

AD-A170 699

PARALLEL MATRIX COMPUTATIONS(U) MARYLAND UNIV COLLEGE
PARK DEPT OF COMPUTER SCIENCE G W STEWART ET AL.
12 MAY 86 AFOSR-TR-86-0548 AFOSR-82-0078

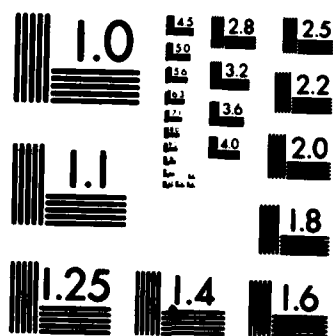
1/1

UNCLASSIFIED

F/G 9/2

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A170 699 REPORT DOCUMENTATION PAGE

2a. SECURITY CLASSIFICATION AUTHORITY		1b. RESTRICTIVE MARKINGS	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		5. MONITORING ORGANIZATION REPORT NUMBER(S) AFOSR-TR. 86-0548	
6a. NAME OF PERFORMING ORGANIZATION University of Maryland	6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION Air Force Office of Scientific Research	
6c. ADDRESS (City, State, and ZIP Code) College Park, Maryland 20742		7b. ADDRESS (City, State, and ZIP Code) Directorate of Mathematical & Information Sciences, Bolling AFB DC 20332-6448	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AFOSR	8b. OFFICE SYMBOL (if applicable) NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR-82-0078	
8c. ADDRESS (City, State, and ZIP Code) Bolling AFB DC 20332-6448		10. SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO. 61102F	TASK NO. 2304
		WORK UNIT ACCESSION NO. A3	
11. TITLE (Include Security Classification) PARALLEL MATRIX COMPUTATIONS			
12. PERSONAL AUTHOR(S) G. W. Stewart, Dianne P. O'Leary			
13a. TYPE OF REPORT Interim	13b. TIME COVERED FROM 1985 TO 1986	14. DATE OF REPORT (Year, Month, Day) 86-5-12	15. PAGE COUNT 11
16. SUPPLEMENTARY NOTATION			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP	
		The...	
19. ABSTRACT (Continue on reverse if necessary and identify by block number)			
This project concerns the design and analysis of algorithms to be run in a processor-rich environment. We focus primarily on algorithms that require no global control and that can be run on systems with only local connections among processors. We investigate the properties of these algorithms both theoretically and experimentally. The experimental work is done on the ZMOB, a working parallel computer operated by the Laboratory for Parallel Computation of the Computer Science Department at the University of Maryland. To give our work direction, we have focused on two areas: 1. Dense problems from numerical linear algebra 2. The iterative and direct solution of sparse linear systems.			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL John Thomas		22b. TELEPHONE (Include Area Code) (202)-767-5026	22c. OFFICE SYMBOL NM

DTIC FILE COPY

DTIC ELECTE
AUG 8 1986

**Technical Summary Report
AFOSR 82-0078
Parallel Matrix Computations
April, 1985 - April, 1986**

*Approved for public release,
distribution unlimited*

**Supported by
Air Force Office of Scientific Research
Numerical Mathematics
Bolling Air Force Base, D.C. 20332**

**Research conducted at
Department of Computer Science
University of Maryland
College Park, MD 20742
(301) 454-2001**

**Principal Investigators
Professor G. W. Stewart (410-68-8197)
Assoc. Prof. Dianne P. O'Leary (359-46-2895)**

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
AI	



1. INTRODUCTION

This project concerns the design and analysis of algorithms to be run in a processor-rich environment. We focus primarily on algorithms that require no global control and that can be run on systems with only local connections among processors. We investigate the properties of these algorithms both theoretically and experimentally. The experimental work has been done on the ZMOB, a parallel computer operated by the Laboratory for Parallel Computation of the Computer Science Department at the University of Maryland, although recently we have gained access to a BBN Butterfly computer as well.

The ZMOB consists of 128 processors which communicate by message passing over a communications network which provides a complete network of connections between processors. The start-up time for interprocessor communication, the per-word transmission overhead, and the floating point computation time is all of the same order of magnitude.

It is important to be precise about how we use the ZMOB in our research. What we do *not* do is to investigate algorithms for the ZMOB itself. Instead we use the fact that the ZMOB appears to be a completely connected network to simulate various locally connected networks of processors. Thus we can investigate, in a realistic setting, the effects on our algorithms of various processor interconnections.

Our activities may be divided into four categories: algorithms, software development, theoretical analysis, and experimental analysis.

To give our work direction, we have focused on dense and sparse problems from numerical linear algebra. We discuss in this summary the research projects that we have pursued under this grant support over the past year.

2. Summary of Work

Our activities have ranged from theoretical analysis to algorithmic design and software development. We summarize this work in the following sections. For details, consult the annotated list of references in Appendix A.

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DTIC
This technical report has been reviewed and is
approved for public release IAW AFR 190-12.
Distribution is unlimited.
MATTHEW J. KERPER
Chief, Technical Information Division

2.1 Data-Flow Algorithms and Domino

We have based most of our work in this area on the notion of a *data-flow algorithm*. The computations in a data-flow algorithm are done by independent *computational nodes*, which cycle between requesting data from certain nodes, computing, and sending data to certain other nodes. More precisely, the nodes lie at the vertices of a directed graph whose arcs represent lines of communication. Each time a node sends data to another node, the data is placed in a queue on the arc between the two nodes. When a node has requested data from other nodes, it is blocked from further execution until the data it has requested arrives at the appropriate input queues. An algorithm organized in this manner is called a data-flow algorithm because the times at which nodes can compute is controlled by the flow of data between nodes.

Data-flow algorithms are well suited for implementation on networks of processors which communicate by message passing. Each node in a computational network is regarded as a process residing on a fixed member of a network of processors. We allow more than one node on a processor. Since many nodes will be performing essentially the same functions, we allow nodes which share a processor also to share pieces of reentrant code, which we call *node programs*. Each processor has a resident node communication and control system to receive and transmit messages from other processors and to awaken nodes when their data has arrived.

Data-flow algorithms have a number of advantages.

1. The approach eliminates the need for global synchronization.
2. Parallel matrix algorithms, including all algorithms for systolic arrays, have data-flow implementations.
3. Data-flow algorithms can be coded in a high-level sequential programming language, augmented by two communication primitives for sending and receiving data.
4. Data-flow computations can be supported by a very simple node communication and control system.
5. The approach allows the graceful handling of missized problems, since several nodes can be mapped onto one processor.
6. By mapping all nodes in a data-flow algorithm onto a single processor, one can debug parallel algorithms on an ordinary sequential processor.

Because of the conceptual convenience and practical utility of the data-flow approach, and because of the absence of any standard for writing transportable

algorithms for parallel machines, we have implemented these ideas in a node communication and control system called Domino. We have documented the system and provided examples of its use in a technical report. The system currently runs on the ZMOB, Vaxes under Unix or VMS, Sun workstations, and IBM PC's. A Butterfly implementation is underway. The code is currently being used for numerical analysis and for neural network studies at Maryland. The system has been very valuable to us in our research, and will be used in a course on parallel computation taught next fall at Maryland. We have already received numerous inquiries from potential users in industry and academics and will make Domino available over the Arpanet through Netlib at Argonne National Lab.

2.2 Theoretical Developments

Work has been done in the design of parallel architectures and in the analysis of parallel algorithms.

Our work on the determinacy of our data-flow model for parallel computation led us to propose a modification of the design of systolic arrays in order to eliminate the need for global synchronization. Each cell in the array is augmented by a feedback circuit so that data is sent from one cell to another only when the receiver is ready to process it. We call such networks *systaltic arrays*.

David C. Fisher completed a thesis partially supported by this grant which studies the complexity of various tasks in matrix computation, assuming that processors perform computations so fast that the communication delay in sending between physically distant processors is significant. Lower bounds on execution time were obtained, and optimal algorithms were derived for several problems.

2.3 Algorithm Design, Analysis, and Testing

The chief difficulty with the data-flow approach is that the behavior of the algorithms cannot be analyzed purely from the local viewpoint of the node programs. This is one reason for supplementing theory with experiment.

This year, we devised a number of new parallel algorithms. For dense matrices we developed algorithms for computing the QR factorization of a matrix and a parallel version of the QR algorithm for computing eigenvalues. For sparse matrices, we are working on simultaneous iteration methods for eigenvalues and block conjugate gradient algorithms for solving linear systems.

3. Summary

Four papers supported under this grant appeared in refereed journals during this year, and two were accepted for publication. Invited talks were given at universities and at conferences whose themes ranged from parallel processing to statistics and mathematical programming. One graduate student completed his dissertation and two others made substantial progress. The Domino software has been documented and prepared for distribution.

Our work resulted in a collection of parallel algorithms for matrix computations, a data-flow operating system to support experiments, and theoretical investigation into complexity and determinacy issues in parallel matrix computations.

Appendix
Accomplishments under Grant AFOSR 82-0078

I. Technical Reports

- (1) G. W. Stewart, *Computing the CS Decomposition of a Partitioned Orthonormal Matrix*, TR-1159, May, 1982.

This paper describes an algorithm for simultaneously diagonalizing by orthogonal transformation the blocks of a partitioned matrix having orthonormal columns.

- (2) G. W. Stewart *A Note on Complex Division*, TR-1206, August, 1982.

An algorithm (Smith, 1962) for computing the quotient of two complex numbers is modified to make it more robust in the presence of underflows.

- (3) D. P. O'Leary, *Solving Sparse Matrix Problems on Parallel Computers*, TR-1234, December, 1982.

This paper has a dual character. The first part is a survey of some issues and ideas for sparse matrix computation on parallel processing machines. In the second part, some new results are presented concerning efficient parallel iterative algorithms for solving mesh problems which arise in network problems, image processing, and discretization of partial differential equations.

- (4) G. W. Stewart, *A Jacobi-like Algorithm for Computing the Schur Decomposition of a Non-Hermitian Matrix*, TR-1321, August, 1983.

This paper describes an iterative method for reducing a general matrix to upper triangular form by unitary similarity transformations. The method is similar to Jacobi's method for the symmetric eigenvalue problem in that it uses plane rotations to annihilate off-diagonal elements, and when the matrix is Hermitian it reduces to a variant of Jacobi's method. Although the method cannot compete with the QR algorithm in serial implementation, it admits of a parallel implementation in which a double sweep of the matrix can be done in time proportional to the order of the matrix.

- (5) D. P. O'Leary and G. W. Stewart, *Data-Flow Algorithms for Matrix Computations*, TR-1366, January, 1984.

In this work we develop some algorithms and tools for solving matrix problems on parallel processing computers. Operations are synchronized through data-flow alone, which makes global synchronization unnecessary and enables the algorithms to be implemented on machines with very simple operating systems and communications protocols. As examples, we present algorithms that form the main modules for solving Liapunov matrix equations. We compare this approach to wavefront array processors and systolic arrays, and note its advantages in handling missized problems, in evaluating variations of algorithms or architectures, in moving algorithms from system to system, and in debugging parallel

algorithms on sequential machines.

- (6) G. W. Stewart, W. F. Stewart, D. F. McAlister, *A Two Stage Iteration for Solving Nearly Uncoupled Markov Chains*, TR-1384, 1984.

This paper presents and analyses a parallizable algorithm for solving Markov chains that arise in queuing models of loosely coupled systems.

- (7) Dianne P. O'Leary and Robert E. White, *Multi-Splittings of Matrices and Parallel Solution of Linear Systems*, TR-1362, December, 1983.

We present two classes of matrix splittings and give applications to the parallel iterative solution of systems of linear equations. These splittings generalize regular splittings and P-regular splittings, resulting in algorithms which can be implemented efficiently on parallel computing systems. Convergence is established, rate of convergence is discussed, and numerical examples are given.

- (8) David C. Fisher, *In Three-Dimensional Space, the Time Required to Add N Numbers is $O(N^{1/4})$* , TR-1431, August, 1984.

How quickly can the sum of N numbers be computed with sufficiently many processors? The traditional answer is $t = O(\log N)$. However, if the processors are in R^d (usually $d \leq 3$), addition time and processor volume are bounded away from zero, and transmission speed and processor length are bounded, $t \geq O(N^{1/d+1})$.

- (9) Dianne P. O'Leary, G. W. Stewart, *On the Determinacy of a Model for Parallel Computation*, TR-1456, November, 1984. (Obsolete: see TR-1553)

In this note we extend a model of Karp and Miller for parallel computation. We show that the model is deterministic, in the sense under different scheduling regimes each process in the computation consumes the same input and generates the same output. Moreover, if the computation halts, the final state is independent of scheduling.

- (10) Dianne P. O'Leary, *Systolic Arrays for Matrix Transpose and Other Reorderings*, TR-1481, March, 1985.

In this note, a systolic array is described for computing the transpose of an $n \times n$ matrix in time $3n-1$ using n^2 switching processors and n^2 bit buffers. A one-dimensional implementation is also described. Arrays are also given to take a matrix in by rows and put it out by diagonals, and vice versa.

- (11) Dianne P. O'Leary, G. W. Stewart, *Assignment and Scheduling in Parallel Matrix Factorization*, TR-1486, April, 1985.

We consider in this paper the problem of factoring a dense $n \times n$ matrix on a network consisting of P MIMD processors when the network is smaller than the number of elements in the matrix ($P < n^2$). The specific example analyzed is a computational network

that arises in computing the LU, QR, or Cholesky factorizations. We prove that if the nodes of the network are evenly distributed among processors and if computations are scheduled by a round-robin or a least-recently-executed scheduling algorithm, then optimal order of speed-up is achieved. However, such speed-up is not necessarily achieved for other scheduling algorithms or if the computation for the nodes is inappropriately split across processors, and we give examples of these phenomena. Lower bounds on execution time for the algorithm are established

- (12) Dianne P. O'Leary, G. W. Stewart, *From Determinacy to Systolic Arrays*, TR-1553, August, 1985.

In this paper we extend a model of Karp and Miller for parallel computation. We show that the extended model is deterministic, in the sense that under different scheduling regimes each process in the computation consumes the same input and generates the same output. Moreover, if the computation halts, the final state is independent of scheduling. The model is applied to the generation of precedence graphs, from which lower time bounds may be deduced, and to the synchronization of systolic arrays by local rather than global control.

- (13) David C. Fisher, *Matrix Computation on Processors in One, Two, and Three Dimensions*, TR-1556, August, 1985.

Suppose a problem is to be solved on a d -dimensional parallel processing machine. Assume that transmission speed is finite. Under this and other "real world" assumptions, if a problem requires I inputs, K outputs and T computations, then time required to solve the problem is greater than or equal to $O(\max(I^{1/d}, K^{1/d}, T^{1/(d+1)}))$. Algorithms for certain matrix computations are developed. The problems are divided into *atoms*. The algorithms are described and analyzed with the use of *step* and *processor assignment functions*. These assign each atom to a step and a processor. Here is a table showing the time for algorithms presented in this paper:

Problem	Linear Grid (1-D)	Square Grid (2-D)	Cubic Grid (3-D)
Summation of k numbers	$O(k)$	$O(k^{1/2})$	$O(k^{1/3})$
Multiply a $k \times k$ matrix by a k vector	$O(k^2)$	$O(k)$	$O(k^{2/3})$
Multiply two $k \times k$ matrices	$O(k^2)$	$O(k)$	$O(k^{3/4})$
Cholesky factorization of a $k \times k$ matrix	-----	$O(k)$	-----

Except for matrix multiplication in 3-dimensions, these times are a constant multiple of the lower bounds. Programs are given which will execute these algorithms on an appropriate parallel processing machine.

- (14) D. P. O'Leary, G. W. Stewart, R. van de Geijn, *DOMINO: a message passing environment for parallel computation*. TR-1648, April, 1986.

This report is a description of DOMINO, a system to coordinate computations on a network of processors. It implements an extension of a model of parallel computation (Karp and Miller, SIAM J. Appl. Math., 1966), in which computations are synchronized by messages passed between the processes performing the computations. The system is organized in such a way that meaningful debugging can be done on a single processor. In order to make DOMINO transportable, system dependent features have been isolated in two

interfaces.

- (15) G. W. Stewart, *Communication in parallel algorithms: an example*, April, 1986.

The purpose of this note is to describe and analyze a parallel algorithm for computing the QR factorization of an $n \times p$ matrix. The algorithm is designed to run on a ring of r processors that communicate by message passing.

II. Presentations

- (1) D. P. O'Leary, *Solving Mesh Problems on Parallel Computers*, Bell Laboratory, Murray Hill, N.J., January, 1983
IBM T. J. Watson Laboratory, Yorktown Heights, N.Y., January, 1983.
- (2) G. W. Stewart, *A Jacobi-like Algorithm for Computing the Schur Decomposition of a Non-Hermitian Matrix* (invited), Symposium on Numerical Analysis and Computational Complex Analysis, Zurich, Switzerland, August, 1983. Also presented at North Carolina State University, September, 1983, and at University of Houston, November, 1983,
- (3) G. W. Stewart, *The Structure of Nearly Uncoupled Markov Chains* (invited), International Workshop on Systems Modeling, Pisa, Italy, September, 1983.
- (4) G. W. Stewart, *Data Flow Algorithms for Parallel Matrix Computations* (invited), SIAM Conference on Parallel Processing for Scientific Computing, Norfolk, VA, November, 1983.
- (5) D. P. O'Leary, *Parallel Computations for Sparse Linear Systems* (minisymposium invitation), SIAM 1983 Fall Meeting, Norfolk, VA, November, 1983.
- (6) D. C. Fisher, *Numerical Computations on Multiprocessors with Only Local Communications* (poster session), SIAM Conference on Parallel Processing for Scientific Computing, Norfolk, VA, November, 1983.
- (7) G. W. Stewart, *Parallel Computations on the ZMOB*, Annual meeting of CER participants, University of Utah, March, 1984.
- (8) D. P. O'Leary, *Data-flow Algorithms for Matrix Computations* (minisymposium invitation), ACM SIGNUM Conference on Numerical Computations and Mathematical Software for Microcomputers, Boulder, Colorado, March, 1984.
- (9) D. P. O'Leary, *Solution of Matrix Problems on Parallel Computers* (invited presentation), Gatlinburg IX Meeting on Numerical Linear Algebra, Waterloo, Ontario, Canada, July, 1984. Also presented at Oak Ridge National Laboratory, September, 1984; National Bureau of Standards, Boulder, Colorado, March, 1984; Yale University, November, 1984; Cornell University, January, 1985; Courant Institute, February, 1985.
- (10) G. W. Stewart, *The Data-Flow Approach to Matrix Computations*, Los Alamos Scientific Laboratory, October, 1984.

- (11) G. W. Stewart, *The Impact of Computer Architecture on Statistical Computing*, (invited) SIAM/ISA/ASA Conference on Frontiers of Statistical Computing, October, 1984.
- (12) G. W. Stewart, *Determinacy*, (invited) Symposium in Honor of G. Dahlquist, Stockholm, January, 1985.
- (13) D. C. Fisher, *Fast Matrix Multiplication on Square and Cubic Grids of Processors*, SIAM Conference on Applied Linear Algebra, Raleigh, April, 1985.
- (14) G. W. Stewart, *The Parallel Solution of Sparse Unsymmetric Eigenvalue Problems*, (invited) IBM Workshop on Large Eigenvalue Problems, Oberlech, Austria, July, 1985.
- (15) D. P. O'Leary, *A Testbed for Parallel Algorithm Development*, (invited) SIAM Conference on Parallel Processing for Scientific Computing, Norfolk, VA, November, 1985.
- (16) D. P. O'Leary, G. W. Stewart, Robert van de Geijn, *Domino, a Transportable System for Parallel Computing*, Army Research Conference on Parallel Computing and Medium Scale Multiprocessors, Stanford, CA, January, 1986.
- (17) G. W. Stewart, *Parallel Scientific Computing*, University of Colorado Conference on Computer Science, Boulder, CO, March, 1986.

IV. Publications

- (1) G. W. Stewart, "Computing the CS Decomposition of a Partitioned Orthonormal Matrix," *Numerische Mathematik* 40 (1982) 297-306.
- (2) D. P. O'Leary, "Ordering schemes for parallel processing of certain mesh problems," *SIAM Journal on Scientific and Statistical Computing* 5 (1984) 620-632.
- (3) D. P. O'Leary, G. W. Stewart, "Data-flow algorithms for parallel matrix computations," *Communications of the ACM*, 28 (1985) 840-853.
- (4) G. W. Stewart, "A Jacobi-like Algorithm for Computing the Schur Decomposition of a Non-Hermitian Matrix," *SIAM Journal on Scientific and Statistical Computing*, 6 (1985) 853-864.
- (5) D. P. O'Leary, R. E. White, "Multi-splittings of matrices and parallel solution of linear systems," *SIAM Journal on Algebraic and Discrete Methods*, 6 (1985) 630-640.
- (6) G. W. Stewart "A Note on Complex Division," *ACM Transactions on Mathematical Software* 11 (1985) 238-241.
- (7) D. P. O'Leary, "Systolic Arrays for Matrix Transpose and Other Reorderings," *IEEE Transactions on Computers*, to appear.

- (8) D. P. O'Leary and G. W. Stewart, "Assignment and Scheduling in Parallel Matrix Factorization," *Linear Algebra and Its Applications*, to appear.

END

DTIC

9-86