A Recursive Strategy for Symbolic Execution Expressed in Coq

Alyssa Byrnes

June 29, 2018

1 Data Types

- Object E to represent set of concrete states.
- Object of type n to represent nodes on the symbolic execution tree.
  - get_s(n) returns s, the symbolic state.
  - get_pc(n) returns π, the path constraint.
- Object of type $\mathcal{E}$ to represent a symbolic execution tree made of objects of type n.
  - is_leaf($\mathcal{E}$, n) returns *true* if n is a leaf in $\mathcal{E}$.
  - is_root($\mathcal{E}$, n) returns *true* if n is the root in $\mathcal{E}$.
  - get_root($\mathcal{E}$) returns object of type n that is the root of the tree.

Shorthand:

- $\bar{s}_{r,m}$ = get_s(get_root($\mathcal{E}_m$))
- $\pi_{r,m}$ = get_pc(get_root($\mathcal{E}_m$))
- $\bar{s}_{l,m}$ = get_s($n_{l,m}$), where $n_{l,m} \in \mathcal{E}_m$.
- $\pi_{l,m}$ = get_pc($n_{l,m}$), where $n_{l,m} \in \mathcal{E}_m$.

2 Circle Operations

These use the definitions defined in Arusoaie et al.'s paper [ALR13].

Definition 1 *circle_op_1 = the set $\varphi \in [\![\beta]\!]$ $\forall$ $\varphi' \in [\![\beta']\!]$ of a given β' such that $[\varphi]_\sim \Rightarrow^S [\varphi']_\sim$ where $\beta \Rightarrow \beta'$.*

Definition 2 *circle_op_2 = the set $\varphi' \in [\![\beta']\!]$ $\forall$ $\varphi \in [\![\beta]\!]$ of a given β such that $[\varphi]_\sim \Rightarrow^S [\varphi']_\sim$ where $\beta \Rightarrow \beta'$.*

circle_op_1 represents all concrete states that will take us down exactly one path in the symbolic execution tree. circle_op_2 represents all concrete output states of a given path in the symbolic execution tree.

3 Properties

For a given E, X = a sequence $\mathcal{E}_0, ..., \mathcal{E}_m$ such that $\forall \mathcal{E}_x$, $\exists n_{l,x}$ such that is_leaf($\mathcal{E}_x$, $n_{l,x}$) = *true* $\rightarrow$ the conjunction of the following:

1. $s_0 \in$ circle_op_1($\bar{s}_{r,0}$, $\pi_{l,0}$)
2. $E \bigcap$ circle_op_2($\bar{s}_{l,m}$, $\pi_{l,m}$) $\neq \{\}$
3. for $j = \{0, ..., m-1\}$, circle_op_2($\bar{s}_{l,j}$, $\pi_{l,j}$) $\subseteq$ circle_op_1($\bar{s}_{r,j+1}$, $\pi_{l,j+1}$)

4 Next Step

Currently, I am working on unifying the circle operation definitions and the tree representation.

References

[ALR13] Andrei Arusoaie, Dorel Lucanu, and Vlad Rusu. A generic framework for symbolic execution. In *International Conference on Software Language Engineering*, pages 281–301. Springer, 2013.