Bonus: Detecting Bug-Introducing Changes (BICs)

1. Introduction

In software engineering, a bug-introducing change (BIC) refers to the earlier commit that originally caused a bug that is later fixed. Identifying BICs is valuable because it helps developers understand how bugs are created, which parts of the code are risky, and how software reliability can be improved. In this bonus section, I explain how my line-mapping tool can be extended to support the detection of bug-introducing changes.

2. Concept

Bug fixes are usually easy to identify from commit messages. Developers often include keywords such as: fix, bug, error, issue, crash, null pointer, resolve. After detecting a bug-fix commit, we can trace backwards to find which earlier commit last modified the buggy lines. This general idea follows the well-known SZZ algorithm, commonly used in research to identify BICs.

3. How My Tool Supports BIC Detection

My tool already generates line-mapping information between versions of a file. For example:

10 -> 12

11 -> 11

12 -> (deleted)

This mapping tells us exactly how each line in one version corresponds to the next version. Using this output, the tool can help detect BICs through the following steps:

Step A — Detect a Bug-Fix Commit

A commit is labeled as a bug-fix commit if its message contains bug-related keywords.

Step B — Identify the Lines Modified by the Fix

The tool compares the previous version with the fixed version and identifies which lines were changed, added, or deleted.

Step C — Trace These Lines Backwards

Using the line-mapping information, the tool can determine which earlier commit last touched the changed lines. That earlier commit is then flagged as the bug-introducing commit. No additional implementation is required—the current mapping logic already provides the necessary information.

4. Example Workflow

Assume Commit B is identified as a bug-fix commit. Commit B modifies lines 20–25. Using the mapping, the tool traces these lines back to lines 18–23 in Commit A. Therefore, Commit A is marked as the likely bug-introducing change.

5. Benefits

Integrating BIC detection into the tool provides several advantages:

- Helps identify risky files or code sections that frequently introduce bugs.

- Reveals patterns that lead to defects, improving long-term code quality.

- Provides clear traceability between historical changes and their effects.

- Supports better project management and prioritization of testing resources.