Program Analysis Verification & Testing Assignment 2

Name:- Vaibhav Rahulkumar Sheth

Roll No.:- 231110054

Department:- MTech CSE

Introduction

The code presented here is designed for program synthesis using symbolic execution. This report provides an overview of the code, its implementation details, and highlights its usage. The code leverages the Z3 solver and related libraries to perform symbolic execution and check equivalence between two sets of constraints.

Implementation:-

Example Function (example): Demonstrates the usage of symbolic variables, constraints, and assignments using the Z3 solver

Check Equivalence Function (checkEq()) :-

First of all, we are loading JSON Data of both Programs P1 and P2. testData.json file is for program P1 and testData2.json file is for Program P2.

Then we are initializing integer variables of Solver() method for solving equations.

Then we are extracting data from JSON files like input parameters and SymbEnc Equations and storing into list.

Then we replace ':' of equation by '==' in Inputs. Then we are doing AND operation between inputs list.

Same thing we are doing with SymbEnc equations of P2 by replacing ':' by '=='.

Same thing doing for program P1 for constraints parameter and SymbEnc equations. AND operation between constraints of P1. Converting into list Inputs and Equations of P1.

Now I'm removing constraints equations from Program p1 and preserve only equations of variables.

Till now we are extracting required data from JSON file and now we will implement our main logic function

We are initializing two instance of Solver() method.

Outer For loop iterates from 0 to number of inputs and inner for loop runs from 0 to constraints of P1.

In every iteration we are reseting instance of for loop.

Now we are ANDing every inputs with every contraints of P1 and give it to solver. In whichever instance, it satisfies then we are taking output equation of P2 corresponding to input given and output equation of P1 corresponding to given Constraint which satisfies solver.

Now we are equating right side of both equations satisfying solver and add equations to solver instance.

At last, if solver satisfies given equations then we will get values of constants.

Remember I am using s.add() method not s.addConstraint() method.

s.add() takes non string parameters and s.addConstraint() takes only strings.

Assumptions:-

1. Input variables should be x,y,z,k only because while initializing variables in Solver() method it returns value of Int instance

```
y = Int('y')
x = Int('x')
k = Int('k')
z = Int('z')
```

   So it is not possible to take dynamically return values of it.

2. Constants cannot be in If conditions part.
3. Structure of both Programs should be similar.
4. There is limit in number of input variables and constant variables.
5. Using previous version of Kachua Framework [one with kw file]

Limitations:-

1. Similar structure of both programs
2. Limit in number of input and constant variables
3. For programs with different structure it might not give correct output
4. It might not work for if Constant variables is in condition part.
5. Only 5 number of input variables named x,y,z,k
6. Only 10 number of Constant variables from c1 to c10.