Techniques for Module-Level Speculative Parallelization on Shared-Memory Multiprocessors

Research Proposal to the PAMP program

Research Plan Sept 1, 2000 - Dec 31, 2001 (Note: only 1.25 years)

Per Stenström

Department of Computer Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
pers@ce.chalmers.se

in collaboration with

Lennart Pettersson
System Processors Department
Ericsson UAB
Box 1505
SE-125 25 Älvsjö, Sweden
lennart.pettersson@uab.ericsson.se

Summary

Multiprocessors have hit the mainstream and cover the whole spectrum of computational needs from small-scale symmetric multiprocessors to scalable distributed shared-memory systems with a few hundred processors. This has made it possible to boost the performance of a number of important applications from the numeric and database domain. Extending the scope of applications that can take advantage of the performance of multiprocessors is however hindered by the fundamental limitation of static (off-line) parallelization methods to only uncovering data dependencies that do not depend on input data.

We consider in this research the prospects of exploiting module- (function-, procedure-, or method-) level data dependence speculation to simplify the process of extracting inherent coarse-grain parallelism out of sequential codes. Given that codes have been developed using good programming practice, our hypothesis is that there is plenty of parallelism to uncover in such codes and a majority of the data dependences will be resolved by the speculation system.

The key question is how to uncover this parallelism and to identify concepts for a speculation system that reduce the number of mis-speculations and yet impose as little overhead on the execution of the threads in parallel. We approach this problem by (1) performing a limit study on available parallelism in commercially used codes in telecommunication systems as well as in benchmark suites such as SPECJava. (2) We then investigate methods for how to extract the parallelism in these codes. In particular, we study the upper-bound on speedup that can be achieved by control-flow and value speculation applied to codes with module-level parallelism.

The expected results of our research is a method that can be integrated in existing software infrastructures for extraction of parallelism in important codes. These results can be transferred to our industrial collaborator and can be used to increase the capacity of telecommunication servers. It will generate important scientific results that contribute to the knowledge about the opportunities and limitations of exploiting module-level parallelism using speculative execution.

1. Problem statement

Multiprocessors have hit the mainstream and cover the whole spectrum of computational needs from small-scale symmetric multiprocessors to scalable distributed shared-memory systems with a few hundred processors. This has made it possible to boost the performance of a number of important application domains from the numeric and database domain. Extending the scope of applications that can take advantage of the performance of multiprocessors is however hindered by the fundamental limitation of static (off-line) parallelization methods to only uncovering data dependencies that do not depend on input data.

Recently, execution models have been proposed that do the data dependence check at run-time. Under these models, the compiler (or user) can be more aggressive in extracting parallelism without conservatively assuming dependences between pointers that are impossible to disambiguate statically. Such execution models are called *thread-level data dependence speculation*. The basic idea is the following. Threads are extracted from the sequential program and assigned a number that designates the original program order. At run-time, threads are executed in parallel in a speculative fashion and any potential data dependency will be detected. If a (true) data-dependence violation is detected, the speculatively executed threads must be aborted, and their effect on the system state must be nullified, before they are re-executed. Recovery actions in addition to name dependences (anti and output) can usually be eliminated by associating a local memory state with each thread and commit the changes to the system memory in program order after the speculatively executed threads have successfully terminated.

Implementations of speculative execution models in the literature either take a predominantly hardware-centric [4, 6, 9, 10] or software-centric [2, 7, 8] approach. In an on-going project at Chalmers together with Ericsson, we are investigating the prospects of taking a purely software-centric approach to design speculation systems[8]. One important advantage of a software-centric approach is that the speculation system can be supported by off-the-shelf multiprocessors. A key challenge, however, is to implement it with an acceptably low overhead. We have demonstrated that it is possible to achieve significant speedups for programs with loop-level parallelism. Indeed, on a 16-way multiprocessor as much as a ten-fold speedup was reported [8]. Yet, many existing programs do not have sufficient inherent loop-level parallelism.

The key issue to be addressed in this research is to extend the applicability of thread-level speculative execution to more general forms of thread-level parallelism which we call module-level parallelism.

2. Main ideas

We aim at exploring to what extent modules taking the form of procedures, methods, and functions can be executed in parallel on a speculation system. Good programming practice tends to encapsulate data dependencies within modules with little or ideally no use of global variables. This points out an interesting opportunity for exploitation of parallelism, namely to exploit parallelism across consecutive modules. Our hypothesis is that since many of the data dependencies are encapsulated, it is feasible to extract parallelism across modules. Assuming that a speculation system catches data dependence violations, it is possible to get significant speedups.

The first question to address is how much parallelism can be exposed in a set of programs developed with a good programming style in mind. We are currently approaching this question by performing a speedup-limit study on a set of Java programs developed using an object- (or method-) oriented programming style.The base for this limit study is an ideal speculation system in which all name dependencies and some data dependencies are eliminated through renaming and forwarding of values between threads, respectively. A complete system simulation platform based on SimICS [5] is used to produce memory reference traces that are processed by the speculation system. The applications being traced are picked from the SPECJava suite. The results of this first project phase will provide (1) an upper-bound on the available parallelism and (2) the fundamental reasons behind the absolute limits on speedup.

Based on the insights from the limit study, we aim at studying how concepts known from the micro-architecture literature such as control-flow dependence and value speculation can be exploited to approach the upper-bound on the speedup offered by coarse-grain, i.e., module-level data dependence speculation

systems. While issues regarding how to implement these strategies are interesting, the fundamental question we first want to address is how much speedup these concepts can provide given ideal conditions.

3. Expected results and impact

The expected results of our research is a method that eventually can be integrated in existing software infrastructures for extraction of parallelism in codes with module-level parallelism. In addition, the experimental investigations will provide valuable insights into what limits further exploitation of coarse-grain parallelism using thread-level speculative execution.

Apart from leading to state-of-the-art reports, these results can be transferred to our industrial collaborator and can be used to increase the capacity of telecommunication servers.

4. Project plan

The proposed project will be carried out by Fredrik Warg who got enrolled in the Ph. D. education program on July 1, 1999. The project is currently funded by NUTEK under the program Complex Technical Systems. Because NUTEK has decided to cancel this program as of August 31 this year, we are asking PAMP to support the project until 2001-12-31 when Fredrik Warg is expected to earn his Licentiate degree.

Milestones and deliverables 000901-011231 (**Note:** only 1 year and 3 months at 80% activity level)

- **Task 1 (Limit study):** The purpose is to find an upper-bound on the available parallelism on a suite of programs using an object-oriented style. The program suite to be tested is SPECJava and possibly other programs. We have developed a tracing tool that can also identify the reasons for the limited speedup.
- **Task 2 (Concept development and evaluation).** Two concepts known from the micro-architecture literature--control-flow and value speculation--will be adapted to be applicable to module-level parallelism. The implementation issues will be identified and a performance evaluation using the tracing tool in Task 1 will be conducted to predict the upper-bound of the speedup that can be obtained using these concepts.
- **Task 3 (Generalization)** The major findings from the experimental phase in Task 1 and 2 will be generalized in this project phase.

The time plan for the first 1.25 years is the following:

- **Task 1:** 3 months (one state-of-the-art report)
- **Task 2:** 9 months (two state-of-the-art reports
- **Task 3:** 3 months (Licentiate thesis)

The limit study is expected to be concluded in September this year. A state-of-the-art report to be submitted to a conference is expected to be finalized by the end of the year. Task 2, that will focus on two concepts (return-value speculation and control-flow speculation) is expected to lead to two state-of-the-art reports to be submitted to conferences. In total, the Licentiate thesis will summarize these three reports.

5. Preliminary budget

Fredrik Warg’s activity level is 80%. Thus, the funding requested is 480 KSEK x 1.25= 600 KSEK.

6. Related research

There is a considerable body of literature on thread-level data dependence speculation systems [2, 4, 6, 8, 9, 10] . However, all of these studies consider loop-level parallelism and in particular intra-procedure parallelism. This form of parallelism is quite restrictive and may not significantly widen the scope of sequential applications that can take advantage of the parallelism offered by multiprocessor systems.

Recently, Chen and Olukotun [1] performed a study in which the goal is to see how much parallelism is inherent in single-threaded Java programs by focusing on method-level parallelism. Interestingly, they found that quite significant speedups can be obtained. However, data and control dependences between methods were the primary factors that limited the speedup.

We are concluding a similar limit study based on an extended set of applications. This limit study will give important insights into how the concept of thread-level speculation can be extended beyond loops. Our long-term goal is to see to what extent data and control dependences can be overcome by incorporating support for value speculation [3] and control-flow speculation in a software-centric speculation system. Control-flow dependence speculation has been studied to some extent in the context of thread-level speculation systems [2] but we want to consider more general forms than only simple loop level control-flow speculation.

7. Industrial relevance

The capacity provided by single processor systems is simply not enough to power the information processing infrastructure that today supports the Internet. Multiprocessors are already the de-facto standard in many transaction-oriented systems such as database systems to provide scalable performance growth as processing resources are added.

The servers that power the telecom infrastructure face the same capacity problem which makes multiprocessors the enabling technology as a scalable platform for telecom applications.

Owing to the enormous investments in software developments behind e.g. the AXE system out-rules the option of making even quite insignificant changes to the existing software. The approach of using a speculation system to extract the parallelism out of existing software, which is the goal of the proposed project, is an interesting means to use off-the-shelf multiprocessors to increase the capacity of the telecom servers. The results from the project is expected to give valuable insights regarding *how* to best exploit the inherent parallelism in existing software.

8. Relation to PAMP and other SSF programs

The goal of PAMP is to develop methods, tools, and generic building blocks for cost-effective design of multiprocessor-based applications. This project obviously fits well with the overall goal of PAMP by focusing on multiprocessor technology as a commodity building block. More precisely, the results will take the form of methods to simplify the extraction of parallelism for cost-effective design of multiprocessor-based applications by contributing with concrete principles of how to approach the theoretical upper-bound on speedup inherent in the applications. The project targets a very important application in the telecom domain. These applications show soft real-time requirements which are met by increasing the capacity of the servers. The method studied in this project has this aim.

Note also that this project has synergies with other PAMP projects such as the project at SICS in terms of using SimICS as a platform for experimentation and the project at Uppsala regarding design principles of multiprocessors.

Regarding other SSF-funded programs of relevance to the proposed project, NGSSC (National Graduate School of Scientific Computing) focuses on high-performance computing. This project is related to the goals of that program in that it can help scientific computational programs to exploit the parallelism in available multiprocessor-based platforms.

9. 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 TFR-funded project on multiprocessor system architectures for transaction-oriented systems. (1) and (2) will be concluded this year. (3) a TFR-funded project on worst-case timing analysis techniques for single-processor systems, (4) an ARTES/PAMP funded project on design strategies for multimedia applications on multiprocessors. Finally, (5) the project that is most related to the proposed research is a TFR-funded project that aims at investigating hardware/software trade-offs involved in the design of data dependence speculation systems. The distinguishing character between the two projects is

that the former has implementation focus whereas the proposed project has an application focus. Clearly, the synergies between these two projects are extremely fruitful.

The group has active collaboration with research groups at University of Southern California (Michel Dubois) and 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.

References

[1] Mike Chen and Kunle Olukotun: Exploiting Method-Level Parallelism in Single-Threaded Java Programs. in *Proc. of Parallel Architectures and Compilation Techniques,* October 1998.

[2] Iffat Kazi and David Lilja, ‘‘Coarse-Grained Speculative Execution in Shared-Memory Multiprocessors’’, in *Proc. of 1998 Int. Conf. on Supercomputing*, Aug. 1998.

[3] M. Lipasti, C. Wilkerson, J. P. Shen: “Value Locality and Load Value Prediction”, in *Proc. of Seventh Int. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS VII),*pp. 138-149, Oct 1996.

[4] Venkata Krishnan and Josep Torrellas. “Hardware and Software Support for Speculative Execution of Binaries on a Chip-Multiprocessor,” in *Proc. of 1998 Int. Conf. on Supercomputing*, Aug 1998.

[5] P. S. Magnusson, F. Dahlgren, H. Grahn, M. Karlsson, F. Larsson, F. Lundholm, A. Moestedt, J. Nilsson, P. Stenström, B. Werner, ‘‘SimICS/Sun4m: A Virtual Workstation’’, in *Proc. of Usenix Annual Technical Conference*, June, 1998.

[6] Jeffrey Oplinger, David Heine, Shih-Wei Liao, Basem A. Nayfeh, Monica Lam, and Kunle Olukotun. “*Software and Hardware for Exploiting Speculative Parallelism with a Multiprocessor,*” Tech. Report Computer Systems Laboratory, Stanford University, CSL-TR-97-715, Feb. 1997.

[7] Lawrence Rauchwerger and David Padua. ‘‘The LRPD Test: Speculative Run-Time Parallelization of Loops with Privatization and Reduction Parallelization,’’ in *IEEE Trans. on Parallel and Distributed Systems*, Vol. 10, No 2., February 1999, Vol. 10, No 2., pp. 160-199, Feb. 1999.

[8] Peter Rundberg and Per Stenström “A Low-Overhead Software Approach to Thread-Level Data Dependence Speculation on Multiprocessors” Tech. Report, Chalmers University of Technology, Submitted for Publication. April 2000.

[9] Gregory Steffan and Todd Mowry. “The Potential for Using Thread-Level Data Speculation to Facilitate Automatic Parallelization,” in *Proc. of Fourth Int. Symp. on High-Performance Computer Architecture*, pp. 2-13, Feb. 1998.

[10] Ye Zhang, Lawrence Rauchwerger, and Josep Torrellas. “Hardware for Speculative Run-Time Parallelization in Distributed Shared-Memory Multiprocessors,” in *Proc. of Fourth Int. Symp. on High-Performance Computer Architecture*, pp. 162-173, Feb. 1998.

Appendix 1: Curriculum Vitae

Per Stenström
Professor, Ph.D., Docent, Senior Member of the IEEE

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:

- Prof. of computer eng. (chair in computer architecture), Chalmers (Sweden), since Nov. 1995. Since April 1999 Vice-dean of the School of Electrical and Computer Engineering.
- Assoc/assist. prof. of computer eng., Lund Univ. (Sweden), 1988-1995. Docent in 1993.
- 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 high-performance computer architectures
- Performance evaluation methodologies and tools
- Parallelization techniques for multiprocessors
- Energy-efficient computer architectures

Selected professional activities:

- Has authored more than 80 journal and conference publications in the computer architecture, performance analysis, and the compiler areas. Is author of two textbooks.
- Is associate editor IEEE Trans. on Computers of the Journal of Parallel and Distributed Computing, 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 more than twenty computer architecture and parallel processing conferences.
- General Chair of ISCA-2001 to be held in Gothenburg July 2-4, 2001.

Journal publications: 1995--present (refereed)

[1] H. Grahn, P. Stenström, and M. Dubois: “Implementation and Evaluation of Update-Based Cache Protocols Under Relaxed Memory Consistency Models,” in *Future Generation Computer Systems*, Vol. 11, No. 3, pp. 247-271, June 1995.

[2] F. Dahlgren and P. Stenström: “Using Write Caches to Improve Performance of Cache Coherence Protocols in Shared-Memory Multiprocessors,” in *Journal of Parallel and Distributed Computing*, Vol 26. No 2, pp. 193-210, April 1995.

[3] 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.

[4] M. Dubois, J. Skeppstedt, and P. Stenström: “Essential Misses and Memory Traffic in Coherence Protocols,” in *Journal of Parallel and Distributed Computing*, Vol. 29, No 2, pp. 108-125, 1995.

[5] F. Dahlgren and P. Stenström “Evaluation of Stride and Sequential Hardware-based Prefetching in Shared-Memory Multiprocessors,” in *Trans. on Parallel and Distributed Systems*, Vol. 7, No. 4, pp. 385-398, April 1996.

[6] M. Brorsson and P. Stenström: “Characterising and Modelling Shared-Memory Accesses in Multiprocessor Programs,” in *Parallel Computing*, No 22, pp. 869-893, 1996.

[7] P. Stenström, M. Balldin, and J. Skeppstedt: “The Design of a Non-Blocking Load Processor Architecture,” in *Microprocessors and Microsystems, No 20, pp. 111-123*, 1996.

[8] J. Skeppstedt and P. Stenström: “Using Dataflow Analysis to Reduce Overhead in Cache Coherence Protocols,” in *Transactions on Programming Languages and Systems*, Vol 18, No 6, pp. 659-682, November 1996.

[9] H. Grahn and P. Stenström: “Evaluation of an Adaptive Update-Based Cache Protocol,” in *Journal of Parallel and Distributed Computing*, 39(2):168-180, December 1996.

[10] 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.

[11] M. Karlsson and P. Stenström: “Effectiveness of Dynamic Prefetching in Multiple-Writer Distributed Virtual Shared Memory Systems,” in *Journal of Parallel and Distributed Computing*, Vol. 43, No. 2, pp. 79-93, 1997.

[12] F. Dahlgren, M. Björkman and P. Stenström: “Reducing the Read Miss Penalty for Flat COMA Protocols, in *the Computer Journal*, Vol. 40, No. 4, pp. 208-219, 1997.

[13] 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.

[14] F. Dahlgren, J. Skeppstedt, and P. Stenström: “An Evaluation of Hardware-Based and Compiler-Controlled Snooping Cache Protocol Extensions,” in *Journal of Future Generation Computer Systems*, No. 13, pp. 469-487, 1998.

[15] F. Dahlgren, M. Dubois, and P. Stenström: “Performance Evaluation and Cost Analysis of Cache Protocol Extensions for Shared-Memory Multiprocessors,” in *IEEE Transactions on Computers*, Vol. 47, No 10, pp. 1041-1055, Oct. 1998.

[16] J. Skeppstedt, F. Dahlgren, and P. Stenström: “Evaluation of Compiler-Controlled Updating to Reduce Coherence-Miss Penalties in Shared-Memory Multiprocessors,” in *Journal of Parallel and Distributed Computing*, Vol. 56, No 2, pp. 122-153, 1999.

[17] H. Grahn and P. Stenström: “Comparative Evaluation of Latency-Tolerating and Reducing Techniques for Hardware-Only and Software-Only Directory Protocols”, *Journal of Parallel and Distributed Computing*, to appear 1999.

[18] T. Lundqvist and P. Stenström: “An Integrated Path and Timing Analysis Method Based on Cycle-Level Symbolic Execution,” Accepted for publication in *Journal of Real-Time Systems*, to appear Nov 1999.

[19] V. Milutinovic and P. Stenström “Opportunities and Challenges for Distributed Shared-Memory Multiprocessors. Guest Editors’ Introduction, *Proceedings of the IEEE*. Vol 87 No 3, pp 399-404, March 1999.

Conference Papers (refereed) 1995-present

[20] M. Björkman, F. Dahlgren, and P. Stenström: “Using Hints to Reduce Read Miss Penalties for Flat COMA Protocols, in *Proc. of 28th Hawaii International Conference on System Sciences*, pp. 242-251, January 1995.

[21] F. Dahlgren and P. Stenström “Effectiveness of Stride and Sequential Hardware-based Prefetching in Shared-Memory Multiprocessors, in *Proc. of First International Conference on High Performance Computer Architecture (HPCA-1)*, pp. 68-77, January 1995.

[22] H. Grahn and P. Stenström: “Efficient Strategies for Software-Only Directory Protocols in Shared-Memory Multiprocessors,” in *Proc. of 22nd Annual International Symposium on Computer Architecture*, pp. 38-47, June 1995.

[23] J. Skeppstedt and P. Stenström: “A Compiler Algorithm that Reduces Read Latency in Ownership-Based Cache Coherence Protocols,” in *Proc. of Parallel Architectures and Compilation Techniques*, pp. 69-78, July 1995.

[24] F. Dahlgren, J. Skeppstedt, and P. Stenström: “Effectiveness of Hardware-Based and Compiler-Controlled Snooping Protocol Extensions,” in *Proc. of the International Conference on High Performance Computing, pages 87-92*, December 1995.

[25] M. Karlsson and P. Stenström: “Performance Evaluation of a Cluster-Based Multiprocessor Built from ATM-Switches and Bus-Based Multiprocessor Servers,” in *Proc. of Second International Conference on High Performance Computer Architecture*, pages 4-13, Jan. 1996.

[26] H. Grahn and P. Stenström: “Relative Performance of Software-Only and Hardware-Only Directory Protocols Under Latency Tolerating and Reducing Techniques,” in *Proceedings of the 11th International Parallel Processing Symposium*, pages 500-506, April 1997.

[27] J. Nilsson, F. Dahlgren, M. Karlsson, P. Magnusson, P. Stenström: “Computer System Evaluation with Commercial Workloads” in *Proc. of IASTED Conference on Modeling and Simulation*. pp. 293-297, May 1998.

[28] 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,*pp. 119-130, June 1998.

[29] 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.

[30] P. Stenström and F. Dahlgren: “A Holistic Approach to Computer System Design Education Based on System Simulation Techniques,” in *Proc. of 1998 IEEE Workshop on Computer Architecture Education*, June 1998.

[31] T. Lundqvist and P. Stenström: “Timing Anomalies in Dynamically Scheduled Processors,” in *Proc. of 1999 IEEE Real-Time System Symposium (RTSS’99)*, pp. 12-21 Dec. 1999.

[32] T. Lundqvist and P. Stenstrom. “A Method to Improve the Estimated Worst-Case Performance of Data Caching”. in *Proc. of 6th International Conference on Real-Time Computing Systems and Applications (RTCSA' 99)*,pp. 255-262, Dec 1999.

[33] M. Karlsson, F. Dahlgren, and P. Stenström: “Prefetching Techniques for Irregular Accesses to Linked Data Structures,” *6th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA-6)*, 2000.

[34] M. Karlsson, F. Dahlgren, and P. Stenström: “An Analytical Model for Working-Set Sizes in Decision Support Systems,” To appear in *SIGMETRICS*, 2000.

[35] A. Saulsbury, F. Dahlgren, and P. Stenström: “Recency-Based TLB Preloading” To appear in *27th IEEE Int. Symp. on Computer Architecture (ISCA-27)*, 2000.

Appendix 2: Letter of Intent

Ericsson UAB is currently collaborating with Per Stenström's group at Chalmers in the NUTEK funded project "Exploitation of Multiprocessor Technology in Transaction-Oriented Applications" within the program Complex Technical Systems. It is unfortunate that NUTEK has terminated this program. The collaboration with Chalmers in the context of this project has resulted in many important insights into the opportunities to use multiprocessors to increase the processing capacities of the AXE-system. The need for processing capacity in telecommunication systems is increasing at a higher rate than ever due to new applications and increased usage of telecommunication services. The use of multiprocessor technology and finding new ways of increasing processing capacity is therefore a research area of high industrial importance.

With this letter, Ericsson is supporting and verifying that the proposed research is extremely relevant to important issues we are dealing with in on-going projects.

Älvsjö 00-05-19

Lennart Pettersson
Project Manager, Ericsson UAB

Phone: 08-7272710
Fax: 08-7273812
Email: lennart.pettersson@uab.ericsson.se