A

Covergroup A: coverage metrics correlate with test knobs



simulated tests



Test-Level Results: Group B

Covergroup B: coverage metrics do not correlate with test knobs