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

Coverage Objective of Symbolic Execution:

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

Path Explosion:
#states is exponential in #branches
#states explodes at deep branches
e.g., 10k-100k states for coreutils

Candidate States:
a₀ b₀ c₀ d₀ e₀

Tests Generated:
a₀-b₀-e₀

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:

Selection with an Ideal Reward Function:

Cannot Calculate testsFrom and statesFrom!
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)

Expectation for Learning: an adaptive strategy subsuming individual heuristics
Learch: our Learned Strategy

State $\xrightarrow{\text{Feedforward Networks}}$ Predicted Reward

$| \bigcup_{t \in testsFrom(s)} \text{coverage}(t) |$

$\frac{1}{\sum_{d \in statesFrom(s)} \text{stateTime}(d)}$

Training Dataset $\xrightarrow{\text{Features}}$

Manuel Heuristics

Icons made by Kiranshastry from www.flaticon.com
Learch: Line Coverage on coreutils

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

Venn Diagram:

- **rps**
  - 10
  - 535
  - 79
- **nurs:depth**
  - 14
  - 523
  - 91
- **sgs**
  - 21
  - 519
  - 95
- **porfolio**
  - 21
  - 551
  - 62
Learch: UBSan Violations on coreutils

- rps
- nurs:depth
- sgs
- portfolio
- Learch

Venn Diagram with overlapping elements for each category and their respective counts.
Obtaining a Supervised Dataset

States  Cov  NewCov
1  a₀-c₀-f₀-g₀  a, c, f, g  a, c, f, g
2  a₀-c₀-f₀-c₁-f₁-g₁  a, c, f, g  ∅
3  a₀-b₀-d₀  a, b, d  b, d

<table>
<thead>
<tr>
<th>a₀</th>
<th>c₀</th>
<th>f₀</th>
<th>g₀</th>
<th>c₁</th>
<th>f₁</th>
<th>g₁</th>
<th>b₀</th>
<th>d₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Time Spent by Each State
Obtaining a Supervised Dataset

<table>
<thead>
<tr>
<th>States</th>
<th>Cov</th>
<th>NewCov</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$a_0$-$c_0$-$f_0$-$g_0$</td>
<td>a, c, f, g</td>
</tr>
<tr>
<td>2</td>
<td>$a_0$-$c_0$-$f_0$-$c_1$-$f_1$-$g_1$</td>
<td>a, c, f, g</td>
</tr>
<tr>
<td>3</td>
<td>$a_0$-$b_0$-$d_0$</td>
<td>a, b, d</td>
</tr>
</tbody>
</table>

Tests Tree

- $a_0$
- $c_0$
- $b_0$
- $d_0$
- $f_0$
- $g_0$
- $c_1$
- $f_1$
- $g_1$
Obtaining a Supervised Dataset

State | Time | TotalCov | TotalTime | Reward
---|---|---|---|---
a₀ | 1 | 6 | 15 | 0.4
b₀ | 2 | 4 | 10 | 0.4
c₀ | 2 | 4 | 8 | 0.5
d₀ | 2 | 2 | 2 | 1
g₀ | 2 | 4 | 2 | 2
c₁ | 1 | 0 | 4 | 0
f₁ | 1 | 0 | 3 | 0
g₁ | 2 | 0 | 2 | 0
b₀ | 2 | 2 | 4 | 0.5
d₀ | 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:

Training Programs → genData → Supervised Data → Learned Strategy (iteration 1)

Manual Heuristics

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 dereference

Run 4 learned strategies, each taking a quarter of the total time limit, and combine all generated tests
### Evaluation: Line Coverage (8h runs)

<table>
<thead>
<tr>
<th>Tool</th>
<th>Portfolio</th>
<th>Learch</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>diff</strong></td>
<td>1136</td>
<td>1791</td>
</tr>
<tr>
<td><strong>grep</strong></td>
<td>2003</td>
<td>2421</td>
</tr>
<tr>
<td><strong>gawk</strong></td>
<td>sgs 2885</td>
<td>Learch 3097</td>
</tr>
<tr>
<td><strong>patch</strong></td>
<td>sgs 1412</td>
<td>Learch 1502</td>
</tr>
<tr>
<td><strong>objc</strong></td>
<td>rss 2131</td>
<td>Learch 2827</td>
</tr>
<tr>
<td><strong>readelf</strong></td>
<td>portfolio 1192</td>
<td>Learch 1179</td>
</tr>
<tr>
<td><strong>make</strong></td>
<td>portfolio 2353</td>
<td>Learch 2398</td>
</tr>
<tr>
<td><strong>sqlite</strong></td>
<td>nurs:cpicnt 5204</td>
<td>Learch 5590</td>
</tr>
<tr>
<td><strong>find</strong></td>
<td>portfolio 3142</td>
<td>Learch 2927</td>
</tr>
<tr>
<td><strong>cjson</strong></td>
<td>portfolio 551</td>
<td>Learch 541</td>
</tr>
</tbody>
</table>

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**
  - sgs: 2489
  - Learch: 2882

- **readelf**
  - nurs:depth: 4133
  - Learch: 4531

- **make**
  - rps: 5582
  - Learch: 5689

- **sqlite**
  - sgs: 4243
  - Learch: 4364

**Detecting UBSan Violations**

- rss: 66
- rps: 68
- nurs:cpicnt: 100
- nurs:depth: 98
- sgs: 118
- portfolio: 97
- Learch: 128
## Evaluation: Design Choices (1h runs)

<table>
<thead>
<tr>
<th>Line Coverage</th>
<th>UBSan Violations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>566 566 560 563</strong></td>
<td><strong>71 75 70 93</strong></td>
</tr>
<tr>
<td>4 individual strategies</td>
<td>4 individual strategies</td>
</tr>
<tr>
<td><strong>618</strong></td>
<td><strong>88</strong></td>
</tr>
<tr>
<td>Learch</td>
<td>Learch</td>
</tr>
</tbody>
</table>

| **517 541 618** | **62 70 88** |
| linear  | rnn  | Learch | linear  | rnn  | Learch |
Summary

State → Feedforward Networks → Predicted Reward

Training Dataset → Features

Manuel Heuristics

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