Machine Learning-Guided Stimulus Generation for Functional Verification

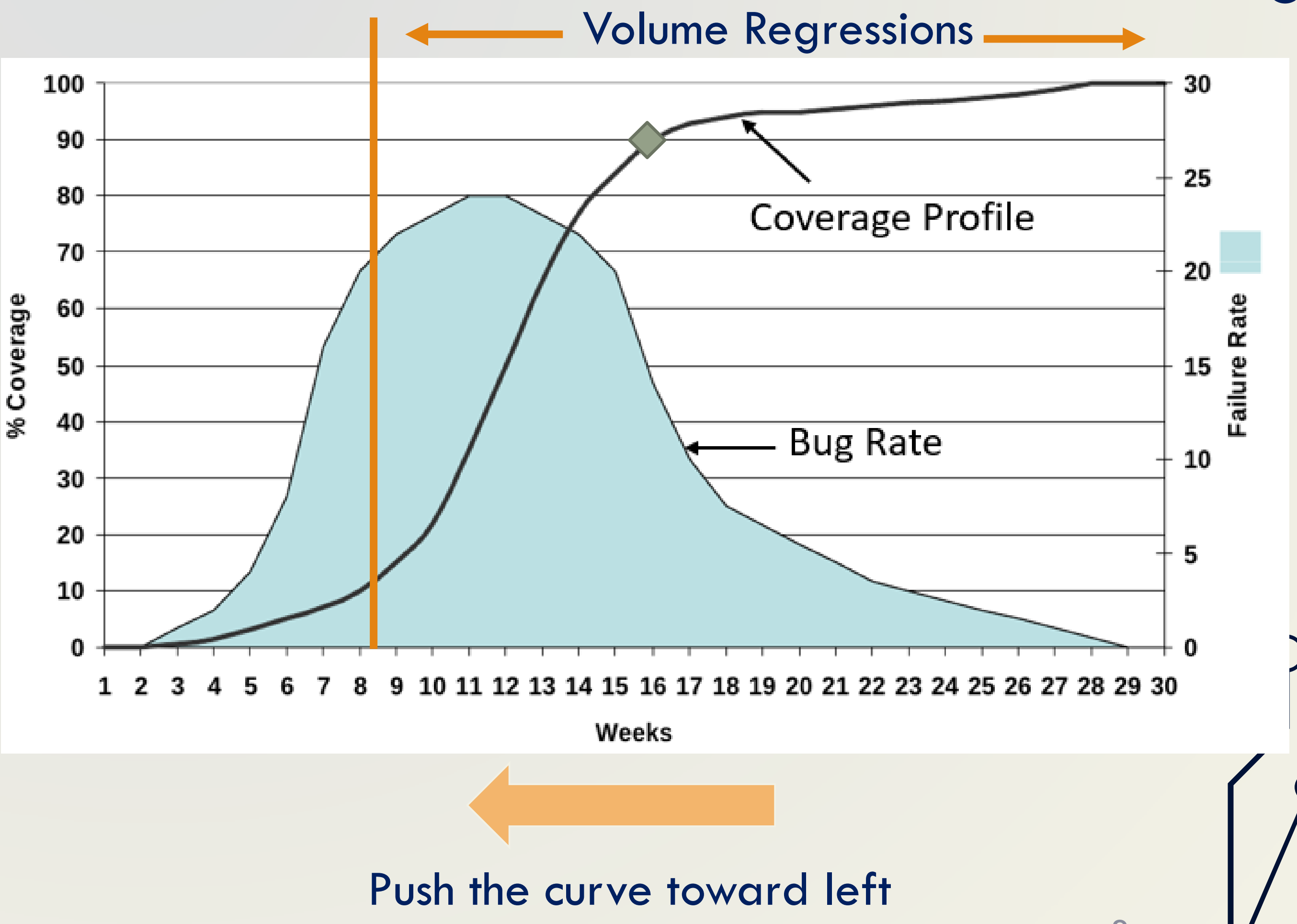
S. Gogri, J. Hu, A. Tyagi, M. Quinn
S. Ramachandran, F. Batool, A. Jagadeesh
Texas A&M University

Outline

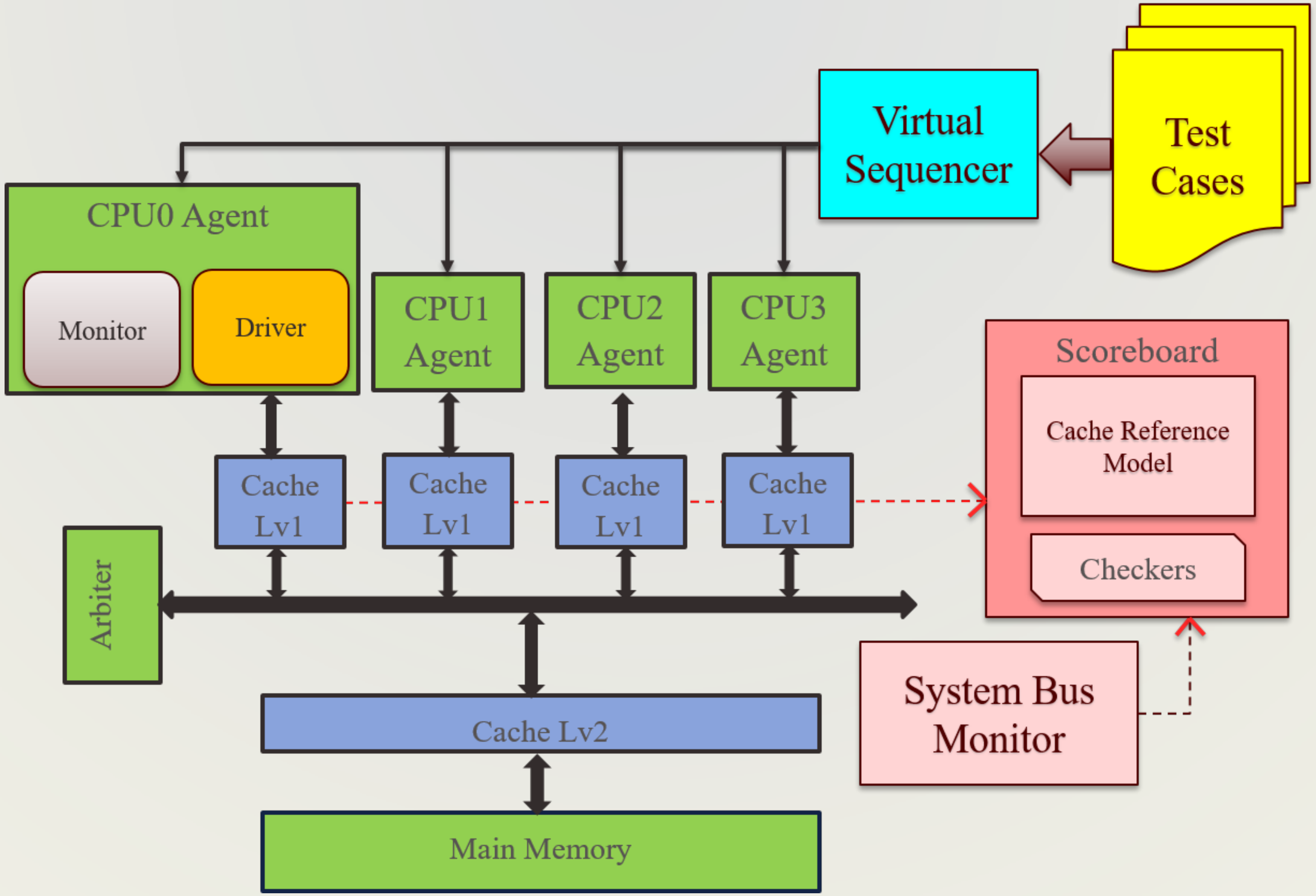
- Challenge of functional verification
- Background
- Previous work
- Machine learning guided stimulus generation
  - Coarse-grained test-level pruning and results
  - Fine-grained transaction-level optimization and results
- Conclusions

Simulation-Based Functional Verification

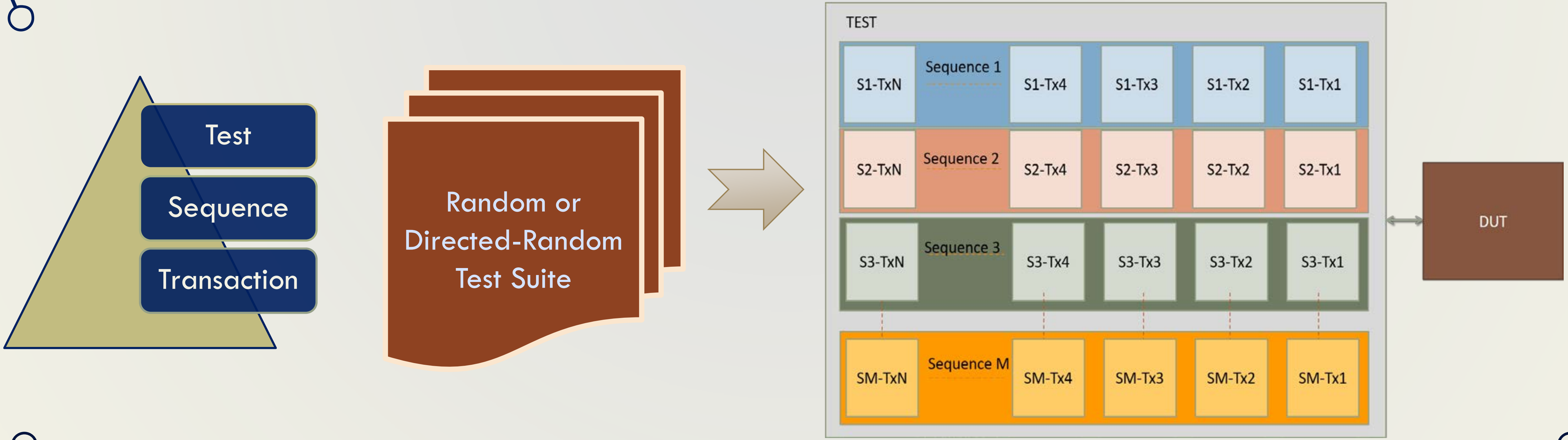
- Functional verification mostly via simulations
- Exercise stimulus and observe response
- Huge space
- Verification time and manpower > design



UVM – Universal Verification Methodology

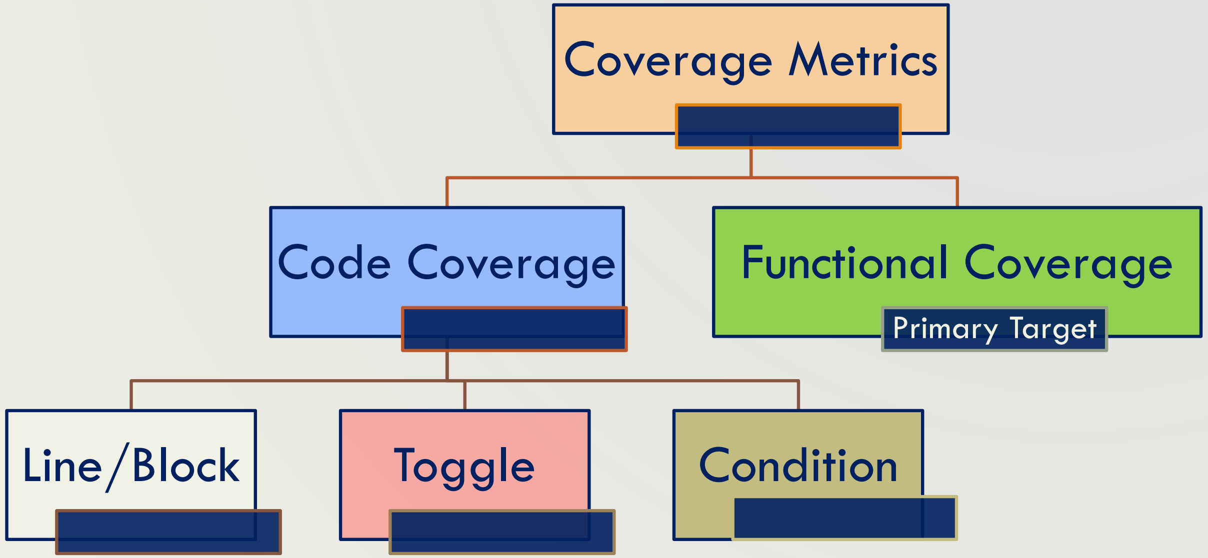


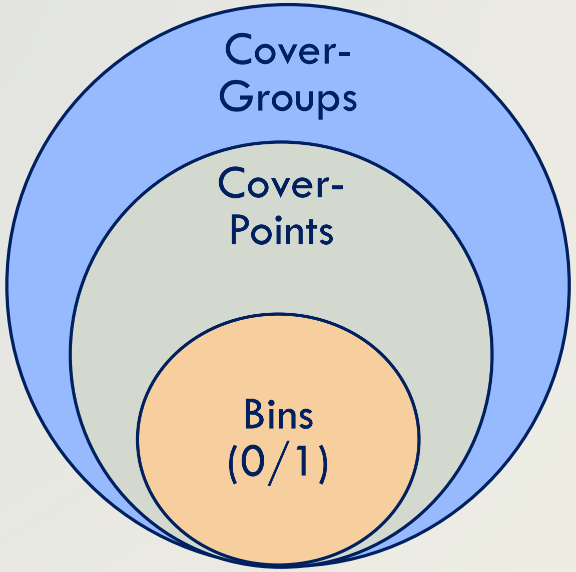
Stimulus



Constraints applied at test-level are called test-knobs

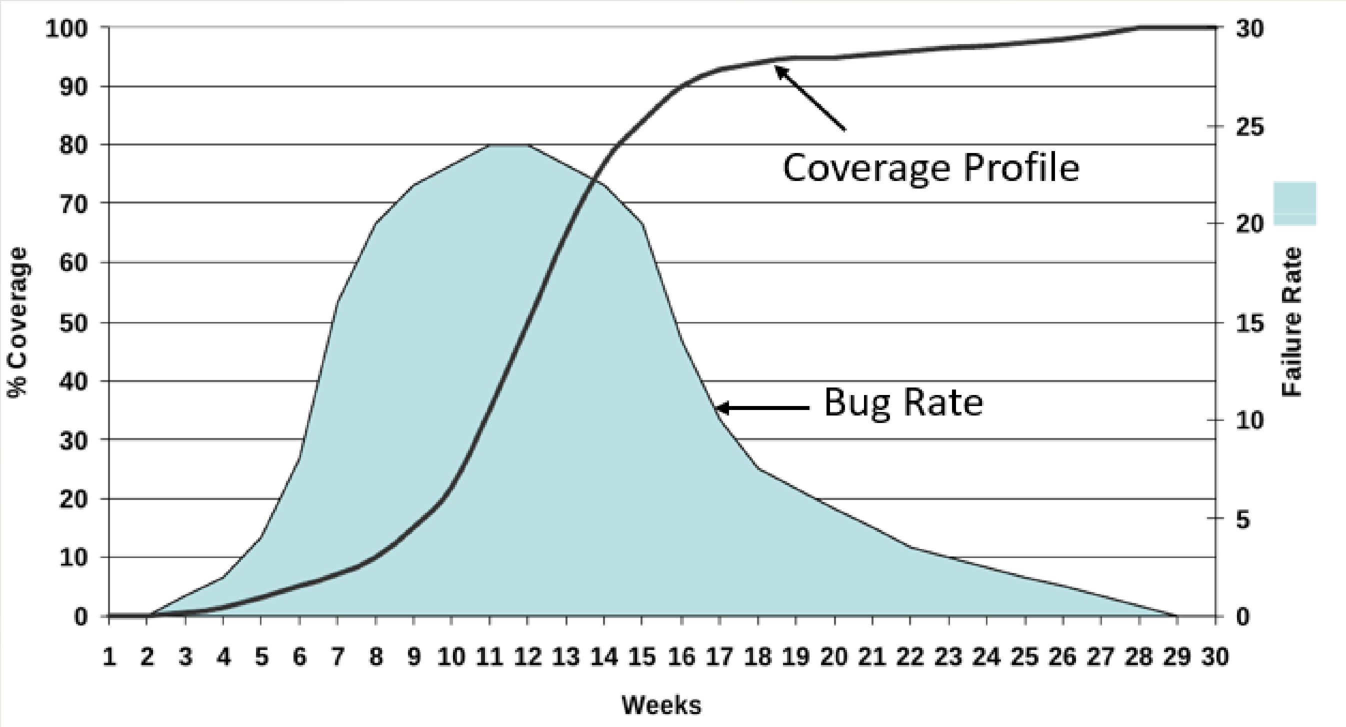
Coverage


Coverage Metrics
Code Coverage
Functional Coverage
Primary Target
Line/Block
Toggle
Condition


Cover-
Groups
Cover-
Points
Bins
(0/1)

Machine Learning for Fast Coverage

- A machine learning model
- Tells if a stimulus ψ will cover an unverified point
- Simulate ψ only if the answer is yes



Prior Art

- ML for functional verification was started 16 years ago
- Earlier than the recent booming of deep learning
- Mostly tried a model and then showed verification time reduction

Coverage Directed Test Generation for Functional Verification using Bayesian Networks

Shai Fine Avi Ziv

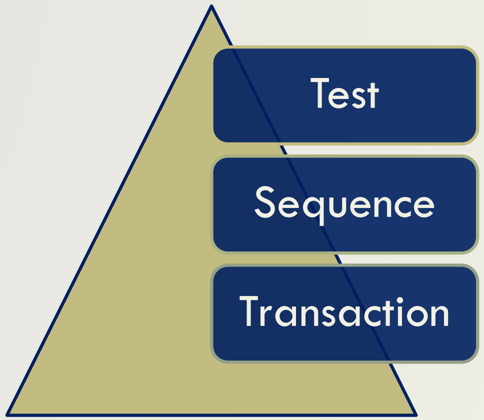
IBM Research Laboratory in Haifa
Haifa, 31905, Israel
{fshai, aziv}@il.ibm.com

Some Previous Works

- Fine and Ziv, DAC 03, Bayesian network
- Guzeya, et al, TCAD 10, SVM
- Ioannides and Eder, TODAES 12, “Coverage-directed test generation automated by machine learning”
- Chen, Wang, Bhadra and Abadir, DAC 13, knowledge reuse
- Sokorac, DVCON17, genetic algorithm for toggle coverage
- Wang, et al, Great Lake Symp. VLSI 18, neural network

Are Existing Techniques Adequate?

- Mostly based on old ML engines
- No study on the granularity of ML application
  - Coarse-grained test level stimulus optimization
  - Fine-grained transaction level stimulus optimization



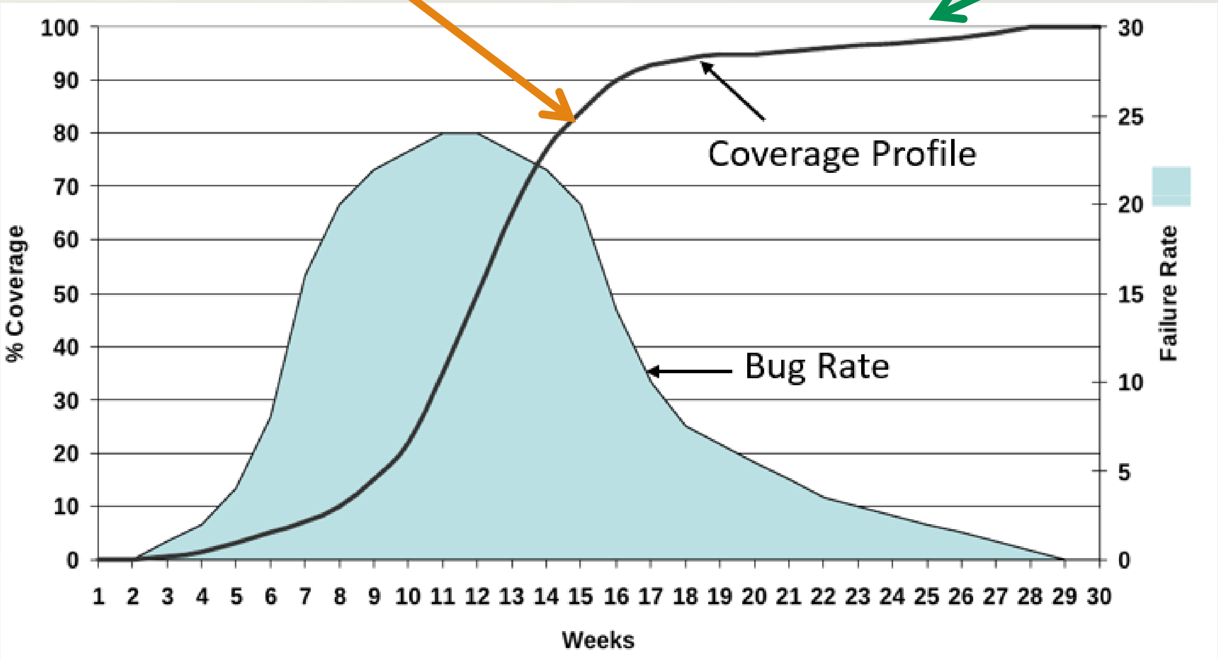
- Stimulus pruning? or constructive stimulus generation?
- No differentiation between Finite State Machine (FSM) and non-FSM design

Test-Level Stimulus Pruning

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

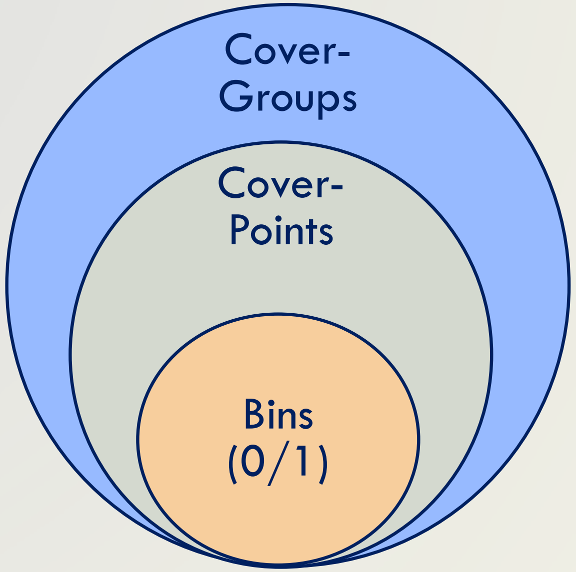
Transition decided by online validation

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



ML Model Setup

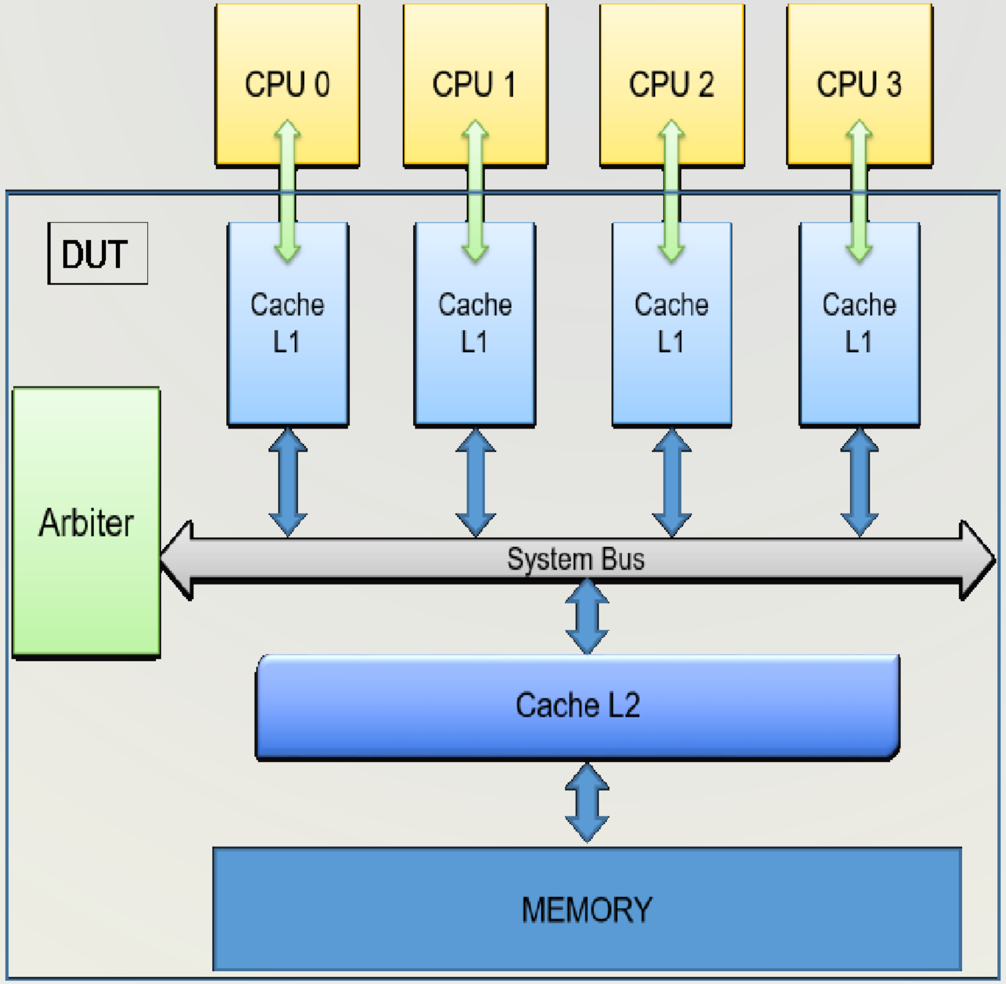
- One ML model for each cover point
- For each model
  - One binary output for each cover bin
  - 1: the bin will be hit by a test
  - 0: the bin will not be hit by a test
- A test is simulated if it will hit any uncover bin



ML Model Features

Features

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

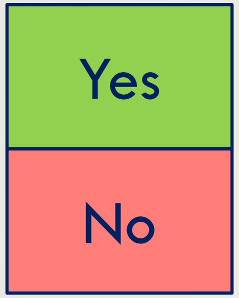


Cover points

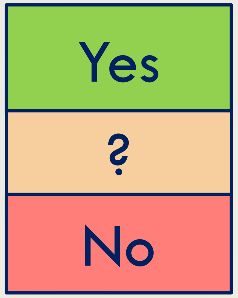
- Address X req type in bins
- Snoop request
- $protocol transitions
- $hit on each address
- …

Ternary Classification

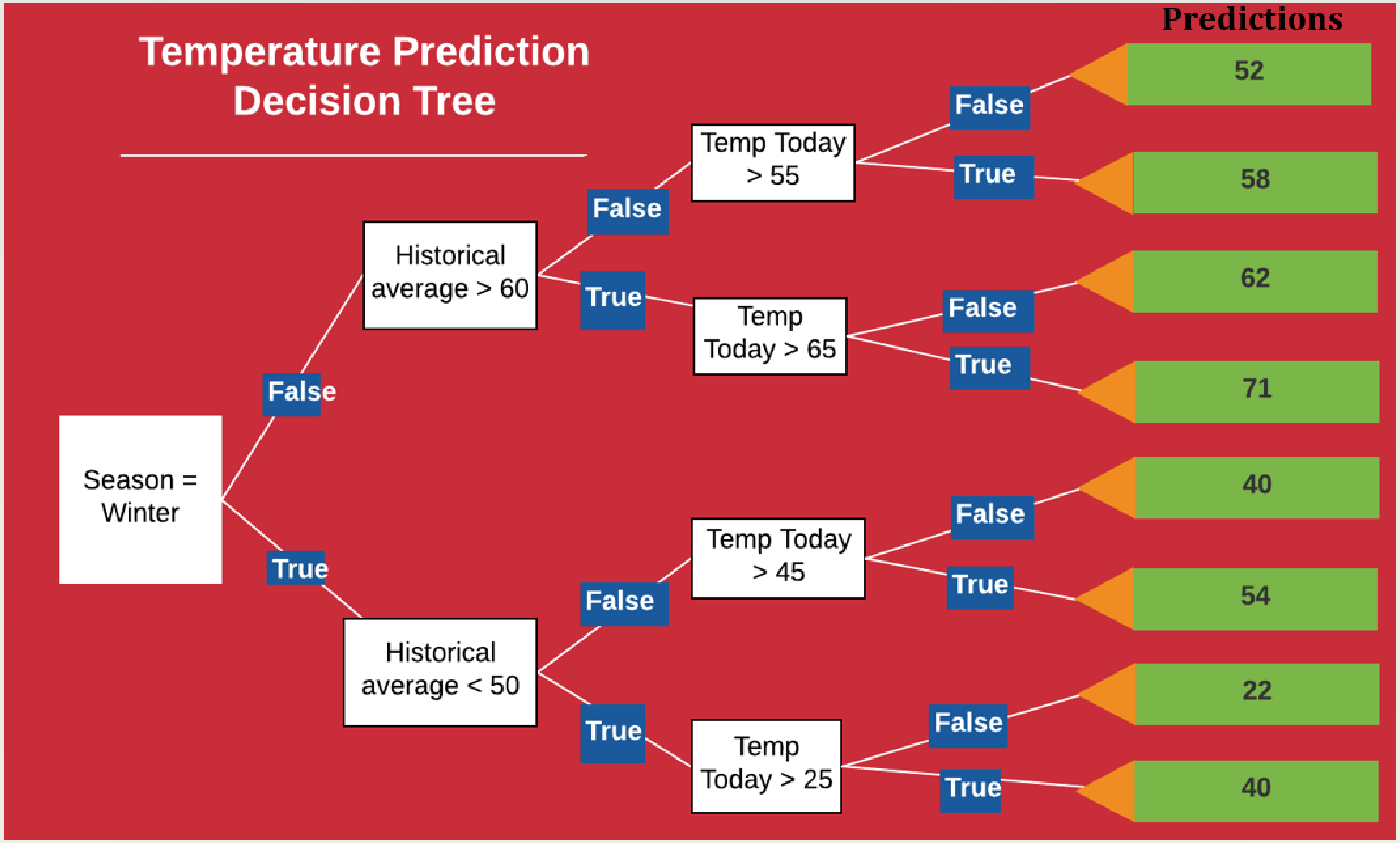
- Will a test improve verification coverage?
- Conventionally: binary classification – yes or no



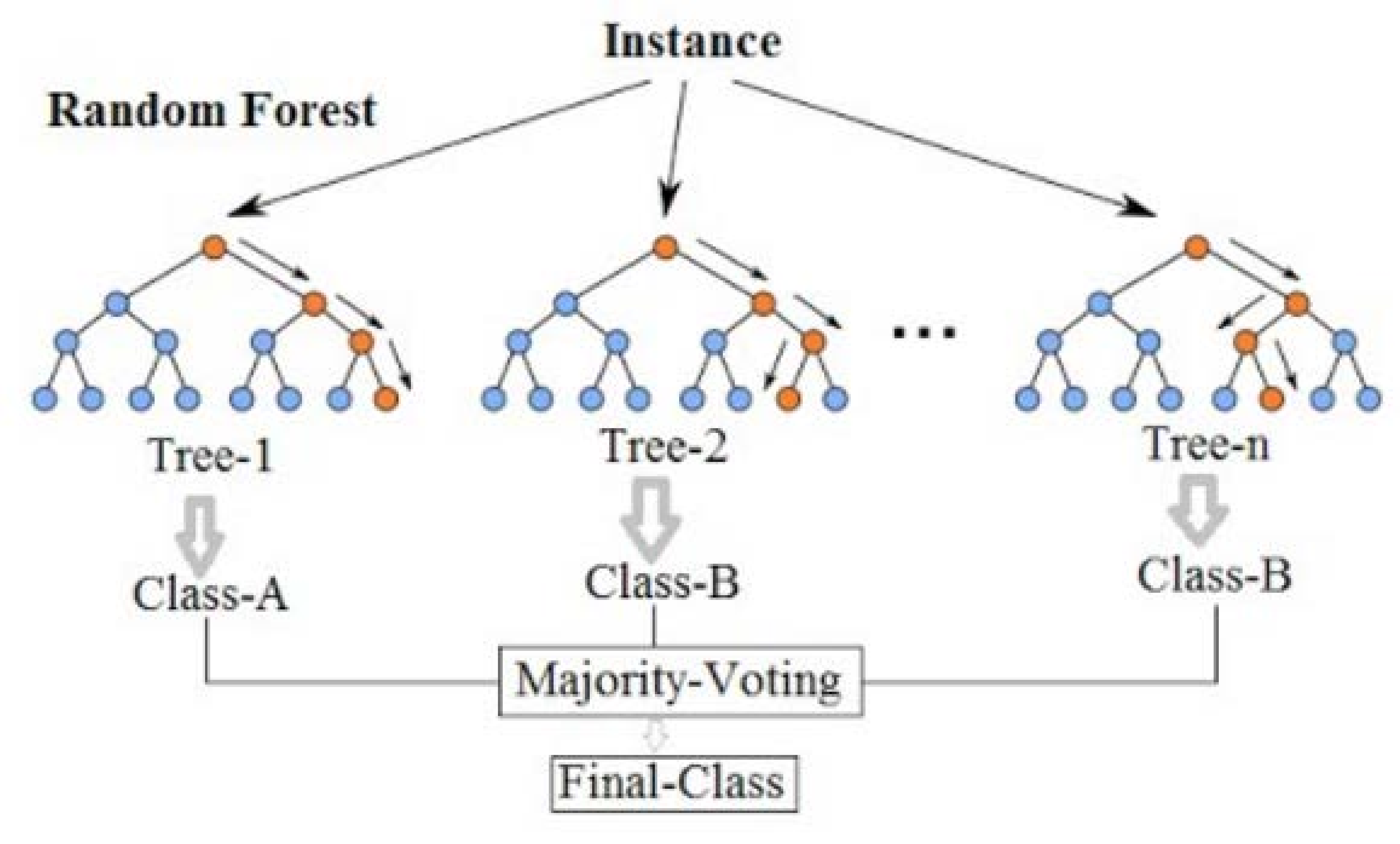
- Our approach:
  - Probability p of improving coverage by test ψ
  - If p is high, simulate ψ
  - If p is low, do not simulate ψ
  - If p is in middle, simulate ψ and use the result to train ML model



Decision Tree Classification

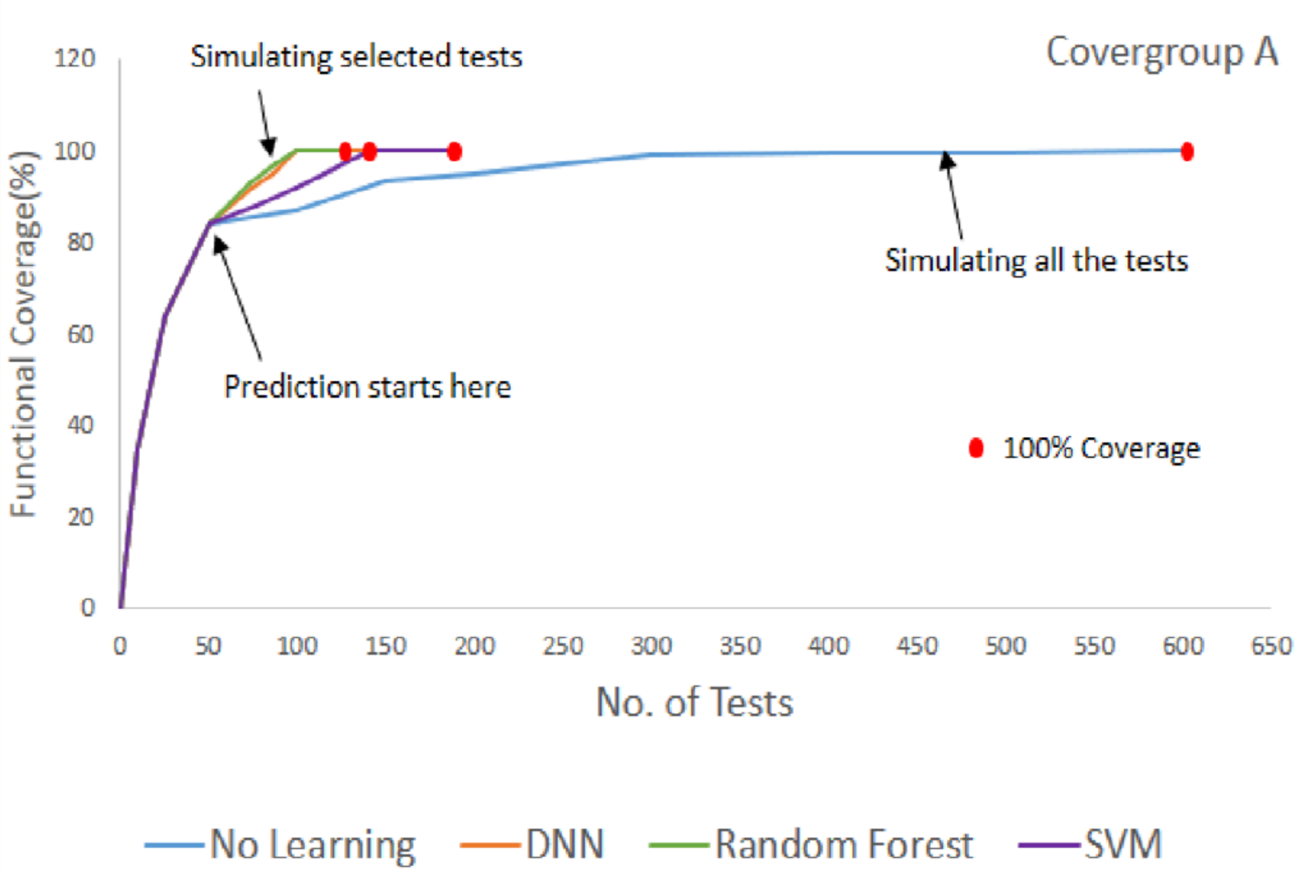


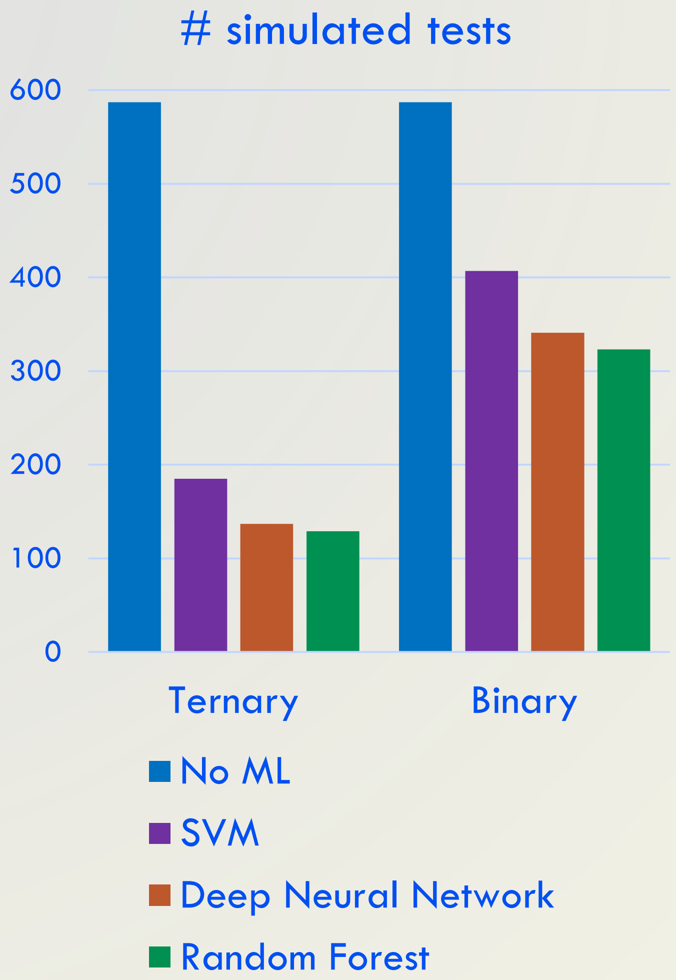
Random Forest Classifier



Test-Level Results: Group A

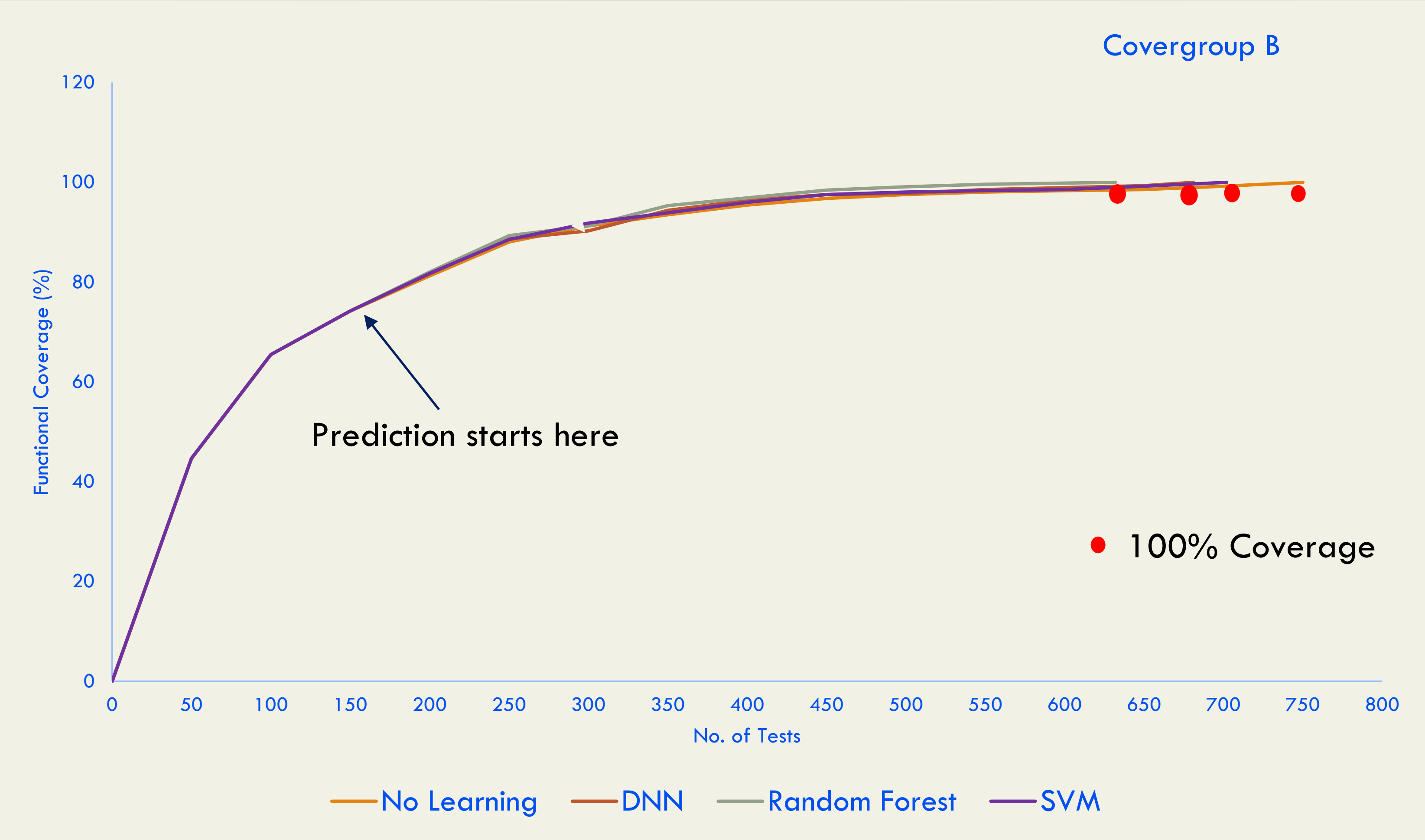
Covergroup A: coverage metrics correlate with test knobs





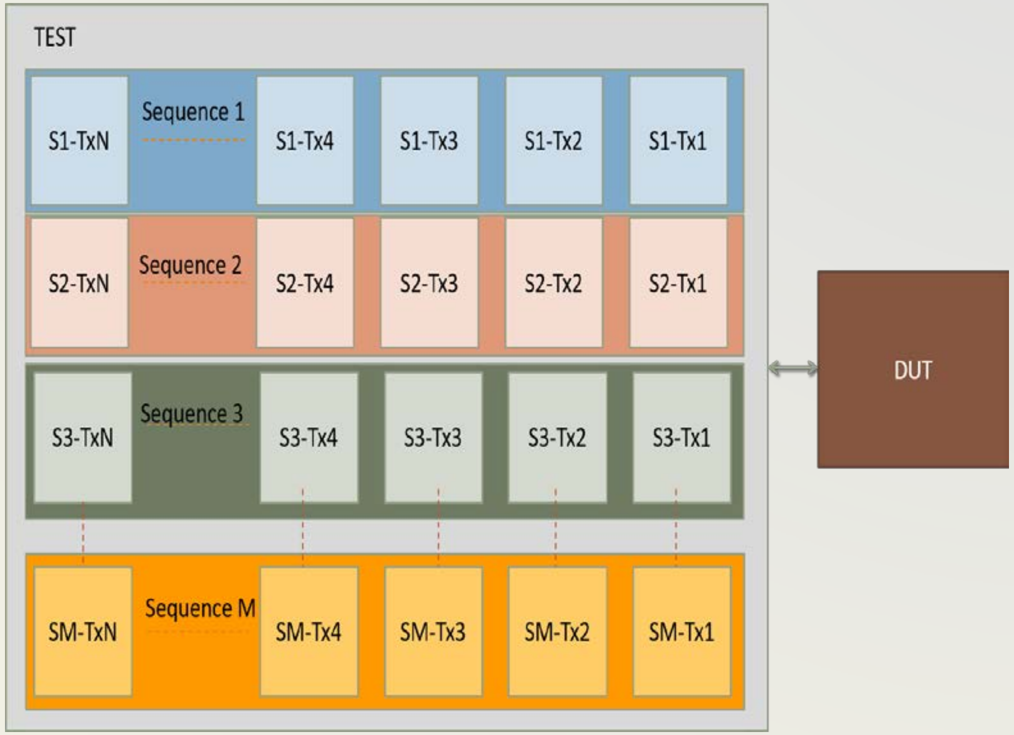
Test-Level Results: Group B

Covergroup B: coverage metrics do not correlate with test knobs



Transaction-Level Stimulus Optimization

Finer-grained control than test-level pruning

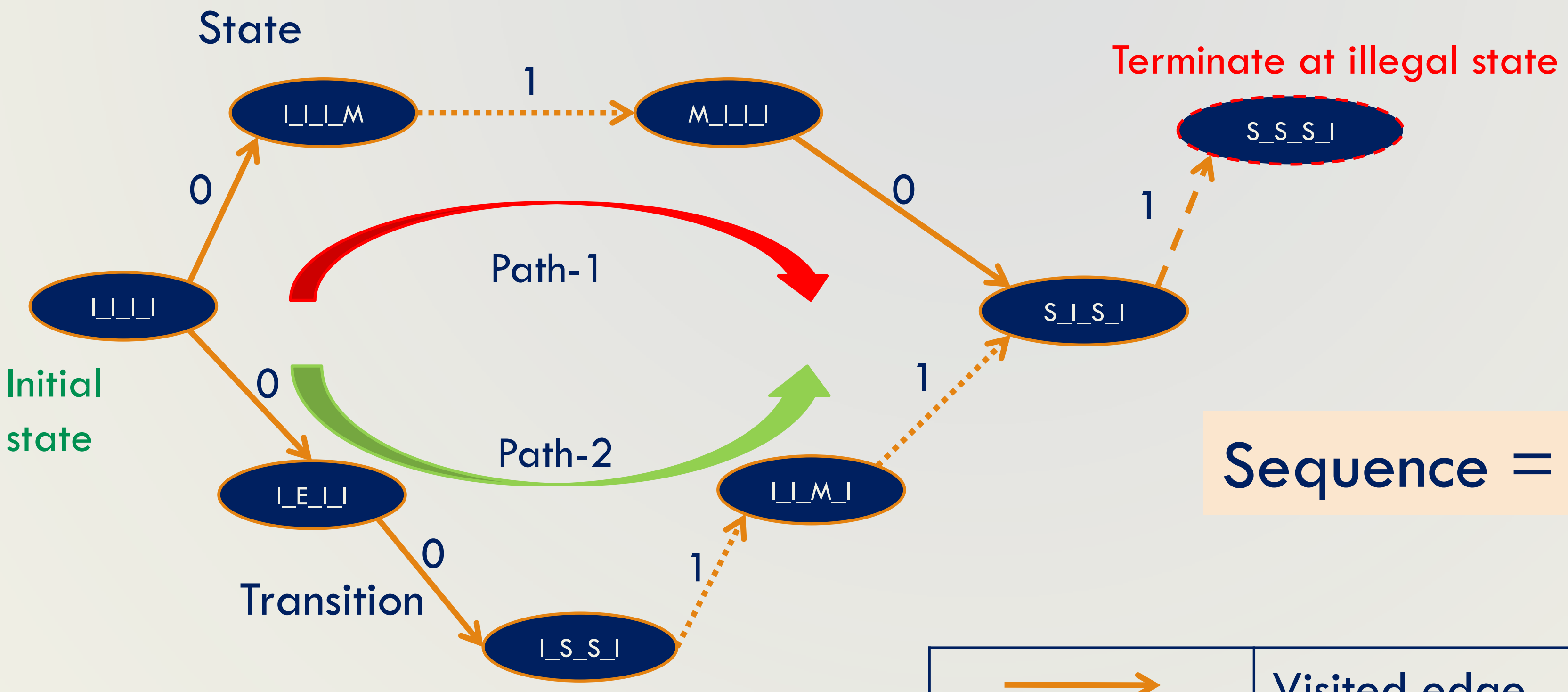


	Finite State Machine	Non-FSM
Online transaction pruning		
Offline sequence generation		

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 = longest path

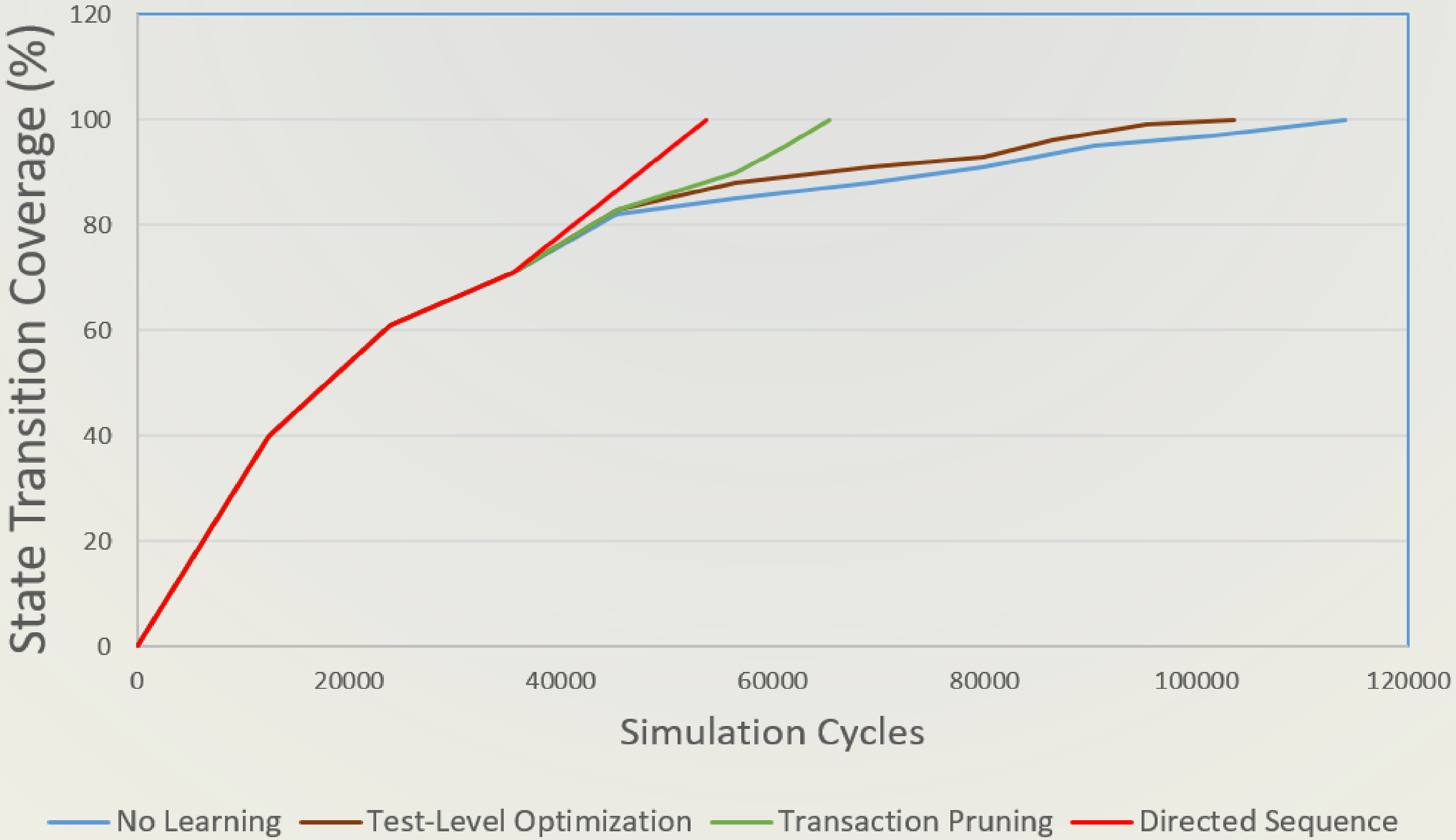
→ (solid)	Visited edge
→ (dotted)	Unvisited and predicted edge
→ (dashed)	Unvisited and illegal prediction

Online Pruning vs. Offline Sequence Generation

- Online transaction pruning
  - Myopic scope at each pruning
- Offline sequence generation
  - Much longer horizon in scope

FSM Transaction Optimization Results

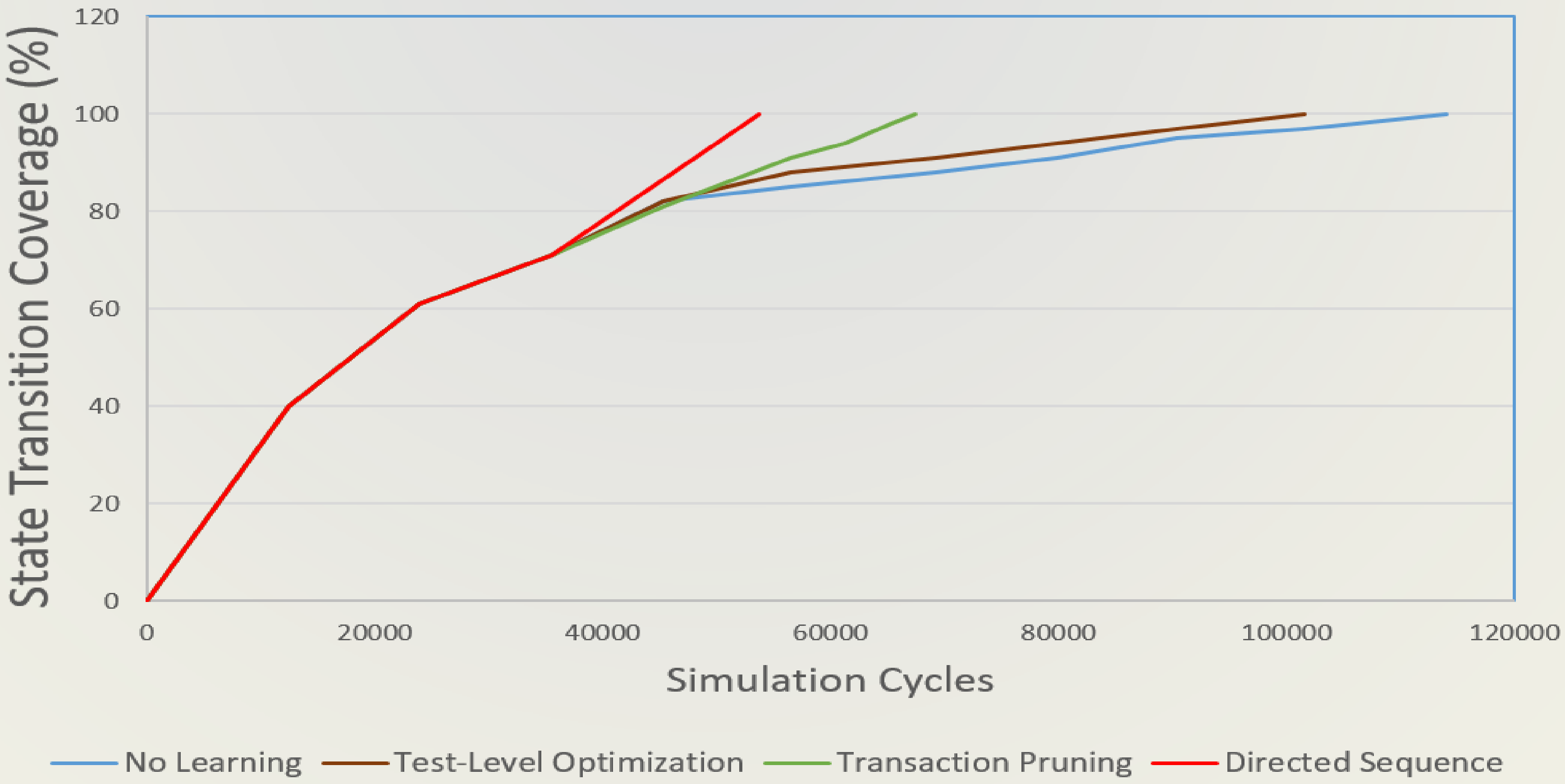
Coverage Metric: MESI state transitions – 143 bins



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

FSM Transaction Optimization Results

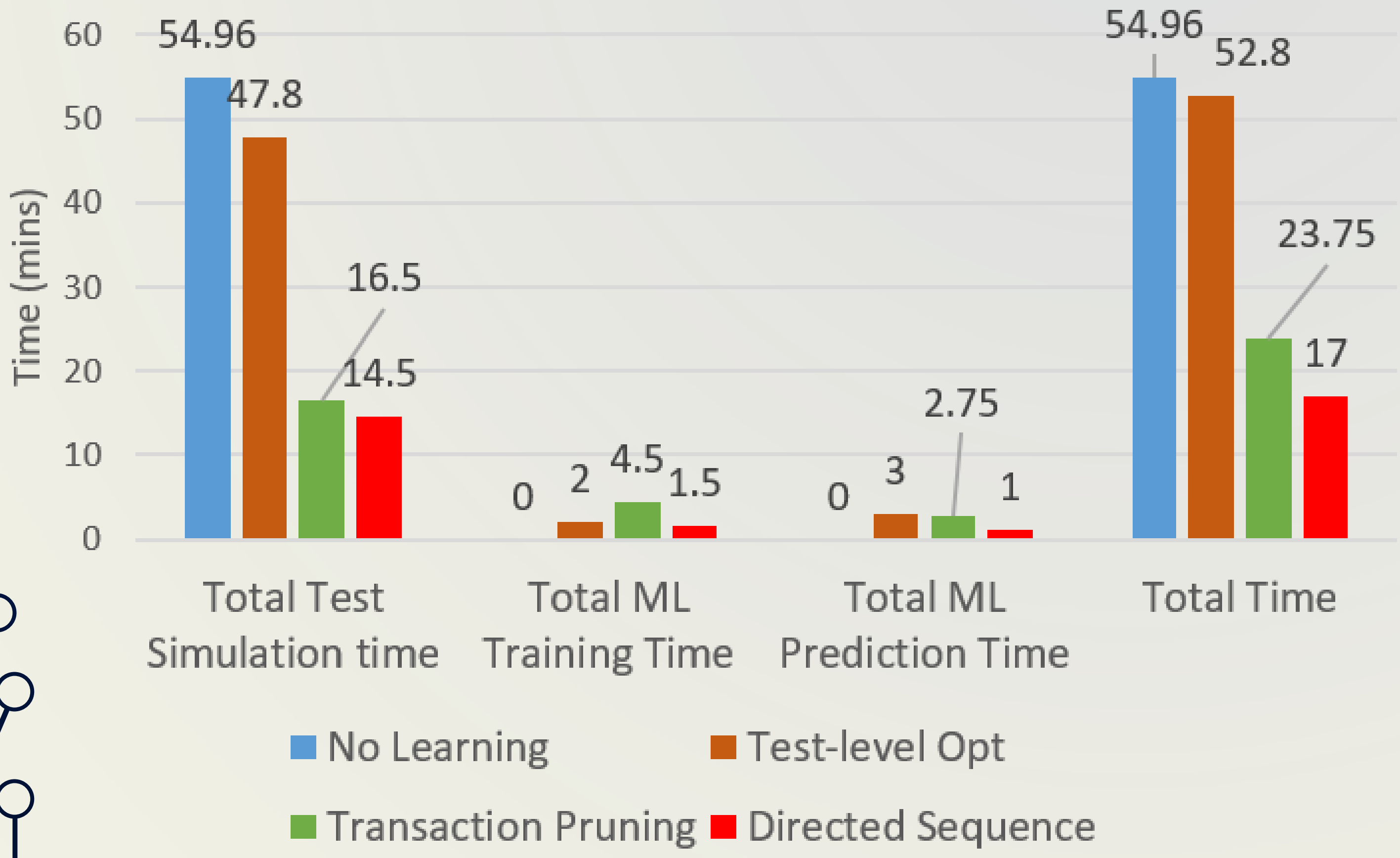
Coverage Metric: MESI state transitions – 143 bins



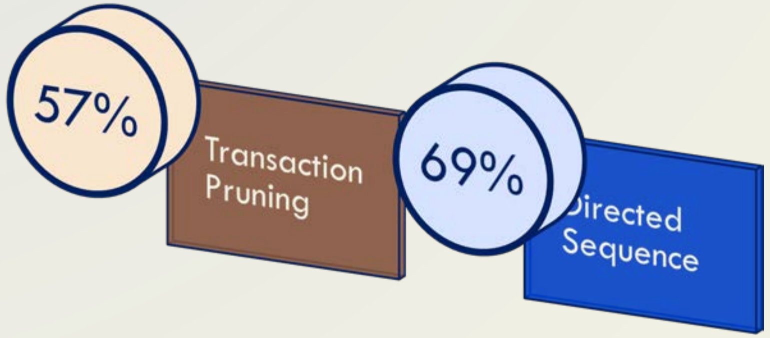
<u>Random Forest Classifier (RF) </u>55% reduction in simulation cycles

FSM Verification Time

ML engine: random forest



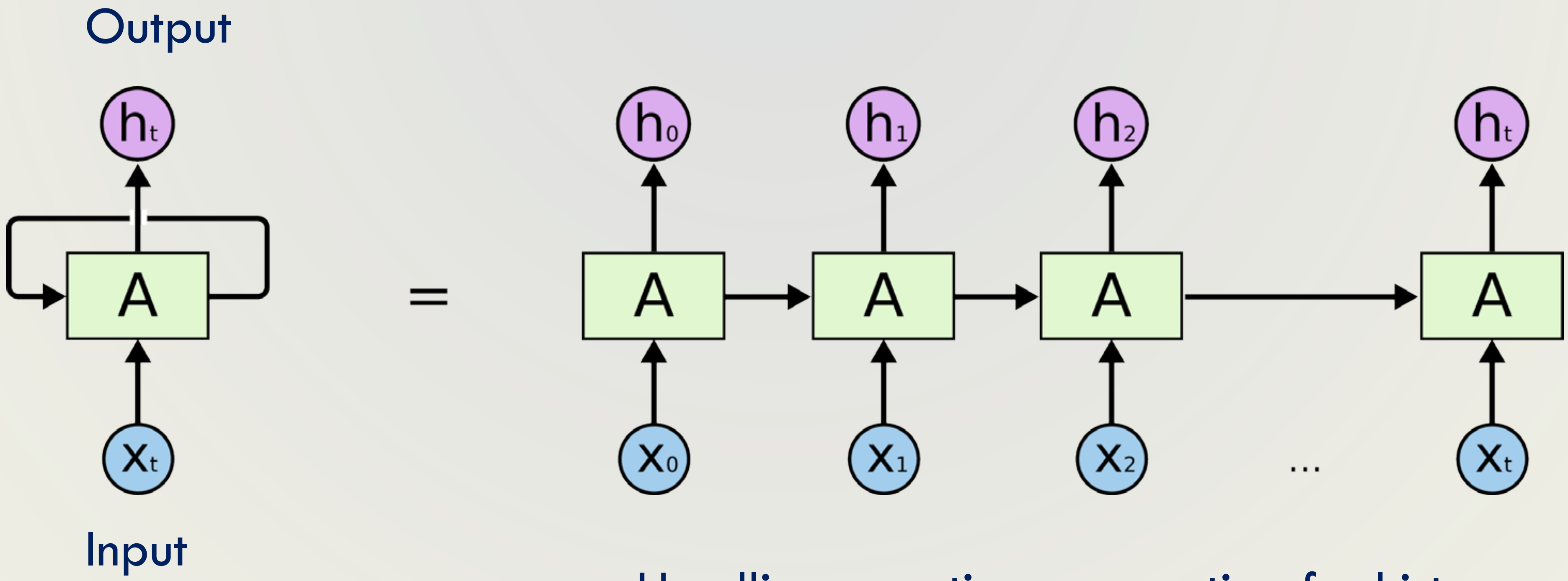
Verification time reduction



Non-FSM Event Coverage

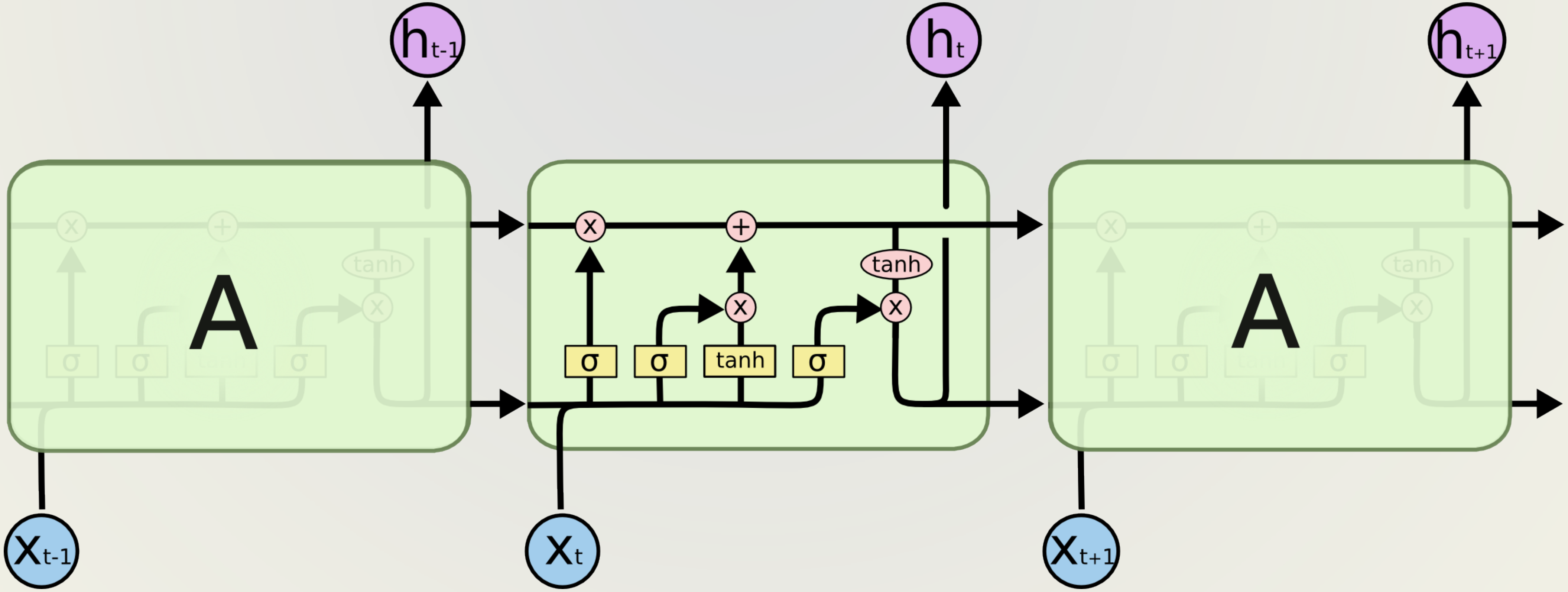
- Events: buffer full, cache hit, etc.
- Almost impossible to deterministically cover events through test-level optimization
- Event coverage depends on transaction history

Recurrent Neural Network (RNN)



Unrolling over time, accounting for history

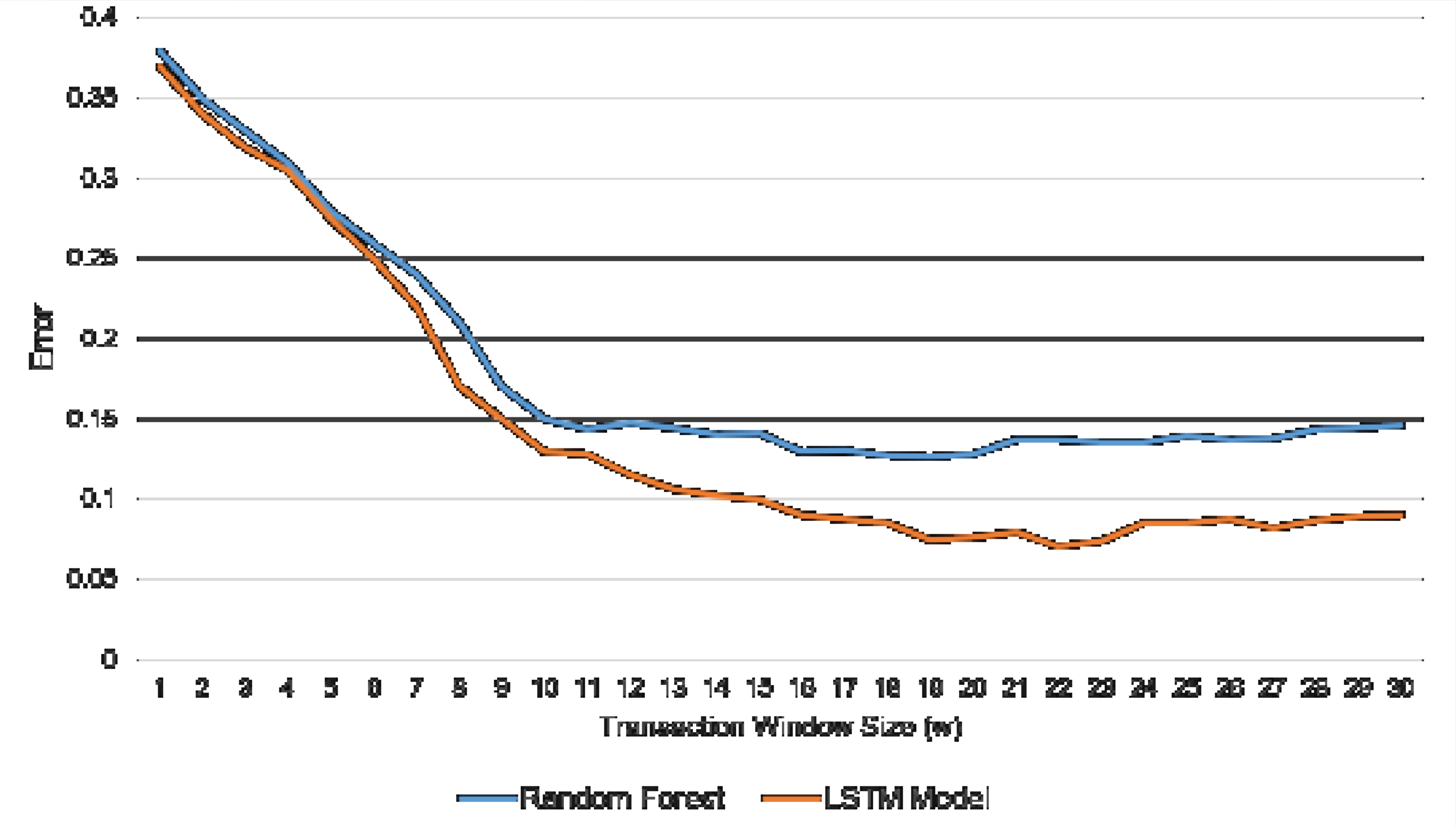
Long Short-Term Memory (LSTM)



σ: gate function, allowing a signal to pass or not

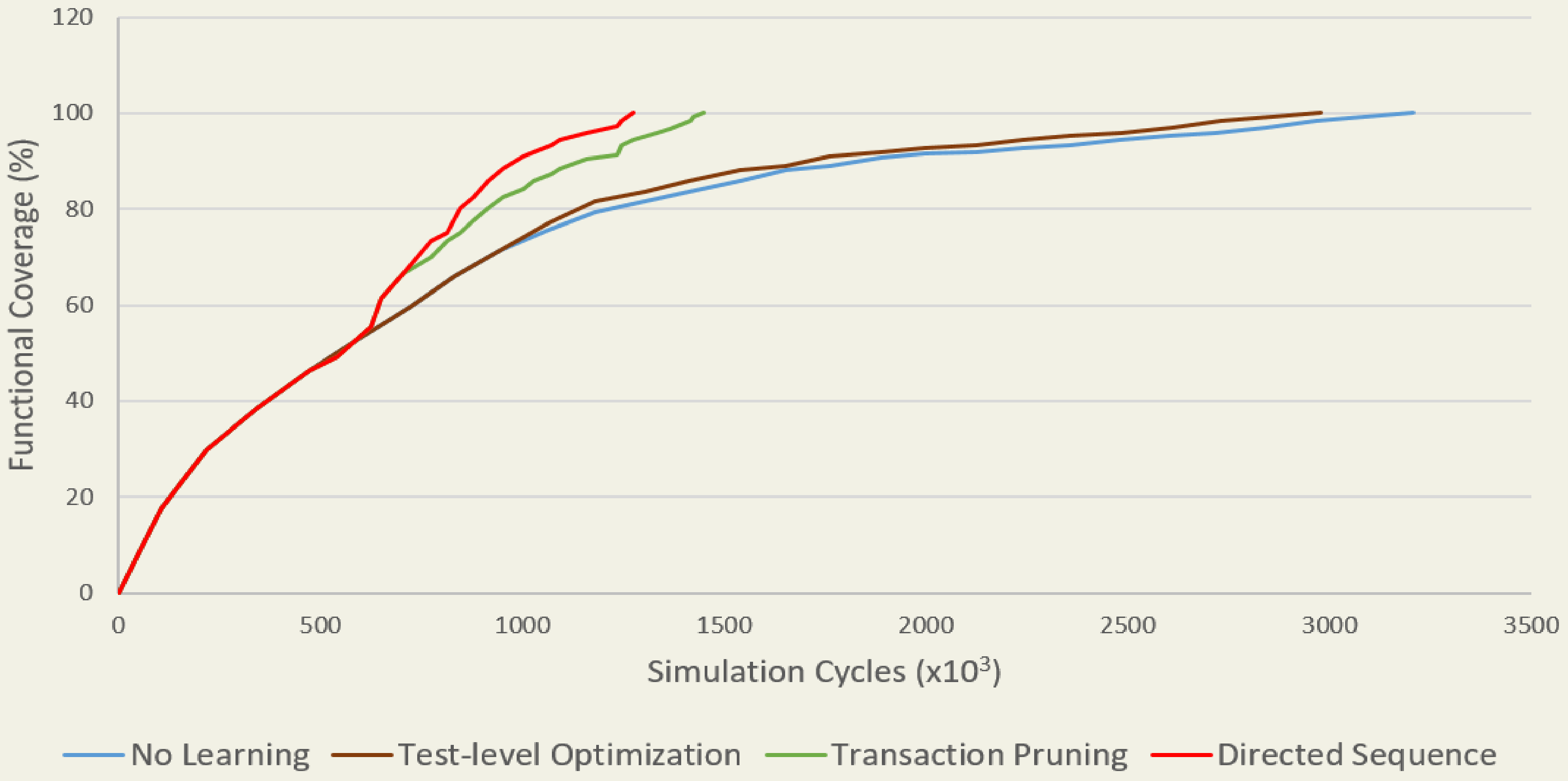
LSTM applications: time series, natural language processing

History Effect



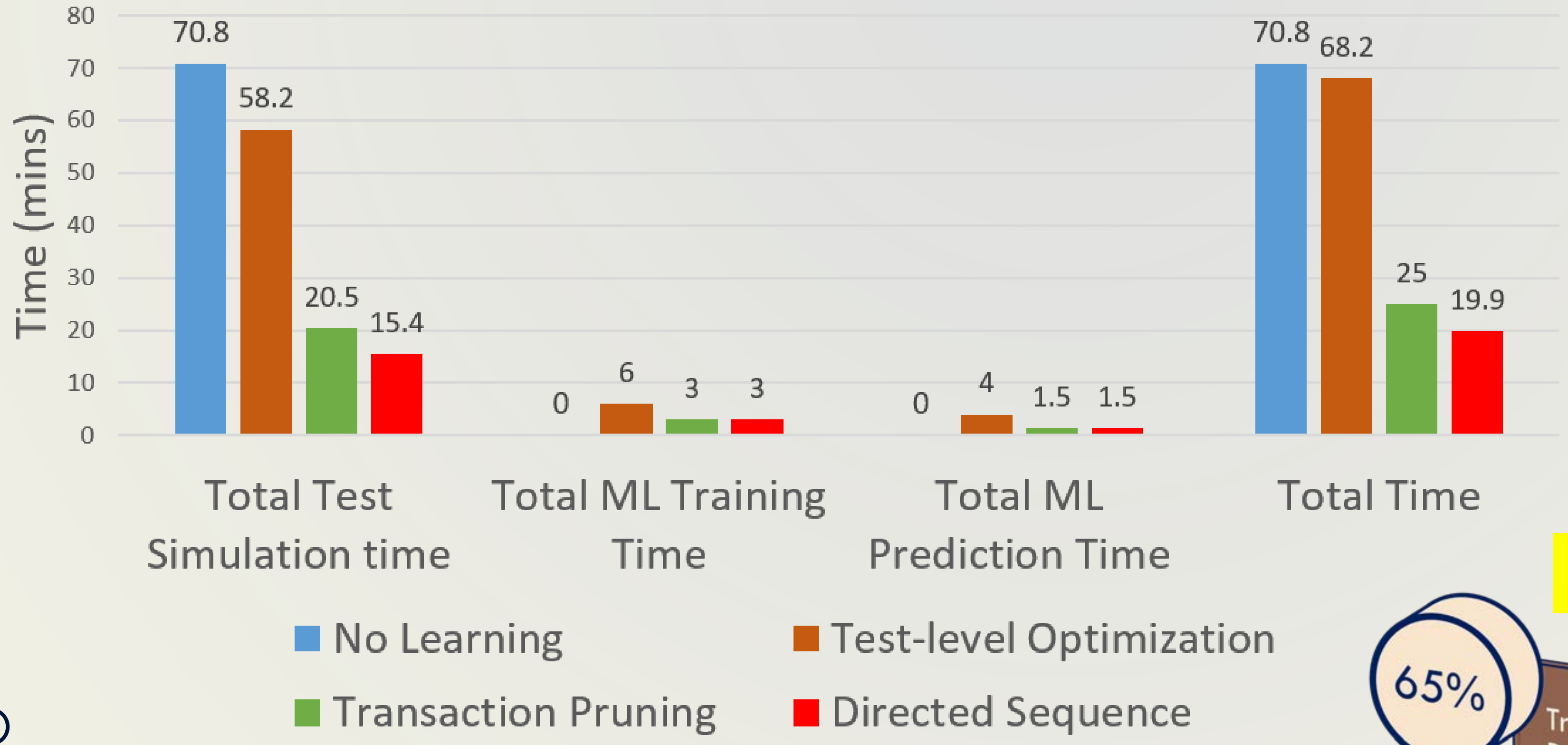
Non-FSM Event Coverage Results

Coverage Metric: cache hit on every address – 768 bins

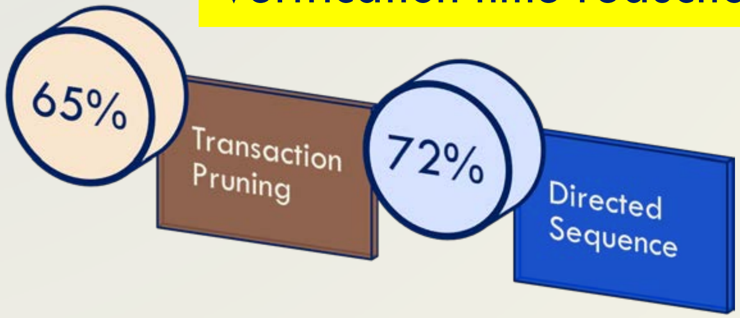


Long Short-Term Memory (LSTM) 61% reduction in simulation cycles

Non-FSM Verification Time



Verification time reduction



Conclusions

- Machine learning-based stimulus optimization for functional verification
- Fine-grained transaction level optimization outperforms coarse-grained test level pruning
- Offline sequence generation is superior to online stimulus pruning
- Random forest and LSTM are helpful
- Around 70% simulation time reduction

Future Research

- Small testcases
- Will work on big cases
- Colleagues with decades of industrial verification experience
- Seek industrial collaboration

Aakash Tyagi

Mike Quinn

Thank You!

Questions?