Master Internship Proposal

Exploring Accuracy and Performance Trade-offs in Functional Array Programs

Location	Department of Information Technology, Uppsala, Sweden or Team Inria CAMUS, ICube Laboratory, Strasbourg (Illkirch campus), France
Advisors	Eva Darulova, Uppsala University Thomas Kœhler, CNRS

1 Context

Program optimization is crucial in high performance computing domains such as image processing, physics simulation, and artificial intelligence. An optimized program is faster, consumes less memory and energy. Concretely, optimization allows to process higher resolution images, to increase the realism of simulations, and to reduce the carbon footprint of artificial intelligence. However, program optimization is a difficult task that is made even more challenging by the fact that programs frequently approximate exact arithmetic using finite precision number representations: e.g. floating-point numbers and fixed-point arithmetic.

On one hand, many high-performance optimizing compilers simply ignore finite-precision rounding errors, treat them as real values [1, 2] and optimize purely for performance. On the other hand, typical general-purpose compilers will not apply optimizations that may affect floating-point results. For example, gcc/clang's O3 optimization level will not re-arrange computations based on associativity; optimizations affecting floating-point behaviour must be explicitly enabled via the fast-math flag and the developer is responsible for any resulting issues.

A few tools explore trade-offs between performance and accuracy, but are fairly limited. Some optimize arithmetic expressions [3, 4, 5], select between explicitly programmed algorithm alternatives [6], or dynamically skip loop iterations [7]. Except for tools exclusively targeting reconfigurable hardware [8], there is no tool today that explores performance and accuracy trade-offs when applying high-performance optimizations to programs with loops and arrays.

The goal of this internship is to remedy this situation by building an empirical design space exploration tool. To achieve this, the Shine compiler [2, 9] will be used to compile an input program with many different optimizations. The resulting programs will be evaluated empirically in terms of performance and accuracy. The Shine compiler takes as input programs defined in the functional array language called Rise, and outputs imperative code (e.g. C, OpenCL, or CUDA).

2 Objectives

We envision the following steps for the internship:

1. Design a process to generate many programs in the design space. This will be facilitated by the fact that Rise optimizations are not hard-coded, but expressed as rewrite rules whose application can be explored using various rewriting techniques.

2. Experimentally evaluate the performance and accuracy of the generated C code on simple benchmarks. Accuracy will be evaluated using so-called shadow execution that executes a floating-point program in a higher precision, e.g. using the MPFR library, side-by-side [10] and estimates the rounding error as the difference in final results. Analyse the results and produce pareto fronts to draw conclusions on the relationship between performance and accuracy.

3. Ideally, tackle benchmarks that require adding new Rise rewrite rules, for example to make use of different algorithms or tune the precision of individual variables and operations. Floating-point sums [11], reductions [12] and scans [13] would be interesting benchmarks as their parallelisation requires associativity properties found in reals but not in floating-points.

3 Required and Acquired Skills

The intern should come with:

- solid programming skills, familiarity with imperative and functional paradigms
- interest in (or experience with) design space exploration, compilation and optimization tools
- willingness to learn (or knowledge of) Scala and C

The intern is expected to acquire:

- experience in basic academic skills: reviewing literature, conducting novel research, collaborating with other researchers, presenting research and communicating ideas
- knowledge in floating-point accuracy, term rewriting and code generation

4 Practical Aspects

The internship can either take place in Uppsala (Sweden), or in Strasbourg (France), and a brief visit of the other location could be organized during the internship. If you are interested, send us an e-mail with a short statement of interest, questions you might have, the desired start date and duration of your internship (4-6 months), as well as a 1 page CV and your Master transcripts. If the internship goes well, there is potential to pursue a PhD in similar topics.

References

[1] Jonathan Ragan-Kelley, Andrew Adams, Sylvain Paris, Marc Levoy, Saman P. Amarasinghe, and Frédo Durand. “Decoupling algorithms from schedules for easy optimization of image processing pipelines”. In: *ACM Trans. Graph.* 31.4 (2012), 32:1–32:12. URL: https://doi.org/10.1145/2185520.2185528.

[2] Michel Steuwer, Thomas Kœhler, Bastian Köpcke, and Federico Pizzuti. *RISE & Shine: Language-Oriented Compiler Design.* 2022. arXiv: 2201.03611 [cs.PL].

[3] Nasrine Damouche and Matthieu Martel. “On the Impact of Numerical Accuracy Optimization on General Performances of Programs”. In: *5th International Conference on Control, Decision and Information Technologies, CoDIT 2018, Thessaloniki, Greece, April 10-13, 2018.* IEEE, 2018, pp. 333–340. URL: https://doi.org/10.1109/CoDIT.2018.8394897.

[4] Brett Saiki, Oliver Flatt, Chandrakana Nandi, Pavel Panchekha, and Zachary Tatlock. “Combining Precision Tuning and Rewriting”. In: *28th IEEE Symposium on Computer Arithmetic, ARITH 2021, Lyngby, Denmark, June 14-16, 2021.* IEEE, 2021, pp. 1–8. URL: https://doi.org/10.1109/ARITH51176.2021.00013.

[5] Eva Darulova, Einar Horn, and Saksham Sharma. "Sound mixed-precision optimization with rewriting". In: *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems, ICCPS 2018, Porto, Portugal, April 11-13, 2018.* Ed. by Chris Gill, Bruno Sinopoli, Xue Liu, and Paulo Tabuada. IEEE, 2018, pp. 208–219. URL: https://doi.org/10.1109/ICCPS.2018.00028.

[6] Jason Ansel, Yee Lok Wong, Cy P. Chan, Marek Olszewski, Alan Edelman, and Saman P. Amarasinghe. "Language and compiler support for auto-tuning variable-accuracy algorithms". In: *Proceedings of the CGO 2011, The 9th International Symposium on Code Generation and Optimization, Chamonix, France, April 2-6, 2011.* IEEE Computer Society, 2011, pp. 85–96. URL: https://doi.org/10.1109/CGO.2011.5764677.

[7] Maxime Schmitt, Philippe Helluy, and Cédric Bastoul. "Automatic adaptive approximation for stencil computations". In: *Proceedings of the 28th International Conference on Compiler Construction, CC 2019, Washington, DC, USA, February 16-17, 2019.* Ed. by José Nelson Amaral and Milind Kulkarni. ACM, 2019, pp. 170–181. URL: https://doi.org/10.1145/3302516.3307348.

[8] Xitong Gao, John Wickerson, and George A. Constantinides. "Automatically Optimizing the Latency, Area, and Accuracy of C Programs for High-Level Synthesis". In: *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, February 21-23, 2016.* Ed. by Deming Chen and Jonathan W. Greene. ACM, 2016, pp. 234–243. URL: https://doi.org/10.1145/2847263.2847282.

[9] Thomas Kœhler. "A domain-extensible compiler with controllable automation of optimisations". In: *arXiv preprint arXiv:2212.12035* (2022).

[10] Sangeeta Chowdhary and Santosh Nagarakatte. "Fast shadow execution for debugging numerical errors using error free transformations". In: *Proc. ACM Program. Lang.* 6.OOPSLA2 (2022), pp. 1845–1872. URL: https://doi.org/10.1145/3563353.

[11] URL: https://orlp.net/blog/taming-float-sums/.

[12] Nikolay M. Evstigneev, Oleg I. Ryabkov, A. N. Bocharov, V. P. Petrovskiy, and I. O. Teplyakov. "Compensated summation and dot product algorithms for floating-point vectors on parallel architectures: Error bounds, implementation and application in the Krylov subspace methods". In: *J. Comput. Appl. Math.* 414 (2022), p. 114434. URL: https://doi.org/10.1016/j.cam.2022.114434.

[13] Ivo Gabe de Wolff, David P. van Balen, Gabriele K. Keller, and Trevor L. McDonell. "Zero-Overhead Parallel Scans for Multi-Core CPUs". In: *Proceedings of the 15th International Workshop on Programming Models and Applications for Multicores and Manycores, PMAM 2024, Edinburgh, United Kingdom, 3 March 2024.* ACM, 2024, pp. 52–61. URL: https://doi.org/10.1145/3649169.3649248.