COM S 413/513 Final Project

Learning Objectives

1. Teamwork and collaboration
2. Connect program analysis and software engineering knowledge learned throughout the semester
3. Gain experience of the LLVM analysis platform
4. Learn to evaluate and improve real-world program analysis tools

Description

In the final project, you will evaluate, improve and extend a real-world program analysis tool, *Hydrogen* https://github.com/iowastateuniversity-programanalysis/hydrogen. As the first two steps: (1) Read the paper "MVICFG: Patch Verification via Multiversion Interprocedural Control Flow Graphs" (2) Get familiar with the source code and tool

413 students will form 3-people teams and 513 students will form 2-people teams. I have provided a few ideas below for your references. You are welcomed to design your own projects that leverage all the program analysis techniques you learned throughout the semester (need instructor's approval).

Project ideas:

1. **Evaluating Hydrogen**: In this project, you will learn how to test and evaluate a program analysis tool. Working in the software quality assurance department, sometimes your job is to look for a tool that can work for your purpose in your working environment. What are some good steps to follow for such tasks? In this project, we will use MVICFG, a tool for analyzing software changes and versions as a case study to learn how to evaluate a real-world program analysis tool:
    a. Read the paper and learn how to evaluate a program analysis tool: An Evaluation of Daikon: A Dynamic Invariant Detector
    b. Find a set of real-world benchmark programs (2 benchmarks per person)
    c. Test benchmarks with Hydrogen and document any issues you find in the Hydrogen code. It can be crash, hang, incorrect MVICFG, feature enhancement needs or refactoring needs; Note that purely inspecting the results for a

real-world project can be hard as the graph is large, so I highly recommend you to also write some code to help tests e.g., you can print CFGs for each version and see if the nodes are correct; and you can print out the nodes that are different from one version to another to see if the results match the diffs you computed using Unix diff tools.

   d. Fix the bugs and make the improvement you plan and send in pull requests (for bugs, please include a test to demonstrate the bug in the unpatched version).

   e. Report statistics for your studies: the size of code churns, size of benchmark, the size of MVICFG (nodes), the time used to generate MVICFG, the failure inducing test cases

2. **Building a Web/GUI demo for Hydrogen**: In this project, you will learn how to present the program analysis results to the end users and how to convert a program analysis tool to a web or GUI based demo. The interface will allow users to upload a few versions of software and a program location, and then display (1) the statistics of MVICFG (2) visualizations of certain part of MVICFG based on the location information (3) anything else you think it's important

3. **Implementing one program analysis algorithm on MVICFG:** In this project, you will learn the challenges of getting an analysis working properly for real-world programs. There are a few options:

   a. Perform dataflow analysis or alias analysis on the first version of ICFG and use MVICFG to determine which reaching definitions and aliases will be affected by the changes, creating a test case with 3+ versions so you can see the history of change impact; or

   b. Integrate your delta debugging done in project 3 with MVICFG

Deliverable and Grading Criteria (40 pt)

1. Report project and team sign up: **3/26 11:59 pm**
   https://docs.google.com/document/d/1JSs2oqTv99PkTZQ7TbY4DW9-FMXLgGPL7Qy7Zn6VT14/edit?usp=sharing

2. (6 pt) Midpoint status writeup (one page): **4/17 11:59 pm (upload to canvas)**

   a. What you have done

   b. What you plan to do (timeline and task allocations among the team members)

   (2 pt) Clarity of the documentation, are we able to understand what you mean?

   (4 pt) Has your team made sufficient progress? 4 yes, 2, somewhat, 0 no

3. (10 pt) Final presentation and demo **May 7 (Thur) 2:15pm** in classroom

   a. (2 pt) Can we understand most of your presentation?

   b. (2 pt) Does your presentation include all the important aspects of the work?
   c. (6 pt) Do you have a workable demo to showcase your improvements for Hydrogen?

4. (24 pt) Submit all the products of the project, please zip the following: **May 7 (Thur) 11:59 pm** on canvas
   a. (10 pt) The articrafts, please zip the following -- source code of your tool/analysis, bug reports, test cases, readme, demo, intermediate results if any
      i. are the artifacts correct, clear, runnable, reproducible? Does it contain a clear readme file?
   b. The presentation slides and demo
   c. (14 pt) A report that explains the following:
      i. (5 pt) What is the impact of your work? Have you made a visible impact on the Hydrogen tool? In which aspects the tool is better than before? How much work each member has done? Is the amount of the work sufficient?
      ii. (3 pt) The design diagram, workflow: correct, clear and interesting?
      iii. (3 pt) What are the results and findings? Using figures and tables.
      iv. (3 pt) What are your experiences with the Hydrogen tools? What are the possible future works?