PEARC’19 Half Day Tutorial

Floating-Point Analysis Tools

Ignacio Laguna, Harshitha Menon, Tristan Vanderbrugge
Lawrence Livermore National Laboratory

Michael Bentley, Ian Briggs, Ganesh Gopalakrishnan
University of Utah

Cindy Rubio González
University of California at Davis

http://fpanalysistools.org/
The Floating-Point number system is not new

Zuse Z1 (~1938)

IEEE Standard for Floating-Point Arithmetic
Floating-Point approximates Reals

- Because of rounding, \((x+y)+z \neq x+(y+z)\)
  - And many more such identities are violated
- Compilers can change your math
  - \(x/y \rightarrow x \times (1/y)\)
- Rounding errors are non-intuitive
  - Because of the uneven FP number scale
The Floating-Point Rounding is Non-Intuitive
The FP number system tries to span a large range using an “insufficient number of bits”
The FP error function is highly non-intuitive

E.g.

Rounding error of \((x+y)\) as a function of \(x\) and \(y\)
Kahan’s observation

Numerical errors are rare, rare enough not to care about them all the time, but yet not rare enough to ignore them.

— William M. Kahan
Floating-Point Analysis is Suddenly “Front and Center” in HPC + many other areas

- Allocating needlessly high precision increases data movement
  - Multiple precision types are on the rise
    - Often driven by ML
- The variety of hardware is increasing
  - GPUs and other accelerators
    - Their normal behaviors as well as EXCEPTIONS are on the rise
- Compilers exploit floating-point in an increasing number of ways
  - Compiler flags mean different things
    - Compilers may heed your flags selectively
Frenetic pace of FP research now

- Multiple conferences
- Many sessions per conference
- Many different issues

Very little that is tangible for a practitioner to try some of these out

Some good resources do exist (will put it on our website)

E.g., fpbench.org

Michael O. Lam, Floating Point Analysis Research
Goals of this Tutorial

● Introduce FOUR mileposts in your repertoire of knowledge
  ○ Four tools you can practice during the tutorials
  ○ You can apply them in your own projects!

● We are a resource you can count on during your future work
  ○ We are invested in multiple research projects in this area
  ○ We know many more researchers and practitioners whose work we can refer

We hope to build a community of researchers and practitioners

See us (if you like) at SC’19 for a full-day tutorial on this + more topics!
Specifics of this tutorial

- **FPChecker**
  - Helps detect FP Exceptions on GPUs
  - **Outcome:** You can use it on your Clang-based GPU projects today!

- **FLiT**
  - Helps diagnose why your compiler optimization produces unacceptable answers!
  - **Outcome:** You can apply it in the context of your CPU projects today!
    - No Clang or Intel dependency!

- **Precimonious**
  - Learn the benefit of precision tuning on actual code
  - **Outcome:** You may apply it in the context of your Clang-based CPU codes today!

- **Adapt**
  - Learn what Automatic Differentiation is, plus how it helps tune precision
  - **Outcome:** You may apply it in the context of your CPU codes today!
    - No Clang, Intel, or CPU specificity
Access to AWS Instances

- You will be given access to AWS instances
  - User, password, and IP address will be provided
- How to access your instance:
  
  `ssh user@1.2.3.4`

- Exercises for each module located in user’s /home directory
Tutorial Material

- **PEARC19**, Chicago, Illinois, USA, Jul 30th, 2019
- **SC19**, Denver, Colorado, USA, Nov 17th, 2019

Background

Dealing with floating-point arithmetic to perform numerical computations is challenging. Not only do round-off errors occur and accumulate at all levels of computation, but also compiler optimizations and low precision arithmetic can significantly affect the final computational results. With accelerators dominating HPC systems, computational scientists are faced with even bigger challenges to program reliable and reproducible floating-point programs. This tutorial demonstrates tools that are available today to analyze floating-point scientific software.

Tools

**FPChecker**

FPChecker is a tool to detect floating point exceptions (e.g., NaNs, division by zero) on NVIDIA GPUs. It uses clang and LLVM to compile and instrument CUDA kernels. The tool informs users the location where exceptions occurred (file and line number).

https://github.com/LUM/FPChecker

**FLIT**

FLIT (Floating-point Linus Testbed) is a C++ test infrastructure for detecting variability in floating-point code caused by variations in compiler code generation, hardware, and execution environments.

https://github.com/PARAINES/FLIT

**ARCHER**

ARCHER is a data race detector for OpenMP programs. ARCHER combines static and dynamic techniques to identify data races in large OpenMP applications, leading to low runtime and memory overheads, while still offering high accuracy and precision.

https://github.com/PARAINES/archer

Website & Schedule

Tutorial on Floating-Point Analysis Tools

PEARC19, Chicago, Illinois, USA

Jul 30th, 2019

Time: 1:30pm-5:00pm (Tutorial Half-day)

<table>
<thead>
<tr>
<th>Time</th>
<th>Module</th>
<th>Presenter</th>
<th>Slides</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:30pm - 1:40pm</td>
<td>Introduction</td>
<td>Ganesh, Ignacio</td>
<td>slides</td>
</tr>
<tr>
<td>1:40pm - 2:20pm</td>
<td>FPChecker</td>
<td>Ignacio</td>
<td>slides, source</td>
</tr>
<tr>
<td>Key Topics:</td>
<td>Floating-point exceptions, GPUs, CUDA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2:20pm - 3:00pm</td>
<td>FLIT</td>
<td>Ganesh, Mike, lan</td>
<td>slides, source</td>
</tr>
<tr>
<td>Key Topics:</td>
<td>Compiler optimizations, floating-point variability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:00pm - 3:30pm</td>
<td>Break</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:30pm - 4:10pm</td>
<td>Precimomous</td>
<td>Cindy</td>
<td>slides, source</td>
</tr>
<tr>
<td>Key Topics:</td>
<td>Floating-point mixed-precision, tuning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4:10pm - 4:50pm</td>
<td>ADAPT</td>
<td>Harshitha</td>
<td>slides, source</td>
</tr>
<tr>
<td>Key Topics:</td>
<td>Algorithmic differentiation, input sensitivity</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

http://fpanalysisistools.org/