CS @ ILLINOIS
COMPUTER SCIENCE

IMPROVING FLIPIT

AN LLVM BASED FAULT INJECTOR FOR HPC

MENTOR: JON CALHOUN **MENTEE:** YUXIAN ZHU

FAULT INJECTOR? FLIPIT?

High Performance Computing (HPC), which has become the fundamental part of scientific investigation and discovery, faces the affliction of hard and soft faults that causing errors in computer systems. Soft faults, in particular, are not systematically reproducible and therefore propose a big challenge to application's fault tolerance capabilities. Common soft faults, like a bit-flip caused by a charged particle, can happen to a large scale HPC system on a daily basis. [1]

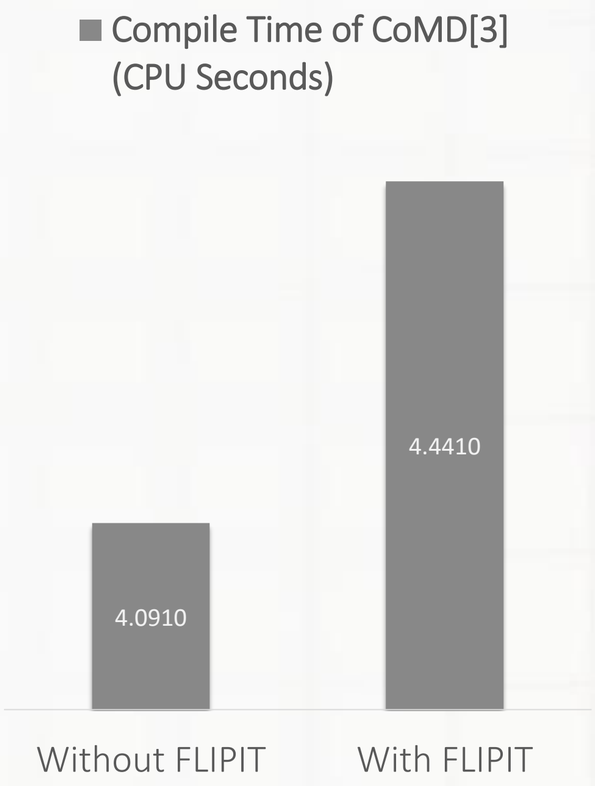
Consequently, Fault Injectors are designed to simulate manifestation of soft faults and help determine their impacts on HPC applications. FlipIt is a fault injector structured as an LLVM compiler pass. It instruments code allowing the result of an instruction to be incorrect with a certain probability [2]

A crash of HPC programs can be disastrous. All unsaved progress will lost and it takes time to recover. Therefore it's crucial that programmer can test the robustness of their code.

@ "Sad Face" appeared when "Windows" crashes

IT'S NOT PERFECT.

FlipIt, despite its improvement over existing fault injectors, can bring considerable overhead in compiling and running processes of the application, due to the fact that it adds function calls into the source code which will take some non-negligible portion of time to be completed; thus making these already time-consuming procedures even worse. Therefore, our main focus has been analyzing the bottlenecks of the fault injection framework and optimizing function structures in order to improve its compile-time and run-time performance.



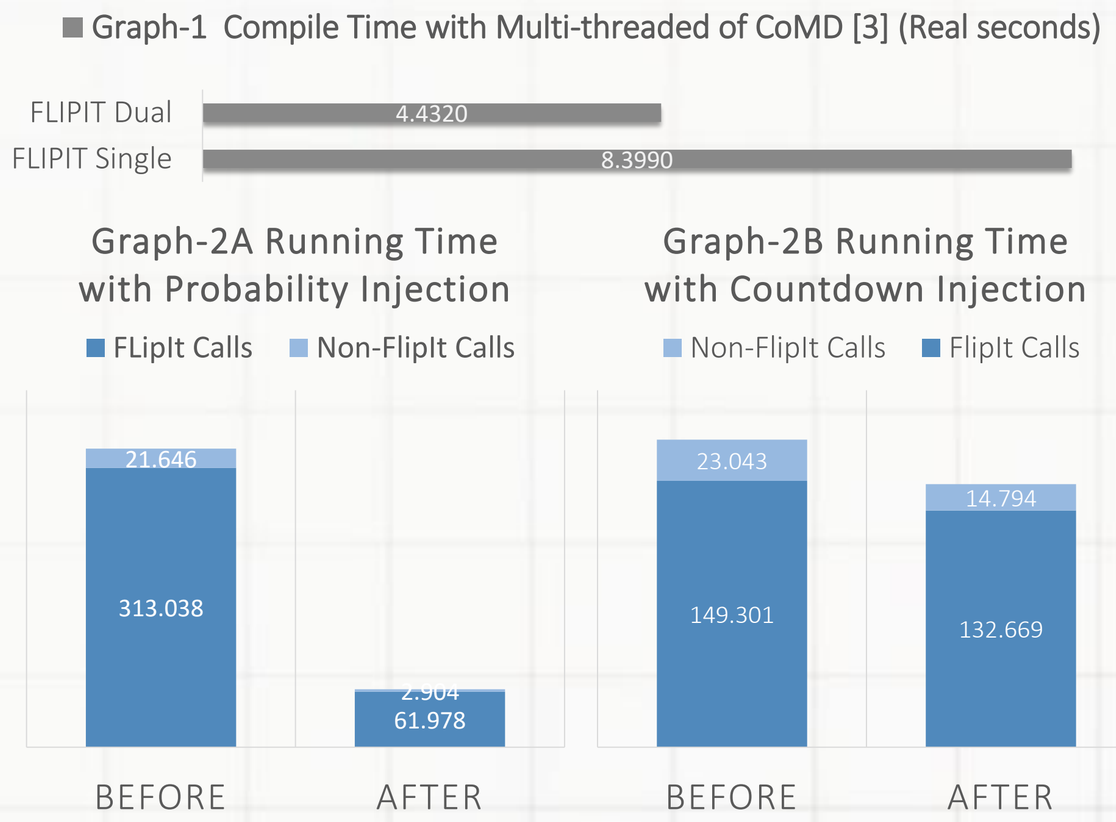
WE HAVE:

Modified FlipIt source code and enabled multi-thread compilation to speed up compile process. (@Graph-1 below)

Profiled FlipIt running-time performance and identified the bottlenecks of existing C/C++ functions. By restructuring fault injection and activation mechanisms, shrinking number of arguments for some C functions, redefining function types and return values, trimming unnecessary methods and branches, adding compiler macros, we were able to eliminate about 40% of original code, and improve FlipIt run-time performance by 17% to 75%. (@Graph-2 below)

Redefined variables definitions, employed more standardized POSIX & C library utilities, which makes FlipIt more portable on different platforms and easier to maintain in the future.

Added option for user to specify the bit and byte which they wish their data to be corrupted, allowing them to have more control over the fault injection process.



THERE IS MORE TO DO:

We are planning on including multi-thread injection support and compiler macros that gives user more control over specific areas that they want to inject faults in. Plus, we are also considering adding static injection as an option to the user therefore further reduces running time overhead.

A BIG THANK YOU TO:

I would like to thank my mentor, Jon Calhoun, for allowing me to work on this fascinating project, for guiding and helping me along the process.

Some information displayed is adopted from:

[1] Calhoun, J., Olson, L. & Snir, M. FlipIt: An LLVM Based Fault Injector for HPC. In Euro-Par 2014: Parallel Processing Workshops, 8805:547-558, Springer International Publishing, 2014.

[2] Effect and Propagation of Silent Data Corruption in HPC Applications. May 2015. 3rd NCSA Blue Waters Symposium for Petascale Science and Beyond

[3] CoMD: A Classical Molecular Dynamics Mini-app. CoMD version 1.1 was used, its document can be found at http://exmatex.github.io/CoMD/doxygen-mpi/index.html.

PURE Promoting Undergraduate Research in Engineering | Rockwell Collins | CAT | CS @ ILLINOIS Department of Computer Science | ENGINEERING AT ILLINOIS