Detected non-deterministic results under various configurations

Closed

Hi, I have recently been using PyCG for an empirical study to detect non-deterministic behaviors in static analyzers. The experiments resulted in discovering some nondeterministic analysis results across multiple runs under various configurations of PyCG.

The details of the experimental setup are as below:

- The experiments were conducted on the PyCG micro and macro benchmarks.
- The experiments were conducted under 3 sample configurations which were generated using a 2-way covering array from the configuration space.
- The timeout set for PyCG running on both micro-benchmark and macro-benchmark was 5 minutes/program.
- We ran PyCG on each program-configuration combination 10 times and compared the results across 10 runs for detecting non-deterministic behaviors.
- All experiments were conducted in docker containers. The hardware environment is a server with 128GB of RAM and 24 Intel Xeon Silver 4116 CPUs@2.10GHz running Ubuntu 16.04.

In the end, the experiments detected non-deterministic results on 3 programs, which were all from micro-benchmarks. These results were observed under 2 out of 3 sampled configurations.

The attached data is the detected nondeterministic results from micro-benchmark and macro-benchmark and configuration files
(*note1:* the configurations are hash-coded in the detected results, but the actual configuration options and values that each hash code stands for can be found in the attached configuration files.)

👀 1

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

Notifications Custom

Unsubscribe

You're receiving notifications because you authored thread.

Owner

Hi, thanks for reporting.

What Python version are you using? Can you please paste the conflicting examples? I can't see the code samples or the corresponding results from the links you provided.

The best way to replicate the micro-benchmark experiments is through the pycg-evaluation artifact repository, which contains the version ran in the paper.

👍 1

Author

Hi Vitsalis, the Python version we are using is 3.10. Here is a conflicting example:

The program being analyzed is returns/call/main.py

result from run0:

```
{"main": ["main.func", "main.return_func"], "main.return_func": [], "main.func": []}
```

result from run1:

```
{"main": ["main.func"], "main.func": []}
```

detected nondeterministic results from micro-benchmark and macro-benchmark contains all detected nondeterministic results, each sub-folder contains results from 10 runs.

Owner

Closing due to archival of repository.