COM S 413x/513x Project 4: Analyzing software changes and versions

Learning Objectives

1. Gain experience with LLVM and MVIFG, the C/C++ program analysis tools
2. Understand the challenges of analyzing program changes and versions
3. Develop extensions for and contribute to existing program analysis tools

Description

In this homework, you are going to work with one of the prominent program analysis platforms llvm https://llvm.org/ and one of the frontier static analysis tools, called multi-version control flow graphs MVICFG https://dl.acm.org/citation.cfm?id=2568304. Here are the set of instructions that can guide you through the work.

1. Create a gitlab account and get access to MVICFG repo.
2. Follow the manual of the MVICFG tool, install the docker and get the MVICFG running with the *test* program.
3. (20 pt) Extending the MVICFG framework and write two functions path_added and paths_removed to report newly added acyclic paths and newly removed acyclic paths for a given code churn.
4. (10 pt) Develop two test cases to demonstrate the output of your code and explain what you are planning to test and why your code works correctly
5. (10 pt) Write a summary report to discuss what are the challenges you face for this projects, any interesting findings? The report should also include the following table that summarize the three tests you have run.
6. **(Extra Credit)** (10 pt +) Try 3 versions of a real-word program and add the results in the table. In addition, you get 1 pt for any patches you submit that get accepted. You get 2 pt if the patch is complex.

Version pairs	Code churn size	MVICFG size (nodes, edges)	Paths added	Paths removed	time used
Test: v1-v2					
Test v2-v3					
Your test program 1					
Your test program 2					

Deliverables and Grading Criteria

Please zip the following files and submit the zipped file to canvas under the “project 4: analyzing changes and versions” column. **The project is due April 8.**

(1) Step 3 (20 pt): source code of the two functions you implemented. Also submit pull requests to the MVICFG repo.

(2) Step 4 (10 pt): source code of the test cases. Output of the results. A test plan that explains what you plan to test and what are the outcomes. Submit the test cases to the MVICFG repo.

(3) Step 5 (10 pt): a summary report.

(4) Step 6 (10pt +): source code of the program versions you tried (or send in the versions and link of the software if it is open source) and output generated for the programs. Patches and pull requests for MVICFG you plan to fix the framework.