COM S 413x/513x Project 3: building an automatic debugger

Learning Objectives

1. Strengthen the understandings of *delta-debugging*
2. Gain experiences of implementing an automatic program analysis tool
3. Understand the challenges of implementing such tools
4. Gain experiences of reproducing work in the research paper

Description

In this homework, we are going to implement an automatic debugging tool based on the delta-debugging technique. Given a buggy version and a correct version (an old release and a new release) with a list of changes in between, delta-debugging systematically search for the changes to identify in which change (s), the bug is introduced. Please see the following guidelines to proceed your project:

(1) Read the paper: “Yesterday, my program worked. Today, it does not. Why?”
(2) Implement the basic delta-debugging technique specified under Algorithm 1 in the paper. You can use C/C++, Java or Python.
(3) Creating a test case, consisting of a successful version and a failure version. There are at least 8 changes between the two versions (note that Algorithm 1 only works with changes that do not have *inconsistency* problems, so you should create changes to satisfy the conditions).
(4) Performing a simple study using your tool and your test case
(5) What have you found? documenting your thoughts, findings and insights
(6) **(extra credit)** in your dbg benchmark, find6 does not trigger the bug in find14, can your tool automatically find which changes between find6 and find14 introduce this bug? (to support real-world case study like this, you will need to extend your tool with Algorithm 2 in the paper)

Some more tips:
You can use “diff -u file1 file2” to create the patch
You can use “patch < my patch -o output” to patch the old version and generate likely buggy versions

Deliverables (30 pt / 45 pt with extra credit)

Please zip the following files and submit the zipped file to canvas under the “project 3: building an automatic debugger” column

(1) (15 pt) From step 2: please submit the source code as well as the binary of your delta-debugging tool. Please provide details in a readme file to explain how I should build and run your tool. If I fail to build your tool due to unclear instructions, your points will be reduced. If your tool can work with your test case, you get full points. Otherwise, I will inspect the source code to give you partial credits.

(2) (5 pt) From step 3: please submit your test case including two versions and a readme file to describe where is the bug, what are the changes, and how I should test your programs.

(3) (10 pt) From steps 4 and 5:
   (a) Please submit all the intermediate/interesting results you generated when running your debugging tool against your test case. (5 pt)
   (b) Please submit a summarization table like Table 1 in the paper. (2 pt)
   (c) Please write up your thoughts and findings from the studies. (3 pt)

(4) **(extra credit)** (15 pt) From step 6, please provide a reproducible package to show how to run your tool on find6 and find14 to find the buggy changes, and also submit your intermediate results (12 pt). You will also need to submit a short report on what you learn from this study (3 pt).

The homework is due Mar 20, Wed 6:00pm. You can continuously submit your homework without a penalty after the deadline. But once I started grading the homework, I am no longer accepting late homework or modifications. Start early!