Evento	Salão UFRGS 2019: SIC - XXXI SALÃO DE INICIAÇÃO CIENTÍFICA DA UFRGS
Ano	2019
Local	Campus do Vale - UFRGS
Título	Evaluation of Compiler Effects on OpenMP Soft Error Resiliency
Autor	JONAS FOGLIARINI GAVA
Orientador	RICARDO AUGUSTO DA LUZ REIS

Universidade Federal do Rio Grande do Sul

Evaluation of Compiler Effects on OpenMP Soft Error Resiliency

Author: Jonas Fogliarini Gava **Advisor:** Ricardo Reis

Software engineers are using different compilers and parallel programming models (e.g., Pthreads, OpenMP) to take the best performance offered by multicore systems. For example, High-performance computing (HPC) feature a hierarchical hardware architecture: multiple nodes connected via a network infrastructure, where each node integrates a local memory, a graphics processing unit (GPU), and a shared-memory multicore processor. Usually, such a system relies on hybrid parallel programming models, i.e., message passing to inter-node data exchange, and shared-memory to support inter-processor synchronization. While Message Passing Interface (MPI) is mostly used for inter-node synchronization, Pthread and OpenMP are widely accepted shared memory models. Underlying programming models differ on how threads are created and managed. For instance, while threads are created and managed by the compiler in OpenMP, the creation and management of threads must be explicitly defined by the programmers in the Pthreads library.

Both programming models and compilers have a direct impact on applications performance, power-efficiency, and reliability. Among the reliability issues, the occurrence of soft errors is getting more attention, since they can dramatically affect the functionality of a life-critical system such as self-driving cars. The malfunction of such kind of system may lead to death or severe injury to human lives. Our work contributes by investigating the impact of distinct compilers on the soft error reliability of applications implemented with OpenMP library, when executing on Arm multicore processors, and Linux kernel.

For the experiments, we utilized the Single Event Upset (SEU) fault model. That is, we assume that exactly one bit is flipped precisely once during a program's execution. Fault injection campaigns were conducted with OVPsim-FI and consider three open-source compilers (GNU/GCC 5.5.0, 7.3.1, Clang 6.0.1), five optimization flags, 16 OpenMP benchmarks executing on single, dual and quad-core versions of the Arm Cortex-A72 processor, all running on a Linux system (kernel 4.3). Each campaign injects a random bit-flip in one of the general purpose registers.

Results show that, on average, Clang is 15.85% more reliable than both versions of GCC, which demonstrate to be more sensitive to the optimization flags when compared to Clang. We conclude that when the complexity of the system increases (e.g., more cores) the difference of faults masked between compilers reduce to around 5% when utilizing a high optimization flag. It was also possible to note that Clang does not maintain good reliability in some biological and physical benchmarks, as well as GCCs in image processing, physics simulation, and biological applications.