TAUmon: Scalable Online Performance Data Analysis in TAU

Chee Wai Lee, Allen D. Malony, and Alan Morris

Department Computer and Information Science,
University Oregon, Eugene, Oregon, 97403

Abstract. In this paper, we present an update on the scalable online support for performance data analysis and monitoring in TAU. Extending on our prior work with TAUoverSupermon and TAUoverMRNet, we show how online analysis operations can also be supported directly and scalably using the parallel infrastructure provided by an MPI application instrumented with TAU. We also report on efforts to streamline and update TAUoverMRNet. Together, these approaches form the basis for the investigation of online analysis capabilities in a TAU monitoring framework TAUmon. We discuss various analysis operations and capabilities enabled by online monitoring and how operations like event unification enable merged profiles to be produced with greatly reduced data volume prior to application shutdown. Scaling results with PFLOTRAN on the Cray XT5 and BG/P are presented along with a look at some initial performance information generated from FLASH through our TAUmon prototype frameworks.

1 Introduction

As the level of parallelism increases in large-scale systems, performance measurement of parallel applications will be affected by the size of the performance data being maintained per process/thread, the effects of measurement overhead, and the cost of output, both during and at the end of execution. The traditional approach of post-mortem (offline) analysis of performance experiments will come under increasing pressure as the sheer volume and dimensionality of performance information drives up I/O and analysis complexity. Enhancing performance measurement systems with online monitoring support is a necessary step to address both challenges. In our prior research, we have explored extensions to the TAU performance system [9] that allow access to the parallel performance data measurement for an application at runtime. The TAUoverSupermon [13] (ToS) and TAUoverMRNet [12] (ToM) prototypes leveraged the online transport infrastructures, Supermon [15] and MRNet [1], respectively. While ToS and ToM demonstrate monitoring functionality, it is becoming increasingly clear that we need to push forward on the scalable algorithms for performance analysis and evaluate their efficiency in real application scenarios.

In this paper, we reconsider the approaches and capabilities of online performance monitoring from a perspective of the online operations necessary to support scalable performance measurement and runtime analysis requirements. In

M.R. Guarracino et al. (Eds.): Euro-Par 2010 Workshops, LNCS 6586, pp. 493–499, 2011.
© Springer-Verlag Berlin Heidelberg 2011

addition to updating ToM to support new machines, we investigate the approach of directly using the parallel infrastructure provided by MPI as an alternative, complementary monitoring framework. Together these different approaches for operations via different transports form the foundation of a framework for TAU performance monitoring we call *TAUmon*. We will structure the rest of the paper as follows. Related work is discussed in section 2. In section 3, we describe the statistical analysis operations enabled by TAUmon. Section 4 briefly covers the changes to ToM and the use of the MPI transport. Section 5 presents scaling results for analysis operations implemented using MPI and ToM for the PFLOTRAN [11] application and FLASH [4]. These experiments were conducted on Jaguar, Oak Ridge National Lab's Cray XT5 and Intrepid, Argonne National Lab's IBM BlueGene/P.

2 Related Work

Prior literature that guided our work can be classified roughly into 2 categories: work that seek to provide general monitoring interfaces and those that seek scalable transport support. The Online Monitoring Interface Specification (OMIS) [8] project provided a general interface between tools and a monitoring system. An event-action paradigm mapped events to requests and responses to actions as part of that interface. J-OMIS [2] was built on top of this interface to provide extensible monitoring facilities for the support of tools for Java applications. Specific target tools were performance characterization and debugging tools. Lee [7] explored the effectiveness of asynchronously collecting profile data transformed from performance traces generated within a running Charm++ application. The work demonstrated how an adaptive runtime system like Charm++ was able to serve as an effective transport medium for such purposes. Periscope [5] made use of hierarchical monitor agents working with applications and an external client in order to address scalability issues with transport. The Distributed Performance Consultant [10] in Paradyn made use of MRNet to support introspective online performance diagnosis. In addition, a number of computation steering frameworks [6,14,16,3] exist where performance information is collected, analyzed and fed through parameter-modifying interfaces in order to change an application's behavior. What distinguishes TAUmon is the attempt to design an integrated abstraction to TAU's interface for swapping independent transport mechanisms in order to provide maximum flexibility for efficiently delivering and processing performance data.

3 Online Performance Monitoring

Our efforts to build monitoring support have grown out of more general concerns of reducing the size, time, and number of files needed to offload parallel profile data at the end of the application execution. For several years, we have been extending the TAU measurement infrastructure with scalable capabilities to access

performance data online via different transports, what we call *performance monitoring.* There are several benefits to performance monitoring including offloading of performance data, on-the-fly analysis, and performance feedback. With performance data offload, it becomes possible to capture a running time-series snapshots of event statistics, histograms and cluster-averages. Presentation of these snapshots as they appear permits on-the-fly analysis, possibly on a remote client while the application is still executing. We now describe the monitoring operations currently supported as a part of the TAUmon framework.

Profile Merging: By default, TAU creates a file for every thread of execution in the parallel program. In this case, the number of files required to save the full parallel profile grows as an application scales. We have added to the monitoring framework an operation that merges profile streams in order to reduce the number of files required. The operation is a concatenation operation used only at the end of the run. The root processor requests for the profiles of the other processors one at a time. On receipt of these profiles, the data is immediately concatenated to a single file by the root processor to avoid running out of memory. When profile merging is coupled with and preceded by event unification, the gains in overall data volume when compared with the traditional approach of profile output are significant with low overhead costs. For example, 27 GB of PFLOTRAN profile output for 131k cores was reduced to 600 MB, taking 12.96 seconds on the Cray XT5.

Event Unification: In TAU, because events are instantiated locally, each thread assigns a unique event identifier. This results in full event maps that need to be stored for each thread. Event unification begins by sorting each processor's events by name. The information is propagated up the transport's reduction tree, where at each stage, the output to the next level of the tree is the unified and sorted by the list of events from the set of inputs. At the same time, each intermediate node in the tree maintains the reverse event maps. When the fully-unified event information arrives at the root, it is broadcast back through the tree during which the full event map for each processor can be constructed using the local maps maintained at each intermediate node. Note that event unification is the pre-requisite to all other monitoring operations that do not involve simple concatenation and thus require global event identification consistency.

Statistics Operations: Basic reductions in the transport support the computation of sum, sum of squares, minimum and maximum values of events across processors. From these, we can derive mean profiles and compute histograms with fixed-sized bins. The mean profile is useful as a summary from which additional statistical information can be sought via other operations. As time-series data, it is capable of highlighting the relative significance of events and their relative rate of growth. Histograms are useful for highlighting work distribution across processors for events which cannot be captured by mean profiles.

4 Transport Infrastructure Updates

Our original ToM instantiation scheme was designed to allow additional MRNet resources to be initialized flexibly and semi-transparently to both the application and job scheduling system. This was achieved to a reasonable degree through the splitting of MPI communicators as described in [12]. This instantiation scheme, however, did not foresee other flexibility problems in the way MRNet trees may have to be set up. For example, because of the way MRNet trees are currently implemented, intermediate tree nodes may only be allocated on the service nodes of BG/P. As a result, we have had to update ToM to make use of new versions of MRNet with the latter's support for much larger machines like the Cray XT5 and BG/P platforms and specialized batch environments.

In addition, we have adopted MPI as an online monitoring transport. This was derived from our work to reduce the overheads associated with end-of-execution profile output and analysis. We found it necessary to deal with a large number of profiles using more efficient profile representations. We had separately implemented parallel event unification, profile merging, and profile data analysis (average, min, max and histogramming) using MPI for use at the end of the execution, and considered how these solutions could be applied to other monitoring operations. The monitoring operations were then implemented in parallel using a binomial reduction tree based on the algorithms used in MPI reduction. Enabling them for online monitoring was a then simple matter.

5 Experiments and Results

Our experiments with TAUmon had two goals. First, we aimed to observe transport performance against different online monitoring operations. Second, we wanted to observe expected performance structures over time not normally captured by a final application profile. We targeted two applications: PFLOTRAN [11], a 3-D reservoir simulator that models subsurface reactive flows, and FLASH [4], a multi-physics multi-scale simulation code that has been used to simulate type Ia supernovae explosions. The input data set used for PFLOTRAN modeled a large 2 billion degree-of-freedom river simulation. This data set was used for both our preliminary experiments as well as our strong scaling experiments above 4,096 processor cores on the Cray XT5 and BG/P machines. For FLASH, we employed the Sod 2d input dataset with varying maximum refinement levels.

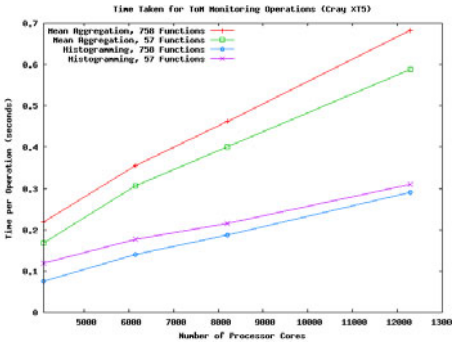

Fig. 1. Time taken for PFLOTRAN monitoring operations on the XT5 using ToM as the transport layer

At each major iteration, event unification followed by monitoring operations to compute per-event means and histograms over 20 bins. In the case of ToM, this is measured by the front-end process which marks the beginning and end of each operation's communication protocol. For MPI transport, the root process of our reduction tree takes responsibility for measuring when the collective operation was begun and when the performance data was finally gathered at the root. To get a perspective on the effects of data volume on online monitoring overhead, we made comparisons of PFLOTRAN when executed with full instrumentation against selective instrumentation of significant events. The full event set numbered around 756, while selective instrumentation yielded about 57 events.

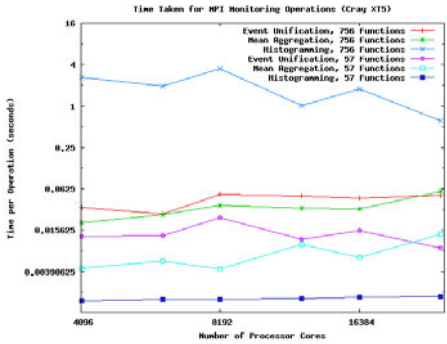

Fig. 2. Time taken for PFLOTRAN monitoring operations on the XT5 using MPI as the transport layer

Figure 1 shows the scaling results for the Cray XT5 using ToM from 4K to 12K cores. Since event unification has not yet been implemented with MRNet, we used MPI event unification results for the other analyses. All times are less than 0.7 seconds, with histogramming taking longer than averaging. Times increase with larger cores counts. Both of these results were expected. We are investigating performance in more detail to determine if optimizations are possible.

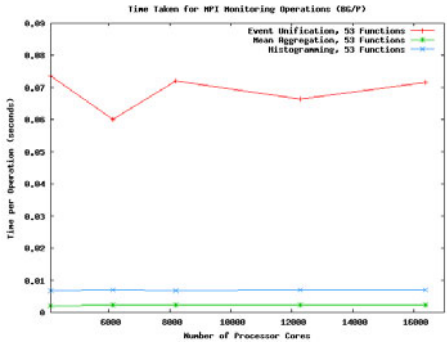

Fig. 3. Time taken for PFLOTRAN monitoring operations on the BG/P using MPI as the transport layer

In contrast, Figure 2 shows the scaling results for the Cray XT5 using the MPI transport for 4K to 24K cores. Except for histogramming on 758 events, MPI analysis operations are all less than 0.06 seconds. Compared to ToM, this is significantly faster. Furthermore, there is little effect of scaling on these times. Clearly, the anomaly is the histogramming results for 758 events. More investigation is needed to uncover the poor performance here. Our suspicions are that there is an interaction between the histogramming algorithm and the core locality boundaries that disrupt performance. In addition to being high, the execution times have the weird behavior of declining at larger scale.

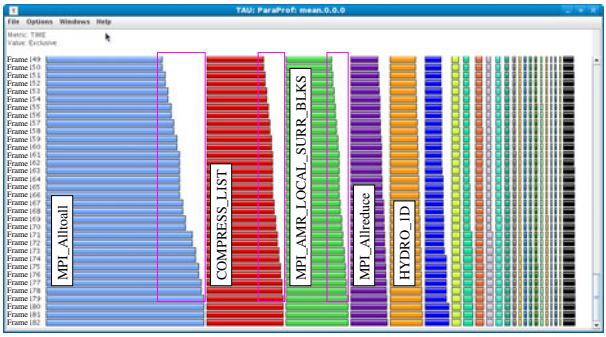

Fig. 4. Online profile snapshots of FLASH execution on 1,536 Cray XT5 processes

Moving to the IBM BG/P, Figure 3 shows the scaling results using MPI transport for 4K to 16K cores and 57 events. Again, the execution times are all less than 0.06 seconds. The interesting effect is the larger event unification time relative to mean and histogram analysis. This is also represented in the XT5 results for 57 events, keeping in mind that Figure 2 is a log-log plot. As before, monitoring analyses with MPI transport appears to be minimally affected by scaling.

Finally, our work with FLASH returned to online monitoring experiments to demonstrate how analysis of parallel profile snapshots taken during execution can highlight performance effects that would otherwise be missed in an aggregate profile. Figure 4 highlights 34 frames of mean profiles from 1,536 processes running FLASH on the Cray XT5. The most significant five events are labeled. The frames were chosen because they show the step-like behavior in the events associated with AMR operations.

6 Conclusions and Future Work

The TAU project is developing a scalable parallel monitoring framework called TAUmon, based on past research prototypes, but with an eye toward leveraging current scalable infrastructure like MRNet and high-performance MPI libraries. The results from initial experiments reported here give confidence that end-of-execution and online monitoring capabilities will provide opportunities for large-scale performance analysis. The parallel performance of the TAU event, merging, and reduction (mean, histogram) operations are good for both MRNet and MPI transport designs. We are currently developing other analysis operations, such as clustering and wavelet analysis, as well as tuning the monitoring analysis for higher efficiency. Long term, we hope to provide a monitoring interface for parallel applications to interrogate performance online from TAUmon, for purposes of adaptive performance optimization.

Acknowledgments. The research was by a grant from the U.S. Department of Energy, Office of Science, under contract DE-SC0001777.

References

1. Arnold, D.C., Pack, G.D., Miller, B.P.: Tree-based Overlay Networks for Scalable Applications. In: 11th International Workshop on High-Level Parallel Programming Models and Supportive Environments (HIPS 2006) (April 2006)
2. Bubak, M., Funika, W., Smętek, M., Kiliański, Z., Wismüller, R.: Architecture of monitoring system for distributed java applications. In: Dongarra, J., Laforenza, D., Orlando, S. (eds.) EuroPVM/MPI 2003. LNCS, vol. 2840, pp. 447–454. Springer, Heidelberg (2003)
3. Eisenhauer, G., Schwan, K.: An object-based infrastructure for program monitoring and steering. In: SPDT 1998: Proceedings of the SIGMETRICS Symposium on Parallel and Distributed Tools, pp. 10–20. ACM, New York (1998)
4. Fryxell, B., Olson, K., Ricker, P., Timmes, F.X., Zingale, M., Lamb, D.Q., MacNeice, P., Rosner, R., Truran, J.W., Tufo, H.: FLASH: An Adaptive Mesh Hydrodynamics Code for Modeling Astrophysical Thermonuclear Flashes. The Astrophysical Journal Supplement Series 131(1), 273–334
5. Gerndt, M., Furlinger, K., Kereku, E.: Periscope: Advanced Techniques for Performance Analysis. Parallel Computing: Current and Future Issues of High-End Computing, 15–26 (September 2005)
6. Gu, W., Eisenhauer, G., Schwan, K., Vetter, J.: Falcon: On-line monitoring for steering parallel programs. In: Ninth International Conference on Parallel and Distributed Computing and Systems, pp. 699–736 (1998)
7. Lee, C.W.: Techniques in Scalable and Effective Parallel Performance Analysis. PhD thesis, Department of Computer Science, University of Illinois, Urbana-Champaign (December 2009)
8. Ludwig, T., Wismuller, R., Sunderam, V., Bode, A.: OMIS - on-line monitoring interface specification (version 2.0). LRR-TUM Research Report Series, 9 (1998)
9. Malony, A.D., Shende, S., Bell, R., Li, K., Li, L., Trebon, N.: Advances in the TAU Performance System, pp. 129–144 (2004)
10. Miller, B.P., Callaghan, M.D., Cargille, J.M., Hollingsworth, J.K., Irvin, R.B., Karavanic, K.L., Kunchithapadam, K., Newhall, T.: The paradyn parallel performance measurement tools. Computer 28(11), 37–46 (1995)
11. Mills, R.T., Lu, C., Lichtner, P.C., Hammond, G.E.: Simulating Subsurface Flow and Transport on Ultrascale Computers using PFLOTRAN. Journal of Physics: Conference Series 78, 012051 (2007)
12. Nataraj, A., Malony, A.D., Morris, A., Arnold, D.C., Miller, B.P.: A Framework for Scalable, Parallel Performance Monitoring using TAU and MRNet. International Workshop on Scalable Tools for High-End Computing (STHEC 2008) (June 2008)
13. Nataraj, A., Sottile, M., Morris, A., Malony, A.D., Shende, S.: TAUoverSupermon: Low-Overhead Online Parallel Performance Monitoring. In: Kermarrec, A.-M., Bougé, L., Priol, T. (eds.) Euro-Par 2007. LNCS, vol. 4641, pp. 85–96. Springer, Heidelberg (2007)
14. Ribler, R.L., Simitci, H., Reed, D.A.: The autopilot performance-directed adaptive control system. Future Gener. Comput. Syst. 18(1), 175–187 (2001)
15. Sottile, M.J., Minnich, R.G.: Supermon: a high-speed cluster monitoring system. In: Proceedings of IEEE International Conference on Cluster Computing, 2002, pp. 39–46 (2002)
16. Tapus, C., I-Hsin Chung, Hollingsworth, J.K.: Active harmony: Towards automated performance tuning. In: ACM/IEEE 2002 Conference on Supercomputing, November 16-22, pp. 44–44 (2002)