Project Descrition: Achieving Optimum Performance for Dense Linear Algebra Computations on Parallel Computers:

Extending ScaLAPACK for Distributed Panels

P.E. Strazdins, Department of Computer Science,
Australian National University

January 4, 1996

Background and Significance

Dense linear algebra computations form one of the major uses of supercomputing. These computations include matrix factorization, the solution of linear systems, eigenvalue computations and singular value decomposition. One of the increasingly promising architectural platforms for supercomputing is distributed memory parallel computers made up of scalar processors. Recent examples of such computers achieving very high performance include the Cray T3D, the IBM SP-2 and the Fujitsu AP+. It is important then to establish and refine techniques for obtaining optimal performance of dense linear algebra computations on such platforms.

Dense linear algebra computations require the technique of 'block-partitioned algorithms' for their efficient implementation on memory-hierarchy processors. For scalar-based distributed memory parallel processors, the register, cache and off-processor memory levels of the memory hierarchy all affect the optimal block-partition size for such algorithms. Most existing studies on dense linear algebra computations have assumed that the block-partition size or panel width for the algorithm, ω, to be the same as the matrix distribution block size, r, where a square $(r \times r)$ block-cyclic matrix distribution is being employed on a $P \times Q$ rectangular processor configuration. Here the choice of $\omega = r$ is essentially determined by the off-processor memory level of the memory hierarchy.

This means that the panel formation part of the computation is not fully parallelized, as the panel is fully contained in a row or column of processors, and hence only that subset of processors can participate in panel formation. On this assumption, a parallel version of the extensive dense linear algebra library LAPACK (Linear Algebra PACKage), called ScaLAPACK (Scalable LAPACK) is currently available under release 1.1. Key references for work in this area include Dongarra (1994) and Choi (1994); these are most readily accessible from the LAPACK Working Notes (numbers 80, 92, and 95) available from `http://www.netlib.org/scalapack`.

This assumption has been re-examined in the context of matrix factorization computations on scalar-based distributed memory parallel processors, such as the Fujitsu AP1000 [1]. There considerations of the register and cache levels of the memory hierarchy require a large panel width ω. 'Distributed panels' versions of these computations, where $\omega > r$ (and typically $\omega > rP$), allow full parallelization of the panel formation stage. It has been shown that on the AP1000, the 'distributed panels' technique yields significantly better performance (15%–20% overall improvement in speed) on these computations (which can still be improved by further optimizations). A preliminary theoretical performance analysis predicts similar benefits on the Cray T3D, the IBM SP-2 and the Fujitsu AP+. It should be further noted that near-future successors to these parallel processors are likely to have secondary caches, which will enhance the benefits of the distributed panels technique.

As well as performance improvements, the distributed panels technique offers a simpler interface and ease of use in a parallel dense linear algebra library. For the traditional techniques, the optimal block size r is an empirically determined function of the type of computation, the architecture and, least convenient of all, the matrix size. However, for the distributed panels technique, it is found that $r = 1$ is optimal (or sufficiently near to optimal) [1].

The foundation for the 'distributed panels' matrix factorization algorithms has been the production of a parallel version of the Basic Linear Algebra Subroutine (BLAS) library. This parallel version (developed for the AP1000) is called the DBLAS (Distributed BLAS) [3, 6]. Its current implementation has the deficiency that in the context of these dense linear algebra computations, applying the 'distributed panels' technique leads to redundant communication, typically doubling the communication load. This occurs with distributed panels because matrix rows or columns first have to be communicated during panel formation, and then later when the whole panel is communicated. It should be noted that while there are other versions of parallel BLAS available (most notably the PBLAS, the foundation of the ScaLAPACK library), currently only the DBLAS has sufficient functionality to support the distributed panels technique.

It is felt that the distributed panels technique has been overlooked by other researchers in the field, mainly due to the less favourable architectural characteristics of the vector-based parallel computers (dominant in the early 90's, but whose dominance is now being challenged), and due to the considerable software engineering difficulties in developing a version of the parallel BLAS with sufficient functionality to support it.

Key References

J. Dongarra, R. van de Geigin, and D. Walker. Scalability Issues in the Design of a Library for Dense Linear Algebra. *Journal of Parallel and Distributed Computing*, 22(3):523–37, 1994.

J. Choi, J.J. Dongarra and D.W. Walker. PB-BLAS: A Set of Parallel Block Basic Linear Algebra Subprograms. In *Proceedings of the Scalable High Performance Computing Conference*, Knoxville, Tennessee, May 1994. Also submitted to *Concurrency: Practice and Experience*.

Aims

This Project can be partitioned into two parts. Part I has the following aims.

1. An investigation of the performance (speed) benefits of the 'distributed panels' technique for LU, Cholesky and QR factorization computations on the Fujitsu AP1000/AP+.
2. An investigation of the performance benefits of the 'distributed panels' technique for Hessenberg, bi-diagonal and tri-diagonal reduction algorithms (the dominant computational components of QR-based eigenvalue computation and singular value decomposition) on the Fujitsu AP1000/AP+.
3. The development of a theoretical performance model to compare 'distributed panels' versions of these computations versus the standard versions. From this, suitable scalar-based (and possibly even vector-based) parallel computer platforms can be identified.
4. The production of a portable version of the DBLAS.
5. The investigation of how a parallel BLAS library (ie. the DBLAS) can be implemented to avoid producing redundant communications without compromising software engineering considerations.

While Part I forms an essential basis, the bulk of the work of the Project lies in Part II, which has the following aims.

1. The production a PBLAS interface to the DBLAS, to enable the DBLAS (which can support distributed panels computations) to be called from the ScaLAPACK codes.
2. Using the experience from Aims 1 and 2, the extension of ScaLAPACK 1.1 (or the latest release available) to accommodate 'distributed panels' (single and double precision datatypes only).
3. The porting and testing of the extended ScaLAPACK codes to other parallel computer platforms. This work can then prove that the 'distributed panels' concept is widely applicable.
4. The release the improved version of parallel LAPACK for researchers in computational science. These improvements include both performance gains and a simplified user interface.

In the context of parallel computing, the Project will challenge several widely-held assumptions amongst parallel computing researchers, including over-emphasizing the importance on minimizing communication startup and under-estimating the importance of the interaction of a parallel algorithm with the cache levels of the parallel memory hierarchy. The techniques may find some application in sparse numerical computations and in regular non-numeric computations on parallel computers.

As well as research issues pertaining to the interaction between a parallel algorithm and a parallel computer architecture, the Project also addresses substantial software engineering issues (especially in Aims I.5 and II.2).

The project will offer a software package that is not only improved in performance, but has a simpler user interface as well. It must be remembered that *computational scientists* are not *computer scientists*, that is, they are interested in solving their problem quickly and easily, without having to worry about which is best of a variety of matrix distributions over a two-dimensional mesh of processors for each particular matrix computation. For the distributed panels technique, a single, simple matrix distribution is optimal in virtually all situations. Not only will this simplify the user interface, it will also obviate the need for application programs using the package to have to re-distribute a matrix between two computations.

The Project has scope for extension as ScaLAPACK itself is expected to increase its functionality in the near future. One of the most relevant of these is parallel banded matrix solvers, where a small matrix block size r is required to get reasonable load balance due to the relative narrowness of the bands. For moderate band widths, the distributed panels technique is strongly applicable, and will be required to ensure good processor computation performance. Indeed, this Project has scope for direct contribution to the ScaLAPACK project.

Background Work for the Project

Over the years 1991–1995, the DBLAS library for the AP1000 has been designed, implemented and developed. While not part of this Project per se, it forms its foundation. In May 1993, the DBLAS was first released to the Fujitsu Parallel Computing Research Facility, with an enhanced version being released in January 1994. However, both these versions supported only the scattered (or 1×1 block-cyclic) matrix distribution.

The generalization of the DBLAS library to support the general $r \times s$ block-cyclic matrix distribution began in July 1994. At the same time, Aim I.1 was begun, with sufficient progress being made to submit a first publication [1] by December 1994. This publication covered only LU and Cholesky factorization, but also contains a preliminary performance analysis (to contribute to Aim I.3). During the first half of 1995, the DBLAS has been optimized to support QR computations and parallel QR factorization algorithms have been developed.

Personnel and Project Plan

The Chief Investigator for this Project is Dr Peter Strazdins. Part I Aims 1–3 are near-complete at the time of writing. Part I Aims 4–5 and Part II Aim I will be undertaken by the Chief Investigator in early 1996.

A part-time Research Assistant is being sought to assist the Chief Investigator for Part II, Aims 2–4 over 1996. It should be noted that most of the ScaLAPACK routines will require only minor modifications to achieve this extension.

Publications

Recent publications of the Chief Investigator are given in the following references. The starred items are strongly relevant to this project.

References

[1] P.E. Strazdins. Matrix Factorization using Distributed Panels on the Fujitsu AP1000.* In IEEE First International Conference on Algorithms And Architectures for Parallel Processing (ICA3PP-95), Brisbane, April 1995.

[2] A. Czezowski and P.E. Strazdins. Optimisations for the memory hierarchy of a Singular Value Decomposition Algorithm implemented on the MIMD Architecture. In *International Conference on High-Performance Computing and Networking*, Munich, April 1994.

[3] R.P. Brent and P.E. Strazdins. Implementation of BLAS Level 3 and LINPACK Benchmark on the AP1000.* *Fujitsu Scientific and Technical Journal*, 29(1), 1993.

[4] A. Czezowski and P.E. Strazdins. Ways of enhancing performance of a Singular Value Decomposition Algorithm on the AP 1000 Array Multicomputer. In *Fifth Australian Transputer and OCCAM User Group Conference*, Melbourne, November 1992.

[5] H. Schroder and P. Strazdins. Program Compression on the Instruction Systolic Array. *Parallel Computing*, 17, 1991.

[6] P.E. Strazdins, *Prototyping Parallel LAPACK using Block-Cyclic Distributed BLAS.** In *Third Parallel Computing Workshop for the Fujitsu PCRF*, Kawasaki, November 1994.