Design Strategies for Real-Time High-Performance Multimedia Applications on Multiprocessors

Per Stenström and Jan Jonsson
Department of Computer Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
{pers,janjo}@ce.chalmers.se

Tomas Möller
Prosolvia/Clarus
SE-412 50 Gothenburg, Sweden
tompa@clarus.se

Summary

Emerging computational problems in e.g. multimedia need high performance and impose real-time requirements in terms of explicit deadlines. Important examples are visualization algorithms in virtual-reality (VR) applications that compute images at a predefined sampling rate. Multiprocessors appear to be promising platforms to meet the performance requirements of these applications because of the inherent parallelism in the problems. While the performance goal can be met by parallel processing, a high performance has traditionally been in conflict with real-time guarantees owing to the considerable gap between average-case and worst-case performance. This project will focus on concepts in terms of (i) *quality-of-service negotiating scheduling algorithms* that adjust the quality-of-service (QoS) level dynamically and (ii) *worst-case performance analysis and improvement techniques* to reduce the gap between average and worst-case performance. The developed concepts shall make it possible to design parallel programs that can achieve high performance on a wide range of multiprocessors under real-time requirements.

To evaluate the concepts, we will apply them to visualization algorithms developed in other research groups and at Prosolvia/Clarus. The project group consists of two PAMP-nodes: Chalmers and Prosolvia/Clarus. Two senior researchers and two Licentiate/Ph.D. students from Chalmers will be actively working on the project. In addition, a scientist at Prosolvia/Clarus will contribute to the project with expertise in visualization algorithms and by providing examples and case studies. The project will use performance prediction methods developed at another PAMP-node (SICS) as one essential experimental resource.

1. Problem statement and main ideas

Multimedia is one of the key drivers of high-performance computer platforms. Apart from needing more performance than existing platforms can deliver, stringent computational deadlines are often a key requirement for such applications. This is because multimedia systems shall react on user intervention and deliver streams of digitized images or sound at a predefined sampling rate. To meet these computational deadlines, quality-of-service in terms of e.g. realism of the computed images is often detrimentally affected.

Multiprocessing appears to be a viable approach to get higher performance, and consequently a higher quality-of-service, for multimedia applications because of the inherent parallelism that can be exploited. For example, recent research in parallel-algorithm implementations of visualization algorithms (e.g., radiosity, ray tracing, and volume rendering) has demonstrated near-linear speedup on symmetric and distributed shared-memory multiprocessors. Unfortunately, current parallelization methods completely ignore predictability requirements often associated with multimedia applications. Because current parallelization methods only provide a high average performance, they are not well-suited to multimedia applications.

On the other hand, advances in distributed real-time scheduling have resulted in scheduling algorithms that can meet real-time requirements. Unfortunately, they do this at the expense of poor performance. The reason for a poor performance level is twofold:

- Discrepancy between average and worst-case performance
- The static nature of existing distributed real-time scheduling algorithms

First, worst-case performance analysis is a prerequisite to real-time scheduling algorithms. Typically, the program is analyzed off-line and safe estimates of the worst-case execution time are then derived. Despite recent progress in worst-case performance analysis, current methods are not applicable to multiprocessors. Moreover, improvement techniques for multiprocessor programs have targeted average rather than worst-case performance.

Even though the discrepancy between average and worst-case performance were low, current off-line distributed scheduling algorithms are not applicable to multimedia applications because of the predominantly dynamic nature of these applications. Imposing a static structure on such applications results in poor performances. As demonstrated by recent work on parallel visualization algorithms, dynamic scheduling algorithms are needed to exploit the performance potential of multiprocessor systems.

The goal of the proposed research is to fill two missing parts to parallelize multimedia applications on multiprocessors to reach a high performance under real-time constraints:

- Techniques that improve the worst-case performance of multiprocessor programs
- Dynamic real-time scheduling algorithms that adjust quality-of-service to available resources and current computational need.

We carry out this research in two Licentiate/Ph.D student projects that are described below.

Worst-case performance analysis and improvement techniques. In order to develop improvement techniques, a necessary methodological prerequisite is to have access to an analysis framework of the worst-case performance of multiprocessor programs. Available methods for estimating the worst-case performance (or worst-case execution time) are, however, mainly applicable to single-processor systems. Typically, timing models of the processor and the memory subsystems are derived and the program is analyzed off-line to encounter the worst-case effects of these features.

We plan to extend this methodology to multiprocessor systems. Because multiprocessors consist of pipelined processors with a memory hierarchy, the basic approaches taken for analyzing single-processor systems seem applicable to multiprocessors. However, the new aspects that need to be considered in the methodology are to also encounter the effects of coordination and communication.

Our approach to extend existing analysis methods is based on the following key observations. Since coordination and communication take place through a shared address-space in multiprocessors, the main problem to solve is to extend worst-case modeling of caches to also encounter the worst-case effects of cache coherence interactions. The analysis can take advantage of statically known properties in the program such as the existence of read-write and read-only data structures. For example, read-only data do not result in coherence actions which simplifies the analysis. The analysis can also take advantage of the fact that coordination is confined to explicit synchronization constructs such as lock and unlock primitives. Our approach to analyze the worst-case performance of these primitives is to analytically determine the worst-case performance of the memory-system transactions that result from them.

After having established an analysis framework, we will develop methods that can improve the worst-case performance of shared-memory programs, especially with respect to shared-memory access, as follows. Latency reducing and hiding techniques are important means to achieve a high performance in multiprocessors. While such methods have been considered in the open literature, virtually no studies have addressed how they can improve worst-case performance which is fundamentally different from how they improve average performance. In terms of latency-reduction techniques, we plan to study cache blocking techniques that have a potential to improve the temporal locality in a predictable way. In terms of latency-hiding techniques, we plan to study bulk data transfer, prefetching, and relaxation of memory consistency models, and multithreading in the context of improving worst-case performance.

Quality-of-service negotiating scheduling algorithms. In cases where a worst-case performance improvement is not sufficient to guarantee a desired performance at run-time, an alternative is to resort to QoS negotiation techniques. The main rationale for choosing a QoS-based approach in favor of an off-line approach is that the system designer can (i) write real-time applications without having to consider the capacity of the target hardware, (ii) write the application once, then install it on an arbitrary platform and have it adapt its timing guarantees transparently to the programmer in accordance with platform capacity, and (iii) write parallel applications without having to devise a particular allocation of application components to hardware resources.

The objective of this project is to develop principles for QoS negotiation whose main purpose is to guarantee that the user-perceived utility of the system is kept at an acceptable level. More specifically, the goals of our work are twofold. First, we want to devise techniques for expressing flexible timing requirements for the application in such a way that target resource capacity and location are abstracted away from the programmer. Second, we want to devise run-time scheduling heuristics that can adapt their behavior dynamically in accordance with available resource capacity and designer specifications.

Our approach for reaching these goals can be described as follows. In VR systems, and other systems where the dynamics of the environment preclude a pre-determined application schedule, the QoS will be a function of the scheduling algorithm's capability to adapt to changes in computational demands and resource availability. In order to give the scheduling algorithm as much freedom as possible in its decisions, it is of paramount importance that one defines for the application not a single set but a spectrum of real-time and QoS requirements in the form of *primary* and *alternative* levels. While traditional real-time systems typically only have one such level, the existence of multiple levels gives the scheduler an increased freedom to “negotiate” with the application concerning a realistic QoS level for a given system state. Existing QoS-negotiation techniques proposed in the literature are based on a course-grain application model, where run-time statistics collection and load balancing take place at the granularity of a process. These techniques cannot take advantage of the performance-improving mechanisms of a modern multiprocessor system which take place at a finer-grain application level, e.g. thread- or instruction-level. In order to circumvent this limitation, we plan to develop QoS-negotiation techniques that operate at a thread-level, or finer, granularity. We also plan to investigate whether QoS-negotiation techniques will benefit from on-line preprocessing operations, such as deadline distribution or periodicity adjustments, as a means for attaining higher QoS. The rationale behind this extension is the demonstrated usefulness of such operations for statically-scheduled systems. Existing QoS techniques do not provide support for such operations.

2. Expected results and impact

The expected research contributions are as follows:

- Worst-case performance analysis methods for multiprocessor programs. Current work on worst-case performance analysis only considers single-processor systems.
- Worst-case performance improvement techniques for multiprocessor programs. Current work has mainly focused on average performance.
- Heuristic thread-level algorithms that maximize the user-perceived QoS for high-performance multiprocessor platforms. Current work on QoS negotiation algorithms only recognize process-level and not intra-application parallelism.

By applying our methods to multimedia applications provided to us from Prosolvia/Clarus, the research will demonstrate practical methods to achieve a high performance under real-time requirements for interesting test cases. The industrial impact is potentially methods to meet performance demands of similar applications using multiprocessor technology. Since multimedia is a rapidly emerging application domain, the research is expected to act as a framework to enable new applications.

3. Project plan

Experimental plan. In order to focus on relevant issues, the first phase of the project will analyze parallel implementations of visualization algorithms and identify their performance and real-time issues in detail. This investigation is carried out by project members at Chalmers and algorithm experts at Prosolvia.

The second phase aims at developing concepts for QoS improvement (scheduling and performance improvement techniques). In order to evaluate the merits of the proposed concepts, we will carry out application case-studies based on the parallel visualization algorithms available from the first phase. Performance of the algorithms will be evaluated using multiprocessor platforms as well as simulation models. The research group has access to three types of platforms: a 14-processor Sun Enterprise server, a 64-processor SGI Origin 2000, and a 64-processor Sun Enterprise 10000.

A simulation platform, called SimICS, and developed at SICS (another PAMP-node) will be an important experimental resource in this project. The platform consists of highly-tuned instruction-set processor simulators to which memory system simulators can be attached. On top of this multiprocessor simulation platform, a multiprocessor version of Linux has been successfully ported in a previous collaborative project between SICS and Chalmers. We will use this platform as a testbed for implementing scheduling algorithms as well as evaluating and identifying performance bottlenecks in the visualization algorithms.

Milestones and deliverables. 980901-030901. 2 x 2 years of research and one year of teaching.

- **Activity 9807-9809:** ARTES/PAMP student recruitment.
- **Activity 9809-9909:** First phase of project. In this phase the application requirements will be analyzed by selecting study objects in terms of parallel visualization algorithms to identify performance and real-time issues. **Deliverable 9909:** Selection of study objects in terms of parallel implementations of visualization algorithms. State-of-the-art report on performance and real-time issues based on preliminary evaluations. Refined specifications of the Licentiate project topics and project plans.
- **Activity 9909-0103.** Development of concepts for worst-case performance improvement techniques of multiprocessor programs and QoS scheduling algorithms. **Deliverable 0103:** Licentiate theses. Specification of the activities in the remaining time period.
- **Activity 0103-0303.** Development of worst-case performance improvement techniques and scheduling algorithms. Generalization of results. **Deliverable 0303:** Demonstration of developed methods on multiprocessor platforms as well as simulated platforms and detailed evaluation of performance and predictability.
- **Activity 0303-0309.** Writing of Ph.D. theses. **Deliverable 0309:** Ph.D. theses.

4. Preliminary budget

We ask for funding of two Ph.D. students. The activity level of the students is 80%. Thus they are expected to get a Licentiate/Ph.D. degree in 2.5/5 years. Potential mobility actions that can be relevant are to invite guest professors or send Ph. D. students to leading groups in this area. The group has a broad international contact net in the multiprocessing and real-time communities. Two such relevant groups in the U.S. would be Professor Jaswinder Pal Singh's group at Princeton and Professor Kang Shin's group at University of Michigan.

The initial annual funding we ask ARTES to support is 800,000 SEK. although additional funding might be asked for above mobility actions.

5. Related research

The most noticeable work on parallel program implementations for this application domain is Professor J. P. Singh and his associates work on parallelization methods of visualization algorithms on SMPs and CC-NUMA machines. These methods offer a high average performance but do not take into account the real-time requirements imposed on these algorithms from the application side. Consequently, they do not

address worst-case performance explicitly. Work on worst-case performance has only targeted single-processor systems and we are not aware of any work targeting multiprocessor systems.

Related work on QoS negotiation techniques for distributed real-time systems is done at the University of Michigan and work on integrated scheduling of multimedia and hard real-time tasks on shared-memory multiprocessors is done at University of Massachusetts. Although their proposed methods work well in a general perspective, they do not target parallelism in shared-memory multiprocessor programs.

6. Industrial relevance

Shared-memory multiprocessing has mainly been applied to scientific applications. The application class targeted in this project has challenging requirements and properties that are expected to represent typical multimedia applications in the future. The methods developed in the project are expected to be applicable and useful for a quite broad range of industrial applications.

7. Relation to PAMP and other SSF programs

The main motivation for PAMP is methods to use multiprocessors in performance-demanding real-time applications. This project matches the key goals of PAMP in that it will focus on performance as well as real-time requirements for an important class of applications that can benefit from multiprocessing. Moreover, apart from directly engaging two PAMP-nodes, it will also draw on the developments in the project carried out at SICS. While difficult to predict at this stage, the project plans to apply the methods to applications with similar requirements such as the applications at Ericsson Software Technology in Ronneby.

We see the following connections to other SSF programs. The Personal Computing and Communication (PCC) program targets mobile communication to enable multimedia applications and focuses on the communications infrastructure. Because a high-performance computing infrastructure is also needed, the PAMP program in general, and this project in particular, complements the work in the PPC program. The National Graduate School in Scientific Computing is also sponsored by SSF. The focus of developing basic technology to use SMPs for high-performance applications in PAMP will complement the more user-oriented activities in this graduate school. Finally, the Excellence Center in Computer Science and Systems Engineering (ECSEL) stresses reduction of development time and cost for new products. This goal is also part of PAMP and ARTES, although these programs focus on real-time high-performance systems.

8. Context

This research will be carried out in the high-performance computer architecture group at Chalmers headed by Per Stenström. The main focus of the group is design principles and design methods for high-performance computer systems with a current focus on multiprocessor systems. The projects that are currently running are (1) an SSF/TFR-funded project on memory system architectures for multiprocessors; (2) a NUTEK and TFR-funded project on multiprocessor system architectures for transaction-oriented systems (3) a TFR-funded project on worst-case timing analysis techniques for single-processor systems. The group collaborates with research groups at University of Southern California (Michel Dubois) and University of Michigan (Kang Shin). It also collaborates with computer manufacturing industry (Sun Microsystems) as well as computer system design industry (Ericsson). The proposed project has no overlap with these projects but will have fruitful synergies.

Appendix A: Curriculum Vita

A.1 Short CV for Per Stenström

Affiliation

Chalmers University of Technology, Dept. of Computer Engineering, S-412 96 Göteborg, Sweden, Phone: +46 (31) 772 1761, Fax: +46 (31) 772 3663, E-mail: pers@ce.chalmers.se

Permanent positions and degrees:

- Professor of computer eng. (chair in computer architecture), Chalmers (Sweden), since Nov. 1995. Senior member of the IEEE (1998)
- Assoc/assist. prof. of computer eng., Lund Univ. (Sweden), 1988-1995. Docent in 1993 in computer science and electrical engineering (Lund University).
- Ph. D degree in computer eng., Lund Univ. (Sweden) in 1990, docent 1993.
- Master of Science degree in electrical eng., Lund Univ. (Sweden) in 1981.

Visiting positions:

- Visiting prof., EE dept., Univ. of Southern Calif. (USA), July/Aug. 1993.
- Visiting prof., Computer Systems Lab, Stanford Univ. (USA), June-Dec. 1991.
- Visiting scientist, CS dept., Carnegie-Mellon Univ. (USA), Aug. 1987- May 1988.

Main research interests

- Design principles for shared-memory multiprocessors
- Performance evaluation methodologies
- Back-end compiler transformation techniques
- Timing analysis techniques for real-time systems

Selected professional activities:

- Has authored more than 50 refereed journal and conference publications in the computer architecture, performance analysis, and the compiler areas. Is author of two textbooks.
- Is area editor of the Journal of Parallel and Distributed Computing, has been guest editor of IEEE Computer and Proceedings of the IEEE.
- Was vice chair of the program committee of the 14th IEEE Int. Conf. on Distributed Computing Systems, Poznan, Poland, 1994.
- Has been on the program committee of about twenty computer architecture and parallel processing conferences.

Selected publications (refereed) that are relevant for the project:

1. J. Skeppstedt and P. Stenström: “Simple Compiler Algorithms to Reduce Ownership Overhead in Cache Coherence Protocols,” in *Proc. of 6th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS VI*), pp. 286-296, October 1994.
2. F. Dahlgren, M. Dubois, and P. Stenström: “Sequential Hardware Prefetching in Shared-Memory Multiprocessors,” in *IEEE Trans. on Parallel and Distributed Systems,* Vol. 6 No 7, pp. 733-746, July 1995.
3. P. Stenström, M. Brorsson, F. Dahlgren, H. Grahn, and M. Dubois: “Boosting Performance of Shared-Memory Multiprocessors,” in *IEEE Computer,* pp. 63-70, July 1997.
4. P Stenström, Erik Hagersten, David Lilja, Margaret Martonosi, and Madan Venugopal: “Trends in Shared-Memory Multiprocessing,” in *IEEE Computer* , Vol. 30, No. 12, pp. 44-50, December 1997.
5. T. Lundqvist and P. Stenström: “Integrating Path and Timing Analysis using Instruction-Level Simulation Techniques,” *Proc. of ACM SIGPLAN Workshop on Languages, Compilers, and Tools for Embedded Systems.* June 1998.
6. P. Magnusson, F Dahlgren, H. Grahn, M. Karlsson, F. Larsson, A. Moestedt, J. Nilsson, P Stenström, and B. Werner: “SimICS/Sun4m: A Virtual Workstation. In *Proc. of USENIX98,* June 1998.

A.2 Short CV for Jan Jonsson

Affiliation

Chalmers University of Technology, Dept. of Computer Engineering, S-412 96 Göteborg, Sweden, Phone: +46 (31) 772 5220, Fax: +46 (31) 772 3663, E-mail: janjo@ce.chalmers.se

Permanent positions and degrees:

- Assistant professor of real-time systems, Chalmers (Sweden), since May 1998
- Assistant professor of data communications, Halmstad University (Sweden), 1997
- Ph.D. degree in computer engineering, Chalmers (Sweden) in 1997
- Master of Science degree in computer science and engineering, Chalmers (Sweden) in 1992.

Visiting positions:

- Visiting scholar, Real-Time Computing Laboratory, University of Michigan (USA), August 1996 - December 1996.

Main research interests

- Real-time scheduling strategies for multiprocessor architectures

Selected conference publications: (refereed)

1 J. Jonsson: “GAST: A Flexible and Extensible Tool for Evaluating Multiprocessor Assignment and Scheduling Techniques,” in *Proceedings of the 27th Int’l Conference on Parallel Processing*, August 1998 (to appear).

2 J. Jonsson: “Exploring the Importance of Preprocessing Operations in Real-Time Multiprocessor Scheduling,” in *Proceedings of the 18th IEEE Real-Time Systems Symposium -- Work-in-Progress session*, December 1997, pp. 31--34.

3 J. Jonsson and K. G. Shin, “A Parametrized Branch-and-Bound Strategy for Scheduling Precedence-Constrained Tasks on a Multiprocessor System,” in *Proceedings of the 26th Int’l Conference on Parallel Processing*, August 1997, pp. 158--165.

4 J. Jonsson and K. G. Shin, “Deadline Assignment in Distributed Hard Real-Time Systems with Relaxed Locality Constraints,” in *Proceedings of the 17th IEEE Int’l Conference on Distributed Computing Systems*, May 1997, pp. 432--440.

5 J. Jonsson and J. Vasell, “Real-Time Scheduling for Pipelined Execution of Data Flow Graphs on a Realistic Multiprocessor Architecture,” in *Proceedings of the 21st IEEE Int’l Conference on Acoustics, Speech, and Signal Processing*, May 1996, pp. 3314--3317.

6 J. Jonsson and J. Vasell, “A Comparative Study of Methods for Time-Deterministic Message Delivery in a Multiprocessor Architecture,” in *Proceedings of the 10th IEEE Int’l Parallel Processing Symposium*, April 1996, pp. 392--398.

Appendix B

Letter of intent from Prosolvia.