scalasca

Scalasca 2.0 | User Guide

Scalable Automatic Performance Analysis

August 2013
The Scalasca Development Team
scalasca@fz-juelich.de

JÜLICH
FORSCHUNGSZENTRUM

German Research School
for Simulation Sciences

The entire code of Scalasca v2 is licensed under the BSD-style license agreement given below, except for the third-party code distributed in the ’vendor/’ subdirectory. See the corresponding COPYING files in ’vendor/∗’ of the distribution tarball for details.

Scalasca v2 License Agreement

Copyright © 1998–2013	Forschungszentrum Jülich GmbH, Germany
Copyright © 2009–2013	German Research School for Simulation Sciences GmbH, Jülich/Aachen, Germany
Copyright © 2003–2008	University of Tennessee, Knoxville, USA
Copyright © 2006	Technische Universität Dresden, Germany

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the names of
  - the Forschungszentrum Jülich GmbH,
  - the German Research School for Simulation Sciences GmbH,
  - the University of Tennessee,
  - the Technische Universität Dresden,

  nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Attention

The Scalasca User Guide is currently being rewritten. Meanwhile, the best sources for usage information are the slide sets from the last VI-HPS Tuning Workshop.

Contents

1 Introduction

Supercomputing is a key technology of modern science and engineering, indispensable to solve critical problems of high complexity. However, since the number of cores on modern supercomputers is increasing from generation to generation, HPC applications are required to harness much higher degrees of parallelism to satisfy their growing demand for computing power. Therefore – as a prerequisite for the productive use of today's large-scale computing systems – the HPC community needs powerful and robust performance analysis tools that make the optimization of parallel applications both more effective and more efficient.

Jointly developed at the Jülich Supercomputing Centre and the German Research School for Simulation Sciences (Aachen), Scalasca is a performance analysis toolset that has been specifically designed for use on large-scale systems such as IBM Blue Gene and Cray XT, but also suitable for smaller HPC platforms. Scalasca supports an incremental performance analysis process for applications using MPI[3] and/or OpenMP[8] which integrates runtime summaries with in-depth studies of concurrent behavior via event tracing, adopting a strategy of successively refined measurement configurations[5]. A distinctive feature of Scalasca is the ability to identify wait states that occur, for example, as a result of unevenly distributed workloads. Especially when trying to scale communication intensive applications to large processor counts, such wait states can present severe challenges to achieving good performance. Compared to its predecessor KOJAK[10], Scalasca can detect such wait states even in very large configurations of processes using a novel parallel trace-analysis scheme[4].

In contrast to previous versions of Scalasca which used a custom measurement system and trace data format, the Scalasca 2.x release series is based on the community instrumentation and measurement infrastructure Score-P [7], which is jointly developed by a consortium of partners from Germany and the US (see[1] for details). This significantly improves interoperability with other performance analysis tool suites such as Vampir[6] and TAU[9] due to the usage of the two common data formats CUBE4 for profiles and the Open Trace Format 2 (OTF2)[2] for event trace data. Nevertheless, the Scalasca 2.x series provides a certain level of backward compatibility, that is, the trace analysis component of Scalasca v2 is still able to process trace files generated by Scalasca 1.x.

Bibliography

[1] Score-P website. http://www.score-p.org. 1

[2] D. Eschweiler, M. Wagner, M. Geimer, A. Knüpfer, W. E. Nagel, and F. Wolf. Open Trace Format 2 - The next generation of scalable trace formats and support libraries. In *Applications, Tools and Techniques on the Road to Exascale Computing (Proc. of Intl. Conference on Parallel Computing, ParCo, Ghent, Belgium, August/September 2011)*, volume 22 of *Advances in Parallel Computing*, pages 481–490, Amsterdam, May 2012. IOS Press. 1

[3] Message Passing Interface Forum. MPI: A message-passing interface standard. http://www.mpi-forum.org, September 2012. Version 3.0. 1

[4] M. Geimer, F. Wolf, B. J. N. Wylie, and B. Mohr. Scalable parallel trace-based performance analysis. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface (Proc. of 13th European PVM/MPI Users' Group Meeting, EuroPVM/MPI, Bonn, Germany)*, volume 4192 of *Lecture Notes in Computer Science*, pages 303–312, Berlin/Heidelberg, September 2006. Springer. 1

[5] M. Geimer, F. Wolf, B.J.N. Wylie, E. Ábrahám, D. Becker, and B. Mohr. The Scalasca performance toolset architecture. *Concurrency and Computation: Practice and Experience*, 22(6):702–719, April 2010. 1

[6] A. Knüpfer, H. Brunst, J. Doleschal, M. Jurenz, M. Lieber, H. Mickler, M. S. Müller, and W. E. Nagel. The Vampir performance analysis toolset. In *Tools for High Performance Computing (Proc. of 2nd Parallel Tools Workshop, July 2008, Stuttgart, Germany)*, pages 139–155, Berlin/Heidelberg, July 2008. Springer. 1

[7] A. Knüpfer, C. Rössel, D. an Mey, S. Biersdorff, K. Diethelm, D. Eschweiler, M. Geimer, M. Gerndt, D. Lorenz, A. D. Malony, W. E. Nagel, Y. Oleynik, P. Philippen, P. Saviankou, D. Schmidl, S. S. Shende, R. Tschüter, M. Wagner, B. Wesarg, and F. Wolf. Score-P – A joint performance measurement run-time infrastructure for Periscope, Scalasca, TAU, and Vampir. In *Tools for High Performance Computing 2011 (Proc. of 5th Parallel Tools Workshop, September 2011, Dresden, Germany)*, pages 79–91, Berlin/Heidelberg, September 2012. Springer. 1

[8] OpenMP Architecture Review Board. OpenMP API specification for parallel programming. http://www.openmp.org, July 2011. Version 3.1. 1

[9] S. Shende and A. D. Malony. The TAU parallel performance system. *International Journal of High Performance Computing Applications*, 20(2):287–311, May 2006. 1

[10] F. Wolf and B. Mohr. Automatic performance analysis of hybrid MPI/OpenMP applications. *Journal of Systems Architecture*, 49(10-11):421–439, November 2003. 1

scalasca

JÜLICH
FORSCHUNGSZENTRUM

German Research School
for Simulation Sciences

www.scalasca.org