Demanding quality for numerical computations using floating-point arithmetic

When a mathematical formula is translated into a numerical computation, it is hoped that computed results are close to the corresponding exact value. However, computers usually employ floating-point arithmetic; the representation of numbers has a finite fixed size. Consequently, rounding errors are made. The first goal of the EVA-Flo project is to evaluate numerically a formula in a fast and accurate way. The quality of the result can be specified, such as a relative or absolute error between the exact value and the computed result, or the guarantee that no overflow occurs (numbers too large to be represented and converted into infinity). The second goal of the EVA-Flo project is that this quality can be quantified (for instance “the relative error is $\frac{\Delta}{x}$”) and certified. The last goal is that this process of evaluation and validation is automated.

The target mathematical formulas of the EVA-Flo project are expressed using arithmetic or algebraic operations and mathematical functions (cos, sin, arctan, ...), and they can contain few conditional branches and loops. Typically, the focus is on small critical portions of large numerical codes.

The mathematical model, as illustrated here by the function on the left, can correspond to an implementation (center) that poorly approximates it. The goal is to obtain a better implementation, such as the one on the right, and as automatically as possible.

Taming roundoff errors... as well as other numerical errors made by your computer

Automate, automate, automate the accumulated expertise

Numerous problems have been handled in a pen and paper manner in the past. The current step consists in automating, at each level, the expertise gained through the handling of these problems. The first level is to specify precisely the desired mathematical result and to determine good approximants (e.g., with a small relative error) that are well-suited to an implementation on a computer. Typically, these approximants are polynomials with floating-point coefficients. The second level is to determine evaluation schemes that are both fast and accurate, using exhaustive search. The architecture of the target processor plays a key role here. Another level is the accuracy of the technique employed to reach the required accuracy:double arithmetic, compensated evaluation schemes... Both method error and implementation error are then bounded and certified. Usually, the method error is estimated but not bounded, whereas the implementation error is rarely handled. Such proofs on the quality of the computed result use a fine knowledge of the properties of the floating-point arithmetic, they are also based on computations using interval arithmetic and extended-precision arithmetic. Then, a proof is reworked and written so that a proof checker can check it, we use the Coq proof checker. Indeed, a typical proof includes many peculiar cases and is thus error-prone when it is performed by a human. This explains why it is essential to check it automatically.

Most significant outcomes

Software production

Research tools have been developed for use with the development of software package:

- Sollya: determination of a good polynomial approximation, including a guaranteed approximation error;
- Gappa (mainly developed prior to EVA-Flo): bounds on evaluation errors, that can be checked by the proof checker Coq;
- Circom (mainly developed prior to EVA-Flo): correctly rounded transcendental functions; this proof of concept library qualified in the recommendation, in the IEEE 754-2008 standard. But elementary functions should be correctly rounded: large parts of its current code are automatically generated by an experimental tool;
- FloatRec: a VHDL code generator for Floating-Point Ores on FPGAs, with application-specific optimizations for non-standard operators;
- SplitSim: software simulation of IEEE-754 floating-point arithmetic on some embedded media processors;
- CSQP: Code Generation for Polynomial Evaluation, taking into account architectural features;
- Fluctuat (developed independently and prior to EVA-Flo): analysis of the numerical quality of scientific codes;
- Tapenade (developed independently and prior to EVA-Flo): automatic differentiation of codes;
- Sardanes project: analysis and rewriting of mathematical expressions to achieve better accuracy.

Scientific production

Apart from the software developments already mentioned, 6 PhD theses, related to these topics, have been defended. Around 15 articles in scientific journals and 20 presentations at international conferences have been produced. Currently, the expertise of the group on floating-point arithmetic has given rise to a collective book, The Handbook of Floating-Point Arithmetic, published by Birkhäuser in November 2009. It sets out the methodology of automated EVA-Flo through their expertise in automated validation of floating-point codes.

The EVA-Flo project: Evaluation et Validation Automatique pour le calcul Flottant - New Automatic Tools for Validated Floating-point Computations is an ANR Blanc-STIC research project. It is headed by Arénaire (LIP, ENS Lyon). It associates Dali (Eliaus, U. Perpignan), MeASI (LIST, U. Côte d’Azur) and Tropics (INRIA Sophia Antipolis - Méditerranée). The project started in November 2006 and its duration is 44 months. It is subsidized by ANR: 134 K€ and the global cost is 1.5 M€.