Numerical Reproducibility based on Minimal-Precision Validation

RIKEN Center for Computational Science (R-CCS) (Japan)
Toshiyuki Imamura, Daichi Mukunoki, Yiyu Tan,

Sorbonne University, CNRS, LIP6 (France)
Fabienne Jézéquel, Stef Graillat, Roman Iakymchuk
What does Reproducibility refer to?

In computational science, reproducibility is considered from several viewpoints depending on the context and demand. Reproducibility refers to a capability of obtaining the identical result, but it often means ``re-playability" or ``re-traceability".

- **Bit-level reproducibility**
  is the capability to reproduce the bit-wise identical result with the same input on any HW/SW configuration. No general approach for any floating-point computation has been proposed yet. It is non-realistic to support bit-level reproducibility on all floating-point computations through the existing approaches.

- **Weak numerical reproducibility**
  the reproducibility, (up to a high probability) of the computation result with a certain accuracy demanded by the user. The underlying numerical validation is performed using a statistical approach that estimates with a high probability the number of correct digits in the computation result.

→ The extension of our minimal-precision computing scheme, which validates the accuracy (demanded by the user) of the result through the minimal-precision use.
The minimal-precision computing
high-performance and energy-efficient as well as reliable (accurate, reproducible, and validated) computations

systematic approach combining internally
1. a precision-tuning method based on Discrete Stochastic Arithmetic (DSA),
2. arbitrary-precision arithmetic libraries,
3. fast and accurate numerical libraries, and
4. Field-Programmable Gate Array (FPGA) with High-Level Synthesis (HLS)

Reliable, General, Comprehensive, High-performance, Energy-efficient, Realistic
Main software/hardware components for minimal-precision computing system:

1. **Arbitrary-precision arithmetic library**
   - MPFR (GNU)

2. **Precision-tuning method based on stochastic arithmetic**
   - Stochastic libraries: CADNA & SAM *(Sorbonne U.)*
   - Precision-tuner: PROMISE *(Sorbonne U.)*

3. **Fast & accurate numerical libraries**
   - Accurate BLAS: ExBLAS *(Sorbonne U.)*, OzBLAS *(TWCU/RIKEN)*
   - Quadruple-precision BLAS and Eigen solver: QPBLAS/QPEigen *(JAEA/RIKEN)*
   - Other open source (QD, MPLAPACK, etc.)

4. **Heterogeneous system with FPGA**
   - FPGA-GPU-CPU system: “Cygnus” *(U. Tsukuba)*
   - Compilers: SPGen *(RIKEN)*, Nymble *(TU Darmstadt/RIKEN)*
each operation executed **3 times** with a random rounding mode

**number of correct digits in the results estimated using Student’s test with the probability 95%**

estimation may be invalid if both operands in a multiplication or a divisor are not significant.

⇒ control of multiplications and divisions: *self-validation* of DSA.

in DSA rounding errors are assumed centered.
even if they are not rigorously centered, the accuracy estimation can be considered correct up to 1 digit.
CADNA: for programs in single and/or double precision

SAM: for arbitrary precision programs (based on MPFR)
http://www-pequan.lip6.fr/~jezequel/SAM

estimate accuracy and detect numerical instabilities
provide stochastic types (3 classic type variables and 1 integer):
  - float_st in single precision
  - double_st in double precision
  - mp_st in arbitrary precision

all operators and mathematical functions overloaded
⇒ few modifications in user programs
Main software/hardware components for minimal-precision computing system:

1. **Arbitrary-precision arithmetic library**
   - MPFR (GNU)

2. **Precision-tuning method based on stochastic arithmetic**
   - Stochastic libraries: CADNA & SAM (Sorbonne U.)
   - Precision-tuner: PROMISE (Sorbonne U.)

3. **Fast & accurate numerical libraries**
   - Accurate BLAS: ExBLAS (Sorbonne U.), OzBLAS (TWCU/RIKEN)
   - Quadruple-precision BLAS and Eigen solver: QPBLAS/QPEigen (JAEA/RIKEN)
   - Other open source (QD, MPLAPACK, etc.)

4. **Heterogeneous system with FPGA**
   - FPGA-GPU-CPU system: “Cygnus” (U. Tsukuba)
   - Compilers: SPGen (RIKEN), Nymble (TU Darmstadt/RIKEN)
Accurate/ Reproducible BLAS (ExBLAS)

Highlights of the Algorithm

- Parallel algorithm with 5-levels
- Suitable for today’s parallel architectures
- Based on FPE with EFT and Kulisch accumulator
- Guarantees “inf” precision → bit-wise reproducibility
Accurate & reproducible dot-product ($x^T y$)

The vectors can be split recursively until $x^{(p)}$ and $y^{(q)}$ become zero

\[
x = x^{(1)} + x^{(2)} + x^{(3)} + \cdots + x^{(p-1)} + x^{(p)}
\]
\[
y = y^{(1)} + y^{(2)} + y^{(3)} + \cdots + y^{(q-1)} + y^{(q)}
\]

$x^T y$ is transformed to the sum of multiple dot-products

\[
x^T y = (x^{(1)})^T y^{(1)} + (x^{(1)})^T y^{(2)} + (x^{(1)})^T y^{(3)} + \cdots + (x^{(1)})^T y^{(q-1)}
+ (x^{(2)})^T y^{(1)} + (x^{(2)})^T y^{(2)} + (x^{(2)})^T y^{(3)} + \cdots + (x^{(2)})^T y^{(q-1)}
+ (x^{(3)})^T y^{(1)} + (x^{(3)})^T y^{(2)} + (x^{(3)})^T y^{(3)} + \cdots + (x^{(3)})^T y^{(q-1)}
+ \cdots
+ (x^{(p-1)})^T y^{(1)} + (x^{(p-1)})^T y^{(2)} + (x^{(p-1)})^T y^{(3)} + \cdots + (x^{(p-1)})^T y^{(q-1)}
\]

Those computations can be performed using standard BLAS (e.g., MKL, OpenBLAS, cuBLAS)

Productive & High-performance
Main software/hardware components for minimal-precision computing system:

1. **Arbitrary-precision arithmetic library**
   - MPFR (GNU)

2. **Precision-tuning method based on stochastic arithmetic**
   - Stochastic libraries: CADNA & SAM *(Sorbonne U.)*
   - Precision-tuner: PROMISE *(Sorbonne U.)*

3. **Fast & accurate numerical libraries**
   - Accurate BLAS: ExBLAS *(Sorbonne U.)*, OzBLAS *(TWCU/RIKEN)*
   - Quadruple-precision BLAS and Eigen solver: QPBLAS/QPEigen *(JAEA/RIKEN)*
   - Other open source (QD, MPLAPACK, etc.)

4. **Heterogeneous system with FPGA**
   - FPGA-GPU-CPU system: “Cygnus” *(U. Tsukuba)*
   - Compilers: SPGen *(RIKEN)*, Nymble *(TU Darmstadt/RIKEN)*
FPGA Performance enhancement

- **SPGen (RIKEN)**
  - a compiler to generate HW module codes in Verilog-HDL for FPGA from input codes in Stream Processing Description (SPD) Format.
  - a data-flow graph representation, which is suitable for FPGA.
  - it supports FP32 only, but we are going to extend SPGen to support arbitrary-precision floating-point.

- **Nymble (TU Darmstadt, RIKEN)**
  - another compiler project for FPGA. It **directly accepts C codes** and has already started to support arbitrary-precision.
Minimal-Precision Computing - System Workflow

Input:
C code with MPFR (and MPLAPACK)

Precision-Optimizer
(with PROMISE and CADNA/SAM)

Precision-Optimizer
• The Precision-Optimizer determines the minimal floating-point precisions, which need to achieve the desired accuracy

C code with MPFR (optimized)

Performance Optimization
• At this stage, if possible to speedup some parts of the code with some other accurate computation methods than MPFR, those parts are replaced with them
• The required-accuracy must be taken into account
• If possible, it considers to utilize FPGA (as heterogeneous computing)

FPGA?
Yes
A part of the C code with MPFR, which is executed on FPGA

No
C code with MPFR + other fast accurate methods

Code Translation for FPGA
(SPGen, Nymble, FloPoCo)

Low-level code for FPGA (VHDL etc.)

Compilation and Execution on CPU/GPU

Compilation and Execution on FPGA

Low-level code for FPGA (VHDL etc.)

Compilation and Execution on FPGA

Low-level code for FPGA (VHDL etc.)
1. The minimal-precision computing system => a black box
   • Though *different paths for execution may be used* either to speed up computations and/or ensure energy-efficiency, required precision is guaranteed.

2. Validation of the requested accuracy of the computation demanded by the user
   • If the computation method can achieve the required result, any methods, any computation environments, and any computation conditions can be accepted.
   • *No longer need to develop some reproducible variant(s)* for each computation method or mathematical problem.

3. Comparing with re-playable and re-traceable methods
   • easier to adapt to different (parallel) architectures.
   • Existing methods and software *for ensuring bit-level reproducibility are still able to contribute to ensure the demanded accuracy*, if such method relies on some accurate method.
A new concept of weak numerical reproducibility

the reproducibility, (up to a high probability) of the computation result with a certain accuracy demanded by the user.

A systematic approach for it with a support of minimal-precision tuning and validation.

The concept of weak numerical reproducibility covers most of the demands for reproducibility in computational sciences.

Besides, if it has been realized with new hardware like FPGAs, the minimal-precision computing system can address the demands for accuracy, high-performance, and energy efficient computation as well.

Future work is Demonstration of weak numerical reproducibility.

Please see also:
Poster 134: Minimal-Precision Computing for High-Performance, Energy-Efficient, and Reliable Computations
The authors would like to thank
- Dr. Kentaro Sano and Dr. Yiyu Tan from RIKEN CCS
- Prof. Taisuke Boku and Dr. Norihisa Fujita from CCS, University of Tsukuba

Partially supported by
- the European Unions Horizon 2020 research, innovation programme under the Marie Skodowska-Curie grant agreement via the Robust project No. 842528,
- the Japan Society for the Promotion of Science (JSPS) KAKENHI Grant No. 19K20286,
- Multidisciplinary Cooperative Research Program in CCS, University of Tsukuba.