CudA Multiple Precision ARithmetic librarY

Mioara Joldes, Jean-Michel Muller, Valentina Popescu
team AriC, LIP Laboratory, ENS-Lyon

Floating-point (FP) arithmetic

Representation: \( x = M_x \cdot 2^{e_x} \)

- With \( \frac{2^p - 1}{2} \leq |M_x| \leq 2^p - 1 \), where:
  - \( p \) is the precision;
  - \( e_x \) is the exponent;
  - \( M_x \cdot 2^{-p+1} \) is the significand.

\( \rightarrow \) Standard: single (binary32) and double (binary64).

Motivation

- Need more precision than standard available:
  - few hundred bits
- Need massive parallel computations
- Dynamical systems: bifurcation analysis, compute periodic orbits, long term stability of the solar system.

Multiple-digit representation

- sequence of digits coupled with a single exponent

Example: GNU MPFR (open-source C library)
  - arbitrary precision
  - correct rounding for each atomic operation

Multiple-term representation (FP Expansions)

- unevaled sum of several standard FP numbers

Example: QD library, with the analogue GPU version GQD
  - uses available, highly optimized hardware FP operations
  - straightforwardly portable to highly parallel architectures

Error-Free Transforms

\( s + e = a + b \)

\( p + e = a + b \)

CAMPARY algorithms

Addition/Subtraction

- Truncated addition/multiplication with FP expansions

Multiplication

- Error analysis \( \rightarrow \) quadratical convergence error bounds

Reciprocal/Division & Square root

Based on an adapted Newton-Raphson iteration.

Newton iteration for root of \( f \):

- \( x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \)

Consider FP expansion \( a = a_0 + \cdots + a_n \cdot 2^{-n} \)

\( \rightarrow \) iterations:

Reciprocal \( \frac{1}{x} \):

- \( x_{i+1} = x_i (2 - ax_i) \)

Square root \( \sqrt{a} \):

- \( x_{i+1} = \frac{1}{2} x_i (3 - ax_i^2) \)

\( \rightarrow \) Truncated addition/multiplication with FP expansions

\( |\frac{x-e}{x} - 1| \leq 2^{-2(p-3)-1} \)

\( |x - \frac{2}{x} + \frac{1}{2} \frac{3}{x^3} - 1| \leq 2^{-3(p-2)-1} \)

Implementation details:

- templated class implemented in CUDA C;
- templates for both the number of terms in the expansion and the native type of the terms (allows static generation of any input/output precision combinations);
- functions defined using \( \text{host}_\text{device}_\text{specifiers} \);
- in-place algorithms to avoid registers spill/loads;
- fully customized operations with variable size expansions;
- interval arithmetic supported in a similar class;
- a normalization algorithm is used to render the result non-overlapping.

GPUs - Graphics Processing Units

- parallelism

Implementation details:

- templated class implemented in CUDA C;
- templates for both the number of terms in the expansion and the native type of the terms (allows static generation of any input/output precision combinations);
- functions defined using \( \text{host}_\text{device}_\text{specifiers} \);
- in-place algorithms to avoid registers spill/loads;
- fully customized operations with variable size expansions;
- interval arithmetic supported in a similar class;
- a normalization algorithm is used to render the result non-overlapping.