*Parallel Computer Algebra on a Shared Memory Multiprocessor (Extended Abstract)**

Wolfgang Schreiner
Wolfgang.Schreiner@risc.uni-linz.ac.at
Research Institute for Symbolic Computation (RISC-Linz)
Johannes Kepler University, Linz, Austria

February 3, 1994

1 Parallel Computer Algebra

Computer algebra is that branch of computer science that aims to provide exact solutions of scientific problems. Research results of this area are e.g. algorithms for symbolic integration, polynomial factorization or the exact solution of algebraic equations and inequalities [1]. All these algorithms operate with arbitrary precision arithmetic; they are therefore more expensive than the corresponding numerical methods. However, they always provide correct solutions and are not influenced by the stability properties of the given problem.

The parallel computation group at RISC-Linz has started a project that pursues the development of PACLIB, a new system for parallel algebraic computation on shared memory multiprocessors [4, 5]. Our goal has been to develop a professional tool for the simple and efficient implementation of parallel algorithms in computer algebra and related areas. Parts of this talk are taken from [6].

2 Shared Memory Multiprocessing

The objects of algebraic computation are symbolic expressions (arbitrary precision integers and rationals, polynomials, sets of polynomials ...) that are dynamically allocated on a heap. These objects tend to grow very much in the course of the computation, which also increases the runtime of the operating tasks. The dynamic behavior of algebraic algorithms concerning time and memory is therefore very complex and often unpredictable even for known input values.

By the dynamic nature of the basic objects, algebraic applications are at least as strongly bound by memory as by time. A garbage collector is required that reclaims no more used objects. In a parallel environment, the issue of garbage collection is even more important, since p processors in general allocate heap space at a rate that is p times higher than a single processor (provided that the total memory size is the same). In order to preserve the ratio of garbage collection time to the actual computation time, the garbage collector itself must be efficiently parallelized.

The high memory consumption also suggests that a distributed memory model of computation may be not appropriate. In such a model, all arguments of a remote task must be explicitly

*Supported by the Austrian Science Foundation (FWF) grant S5302-PHY "Parallel Symbolic Computation".

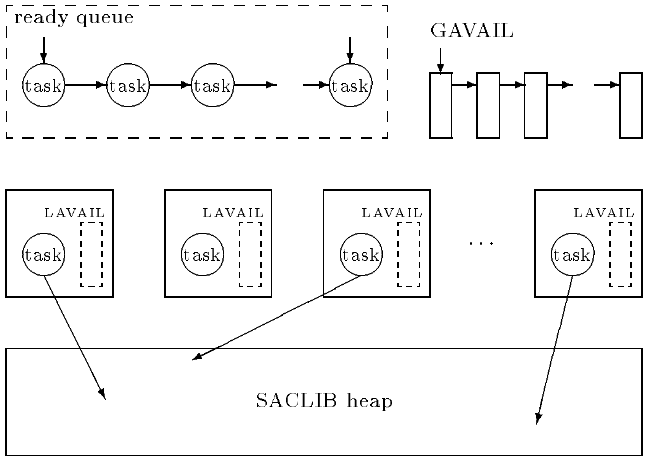

Figure 1: The System Structure

copied into the heap of the executing processor and the results must be copied into the heap of any requesting processor. Eventually, many duplicate structures will exist in the system and even within one heap (since initially shared substructures will be duplicated). The size of solvable problems is therefore drastically smaller than on a single processor machine with the same amount of memory. A global address space that allows the sharing of objects between processors is therefore preferred[1].

3 The PACLIB System

PACLIB is a combination and extension of two free software packages:

- SACLIB [2] is a library of C functions that is based on a list processing kernel with automatic garbage collection and supports all fundamental objects and methods of computer algebra. Several algebraic applications have been implemented in this library such as the Partial Cylindric Algebraic Decomposition Method and the Gröbner Bases Algorithm.
- The μSystem [3] is a library that supports light-weight concurrency on shared memory multiprocessors and UNIX workstations. The μSystem kernel distributes *tasks* (light-weight processes) among *virtual processors* which are UNIX processes scheduled by the operating system. On a multi-processor, virtual processors are therefore executed by multiple hardware processors truly in parallel.

We applied and considerably extended the mechanisms of the μSystem to develop a suitable parallel programming interface for SACLIB. The PACLIB kernel consists of the following main components (depicted in Figure 1):

1. **Ready Queue:** This queue contains all tasks that are *active* but not *running*. The μSystem scheduler selects tasks for execution from the head of this queue; tasks that are preemptively descheduled are inserted at the tail.
2. **Shared Heap:** The SACLIB heap is an array of fixed-size list cells and serves as the only communication medium between tasks. There is no communication bottleneck since all PACLIB tasks may read all heap cells simultaneously.

[1] Currently, such a global address space is only efficiently supported by computers with few ($\leq$ 32) processors. However, new *virtual shared memory* architectures promise to fulfill this demand also for massively parallel machines.

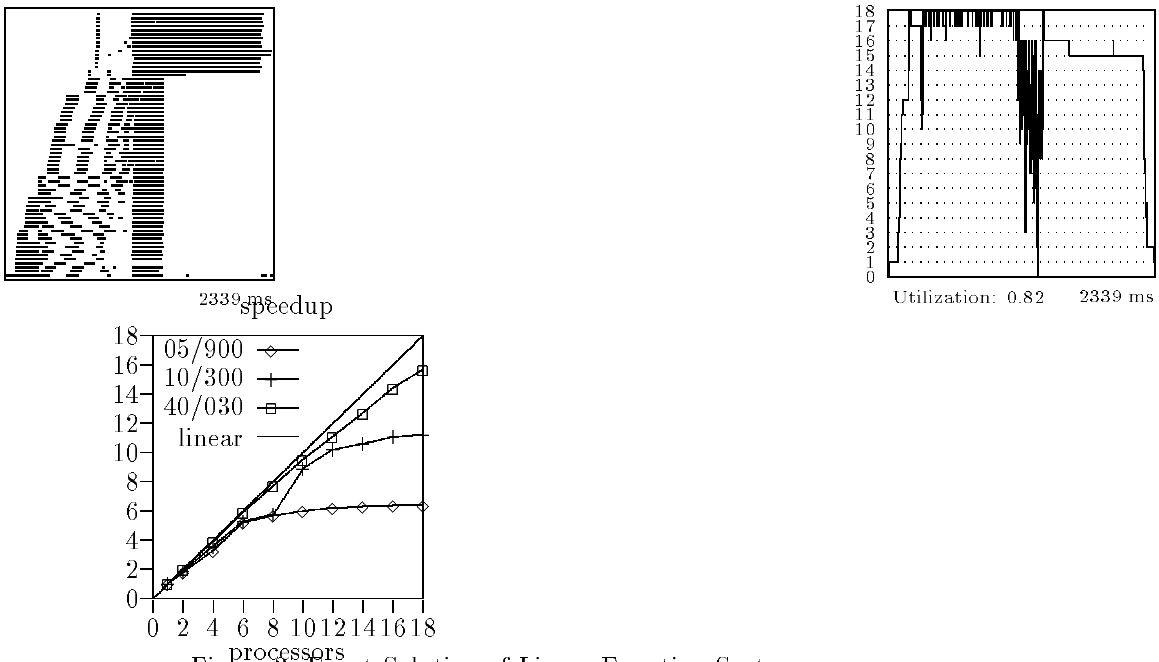

Figure 2: Exact Solution of Linear Equation Systems

3. **Virtual Processors:** Each virtual processor is a UNIX process that exclusively owns a local list `LAVAIL` of free heap cells. Each task executed on this processor allocates new heap cells from this list.

4. **Global Available List:** `GAVAIL` denotes a global list of free lists (of fixed but arbitrary length) in which the unused heap cells are linked together. Processors whose local lists run out of free cells receive their new `LAVAIL` from `GAVAIL`.

The PACLIB kernel has been used to implement a variety of parallel computer algebra algorithms. Figure 2 shows some vizualization and benchmark results of a parallel algorithm for the exact solution of linear equation systems with arbitrary precision integer coefficients.

References

[1] B. Buchberger, G. E. Collins, R. Loos, and R. Albrecht, editors. *Computer Algebra — Symbolic and Algebraic Computation.* Springer, Vienna, New York, 1982.

[2] Bruno Buchberger, George Collins, Mark Encarnation, Hoon Hong, Jeremy Johnson, Werner Krandick, Rüdiger Loos, Ana Mandache, Andreas Neubacher, and Herbert Vielhaber. A SACLIB Primer. Technical Report 92-34, RISC-Linz, Johannes Kepler University, Linz, Austria, 1992.

[3] Peter A. Buhr and Richard A. Stroobosscher. The μSystem: Providing Light-weight Concurrency on Shared-Memory Multiprocessor Computers Running UNIX. *Software — Practice and Experience*, 20(9):929–964, September 1990.

[4] Wolfgang Schreiner and Hoon Hong. A New Library for Parallel Algebraic Computation. In Richard F. Sincovec et al., editors, *Sixth SIAM Conference on Parallel Processing for Scientific Computing,* volume II, pages 776–783, Norfolk, Virginia, March 22–24, 1993.

[5] Wolfgang Schreiner and Hoon Hong. PACLIB — A System for Parallel Algebraic Computation on Shared Memory Computers. In Hussein M. Alnuweiri, editor, *Parallel Systems Fair*

at the Seventh International Parallel Processing Symposium, pages 56–61, Newport Beach, CA, April 14, 1993. IPPS '93.

[6] Wolfgang Schreiner and Hoon Hong. The Design of the PACLIB Kernel for Parallel Algebraic Computation. In Jens Volkert, editor, *Parallel Computation*, 2nd International ACPC Conference, Gmunden, Austria, October 4–6, 1993. Volume 734 of Lecture Notes in Computer Science, Springer, Berlin.