Proposition de thèse CIFRE

Polyhedral Compilation Strategies and Runtime Support for High-Performance Computing

Advisors: Christophe Alias (Inria/ENS de Lyon), Jean-Baptiste Besnard (Paratools)
mail: Christophe.Alias@inria.fr, jbbesnard@paratools.fr

Place: Paratools (Bruyères-le-Châtel)

Context

Since the early days of parallel computing, industry is pushing towards programming models, languages and compilers to help the programmer in the tedious task to parallelize a program. Task-based programming models [1, 7, 6] view the program as a composition of coarse-grain tasks to be executed in a dataflow fashion. A runtime is generally in charge of orchestrating the computation, and mapping the tasks to processing units so load balancing and data transfers are optimized. Usually, the task partitioning is driven algorithmically, (eg BLAS calls for StarPU), though better, but less intuitive, partitionings may be inferred at compile-time [2] with compilation techniques from the polyhedral model [5], a general framework to design loop transformations, historically geared towards source-level automatic parallelization and data locality improvement. Loop iterations are abstracted as a union of convex polyhedra – hence the name – and data accesses as affine functions. This way, exact – iteration-level – compiler algorithms may be designed [4, 3], enabling levels of performance far beyond classical code optimization approaches.

Goals

The overall objective of the PhD thesis is to investigate polyhedral compilation techniques to translate a sequential C kernel into a task-based program, and to guide the runtime for an optimized parallel execution. We focus on a new runtime paradigm, where processing units are viewed as data storages offering a computing service. During the execution, the processing units receive the code of the task to be executed, not the data (or as less as possible). The PhD student will address the following points:

• Propose polyhedral compilation algorithms to expose and exploit dataflow parallelism in the context of a data-driven computation placement.

• Propose a relevant runtime architecture and compilation schemes to generate automatically the runtime program.

• Investigate the extensions of the polyhedral model to handle multiscale stencil algorithms.

• Validate the proposed approach on scientific computing benchmarks.

Skills expected. Notions in compilers, parallelism, parallel architectures. Experience with C++.
References


