FPSanitizer: Finding instability in floating point applications

Sangeeta Chowdhary
Dept. of Computer Science

1/29/2019 at 02:00 pm
CoRE B (305)

Abstract

Real numbers cannot be represented accurately in computer systems due to space and performance limitations. IEEE 754-2008 floating point standard provides a way to approximate reals in computers. The approximation of reals causes rounding errors. Rounding errors can be propagated, magnified due to cancellation and accumulated throughout the program. In contrast to computation with reals, branch computation can differ in floating point computation because of rounding errors. In our work, we run a shadow execution of an application with higher precision and compare the actual computation to find rounding errors. Although running in parallel with higher precision is not exactly doing computation on reals but it is closer to reals since parallel computation is more precise than actual computation. We have built a tool (fpsan) which can detect such branch flips in applications with thousand lines of code and report the location of a branch flip in the source code to the developer.

We have built fpsan which operates at LLVM IR level which gives the flexibility to detect such errors in any application which can be compiled to LLVM IR. We can also compare floating point rounding errors caused by different compiler optimizations.

The most related previous work to our approach is Herbgrind. Herbgrind instruments binary to run computations in higher precision to find rounding errors. Herbgrind also builds an expression to give more insights to the developer about an expression which caused the error. Although building expression is useful but it takes more memory. Our tool is lightweight and takes less memory than the state of the art.

Examination Committee: Prof. Santosh Nagarakatte (Chair), Prof. Richard Martin, Prof. Mridul Aanjaneya, Prof. Swastik Kopparty