PRUNERS: Providing Reproducibility for Uncovering Non-deterministic Errors in Runs on Supercomputers

Kento Sato, Ignacio Laguna, Gregory L. Lee, Martin Schulz and Christopher M. Chambreau and Dong H. Ahn
Lawrence Livermore National Laboratory
Livermore, USA
{kento, ilaguna, lee218, schulzm, chambreau1, ahn1}@llnl.gov

Simone Atzeni, Michael Bentley,
Genesh Gopalakrishnan, Zvonimir Rakamaric and Geof Sawaya
University of Utah
Salt Lake City, USA
{simone, mbentley, ganesh, zvonimir, sawaya}@cs.utah.edu

Joachim Protze
RWTH Aachen University
Aachen, Germany
protze@itc.rwth-aachen.de

***Abstract*—Large scientific simulations must be able to achieve the full-system potential of supercomputers. When they tap into high-performance features, however, a phenomenon known as non-determinism may be introduced in their program execution, which significantly hampers application development. PRUNERS is a new toolset to detect and remedy non-deterministic bugs and errors in large parallel applications. To show the capabilities of PRUNERS for large application development, we also demonstrate their early usage on real-world, production applications.**

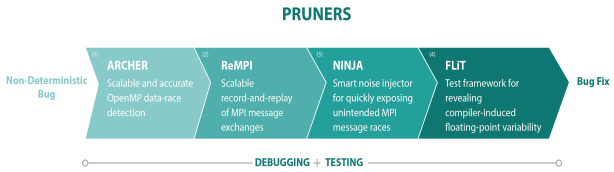

Fig. 1. The PRUNERS toolset increases non-determinism coverage for debugging and testing workflows

I. INTRODUCTION

High-Performance Computing (HPC), the applied use of supercomputers, has become critically important for R&D efforts, scientific discovery and for many other fields. Applications running on supercomputers must rely on multiple communication and synchronization mechanisms as well as compiler optimization options to effectively utilize the hardware resources. These complexities often produce errors that occur only occasionally, even when run with the exact same input (i.e., same binaries, configuration and input data) on the same hardware. These so-called non-deterministic bugs are remarkably challenging to reproduce and thus challenging to debug.

PRUNERS is a new toolset to detect and remedy non-deterministic bugs and errors in large parallel applications. Prior to PRUNERS, programmers relied on manual techniques using traditional parallel debuggers such as TotalView, DDT, GDB and `printf` debugging. Although these techniques are effective when the error deterministically occurs at the same point of execution, stepping through source lines is often ineffective on non-deterministic bugs because the process of stepping may perturb the execution enough to mask the bug. One of our recent case studies indicates that resolving a single non-deterministic bug can require 3 person-months worth of programmers effort and 19 years of compute-core time in total [3]. Thus, tools that can detect and remedy non-deterministic errors are highly valuable.

PRUNERS is the only software toolset that is designed specifically to solve the challenges of debugging and testing for non-deterministic software bugs with the scalability, accuracy, composability and portability demanded by todays high-end systems. Early usage on real-world bugs at LLNL proves that PRUNERS is the first coordinated toolset for non-deterministic software bugs and errors.

II. THE PRUNERS TOOLSET

Sources of non-determinism are multilevel — arising from multiple levels of the software stack (e.g., applications, libraries and/or system) and from different programing models and paradigms (e.g., message passing versus shared memory). A single tool does not fit all required levels of reproducibility. Our solution to the challenge is the PRUNERS toolset (Figure 1). A reason for having a toolset is to flexibly control sources of non-determinism and provide required levels of reproducibility depending on specific code development needs by users — levels of reproducibility versus performance/scalability.

The PRUNERS toolset offers four novel debugging and testing tools: Archer [1], ReMPI [2], NINJA [3] and FLiT [4]. Our

toolset specifically aims at the non-determinism introduced by use of todays two most dominant parallel programming models, MPI and OpenMP, as well as major compilers.

1) ARCHER: Archer is a scalable and accurate OpenMP data-race detection tool. Archer's static analysis passes classify the given OpenMP code regions into two categories: guaranteed race-free and potentially racy. Its dynamic analysis then applies state-of-the-art data-race detection algorithms to check only the potentially racy OpenMP regions of code. The static/dynamic analysis combination is central to the scalability (while maintaining analysis precision) of Archer [1].

2) ReMPI: ReMPI is a scalable MPI record-and-replay tool. After a buggy, non-deterministic run is recorded by ReMPI, programmers can deterministically replay MPI message exchanges, thereby, reproducing a target bug, even under the control of a parallel debugger, for further root-cause analysis. With a new compression algorithm called Clock Delta Compression (CDC), ReMPI enables scalable MPI record-and-replay [2].

3) NINJA: NINJA is a smart noise injection tool for exposing unintended MPI message races. NINJA controls and manipulates the timings of non-deterministic message matches in ways that maximize the chance to expose MPI message-race bugs. With dynamic analysis of communication patterns of applications, NINJA can minimize the overhead by selectively injecting noise into only racy MPI message exchanges [3].

4) FLiT: FLiT is a test framework for quickly revealing compiler-induced floating-point variability. FLiT can help a programmer quickly quantify the extent of floating-point variability induced by compilers and their options, and by different computing platforms. Give that there is a lack of compiler standardization, it is crucial to have tools like FLiT to offer warnings about the portability of code to different compilers and platform [4].

To ensure wide applicability and flexibility, these components are designed to complement each other while each individual component also works effectively as a single, standalone tool. PRUNERS provides a common multilevel toolset that unifies complementary and targeted tools that can build on the strengths of one another.

III. State-of-the-practice

To ensure that our capabilities work for large application development, we have actively engaged many production code-development teams at the Lawrence Livermore National Laboratory (LLNL) and tested the utility of our tools on large code bases. In this process, these users have challenged our research team with those bugs that they previously deemed intractable, and PRUNERS has consistently proven to be effective on them.

For example, Archer detected and helped fix highly elusive OpenMP data races in HYDRA, a very large multiphysics application for Livermore's National Ignition Facility (NIF), that caused code crashes that only intermittently manifested themselves after varying numbers of time steps and only at large scales (8,192 MPI processes or higher). Further, as LLNL code teams increasingly multi-thread their applications, they have begun to integrate Archer directly into their build-and-test systems to catch data-race bugs at testing time, before production runs are conducted.

ReMPI has significantly been helping ParaDiS (dislocation dynamics application) and Mercury (domain-decomposed particle transport application) debug and test MPI non-determinism. NINJA has been shown to manifest unsafe message races consistently within LLNL Diablo's (a massively parallel implicit finite element application) use of Hypre (a scalable linear solvers and multigrid method library). Hypre had a message-race bug that has not been uncovered until recently.

An easy-to-use and community-extensible tester like FLiT is also becoming increasingly important in scientific work dependent on supercomputers. Critical supercomputing applications such as the Community Earth Simulation Model, for instance, have yielded inconsistent results when ported across platforms and compilers, which hampered validation effort. Similarly, errors in floating-point calculations have led to inaccurate findings based on data analysis of Large Hadron Collider experiments. FLiT offers earlier warnings as to the portability of their code to different compilers.

IV. Summary

Non-deterministic execution is becoming increasingly common and is particularly difficult for programmers to comprehend and debug. PRUNERS is the first coordinated toolset that is designed specifically to help debug and test for non-deterministic bugs, with features and attributes commensurate for large supercomputers. It was designed specifically with scalability, accuracy, and composability in mind. PRUNERS has demonstrated early success on real-world bugs and already resulted in cost savings at LLNL.

Acknowledgment

This work was performed under the auspices of the U.S. Department of Energy by LLNL under contract DE-AC52-07NA27344 (LLNL-CONF-737603).

References

[1] S. Atzeni, G. Gopalakrishnan, Z. Rakamaric, D. H. Ahn, I. Laguna, M. Schulz, G. L. Lee, J. Protze, and M. S. Mller, "ARCHER: Effectively Spotting Data Races in Large OpenMP Applications," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2016, pp. 53–62.

[2] K. Sato, D. H. Ahn, I. Laguna, G. L. Lee, and M. Schulz, "Clock Delta Compression for Scalable Order-replay of Non-deterministic Parallel Applications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '15. New York, NY, USA: ACM, 2015, pp. 62:1–62:12.

[3] K. Sato, D. H. Ahn, I. Laguna, G. L. Lee, M. Schulz, and C. M. Chambreau, "Noise Injection Techniques to Expose Subtle and Unintended Message Races," in *Proceedings of the 22Nd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP '17. New York, NY, USA: ACM, 2017, pp. 89–101.

[4] G. Sawaya, M. Bentley, I. Briggs, G. Gopalakrishnan, and D. Ahn, "FLiT: Cross-Platform Floating-Point Result-Consistency Tester and Workload," in *2017 IEEE International Symposium on Workload Characterization (IISWC)*, October 2017 (To appear).