# Learning to Explore Paths for Symbolic Execution

**Jingxuan He**, Gishor Sivanrupan, Petar Tsankov, Martin Vechev

SRILAB ETH zürich

@ ACM CCS 2021

# Symbolic Execution

## A powerful technique widely adopted in security

Analyzing Protocol Implementations

Validating Hardware Design

Securing Smart Contracts

## Can be used to generate "good" tests

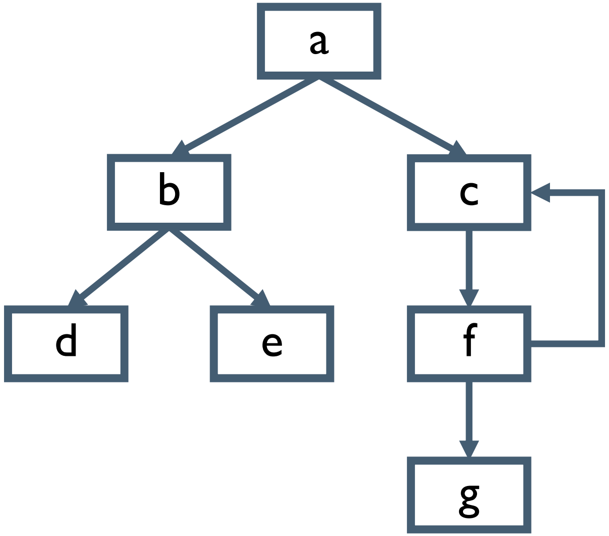
SAGE

Symbolic PathFinder

Apollo

Icons made by Kiranshastry from www.flaticon.com

# Path Exploration and Explosion



**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:**

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

**Path Explosion:**

**#states is exponential in #branches**

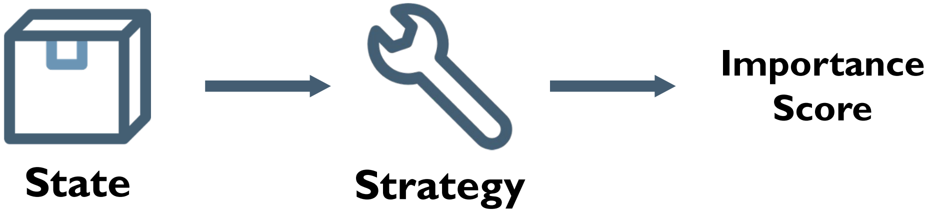
**#states explodes at deep branches**

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

**Need a Good Strategy to Select Promising States!**

# State Selection Strategies

**State Selection Strategies:**
(can be deterministic or probabilistic)



**The Ideal State Selection Strategy?**

**Coverage Objective of Symbolic Execution:** $\underset{tests}{\arg\max} \frac{|\bigcup_{t \in tests} \mathrm{coverage}(t)|}{totalTime}$

**Selection with an Ideal Reward Function:** $\mathrm{reward}(s) = \frac{|\bigcup_{t \in \mathrm{testsFrom}(s)} \mathrm{coverage}(t)|}{\sum_{d \in \mathrm{statesFrom}(s)} \mathrm{stateTime}(d)}$

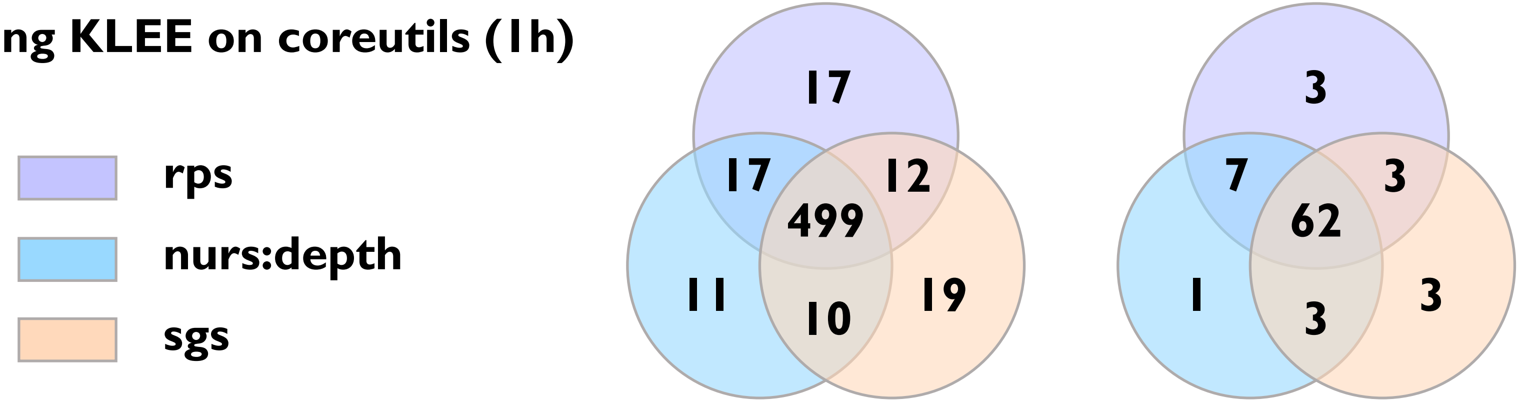
**Cannot Calculate testsFrom and statesFrom!**

Icons made by Kiranshastry from www.flaticon.com

# Existing State Selection Heuristics

**Existing Heuristics:** select states based on certain property of the states. Often get stuck in program parts favoring the property but fail to explore other parts

**Running KLEE on coreutils (1h)**

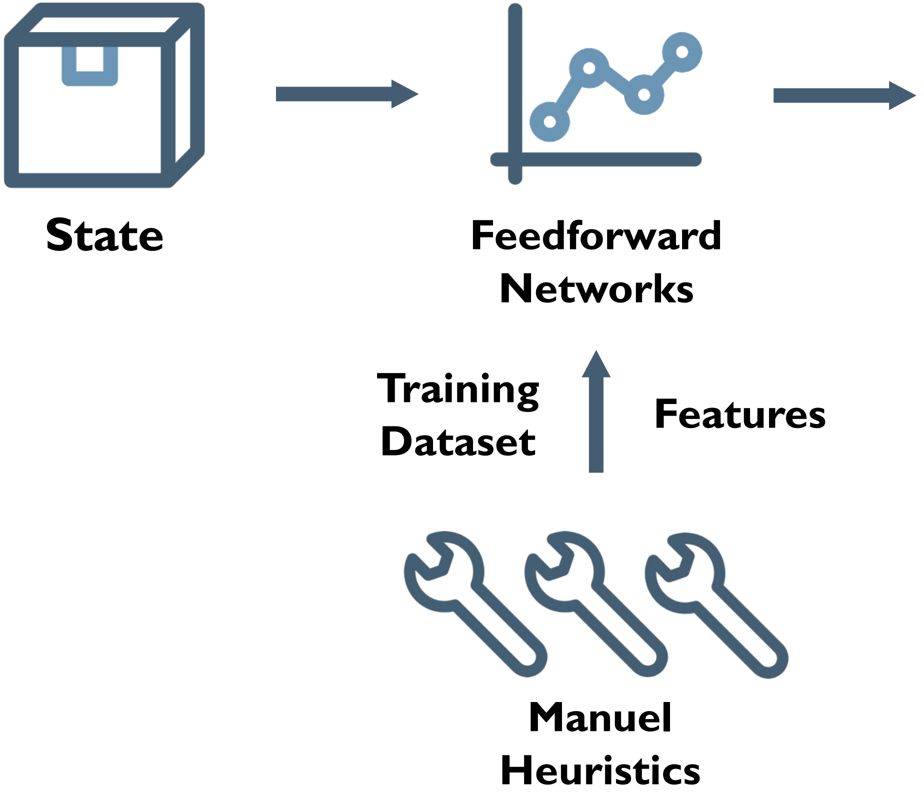


**Line Coverage**
(585 in total)

**UBSan Violations**
(82 in total)

**Expectation for Learning:** an adaptive strategy subsuming individual heuristics

# Learch: our Learned Strategy



State

Feedforward Networks

Training Dataset

Features

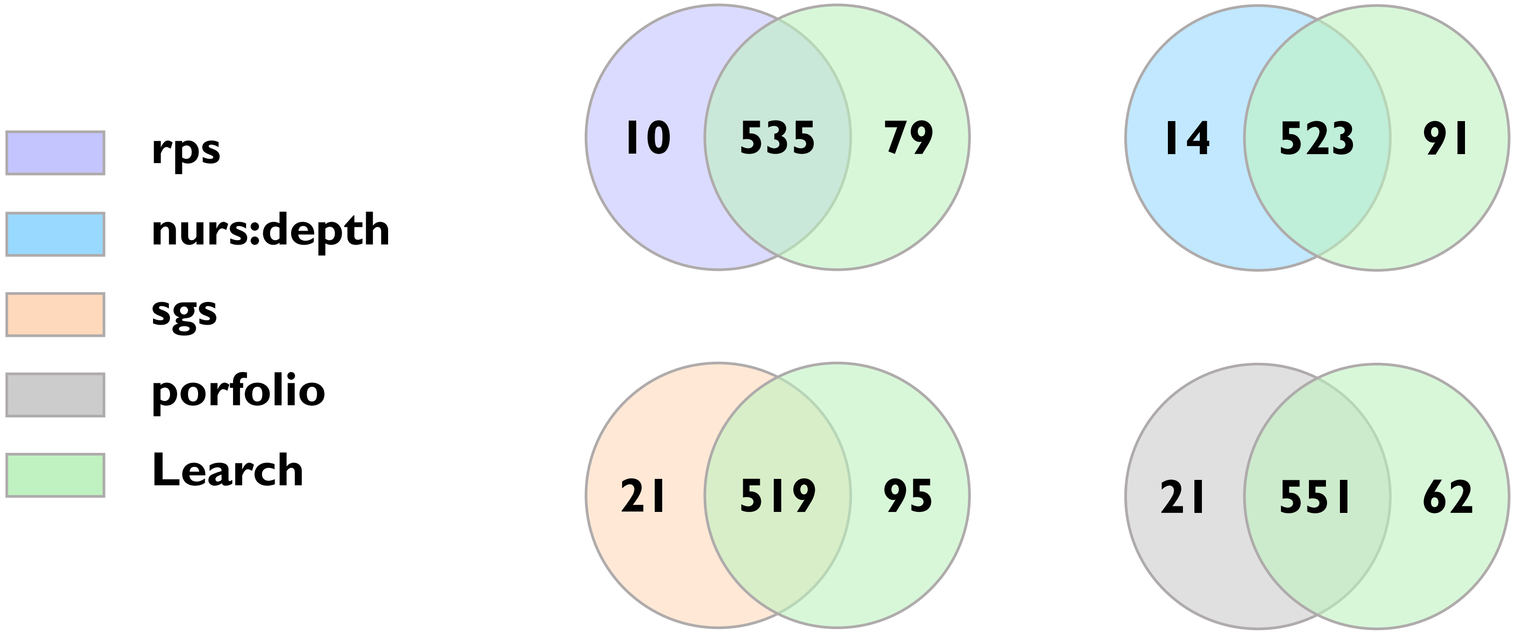
Manuel Heuristics

Predicted Reward

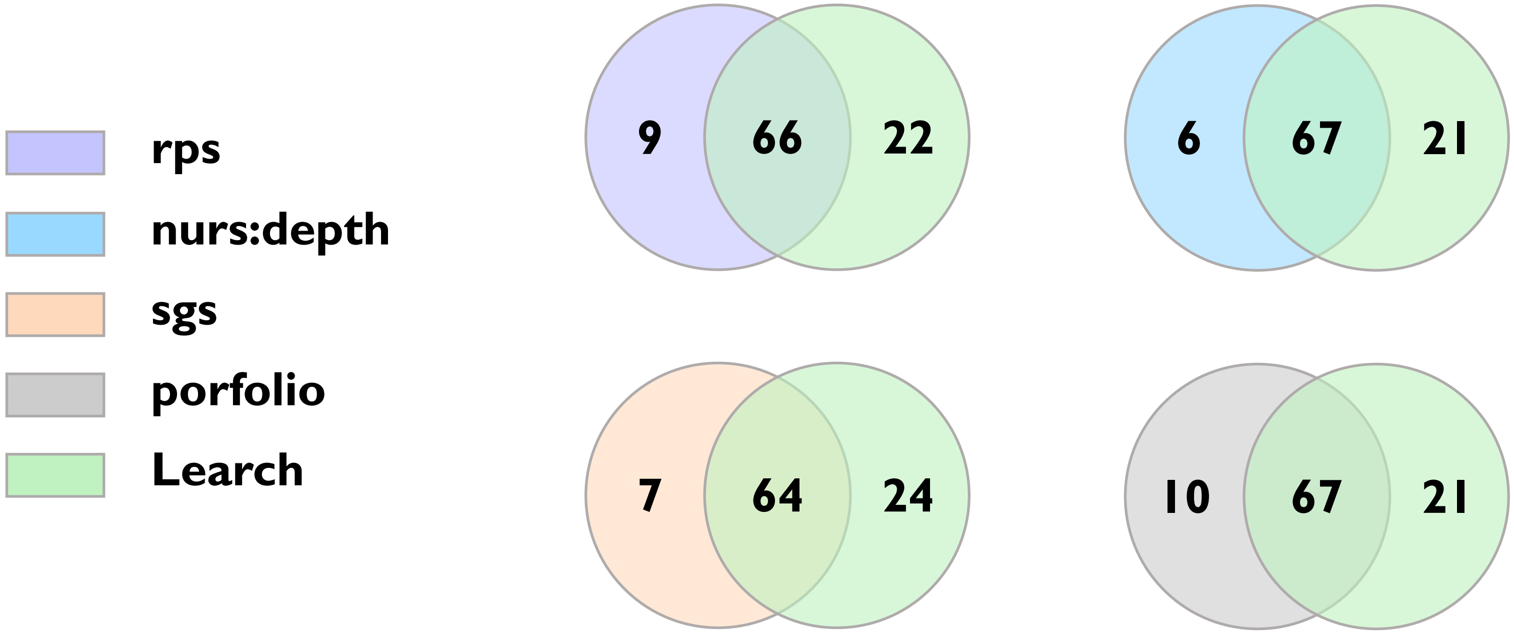
$$\frac{|\bigcup_{t\in\text{testsFrom}(s)} \text{coverage}(t)|}{\sum_{d\in\text{statesFrom}(s)} \text{stateTime}(d)}$$

Icons made by Kiranshastry from www.flaticon.com

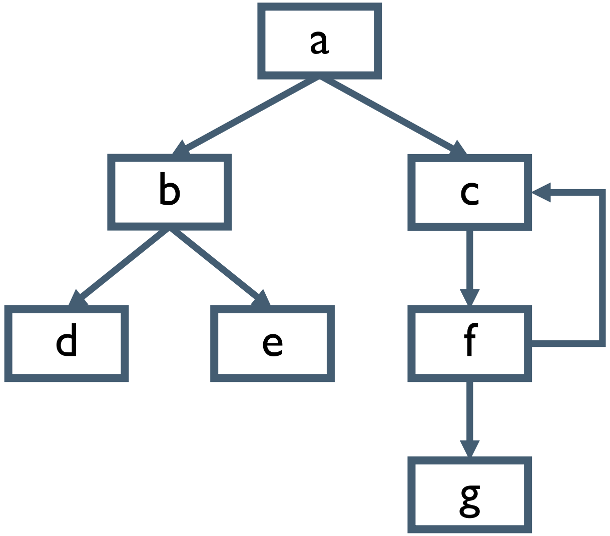
# Learch: Line Coverage on coreutils



# Learch: UBSan Violations on coreutils



# Obtaining a Supervised Dataset



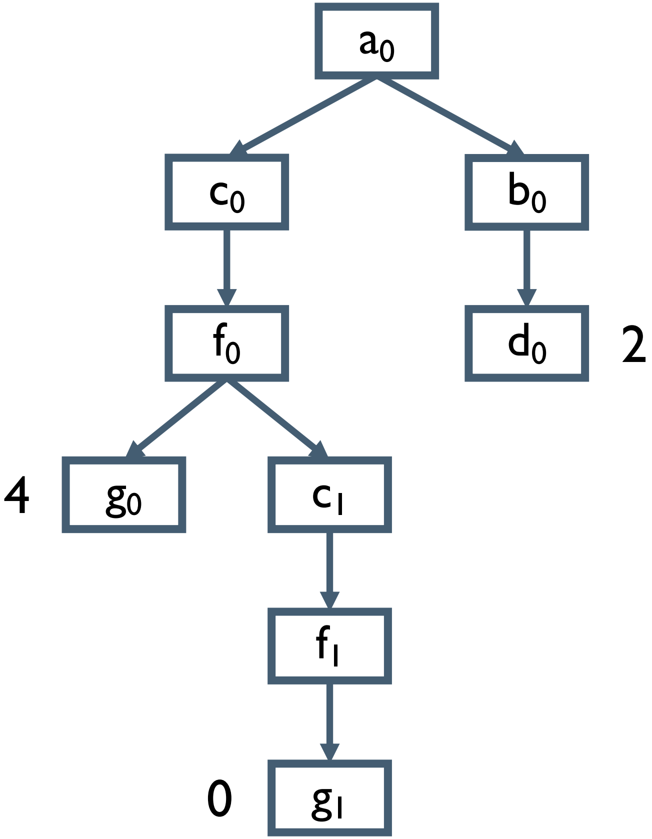
| | **States** | **Cov** | **NewCov** |
|---|---|---|---|
| 1 | $a_0$-$c_0$-$f_0$-$g_0$ | a, c, f, g | a, c, f, g |
| 2 | $a_0$-$c_0$-$f_0$-$c_1$-$f_1$-$g_1$ | a, c, f, g | Ø |
| 3 | $a_0$-$b_0$-$d_0$ | a, b, d | b, d |

| $a_0$ | $c_0$ | $f_0$ | $g_0$ | $c_1$ | $f_1$ | $g_1$ | $b_0$ | $d_0$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 2 | 2 | 1 | 1 | 2 | 2 | 2 |

**Time Spent by Each State**

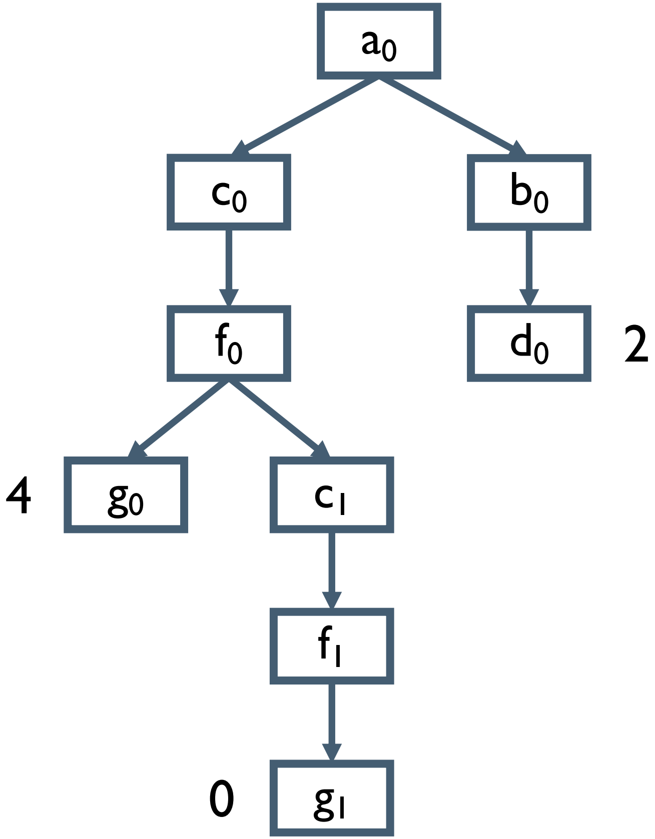
# Obtaining a Supervised Dataset

| | **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 |

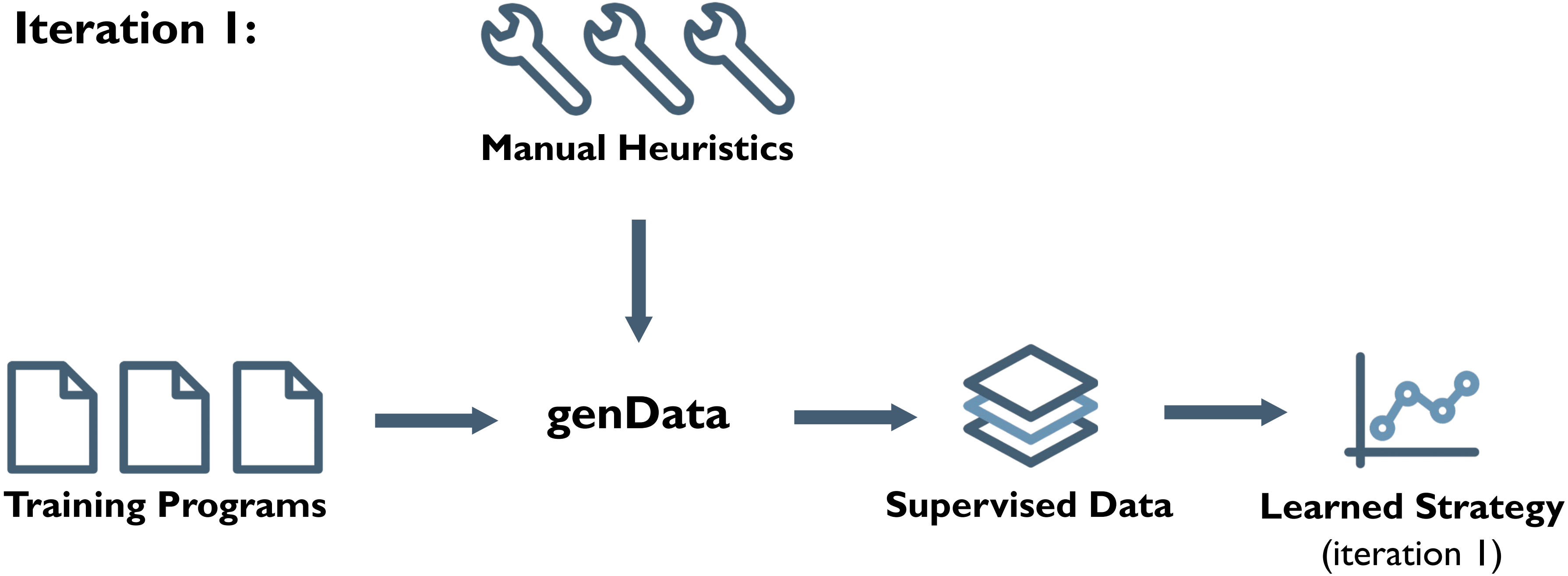
# Obtaining a Supervised Dataset

```
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 ≋
```

Icons made by Kiranshastry from www.flaticon.com

# Final Iterative Learning Algorithm

**Iteration 1:**



**Manual Heuristics**

**Training Programs**

**genData**

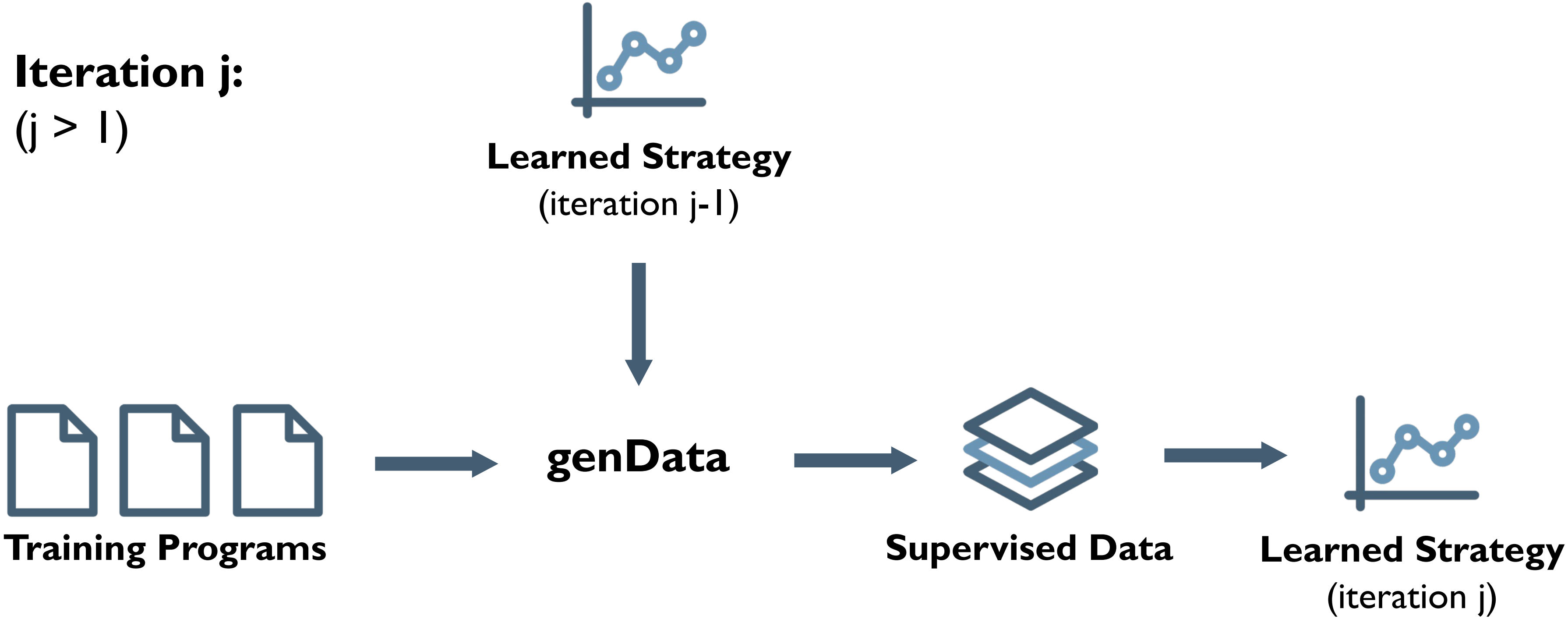
**Supervised Data**

**Learned Strategy**
(iteration 1)

Icons made by Kiranshastry from www.flaticon.com

# Final Iterative Learning Algorithm

**Iteration j:**
(j > 1)



**Learned Strategy**
(iteration j-1)

**genData**

**Training Programs**

**Supervised Data**

**Learned Strategy**
(iteration j)

Icons made by Kiranshastry from www.flaticon.com

# Instantiation Learch on KLEE

**Features:** stack, successor, testCase, coverage, constraint, **depth**, **cpicnt**, **icnt**, **covNew**, **subpath**

**UBSan Violations:** Integer overflow, oversized shift, out-of-bound array reads/writes, pointer overflow, null deference

Run 4 learned strategies, each taking a quarter of the total time limit, and combine all generated tests

# Evaluation: Line Coverage (8h runs)

**diff**

| | |
|---|---|
| portfolio | 1136 |
| Learch | 1791 |

**grep**

| | |
|---|---|
| portfolio | 2003 |
| Learch | 2421 |

**gawk**

| | |
|---|---|
| sgs | 2885 |
| Learch | 3097 |

**patch**

| | |
|---|---|
| sgs | 1412 |
| Learch | 1502 |

**objcopy**

| | |
|---|---|
| rss | 2131 |
| Learch | 2827 |

**readelf**

| | |
|---|---|
| portfolio | 1192 |
| Learch | 1179 |

**make**

| | |
|---|---|
| portfolio | 2353 |
| Learch | 2398 |

**sqlite**

| | |
|---|---|
| nurs:cpicnt | 5204 |
| Learch | 5590 |

**find**

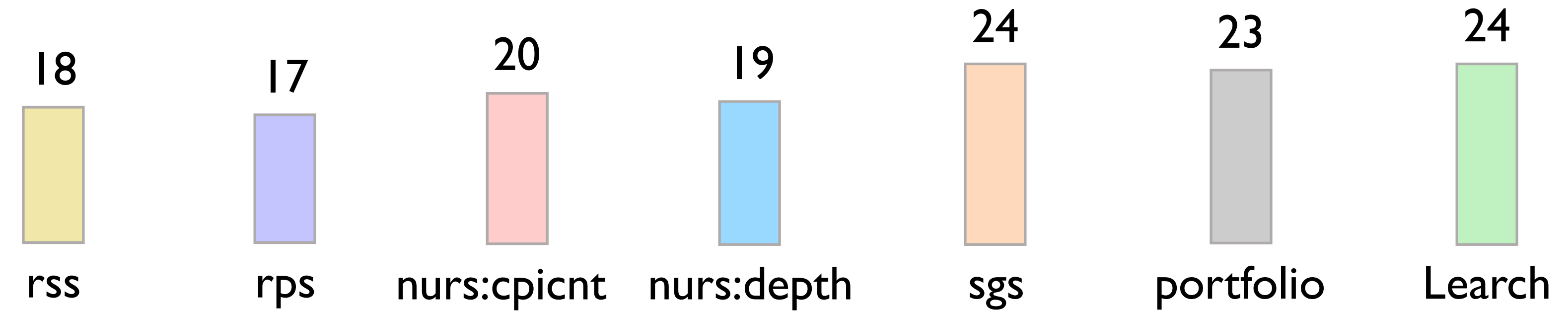
| | |
|---|---|
| portfolio | 3142 |
| Learch | 2927 |

**cjson**

| | |
|---|---|
| portfolio | 551 |
| Learch | 541 |

**On Average, >20% increase than all heuristics**

# Evaluation: UBSan Violations (8h run)



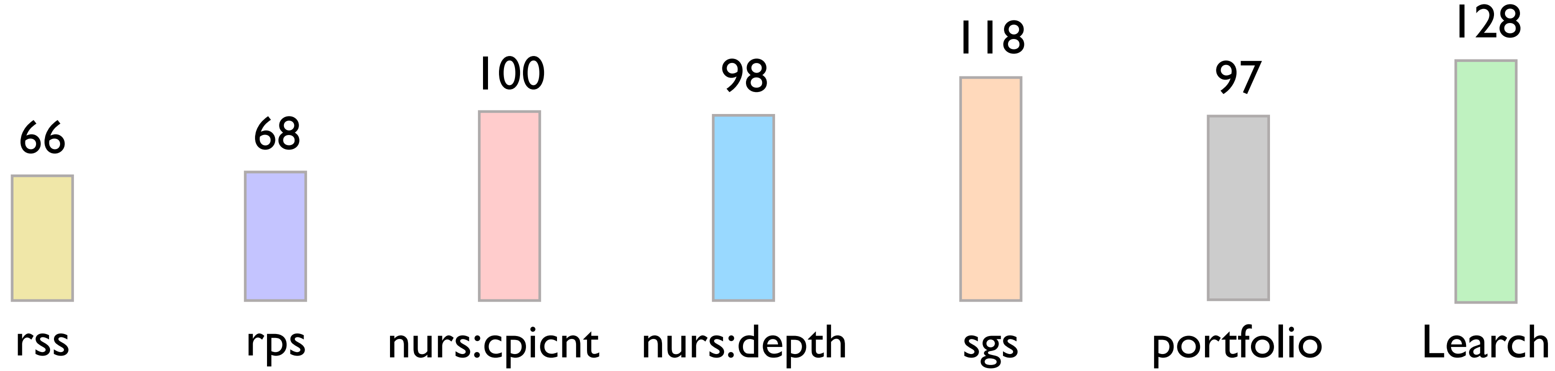
**46 reports to developers, 13 confirmed, 11 fixed**

# Evaluation: Seeding AFL (8h runs)

## 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



# Evaluation: Design Choices (1h runs)

## Line Coverage

| | | | | |
|---|---|---|---|---|
| **566** | **566** | **560** | **563** | **618** |
| 4 individual strategies | | | | Learch |

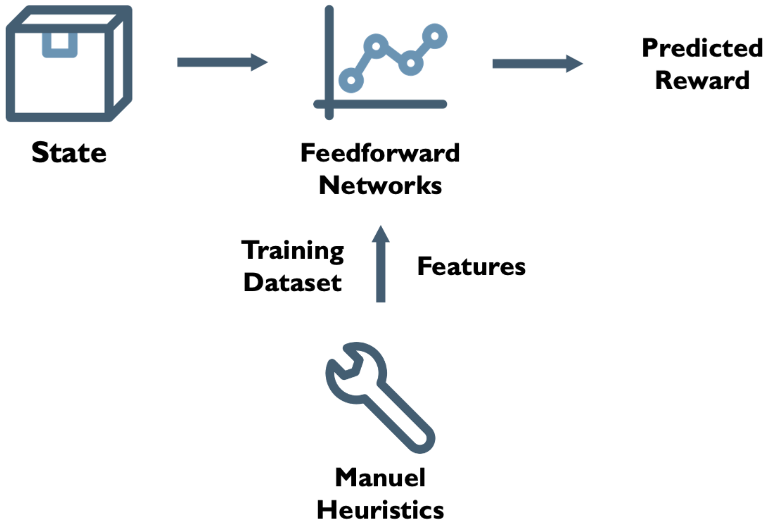
| | | |
|---|---|---|
| **517** | **541** | **618** |
| linear | rnn | Learch |

## UBSan Violations

| | | | | |
|---|---|---|---|---|
| **71** | **75** | **70** | **93** | **88** |
| 4 individual strategies | | | | Learch |

| | | |
|---|---|---|
| **62** | **70** | **88** |
| linear | rnn | Learch |

# Summary



State

Feedforward Networks

Predicted Reward

Training Dataset

Features

Manuel Heuristics

eth-sri/learch

SRILAB

https://www.sri.inf.ethz.ch/