Prototype demonstration of performance analysis tools on the WP7 prototype
Version 1.0

Document Information

Contract Number	288777
Project Website	www.montblanc-project.eu
Contractual Deadline	M45
Dissemination Level	PU
Nature	P
Author	Marc Schlütter (JUELICH)
Contributors	Harald Servat (BSC), Judit Gimenez Lucas (BSC), Markus Geimer (JUELICH)
Reviewer	Chris Adeniyi-Jones (ARM)
Keywords	performance analysis tools, Scalasca, Score-P, Extrae, Paraver, Dimemas

Notices:
The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement n° 288777.

Change Log

Version	Description of Change
V0.1	Basic Template with contributions from JSC
V0.2	Combined input from JSC and BSC
V0.3	Language check and minor changes & changes from the internal review
V1.0	Version ready to send to the EC

Table of Contents

Executive Summary

In this deliverable, we present the current status of the prototype versions of the performance analysis tools considered in the Mont-Blanc project. This includes the community instrumentation and measurement system Score-P, the performance analysis toolset Scalasca with its result browser CUBE, developed by Jülich Supercomputing Centre, and the Barcelona performance tool-suite, containing the instrumentation library Extrae, the analysis tool Paraver and the simulation tool Dimemas. For all of these tools, we describe the current status of the porting to the Mont-Blanc platform, in particular the testing on the WP7 prototype, as well as the implemented extensions for supporting the OmpSs programming model.

1 Introduction

The main objective of the Mont-Blanc project is to build the first high-performance computing system based on low-power processors using the ARM architecture. In this context, it is essential that application developers are provided with tools supporting their efforts in porting their codes to the ARM platform and optimizing their performance on this architecture.
The initial porting results for the ARM architecture have been presented in deliverable 5.7. This document summarizes the work done on extending the performance tools of the Mont-Blanc partners to support the final WP7 Mont-Blanc prototype. In the context of the Mont-Blanc project, Score-P and Scalasca have been ported to the ARM platform, with different intermediate systems like the Tibidabo prototype, a set of Arndale and Odroid development boards, and now the final WP7 prototype as test platforms. Also, Score-P has been extended with prototypical support of the OmpSs programming model. The instrumentation library Extrae has been ported to ARM as well and allows gathering measurement results for use with Paraver and Dimemas.

2 Score-P, Scalasca and CUBE

Scalasca[1] is a free software tool, developed at the Jülich Supercomputing Centre, which supports the performance optimization of parallel programs by measuring and analyzing their runtime behavior. In the Scalasca v2 release series, the previous custom instrumentation and measurement system has been replaced by the community measurement system Score-P[2], allowing measurement of applications implemented with MPI, OpenMP and CUDA. As part of the tool suite, the CUBE result browser is responsible for the visualization of performance profiles and trace analysis results.

2.1 Score-P

The latest public release of Score-P, version 1.4, has been successfully tested on the WP7 prototype as well as on an ARM64 development board. OmpSs support, currently still maintained as a project internal, separate development branch, has been tested on the Mont-Blanc cluster as well. Due to the per node restrictions (memory availability, Qt framework), only the libraries for reading and writing CUBE files have been installed and are used on the cluster itself. Visualization of results with the CUBE viewer should be done on another system. In the context of tasks, support for untied tasks in profiling with Score-P has been added, allowing the

migration of tasks between threads during their lifetime. For GCC compilers 4.5 to 4.9, the latter being the default compiler on the prototype, Score-P now offers an alternative function instrumentation method using the GCC compiler plugin interface. This potentially improves the measurement performance and allows compile-time filtering. Also, initial Pthreads and improved CUDA support have been added since the last deliverable. As part of the Mont-Blanc-2 project, currently running in parallel, support for OpenCL has been added as well.

2.2 Scalasca

Scalasca 2.2 has been released and tested on the WP7 prototype. In the Scalasca trace analyzer, basic infrastructure support for tasks has been added. This covers the ability to read traces of application runs which use tasks. Figure 1 shows an example trace analysis result of a hybrid MPI+OmpSs test case, a taskified n-body simulation. The analysis itself does not yet detect tasking-specific patterns, however, the MPI and common thread level analysis is now available for hybrid applications based on tasks. All tasks in this context are described by a generic task model and any task-based programming model that can be mapped to this model can be used with Scalasca, e.g., OpenMP or OmpSs. The model representation in CUBE, while not yet in a final state, currently separates the asynchronous execution instances of tasks into a separate root node, called *task root*. In the call-tree panel, regions display the submission of tasks and artificial stub nodes representing the execution points of task instances, which can be used to link the main call-tree position to a task execution tree below the task root. Since the number of supported paradigms still increases, the model will be revised to cover more task and also accelerator paradigms in a unified style, as part of Mont-Blanc-2.

While the task support of Score-P and OTF2 also covers untied tasks, Scalasca currently requires all tasks to be tied. Another restriction of Scalasca is that only MPI thread modes up to MPI_THREAD_FUNNELED are supported. Since MPI only operates on ranks (i.e., processes), detailed information about communicating threads cannot be easily collected via the standard PMPI interface. However, the Scalasca trace analysis depends on this information for correct message matching. Since tasks are scheduled to run on arbitrary threads by the OmpSs runtime, it is currently not possible to analyze applications using MPI communication inside tasks.

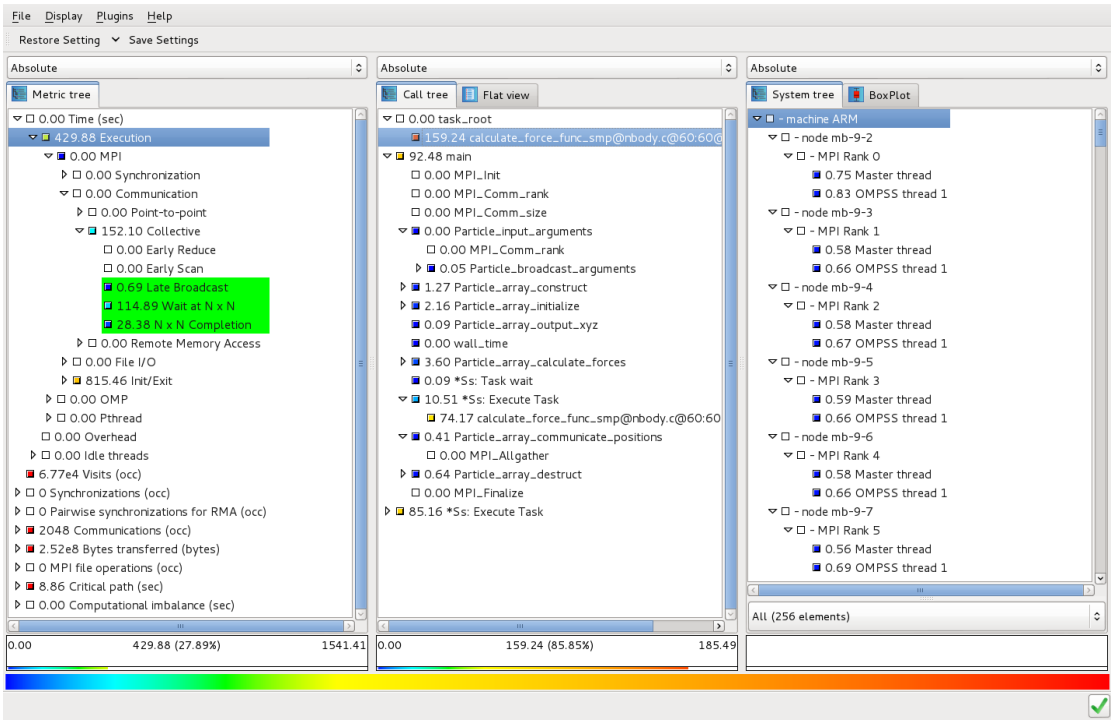

Figure 1: Hybrid MPI/OmpSs trace analysis example test case illustrating task model representation, OmpSs specific management regions in the main call tree, and the MPI patterns for wait states found in the code.

3 Extrae, Paraver and Dimemas

The performance tool-suite developed at Barcelona Supercomputing Center includes Extrae, Paraver and Dimemas. Extrae is the instrumentation package that generates time-stamped trace-files for multiple parallel programming paradigms (including MPI, OpenMP, and Pthreads, among many others). Paraver is a very powerful performance visualization and analysis tool based on traces that can be used to analyze any information that is expressed in its input trace format. Finally, Dimemas is a simulation tool for the parametric analysis of the behavior of message-passing applications on a configurable parallel platform.

3.1 Extrae

Extrae has been successfully ported to the Mont-Blanc prototype but some issues appeared during the initial stages of the porting process. For instance, the MPI installations on the prototype implement the MPI v3 API and Extrae did not support them in the earlier stages. This support was introduced with Extrae 3. Another issue that appeared during the first stages of the porting was spurious errors related with the libunwind library [5]. Extrae uses libunwind to capture the call-stack at several instrumentation points as well as at sample points. The library has recently received several modifications for the ARM processors to improve its functionality and stability. As of writing this document, we are using a copy of the git repository.

The ARM version of the instrumentation package supports most of the standard Extrae functionality. Besides supporting manual instrumentation, the package automatically gathers

performance information for many parallel programming models, such as MPI (in the several MPI flavors installed in the system), OpenMP, OmpSs, Pthreads, CUDA, and OpenCL. However, there is no support for dynamic instrumentation using DynInst because this library has not been ported to ARM. With respect to the performance data gathered at the instrumentation points, Extrae attributes the timestamp to the events using the POSIX clock which provides nanosecond resolution on this architecture. Additionally, the instrumentation library captures the performance counters of the ARM Cortex-A15 processors using the PAPI library without any kernel changes as were necessary for the earlier Tibidabo system. Extrae has been used to instrument multiple applications on the ARM prototype under different conditions. Among the applications instrumented, we can name Lulesh [3] (as depicted in Figure 2 and Figure 3) and CoMD [4] (depicted in Figure 4 and Figure 5). The Lulesh benchmark used a hybrid parallel binary using MPI and OpenMP and the execution ran on 27 nodes using the two cores of each node. With respect to CoMD, it executed using 540 MPI processes using only one of the available cores in each node.

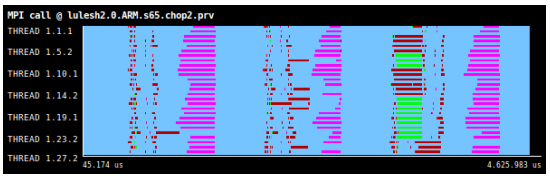

Figure 2: Paraver depicting the MPI calls from Lulesh.

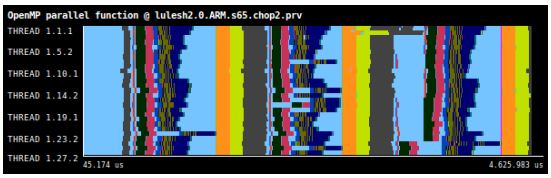

Figure 3: Paraver depicting the OpenMP outlined routines from Lulesh.

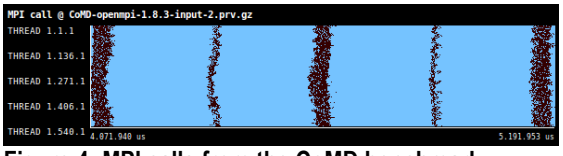

Figure 4: MPI calls from the CoMD benchmark.

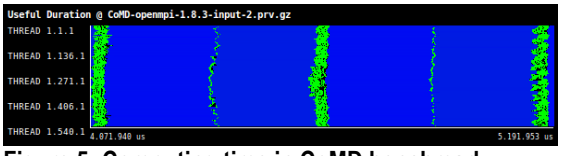

Figure 5: Computing time in CoMD benchmark.

3.2 Paraver

Paraver is a very flexible data browser that is part of the CEPBA-Tools toolkit developed by BSC and is available for download under an LGPL open-source license. The tool allows a detailed and powerful exploration of trace data. Programmable through configuration files, Paraver can visualize performance data via time-line displays (showing metrics per process or thread over time) or histogram displays (showing statistical data), independently from the programming model used. While Extrae is required to be executed on the target architecture, and therefore had to be ported, Paraver is typically executed on a desktop machine. Hence, we did not port this part of the tool-suite.

3.3 Dimemas

Dimemas is a performance analysis tool for message-passing programs. The Dimemas simulator reconstructs the time behavior of a parallel application trace-file based on a machine modeled by key factors influencing the performance. With a simple model, Dimemas allows us to simulate complete parametric studies in a very short time frame. Dimemas generates as part

of its output a Paraver trace-file, enabling the user to conveniently examine the simulator run in Paraver. As with Paraver, Dimemas is typically executed in a desktop machine, and therefore we did not port the simulator to the architecture.

Acronyms and Abbreviations

- MPI : Message Passing Interface
- OpenMP : Open Multi-Processing
- OmpSs : OpenMP Super-Scalar
- CUDA: Compute Unified Device Architecture
- OpenCL: Open Computing Language
- DynInst: Dynamic Instrumentation
- PAPI: Performance Application Programming Interface
- POSIX: Portable Operating System Interface uniX
- OTF2: Open Trace Format 2

References

[1] Markus Geimer, Felix Wolf, Brian J. N. Wylie, Erika Ábrahám, Daniel Becker, Bernd Mohr: The Scalasca performance toolset architecture. Concurrency and Computation: Practice and Experience, 22(6):702–719, April 2010.
[2] Andreas Knüpfer, Christian Rössel, Dieter an Mey, Scott Biersdorff, Kai Diethelm, Dominic Eschweiler, Markus Geimer, Michael Gerndt, Daniel Lorenz, Allen D. Malony, Wolfgang E. Nagel, Yury Oleynik, Peter Philippen, Pavel Saviankou, Dirk Schmidl, Sameer S. Shende, Ronny Tschüter, Michael Wagner, Bert Wesarg, Felix Wolf: Score-P – A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir. In Proc. of 5th Parallel Tools Workshop, 2011, Dresden, Germany, pages 79-91, Springer Berlin Heidelberg, September 2012.
[3] https://codesign.llnl.gov/lulesh.php
[4] http://www.exmatex.org/comd.html
[5] https://savannah.nongnu.org/projects/libunwind