HW5 and Github

CS 4278/5278: Principles of Software Engineering

Skyler Grandel
Graduate Teaching Assistant
skyler.h.grandel@vanderbilt.edu

Delta Debugging Review

From previous lecture slides:

- Delta debugging is an **automated debugging approach** that finds a one-minimal **interesting subset** of a given set.
- Delta debugging is based on **divide and conquer** and relies on critical **assumptions** (monotonicity, unambiguity, and consistency).
- It can be used to find which code changes cause a bug, to minimize failure inducing inputs, and even to find harmful thread schedules.

Delta Debugging Review

Remember the three main assumptions around Delta Debugging…

- Monotonicity - if X is interesting, set of X & anything is interesting
- Unambiguity - if X & Y are interesting, intersection of X & Y is interesting
- Consistency - X is either interesting or not interesting

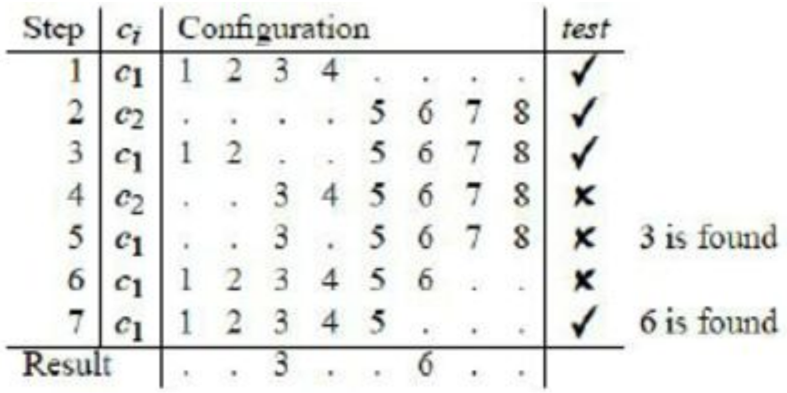
And the problems that delta debugging seeks to solve are simplifying, isolating, and identifying failure-inducing components

Homework 5

- Four parts
    - 5a - delta.py file
    - 5b - is-failure-inducing-change, minimizing failure-inducing changes
    - 5c - report, minimizing test suites
    - 5d - faultloc.py file, coverage based fault localization

Homework 5a

- delta.py implements the basic divide-and conquer algorithm discussed in class, seen below:



Step	c_i	Configuration								*test*	
1	c_1	1	2	3	4	.	.	.	.	✓	
2	c_2	.	.	.	.	5	6	7	8	✓	
3	c_1	1	2	.	.	5	6	7	8	✓	
4	c_2	.	.	3	4	5	6	7	8	✗	
5	c_1	.	.	3	.	5	6	7	8	✗	3 is found
6	c_1	1	2	3	4	5	6	.	.	✗	
7	c_1	1	2	3	4	5	.	.	.	✓	6 is found
Result		.	.	3	.	.	6	.	.		

- os and sys libraries are important here

Homework 5b

- Writing a simple bash script to apply given patches into a file and check whether it still compiles with GCC
- File that patches are applied to should be returned to its original state
- This functions acts as a replacement of the “is-interesting.sh” component of part a - it only checks if the given patches cause a compilation error or not

Homework 5c

- Find one-minimal test suite, given the 1639-image test cases on the libpng
- Highlights from report guidelines…
  - What constituted “interesting”? How did you implement it?
  - Were you able to use delta debugging to find a one-minimal subset of the test suite with same coverage?
- Start early!! More information about the timing of this part can be found on the assignment page

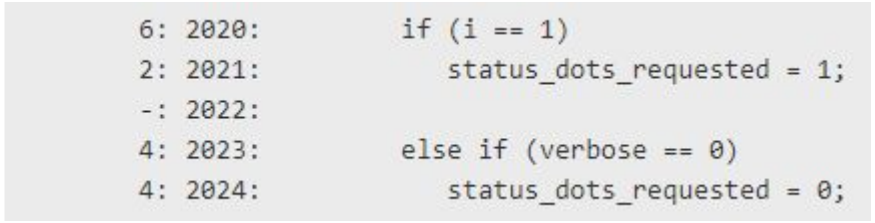
Homework 5d

- Using the *Ochiai Suspiciousness Score* to find the 100 most suspicious lines
- Ochiai formula (can be found in linked IEEE paper, right after table 3):

$$Suspiciousness(Ochiai) = \frac{N_{CF}}{\sqrt{N_F \times (N_{CF} + N_{CS})}}.$$

Homework 5d

- Small but important details listed on homework page
- Idea is to take provided command line arguments and find the 100 most suspicious lines



Visits Line no. Source code

```
        6:  2020:        if (i == 1)
        2:  2021:           status_dots_requested = 1;
        -:  2022:
        4:  2023:        else if (verbose == 0)
        4:  2024:           status_dots_requested = 0;
```

- Lots of whitespace visible - strip, rstrip, split can all be helpful

Homework 5d

- Once computed, sort and print the top 100 most suspicious lines as pairs (ex: (5, 0.75) for line 5 having a score of .75)
- If < 100 pairs, then print all pairs, if more, only print the top 100
- Do not print using iteration - use print function on the list of pairs (but ensure only 100 get printed)

Open Source Contribution (HW6 for Undergrad)

- Goals:
    - Engage with software engineering
    - Make a meaningful contribution (Pull Request)
    - Reflect on the process and results (Project Report)
- Logistics Overview:
    - 2-student teams are allowed (w/o mixing undergrad and grad)
        - Higher expectations for contribution and project report
    - HW6(A): Task Selection Report (due on 04/02/23)
        - See https://huang.isis.vanderbilt.edu/cs4278/oss4sg.html for ideas
        - Feel free to look around and find a project that you resonate with
    - HW6(B): Project Report (due on 04/18/23)
        - See examples on course website
        - +6% bonus points on HW6(B) if your pull request(s) is/are accepted!

How to make an open source contribution?

- Where to find open source projects?
  - GitHub!!!
- How to spot a good project?
  - Hang on… (next couple of slides)
- How to actually contribute?
  - Pull Requests! (lots of tutorials online)
    - Fork the repository
    - Clone the repository to local machine (git clone)
    - Create a new branch (git checkout -b [branch-name])
    - Make the changes
    - Commit the changes (git commit)
    - Push the changes (git push)
    - Create a pull request on GitHub UI

Task Selection

- Find an ***active*** project that is meaningful to you!!
- Where to start?
    - GitHub trending repositories
        - https://github.com/trending?since=monthly
        - Lots of OpenAI-related or ChatGPT-related repositories right now!!
        - Generally very active and fast paced
    - Third Party Monthly Picks
        - https://star-history.com/blog/star-history-monthly-pick-202302
    - Popular Projects (generally very well maintained)
        - Raspberry Pi Projects (https://github.com/raspberrypi)
        - Hyperledger Foundation Projects (https://github.com/orgs/hyperledger)
        - Kubernetes Projects (https://github.com/kubernetes)
        - Google Project, Microsoft Project, etc.

Task Selection Cont.

- Python Projects:
  - TensorFlow
  - OpenCV
  - Flask
- C++ Projects:
  - Microsoft Cognitive Toolkit
  - IncludeOS
  - Kodi
- Java Projects
  - Jenkins
  - Elasticsearch
- Lots of online articles/blogs that can guide you to finding a good project!
  - e.x. https://www.rocket.chat/blog/open-source-projects

Task Selection Advice

- Choose an ***<u>active</u>*** project with many contributors!
- Scope the project well (don’t get overly ambitious)
- Choose one large task or several smaller tasks
- Read the entire homework description on the course website!!!
- Once you identify a task to do, claim it!
  - Especially important for well-maintained projects
  - Someone may already be on the task!
- Create a timeline (both for yourself and for the report)
  - Try to stick to it!
- Start early!!
  - Especially if you want your pull request to get accepted before the deadline.
  - (only PRs accepted before the deadline will get extra-credit)

Project Report Overview

- NO LATE SUBMISSION for HW6(B)

- NO excuse will be accepted!

Project Report Overview

- Show us what you did!
  - Be proud of your contribution!!
- Explain your strategy/approach
- Share your engineering experience
  - What issues/roadblocks did you encounter?
  - How was communication with other community members?
  - How did you fix the bug or make an improvement?
  - Show some evidence of your work :)
- Compare your initial plan to what you have achieved
  - Any differences?
- Many examples on course website!!

Final Remarks

We hope you can have some fun with open source contribution, as it is a vital component of the software engineering community. Maybe you’ll become a regular open source contributor in the future!

Research Proposal (HW6 for Grad)

- Format based on the NSF requirements
- 5-7 pages excluding references
- Use the provided LaTeX template

Research Proposal (HW6 for Grad)

- First Page: Project, Summary, Intellectual Merit, Impact
- Introduction
- Background and Related Work
- Proposed Research
- Proposed Experiments
- Preliminary Work
- Conclusion
- References (IEEE format)