YouVerify

Release 1.0

Griffin Prechter

Dec 09, 2021

CONTENTS

class State.State

Abstract class template for a representation of State in *YouVerify*.

abstract advance_pc(*i: int*)

This abstract method is invoked when the framework wants to advance the program counter.

Parameters i – The number of statements to advance through.

Returns None

abstract assume(*cond: object*)

This abstract method is invoked when the framework wants to add an assumption to the current state.

Parameters cond – The condition to be affixed to the path constraint.

Returns None

abstract conditional_branch(*cond: object*, *destination: int*)

This abstract method is invoked when the framework reaches a conditional branching statement. Here, a developer would likely want to implement state splitting.

Parameters

- cond – The condition guarding the branch.
- destination – The destination program counter for the branch.

Returns None

abstract property current_statement

This abstract property is invoked to retrieve the next statement to be executed by the symbolic interpreter.

Returns The next Statement object to be *executed*.

Return type Statement

abstract jump(*destination: int*)

This abstract method is invoked when the framework reaches an unconditional branching statement.

Parameters destination – The destination program counter for the branch.

Returns None

abstract load_variable(*var: str*)

This abstract method is invoked when an identifier is evaluated and a variable is to be fetched from the states variable store.

Parameters `var` – The identifier of the variable that will be retrieved.

Returns The value of the variable that was retrieved.

`abstract next_state()`

This abstract method is invoked by the framework when before the next statement is to be executed. This gives the `State` object the opportunity to modify the backend and setup for the next statement execution. A developer would modify this to change how the paths of a program are explored and how state is handled.

Returns `True` if there is more state to be executed, `False` otherwise.

Return type bool

`abstract store_variable(`*var: str*, *val: object*`)`

This abstract method is invoked when the framework wants to assign a value to a variable.

Parameters

- `var` – The identifier of the variable being assigned to.
- `val` – The value to be assigned to the variable.

Returns None

`Wrappers.unary_operator_wrapper(`*f: function*`)`

The wrapper takes in the original unary operator as its parameter and optionally returns a new function with the new behavior.

Parameters `f` – The unary function to be wrapped.

Returns A new unary function.

`Wrappers.binary_operator_wrapper(`*f: function*`)`

The wrapper takes in the original binary operator as its parameter and optionally returns a new function with the new behavior.

Parameters `f` – The binary function to be wrapped.

Returns A new binary function.

PYTHON MODULE INDEX

INDEX