

The Future of Floating-Point Computing is in Reliable Computing

Presenter:

Ruud van de Pas, Scalable Systems Group, Sun Microsystems

Background:

As computers get more powerful, more accurate numerical simulations of all sorts of real-world phenomena can be conducted. Drug design, weather prediction, risk analysis and car design are just a few examples.

Very often, these simulations are based on floating-point arithmetic. Outside of pathological cases, the application will run to completion and the results are analyzed. The problem is that for real-life problems, nobody is able to assert the quality of these numerical results.

Valid questions to ask are: how about the propagation of round-off error? Did the algorithm converge to the right solution? Are we able to detect a bug in the application? Can we use numerical results to falsify a theory?

Current practice is to use a fixed test suite to validate the simulation software. If the outcome on this suite is within the specified criteria, the software is assumed to be correct.

It is clear that this approach is risky. One can not capture all possible cases in this way and can only hope for the best.

History has shown that the price we pay is high.

For example, during the Gulf War in 1991, an American Patriot Missile Battery failed to track and intercept an incoming Scud missile, resulting in the loss of 28 lives. The cause was traced back to an inaccurate calculation of the time since boot due to computer arithmetic errors.

Another example is the explosion of the Ariane 5 rocket in 1996. This happened forty seconds after lift-off. The cost of the lost rocket only was already \$500 million. The cause of the failure was in the inertial reference system. A 64 bit floating point number relating to the horizontal velocity of the rocket was converted to a 16 bit signed integer. Unfortunately, the number did not fit in 16 bits and the conversion failed. A very expensive mistake.

Reliable Computing is a rather general term. We present it in the context of Interval Analysis and Interval Arithmetic, which provide for a much better and more solid understanding of the quality of the results of a computation.

Thanks to these techniques one can also solve problems that can not be tackled with conventional single number floating-point arithmetic.

In this talk we go into the underlying mechanisms that allow us to perform these kind of improved computations. In other words, why is this approach so much better than the current practice?

Next we show examples how this technology can be leveraged to provide better insight in the quality of a numerical simulation.

At the end of this talk, attendees will hopefully have a good understanding of the risks associated with the way we do floating-point computations today and also know what Reliable Computing offers to improve upon this.