07341 Abstracts Collection
Code Instrumentation and Modeling for Parallel Performance Analysis
— Dagstuhl Seminar —

Adolfy Hoisie[1], Barton P. Miller[2] and Bernd Mohr[3]

[1] Los Alamos National Laboratory, US
hoisie@lanl.gov
[2] Univ. Wisconsin - Madison, US
bart@cs.wisc.edu
[3] Forschungszentrum Jülich, DE
B.Mohr@fz-juelich.de

Abstract. From 20th to 24th August 2007, the Dagstuhl Seminar 07341 "Code Instrumentation and Modeling for Parallel Performance Analysis" was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. Abstracts of the presentations given during the seminar as well as abstracts of seminar results and ideas are put together in this paper. The first section describes the seminar topics and goals in general. Links to extended abstracts or full papers are provided, if available.

Keywords. Parallel programming, performance analysis, performance modeling, code instrumentation

07341 Executive Summary – Code Instrumentation and Modeling for Parallel Performance Analysis

Given the exponential increase in the complexity of modern parallel systems, parallel applications often fail to exploit the full power of the underlying hardware. At scale, it is not uncommon for applications to run at parallel efficiencies in the low single digits. Moreover, their optimization is extremely difficult due to the inherent complexity of the systems and the applications themselves. Therefore, a variety of projects have been developing tools and techniques for the measurement, analysis, and visualization of parallel program performance in order to help guide users in the optimization process. This meeting was the third in a series of seminars related to the topic "Performance Analysis of Parallel and Distributed Programs", with previous meetings being the Dagstuhl Seminar 02341 on "Performance Analysis and Distributed Computing" held in August 2002 and Seminar 05501 on "Automatic Performance Analysis" in December 2005. While

these seminars concentrated on the "analysis" part of performance analysis, at the most recent seminar the focus was on the building blocks of program instrumentation and modeling that are prerequisites for the analysis phase. As a result, the presentations of the participants concentrated on several fundamental issues related to instrumentation for generating high-quality performance data, methodologies for performance modeling leading to accurate predictions for the performance, and on the ways in which these techniques are combined for the performance analysis of applications and systems.

Keywords: Performance analysis, parallel programming

Joint work of: Hoisie, Adolfy; Miller, Barton P.; Mohr, Bernd

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2007/1267

Application Performance Modeling: Predictive Accuracy in the Presence of Simplifying Assumptions

Kevin Barker (Los Alamos National Laboratory, USA)

Many of today's large-scale parallel scientific codes exhibit characteristics that complicate the task of accurate performance prediction. Such characteristics include irregularly shaped sub-domains resulting from the use of third-party mesh partitioning tools and unstructured underlying meshes. However, an accurate performance analysis of such codes is critical for the development of new algorithms, code optimization, and system performance validation. From the point of view of the performance analyst, such codes present difficulties in understanding both the processing time on a single processor and the scaling characteristics that arise from execution on a parallel system. In this talk we will examine several such codes and demonstrate that accurate performance predictions on large parallel systems can be obtained after introducing simplifying abstractions that eliminate the complexities and idiosyncrasies arising in individual codes while maintaining essential performance and scaling characteristics. We will validate our approach by comparing performance measurements with predictions on several parallel systems.

Keywords: Application Performance Modeling

Timestamp Synchronization for Event Traces of Large-Scale Message-Passing Applications

Daniel Becker (Forschungszentrum Jülich, D)

Identifying wait states in event traces of message-passing applications requires measuring temporal displacements between concurrent events.

In the absence of synchronized hardware clocks, linear interpolation techniques can already account for differences in offset and drift, assuming that the drift of an individual processor is not time dependant.

However, inaccuracies and drifts varying in time can still cause violations of the logical event ordering. The controlled logical clock algorithm accounts for such violations in point-to-point communication by shifting message events in time as much as needed while trying to preserve the length of intervals between local events. In this talk, I describe how the controlled logical clock is extended to collective communication to enable a more complete correction of realistic message-passing traces. In addition, I present a parallel version of the algorithm that is intended to scale to thousands of application processes and outline its implementation within the framework of the scalasca toolkit.

Joint work of: Becker, Daniel; Rabenseifner, Rolf; Wolf, Felix

See also: Daniel Becker, Rolf Rabenseifner, Felix Wolf: "Timestamp Synchronization for Event Traces of Large-Scale Message-Passing Applications". Proceedings EuroPVM/MPI 2007, LNCS 4757, p. 315-325, Paris, France, September 2007. Springer.

Dyninst

Andrew Bernat (University of Wisconsin - Madison, USA)

The Paradyn Project is actively engaged with the deconstruction of Dyninst, our dynamic binary instrumentation tool. The goal of our project is to develop a suite of components that each provide a discrete piece of functionality in the areas of binary analysis, binary modification, and process inspection. The first components of the Dyninst deconstruction are the SymtabAPI, an API for reading and writing symbol and debug information from binary files; the Stackwalker, an extendable API for performing stack walks; and our abstract instruction representation. In this talk, we will provide an overview of the Dyninst deconstruction, focusing on these three components. We will also demonstrate two tools built on top of these components: Unstrip, a tool for recovering and reconstructing symbol tables in stripped binaries, and Stat, a tool that takes scalable stack walks.

Keywords: Dynamic binary instrumentation

Compiler Support for Scalable Program Instrumentation

Barbara M. Chapman (University of Houston, USA)

With precise information on code structure, and knowledge of the optimizations applied to a given program, a compiler has insights that may be exploited to enhance program instrumentation and tracing.

A compiler is able to help performance tools by selecting regions of code for measurement, potentially at both coarse (parallel region level, call path/procedure level) and fine granularity (control flow level). Sophisticated internal cost models in the compiler may enable static estimation of the relative importance of a region of code: this may be used to set individual thresholds that a region must meet in order to be instrumented. This approach has been shown to significantly reduce overheads for both profiling and tracing.

With support from the National Science Foundation, we are developing an integrated environment for application tuning that combines robust, existing, open source software - the OpenUH compiler, Dragon program analysis tool and three performance tools, TAU, KOJAK and PerfSuite. The goal of this work is to provide automated, scalable performance measurement and optimization to increase user productivity by reducing the manual effort of existing approaches.

As part of our collaborative effort, we have learned how to exploit the compiler to perform selective program instrumentation and thereby to enable more scalable approaches to performance analysis. We will show how our compiler was able to help select the important regions of code in several benchmarks and in a full-scale application, significantly reducing instrumentation overheads for the latter.

Keywords: Compiler tools integration, automatic instrumentation

Joint work of: Chapman, Barbara M.; Hernandez, Oscar

The Cray Performance Tools

Luiz De Rose (Cray Inc. - Mendota Heights, USA)

In order to achieve and sustain high performance on current and future supercomputers users need state of the art performance analysis tools to help detect and understand the large number of potential application performance bottlenecks, such as load imbalance, synchronization and I/O overhead, and memory hierarchy related issues. In this talk I will present with examples the Cray performance measurement and analysis infrastructure, a toolset consisting of the CrayPat performance collector and the Cray Apprentice2 performance visualizer. CrayPat and Cray Apprentice2 provide an intuitive and easy to use interface for performance tuning of scientific applications on Cray systems.

Application Performance Analysis on High Performance Computers

Sudip Dosanjh (Sandia National Laboratories - Albuquerque, USA)

This presentation discusses recent application scaling results on Cray XT3, IBM Purple and IBM Blue Gene systems. Also analyzed is the speedup of several science and engineering codes on dual core AMD opteron systems.

A recently developed Structural Simulation Toolkit (SST) is described. SST is a discrete-event simulation code that can be used to determine the impact of architectural changes/features on application performance. It has been applied to a variety of science and information codes and has been used to show that even "floating-point" intensive codes spend most of their time performing memory operations.

Keywords: Scaling, speedup, discrete-event simulation

(Are you sure that) Tracing is not scalable?

Judit Gimenez (BSC - UPC, E)

Executing an instrumented application so that timestamped records for all relevant events are emitted to a trace file can generate huge amounts of data. For this reason the approach is often deemed as impractical for analyzing long runs on large numbers of processors.

We do think that the answers to most of the performance analysis questions are on the details and on the variance that often is lost when the data is accumulated.

We will present the techniques by which the pretended scalability problem can be addressed and demonstrate examples of how our environment is able to capture and analyze with a lot of detailed applications with hundreds and thousands of processors.

Deep Binary Analysis & The Need for Cooperating Analyses

Jeff Hollingsworth (University of Maryland - College Park, USA)

As instrumentation needs grow more complex, instrumentation tools require a deeper analysis of binaries to better understand what the program is doing and how to insert their instrumentation. In this talk, I will describe some recent work to add binary slicing to the the Dyninst tools. I will also outline the need for different analysis techniques to share information. I will describe a framework we are developing in the Dyninst project to allow the sharing of information between different analysis modules.

Knowledge Support for Parallel Performance Data Mining

Kevin Huck (University of Oregon, USA)

Current parallel performance tools provide the ability to summarize and report statistical data about an application's performance on a given platform, and/or report the performance differences between two executions of a particular application.

However, few tools provide an explanation of performance results in a way that includes performance data results and the integration of the experiment context.

In order to take advantage of sophisticated instrumentation and produce more meaningful models, new analysis methods are needed which incorporate information beyond the raw performance data.

In this presentation, we will describe our design for a parallel performance analysis framework which integrates context metadata and performance assumptions, or "expert knowledge", about an experiment into the analysis process. Our framework implements an inference engine to encode and process the expert knowledge, provides user-configurable, scripted control over the process, and includes persistent storage of all intermediate results and analysis provenance.

We will also present analysis examples and describe our plans for further exploration of this analysis space.

Keywords: Parallel performance, data mining, expert systems

Joint work of: Huck, Kevin A.; Malony, Allen D.

See also: Kevin A. Huck, Allen D. Malony, Sameer Shende and Alan Morris: "Scalable, Automated Performance Analysis with TAU and PerfExplorer". Proceedings of ParCo 2007, NIC Series Volume 38, p629-636, Jülich, Germany, September 2007.

Random Access to Event Traces with OTF

Heike Jagode (TU Dresden, D)

The Open Trace Format (OTF) is a proportionally new trace definition and representation for use with large-scale parallel platforms. It utilizes a multi-file storage scheme with a variable number of so called streams to encourage parallel I/O. At the same time, strictly sequential reading of parallel traces is still supported.

As a special feature, OTF allows very fast selective access to process traces and even at arbitrary time intervals. Now, there is no need to always read traces linearly from the beginning. In order to make this really useful some provisions are required.

Firstly, concise overview information would be most convenient. They should point to certain sections worth accessing. After all, there is no sense in reading everything beforehand.

Secondly, resumption points are needed to start reading at an intermediate position. They need to provide the full status information for one or several processes at a given point in time. For example, they give the full call stack at current time stamp that all subsequent enter and leave events relate to.

The talk presents how both kinds of information are incorporated in OTF. Furthermore, it demonstrates a proof-of-concept tool taking advantage of the novel features.

Keywords: OTF, Open Trace Format, trace file, vampir trace, vampir, performance analysis

Joint work of: Jagode, Heike; Knuepfer, Andreas

Peering Inside Black Boxes: Strategies for Uncovering Scaling Issues In Lower Software Stack Levels

Terry Jones (LLNL - Livermore, USA)

Deciphering what is happening within lower software stack levels can be frustrating to both the tool developer and application developer. For example, it is difficult to determine if an application is exhausting a resource within the MPI library. Or suppose there is concern for operating system jitter; how does one decipher whether the operating system is interfering with an application's scalability? In this talk, I will describe and compare several strategies for dealing with lower software stack levels. The talk includes detail on one such strategy, the PERUSE api for examining MPI.

Keywords: MPI, PERUSE, API, tools

Inclusion of Co-Processor Usage in Program Traces

Guido Juckeland (TU Dresden, D)

A variety of different co-processors, ranging from special processing elements, graphic cards, physics accelerators, and cryptography units to FPGAs, are emerging in HPC today to provide an additional performance boost. One important aspect of this program "outsourcing" from the main processor to specialized hardware will be performance analysis and optimization to ensure the additional hardware can provide its full potential. While live monitoring and sampling of the hardware might indicate hardware limitations, a post-mortem trace analysis will provide insight what part of the application is suffering from a bottleneck.

In this talk I will present ideas on how a current trace facility could be extended to include execution data from a program running on a co-processor in general and (as a first approach) on FPGAs in particular. Possible access points for trace engines will be evaluated for their feasibility and limitations will be outlined.

Keywords: Tracing Co-Processor FPGA OTF

Implementation and Usage of the PERUSE-Interface in Open MPI

Rainer Keller (Universität Stuttgart, D)

In this short talk, we introduce the work presented at EuroPVM/MPI 2006.

The PERUSE interface may offer the tool developers a better understanding of the time lost within MPI, information that otherwise could not be gathered with PMPI-based tracing.

Keywords: PERUSE, MPI, Introspection, Tracing

Joint work of: Keller, Rainer; Bosilca, George

Full Paper:
http://www.hlrs.de/people/keller/PAPERS/pubs.html

See also: Rainer Keller and George Bosilca and Graham Fagg and Michael M. Resch and Jack J. Dongarra: "Implementation and Usage of the PERUSE-Interface in Open MPI". LNCS vol. 4192, p. 347-355, Bonn, Germany, September 2006. Springer.

Performance analysis and modeling of High Performance Networks

Darren Kerbyson (Los Alamos National Laboratory, USA)

The achievable network performance is governed both by the characteristics of individual communication channels (bandwidths and latencies) as well as by the contention within the network. Contention results from the simultaneous use of the network by a set of application communications. For instance, in possible Optical Circuit Switched (OCS) networks, datapaths can be dynamically arranged to match the communication pattern of an application. However, in current Infiniband networks messages are statically routed and thus do not take into account dynamic loading of the communication channels. In this work we explore the feasibility of an OCS for high performance computing by examining the requirements of an application, specifically its communication pattern. Similar analysis is applied to optimize the routing tables in a large-scale Infiniband cluster to reduce network contention. Empirical data confirms our contention calculation and custom routing tables for Infiniband are defined that provide optimum as well as worst-case performance for a large-range of communication patterns.

The IBM HPC Toolkit

David Klepacki (IBM TJ Watson Research Center, USA)

The IBM HPC Toolkit is an ongoing project at IBM Research, which has its emphasis on studying the simplification of performance analysis for HPC applications. It is an integrated framework for performance data collection and analysis constructed around five primary dimensions: cpu, memory, threads, messages, and i/o. The performance data is collected in such a way as to be

able to correlate this data with the corresponding data structures and source code statements, so that inferences can be made with regard to the structure of the application's program organization and the underlying hardware for its execution. These inferences can be used to make coding decisions that optimize execution for a given hardware configuration. The IBM HPC Toolkit supports IBM's pSeries servers, and includes the Blue Gene system, currently the most powerful computing system, as designated by Top500. This presentation will review the status of the IBM HPC Toolkit, address dynamic and scalable requirements that became necessary for Blue Gene, and discuss future directions.

Concurrent Approaches to Analyze I/O Problems

Michael Kluge (TU Dresden, D)

Within this talk different approches to analyze disk I/O are examined and evaluted. Not only on compute nodes but also on the whole SAN environment. The storage backends of supercomputers consists of multiple levels of disks, raid controllers, SAN switches as well as storage servers, HBA's and other components. All these devices add their share to the overall behavior of an I/O subsystem. Usually each component provides data that can help to understand the system as a whole. That data can be collected in different ways, like reading the /proc file system or using proprietary firmware interfaces.

Miscellaneous approaches to gather all this informations and to merge these with an application trace will be presented. Different (partly unusual) approches to gather all disk I/O on compute nodes are also introduced.

Keywords: Disk I/O, tracing, visualization

Using Communication Patterns for Parallel Program Optimization

Dieter Kranzlmüller (GUP-University of Linz, A)

Optimizing both performance and scalability of HPC software requires a high-level understanding of communication patterns as well as their relation to source code structures. Such patterns can be either user-defined or automatically extracted from program traces, requiring minimal interaction from users. In both cases, tough, they benefit from being associated with static program information to enable efficient and automatic source transformations.

In this talk we describe early experiences in extracting patterns from parallel traces and using this information to enhance static analysis and optimization of MPI codes. We first detect interesting patterns that identify potential optimizations in MPI communication traces and then associate them with the corresponding nodes in an abstract syntax tree. Finally, this combined information is used to guide static optimizations like code motion or the automatic introduction of MPI collectives.

Keywords: Communication patterns, scalability, performance optimization

Joint work of: Preissl, Robert; Kranzlmüller, Dieter; Schulz, Martin; de Supinski, Bronis; Quinlan, Dan

Characterization of parallel application

Jesus Labarta (BSC, E)

Performance analysis tools capture a significant amount of data about the behavior of an application. In today's typical practice, such data (either traces or profiles) is presented to the user in a relatively unprocessed forms, essentially global counts, statistics or timelines. This summarized information may give hint to the user/analyst of some aspects of the application's behavior but it is typically very far from presenting a clean, holistic description of the application performance/scalability. In this presentation we will describe an analysis methodology that combines of signal processing techniques, modeling of message passing architectures, clustering techniques and sequential processor performance models. The methodology results in an abstract but precise description of the scalability and performance of a parallel program. The methodology can also be automatized to directly generate reports describing which are the high level factors (and their quantitative impact) determining the observed behavior.

Exploiting the MPI Profiling Interface

Rusty Lusk (Argonne National Laboratory, USA)

The MPI forum decided not to define a standard set of tools or even a tool interface, but rather a specific method by which MPI calls could be intercepted and modified by users. This empowers tool writers, even those without access to the source code of an MPI implementation. In this talk I will remind everyone of some details of the MPI profiling interface and encourage the adoption of its key ideas in non-MPI contexts. I will then illustrate its use with three quite different profiling libraries: one for providing detailed trace files in a scalable format, one for scalability accumulating useful statistics and identifying often-hidden causes of performance problems, and a third for runtime checking of the use of MPI collective operations. I will demonstrate the use of the Jumpshot tool for graphically displaying trace files.

Keywords: MPI profiling interface Jumpshot

Constructing Application Performance Models using Neural Networks

Sally McKee (Cornell University, USA)

Increasing system and algorithmic complexity combined with a growing number of tunable application parameters pose significant challenges for performance modeling. We attack this problem via an automated approach that uses Artificial Neural Networks to build accurate, confident predictive models using only a sparse sample of the targeted parameter space.

We will augment this talk with live demonstrations of our toolset, which is built on top of freely available software.

We will show how these techniques can be applied to model the performance of two well known scientific benchmarks run on three large scale platforms at LLNL. Further, we will provide the audience with all information necessary to apply these techniques themselves on their own datasets.

Keywords: Performance modeling; performance prediction; neural networks

Joint work of: McKee, Sally A.; Schulz, Martin

Measurement, Analysis, and Modeling of Parallel Program Performance

John Mellor-Crummey (Rice University, USA)

In this talk, I will describe and demonstrate measurement and analysis of parallel program performance using components from Rice University's HPCToolkit; I will also describe work on detailed modeling and analysis of node program performance. I will begin with a brief overview of capabilities in HPCToolkit including strategies for call path profiling of optimized code, binary analysis for interpretation of performance measurements, and the hpcviewer interface for presenting profile-based performance data. I will then describe and demonstrate a new technique for identifying scalability bottlenecks in executions of SPMD parallel programs, quantifying their impact on performance, and associating this information with the program source code. Finally, I will provide an overview of our work on modeling the node performance of parallel programs, show examples of the models we compute, and describe how they have been used for application tuning.

Keywords: Call path profiling, binary analysis, scalability analysis, performance tools.

Full Paper:
http://doi.acm.org/10.1145/1274971.1274976

See also: Coarfa, C., Mellor-Crummey, J., Froyd, N., and Dotsenko, Y. 2007. Scalability analysis of SPMD codes using expectations. In Proceedings of the 21st Annual international Conference on Supercomputing (Seattle, Washington, June 17 - 21, 2007). ICS '07. ACM Press, New York, NY, 13-22.

Summary of CScADS Workshop on Performance Tools for Petascale Systems

John Mellor-Crummey (Rice University, USA)

This brief presentation provides an overview of a workshop held in Snowbird Utah in July 2007. The principal aim of the workshop was to explore opportunities for collaboration among members of the performance tools community. Workshop presentations and discussion summaries are available from the CScADS WWW site:
http://cscads.rice.edu/workshops/july2007/perf-slides-07

Performance Analysis and Tuning of Parallel/Distributed Applications

Anna Morajko (Universitat Autònoma de Barcelona, E)

This presentation summarizes the research line that focuses on automatic and dynamic performance analysis and tuning of parallel/distributed applications. Our research group investigates performance models that represent execution of such applications. We develop tools for a post-mortem performance analysis, a run-time automated performance analysis, and finally a dynamic performance tuning. We consider automatic tuning of applications developed by using programming frameworks or based on mathematical libraries. Our goal is to analyse and tune applications executed in cluster or grid environments.

Scalable Compression and Replay of Communication Traces

Frank Mueller (North Carolina State University, USA)

Characterizing the communication behavior of large-scale applications is a difficult and costly task due to code/system complexity and their long execution times. An alternative to running actual codes is to gather their communication traces and then replay them, which facilitates application tuning and future procurements. While past approaches lacked lossless scalable trace collection, we contribute an approach that provides orders of magnitude smaller, if not near

constant-size, communication traces regardless of the number of nodes while preserving structural information. We introduce intra- and inter-node compression techniques of MPI events and present results of our implementation for BlueGene/L. Given this novel capability, we discuss its impact on communication tuning and beyond.

To the best of our knowledge, such a concise representation of MPI traces in a scalable manner combined with deterministic MPI call replay are without any precedence.

Keywords: High-performance computing, performance analysis; communication tracing

Joint work of: Mueller, Frank; Noeth, Michael; Schulz, Martin; de Supinski, Bronis

Full Paper:
http://moss.csc.ncsu.edu/~mueller/ftp/pub/mueller/papers/ipdps07.pdf

See also: International Parallel and Distributed Processing Symposium, April 2007

Applying Performance Tools to Real World Applications

Matthias Müller (TU Dresden, D)

With petaflop systems consisting of hundreds of thousands of processors at the high end and multi-core CPUs entering the market at the low end application developers face the challenge to exploit this vast range of parallelism by writing scalable applications.

Any tool that is targeted to be used by real application developers should work reliably, be easy to use and fast to learn. In other words: it should be of help rather than adding another burden. In this talk we present some of the experiences with different applications that have been instrumented and analyzed at ZIH. The analysis covers all phases of performance analysis: instrumenting the application, collecting the performance data, and finally viewing and analyzing the data. Examined aspects include instrumenting effort, monitoring overhead, trace file sizes, load time and response time during analysis. Applied tools are mainly VampirTrace and Vampir, but also other tools like Kojak.

We conclude with a feature request list for tools from the user point of view. The planned/optional demo part could include challenges like “can your tool/expert find this performance bug faster?”

Joint work of: Müller, Matthias; Brunst, Holger; Henschel, Robert; Jurenz, Matthias; Knüpfer, Andreas; Mix, Hartmut; Nagel, Wolfgang

See also: Matthias S. Müller, Andreas Knüpfer, Matthias Jurenz, Matthias Lieber, Holger Brunst, Hartmut Mix, and Wolfgang E. Nagel: "Developing Scalable Applications with Vampir, VampirServer and VampirTrace". Proceedings of ParCo 2007, NIC Series Volume 38, p637-644, Jülich, Germany, September 2007.

The Impact of Scalable Tool Infrastructure on Systems with 3D Mesh and Torus Interconnection Networks

Philip Roth (Oak Ridge National Laboratory, USA)

Tree-Based Overlay Networks (TBONs) are an effective way to provide functionality in scalable performance and system administration tools. A TBON provides support for scalable data multicast, aggregation, and reduction operations using a collection of processes connected in a tree topology. There is great flexibility in the number, placement, and connection topology of these overlay network processes. Furthermore, the chosen overlay network characteristics have performance, reliability, and possibly economic implications. In this talk, we explore these implications for systems with 3D torus or mesh interconnection networks, such as the United States Department of Energy's Leadership Computing Facility Cray XT system deployed at Oak Ridge National Laboratory.

P^nMPI: A Dynamic MPI Tool Infrastructure

Martin Schulz (LLNL - Livermore, USA)

The PMPI profiling interface, as defined in the MPI standard, provides tool builders with an efficient way to instrument and monitor MPI applications. Although many tools have successfully used this interface, it also has its drawbacks: all tools are forced run in a global scope (i.e., monitor the complete application behavior); it is not possible to combine multiple tools in a single run; and tools lack support for tool modularity. These limitations restrict tool flexibility and increase the threshold for implementing PMPI tools.

In this talk and on-line demonstration I will present our work on P^nMPI, an infrastructure that supports multiple concurrent MPI tools while being binary compatible with the existing PMPI interface. It achieves this flexibility by enabling users to dynamically assemble and load stacks of multiple PMPI tools. Further, P^nMPI includes new services for tool interoperability and modularity as well as enables tools to switch dynamically between different tool stacks. The latter enables the transparent creation of scope specific tool stacks, i.e., allows the restriction of tools to dynamic subsets of communication calls.

I will demonstrate the features and capabilities of the P^nMPI infrastructure using three usage scenarios: the transparent combination of existing PMPI tools; the creation of a cooperative tool stack reusing common MPI tool components; and the transparent use of an MPI profiler on subsets of an MPI application.

Full Paper:
http://sc07.supercomp.org/schedule/pdf/pap224.pdf

See also: Martin Schulz, Bronis R. de Supinski: "P^nMPI Tools: A Whole Lot Greater than the Sum of Their Parts". Proceedings of SC 07, Reno, Nevada, November 2007.

System Architectures and Application Code Performance Bottlenecks

Jim Tomkins (Sandia National Laboratories - Albuquerque, USA)

This presentation will provide a general overview of current large-scale MPP supercomputer system architectures and architectural "features" that impact application code performance. The balance between processor speed, memory bandwidth and latency, and communication bandwidth and latency will be discussed. Also, the presentation will present some ideas on what future commodity based large-scale MPP supercomputers might look like in terms of the performance balance between processor speed, memory system performance, and communication performance.

Self-consistent MPI Performance Requirements

Jesper Larsson Träff (NEC Europe - St. Augustin, D)

The MPI Standard does not make any performance guarantees, but users expect (and like) MPI implementations to deliver good performance. A common-sense expectation of performance is that an MPI function should perform no worse than a combination of other MPI functions that can implement the same functionality. In this talk we formulate and discuss some performance requirements and conditions that good MPI implementations can be expected to fulfill by relating aspects of the MPI standard to each other. Such a performance formulation could be used by benchmarks and tools, such as SKaMPI and Perfbase, to automatically verify whether a given MPI implementation fulfills basic performance requirements. We present examples where some of these requirements are not satisfied, demonstrating that there remains room for improvement in MPI implementations.

Joint work of: Träff, Jesper Larsson; Gropp, Willieam; Thakur, Rajeev

See also: Jesper Larsson Träff, William Gropp, Rajeev Thakur: "Self-consistent MPI Performance Requirements". Proceedings EuroPVM/MPI 2007, LNCS 4757, p. 36-45, Paris, France, September 2007. Springer.

The Virtual Institute High-Productivity Supercomputing

Felix Wolf (Forschungszentrum Jülich, D)

The Virtual Institute High-Productivity Supercomputing (VI-HPS) is a joint institute of Forschungszentrum Jülich, RWTH Aachen University, Technische Universität Dresden, and the University of Tennessee.

It has been founded to combine the expertise of the four partners with the goal of improving the quality and accelerating the development process of complex simulation codes in science and engineering that are being designed for the most advanced parallel computer systems. Our mission is to develop integrated state-of-the-art programming tools for high-performance computing that assist programmers in diagnosing programming errors and optimizing the performance of their applications. The collaboration of the four partners is sponsored by the Helmholtz Association of German Research Centers.

Keywords: HPC productivity, programming tools, performance analysis, error detection

Chasing Scalability Bottlenecks: Recent Activities

Patrick Worley (Oak Ridge National Laboratory, USA)

We describe recent work in identifying and eliminating performance scaling bottlenecks in codes drawn from atmosphere, fusion, and ocean modeling. The focus is on algorithmic bottlenecks and the labor-intensive methodology used to identify the problems and to evaluate the solutions. We hope to engage the performance tool and modeling communities in a discussion on how their tools and methodologies might improve the process of optimizing parallel algorithms in codes in active development.

Keywords: Parallel algorithms, performance scalability, performance optimization methodology

Large-scale application measurement and analysis with Scalasca

Brian J. N. Wylie (Forschungszentrum Jülich, D)

The Scalasca toolset has been designed to scale to tens of thousands of processess and successfully used to instrument, measure and analyse the execution of MPI applications on a variety of leadership-class systems.

Real-world examples include ASC SMG2000 with 22k processes on Cray XT3/4, WRF-NMM with 1600 processes on Mare Nostrum and XNS CFD solver with 4096 processes on BlueGene/L, which were successfully summarised at runtime, traced and the traces analysed with a collection and analysis manager.

Scalability problems encountered and techniques employed to circumvent them will be presented with real-world examples, and concluded with on-going work for further improvement.

Keywords: Application instrumentation, performance measurement, performance analysis

Joint work of: Wylie, Brian J. N.; Geimer, M.

See also: Markus Geimer, Björn Kuhlmann, Farzona Pulatova, Felix Wolf, and Brian J. N. Wylie: "Scalable Collation and Presentation of Call-Path Profile Data with CUBE". Proceedings of ParCo 2007, NIC Series Volume 38, p645-652, Jülich, Germany, September 2007.