Automated defect correction

Tom Rochette <tom.rochette@coreteks.org>

November 2, 2024 — 36c8eb68

0.1 Context

0.2 Learned in this study

0.3 Things to explore

- What sort of defect can be automatically detected? Corrected?
  - It seems that this is mostly targeted at defects that will cause an uncaught exception or a crash
  - Defects as defined by "unexpected" behavior by the user is more akin to a requirement specification being too weak

1 Overview

2 Defect discovery

- Automated tests
- Users using the application

3 Defect diagnosis

- Determine keywords related to the symptoms/cause
- If a trace is available, use it
- If a dump is available, use it
- Rebuild an execution history based on the given final conditions (for example, you define that you want variable x to be null on line y, the assistant will attempt to generate a case where this may be possible)
- Suggest a fix
- Create fix in a branch (assumes a branch-based version control system such as git)
- Create a test case

4 ACS

The key to ACS's high precision is the use of multiple information sources, especially the "big code" existing on the Internet. Compared with existing techniques, ACS uses three new types of information sources.

- First, researchers noticed that the principle of locality holds on variable uses, and apply such information to sort the variables to be used in the patches.
- Second, ACS uses natural language analysis techniques to analyze Javadoc, and then uses the information in Javadoc to filter incorrect patches.
- Last, and most importantly, ACS performs statistical analysis on the open-source program on the Internet, discovers the conditional probabilities of the operations over the variables and further generates correct patches.

5 See also

6 References

- https://www.microsoft.com/en-us/research/blog/program-repairs-programs-achieve-78-3-percent-precision-automated-program-repair/