No PAIN, No Gain?
The Utility of PArallel Fault INjections

Stefan Winter[1] Oliver Schwahn[1] Roberto Natella[2] Neeraj Suri[1] Domenico Cotroneo[2]

Abstract: The article reports on interferences between concurrent fault injection test executions.

Keywords: Software fault injection, robustness testing, test interference

Software Fault Injection (SFI) emulates defects in software components to assess the robustness of other software components they interact with. After such defects have been introduced, the software composition is exposed to a workload and the effects of the introduced defect on the component under assessment are monitored. SFI tests entail relatively long execution times for three reasons: (1) They operate on a fully integrated software system, which entails corresponding loading and initialization times. (2) To activate the introduced defects and assess their impact on possibly complex component interactions, the execution of complex workloads is required. (3) After each SFI test execution, the entire software system under test (SUT) needs to be reset to a known fault-free state to prevent residual side effects of injected faults from affecting subsequent tests. Especially the last point entails significant execution time overhead. As the possible effects of an injection cannot be predicted (if they could, no SFI tests were needed), resetting the SUT usually requires a complete termination and re-initialization, sometimes even of its execution environment (e.g., the test machine's file system) if it can be affected by the injected fault.

To improve test throughput, we propose to exploit parallel hardware and execute SFI tests concurrently. While this appears to be a simple and straight-forward solution, it is based on an assumption of non-interference between SFI tests. In a paper [Wi15] that we presented at ICSE this year, we experimentally evaluated this assumption. We executed SFI tests on the Android OS kernel by injecting faults into the SD card driver. To contain the effects of fault activations during these tests, the system was executed in an emulator that was reset after each test. We repeated the tests with varying degrees of concurrency by instantiating varying numbers of emulator instances. To assess, whether concurrency has an effect on the experiment outcome, we compared the result distributions for the varying degrees of concurrency. Our initial results showed significant deviations for higher degrees of concurrency, indicating that an unreflected replication of SUT instances threatens the validity of test results. We identified the SUT instances' competition for shared system resources and the resulting execution latency increases, which directly affected some of

[1] Technische Universität Darmstadt, DEEDS Group, Hochschulstr. 10, 64289 Darmstadt, Germany, {sw | os | suri}@cs.tu-darmstadt.de

[2] Federico II University of Naples, DIETI, via Claudio 21, 80125 Naples, Italy, {roberto.natella | cotroneo}@unina.it

the employed test oracles, as the root cause for the observed deviations. We then devised a pre-test measurement approach to adjust these oracles for the concurrent execution of a given number of SUT instances on a given test machine. Using this approach, we were able to execute up to 44 SFI tests concurrently without any significant test result deviations on a machine with 16 CPU cores and 64 GiB main memory. The highest throughput for this configuration was 157 experiments per hour with 36 concurrent instances, a more than 12-fold throughput increase compared to sequential test execution. For this configuration we also observed the lowest correlation between the degree of concurrency and the test result distribution in a χ^2 test for independence, which indicates that the initially observed result deviations were indeed caused by performance interference of concurrent test executions.

Besides the direct impact of our result on SFI and other robustness testing approaches, where performance sensitive oracles are used to detect so-called hang failures, our result indicates that test parallelization requires careful analysis to obtain valid results *whenever tests rely on execution latencies*. For example, any JUnit tests that use the `timeout` parameter or `Timeout` rule would be similarly affected. An interesting observation from our experiments was that the SUT initialization contributed significantly to the observed test latencies. This is not surprising, as we used heavy-weight isolation measures to make test executions as independent as possible from each other by running them in (almost, as our results show) completely isolated environments. This opens up the possibility for a trade-off: Performance interference decreases with less isolation, which on the other hand increases the risk for other types of test interference [Zh14].

While our pre-test measurement approach proved effective for time-dependent oracle adjustment, it required the execution of around 800 tests for reliable predictions. Our goal is to reduce this calibration overhead and, ideally, devise an analytical model for accurate predictions of *safe* time-dependent oracles and achievable concurrency degrees for a given test type and test machine configuration, that do not even require additional test executions for calibration. To achieve this, we need to better understand the root causes behind latency increases. We hope the related research to also shed some light on the factors that caused throughput to degrade when more than 36 concurrent SUT replica were instantiated in our experiments and to guide hardware and scheduler configuration for better test throughput.

Acknowledgments: This research has been supported in part by DFG GRK 1362, CASED, EC-SPRIDE, EC H2020 #644579, CECRIS FP7 (GA no. 324334), and SVEVIA MIUR (PON02 00485 3487758).

References

[Wi15] Winter, Stefan; Schwahn, Oliver; Natella, Roberto; Suri, Neeraj; Cotroneo, Domenico: No PAIN, No Gain?: The Utility of PArallel Fault INjections. In: Proceedings of the 37th International Conference on Software Engineering - Volume 1. ICSE '15, IEEE Press, Piscataway, NJ, USA, pp. 494–505, 2015.

[Zh14] Zhang, Sai; Jalali, Darioush; Wuttke, Jochen; Muşlu, Kivanç; Lam, Wing; Ernst, Michael D.; Notkin, David: Empirically Revisiting the Test Independence Assumption. In: Proceedings of the 2014 International Symposium on Software Testing and Analysis. ISSTA 2014, ACM, New York, NY, USA, pp. 385–396, 2014.