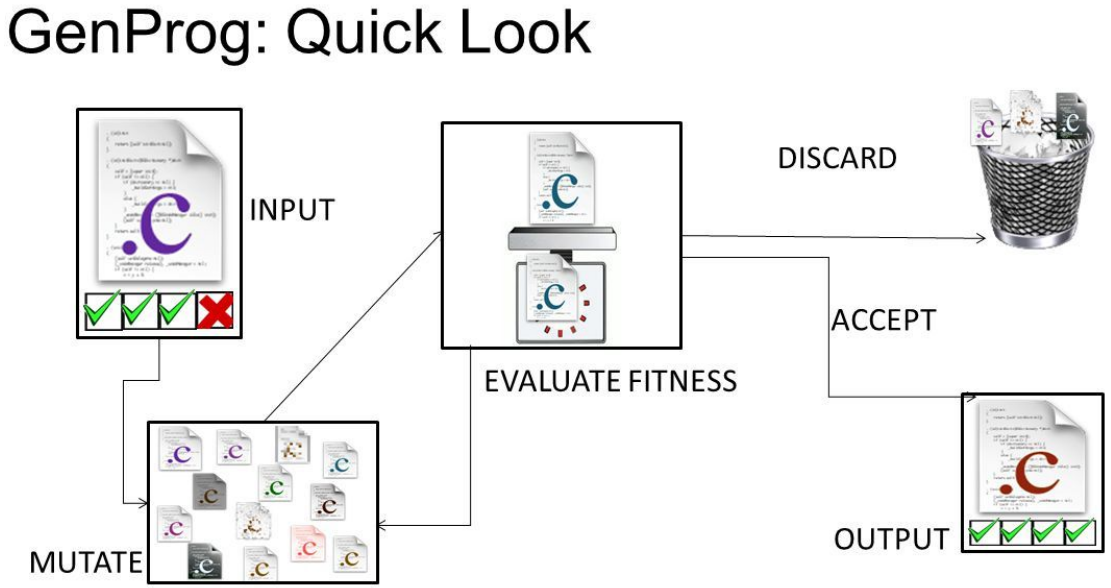
TOOL DEMO
KGenProg

Melisa Panaccione

Genetic Programming



OVERVIEW

TYPE OF TOOL

- Genetic programming
- Successor of GenProg and jGenProg

ALGORITHMS

- Ochiai (fault localization)
- Genetic algorithm (create modified versions)

USE

- Repair bugs given a faulty code and a set of test cases

WHY I CHOSE THE TOOL

- Curious about the use of GenProg
- Clear setup instructions

Example

```
17  public int close_to_zero(int n) {
18    if (n == 0) {
19      n++; // bug here
20    } else if (n > 0) {
21      n--;
22    } else {
23      n++;
24    }
25    return n;
26  }
```

(a) Faulty program

```
 7  @Test
 8  public void test01() { // passed
 9    assertEquals(9, new CloseToZero().close_to_zero(10));
10  }
11  @Test
12  public void test02() { // passed
13    assertEquals(99, new CloseToZero().close_to_zero(100));
14  }
15  @Test
16  public void test03() { // failed
17    assertEquals(0, new CloseToZero().close_to_zero(0));
18  }
19  @Test
20  public void test04() { // passed
21    assertEquals(-9, new CloseToZero().close_to_zero(-10));
22  }
```

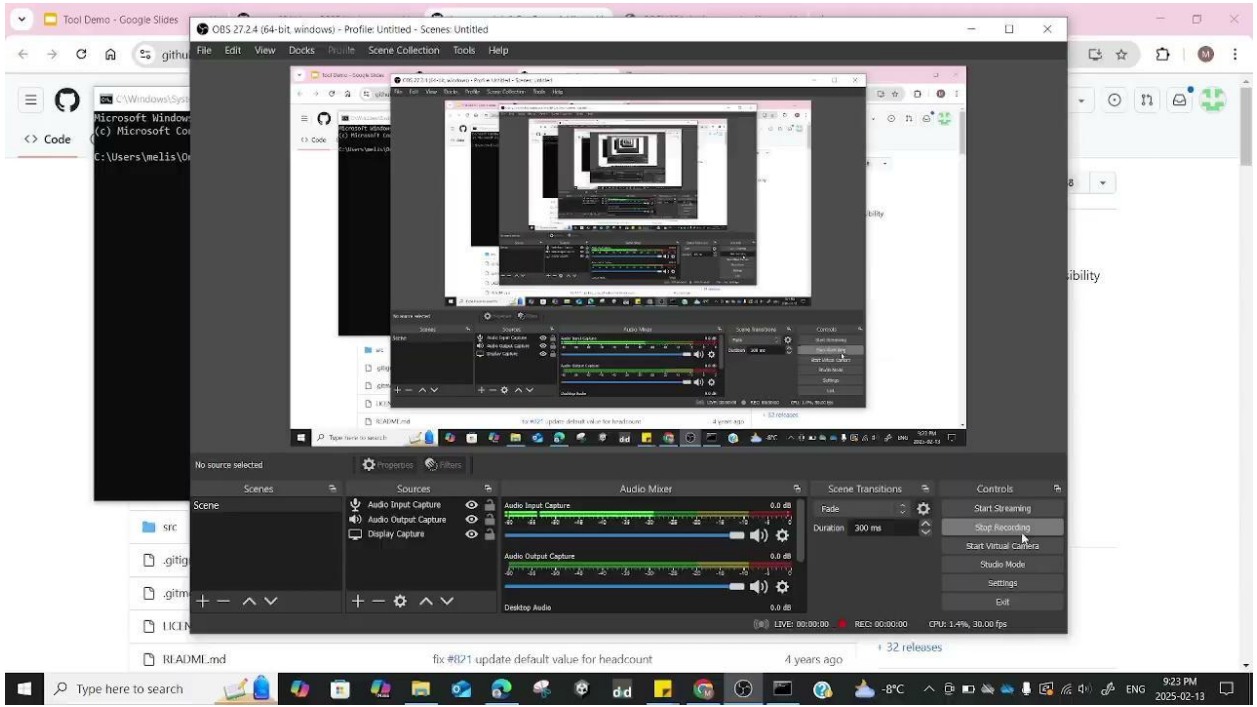
(b) Test cases

Example

```
--- example.CloseToZero
+++ example.CloseToZero
@@ -16,7 +16,6 @@
     */
    public int close_to_zero(int n) {
      if (n == 0) {
-       n++; // bug here
      } else if (n > 0) {
        n--;
      } else {
```

(c) Generated Patch

Demo


Tool Demo - Google Slides
OBS 27.2.4 (64-bit, windows) - Profile: Untitled - Scenes: Untitled
File Edit View Docks Profile Scene Collection Tools Help
No source selected
Properties
Filters
Scenes
Scene
Sources
Audio Input Capture
Audio Output Capture
Display Capture
Audio Mixer
Audio Input Capture 0.0 dB
Audio Output Capture 0.0 dB
Desktop Audio 0.0 dB
Scene Transitions
Fade
Duration 300 ms
Controls
Start Streaming
Stop Recording
Start Virtual Camera
Studio Mode
Settings
Exit
LIVE: 00:00:00 REC: 00:00:00 CPU: 1.4%, 30.00 fps
README.md
fix #821 update default value for headcount
4 years ago
32 releases
Type here to search
-8°C
ENG
9:23 PM
2025-02-13

RQ) How does kGenProg differ from its successor algorithms?

1) In-memory computation
2) Strategy pattern
3) High portability
4) Visualizing the process of fault modification

BIGGEST LIMITATIONS

- Can only add code that already exists in the program
- Constrained to predefined rules
- Can’t fix complex logical errors (no refactoring code)

Thank you