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
Symbolic Execution: **Challenges and Goal**

**Coverage Objective of Symbolic Execution:**

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

**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?

reward(s) = \frac{\left| \bigcup_{t \in \text{testsFrom}(s)} \text{coverage}(t) \right|}{\sum_{d \in \text{statesFrom}(s)} \text{stateTime}(d)}

Selection with an ideal reward function

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

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

State $\rightarrow$ Feedforward Networks $\rightarrow$ Predicted Reward

Training Dataset $\uparrow$ Features

Manuel Heuristics
(based on some simple properties of the input state)
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

Time Spent by Each State

<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>
## Obtaining a Supervised Dataset

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

**Tests Tree**
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
g₀ | 2 | 4 | 2 | 2
c₁ | 1 | 0 | 4 | 0
f₀ | 2 | 4 | 8 | 0.5
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

\[ \text{Empty Set} \]

**For each** and

**Obtain new data** on with

**Add** to

**Return**
Final Iterative Learning Algorithm

Iteration 1:

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

Manual Heuristics
Final Iterative Learning Algorithm

Iteration j:
\( (j > 1) \)

Learned Strategy
\( \text{(iteration } j-1) \)

Training Programs
\( \text{genData} \)

Supervised Data
\( \text{Learned Strategy} \)
\( \text{(iteration } j) \)

Half of the Coreutils Programs
Evaluation: Coreutils Test Set

Line Coverage

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

![Venn Diagrams]

- First set: rps
- Second set: nurs:depth
- Third set: sgs
- Fourth set: porfolio
- Fifth set: Learch

- Intersection counts:
  - rps and nurs:depth: 10
  - rps and sgs: 535
  - rps and porfolio: 79
  - rps and Learch: 21
  - nurs:depth and sgs: 14
  - nurs:depth and porfolio: 523
  - nurs:depth and Learch: 91
  - sgs and porfolio: 21
  - sgs and Learch: 519
  - porfolio and Learch: 95
Evaluation: Coreutils Test Set

UBSan Violations

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

Venn Diagrams:

1. Purple circle (rps): 9, 66, 22
2. Light blue circle (nurs:depth): 6, 67, 21
3. Orange circle (sgs): 7, 64, 24
4. Gray circle (portfolio): 10, 67, 21

Legend:
- Purple: rps
- Light blue: nurs:depth
- Orange: sgs
- Gray: porfolio
- Green: Learch
Generalization: 10 Real-world Programs

Line Coverage over Time (h)

grep

Detecting UBSan Violations

18
rss

17
rps

20
nurs:cpicnt

19
nurs:depth

24
sgs

23
portfolio

24
Learch
Generating Seeds for AFL

Discovering Paths

<table>
<thead>
<tr>
<th>Tool</th>
<th>sgs</th>
<th>nurs:depth</th>
<th>Learch</th>
</tr>
</thead>
<tbody>
<tr>
<td>objcopy</td>
<td>2489</td>
<td>4133</td>
<td>2882</td>
</tr>
<tr>
<td>readelf</td>
<td>4133</td>
<td>Learch</td>
<td>4531</td>
</tr>
<tr>
<td>make</td>
<td>rps</td>
<td>5582</td>
<td>Learch</td>
</tr>
<tr>
<td>sqlite</td>
<td>sgs</td>
<td>4243</td>
<td>4364</td>
</tr>
</tbody>
</table>

Detecting UBSan Violations

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>rss</td>
<td>66</td>
</tr>
<tr>
<td>rps</td>
<td>68</td>
</tr>
<tr>
<td>nurs:cpicnt</td>
<td>100</td>
</tr>
<tr>
<td>nurs:depth</td>
<td>98</td>
</tr>
<tr>
<td>sgs</td>
<td>118</td>
</tr>
<tr>
<td>portfolio</td>
<td>97</td>
</tr>
<tr>
<td>Learch</td>
<td>128</td>
</tr>
</tbody>
</table>
ML-driven Program Analysis

**Paradigm**

- **Classic Analysis**
- **Learned Models**

**General Recipe**
- Identify challenges 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]

**Effective and Efficient**

- Analysis provides guarantees
- Based on classic framework
Learch: ML-driven Path Exploration

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