ISC'09 Poster Abstract *: I/O Performance Analysis for the Petascale Simulation Code FLASH

Heike Jagode, Shirley Moore, Dan Terpstra, Jack Dongarra
The University of Tennessee, USA
[jagode | shirley | terpstra | dongarra]@eecs.utk.edu

Andreas Knüpfer, Matthias Jurenz,
Matthias S. Müller, Wolfgang E. Nagel
Technische Universität Dresden, Germany
[andreas.knuepfer | matthias.jurenz |
matthias.mueller | wolfgang.nagel]@tu-dresden.de

September 21, 2009

1 Introduction and Background

Performance analysis of applications on modern high-end Petascale systems is increasingly challenging due of the rising complexity and quantity of the computing units. Many of the scientific applications running on contemporary high-end computing platforms are very data-intensive. This poster presents a performance analysis study with the Vampir performance analysis suite that examines the application behavior as well as the fundamental system properties.

The study is done on the Cray XT4 Jaguar system at Oak Ridge National Laboratory (ORNL), consisting of more than 30,000 CPU cores. We analyze the parallel simulation code called FLASH that is designed to scale towards tens of thousands of CPU cores. This situation - analyzing a highly complex parallel code on a high-end Petascale system - makes it very demanding to apply existing performance analysis tools. Yet, the trace-based performance evaluation of the FLASH software with Vampir exposes I/O performance issues that become relevant for very high CPU counts. For this class of problems, referring to massive I/O within the checkpointing mechanism, solutions are presented and verified with examples from the event-trace visualization. The suggested improvements require only local modifications, not affecting the general structure of the code.

*ISC'09 Research Poster Award 2009

1.1 Computer Systems

The Jaguar system at ORNL is based on Cray XT4 hardware. It utilizes 7,832 quad-core AMD Opteron processors with a clock frequency of 2.1 GHz and 8 GBytes of main memory (maintaining 2 GBytes for each CPU core). Jaguar offers a theoretical peak performance of 260.2 Tflops/s and a sustained performance of 205 Tflops/s on Linpack [1]. The nodes are arranged in a three-dimensional torus topology of size $21 \times 16 \times 24$ with a full SeaStar2 router through HyperTransport.

Jaguar has three Lustre file systems of which two have 72 Object Storage Targets (OST) and one has 144 OSTs [2]. All three of these file systems share 72 physical Object Storage Server (OSS). The theoretical peak performance of I/O bandwidth is $\sim$50GB/s across all OSSes.

1.2 Performance Analysis Suite: Vampir

The Vampir (Visualization and Analysis of MPI Resources) suite is a collection of tools for performance analysis that enables application developers to visualize program behavior at any level of detail. It consists of the VampirTrace part for instrumentation, monitoring and recording as well as the VampirServer part for visualization and analysis [3, 5].

To analyze details of the FLASH application, event-based program traces have been recorded using VampirTrace [3]. The generated Open Trace Format (OTF) [4] trace files have then been analyzed using the scalable visualization tool VampirServer [5].

1.3 Application Case Study: FLASH

FLASH is a modular, Adaptive Mesh Renement (AMR), parallel simulation code that computes general compressible flow problems for a large range of application scenarios [7]. The application test case is drawn from the workload configurations that are expected to scale to large number of cores and that are representative for Petascale problem configurations. The FLASH code is very data-intensive and large, containing complex performance characteristics and numerous production configurations that cannot be captured or characterized adequately in the current study. The intent is rather to provide a qualitative view of system performance using the `WD_Def` test case to highlight how the Vampir performance analysis suite can provide detailed analysis that offers deeper insight into I/O performance and scalability problems.

The investigated three-dimensional simulation test case `WD_Def` is a deflagration phase of the gravitationally confined detonation mechanism for Type Ia supernovae. In this mechanism, ignition occurs at one or several off-center points, resulting in a burning bubble of hot ash that rises rapidly, breaks through the surface of the star, and collides at a point opposite the breakout on the stellar surface [6]. This is an important astrophysical problem and the `WD_Def` test

case is generated as a weak scaling problem for up to 15,812 processors where the number of blocks remain approximately constant per computational thread.

2 I/O Measurements and Analysis

As indicated earlier, one of the key goals of this study is to understand and categorize the application behavior on a Teraflops-scale leadership computing platform. In order to achieve this goal, we systematically analyze I/O data that allows a better understanding of the complex performance characteristics of the parallel Lustre file system. We collect I/O data from FLASH on Jaguar for jobs ranging from 256 to 15,812 cores. From this weak-scaling study it is apparent that time spent in I/O routines began dramatically to dominate as the number of cores increased. As an example, for midsize and large core counts the time spent in I/O routines rose of a factor of >22 while the time spent in compute and communication routines was "only" of a factor of ∼1.5 higher from lower processor counts. Because it appears that the FLASH performance on the Jaguar system is extremely affected by the I/O performance of the Lustre file system, multiple tests have been performed with the goal of tuning and optimizing I/O performance so that the overall performance of FLASH can be signicantly improved.

Collective I/O via HDF5: For the FLASH investigation described here, the Hierarchical Data Format 5 (HDF5) is used as I/O library. By default, the parallel mode of HDF5 uses an independent access pattern for writing datasets and performs I/O without aggregating disk accesses for writes [7]. In that case, every I/O process operates independently and no communication among processes is required. However, signicant performance penalties may be expected when performing non-contiguous writes to disk. Otherwise, parallel HDF5 can also be run so that the writes to the file's datasets are aggregated, allowing the data from multiple processes to be written to disk in a single chunk. Although it involves network communications among processes, combining I/O requests from different processes in a single contiguous operation can yield a signicant speedup [8].

File Striping: Lustre is a parallel file system that provides high aggregated I/O bandwidth by striping file extents across many storage devices [3]. The parallel I/O implementation of FLASH creates a single file and every process writes its data to this file simultaneously which relies on the underlying MPI-IO layer in HDF5 [7]. The size of such a checkpoint file grows linearly with the number of cores. As an example, in the 15,812 core case the size of the checkpoint file is approximately 260 GByte when setting three checkpoints in total (beginning, middle, end). As mentioned in section 1.1, Jaguar consists of three file systems of which two have 72 OSTs and one has 144 OSTs. Hence, by increasing the default stripe size, the single checkpoint file may take advantage of the parallel file system which should improve performance.

Split Writing: Since the size of a checkpoint file grows linearly with the number of cores, I/O may perform better if all processes write to a limited number of separate files rather than a single file. Split file I/O can be enabled by setting the `outputSplitNum` parameter to the number N of files desired [7]. Every output file will be then broken into N subfiles. It is important to note that the use of this mode with FLASH is still experimental and has never been used in a production run. We experiment with different numbers for the parameter `outputSplitNum` ranging from 4 to 4,096.

A deeper trace-based investigation of the I/O behavior allows a better understanding of the complex performance characteristics of the parallel Lustre file system. Using various techniques like aggregating write operations in combination with files striping across all OSTs yields a significant performance improvement of a factor of 2 for midsize CPU counts and approximately 4.5 for large CPU counts for the entire FLASH application. Furthermore, writing data to multiple files instead of a single file delivers an additional performance gain of nearly a factor of 2.7 for 8,192 cores as an example. Since the size of the output files grows linearly with the number of cores, it is a future intent to find the optimal file size or optimal number of output files to obtain best performance for various core cases.

References

[1] Top500 list, June 2008, `http://www.top500.org/list/2008/06/100`

[2] J. Larkin and M. Fahey, "Guidelines for Efficient Parallel I/O on the Cray XT3/XT4", in: Proceedings of Cray User Group, 2007

[3] M. Jurenz, "VampirTrace Software and Documentation", ZIH, Technische Universität Dresden, `http://www.tu-dresden.de/zih/vampirtrace`

[4] A. Knüpfer, R. Brendel, H. Brunst, H. Mix, W. E. Nagel, "Introducing the Open Trace Format (OTF)", Proceedings of the ICCS 2006, part II. pp. 526–533, 2006.

[5] "VampirServer User Guide", `http://www.vampir.eu`

[6] G. C. Jordan, R. T. Fisher, D. M. Townsley, A. C. Calder, C. Graziani, S. Asida, D. Q. Lamb, J. W. Truran, "Three-Dimensional Simulations of the Deflagration Phase of the Gravitationally Confined Detonation Model of Type Ia Supernovae", The Astrophysical Journal, 681, pp. 1448–1457, July 2008.

[7] ASC FLASH Center University of Chicago, "FLASH Users Guide Version 3.1.1", January 2009.

[8] M. Yang, Q. Koziol, "Using collective IO inside a high performance IO software package - HDF5", www.hdfgroup.uiuc.edu/papers/papers/ParallelIO/HDF5-CollectiveChunkIO.pdf