PROMISE: floating-point precision tuning with stochastic arithmetic

Stef Graillat[1], Fabienne Jézéquel[1,2], Romain Picot[1,3],
François Févotte[3] and Bruno Lathuilière[3]

[1] Sorbonne Universités, UPMC Univ Paris 06, CNRS, UMR 7606,
Laboratoire d'Informatique de Paris 6 (LIP6), Paris, France
[2] Université Panthéon-Assas, 12 place du Panthéon, 75231 Paris
CEDEX 05, France
[3] EDF R&D 7 boulevard Gaspard Monge, F-91120, Palaiseau, France
`{Stef.Graillat,Fabienne.Jezequel,Romain.Picot}@lip6.fr`
`{Francois.Fevotte,Bruno.Lathuiliere}@edf.fr`

Nowadays, most floating-point computations in numerical simulations are performed in IEEE 754 `binary64` precision (double precision). This means that a relative accuracy of about 10^{-16} is provided for every arithmetic operation. Indeed, in practice, programmers tend to use the highest precision available in hardware which is the double precision on current processors. This approach can be costly in terms of computing time, memory transfer and energy consumption [1]. A better strategy would be to use no more precision than needed to get the desired accuracy on the computed result. The challenge of using mixed precision is to find some parts of codes (and so variables) that may be executed with lower precision. Unfortunately the amount of possible configurations is exponential in the number of variables.

To overcome this difficulty, we propose an algorithm and a tool called PROMISE (PRecision OptiMISEd) based on the delta debugging search algorithm [2] that provide a mixed precision configuration with a worst-case complexity quadratic in the number of variables.

variables. To estimate the numerical quality of results, PROMISE uses Discrete Stochastic Arithmetic (DSA) [3] which controls round-off errors in simulation programs. Unlike Precimonious [4], PROMISE does not focus on performance constraints, it can rather be seen as a tool helping developers to reduce the cost of double precision variables and improve the memory usage of their code. The PROMISE tool has been successfully tested on programs implementing several numerical algorithms including linear system solving and also on an industrial code that solves the neutron transport equations [5].

References:

[1] M. BABOULIN, A. BUTTARI, J. DONGARRA, J. KURZAK, J. LANGOU, J. LANGOU, P. LUSZCZEK, AND S. TOMOV, Accelerating scientific computations with mixed precision algorithms, *Computer Physics Communications*, vol. 180, No. 12, pp. 2526–2533, 2009.

[2] A. ZELLER AND R. HILDEBRANDT, Simplifying and isolating failure-inducing input, *IEEE Trans. Softw. Eng.*, vol. 28, No. 2, pp. 183–100, Feb. 2002.

[3] J. VIGNES, Discrete Stochastic Arithmetic for validating results of numerical software, *Numerical Algorithms*, vol. 37, no. 1–4, pp. 377– 390, Dec. 2004.

[4] C. RUBIO-GONZÁLEZ, C. NGUYEN, H. D. NGUYEN, J. DEMMEL, W. KAHAN, K. SEN, D. H. BAILEY, C. IANCU, AND D. HOUGH, Precimonious: Tuning assistant for floating-point precision, *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC'13. New York, NY, USA: ACM, 2013, pp. 27:1– 27:12.

[5] F. FÉVOTTE AND B. LATHUILIÈRE, MICADO: Parallel implementation of a 2D–1D iterative algorithm for the 3D neutron transport problem in prismatic geometries, *Proceedings of Mathematics,*