Learning to Explore Paths for Symbolic Execution

Jingxuan He https://www.sri.inf.ethz.ch/people/jingxuan

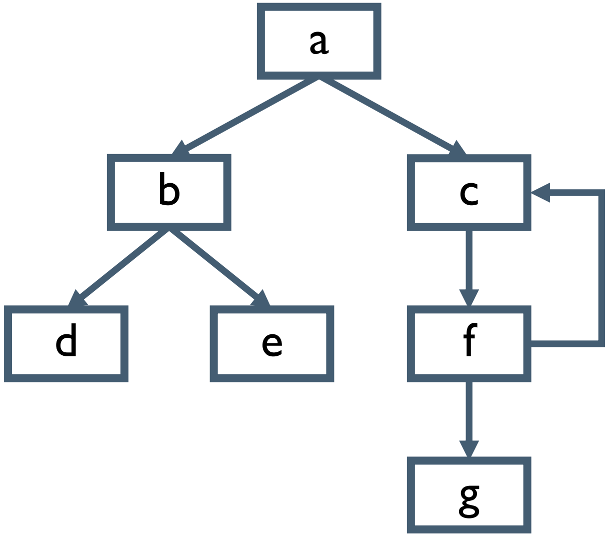
Joint work with: Gishor Sivanrupan, Petar Tsankov, Martin Vechev

Sep 2022 @ KLEE Workshop

SRILAB

ETH zürich

Symbolic Execution: Challenges and Goal



Candidate States:

a_0 b_0 c_0 d_0 e_0

Tests Generated:

a_0-b_0-e_0

Coverage Objective of Symbolic Execution:

$$\arg\max_{tests} \frac{|\bigcup_{t \in tests} \text{coverage}(t)|}{totalTime}$$

The Path Explosion Challenge:

#states is exponential in #branches

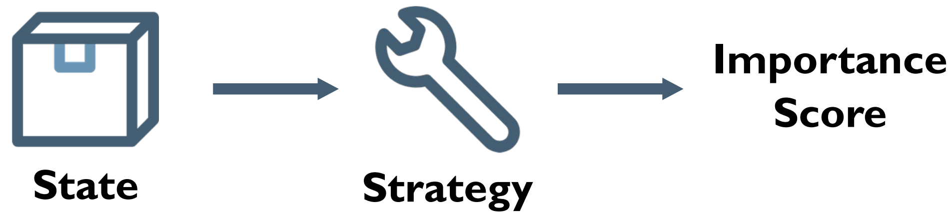
#states explodes at deep branches

e.g., 10k-100k states for coreutils

Goal: Obtain a good strategy that can select promising states

Define ML Problem and Model

State Selection Strategies:
(can be deterministic or probabilistic)



What is the ideal state selection strategy?

$$\text{reward}(s) = \frac{|\bigcup_{t \in \text{testsFrom}(s)} \text{coverage}(t)|}{\sum_{d \in \text{statesFrom}(s)} \text{stateTime}(d)} \quad \longleftrightarrow \quad \underset{tests}{\arg\max} \frac{|\bigcup_{t \in tests} \text{coverage}(t)|}{totalTime}$$

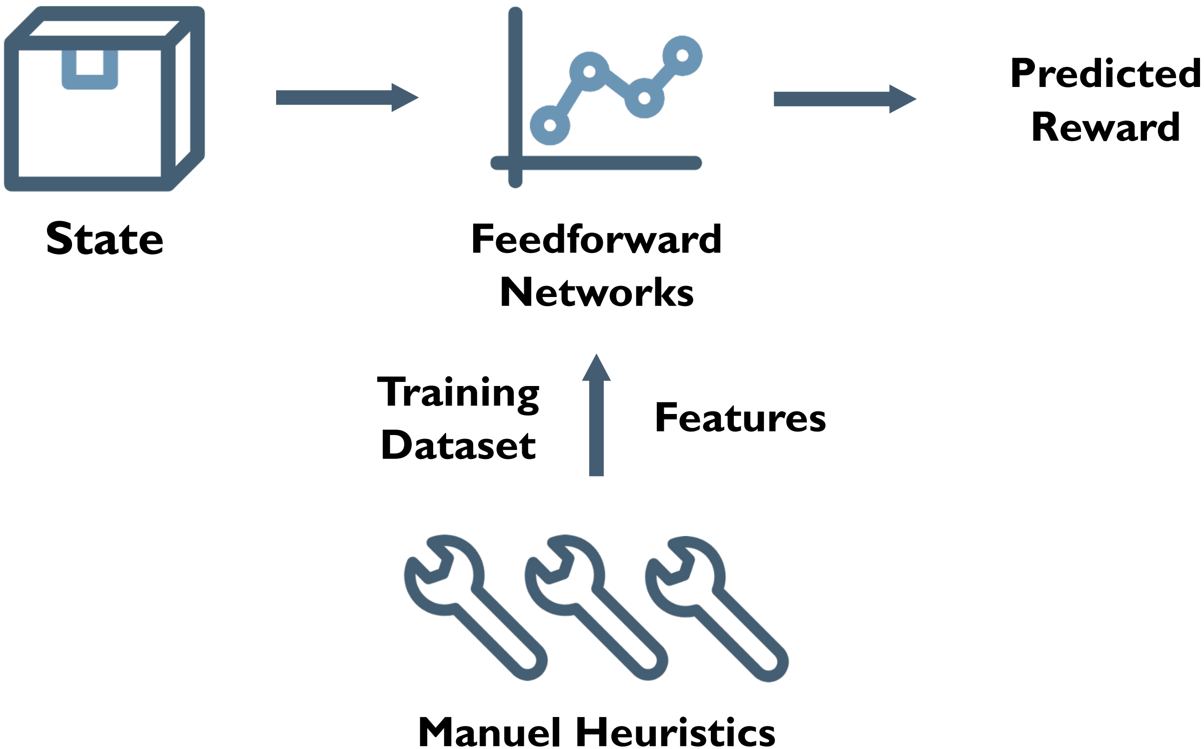
Selection with an ideal reward function | **Coverage objective of symbolic execution**

Cannot calculate testsFrom and statesFrom at test time!

The ideal selection cannot be achieved in general!

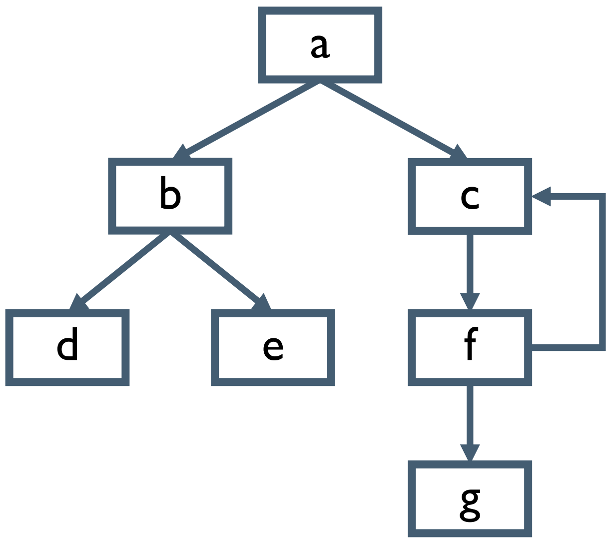
However, we can train a model to predict the ideal reward!

Learch: Our Learned Strategy



(based on some simple properties of the input state)

Obtaining a Supervised Dataset



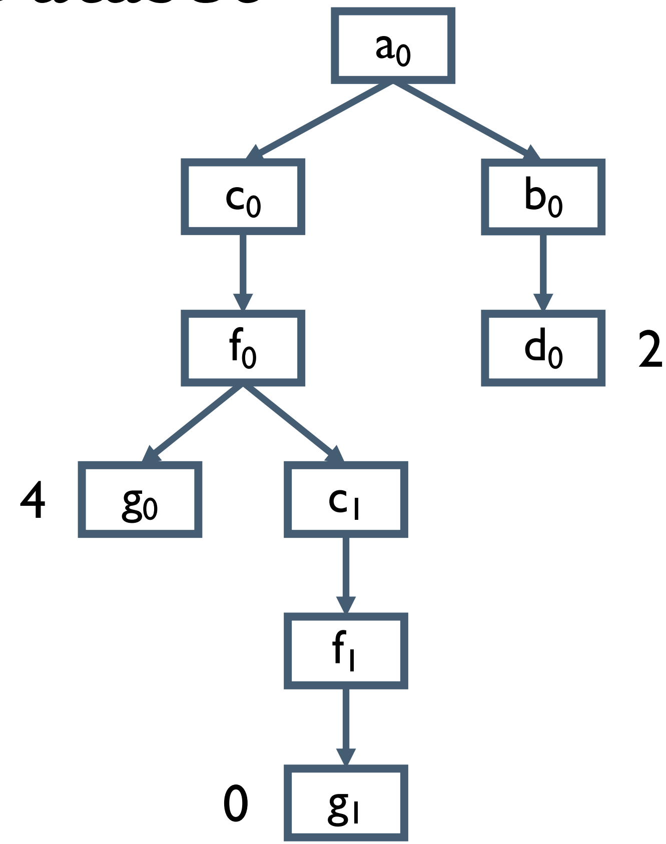
	States	Cov	NewCov
1	a_0-c_0-f_0-g_0	a, c, f, g	a, c, f, g
2	a_0-c_0-f_0-c_1-f_1-g_1	a, c, f, g	Ø
3	a_0-b_0-d_0	a, b, d	b, d

a_0	c_0	f_0	g_0	c_1	f_1	g_1	b_0	d_0
1	2	2	2	1	1	2	2	2

Time Spent by Each State

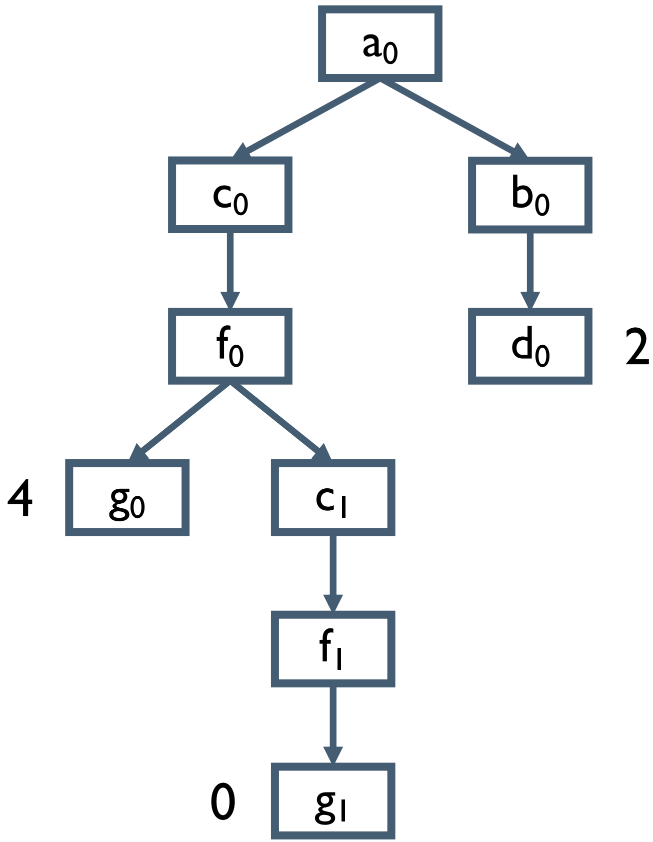
Obtaining a Supervised Dataset

	States	Cov	NewCov
1	a_0-c_0-f_0-g_0	a, c, f, g	a, c, f, g
2	a_0-c_0-f_0-c_1-f_1-g_1	a, c, f, g	Ø
3	a_0-b_0-d_0	a, b, d	b, d



Tests Tree

Obtaining a Supervised Dataset



State	Time	TotalCov	TotalTime	Reward
a_0	1	6	15	0.4
c_0	2	4	10	0.4
f_0	2	4	8	0.5
g_0	2	4	2	2
c_1	1	0	4	0
f_1	1	0	3	0
g_1	2	0	2	0
b_0	2	2	4	0.5
d_0	2	2	2	1

Obtaining a Supervised Dataset

Procedure genData

Input: a set of training programs 
a set of strategies 

Output: a supervised dataset 

 ← ∅

For each  **and** 

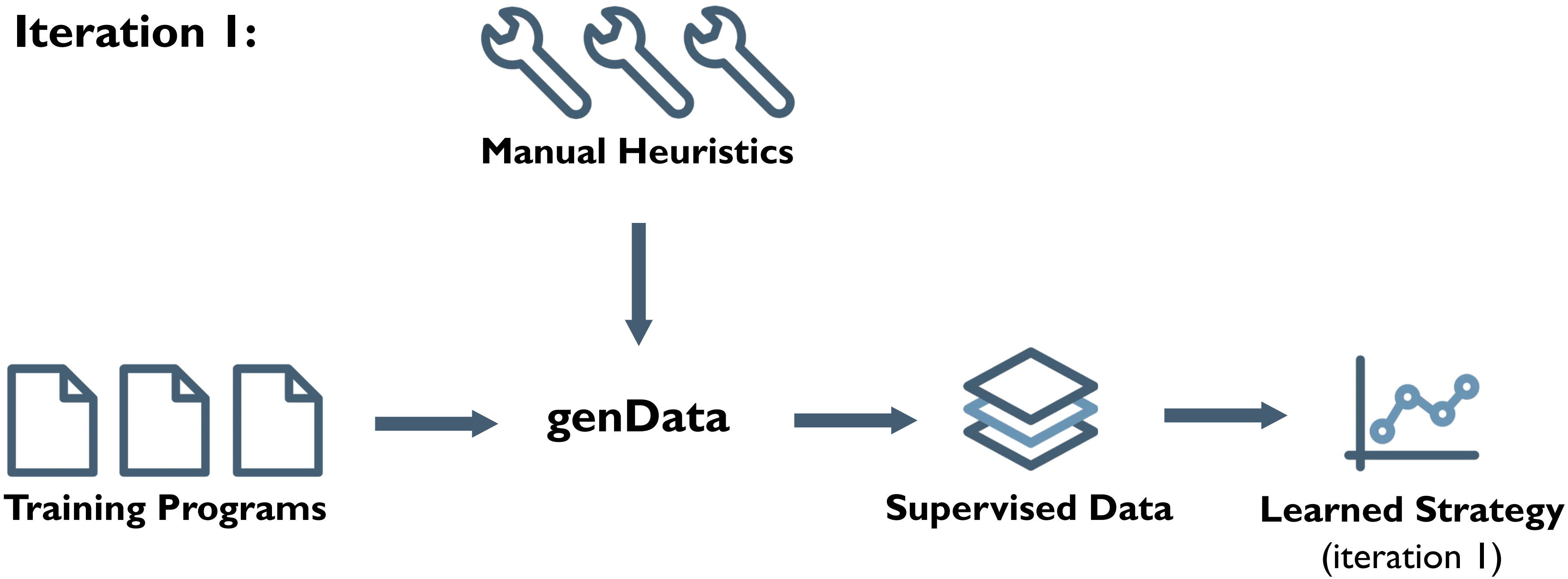
Obtain new data  **on**  **with** 

Add  **to** 

Return 

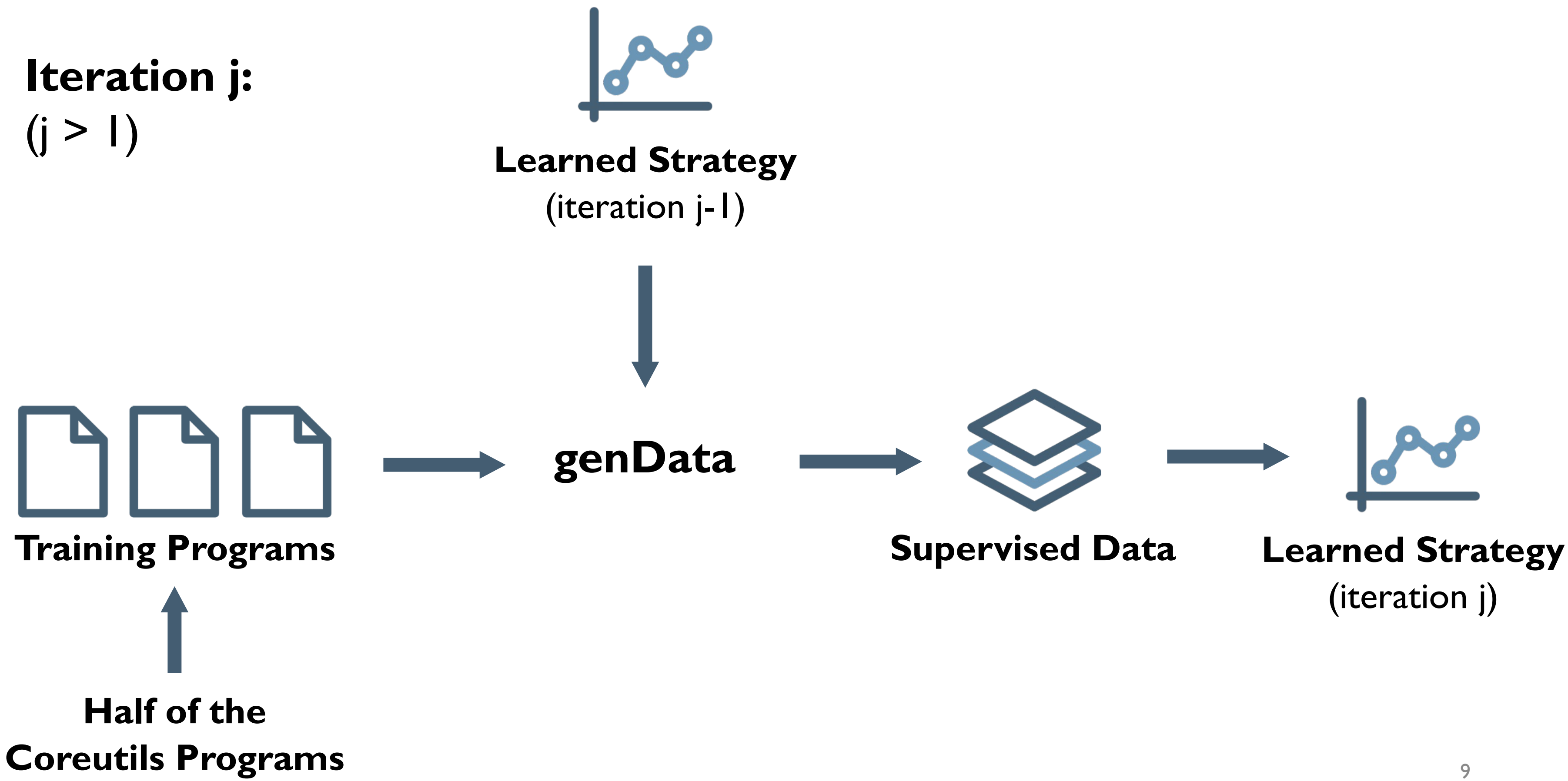
Final Iterative Learning Algorithm

Iteration 1:



Final Iterative Learning Algorithm

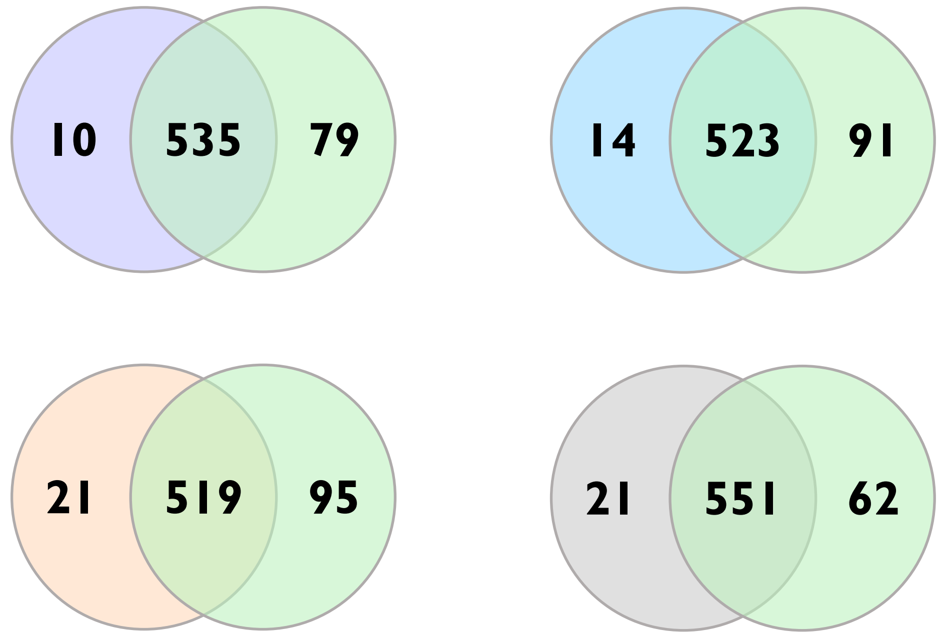
Iteration j:
(j > 1)



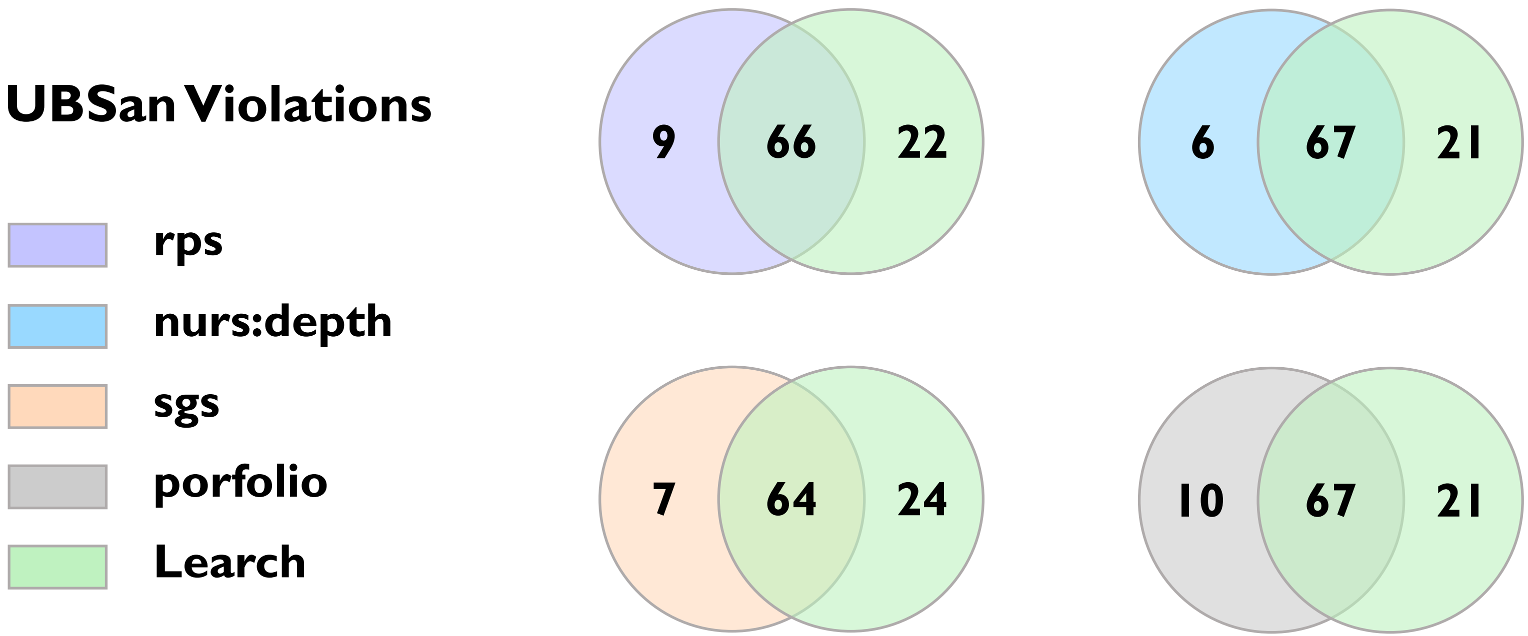
Evaluation: Coreutils Test Set

Line Coverage

- rps
- nurs:depth
- sgs
- porfolio
- Learch



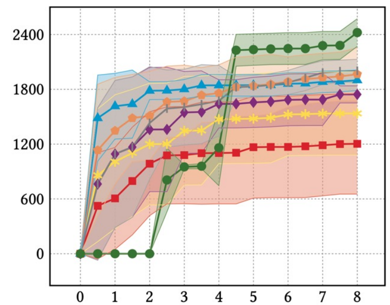
Evaluation: Coreutils Test Set



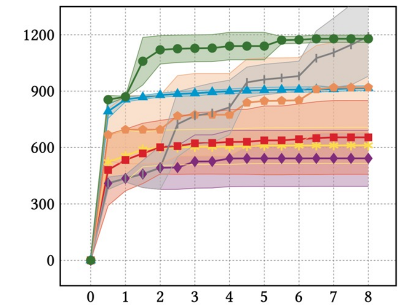
Generalization: 10 Real-world Programs

Line Coverage over Time (h)

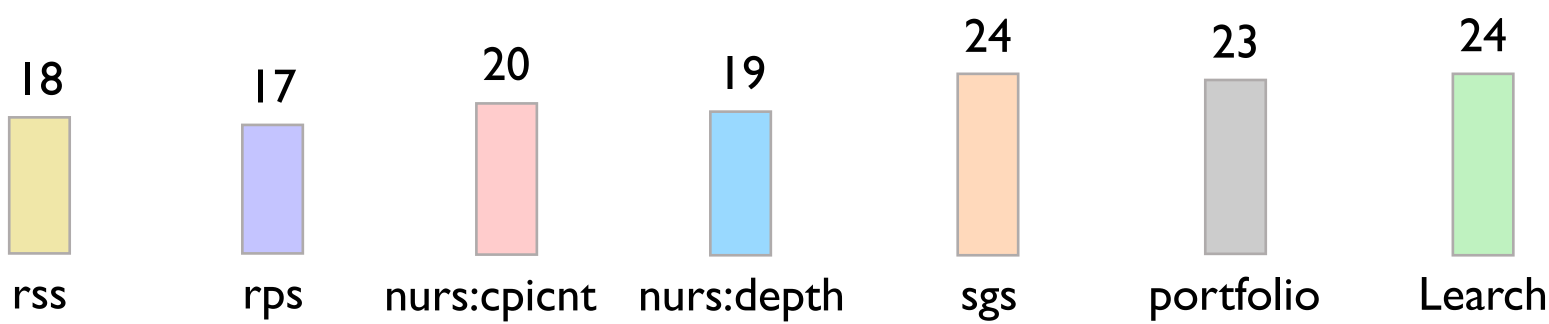
grep



readelf



Detecting UBSan Violations

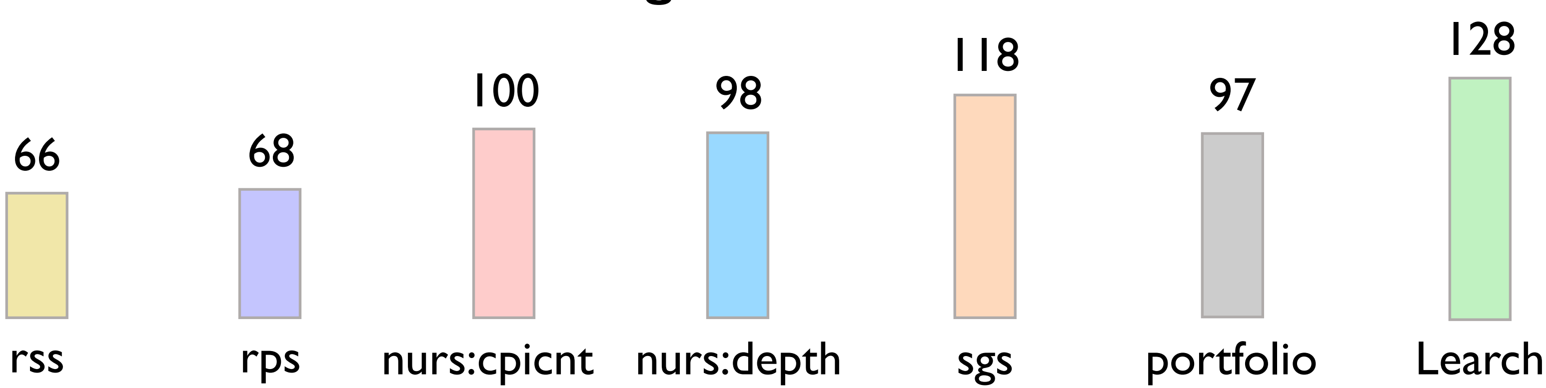


Generating Seeds for AFL

Discovering Paths

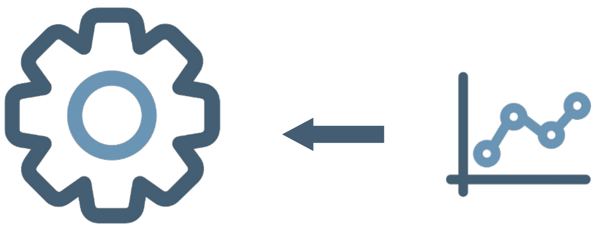
objcopy		readelf		make		sqlite	
sgs	2489	nurs:depth	4133	rps	5582	sgs	4243
Learch	2882	Learch	4531	Learch	5689	Learch	4364

Detecting UBSan Violations



ML-driven Program Analysis

Paradigm



Classic Analysis

Learned Models

Effective and Efficient

Analysis provides gaurantees

Based on classic framework

General Receipe

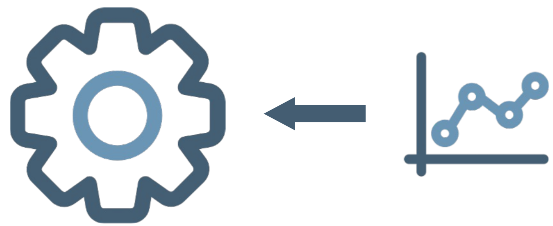
- **Identify challengs and goal**
- **Define ML problem and model**
- **Obtain a supervised dataset**
- **Iteratively refine learned models**

Instantiations

Learch — **Learn to explore paths for symbolic execution**
[CCS' 21. **He**, Sivanrupan, Tsankov, Vechev]

Lait — **Learn to approximate for numerical analysis**
[PLDI' 20. **He**, Singh, Püschel, Vechev]

ILF — **learn to fuzz from symbolic execution**
[CCS' 19. **He**, Balunovic, Ambroladze, Tsankov, Vechev]



Learch: ML-driven Path Exploration

https://github.com/eth-sri/learch