Master 2 internship proposal
High-performance compilation schemes using dynamic analysis

Advisor: Christophe Alias, chargé de recherche HDR, INRIA.
mail: `Christophe.Alias@ inria.fr`
web: `http://perso.ens-lyon.fr/christophe.alias`
Co-advisor: Keiji Kimura, professor, Waseda University, Tokyo, Japan.

Duration: from 4 to 6 months (stip-end ≈ 500 euros/month)

Place: Laboratoire de l'Informatique du Parallélisme (LIP)
École Normale Supérieure de Lyon

This internship can be followed by PhD thesis (funding available).

Context

Compilers must restructure the application to use as best as possible computing and storage resources. In general, parallel runtimes [2] (*interpreters*) do a better job than compilers, as the availability of dynamic informations (variable values, tests, loop iterations) make possible to make the right decisions. However, runtimes come with an overhead, which push them to *coarse-grain* task scheduling, while compilers are usually in charge of mapping the tasks to computation units (GPU, FPGA, etc) and then to extract *fine-grain* parallelism.

The POLYTRACE Inria exploratory action investigates how compilers can exploit runtime analysis on several execution traces to infer a general code optimization scheme. We focus on programs from the polyhedral model [3], where the operations of the execution trace depends only on input size and where the compilation schemes are affine functions (schedule, resource allocation, etc). We believe this will make possible to obtain levels of optimizations out-of-reach by a purely static compilation.

Goals

In this internship, we focus on the inference of a *buffer allocation function* [1] from several execution traces. Given a program manipulating a temporary array a, we want to derive the best possible allocation function σ such that any access $a[\vec{i}]$ can be safely substituted by $\texttt{buffer}[\sigma(\vec{i})]$, while minimizing the buffer size (the size of range(σ)).

- *Bibliography.* Methods for (static/dynamic) buffer allocation and piecewise affine regression.
- *Research.* On some simple examples, find how to infer an optimal mapping σ, then propose an algorithm.
- *Experimental validation.* The approach will be implemented and checked on the polyhedral community benchmarks [4].

Skills expected. Notions in compilers, experience with C++.

References

[1] Christophe Alias, Fabrice Baray, and Alain Darte. Bee+cl@k : An implementation of lattice-based array contraction in the source-to-source translator rose. In *ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'07)*, 2007.

[2] Cédric Augonnet, Samuel Thibault, Raymond Namyst, and Pierre-André Wacrenier. Starpu: a unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation: Practice and Experience*, 23(2):187–198, 2011.

[3] Paul Feautrier and Christian Lengauer. Polyhedron model. In *Encyclopedia of Parallel Computing*, pages 1581–1592. 2011.

[4] Louis-Noël Pouchet. Polybench: The polyhedral benchmark suite. *URL: http://www. cs. ucla. edu/˜ pouchet/software/polybench/[cited July,]*, 2012.