Lab Assignment 4

Course: CS202 Software Tools and Techniques for CSE
Lab Topic: Exploration of different diff algorithms on Open-Source Repositories
Date: 25th August 2025

Objective

The purpose of this lab is to explore differences in diff outputs for Open-Source Repositories in the wild.

Learning Outcomes

By the end of this lab, students will be able to:

- ✓ Analyze diff output due to variants of the diff algorithm applied in the wild.
- ✓ Analyze the impact of different diff algorithms on code versus non-code artifacts.

Pre-Lab Requirements

- Any Operating System (Windows, Linux, MacOS, etc.)
- Python 3.10 or later
- Software Tool Setup, Code Infrastructure Preparation, and Familiarity:
  - ➢ Install, and configure **`pydriller`** (https://github.com/ishepard/pydriller) for experiments, and software analyses.
  - ➢ Read the tool documentation: (https://pydriller.readthedocs.io/en/latest).

Lab Activities

(a) **Repository Selection**:

- Choose **THREE** medium-to-large scale open-source repositories to analyze with **`pydriller`**. Make sure these are **real-world** projects (like butterknife or larger) and not toy projects on GitHub. You must not reuse the same repositories that you selected in your previous assignments.

(b) **Define Selection Criteria**:

- Establish your own criteria for selection (inclusion/exclusion) of repositories and include this information in your report. Basically, you need to specify how you settled with the final set of repositories. Recall the hierarchical funnel diagram from the slides from Lecture 2.
- Examples of selection criteria may include metrics such as the number of GitHub stars, forks, etc.
- You may use the SEART GitHub Search Engine to perform this task.

(c) **Run Software Tool on the Selected Repository**:

Note: Please reach out to the TAs for any queries/issues.

- Execute `pydriller` on the selected repositories to generate a consolidated dataset (.csv). **For a *modified* file in a commit of a repository,** store the following information (in plain text).

old_file path	new_file path	commit SHA	parent commit SHA	commit message	diff_myers[1]	diff_hist[2]
…	…	…	…	…	…	…
…	…	…	…	…	…	…

Note: Ignore whitespace when comparing lines. This will ignore differences even if one line has whitespace where the other line has none. Also, ignore blank lines.

(d) **Compare Diff Outputs for Discrepancy Analysis**:

- In your stored dataset (.csv), determine[3] whether the diff outputs match by selecting {**Yes/No**}. Include this label in a **new column** (final dataset):

old_file path	new_file path	commit SHA	parent commit SHA	commit message	diff_myers	diff_hist	Discrepancy
…	…	…	…	…	…	…	Yes
…	…	…	…	…	…	…	No

(e) **Report Final Dataset Statistics with Plots Generated Using Python Code**:

- #Mismatches for Source Code files
- #Mismatches for Test Code files[4]
- #Mismatches for README files
- #Mismatches for LICENSE files

(f) **If you were asked to automatically find which algorithm performed better, how would you proceed? Explain.**

Resources

- Lecture 4 slides
- https://git-scm.com/docs/git-diff
- https://github.com/ishepard/pydriller
- https://doi.org/10.1007/s10664-019-09772-z
- https://github.com/yusufsn/DifferentDiffAlgorithms
- SEART GitHub Search Engine

[1] git diff due to the Myers algorithm.
[2] git diff due to the Histogram algorithm.
[3] The way to perform this is left as an exercise.
[4] Selected repositories must have them.

Note: Please reach out to the TAs for any queries/issues.