CSU44099 Final Year Project

Interim Report

Name: Kai Suzuki

1. Description of the project

Discovering violated properties in a program is a method to check the verification of the program execution. Formal verification tools are used to enable human to see how correct the program execution is by examining if there is any error or false positive execution in the program.

In this project, the goal is to visualize, for human, the correctness of the program execution with the formal verification tools.

This project uses Hobbit as a developing programming language which is created by Vasileious Koutavas who is my supervisor.

“Hobbit (Higher-Order Bounded Bisimulation Tool) is a higher-order contextual equivalence checking tool that combines Bounded Model Checking and Up-To Techniques to verify equivalences and find inequivalences.”

In other words, my program can examine if the two input code fragments in Hobbit are equivalent or inequivalent

In terms of visualization, it should be web application or web page as front-end for Hobbit.

The web page must be hosted in the School, using Virtual Machine (VM) with a domain name. Also, it is going to work with HTML and JavaScript for UI in front-end while it runs Hobbit in back-end with the user inputs.

Regarding user inputs, as discussed previously, they have to be the two input code fragments in Hobbit and user can enter them with some command lines to let it run with advanced multiple types of options.

When it comes to outputs of the visualization, it just shows the indication of inequivalence when the result is equivalent. On the other hand, if it results in the inequivalence, it has to illustrates how each of them are different in text, graphical representation or constructing a distinguishing program, for human.

Furthermore, it needs to have a set of test programs user can attempt to include the current test cases, expand the running program easily and organizing the test and examples.

Finally, it is necessary to consider the security of the webpage. One of the possible ways for protect can be Denial of service attack (DoS) and distributed DoS attack.

Also, the previous student, whom Vasileious Koutavas supervised, worked for visualization of program execution with another programming language as well, instead of using Hobbit. Therefore, my project is going to use and improve the available parts of his work for UI in the front-end.

1. Review of literature

* *From Bounded Checking to Verification of Equivalence via Symbolic Up-to Technique*/ Vasileious Koutavas, Yu-Yang Lin, Nikos Tzevelekos

This is the paper on Hobbit and it explains the concept and mechanics on it. Also, in this paper, there is a mention on the command line options when entering inputs by user.

* Hobbit: A tool for Contextual Equivalence Checking Using Bisimulation Up-to Techniques

This is the presentation slides on Hobbit at International Conference on Functional Programming(ICFP) in 2021.

* *Readme.md* (Hobbit)

This Readme file mentions how to install Hobbit on each laptop for Windows, Mac OS and Linux. There are several example commands to run in both equivalent and inequivalent cases.

* Visualization framework for verification tools with graph output/ Tianyi Zhang

This is the previous year student’s report supervised by Vasileious Koutavas.

This paper mentions visualization of program execution with another programming language instead of using Hobbit. Furthermore, the employed technology for this project, especially UI is stated.

* Readme.md (by Tianyi Zhang)

There are instructions on how to run the previous year student’s program. It also shows what things to be installed.

1. Plan of work for the completion of the project

Tianyi Zhang, the previous year student, generated three prototypes until reaching to the final version. In my project, there will be several prototypes as well, improving them, based on feedbacks from Vasileious Koutavas each time.

The other things to do in this project are to read papers, run Tianyi Zhang’s work, implement back-end and VM with domain and finally write a paper.

These tasks are divided into Gantt chart like below:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Week1(24/1~) | Week2(31/1~) | Week3(7/2~) | Week4(14/2~) | Week5(21/2~) | Week6(28/2~) | RWeek(7/3~) | Week7(14/3~) | Week8(21/3~) | Week9(28/3~) | Week10(4/4~) | Week11(11/4~) |
| Read Papers |  |  |  |  |  |  |  |  |  |  |  |  |
| Run last year student's work |  |  |  |  |  |  |  |  |  |  |  |  |
| Implement Back-end for Hobbit |  |  |  |  |  |  |  |  |  |  |  |  |
| 1st UI prototype |  |  |  |  |  |  |  |  |  |  |  |  |
| 2nd UI prototype |  |  |  |  |  |  |  |  |  |  |  |  |
| 3rd UI prototype |  |  |  |  |  |  |  |  |  |  |  |  |
| Implement VM with domain |  |  |  |  |  |  |  |  |  |  |  |  |
| Write a paper |  |  |  |  |  |  |  |  |  |  |  |  |

In this chart, there is no task during reading week (7/3~) and week11(11/4~) because if something goes wrong against as planned, those tasks has to be done during the spare weeks (reading week and week11).

1. Broad review of ethical issues

This project does not significantly involve gender and race because this project is visualizing the program execution by outputting the results on the webpage. There is