DOE/ER/25215--T1

Final Report on
Numerical Methods for
Some Structured Matrix Algebra Problems
DOE Contract DE-FG03-94ER25215

Elizabeth R. Jessup
Department of Computer Science
University of Colorado
Boulder, CO 80309-0430

This proposal concerned the design, analysis, and implementation of serial and parallel algorithms for certain structured matrix algebra problems. It emphasized large order problems and so focused on methods that can be implemented efficiently on distributed-memory MIMD multiprocessors. Such machines supply the computing power and extensive memory demanded by the large order problems. We proposed to examine three classes of matrix algebra problems: the symmetric and nonsymmetric eigenvalue problems (especially the tridiagonal cases) and the solution of linear systems with specially structured coefficient matrices. As all of these are of practical interest, a major goal of this work was to translate our research in linear algebra into useful tools for use by the computational scientists interested in these and related applications. Thus, in addition to software specific to the linear algebra problems, we proposed to produce a programming paradigm and library to aid in the design and implementation of programs for distributed-memory MIMD computers. We now report on our progress on each of the problems and on the programming tools.

In [8], we present the results of our study of a parallel method for solving the symmetric eigenproblem. This method is to reduce the symmetric matrix to tridiagonal form via Householder transformations, to compute the eigensystem of the tridiagonal matrix, and to backtransform the result. Bruce Hendrickson of Sandia National Laboratory, Chris Smith of the University of Wisconsin, and I have developed an efficient code for Householder reduction and backtransformation on the Intel Paragon. The algorithms are based on a square torus–wrap mapping of matrix elements to processors to reduce communication, and the code uses level 3 BLAS routines for efficient numerical kernels. The experimental results presented in [8] demonstrate that the torus wrap mapping leads to substantially better performance than the block mapping used in ScaLAPACK [3].

With Silvia Crivelli (formerly a grad student at CU and now a postdoc at CU), we have developed PMESC–a medium- to coarse-grained environment for managing dynamic computations on distributed-memory MIMD computers. The goal of PMESC is to free the programmer from dealing with application-independent issues such as load balancing, interprocessor communication, and program termination, while allowing her or him to concentrate on the application specific ones. PMESC combines the ingredients that are necessary to make a useful tool: portability, efficiency, and ease of use. The name PMESC comes from the phases of dynamic computation handled by the environment.

PMESC achieves portability by following a two-layered approach that concentrates all of the machine-dependent issues at the lower level and builds the upper level that takes care of more specific abstractions on top of that. As a result of this design, not only the user code but also the high-level layer of PMESC remain unchanged across different computers.

HH

Furthermore, each level is independent of the other which makes it possible to easily replace the lower level by an standard package such as MPI, and we are in the process of translating the lower level to MPI. PMESC achieves efficiency by providing different building blocks to address different programming issues. This flexible approach allows the programmer —not the language or system— to decide which of those blocks are most suitable for the particular application and the computer architecture. Finally, PMESC is easy to use because it was designed based on the applications and their requirements and it was implemented to fulfill them. Thus, it provides a methodology that is within the grasp of a C programmer and a syntax that is not different from standard C programming.

PMESC is the subject of Silvia's Ph.D. thesis [4]. We are presently writing a series of papers about PMESC and our mechanisms for evalutating it. The first of these is [5]. We expect to release the PMESC library shortly and are in discussions with Bill Gropp of Argonne National Laboratory about merging PMESC with Argonne's PETSc (Portable, Extensible Tookit for Scientific Computation) package.

Another problem studied was the parallel solution of structured linear systems arising in the solution of two-stage stochastic programs. A two-stage stochastic problem translates into a process of making a guess at a specified action based on present day information then correcting that action based on the effects of a collection of possible future events. This sort of problem arises, for example, in portfolio management: what seems like a good investment today may turn out to seem less promising once the effects of future happenings (wars, interest rate drops, etc.) on its earnings are considered [11].

The deterministic equivalent formulation of two-stage stochastic programs using interior point algorithms requires the solution of linear systems of the form $(AD^2A^T)dy = b$. The constraint matrix A has a dual, block-angular structure. Together with Stavros Zenios and Dafeng Yang of the Wharton School of Business, we have developed a parallel matrix factorization procedure using the Sherman-Morrison-Woodbury formula and based on the work of Birge and Qi [2]. This procedure requires the solution of smaller, independent systems of equations. With the use of optimal communication algorithms and careful attention to data layout we have obtained a parallel implementation that achieves near perfect speedup. Our analysis and experiments on an Intel iPSC/860 hypercube and a Connection Machine CM–5 show that the good performance scales to large problem sizes on large machines. The results of this work are published in [10].

Finally, we were motivated to study manipulation of the Gerschgorin disks by our study of how to solve the symmetric tridiagonal eigenproblem on the hypercube multiprocessor [9]. In that study, we noted that while it is easy to implement the bisection method on distributed-memory multiprocessors, it is not easy to achieve good performance in general. Processors assigned disjoint Gerschgorin disks can compute the eigenvalues within those disks independently of one another in parallel. However, processors assigned large disks or disks containing many eigenvalues take longer to complete their computations than those assigned small disks with few eigenvalues. Applying similarity transformations is one way to improve the processor load balance. We were able to determine the best way to compute a single eigenvalue of a symmetric tridiagonal matrix on a distributed-memory MIMD multiprocessor, and this work is published in [6]. Generalizing this result to an arbitrary number of eigenvalues seems now to be a problem that is intractable or that, at least, requires enormous effort for little reward. We were able to derive some specific preprocessing rules for symmetric eigenproblems but were not able to develop a general and practical approach. Parallel nonsymmetric eigensolvers based on tridiagonalization turned out to be a similarly unappealing problem. We were also able to make small improvements to the nonsymmetric method but were not able to overcome its most basic failings. Both the preprocessing problem and the nonsymmetric tridiagonalization problem are now on the back burner.

We have turned our attentions instead to two other general problems. With CU postdoc Zlatko Drmač, we are developing numerical methods and software for solving generalized eigenvalue and singular value problems with a focus on accuracy. Our preliminary results with the latter problem show that we will be able to produce software for those problems that, in some cases, is more accurate and more robust than that provided in LAPACK [1].

In conjunction with this large scale software development, we are producing a set of software tools to support the basic building blocks of linear algebra computation. Examples of these include norm computations and the computation of rotation angles. The first emphasis of this tool development will be improved accuracy. The second, and equally important, will be improved efficiency. Our emphasis will be on producing tools that run efficiently on cache-based workstations. While tools for fundamental computations are available among the BLAS and in LAPACK, we are discovering that the available tools do not always make good use of the memory hierarchy and large scale memory of modern workstations.

During the period of this grant, we also published the textbook [7]. It was released in April 1996. This book is an undergraduate textbook about the use of supercomputers in scientific applications. As such, it is a bridge between our research and educational interests. The textbook and the course based on it were awarded a DOE Undergraduate Computational Science Education Award in August 1995. We gave several invited talks about the book and course at educational conferences during the last year.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, and S. Hammarling, *LAPACK User's Guide*, SIAM, 1992.

[2] J. Birge and L. Qi, *Computing block-angular Karmarkar projections with applications to stochastic programming*, Management Science, 34 (1988).

[3] J. Choi, J. Dongarra, and D. Walker, *The design of a parallel, dense linear algebra software library: Reduction to Hessenberg, tridiagonal, and bidiagonal form*, Numerical Algorithms, 10 (1995), pp. 379–400.

[4] S. Crivelli, *A programming paradigm for distributed-memory computers*, PhD thesis, Dept. of Computer Science, University of Colorado at Boulder, 1995.

[5] S. Crivelli and E. Jessup, *A programming paradigm for distributed-memory computers.* Submitted for publication.

[6] ——, *The cost of eigenvalue computation on distributed-memory MIMD multiprocessors*, Parallel Computing, 4 (1995), pp. 833–846.

[7] L. Fosdick, E. Jessup, C. Schauble, and G. Domik, *An Introduction to High-Performance Scientific Computing*, MIT Press, 1996.

[8] C. Hendrickson, E. Jessup, and C. Smith, *A parallel eigensolver for dense symmetric matrices.* Submitted for publication.

[9] E. Jessup, *Parallel Solution of the Symmetric Tridiagonal Eigenproblem*, PhD thesis, Dept of Computer Science, Yale University, 1989.

[10] E. Jessup, D. Yang, and S. Zenios, *Parallel factorization of structured matrices arising in stochastic programming*, SIAM J. Optimization, 4 (1994), pp. 833–846.

[11] D. Yang. The Wharton School, personal communication.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.