Dagstuhl Seminar 02341
on
Performance Analysis and Distributed Computing

Organized by

Michael Gerndt (Technische Universität München)
Vladimir Getov (University of Westminster)
Adolfy Hoisie (Los Alamos National Laboratory)
Allen Malony (University of Oregon)
Barton Miller (University of Wisconsin)

Schloß Dagstuhl, 19. – 23.08.2002

Contents

1 Preface

The performance of parallel and distributed systems and applications - its evaluation, analysis, prediction and optimization - is a fundamental topic for research investigation and a technological problem that requires innovations in tools and techniques to keep pace with system and application evolution. This dual view of performance "science" and performance "technology" jointly spans broad fields of performance modeling, evaluation, instrumentation, measurement, analysis, monitoring, optimization, and prediction.

Most of the past and current research on performance analysis is focused on high-performance computing using dedicated parallel machines since performance is the ultimate goal in this environment. Future applications in the area of high-performance computing will not only use individual parallel systems but a large set of networked resources. This scenario of computational and data grids is attracting a lot of attention from application scientists as well as from computer scientists. In addition to the inherent complexity of program tuning on parallel machines, the sharing of resources and the transparency of the actual available resources introduce new challenges on performance analysis systems and performance tuning techniques. To meet those challenges, experts in parallel computing have to work together with experts in distributed computing. Aspects such as network performance, quality-of-service, heterogeneity, and middleware systems - just to mention a few - will have a big impact on the performance of grid computing.

Therefore, the workshop brought together people from high-performance and distributed computing to discuss the impact of new the aspects of grid environments on performance analysis tools and techniques.

The topics covered in the workshop came from six areas:

1. Grid Computing

   The presentations concentrated on programming aspects of Grids. In addition, UNICORE was presented as an representative Grid architectures as well as performance aspects of Web servers were introduced.

2. Parallel Architectures

   This area covered quite diverse aspects, such as mobile agents, cellular architectures in the context of the IBM Blue Gene project, and the Quadrics interconnection network being part of the ASCI Q machine.

3. Performance Analysis Tools and Techniques

   Two major aspects where covered in this area: scalability of performance analysis tools and tool automation. Other presentations covered performance analysis for Java, performance visualization, and performance data management.

4. Performance Modeling

   Performance prediction and its application in different contexts, such as DSM multiprocessors, large scale systems, task parallel programs and grid computing was the focus of the workshop in this area.

5. Performance Analysis and Grid Computing

   The presentations in this area gave an overview of the current approaches on monitoring and performance analysis in several grid projects, i.e. Crossgrid, Datagrid, and DAMIEN.

6. Performance Optimization

   Performance optimization is the major reason for performance analysis. In Grid environment, optimization mainly means optimizing scheduling decisions in dynamic and heterogeneous environments as well as online performance tuning or performance steering.

The presentations during the seminar led to two evening discussions on *Grid Computing* and on *Future Architectures*. Major open questions raised in the Grid Computing discussion were:

- Will QoS ("Quality of Service") become reality in the context of grids? The opinion of the group was that this depends fully on economic reasons. If people will pay for using the grid, performance guarantees are required.
- Does the analogy of the Grid and the electrical power grid hold? Two major differences were identified: users of resources pay for those resources and, second, users transfer information into the grid which raises major security problems.
- How will Grids be used? The majority of people favored the concept of virtual organizations as the main usage of the Grid instead of metacomputing applications. The major programming paradigm might be a component based approach.

In the area of future architectures many questions were raised:

- How will the available chip space be used? Several approaches were suggested as possible candidates, such as combining scalar and vector processing, processor in memory systems, as well as multiprocessor and multithreading architectures.
- When will Quantum Computing become reality? The prevaling opinion seemed to be that this technology will take at least another 50 years.
- What will be the role of reconfigurable architectures?

The workshop highlighted that multiple approaches are currently pursued for grid monitoring. The Global Grid Forum defined the Grid Monitoring Architecture (GMA) which is based on a directory service for detecting producers of monitoring data and consumers requiring specific data. The data transfer itself, between producer and consumer, is realized via individual connections.

In all three projects presented in the workshop, i.e. DAMIEN, Datagrid, and Crossgrid, system-level and application-level monitoring are not integrated. Only the Datagrid project uses a unified infrastructure for system-level and application-level monitoring, the Relational Grid Monitoring Architecture (R-GMA). But, the information on system-level is not taken into account in performance analysis with GRM/Prove. This integration is the only way to assess performance data measured on application-level, i.e. to answer the question whether bad performance results from application coding or from dynamic changes in the resource capabilities.

Only with the assumption of QoS on the target computers and the network performance, analysis for grid applications ignoring system-level information makes sense. The DAMIEN approach is based on QoS and thus pure application-level monitoring can be used to analyze the application and grid component information with the help of VAMPIR.

A closer integration of system-level and application-level monitoring as well as the integration of runtime performance analysis and performance optimization (performance steering) will be very important in grid environments and will certainly be the focus of the future work of the APART working group (www.fz-juelich.de/apart).

Michael Gerndt, Vladimir Getov, Adolfy Hoisie, Allen Malony, Barton Miller

2 Grid Computing

2.1 Performance, Domain Decomposition, and the Grid
Cliff Addison, University of Manchester

In this talk, a new solution technique for mesh-based problems in computational engineering is described. This technique allows users to reuse existing component meshes to build a domain mesh in a simple drag and drop fashion. Domain decomposition can be applied to define a mathematically valid way to solve such problems on a mesh.

This approach increases computational time, but it allows existing application codes to be applied to each individual component mesh, so the process is inherently parallel. Furthermore, the granularity of this approach is such that it could be deployed on a Grid environment.

The performance issues relating to such a deployment, particularly load balancing are discussed.

2.2 Optimistic Grid Computing
Craig A. Lee, The Aerospace Corporation

Computational grids offer a vast pool of computational and storage resources that can be utilized by large-scale engineering and scientific computing applications. The efficient utilization of these vast resources, however, is difficult. Grids provide an environment where resources and performance can be dynamic and heterogeneous, hence, loosely coupled applications will be naturally grid-friendly. However, grids offer a way of aggregating more compute power than can be economically done any other way. Hence, there is strong motivation to find ways that enable more tightly coupled applications to effectively utilize grid resources. This talk presented a newly funded project to investigate the use of a speculative or optimistic model of execution to help applications cope with the grid performance environment. Such models can "loosen-up" an application by relaxing synchronization requirements but at the cost of "roll-backs" that must occur when incompatible results are produced by allowing optimistic computation. Mesh generation will be used as a first test case, where optimism is controlled by the insertion of mesh points at partition boundaries (as determined by the goal of preserving the Delauney criterion). This project will quantify the benefits and overheads of optimistic grid computing, initially for mesh generation, and generalize the execution support tools for use in other application domains.

2.3 The UNICORE Grids and its Options for Performance Analysis
Mathilde Romberg, Forschungszentrum Jülich

UNICORE (Uniform Interface to Computer Resources) is a software infrastructure to support seamless and secure access to distributed resources. It has been developed by the projects UNICORE and UNICORE Plus in 1997 - 2002 (funded by the German Ministry of Education and Research) and is going to be enhanced in the EU-funded projects EUROGRID and GRIP. The UNICORE system allows for uniform access to different hardware and software platforms as well as different organizational environments. The core part is the abstract job model. The abstract job specification is translated into a concrete batch job for the target system. Besides others application specific support is a major feature of the system. By exploiting the plug-in mechanism support for performance analysis of grid applications can be added. As an example support for VampirTrace has been integrated. The UNICORE user interface then gives the option to add a task using VampirTrace with runtime configuration support into a UNICORE job and retrieve the generated trace files for local visualization. Together with the support for compilation and linkage and for metacomputing the plug-in mechanism may be used to integrate other performance analysis tools.

2.4 Critical Issues in Web Performance
Arun Iyengar, IBM Research

This talk provides an overview of critical issues and key techniques for improving Web performance. For improving server performance, multiple Web servers can be used in combination with efficient load balancing techniques. We compare the load balancing capabilities of DNS and connection routers We also discuss how the choice of server architecture affects performance. We discuss caching and content distribution networks (CDN's). While Web performance can be improved using caching, a key problem with caching is consistency. We present different techniques for achieving varying forms of cache consistency. Finally, we show how dynamic and personalized pages can be at least partially cached by breaking up Web pages into fragments.

2.5 Programming the Grid with Distributed Objects and Components
Thierry Priol, IRISA/INRIA

Computational grids are seen as the future emergent computing infrastructures. They raise the problem of how to program them. We advocate a programming model based on a combination of parallel and distributed paradigms, namely message-passing, distributed objects and components. However, current distributed object

and component technologies are not really well adapted to get the maximum performance of such computing infrastructures. The talk outlines two problems we identified: the encapsulation of message-based parallel codes into distributed objects/components and the inability of these technologies to exploit various networking hardware such as those available in a computational Grid. The first problem is solved by an extension of the CORBA architecture, we called Parallel CORBA object or PaCO/PaCO++, to encapsulate parallel codes into objects while maintaining a scalable connection. The second problem is addressed by an open integration framework, that is called PadicoTM, that allows several communication middleware and runtimes to share efficiently the networking resources available in a computational grid.

3 Parallel Architectures

3.1 Performance Characteristics of Cellular Architectures
Jose Castanos, IBM TJ Watson Research Center

Blue Gene is a research project at IBM T.J. Watson whose goal is to design petaflops scale supercomputers. In this talk we describe two of the architectures that we have developed, Blue Gene/L and Blue Gene/C, and we present early performance figures of the latter one. Both designs are cellular systems: simple, homogeneous and relatively independent processing elements (or cells) that integrate memory, processing logic and interconnection hardware are replicated in a regular fashion to form systems with 10,000s to 100,000s cells and large aggregated performance.

Blue Gene/L is a joint collaboration with LLNL to design a 180TF machine. This system-on-a-chip architecture is based on embedded PowerPC technology. Blue Gene/C is a new experimental architecture based on three principles: it is cellular within a chip, allowing the placement of large number of very simple processing units in a chip; processor-in-memory technology provides the large bandwidth required by these processors; and simultaneous multithreading to tolerate latencies. To evaluate this architecture we developed a fine-tuned suit of common and simple scientific kernels. Using a software based simulator of the architecture, we demonstrate the effect of compiler optimizations, different cache latencies and memory bandwidth on the performance of BG/C.

3.2 Collective Communication Patterns on the Quadrics Interconnection Network
Salvador Coll, Technical University of Valencia

The efficient implementation of collective communication is a key factor to provide good performance and scalability of communication patterns that involve global data movement and global control. Moreover this is essential to enhance the fault-tolerance of a parallel computer. For instance, to check the status of the nodes,

perform some distributed algorithm to balance the load, or, even, synchronize the local clocks or do performance monitoring. For these reasons the support for multicast communications can improve the performance and resource utilization of a parallel computer.

The Quadrics interconnect (QsNET), which is being used in some of the largest machines in the world, provides hardware support for multicast. The basic mechanism consists of the capability for a message to be sent to any set of contiguous nodes in the same time it takes to send a unicast message. The two main collective communication primitives provided by the network system software are the barrier synchronization and the broadcast. Both of them are implemented in two different ways, either using the hardware support, when nodes are contiguous, or a balanced tree and unicast messaging, otherwise.

In this presentation some performance results are given for the above collective communication services, that show, on the one hand, the outstanding performance of the hardware-based primitives even in the presence of a high network background traffic; and, on the other hand, the relatively lower performance achieved with the software-based implementation. In addition, it is shown that, with current and future machines with several thousands of nodes that are likely to have some faulty nodes, the hardware-based services will be unusable in practice.

An overview of a new mechanism to overcome the limitations of the hardware-based primitives is presented. It is based on the implementation of a tree-based multicast algorithm in which each transaction of the tree is a hardware-based multicast addressed to a set of contiguous nodes, instead of a point-to-point message. In this way the depth of the tree will be dramatically reduced.

3.3 Performance Evaluation of Mobile-Agent Systems Marios Dikaiakos, University of Cyprus

In this presentation we described work on a hierarchical framework for quantitative performance evaluation of Mobile-agent Middleware. This framework is established upon an abstraction of the typical structure of mobile-agent systems and applications, which identifies "basic elements" and "application frameworks" as core concepts that need to be examined from a performance perspective, in order to investigate the performance of mobile-agent applications and to compare different platforms quantitatively.

We propose an implementation of this framework as a hierarchy of benchmarks. In particular, micro-benchmarks, micro-kernels and micro-applications. Micro-benchmarks seek to characterize the performance traits of "basic elements" of MA middleware. Micro-kernels are implementations of "application frameworks," i.e. skeletons of popular MA patterns and relevant models of distributed computation; micro-kernels are executed upon simple workloads. Micro-applications are micro-kernels extended with particular functionalities and running on realistic workloads; for instance, a client-server application providing Web-database access under a TCP-W workload.

We have implemented the first two layers of the hierarchy on top of three MA platforms: Aglets, Concordia and Voyager, and ran numerous experiments. Results provide insights as to the performance of MA middleware and a basis for comparing the different platforms quantitatively.

3.4 Mobile agents for Distributed Computing - performance issues
Beniamino DiMartino, Seconda Universita di Napoli

The Mobile Agents model has the potential to provide a flexible framework to face the challenges of high performance computing on distributed, heterogeneous and dynamically changing execution environments. Main features of the MA paradigm, mobility and adaptivity, cannot be exploited optimally without a performance aware online analysis of the execution environment and of the applications' characteristics, aimed at controlling and dynamically adapt the MA application execution to the changes of the execution environment and application behavior. In this talk a framework is presented for supporting programming and execution of mobile agent based distributed applications, the MAGDA (Mobile AGents Distributed Applications) tool set. It supplements mobile agent technology with a set of features for supporting parallel programming on a dynamically heterogeneous distributed environment. Its features are introduced, and related performance issues and devised possible solutions are considered.

4 Performance Analysis Tools and Techniques

4.1 A Path to Performance Diagnosis on 1000+ Nodes
Barton P. Miller, University of Wisconsin

Current performance profiling tools are limited in the size of the system on which they can measure application programs. These limitations are based on (1) the amount of data that must be transferred between the components of the tool and (2) the centralized control and monitoring that is done at the tool front-end process.

The Paradyn project is developing four new techniques which, when used together, will allow Paradyn to be used to effectively profiling applications on 1000+ node systems. The four major components of this effort are:

1. A software multicast/reduction network layer (MCRNL) that allows for efficient distribution of control operations and gathering of results from the nodes. MCNRL incorporates innovative reduction operations and will support a fault tolerant recovery facility.

2. A scalable start-up protocol that all reduces the amount of front-end to daemon communication needed. In systems with 1000's of node, the initial interaction with the corresponding 1000's of daemons can be overwhelming.

3. A distribute Performance Consultant that uses MCRNL to efficiently evaluation global (all node) bottlenecks and distributes evaluation of local (node specific) bottlenecks to local Performance Consultant agents. As part of this effort, we have developed a new, more detailed model of instrumentation overhead and feedback.

4. Sub-Graph Folding, a scalable visualization technique for displaying the results of the Performance Consultant's bottleneck search. This technique exploits the regular structure of SPMD programs, to combine results into equivalence classes and presenting only the exemplars of the class.

4.2 Scalable Parallel Program Analysis
Dieter Kranzlmueller, Johann Kepler Universität Linz

Scalability is an important issue of parallel processing, if applications require the performance and memory capabilities of parallel and distributed computing infrastructures. This fact must also be addressed by performance analysis and debugging tools by providing certain means to cope with the large-scale, long-running programs and the associated high amount of analysis data. This talk discusses some of the issues in developing scalable parallel program analysis tools, and offers an overview of the DeWiz tool set approach. The idea of DeWiz is to use the event graph as a model of a programs execution behavior, and to use this model as the target for abstraction and automated operations. The analysis activities are implemented as DeWiz modules with the provided DeWiz API in order to simplify the development of new event graph analysis techniques.

4.3 Distributed Performance Analysis
Michael Gerndt, Technische Universität München

The presentation described a the design of a new analysis system that performs an automatic search for performance problems on terascale systems. This search is guided by a specification of performance properties based on the APART Specification Language. The system is being implemented as a network of analysis agents that are arranged in a hierarchy. Higher level agents search for global performance problems while lower level agents search local performance problems. Leaf agents request and receive performance data from the monitoring library linked to the application. The analysis system is currently being implemented for the Hitachi SR8000 teraflop computer at the Leibniz-Rechenzentrum in Munich within the Peridot project.

4.4 Uniform Resource Visualization: Software and Services
Diane T. Rover, Iowa State University

Computing environments continue to increase in scale, heterogeneity, and hierarchy, with resource usage varying dynamically during program execution. Computational

and data grids and distributed collaboration environments are examples. To understand performance and gain insights into developing applications that efficiently use the system resources, performance visualization has proven useful. However, visualization tools often are specific to a particular resource or level in the system, possibly with fixed views, and thus limit a user's ability to observe and improve performance. Information integration is necessary for system-level performance monitoring. Uniform resource visualization (URV) is a component-based framework being developed to provide uniform interfaces between resource instrumentation (called resource monitoring components, RMC) and performance views (called visualization components, VC). URV supports services for connecting VCs to RMCs, and creating multi-level views, as well as visual schema definitions for sharing and reusing visualization design knowledge. This talk presents the URV model, including RMC, VC, and connector elements. It compares URV elements to the Global Grid Forum's Grid Monitoring Architecture. It describes two key issues: connection between RMCs and VCs and reuse of VCs. Connection and composition services are built upon component, metadata, and object messaging technologies.

4.5 Comparative Profiling with PPerfDB
Karen L. Karavanic, Portland State University

The PPerfDB research project is developing methods for diagnosing the performance of large-scale parallel applications using data from more than one program execution. This multi-execution view facilitates comparison of different code versions and different platforms. PPerfDB provides a common interface to performance data collected with a variety of trace or dynamic instrumentation based tools. We can now demonstrate: incorporation of data spanning different application versions, collected using a variety of measurement tools; use of our prototype to navigate through this data; and an experiment launch facility to collect performance data that is requested but not already available. PPerfDB's PPerfXchange module allows geographically dispersed data stores to be included as sources of data for PPerfDB. PPerfXchange models each site's performance data as XML documents based upon a global XML schema; data is requested and returned using the XQuery Language. Each site utilizes a mapping to translate between the actual format of its local data, and the single global schema.

4.6 Distributed Java applications: dynamic instrumentation and automatic performance optimization
Paul H. J. Kelly, Imperial College, London

This talk began by showing a prototype tool for dynamic instrumentation (in the tradition of Barton Miller's DynInst tool) for Java. This supports instrumentation of distributed Java applications by on-the-fly deployment of interposition code at user-selectable program points. The virtual JVM framework on which this is built forms part of a more ambitious attempt to optimize distributed RMI and EJB applications

automatically. We presented preliminary results showing run-time aggregation and short-circuiting ("server forwarding") of RMI calls.

4.7 Aksum: A Tool for the Search of Performance Problems in Parallel/Distributed Programs based on Multi-Experiment Analysis
T. Fahringer, University of Vienna

Aksum is a new tool for multi-experiment automated performance analysis that executes and evaluates an application for a set of problem and machine sizes in order to locate all important performance problems. The user can control the search for performance problems (properties) by restricting the performance analysis to specific code regions, by creating new or modifying existing property specifications and property hierarchies, by indicating the maximum search time and maximum time a single experiment may take, by providing thresholds that define whether a property is critical, and by indicating conditions under which the search for properties stops. Aksum automatically decides which regions are instrumented and controls the evaluation of performance properties in a property hierarchy. Heuristics are incorporated to prune the search for performance properties. We present experiments with a financial modeling code in order to demonstrate the usefulness of Aksum.

4.8 Advances in the TAU Performance System
Allen D. Malony, University of Oregon

Increasing complexity in parallel and distributed systems and software necessitates advances in performance technology towards robust tools and broad implementation. However, system and software complexity introduces challenges for performance instrumentation, measurement, analysis, and visualization. This talk presents recent advances in the TAU performance system in four areas: instrumentation control, performance mapping, performance interaction and steering, and component software. In the area of instrumentation control, we are concerned with the removal of instrumentation in cases of high measurement overhead using rule-based analysis. Performance mapping focuses on measuring performance with respect to dynamic calling paths when the static call-graph cannot be determined prior to execution. A prototype online performance data access, analysis, and visualization tool is reported as an example of performance interaction and steering system. Finally, we describe our preliminary approach to performance analysis of component software based on performance technology integration in component systems.

4.9 Automatic Performance Analysis of Hybrid OpenMP/ MPI Programs with EXPERT
Bernd Mohr, Forschungszentrum Jülich

The EXPERT performance-analysis environment provides a complete tracing-based solution for automatic performance analysis of MPI, OpenMP or hybrid applications running on parallel computers with SMP nodes. EXPERT describes performance problems using a high level of abstraction in terms of common situations that result from an inefficient use of the underlying programming model(s). The set of supported problems is extensible and can be custom tailored to application-specific needs. The analysis is carried out along three interconnected dimensions: class of performance behavior, call-tree position, and thread of execution. Each dimension is arranged in a hierarchy, so that the user can investigate the behavior on varying levels of detail. All three dimensions are interactively accessible using a single integrated view.

4.10 Automatic Performance Analysis with Kappa-Pi
Tomàs Margalef, Universitat Autònoma de Barcelona

Performance analysis and tuning of parallel/distributed applications are very difficult tasks for non-expert programmers. It is necessary to provide tools that automatically carry out these tasks. These can be static tools that either carry out the analysis on a post-mortem phase or can tune the application on the fly. Both kind of tools have their target applications. Static automatic analysis tools are suitable for stable application while dynamic tuning tools are more appropriate to applications with dynamic behavior. KappaPi is an example of a static automatic performance analysis tool that takes trace file, makes an automatic performance analysis detecting performance bottlenecks, determines the causes of such bottlenecks and provide some suggestions to the user to improve the performance of the application. On the other hand, applications with a dynamic behavior require an automatic performance tuning. In this case, the tool must monitor the parallel/distributed application, determine the performance problems, evaluate the required solutions to overcome these problems and apply the solutions during the execution of the application. The general environment based on parallel patterns for developing and dynamically tuning parallel/distributed applications we are developing is oriented to reach these goals.

4.11 Performance Analysis of Hybrid Programming Models
Vladimir Getov, University of Westminster

With the trend in the supercomputing world shifting from homogeneous machine architectures to hybrid clusters of SMP nodes, a performance study of different programming models and their combinations is of benefit for both the vendors and the user community. As part of our project, we have focussed on analysis and experiments to test the behavior of hybrid MPI and OpenMP micro-codes. In this

context, we have identified the various possible combinations of the two models in order to provide qualitative and quantitative justification of situations in which any one of them is to be favored. Collective operations are particularly important to analyze and evaluate on a hybrid platform and therefore we concentrate our study on three of them - barrier, all-to-all, and all-reduce. The performance results supporting our investigation were taken on the IBM Power-3 machine at SDSC using our suite of microbenchmarks developed at Los Alamos National Laboratory.

5 Performance Modeling

5.1 Scal-Tool: Pinpointing and Quantifying Scalability Bottlenecks in DSM Multiprocessors
Josep Torrellas, University of Illinois - Urbana

Distributed Shared-Memory (DSM) multiprocessors provide an attractive combination of cost-effective mass-market architectural design and, thanks to the shared-memory abstraction, relative ease of programming. Unfortunately, it is well-known that tuning applications for scalable performance in these machines is often a time-consuming effort. While performance monitoring tools can help, they often present only low-level information, often lack integration, and are usually costly to run.

In this talk, I present an empirical model that isolates and quantifies scalability bottlenecks in parallel applications running on DSM machines, in a relatively inexpensive and integrated manner. The scalability bottlenecks currently quantified include insufficient caching space, load imbalance, and synchronization. The model uses as inputs measurements from hardware event counters in the processor. A major advantage of the model is that it is quite inexpensive to run: it only needs the event counter values for the application running with a few different processor counts and data set sizes.

5.2 Performance Prediction: Where the Rubber Meets the Road
Adolfy Hoisie, Los Alamos National Laboratory

In this talk we present a brief introduction to performance modeling and its applications. We introduce the methodology for modeling, and the techniques developed at Los Alamos that allow modeling of entire applications. A number of models for applications from the ASCI workload are explained and analyzed. We also show applications of modeling to performance analysis, system architecture design, system integration and application performance improvement. Resources and other relevant work in the area are shown at the end of the talk.

5.3 Performance Prediction of Large Systems
Darren J. Kerbyson, Los Alamos National Laboratory, USA

In this work we present performance models that can be used to gain insight into the achieved performance of large-scale systems. The achieved performance depends upon a multitude of factores in terms of both system resource performance characteristics and also in the way in which an application uses them.

A model for an adaptive mesh refinement application is presented. This is representative of part of the ASCI workload. It is shown that after a suitable validation process a performance model can be trusted and used in many ways including:

- explanation of the performance on existing machines
- predicting the performance on possible future/larger systems
- during system installation (to validate actual measurements)
- in procurement of large systems
- in the comparison of performance between systems.

An overview of the Earth Simulator is also briefly described in which the achievable performance is compared to systems built from AlphaServer SMP nodes.

5.4 Performance Modelling for Task Parallel Programs
Thomas Rauber, Universität Halle-Wittenberg

The presentation describes a modeling approach for task parallel programs. Many applications from scientific computing and physical simulations can benefit from a mixed task and data parallel implementation on parallel machines with a distributed memory organization, but it may also be the case that a pure data parallel implementation leads to faster execution times. Since the effort for writing a mixed task and data parallel implementation is large, it would be useful to have an a priori estimation of the possible benefits of such an implementation on a given parallel machine. In this article, we propose an estimation method for the execution time that is based on the modeling of computation and communication times by runtime formulas. The effect of concurrent message transmissions is captured by a contention factor for the specific target machine. To demonstrate the usefulness of the approach, we consider a complex method for the solution of ordinary differential equations with a potential for a mixed task and data parallel execution. As target platform, we consider a Cray T3E and a Beowulf cluster.

5.5 Performance and Fungibility of Resources
Scott B. Baden, University of California, San Diego

A technical challenging in employing heterogeneous resources is how to enable the resources to be viewed as fungible, or interchangeable entities. To cope with this problem, application programmers traditionally write program variants covering the range of possible candidate platforms that they plan to use. However, this model is not feasible if the candidate resources cannot be known in advance. This talk discusses canonical variant programming that enables an application to optimize its behavior at run time, according to user supplied application and performance meta-data. A model is described, and some preliminary results are presented.

6 Performance Analysis and Grid Computing

6.1 Monitoring of Interactive Grid Applications
Marian Bubak, AGH Cracow

The subject of this talk is an application monitoring system called OCM-G – Grid-enabled OMIS Compliant Monitoring System. The OCM-G is meant to enable investigation and manipulation of parallel distributed applications running on the Grid, and it provides a basis for building tools supporting development of parallel applications for the Grid.

This research is a part of the EU CrossGrid Project [1]. The applications of the CrossGrid Project are characterized by interaction with a person in a processing loop and they are compute- and data-intensive [2]. To facilitate development of these applications, the Project develops a tool, named G-PM, which not only allows to measure just the standard performance metrics, but allows to determine higher-level performance properties and application specific metrics [3]. This, in turn, requires a specific monitoring infrastructure [4].

Our approach to Grid monitoring is application oriented - it provides information about running applications and it works on-line. The information is collected by means of run-time instrumentation which can be activated or deactivated on demand, what results in a reduced overhead of monitoring, since we can focus on the data we are interested in at the given moment.

The OCM-G provides abstraction and increases modularity: the tools may be developed independently from the monitoring system.

OMIS (On-line Monitoring Interface Specification) is applied as an universal interface between the OCM-G and tools. We have extended OMIS to fit the Grid requirements. A new layer called sites has been added to OMIS object hierarchy. We have also introduced the following new services, related to the Grid environment: for sites, for application and infrastructure metrics, for benchmarks, for application handling and for probes. In this form OMIS is consistent with the GGF's Grid Monitoring Architecture.

Monitoring applications on the Grid should be reliable, scalable, and efficient. For this reason there are two types of components out of which the monitoring system will be composed: Local Monitor (LM) and the Service Manager (SM). A SM is a part of OCM-G to which tools submit requests and from which they receive replies. SMs are permanent and related to sites in the target system. LMs are transient (created and destroyed when needed) and they reside on each node of the target system. A LM receives OMIS requests from a SM, executes them and passes the reply back.

Each OCM-G component is composed of the following modules: core, communication, internal localization, external localization, services, request management, application context, users.

The important aspect of the Grid monitoring is security issues since the single monitoring system will handle many users executing different applications. An authentication mechanism is foreseen to ensure that a user is allowed to monitor only his own applications.

The first prototype of the OCM-G is expected by December 2002.

References:

1. EU CrossGrid Project IST-2001-32243: http://www.eu-crossgrid.org

2. Bubak, M., Malawski, M., and Zajac, K.: Towards the CrossGrid Architecture. In Proc. 9th European PVM/MPI Users' Group Meeting, Linz, Austria, Sept. 2002, LNCS, Springer-Verlag

3. Bubak, M., Funika, W., and Wismueller, R.: The CrossGrid Performance Analysis Tool for Interactive Grid Applications. In Proc. 9th EuroPVM/MPI Users' Group Meeting, Linz, Austria, Sept. 2002, LNCS, Springer 2002

4. Balis, B., Bubak M., Funika W., Szepieniec, T., and Wismueller R.: An Infrastructure for Grid Application Monitoring. In Proc. 9th European PVM/MPI Users' Group Meeting, Linz, Austria, Sept. 2002, LNCS, Springer-Verlag

6.2 Performance Analysis in GRID Environments Hans-Christian Hoppe, Pallas GmbH

Compared to the well-established field of application performance analysis, GRID applications and systems do present interesting new challenges: their behavior is highly dynamic, they involve heterogeneous components, and they impose a hierarchy of analysis levels from the complete GRID system down to a single application running on the GRID. Dynamic control over data acquisition and advances in the organization of data are clearly required to allow meaningful analysis of large production GRID applications and systems.

The European research project DAMIEN (IST-25406) is implementing prototypes of a GRID-enabled MPI, an application coupling library and a GRID-capable performance analysis tool. The talk presents the design and implementation of the

performance analysis tool: based on the well-known Vampir and Vampirtrace MPI-related tools, the monitoring component has been extended to a GRID application, using GRID services itself to collect and consolidate performance data, generic performance counters have been introduced to capture numerical information (like load averages, achieved bandwidth etc.) changing over time, and hierarchical structures can be defined for the active elements (processes or threads) involved. Finally, the trace data can be subdivided into logical pieces (frames) by time, active elements or kind of data, and physically striped across multiple files to increase I/O performance and file locality. Each frame has precomputed statistics associated with it, allowing the graphical analysis tool to very quickly provide an overview of the available frames and let the user chose any subset for an in-depth analysis.

6.3 Monitoring of Grid Applications
Norbert Podhorszki, Hungarian Academy of Sciences

The GRM application monitoring tool and the PROVE performance visualization tool are integrated parts of the P-GRADE graphical parallel programming environment. P-GRADE applications are designed graphically; the process topology and the communication instructions are defined by graphics. The applications can be executed on clusters or supercomputers. GRM and PROVE support the on-line monitoring and performance visualization of the application. With the emergence of the grid technologies, we would like to use GRM and PROVE as tools for grid applications and to separate them from P-GRADE. Two different directions has been explored. First, a stand-alone version of GRM has been created. For its usage on the grid, its start-up mechanism had to be changed completely. Instead of the full control of P-GRADE and the knowledge about the configuration, GRM now starts-up together with the application. The key issue to achieve this is that the local monitor code is linked into the application and it is forked out at start of the application process. The other direction is to connect GRM to the RGMA, a relational implementation of the Grid Monitoring Architecture in the EU DataGrid project. The instrumentation library and the main monitor process of GRM has been kept only while the functionality of the local monitors is provided by RGMA. The talk presents these two ways of turning GRM to be a grid application monitor.

6.4 Performance Analysis on the Grid Activities in the CrossGrid Project
Roland Wismüller, Techn. Univ. München

In March 2002, the European project CrossGrid has been launched. Its goal is to make the Grid usable for interactive applications, i.e. applications with a "person in the loop". In order to support the optimization of such applications, a new performance analysis tool G-PM is developed within CrossGrid. Its distinctive features are:

- G-PM works in on-line mode, i.e. the user can specify measurements while the application is running and can immediately see the results. This allows to correlate application performance data with user interactions and avoids the necessity of storing large trace files.
- Besides standard application oriented measurements, G-PM will be able to display performance data on the Grid infrastructure. This data can be used for comparison, but also for resource selection and interactive steering.
- Finally, G-PM enables a user to define new metrics, which can be based on already existing metrics and/or an application-specific instrumentation. This allows to examine application-specific performance data, like e.g. response time.

The talk motivates the requirements that will be met by G-PM and presents its structure and major design ideas.

7 Performance Optimization

7.1 On the Nature and Implications of Performance Control John Gurd, University of Manchester

Starting from a "whole process" view of distributed, heterogeneous and dynamic high-performance applications, we argue that performance analysis is a part of a bigger process which entails controlling resource allocation and consequent performance in order to implement a chosen "whole system" performance strategy. We call this process performance control, and note the close analogy between performance control systems and classical negative feedback control systems. The theory and practice of the latter are well-developed, and offer important lessons for the future development of stable performance control systems. In particular there is a need for fast and accurate means for analyzing instantaneous performance levels and for well-understood mechanisms for changing the configuration of, and resources available to (and, hence, the performance of), the underlying computations.

7.2 Performance Steering in RealityGrid Graham Riley, University of Manchester

We present an approach to Grid application development involving their construction from components which provide mechanisms to enable them to react to changes in their execution environment. We term such components "malleable". In this talk, the design of a Performance Steering system being developed in the UK e-Science test-bed project RealityGrid, is presented. Given an initial deployment of a set of malleable components onto a set of Grid resources, the system will automatically

steer the overall performance of the application at run-time, as its behavior and the Grid resources available for its use change dynamically.

The mechanisms which a malleable component may make available to the Performance Steering system include the ability to directly utilize less (or more) processors, memory or bandwidth etc, or to be able, on request, to change their algorithm to one which requires less (or more) resources.

Applications exhibit phased behavior in their execution and their demand for resource use will vary in different phases. The notion of "progress points" which mark these phases is introduced. Progress points are points in the applications lifetime where past performance can be reviewed and performance in the next phase steered in order to achieve some overall performance goal. Finally, the role of performance modeling, performance prediction and performance analysis in the making of (automated) steering decisions is discussed.

7.3 Economic Scheduling in Grid-Computing
Carsten Ernemann, Universität Dortmund

Grid computing is a promising technology for future computing platforms. Here, the task of scheduling computing resources proves difficult as resources are geographically distributed and owned by individuals with different access and cost policies. This talk addresses the idea of applying economic models to the scheduling task. To this end a scheduling infrastructure and a market-economic method is presented. As one example ot the efficiency of this approach the response time minimization is evaluated by simulations with real workload traces. The evaluations show that the presented economic scheduling algorithm provides similar or even better average weighted response-times as common algorithms like backfilling. This is especially promising as the presented economic models have additional advantages as e.g. support for different price models, optimization objectives, access policies or quality of service demands.

7.4 Autonomous Bandwidth-Centric Scheduling of Heterogeneous Grids
Larry Carter, University of California at San Diego

Consider a heterogeneous distributed computing platform and a large collection of independent, equal-sized tasks. Assume that all data for the tasks originate from a distinguished node. The subset of the computing resources that should be used to process the tasks depends on the communication and computation requirements of the tasks and the speeds of the computing components. If the network can be modeled as a tree, then the "bandwidth-centric bound" (described in Beaumont, Carter, Ferrante, Legrand and Robert's paper in IPDPS 2002) is an easy-to-compute bound on the throughput of the system. This rate is asymptotically achieved by a simple scheduling policy: each node has a buffer that can hold K tasks; when the pool

is not full, the node requests replacements from its parent; and the parent prioritizes the requests according to the child's communication time (NOT the computation speed!).

This result offers the hope that an autonomous and adaptive scheduler can be used for such problems. All scheduling decisions can be made indpendently by the processors, using only locally-available information. Our group has been working to turn this hope into a practical scheduling algorithms. One problem is determining the pool size K. Our simulations and analysis have shown that using preemptive communication of tasks can greatly reduce the size of K required. We are also looking at other issues, including how to model a general network as a tree, and how to optimally schedule unequal-sized tasks.

7.5 Pipelining for Performance Optimizations in Scientific Computing
Gudula Rünger, TU Chemnitz

Many problems in scientific computing are modeled by time-dependent partial differential equations which result in large systems of ordinary differential equations when e.g. solved by the method of lines. Runge-Kutta methods are popular methods for the solution of systems of ordinary differential equations and are provided by many scientific libraries. Since a large number of consecutive time steps is usually needed to simulate a suitable time interval it is advantageous to find an efficient implementation for a single time step. For processors with memory hierarchy, the locality of data referencing pattern has a large impact on the efficiency. In the talk several transformations have been presented which improve the locality behavior on many recent processors. The transformations are based on properties of the solution method but are independent from the specific application problem or the specific target machine, so that the resulting implementation is suitable as library function. A block-based pipelining approach with variable blocksize leads to further performance gains for a specific class of applications.