Universidade: presente!

UFRGS PROPESQ

XXXI SIC

Salão UFRGS 2019 — Conhecimento to Formação to Inovação

21 . 25 . OUTUBRO . CAMPUS DO VALE

Evaluation of Compilers Effects on OpenMP Soft Error Resiliency

Autor: Jonas Gava[1] Orientador: Ricardo Reis[2]

Introduction and Motivation

The occurrence of **soft errors** in **multicore systems** is a **growing reliability issue** in several domains (e.g., automotive, medical, avionics)

Both **programming models** and **compilers** have a **direct impact** on applications **performance**, **power-efficiency** and **reliability**

Necessity to investigate **soft error resilience of parallel applications** using **different compilers as HPC importance grows.**

Contributions

Investigation of the impact of distinct compilers on the soft error reliability of applications implemented with OpenMP library

Evaluation on single-core, dual-core and quad-core ARM processor.

Analysis of applications executed instructions

Analysis of registers usage

Analysis of the impact of code optimisation flags on reliability

Experimental Setup

Processor

Arm Cortex A72

Software Stack

Linux (kernel 4.3)

Clang 6.0, GCC 5.5, and 7.3

OpenMP LIbrary

Optimisation Flags

O1, O2, O3, Os, Ofast

Number of Applications: 16

Single Event Upset

Total Fault Injections: 691,200

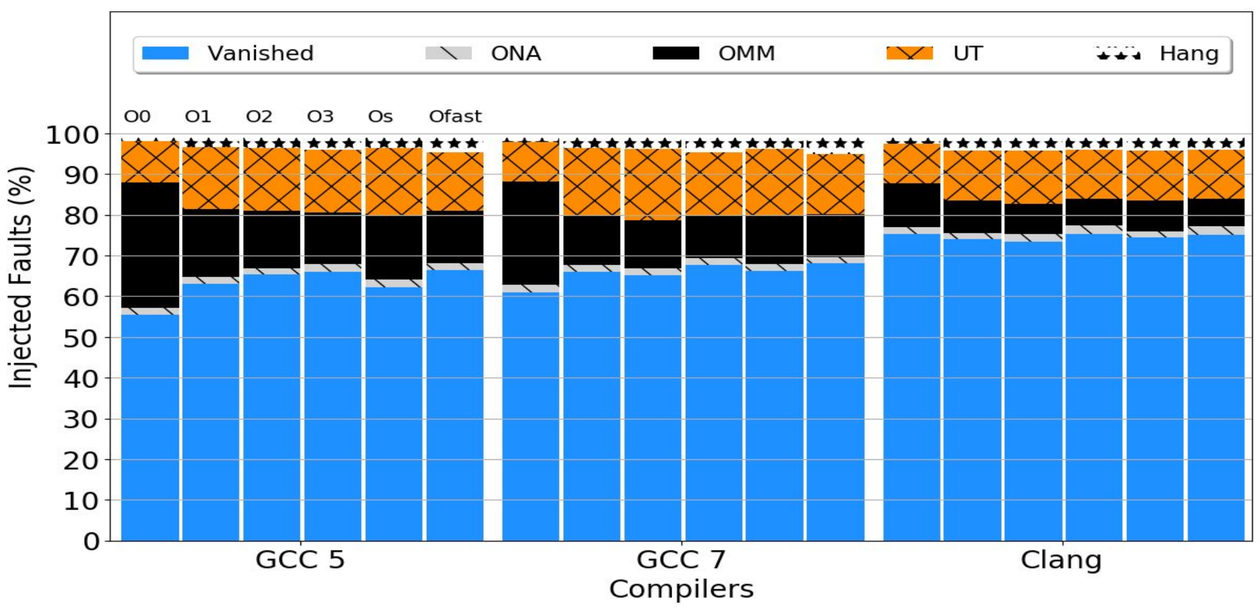
Instructions Profile

Instructions were classified as:
Mem, memory op. (e.g., ld, st, mov);
Ctrl, control flow (e.g., bne, jump);
Alu, arithmetic and logic op. (e.g., add, sub, mul)

In general, **Clang** generates more **memory instructions** while **GCC** generates more **control** and **arithmetic instructions**.

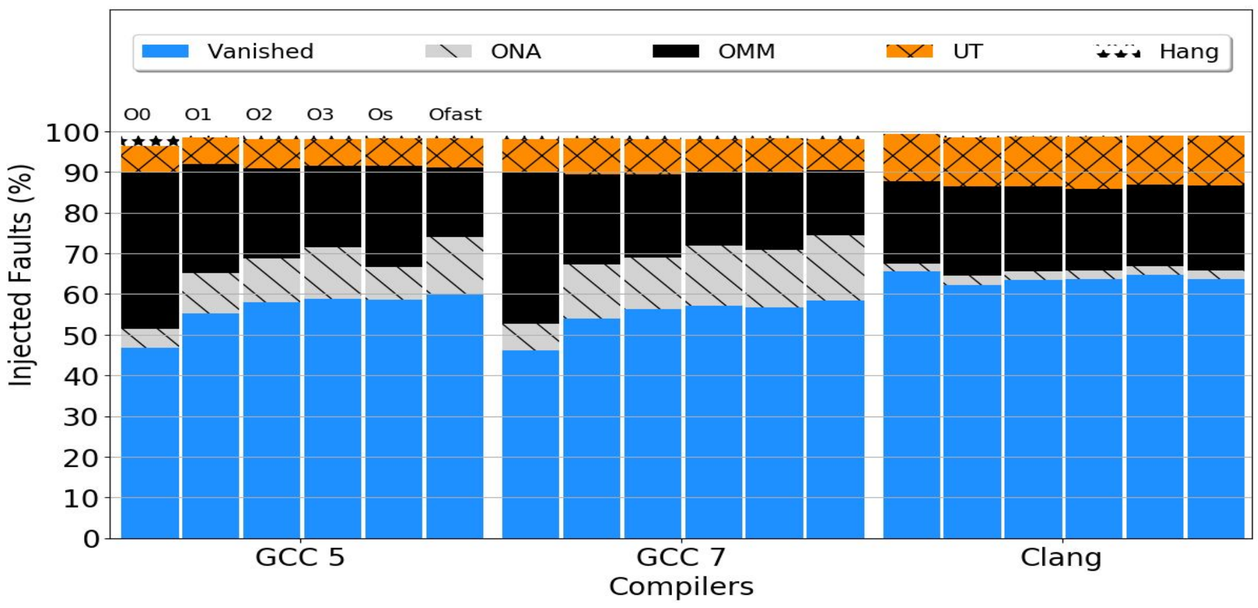
Results

FI Result for each Compiler/Flag (Single-Core)



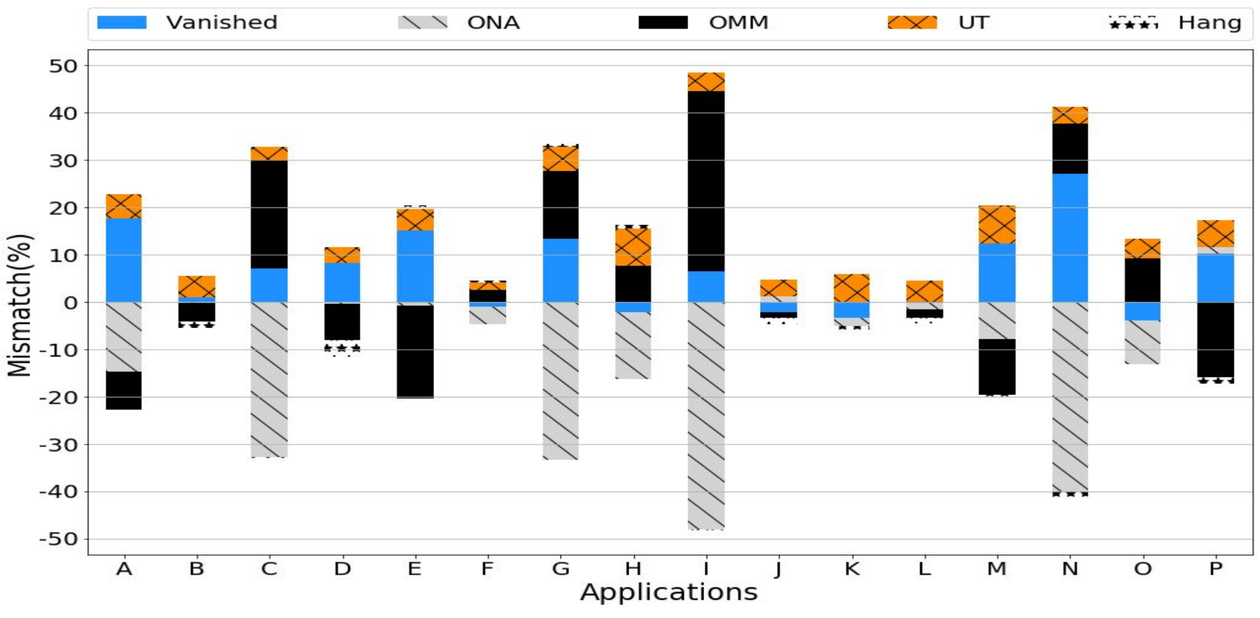
The use of **Clang** brings **stable reliability results**. For all flags, the minimum Vanished is 75.09%, and max is 75.33%. In turn, **GCC** compilers show a **direct correlation** between **optimisation flags and reliability**. For instance, when comparing the GCC5 O0 and Ofast, it is possible to identify an improvement of 10.5% on Vanished.

FI Result for each Compiler/Flag (Quad-Core)



When we **increase** the number of **cores**, it considerably **decreases** the **reliability** of the system, increasing OMMs and reducing Vanishes. With quad-core and maximum code optimisation, all compilers present similar results, with a more considerable difference between Clang and GCC7—Vanished (5.38%) and OMM (4.95%).

Results Mismatch for each Application Clang vs GCC 7 (Quad-Core)



This figure shows the mismatches between the results of the fault injections (i.e., Clang (%) − GCC 7 (%)) per application. The positive blue bars indicate the reliability improvement from Clang comparing with GCC 7.

Conclusions

This work evaluates the reliability of OpenMP application executing on a multicore system through 864 scenarios using three compilers, three processor-core variants (i.e., single-core, dual-core, quad-core) and five optimisation flags. We conclude that **when the complexity of the system increases** (e.g., more cores) the **difference of faults masked between compilers reduce**, but on average Clang is around 10% more reliable than GCC for all experimental variations we did.

UFRGS UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Departamento de Informática Aplicada

Grupo de Microeletrônica

Instituto de Informática

Campus do Vale, Bloco IV. Prédio 43413

GME microelectronics group

[1] **jfgava@inf.ufrgs.br**

[2] **reis@inf.ufrgs.br**

.inf INSTITUTO DE INFORMÁTICA UFRGS

CNPq