Towards More Effective Performance Fuzzing

Nov. 2022

Yiqun Chen, Matthew Bradbury and Neeraj Suri

y.chen101@lancaster.ac.uk

Agenda

- Motivation
- Background
- Early experiments

Motivation

- Performance issues
- Worst algorithmic cases
  - Example: Bad map (dictionary) implementation
  - Constant complexity -> linear complexity
- Bad algorithm -> bad performance
- DoS attack: from < *1 min* to *44 min*

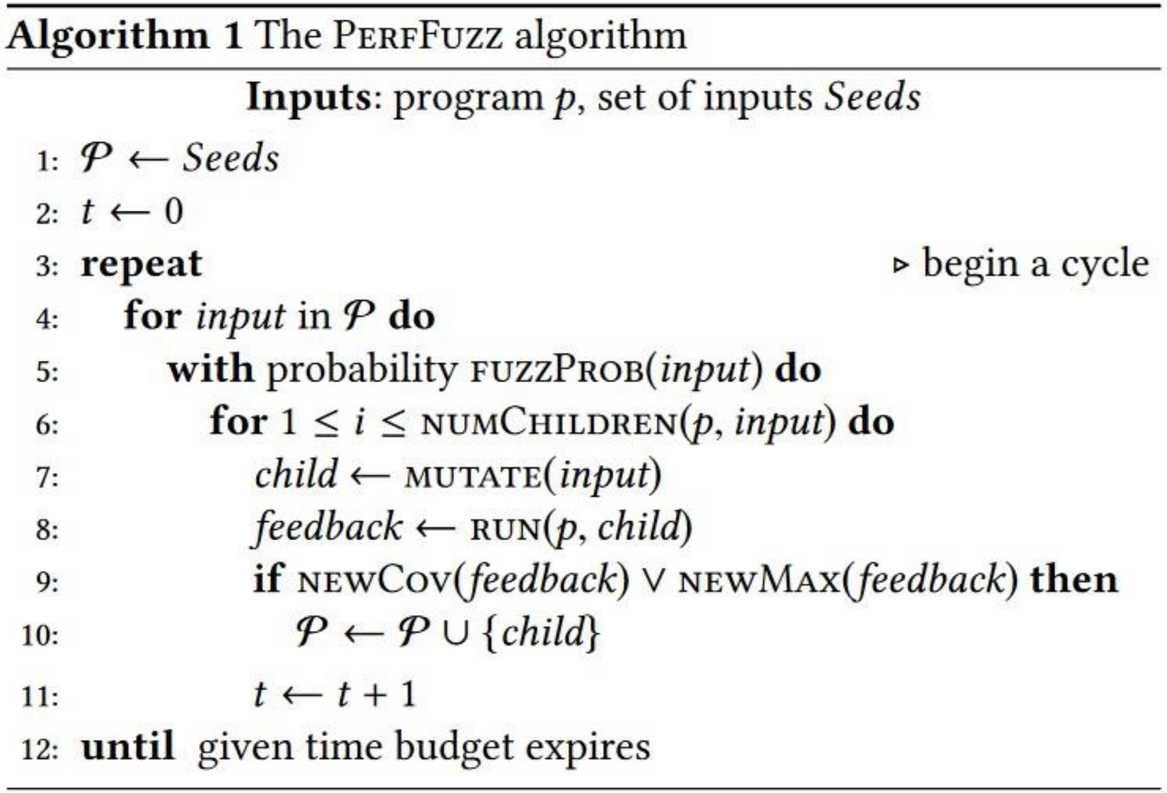
Motivation

- Many approaches for performance diagnostics: mostly profilers
- How to determine performance issues in the first place?
- Fuzzing: automatic generation of test cases

Background – Fuzzing

- A random process
  - Run test cases with inputs
  - Trace coverage
  - Select inputs (guide the fuzzing)
  - Mutate inputs
  - Repeat
- Searching for inputs yield larger code coverage

Background – Performance Fuzzing



Algorithm 1 The PerfFuzz algorithm

Inputs: program p, set of inputs *Seeds*

```
$\mathcal{P} \leftarrow Seeds$
$t \leftarrow 0$
repeat                                              ▹ begin a cycle
    for $input$ in $\mathcal{P}$ do
        with probability FUZZPROB($input$) do
            for $1 \leq i \leq$ NUMCHILDREN($p$, $input$) do
                $child \leftarrow$ MUTATE($input$)
                $feedback \leftarrow$ RUN($p$, $child$)
                if NEWCOV($feedback$) $\vee$ NEWMAX($feedback$) then
                    $\mathcal{P} \leftarrow \mathcal{P} \cup \{child\}$
                $t \leftarrow t + 1$
until given time budget expires
```

By Lemieux et al.

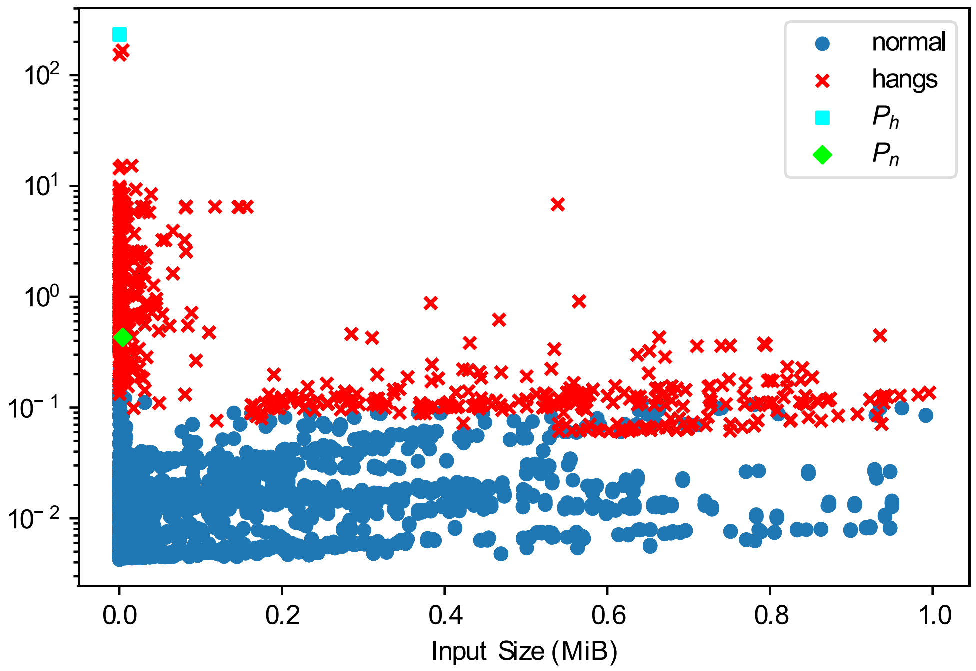
Motivation – Hypotheses

- Fuzzing parameters
  - Test case size: 1MB
  - Timeout: 1 second
- Path length as the measure of performance fuzzing?
  - Each piece of the path (basic block) has different performance

Early Experiments

- S1: default parameters (1 second 1 MiB)
- S2: custom parameter (10 seconds 100 MiB)
- Run fuzzing for 16 hours with 8 fuzzers (8 hours on 1 fuzzer on perffuzz paper)
- Measure execution time of all normal test cases and timeout test cases
- Repeat 100 times

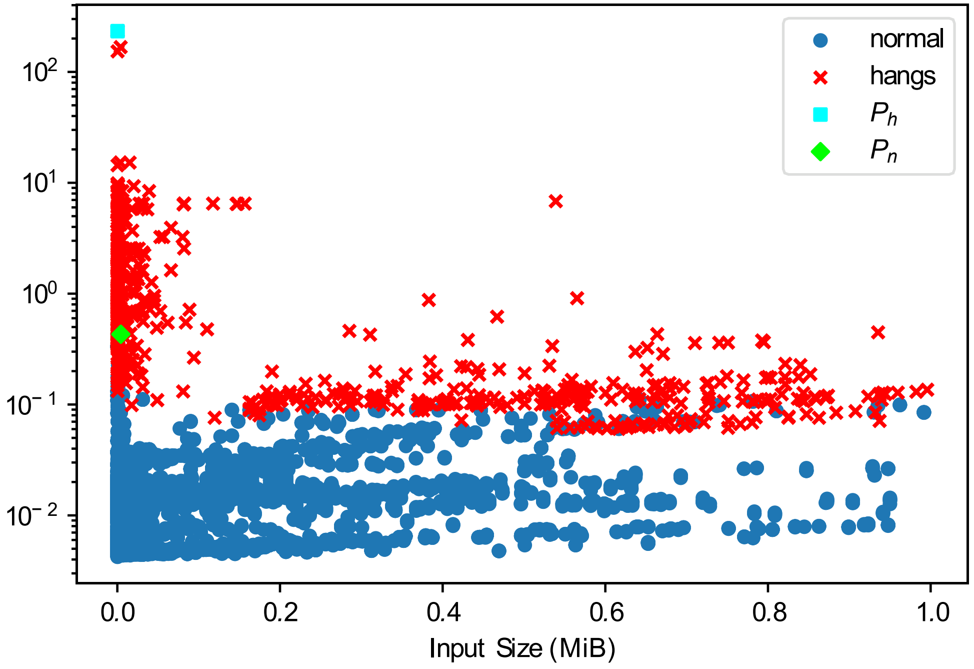
Early Experiments



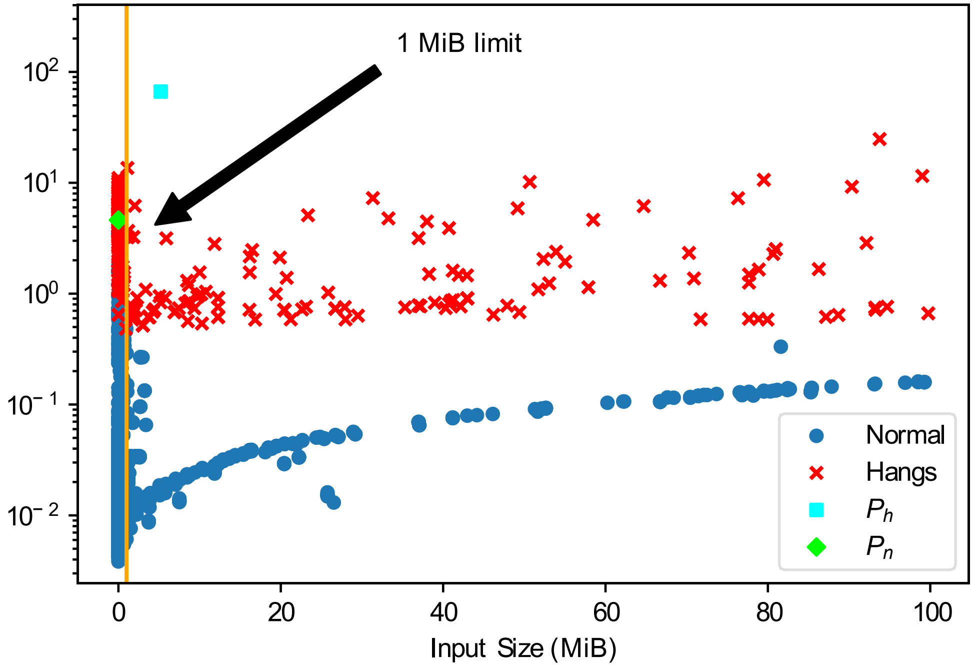
- X-axis: Test case size (MiB)
- Y-axis: Execution time (seconds)
- Blue dots: median of 100 execution times from normal test cases
- Red crosses: median of 100 execution times from timeout test cases
- Green diamond: slowest normal test case
- Teal square: slowest timeout test case

Early Experiments – File Size

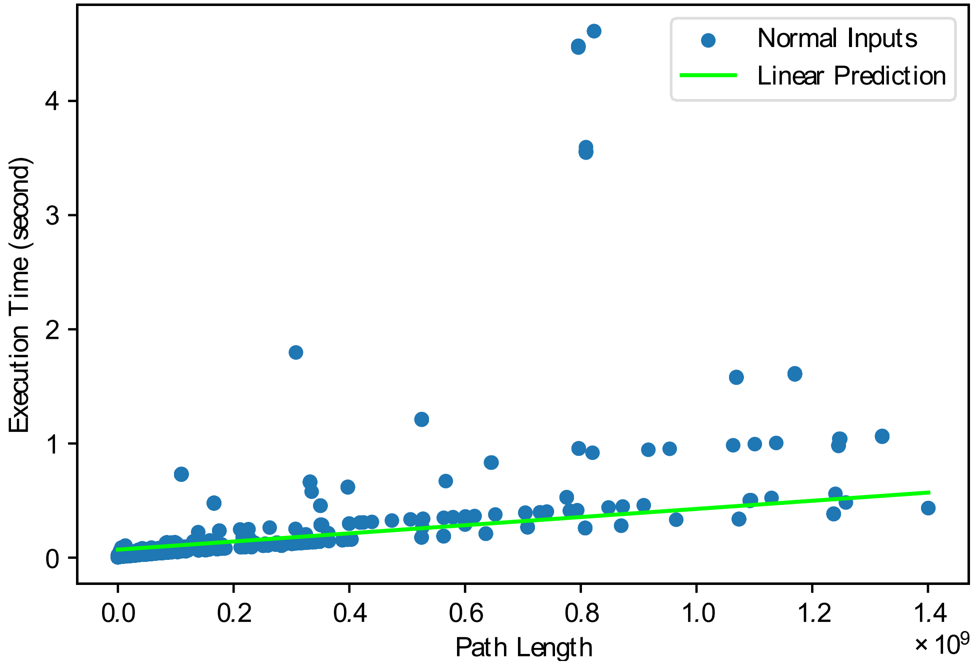
S1: 1 MB / 1 second (default)



S2: 100 MB / 10 seconds



Early Experiments – Path Length



- Blue dots: median of 100 execution times
- Green line: linear regression
- Performance is somehow correlated to path length
- Non-correlated test cases are more interesting

Lancaster University

Thank you for attending, any questions?