COM S 413x/513x 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. You will work with two other students in a team to accomplish the project together. The 413x students will work on an implementation oriented project, and the 513x students will work on a research oriented project. The source code of the tool is located on gitlab @ Iowa State University: https://git.linux.iastate.edu/users/sign_in. Please apply an account and I will add you to the project. Please see the following further details.

413x: implementation project

In this project, you will learn how to evaluate and improve a program analysis tool. Working for software quality assurance, sometimes your job is to look for a tool that can work for your purpose in your working environment. What are the 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 and improve it when it is not perfect. Here are the set of instructions you should follow:

1. Read the paper and learn how to evaluate a program analysis tool: An Evaluation of Daikon: A Dynamic Invariant Detector
2. Read the paper and learn what is MVICFG: Patch Verification via Multiversion Interprocedural Control Flow Graphs
3. Initial studies on the source code and tool
4. **April 5 (Fri) 3pm office hour**: initial ideas on how to evaluate the MVICFG, Q&A
5. Perform further studies on the tool and propose a comprehensive evaluation plan.
6. **April 22 (Mon) inclass (20 min)**: Present your evaluation plan, addressing questions such as:

    a. What are your goals for this evaluation?
    b. What benchmarks you plan to use?
    c. What is your plan in terms of people and time allocation?
7. Conduct the detailed evaluation, improve the tool and write up your study: the evaluation report should summarize
    a. What is the current state of the tool? List some tables on the results collected from your benchmarks
    b. What are the problems you found and how you found them?
    c. What are the improvements you made for the repo? Including code, documentation, tests?
    d. What are the improvements you want to make but yet have enough time to make them?
8. **May 7 (Tue) 2:15pm (30 min)** Present your final findings, and a demo show how the MVICFG is improved
9. **May 8 (Wed) 6:00pm** Submit the final package (.zip file) to canvas

513x: research project

In this project, you will use the LLVM and MVICFG platform to design and implement an analysis. The analysis should help other tools, e.g., fuzzing or delta-debugging, or to automate some software engineering tasks that currently only can be done manually, e.g., find the origin of the bugs. Construct a study to show your analysis are correct and useful. You can find existing software subjects or create a program yourself. Here are the set of instructions you should follow:

1. Read the paper to understand the MVICFG work: Patch Verification via Multiversion Interprocedural Control Flow Graphs
2. **April 5 (Fri) 3pm**: meet for proposal and Q&A
    a. What is the analysis?
    b. What is the application?
    c. What is your plan in terms of people and time allocation?
3. **April 22 (Mon) inclass (20 min)**: midpoint check: report the current progress of the project
4. **May 7 (Tue) 2:15pm (30 min)**: demo and final presentation
5. **May 8 (Wed) noon**: submit the final package (.zip file) to canvas

Deliverable and Grading Criteria

413x: implementation project

The evaluation of the implementation project is based on the following criteria: 50 points

1. The nature of the work (20 pt):
    a. Correctness (10 pt):

        i. Are the benchmark and evaluation suitable for the current MVICFG framework?
        ii. The bugs and issues found are real, important problems?
        iii. Is the improvement correct? Can the pull requests be accepted?
    b. Completeness (10 pt):
        i. Is the evaluation comprehensive and including all the important aspects?
        ii. Are the benchmark used representative and sufficient?
2. Evaluation report (4 pages including the references, acm format) (10 pt):
    a. Clarity (3 pt): Is the paper understandable? Does the paper follow a good structure? Does the paper contain grammatical mistakes?
    b. Insightfulness : Does the report include interesting perspectives on what have been improved and what can be further improved?
3. The articrafts (15 pt):
    a. Benchmarks
    b. Bug reports
    c. Patches and pull requests, any other tools and scripts developed or used
4. Demo and presentations (10 pt)
    a. Midpoint check (5 pt): Is the proposal well-thought and comprehensive? Is the presentation clear?
    b. Final demo and presentation (5 pt): Is the presentation clear? Do you have a workable demo to show what is/are the problems of the tool and how to improve it?

Please upload a zip file to Canvas under “final project”, including the following files:

1. The paper, both latex and pdf
2. The articrafts: but reports, patch and pull requests, test cases, readme, demo, intermediate results if any
3. The presentation slides (both midpoint check and final)

513x: research project

The evaluation of the research project is based on the following criteria: 50 points

1. The nature of the work (15 pt):
    - Novelty (5 pt): is this a novel analysis or novel application or both?
    - Correctness (5 pt): is the idea and implementation correct?
    - Completeness (5 pt): do you contain sufficient work?

2. The paper (6 pages including the references, acm format) (10 pt):
    - Clarity (3 pt): Is the paper clearly written? Does the paper have a good structure? Does the paper have any grammar mistakes?
    - Insightfulness (3 pt): Is the paper insightful and interesting to read?

- Completeness (4 pt): Is the study and implementation reproducible based on the paper? Do you include all the related work?

3. Articrafts (15 pt):
    - Is your demo runnable?
    - Is your implementation buggy?
    - Is your implementation understandable and containing comments?
    - Have you submitted all the data and code?
    - Is your documentation completed and easy to navigate?

4. Demo and presentations (10 pt):
    a. Midpoint check (5 pt): have you made progresses on your work? Is the presentation clear?
    b. Final demo and presentation (5 pt): Is the presentation clear? Do you have a workable demo?

Please upload a zip file to Canvas under “final project”, including the following files:

4. The paper, both latex and pdf
5. The articrafts: code, test cases, readme, demo, intermediate results if any
6. The presentation slides (both midpoint check and final)

Extra credit: every accepted patches and APIs will get an extra credit. The concrete score is dependent on the actual patches and APIs, and will start at 1 pt (a small patch) and scale up to 10 pt (an API). For the pull requests to be accepted, they need to be correct and well tested beyond the examples you use in the demo.