Adaptive Program Optimization
PhD Proposal

Cédric Bastoul (ICube)

1 Summary

Compute-intensive applications now range from image processing on smartphones to large simulations on supercomputers. A significant part of their development effort is devoted to making the computation done within a finite budget of time, space or energy. Given the complexity of modern architectures, writing such applications is typically a two-step workflow. First, developers design a sequential program for algorithmic tuning and debugging purpose. Second, experts optimize and parallelize the sequential program to scale to the actual problem size and to exploit the target architecture. While the first step remains close to the original problem, the second one corresponds to a development time overhead.

To minimize the development effort, automated optimization and parallelization approaches exist and provide good results at vectorizing and extracting thread-level parallelism for some classes of very large loops. However, those techniques have been historically designed for precise and long-lasting computations on supercomputers with well-known characteristics, and they are not well adapted to the new range of applications on mainstream parallel architectures, including short and approximate computations, which may be run on very different devices while being compiled only once. The goal of this PhD proposal is to target this issue by researching, designing and evaluating new compiler techniques for automatic optimization and parallelization with adaptive capabilities. Building on domain-specific knowledge about the application and on state-of-the-art program optimization techniques, this PhD will propose new ways to exploit the dynamic execution environment and computation properties to bridge the gap between ideal implementations and production optimized versions in an automated way.

2 Scientific Context

This PhD proposal falls within the research conducted in the ICPS team (Parallel and Scientific Computing) of the ICube laboratory at the university of Strasbourg, and the CAMUS group (Compilation for MultiCore Architectures) at Inria, the French Institute for Research in Computer Science. ICPS/CAMUS focuses at providing developers with theoretical and software tools to develop efficient applications for parallel architectures without sacrificing productivity. The originality of this project is to focus on the exploitation of static and dynamic information of both the application and the execution environment to adapt the optimization strategy in the most efficient way to generate highly efficient codes with a minimal effort from the programmers.

Parallel systems are now omnipresent, from supercomputers to mobile devices, and their effective use requires developers to design parallel programs or to rewrite legacy sequential applications. However, parallel programming is still complex and out of reach for non experts, at either designing, writing or debugging level. To address this issue, ICPS is developing source-to-source compiler techniques for automatic program optimization and parallelization. Using these technologies, developers can continue to write sequential programs while the mapping to

parallel architecture is computed automatically. Our group has an international visibility built by researching, developing and disseminating high-level compilation techniques now included in production compilers like GCC and LLVM, and in research compilers like PoCC [1] [5] or Pluto [2] [3]. They rely on the so-called "polyhedral model", a very effective research approach, suitable for some classes of loop- and array-centric applications, which demonstrated superior ability to automatically parallelize and optimize compute-intensive programs [1].

While polyhedral compilation frameworks can perform aggressive program transformations by rescheduling program instructions at the loop iteration level, recent studies show that state-of-the-art optimization strategies may be strongly inadequate in many cases [4]. To improve the situation, we propose to investigate static-dynamic adpative strategies by extracting static information from both the code and domain-specific knowledge provided by the user, and to exploit this information dynamically to adapt to the optimization strategy to the execution and application context. The execution context may include the target architecture and dynamic parameters do drive pertinent optimization selection. The application context may include manipulated data which, alog with domain-specific information, may allow to relax the rigid data dependence model of existing compiler techniques. For instance, for some classes of applications, including simulation or signal processing, optimization techniques include tuning the accuracy of the result in some regions when the full computation cannot be done for computation time, disk space, or energy dissipation reasons. For instance, Adaptive Mesh Refinement [2] (AMR) is a classical numerical analysis technique providing the ability to dynamically tune a computational grid to achieve precise computation only where it matters. New studies are now necessary to leverage domain-specific knowledge, numerical analysis methods and state-of-the-art optimizing compiler techniques to provide the next level of optimization to compilers.

3 Mission

The goal of this PhD is to study, design and evaluate ways for compute-intensive application programmers to input pertinent domain-specific knowledge to their codes and ways for compilers to use both the code and this information to generate adaptive optimized versions. First, the student will have to master the different technologies and tools for polyhedral compilation developed by the ICPS team. He/She will then study classical automatic optimization algorithms an numerical analysis techniques to compute approximated solutions from a review of the state of the art, with a special focus on AMR. He/She will design ways for programmers to statically specify regions of (various) interest in compute-intensive codes and code generation techniques that take this information into account to build a code that achieves only the necessary computation. These methods will be extended to support the dynamic case, e.g., in a way similar to AMR, but with automatically generated codes and relying on the power of the polyhedral model to deal with more general boxes than (hyper-)rectangles, with an optimization adapted to the execution context. The approach will be evaluated with an empirical approach on compute-intensive codes as well as preliminary studies with potential users (participatory design with programmers).

4 Research Team

The PhD student will join the Scientific and Parallel Computing research group (ICPS) at ICube Laboratory, linked to the research team on Compilation for Multicore Architectures (CAMUS)

[1] `http://pocc.sf.net`

[2] `http://pluto-compiler.sf.net`

at Inria, the French Institute for Research in Computer Science. This team aspires to contribute to state-of-the-art techniques and technologies in the field of high performance computing. The group's area of expertise lies in the parallelization and optimization of programs. A large class of programs is addressed by its research work, ranging from programs for multi-cores to message-passing parallel programs on grids or clouds. The team has 9 faculty members and a dozen of PhD students, postdocs or engineers. Its members are deeply involved in European and international collaborations on scientific projects both with academia and industry. Its most renowned technical contributions in the field of this PhD proposal include the PolyLib library, the code generator CLooG, the dynamic optimizer VMAD or the parallelizer of binary programs BinPar.

The PhD student will join the ICPS group and work with its researchers at Strasbourg's Innovation Center, France. He/She will also benefit from the scientific context of the ICPS group and the University of Strasbourg in general, with collaborations with applied mathematics groups through the Excellence Laboratory IRMIA.

5 Skills and Profile

This PhD proposal is open to any applicant with (or finishing) a Master Degree in Computer Science or Applied Mathematics with a strong motivation to work on performance and compiler techniques for automatic parallelization and optimization. No particular expertise on compilers, architecture or linear algebra is necessary, but it would be a plus. A reasonable experience on C programming and working on Linux/Unix environment is required.

6 Contact Information

Cédric Bastoul, Professor at Strasbourg University, ICube Laboratory

ICPS – Pôle API	Tel.:	+33(0) 3 68 85 45 52
300 boulevard Brant CS 10413	Mail:	`cedric.bastoul@unistra.fr`
F-67412 Illkirch	Web:	`icps.u-strasbg.fr/~bastoul`

References

[1] C. Bastoul. *Contributions to High-Level Program Optimization.* Habilitation Thesis. Paris-Sud University, France, Dec. 2012.

[2] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82(1):64–84, May 1989.

[3] U. Bondhugula, A. Hartono, J. Ramanujam, and P. Sadayappan. A practical automatic polyhedral parallelizer and locality optimizer. In *ACM SIGPLAN Intl. Conf. on Programming Language Design and Implementation (PLDI'08)*, Tucson, USA, June 2008.

[4] P. Clauss. Mind the gap! a study of some pitfalls preventing peak performance in polyhedral compilation using a polyhedral antidote. In *IMPACT'2015 5th International Workshop on Polyhedral Compilation Techniques*, Amsterdam, The Netherlands, Jan. 2015.

[5] L.-N. Pouchet, C. Bastoul, A. Cohen, and J. Cavazos. Iterative optimization in the polyhedral model: Part II, multidimensional time. In *ACM SIGPLAN Intl. Conf. on Programming Language Design and Implementation (PLDI'08)*, pages 90–100, Tucson, USA, June 2008.