

Research Plans Within IFIP WG 2.11

Albert Cohen

April 15, 2004

My activities in the work-group cover two complementary research areas currently investigated in the ALCHEMY group at INRIA.

A large part of this work takes place in a collaboration with other members of the working group, including Christoph Herrmann, Christian Lengauer and Martin Griebel at the University of Passau, Germany, and with Paul Feautrier from the École Normale Supérieure in Lyon, France. This work also benefits from a collaboration with David Padua from the University of Illinois at Urbana-Champaign.

Programming Loop Nest Transformations in the Polytope Model. Research on the first topic is conducted jointly with one professor from our group, Olivier Temam, two PhD students, two Master students and one engineer. The baseline is iterative and feedback-directed optimization for high-performance and embedded architectures. Currently, iterative compilation techniques are mostly limited to finding the parameters of program transformations. We want to extend the scope and efficiency of iterative optimization techniques by searching not only for the appropriate parameters of a given transformation, but for the program transformations themselves, and especially for *compositions* of program transformations.

We introduce a framework for easily expressing compositions of program transformations. This framework combines extraction, transformation and generation tools for a polyhedral representation of programs, with a *domain-specific transformation language*. Compared to approaches based on the program syntax or on inductive program semantics (denotational or operational), the key is to clearly separate the impact of each program transformation on the following three components: the iteration domain, the schedule and the memory access functions. We show that, within this framework, composing a long sequence of loop nest transformations (including fusion, tiling, unrolling and peeling) induces no code size explosion. As a result, searching for compositions of transformations is not hampered by the multiplicity of compositions, and in many cases, it is equivalent to testing different values for the coefficients of the representation matrices. Our techniques have been implemented on top of the Open64/ORC compiler.

Combining Syntactic and Polyhedral Code Generation. Beyond applications of metaprogramming to iterative optimization, we wish to investigate the interactions and possible combinations or *multi-staged languages* with *polyhedral generation* techniques. The goal is to raise the level of abstraction and the ability to make complex choices like selecting the best algorithm and the best architecture-specific implementation for it. Instead of building code expressions from syntactic representations only, part of the control-flow of the generated code could come from polyhedral domains and schedules.

In a collaboration with our colleagues from Urbana-Champaign and Denis Barthou from the University of Versailles, we are building a two-level imperative language based on C, with constructs inherited from the syntax and semantics of MetaOCaml, supported with a library of operation research and machine learning algorithms for iterative optimization. This framework will be applied to the rapid and effective design of *adaptive libraries*, with static or dynamic code generation, versioning and selection features, and feedback-directed architecture-specific optimizations.

Simultaneously, together with our colleagues from Passau, we are investigating the potential for domain-specific and architecture-specific optimizations in parallel computing. Practically, we study the domain of *parallel branch-and-bound* algorithms and try to build an adaptive implementation for it. We aim at competitive performances compared to hand-tuned implementations in C, using OCamlMPI and MetaOCaml.