CSU44099 Final Year Project
Interim Report

Kai Suzuki, 18308704

21th January, 2022

Declaration

"I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at http://www.tcd.ie/calendar.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at http://tcd-ie.libguides.com/plagiarism/ready-steady-write."

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 humans 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 humans, the correctness of the program execution with the formal verification tools.

    For humans, the results of executing the program are difficult to see. Usually, it is necessary to read the output text which shows the result. However, it sometimes outputs long and/or complex texts and is difficult even to read.
    Therefore, visualizing the result in graphical representation is much easier to understand the meaning of the output and people who do not know the program well can figure it out. These are the beneficial points of this project.

    This project uses Hobbit as a developing programming language 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."[1]
    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 a web application or web page as the 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 the front-end while it runs Hobbit in the back-end with the user inputs.
    Regarding user inputs, as discussed previously, they have to be the two input code fragments in Hobbit and users can enter them with some command lines to let it run with multiple advanced types of options.
    After dealing with program executions at the back-end, it just shows the indication of inequivalence when the result is equivalent. On the other hand, if it results in inequivalence, it has to illustrate how each of them is different in text, a graphical representation or constructing a distinguishing program for humans.
    Furthermore, it needs to have a set of test programs the user can attempt to include the current test cases, expand the running program easily and organize the test and examples.

Moreover, it is necessary to consider the security of the webpage. One of the possible protection methods can be Denial of service attack (DoS) and distributed DoS attack.

Finally, 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.

2. 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 explains the concept and mechanics. Also, in this paper, there is a mention on the command line options when entering inputs by users.[1]

    - 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.[2]

    - 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.[3]

    - 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.[4]

- Readme.md (by Tianyi Zhang)

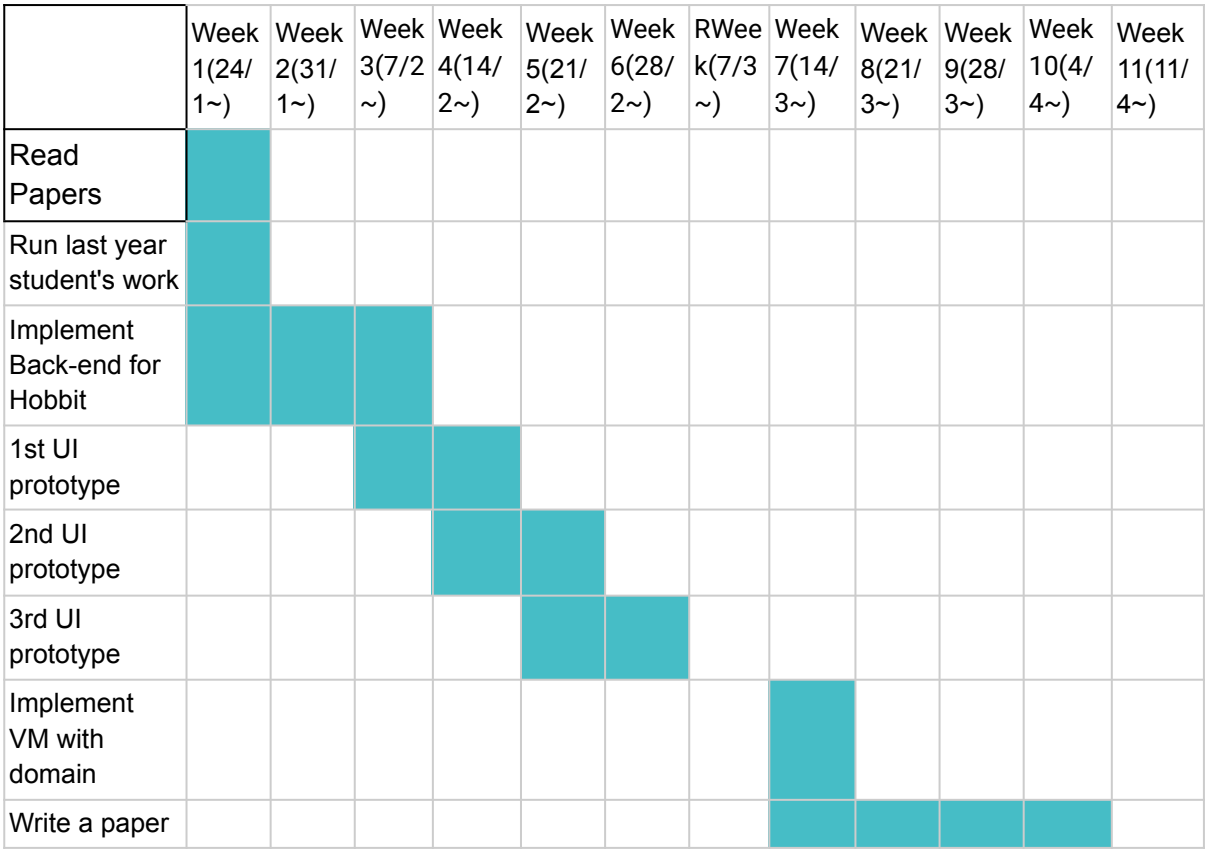
  There are instructions on how to run the previous year student's program. It also shows what is to be installed.[5]

3. 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:



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 have to be done during the spare weeks (reading week and week11).

4. Broad review of ethical issues

   This project does not significantly involve gender and race because this project visualizes the program execution by outputting the results in the webpage. There are no ethical issues in examining the similarities of the two Hobbit inputs.
   The UI design is the only place where ethical problems can be concerned. The website should be fairly easy to use for every user which includes any gender, race and disabled people. One of the possible topics to be considered is the colors in the webpage are so fairly clear to see for every user, even color-blindness, that those colors have to be we-thought and selected.

<Reference>

[1] From Bounded Checking to Verification of Equivalence via Symbolic Up-to Technique/ Vasileious Koutavas, Yu-Yang Lin, Nikos Tzevelekos/ 6 May 2021/ https://arxiv.org/abs/2105.02541

[2] Hobbit: A tool for Contextual Equivalence Checking Using Bisimulation Up-to Techniques/ Vasileious Koutavas, Yu-Yang Lin, Nikos Tzevelekos/ 26 August 2021/ https://icfp21.sigplan.org/details/mlfamilyworkshop-2021-papers/1/Hobbit-A-Tool-for-Contextual-Equivalence-Checking-Using-Bisimulation-Up-to-Technique

[3] Readme.md (Hobbit)

[4] Visualization framework for verification tools with graph output/ Tianyi Zhang/ April 2021

[5] Readme.md (by Tianyi Zhang)