PEARC’19 Half Day Tutorial

Floating-Point Analysis Tools

Ignacio Laguna, Harshitha Menon
Lawrence Livermore National Laboratory

Michael Bentley, Ian Briggs, Ganesh Gopalakrishnan
University of Utah

Cindy Rubio González
University of California at Davis

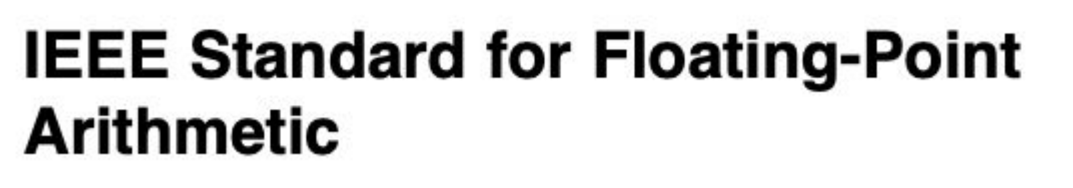
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-PRES-780623).

The Floating-Point number system is not new

Then

Zuse Z1 (~1938)

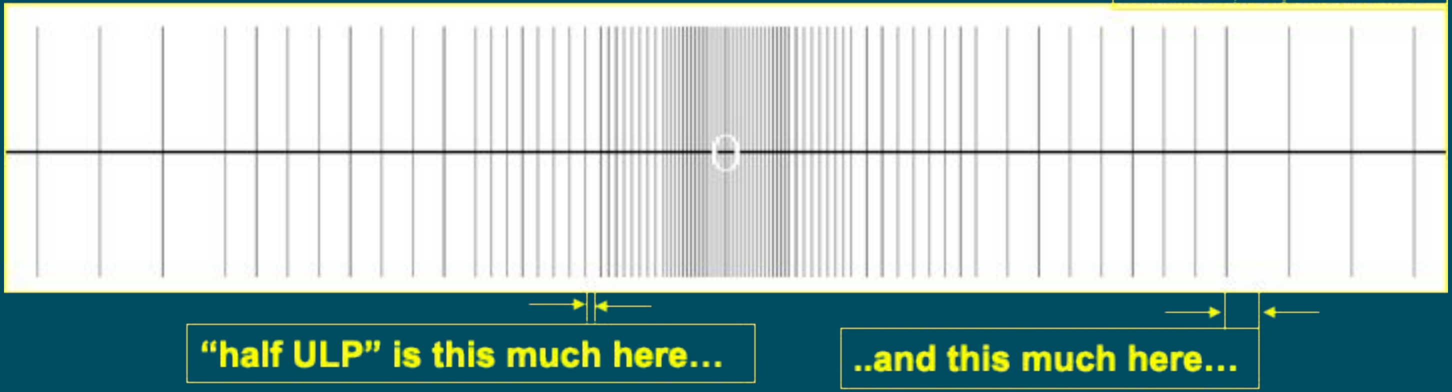
Now



IEEE Standard for Floating-Point Arithmetic

The Floating-Point Rounding is Non-Intuitive

From Goualard, "...Midpoint of an interval..."

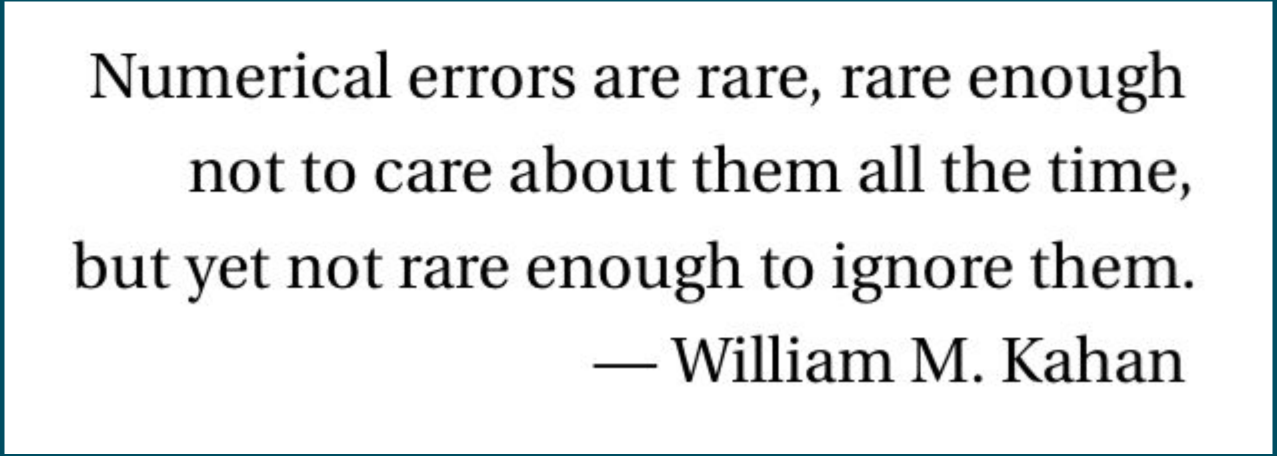


"half ULP" is this much here...

..and this much here...

What FP number scale really looks like :-)

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

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/GPU codes today!
        - No Clang, Intel, or CPU/GPU specificity

Access to AWS Instances

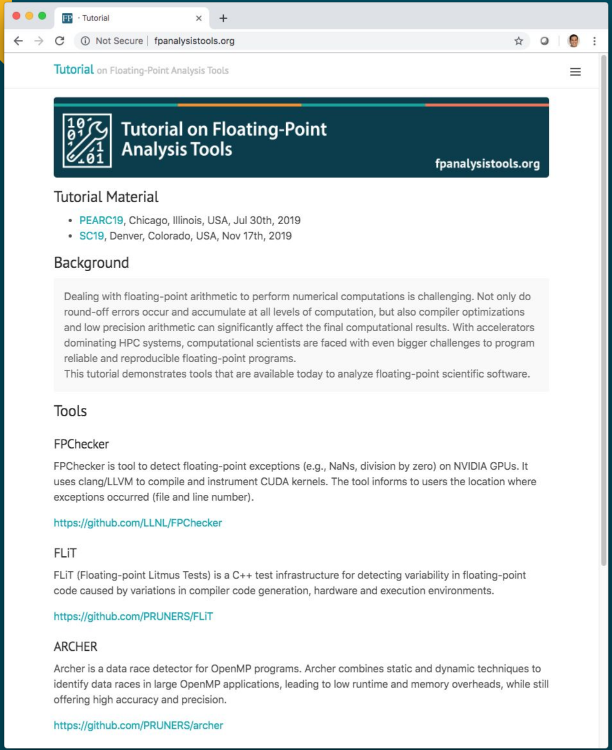
- You will be given access to AWS instances
  - User, password, and IP address will be provided
- How to access your instance:

```
ssh user@1.2.3.4
```

- Exercises for each module located in user's /home directory

```
/home/user1/
   |---Module-TOOL1
      |---exercise-1
      |---exercise-2
      |---exercise-3
   |---Module-TOOL2
      |---exercise-1
      |---exercise-2
      |---exercise-3
   ...
```

Website & Schedule


FP · Tutorial
Not Secure | fpanalysistools.org
Tutorial on Floating-Point Analysis Tools
Tutorial on Floating-Point Analysis Tools
fpanalysistools.org

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

https://github.com/LLNL/FPChecker

FLiT

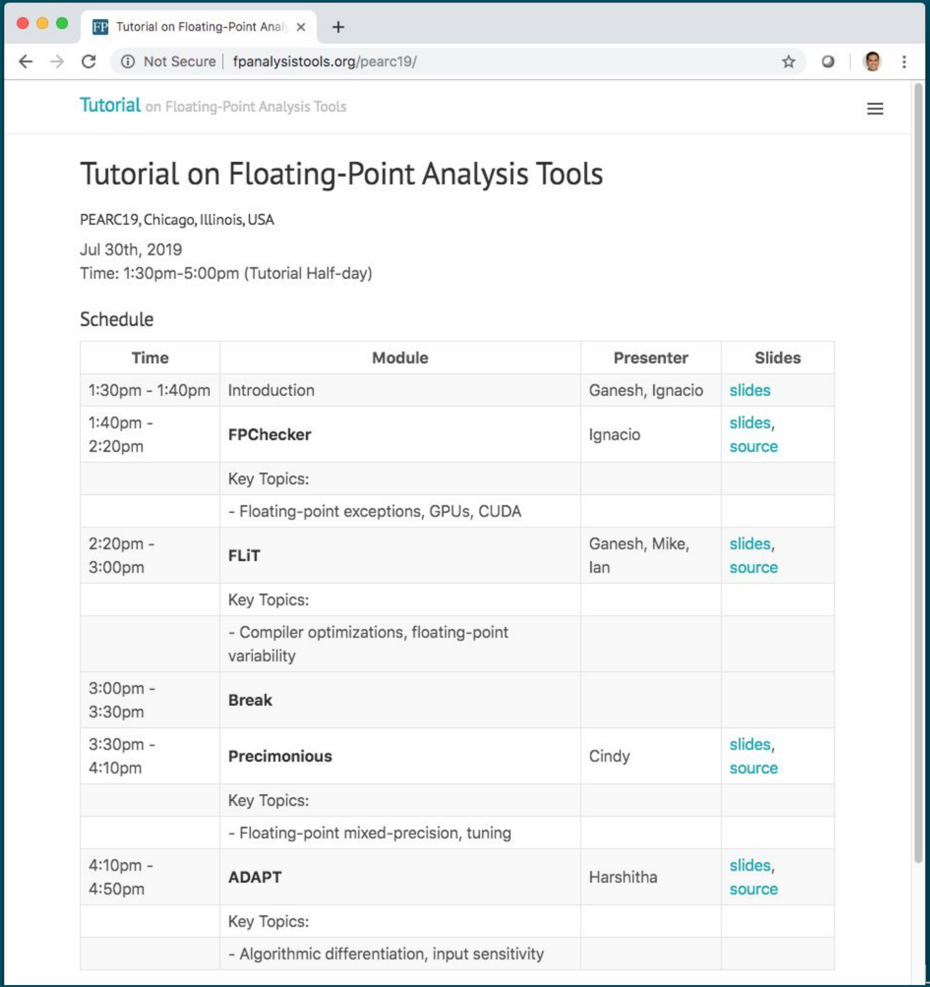
FLiT (Floating-point Litmus Tests) is a C++ test infrastructure for detecting variability in floating-point code caused by variations in compiler code generation, hardware and execution environments.

https://github.com/PRUNERS/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/PRUNERS/archer


FP Tutorial on Floating-Point Anal
Not Secure | fpanalysistools.org/pearc19/
Tutorial on Floating-Point Analysis Tools

Tutorial on Floating-Point Analysis Tools

PEARC19, Chicago, Illinois, USA

Jul 30th, 2019
Time: 1:30pm-5:00pm (Tutorial Half-day)

Schedule

Time	Module	Presenter	Slides
1:30pm - 1:40pm	Introduction	Ganesh, Ignacio	slides
1:40pm - 2:20pm	**FPChecker**	Ignacio	slides, source
	Key Topics:		
	- Floating-point exceptions, GPUs, CUDA		
2:20pm - 3:00pm	**FLiT**	Ganesh, Mike, Ian	slides, source
	Key Topics:		
	- Compiler optimizations, floating-point variability		
3:00pm - 3:30pm	**Break**		
3:30pm - 4:10pm	**Precimonious**	Cindy	slides, source
	Key Topics:		
	- Floating-point mixed-precision, tuning		
4:10pm - 4:50pm	**ADAPT**	Harshitha	slides, source
	Key Topics:		
	- Algorithmic differentiation, input sensitivity		

http://fpanalysistools.org/