Mining Succinct Predicated Bug Signatures Artifact Description

Chengnian Sun, Siau-Cheng Khoo
School of Computing, National University of Singapore
{suncn, khoosc}@comp.nus.edu.sg

1 Selection of Experimental Subjects

We chose 5 buggy C programs from SIR. Each program is equipped with several versions, and each version contains a single bug. We use *proximity* to measure the effectiveness of MPS and LEAP. In order to compute the distance between the buggy vertex and the reported suspicious statements, we first need to locate the buggy vertex in the system dependence graph. However, it is difficult to locate buggy vertices for bugs which are due to missing of statements. Moreover, in our experiment, not all the bugs can be triggered by the associated test cases, So we skip these bugs and in total 75 faults are studied in our experiment.

We provide the detailed experimental data, program subjects, signatures output by LEAP and MPS, the details of *proximity* measurement, the profile datasets, and the MPS tool with its source code.

2 Detailed Experimental Results

- *proximity* values for all the bugs

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/MPS-EXP-Result.pdf`

- *runtime* statistics for all the bugs

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/mps-runtime-performance.pdf`

- *information gain* improvement for all the bugs

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/mps-ig-improvement.pdf`

3 Experimental Subjects and Signatures by MPS and LEAP

- The source code and the system dependence graphs of the buggy programs, and the mined predicated signatures can be found at

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/subjects-and-signatures.7z`

  In the archive file, each folder stores the data for a buggy program. Taking the folder *grep* as an example.

  Its subfolder *versions* contains all the bugs used in our experiment. In each version, the file *sdg.out* is the system dependence graph of that buggy program. Regarding the location of the bug, please refer to the manual in SIR (`http://sir.unl.edu/`).

  The subfolder *signatures* stores the predicated signatures mined by MPS. The *inter* folder stores the inter-procedural signatures, and the *intra* folder stores the intra-procedural signatures. The file *predicate.mapping* maps the numbers in the signature file to predicates instrumented in the buggy program. Note that a signature may contain a number of predicates, however after mapping these predicates back to the source code, each signature usually contains few statements (usually one or two lines).

- The source code of the buggy programs and the signatures mined by LEAP can be found at

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/LEAP-signatures.7z`

  As LEAP mines control-flow information, we use a difference instrumentor by instrumenting at basic-block level. The folder structure of this archive file is the same as the one above, except that each buggy version has a source file with a suffix *.cil.c. This file is the instrumented source code, serving as a mapping between numbers in LEAP signatures and basic blocks in source file.

4 Profile Dataset and Proximity Measurement

- The profile datasets are available at

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/MPS-profile-datasets.7z`

  The archive file contains 5 folders, each for a buggy program. Each program folder contains several version folders, each for a bug. In a version folder, the file *mps-ds.pb* is the dataset, saved in Google proto-buffer format. The file *predicate.mapping* stores the mapping between numbers and predicates instrumented in source code.

- The proximity measurement was conducted by four students in our lab. For each bug, they first manually locate the buggy vertex in the system dependence graph. Next they map the elements in signatures by MPS or LEAP to statements in source code, then to vertices in the system dependence graph. At last, they use standard breadth-first search algorithm to compute the distance between the buggy vertex and the statements pointed out by the signatures, and further they get the *proximity* values.

  The proximity data is available at

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/mps-proximity-data.7z`

  The mapping between signatures and vertices in SDG is stored in a file named *tc.in.* So you can find all such files in this archive. The first line of this file is the name of the bug (including the program subject name and the version). The third line of the file is the buggy vertex, a pair of numbers of which the first number is the procedure id and the second number is the vertex id in that procedure.

5 Tool Availability

- There is a binary release of our tool available at

  `http://www.comp.nus.edu.sg/~specmine/suncn/mps-artifacts/mps.tar.gz`

  And there is a short README and a sample dataset to run the tool. You can also run the tool with the provided dataset above.

- The source code of the tool is available at

  `https://bitbucket.org/chengniansun/mbs`