Chapter 1. Introduction

João Bispo[1], Pedro Pinto[1], João M.P. Cardoso[1], Jorge G. Barbosa[1], Hamid Arabnejad[1], Davide Gadioli[2], Emanuele Vitali[2], Gianluca Palermo[2], Cristina Silvano[2], Stefano Cherubin[2], Giovanni Agosta[2], Loïc Besnard[3], Antonio Libri[4], Daniele Cesarini[5], Andrea Bartolini[5], and Luca Benini[4],[5]

[1] *Faculty of Engineering, University of Porto, Porto, Portugal*
{jbispo, p.pinto, jmpc, jbarbosa, hamid.arabnejad}@fe.up.pt

[2] *Politecnico di Milano, Milano, Italy*
{davide.gadioli, emanuele.vitali, gianluca.palermo, cristina.silvano, stefano.cherubin, giovanni.agosta}@polimi.it

[3] *Univ. Rennes, CNRS, IRISA, Rennes,France*
loic.besnard@irisa.fr

[4] *ETH Zurich, Zurich, Switzerland*
{antonio.libri, lbenini}@iis.ee.ethz.ch

[5] *Università di Bologna, Bologna, Italy*
{daniele.cesarini, a.bartolini}@unibo.it

1 Overview

The ANTAREX[1] research project was funded by the H2020 Future and Emerging Technologies programme on High Performance Computing (HPC). The project involved CINECA, the Italian Tier-0 Supercomputing Centre, and IT4Innovations, the Czech Tier-1 Supercomputing Center. The Consortium also included three top-ranked academic partners (ETH Zurich, University of Porto and INRIA), one of the Italian leading biopharmaceutical companies (Dompé) and the top European navigation software company (Sygic). The project started on September the 1st, 2015 and was concluded on November 30, 2018.

The main goal of the ANTAREX project was to provide a breakthrough approach to map, runtime manage and autotune applications for green and heterogeneous HPC systems up to the Exascale level. The key ANTAREX innovations were designed and engineered since the beginning to be scaled-up to the Exascale level. One key innovation of the proposed approach consists of introducing a separation of concerns (where self-adaptivity and energy efficient strategies are specified aside to application functionalities) promoted by the definition of a Domain Specific Language (DSL), inspired by aspect-oriented programming (AOP) concepts for heterogeneous systems. The DSL was introduced for expressing the adaptivity/energy/performance strategies and to enforce at runtime application autotuning and resource and power management. The goal was to support the parallelism, scalability and adaptability of a dynamic workload by exploiting the full system capabilities (including energy management) for emerging large-scale

[1] http://http://antarex-project.eu/

and extreme-scale systems, while reducing the Total Cost of Ownership (TCO) for companies and public organizations.

The ANTAREX project was driven by two use cases chosen to address the self-adaptivity and scalability characteristics of two highly relevant HPC application scenarios:

- a biopharmaceutical HPC application for accelerating drug discovery deployed on the 1.2 PetaFlops heterogeneous NeXtScale Intel-based IBM system at CINECA;

- a self-adaptive navigation system to be used in smart cities deployed on the server-side on an heterogeneous Intel-based 1.46 PetaFlops class system provided by IT4Innovations.

These use cases have been selected due to their significance in emerging application trends and thus by their direct economic exploitability and relevant social impact.

More information about the ANTAREX project can be found in [1][2][3][4][5][6].

2 Book Structure

This online book describes the main tools and libraries that are part of the ANTAREX tool flow. The book consists of the following chapters:

1. ”DSL and Source to Source Compilation: the Clava+LARA approach,” by João Bispo, Pedro Pinto, and João M.P. Cardoso (Faculdade de Engenharia da Universidade do Porto, Portugal). CLAVA is a source to source (C++ to C++) compiler entirely developed during the ANTAREX project. It includes an aspect-oriented programming approach, implemented by an internal weaver and the technology provided by the LARA DSL, in order to describe source-to-source strategies, such as code transformations and instrumentation. In most cases, the strategies are applied offline and/or translated to code in the target programming language and embedded in the application code. The version of CLAVA presented in this chapter is able to compile a wide range of applications, and several kinds of strategies have been written in the ANTAREX DSL, including auto-parallelization, design space exploration, source-code generation and automatic integration of other ANTAREX libraries and tools. In addition to the capability for code transformations, code instrumentation, and code injection for integration of runtime autotuning and adaptivity schemes, CLAVA also includes data dependence analysis stages that are used by the autoparallelizer via OpenMP directives. This chapter presents the CLAVA compiler how the interaction with LARA works and includes a number of examples (with all software code available) showing some of the advantages and usefulness of the approach.

2. ”Runtime Autotuning: the mArgot Approach,” by D. Gadioli, E. Vitali, and G. Palermo, C. Silvano (Politecnico di Milano, Italy). In the autonomic computing context, applications are perceived as autonomous agents that are able to adapt at runtime, according to the evolution of the system. The proposed framework aims at enhancing a target application with an adaptation layer, to provide self-optimization capabilities. In particular, *mARGOt* is a C++ library requiring a limited intrusiveness in the target application to identify the region of interest and the software knobs to be manipulated. The library is instantiated and customized according to extra-functional requirements of the application specified in a configuration file. *mARGOt* exploits design-time knowledge and multi-objective requirements expressed by the user, to drive the autotuning process at the runtime.

3. "The OpenMP-based Auto Parallelization AutoPar-Clava Approach," by Hamid Arabnejad, João Bispo, Jorge G. Barbosa, and João M.P. Cardoso (Faculdade de Engenharia da Universidade do Porto, Portugal). Modern processors are composed by several processing elements, known as multicore architectures, which brings to the common user the possibility to use parallel computing techniques in order to fully exploit the computational power available in modern machines. Directive-driven programming models, such as OpenMP, are a common solution for exploring the potential of multicore architectures, and enable users (i.e., developers) to accelerate software applications by adding annotations on for-type loops and code regions to change the execution pattern of their code. However, to transform a sequential code into a parallel version requires advanced programming knowledge and it is also a time consuming task to achieve performance. To overcome this burden, we present in this chapter a compilation tool, `AutoPar-Clava`, that is able to automatically detect parallelizable loops in a C application without any user intervention or profiling information; that classifies variables used inside the target loop based on their access pattern; and that generates a C OpenMP parallel code from the input sequential version. The tool is also able to implement automatically *reduction* operations either for scalar and array data. The implementation details of `AutoPar-Clava`, its usage and application examples are reported in this chapter. The tool showed to be of practical use and achieved good performance for several benchmarks, such as the NAS and Polyhedral Benchmark suites, targeting a 16-cores x86-based computing platform.

4. "Split Compilation: the libVersioningCompiler Approach," by Stefano Cherubin and Giovanni Agosta (Politecnico di Milano, Italy). libVersioningCompiler is a library that allows partial dynamic recompilation within an application. It is designed to ease the burden of performing continuous program optimization within the context of High Performance Computing applications. In the context of the ANTAREX tool flow, libVersioningCompiler can be employed through the ANTAREX DSL, so that its operation can be combined with that of other components of the toolchain, to achieve fine tuning of compilation options and code version management.

5. "LARA Strategies for Data Type Conversions," by Loïc Besnard (IRISA-CNRS, France). To easily explore different representations of C numerical types (double, float, fixed point, half precision...), the user should develop its applications by the introduction of data types abstraction. But when it is not the case, it becomes fastidious to do it after. In this chapter, we propose some LARA aspects developed in ANTAREX projects that automatically abstract types of applications.

6. "LARA Strategies for Loop Splitting," by Loïc Besnard (IRISA-CNRS, France). This chapter presents a technique, called loop splitting, that takes advantage of long running loops to explore different compiling options to optimize the user applications. This technique may be also used to explore different implementation of algorithms. LARA aspects have been developed to apply the technique in a very simple way.

7. "Memoization Approach," by Loïc Besnard (IRISA-CNRS, France). This chapter presents a technique, called memoization, that catches results of pure functions and retrieves them instead of recomputing a result to optimize applications for energy efficiency. The definition of LARA aspects allows to the user to apply the memoization in a very easy way to C and C++ applications.

8. "ExaMon: Exascale Holistic Monitoring," by Francesco Beneventi, Antonio Libri, Andrea

Bartolini and Luca Benini (ETH Zurich, Switzerland). EXAMON, which stands for EX-Ascale MONitoring, aims to develop a portable and extensible monitoring framework at application-level that gives the possibility to the application to inspect extra-functional properties (such as, energy) instead of only raw architecture-specific metrics (such as low level values coming from HW counters). The monitoring framework is also developed to support the monitoring of the runtime behaviour of the system. The goal is to continuously and dynamically collect data from the system to make them available to applications and management layers. Besides traditional feedback on HW performance and throughput, novel kinds of scalable monitors, specific for HPC systems, is developed to provide feedback about performance, energy efficiency and thermal efficiency. This will be achieved by designing monitor blocks, as well as the data collectors, which observe application execution and phases, and which are able to detect patterns / signatures of the instructions. Furthermore, access patterns to the instruction and data cache / memory are observed to detect possible optimization of the memory allocation. Proper interfacing SW driver layers are developed for the target platforms to communicate data and events to applications as well as APIs to propagate application events to the collectors. The solution leverages big-data infrastructure to support the exascale monitored data flow.

9. ”Energy-Efficiency Run-time: the COUNTDOWN Approach,” by Daniele Cesarini, Andrea Bartolini and Luca Benini (ETH Zurich, Switzerland). Energy and power consumption are prominent issues in today’s supercomputers and are foreseen as a limiting factor of future installations. In scientific computing, a significant amount of power is spent in the communication and synchronization-related idle times among distributed processes participating to the same application. However, due to the time scale at which communication happens, taking advantage of low-power states to reduce power in idle times in the computing resources, may introduce significant overheads. In this paper we present COUNTDOWN, a methodology and a tool for identifying and automatically reducing the power consumption of the computing elements during communication and synchronization primitives filtering out phases which would detriment the time to solution of the application. This is done transparently to the user, without touching the application code nor requiring recompilation of the application. We tested our methodology in a production Tier-0 system, a production application - Quantum ESPRESSO (QE) - with production datasets which can scale up to 3.5K cores. Experimental results shows that our methodology saves 22.36% of energy consumption with a performance penalty of 2.88% in real production MPI-based application.

References

[1] Cristina Silvano, Giovanni Agosta, Stefano Cherubin, Davide Gadioli, Gianluca Palermo, Andrea Bartolini, Luca Benini, Jan Martinovič, Martin Palkovič, Kateřina Slaninová, et al. The antarex approach to autotuning and adaptivity for energy efficient hpc systems. In *Proceedings of the ACM International Conference on Computing Frontiers*, CF ’16, pages 288–293, New York, NY, USA, 2016. ACM.

[2] Cristina Silvano, Giovanni Agosta, Andrea Bartolini, Andrea R Beccari, Luca Benini, João Bispo, Radim Cmar, João MP Cardoso, Carlo Cavazzoni, Jan Martinovič, et al. Autotuning and adaptivity approach for energy efficient exascale hpc systems: the antarex approach. In *Proceedings of the 2016 Conference on Design, Automation & Test in Europe*, DATE ’16, pages 708–713, 2016.

[3] Cristina Silvano, Giovanni Agosta, Jorge Barbosa, Andrea Bartolini, Andrea R Beccari, Luca Benini, João Bispo, João MP Cardoso, Carlo Cavazzoni, Stefano Cherubin, et al. The antarex tool flow for monitoring and autotuning energy efficient hpc systems. In *Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS), 2017 International Conference on*, pages 308–316. IEEE, 2017.

[4] Cristina Silvano, Gianluca Palermo, Giovanni Agosta, Amir Ashouri, Davide Gadioli, Stefano Cherubin, Emanuele Vitali, Luca Benini, Andrea Bartolini, Daniele Cesarini, et al. Autotuning and adaptivity in energy efficient hpc systems: The antarex toolbox. In *International Conference on Computing Frontiers*, pages 270–275, 2018.

[5] Davide Gadioli, Ricardo Nobre, Pedro Pinto, Emanuele Vitali, Amir H Ashouri, Gianluca Palermo, Joao Cardoso, and Cristina Silvano. Socrates—a seamless online compiler and system runtime autotuning framework for energy-aware applications. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2018*, pages 1143–1146. IEEE, 2018.

[6] Cristina Silvano, Giovanni Agosta, Andrea Bartolini, Andrea R Beccari, Luca Benini, Loïc Besnard, João Bispo, Radim Cmar, João MP Cardoso, Carlo Cavazzoni, et al. Antarex: A dsl-based approach to adaptively optimizing and enforcing extra-functional properties in high performance computing. In *2018 21st Euromicro Conference on Digital System Design (DSD)*, pages 600–607. IEEE, 2018.