Parallel Programming Tools for Multi-core Architectures

Bernd MOHR [a], Bettina KRAMMER [b] and Hartmut MIX [c]
[a] *Forschungszentrum Jülich, Jülich Supercomputing Centre (JSC), Germany*
[b] *University of Versailles Saint-Quentin-en-Yvelines (UVSQ), France*
[c] *Tech. Universtät Dresden, Center for Information Services and High Performance Computing (ZIH), Germany*

1. Introduction

The improvement of the processor performance by increasing the clock rate has reached its technological limits. Increasing the number of processor cores rather than clock rates can give better performance and reduce problems such as energy consumption, heat dissipation and design complexity. We are witnessing the emergence of multi-core processors in all markets - from laptops and game consoles to servers and supercomputers. At the same time, architectures are becoming heterogeneous and memory hierarchies and interconnection networks are increasingly complex.

Multi-core architectures present new opportunities as well as challenges to software that will run on computer systems built up on such multi-core chips. To keep pace with the evolution from dual-, quad-, to large many-core systems, the keys to unleashing significant performance enhancements for such multi-core systems are well-defined parallel programming models and robust, easy-to-use tools supporting the parallelization process. Compared to sequential applications, software engineers require a more powerful, well matched repertoire of development tools and methods for costeffectively achieving highly optimized, reliable, fault tolerant and robust parallel applications.

2. Papers and Presentations

The purpose of this mini-symposium was to bring together researchers and practitioners with diverse backgrounds in order to advance the state of the art in software engineering for multi-core parallel applications. In this spirit, many of the papers do not only present parallel programming tools and methods, but demonstrate their usefulness on real-world industrial applications.

The paper “Parallel Programming for Multi-core Architectures” by Jean-Marc Morel gives an overview on the European ParMA project (2007-2010). This project targets the development and extension of programming methods, tools, libraries and Linux operating system, to enable parallel applications to exploit fully the power of multi-core architectures, both for High-Performance and Embedded Computing.

Andres Charif-Rubial et al. present a "Methodology for Application Performance Tuning". This methodolgy combines static assembly analysis using the MAQAO tool and dynamic hardware performance monitoring and memory tracing. Applying it to an iterative linear solver developed by Dassault Aviation, a speed-up of almost up to 2.5 could be achieved.

Another success story of applying different tools to an industrial use-case is presented by Benedetto Risio et al. in "How to Accelerate an Application: a Practical Case Study in Combustion Modelling". The tool-chain comprises both low-level (MAQAO) and high-level (VAMPIR, SCALASCA) analysis of the application: optimising memory access, cache utilisation and MPI synchronisation leads to a significant performance improvement.

Alejandro Duran et al. gave a presentation on "Supporting OpenMP in a heterogeneous world". Efficiently programming heterogeneous architectures with accelerator devices (e.g. graphics processors, vector units) is currently a complex task. They present a series of extensions to OpenMP, inspired in the work of the StarSs programming model, that allow developers to easily write portable codes for a number of different platforms, relieving them from developing the specific code to off-load tasks to the accelerators and the synchronization of tasks.

Daniel Millot et al. present the STEP tool in "From OpenMP to MPI: first experiments of the STEP source-to-source transformation". STEP can transform OpenMP code into MPI code, thus allowing easy transition from shared to distributed memory systems. Some benchmark results are presented.

The paper "Using Multi-Core Architectures to Execute High Performance-Oriented Real-Time Applications" by Christophe Aussaguès et al. describes the OASIS design, compilation and execution framework for embedded real-time systems that execute High-Performance oriented applications on multi-core architectures. OASIS is demonstrated on the example of a 2D-tracking algorithm from Dassault Aviation.

Allen D. Malony et al. report early experiences of "Performance Tool Integration in Programming Environments for GPU Acceleration: Experiences with TAU and HMPP". They describe the design approach of integrating the TAU parallel performance system, the TAUcuda accelerator performance tool and the HMPP workbench for programming GPU accelerators using CUDA. Two case studies are presented.

Tobias Hilbrich et al. present UniMCI, "An Interface for Integrated MPI Correctness Checking". This interface provides functionality for coupling performance analysis frameworks and correctness checking tools in a generic way. First results of a prototype implementation using the Marmot correctness checker and the VampirTrace performance analysis tool are shown.

Last not least, Thomas William et al. describe an approach of "Enhanced Performance Analysis of Multi-core Applications with an Integrated Tool-chain". In this case, the VAMPIR and SCALASCA performance analysis tools are connected through a common bus system, which allows users to identify and locate performance problems quickly with SCALASCA and to use the powerful VAMPIR visualistaion tool to dive into the execution history. The usefulness and effectiveness of this integrated tool-chain is demonstrated with the INDEED metal forming simulation software.