Wiley Series on Parallel and Distributed Computing
Albert Y. Zomaya, Series Editor

NEW PARALLEL ALGORITHMS FOR DIRECT SOLUTION OF LINEAR EQUATIONS

C. SIVA RAM MURTHY
K. N. BALASUBRAMANYA MURTHY
SRINIVAS ALURU

NEW PARALLEL ALGORITHMS FOR DIRECT SOLUTION OF LINEAR EQUATIONS

C. Siva Ram Murthy
Indian Institute of Technology, Madras

K. N. Balasubramanya Murthy
Malnad College of Engineering, India

Srinivas Aluru
Iowa State University

A WILEY-INTERSCIENCE PUBLICATION

JOHN WILEY & SONS, INC.

New York • Chichester • Weinheim • Brisbane • Singapore • Toronto

This book is printed on acid-free paper. ♾

Library of Congress Cataloging-in-Publication Data:

Murthy, C. Siva Ram.
New parallel algorithms for direct solution of linear equations / C. Siva Ram Murthy, K. N. Balasubramanya Murthy, Srinivas Aluru.
p. cm. — (Wiley series on parallel and distributed computing)
Includes bibliographical references and index.
ISBN 0-471-36165-8 (cloth: alk.paper)
1. Differential equations, Linear–Data processing. 2. Computer algorithms. 3. Parallel processing (Electronic computers) I. Murthy, K.N. Balasubramanya. II. Aluru, Srinivas. III. Title. IV. Series.

QA372 M87 2000
515′354′0285 — dc21

99-086570

Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

NEW PARALLEL ALGORITHMS FOR DIRECT SOLUTION OF LINEAR EQUATIONS

PREFACE

The problem of solving a set of linear algebraic equations $Ax = b$ (where A is a known $N \times N$ dense or sparse matrix, and x and b are unknown and known N vectors, respectively) is one of the central problems in computational mathematics and computer science. Many books have been written on the topic of solving systems of linear equations in parallel using direct methods. The approach used, so far, has been to parallelize the existing efficient sequential algorithms and address the implementaion of these algorithms on different parallel architectural platforms. Algorithms for solving systems of linear equations typically consist of multiple solution phases. The fundamental problem with this approach is that the strict sequentiality among the multiple solution phases and the problems of numerical stability associated with the sequential algorithms also transcend into their parallelized versions. What sets this book apart is the development of a new approach called *bidirectional elimination* for the solution of both dense and sparse linear systems. The bidirectional elimination approach allows for improved speedup by avoiding multiple solution phases, better numerical stability, and efficient implementation on multiprocessor systems.

The book, a repertoire of new parallel algorithms based on bidirectional elimination for direct solution of linear equations, is organized into nine chapters. Chapter 1 provides an introduction to solving systems of linear equations and a survey of parallel linear system solvers. Chapter 2 presents introductory material about parallel computing including issues in parallel algorithm design and the model of computation used in the rest of the book. Chapters 3 through 6 deal with parallel bidirectional Gaussian elimination (BGE), division-free BGE, bidirectional LU (BLU) factorization, bidirectional Householder reductions (BHR), bidirectional modified Gram-Schmidt (BMGS) orthogonal factorization, and bidirectional Givens rotations (BGR) algorithms for solving dense systems of linear equations. Chapters 7 and 8 are devoted to parallel bidirectional algorithms for solving sparse linear systems with explicit discussion on ordering and symbolic factorization. Chapter 9 deals with controlling of inherent parallelism in bidirectional elimination based algorithms and scope for further research work in numerical linear algebra by using the novel bidirectional elimination approach. We describe the new parallel algorithms in an architecture-independent manner using well understood communication primitives such as send/receive and broadcast. In analyzing the parallel algorithms, we derive expressions for computation and communication times separately using *permutation network* model.

The primary purpose of writing this book is for easy dissemination of new research results in parallel numerical algebra. Thus, we expect it to be of value to researchers and practicing professionals. There is a great need and interest in many scientific disciplines (physics, astronomy, chemical engineering, civil engineering, and mechanical engineering to name a few) for solving problems using high-performance computers. Most of the heavy users of high-performance computing have large-scale problems to solve and parallel numerical algebra is an indispensable tool for them. We expect the book to be of interest of such audience. As for classroom use, the book can serve as a supplementary text in courses such as parallel computing, parallel algorithms, parallel scientific computing, parallel numerical methods, and computational science. Most chapters of the book include examples, problems, and bibliographic notes to improve reader's understanding of the material and aid interested readers in learning more about the related and advanced topics. We encourage the readers to send suggestions and any information related to the material presented in the book to us. We welcome their comments and suggestions.

The financial support for this books writing project came from the Ministry of Human Resource Development, New Delhi, India and the National Science Foundation under NSF Career Award CCR-9702991. We are indebted to these agencies for the same. Last though not least, we acknowledge the love and affection from our families. This project would never have been successfully completed but for their understanding and patience.

C. SIVA RAM MURTHY
K. N. BALASUBRAMANYA MURTHY
SRINIVAS ALURU

August 2000

CONTENTS

CHAPTER 1

Introduction

1.1 SOLVING LINEAR EQUATIONS

The problem of solving a system of linear equations is central to many applications in engineering and scientific computing. Algorithms and architectures for the high-speed solution of linear equations were the focus of considerable research attention in the late 1990s because of their presence in a wide variety of fields in science and engineering.

A system of N linear equations is represented as follows:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N &= b_2 \\ &\vdots \\ a_{N1}x_1 + a_{N2}x_2 + \cdots + a_{NN}x_N &= b_N \end{aligned}$$

or in matrix form

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1N} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2N} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{N1} & a_{N2} & a_{N3} & \cdots & a_{NN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_N \end{bmatrix}$$

as $Ax = b$, where $A = [a_{ij}](i, j = 1, 2, \ldots, N)$ is an $N \times N$ nonsingular coefficient matrix, x is an $N \times 1$ unknown solution vector, and b is an $N \times 1$ known right-side vector. The linear system $Ax = b$ relates every unknown to N coefficients so that, to obtain $x_i (i = 1, 2, \ldots, N)$, the influence of at most $(N - 1)$ other unknowns on its value must be eliminated.

Depending on the nature of the coefficient matrix, A, we have many types of linear systems such as

- Real and nonsingular
- Real and symmetric positive definite

- Sparse and symmetric positive definite
- Sparse
- Tridiagonal

To solve this class of linear systems, we have different linear system solvers (or methods). These solvers can be mainly of two classes as iterative and direct (however, it is also possible to have methods that have the features of both). *Iterative methods* employ a starting solution and converge to a value that is close to the exact solution by iteratively refining the starting solution. However, iterative methods do not guarantee a solution for all systems of linear equations. Commonly used iterative methods for solving systems of linear equations are Jacobi iterative method, Gauss–Seidel method, successive overrelaxation (SOR) method, and conjugate gradient (CG) method. Iterative methods are frequently used to solve large sparse systems of linear equations generated while working with partial differential equations using discrete methods. Iterative methods have the following two distinct advantages over direct methods:

- If iterative methods do not formally yield a solution in a finite number of steps, one can terminate such a procedure after a finite number of iterations by which time it will have produced a sufficiently good approximate solution.
- Most iterative methods possess the attractive feature that they require arithmetic operations only on nonzero entries of the coefficient matrix A.

It is important to note that the solutions obtained by using iterative methods will have both truncation (due to truncation of iterative process) and roundoff (due to finite-digit representation of numbers in computers following an arithmetic operation) errors.

Direct methods are those in which for a given linear system of equations, exact solutions (within a roundoff bound determined by machine precision) can be obtained with a preestimated number of arithmetic operations. The central idea of standard direct methods of solving linear equations is to either

1. find the inverse of the coefficient matrix, A and then compute $x = A^{-1}b$ [45, 56] (this approach is seldom used because of very large operation count) or
2. transform the coefficient matrix A into an equivalent triangular or diagonal form in which coupling between the unknowns is reduced.

Commonly used direct methods for solving a given system of linear equations include the Gaussian elimination (GE) method, the lower–upper (LU) decomposition method, Gauss–Jordan (GJ) method, Cramer's rule (CR), the Givens rotations–based method, the Householder reductions method, and Gram–Schmidt orthogonalization method.

1.2 A BRIEF SURVEY OF PARALLEL LINEAR SYSTEM SOLVERS

There is an enormous amount of literature on parallel solutions of linear equations in the form of research papers, technical reports, and books. In this section, an attempt is made to briefly present the direct linear system solvers. Parallel algorithms based on iterative techniques have been studied in considerable detail in the literature [36, 101, 128, 156, 163, 199] and are not considered further in this book. Bertsekas and Tsitsiklis [36], Demmel et al. [57], Evans [68], Gallivan et al. [74], Golub and Van Loan [91], Heller [101], Kamal Abdali [121], Lakshmivarahan and Dhall [132], Miranker [155], Ortega and Voigt [163], Sameh and Kuck [179], Peng and Sedukhin [185, 186], and V. Kumar et al. [199] describe the current state of art in parallel numerical algebra. A substantial portion of the large body of literature that deals with the parallel linear system solvers has paid considerable attention only to the parallelization of the sequential methods and to devising efficient strategies for scheduling coarse-, medium-, or fine-grain tasks in them onto the processors in a parallel computer system.

1.2.1 Dense Linear Systems

The parallel algorithms based on Gaussian elimination, LU decomposition, Cholesky factorization, and Givens rotations for solving dense linear systems are discussed in the literature [2, 10, 12, 32, 46, 47, 48, 59, 71, 77, 128, 147, 181, 195]. In these papers, attempts have been made to improve the performance of the algorithms on different parallel computer architectures by using strategies such as pipelining, load balancing, subcube matrix decomposition, and vector segmentation. However, the strict sequentiality among the multiple solution phases enforced by these algorithms impairs the speedup. Another disadvantage of these algorithms is that the multiple solution phases may not optimally be implemented on a single multiprocessor system using their respective efficient algorithms. Faddeeva algorithms for solving linear equations have also been discussed [58, 153, 162].

Although the Gauss–Jordan method [54, 89, 138, 154, 203] has no multiple solution phases, it has some numerical stability problems [17] and, in addition, it does not support pairwise pivoting for efficient implementation on processor arrays. An overlaying technique (which is closely related to the idea of iterative refinement) has been proposed [35] for solving linear equations in real-time computing. This technique utilizes an approximation of an inverse matrix as a sum of matrix products that allows the required solution to be obtained in $O(N^2)$ rather than in $O(N^3)$ operations.

As the aforementioned parallel algorithms have been derived by parallelizing the sequential algorithms, the inherent degree of parallelism in them is accordingly limited. To explore afresh the possibility of designing faster solution methods, the following notable attempts have been made:

- Evans and Hatzopoulos [65] have proposed a new factorization method called *quadrant interlocking factorization* (QIF) with a large degree of parallelism [65, 204, 205] for the solution of a linear system. This method first decomposes the matrix A into WZ factors and then finds x by solving two alternative systems: $Wy = b$ and $Zx = y$. Again, it is important to note that these three phases have to be carried out in a sequential order. Garcia et al. discussed the implementation of QIF on hypercube computers [75].

- A modified parallel Cramer rule (PCR) algorithm with a very high degree of inherent parallelism has been proposed by Sridhar and others [18, 189, 190]. But, this algorithm is applicable only for diagonally dominant systems as it does not support pivoting. The PCR algorithm does not have the back substitution phase. Further, the authors have given systolic array implementation of their algorithm [190].
- Balasubramanya Murthy and Siva Ram Murthy have proposed the bidirectional Gaussian elimination and QR decomposition algorithms and discussed their implementation on hypercube and mesh connected multiprocessors [19–22, 24, 26, 27, 29–31, 37, 38]. These algorithms are back substitution-free and exhibit greater numerical stability characteristics. Further, they have explained the problem partitioning techniques (when the problem size exceeds the available number of processors) and fault tolerance aspects of their algorithms.
- Kamal Abdali has discussed a method for solving linear systems of equations using `*-semirings` [121]. The author uses the concept of eliminants and asterates/closures, which behave analogously in many ways to determinants and matrix inverses, respectively, to solve various problem classes, including the solution of linear equations. However, the implementation of the method on parallel computers is not addressed in Abdali's paper.

1.2.2 Sparse Linear Systems

Large sparse systems of linear equations arise in various applications such as finite-element analysis, computational fluid dynamics, and power system analysis. Developing fast parallel algorithms for solving sparse linear systems was a major focus of research in the late 1990s. The techniques for solving sparse linear systems involve more complex data structures and algorithms than do their dense counterparts. The current state of art in developing parallel algorithms for sparse linear systems is described in the literature [1, 6, 7, 11, 13, 16, 42, 62, 63, 67, 82, 88, 90, 93–95, 99, 100, 104–106, 111, 113, 116, 117, 122, 123, 124, 127, 132, 134, 135, 142, 149, 164, 174, 179, 183, 184, 194, 196, 197, 199, 201, 209].

Although there is substantial parallelism inherent in sparse linear systems, efforts made till date to develop efficient parallel algorithms for solving these have achieved only limited success. This is because the parallelized versions of sequential algorithms are unable to fully exploit limited inherent parallelism offered by the sparse linear systems. Further, the goal of a good sequential algorithm (i.e., minimizing the total operation count) directly conflicts with the goal of a good parallel algorithm (i.e., maximizing the number of concurrent subproblems).

Existing work on solving sparse symmetric and general sparse linear systems may be classified according to the phases of solution (such as numerical factorization, substitution, and ordering) that each work addresses.

Parallelization of the numerical factorization phase has received much attention [3, 14, 84, 86, 88, 99, 199] because it is a computationally intensive phase. A class of algorithms called *multifrontal algorithms* has also gained popularity [62, 146]. Ashcraft et al. have compared the fan-out, fan-in, and multifrontal approaches to sparse numerical factorization [15].

The substitution phase, which involves solution of triangular systems, has limited inherent parallelism. Therefore, efforts toward parallelizing this phase have received much less attention. Solving sparse triangular systems in parallel is also discussed [86, 103, 192].

Literature on the various techniques for the ordering phase is available [85, 115, 134, 135, 143, 144]. Work on developing parallel ordering algorithms has been fairly rudimentary to date [52, 141, 167]. Work on parallel algorithms for the symbolic factorization phase is also available [3, 90, 191]. Bidirectional elimination parallel algorithms for solving sparse linear equations are also discussed [23, 25, 28, 122, 123, 124].

In this book, we discuss new back substitution–free parallel algorithms using *bidirectional* (*two-sided* or *successive*) elimination technique for the solution of both dense and sparse linear equations. The algorithms, apart from being numerically stable, provide an opportunity for efficient implementation on a single multiprocessor system. Most importantly, the bidirectional algorithms are capable of producing the diagonal form in the same amount of parallel time required for obtaining the triangular form using the existing methods.

1.3 ORGANIZATION OF THE BOOK

The book is organized into nine chapters as follows:

- Chapter 1 contains introduction to solving systems of linear equations and a survey of parallel linear system solvers.
- Chapter 2 presents introductory material about parallel computing, including issues in parallel algorithm design and the model of parallel computation we use throughout the rest of the book.
- Chapter 3 deals with the parallel bidirectional Gaussian elimination (BGE) algorithm for solving linear system of equations. It also describes a division-free BGE method for the solution of linear equations. It covers the implementation of the BGE algorithm on parallel platforms.
- Chapter 4 is devoted to the parallel bidirectional LU (BLU) factorization algorithm for solving dense linear systems with multiple b vectors. The implementation of this algorithm on parallel platforms is discussed in detail.
- Chapter 5 presents parallel bidirectional Householder reductions (BHRs) and bidirectional modified Gram–Schmidt (BMGS) orthogonal factorization algorithms and also their implementations for solving dense linear systems on parallel computing systems.

- Chapter 6 discusses the parallel bidirectional Givens rotations (BGR) algorithm for solving dense system of linear equations. It also contains the implementation of the algorithm on parallel computing systems.
- Chapters 7 and 8 are devoted to parallel bidirectional algorithms for solving sparse symmetric and general sparse linear systems with explicit discussion on ordering, symbolic factorization, and performance-related topics.
- Chapter 9 deals with the controlling of inherent parallelism in the bidirectional elimination-based algorithms. Finally, it gives scope for further research work in numerical linear algebra by using novel bidirectional elimination principles.

Each chapter contains a few carefully framed problems for the reader to analyze and solve.

PROBLEMS

1. Prepare a chart of different types of linear systems.

2. Enumerate different numerical methods that are used for solving linear systems.

3. Determine whether each of the following matrices is symmetric and/or singular.

(**a**)

$$A = \begin{pmatrix} 4 & 7 & 1 \\ 7 & -3 & 6 \\ 1 & 6 & 8 \end{pmatrix}$$

(**b**)

$$B = \begin{pmatrix} 7 & 6 & 5 \\ 1 & 2 & 1 \\ 3 & -2 & 1 \end{pmatrix}$$

4. Determine the sparsity (i.e., the ratio of the number of zero elements to the total number of elements) associated with each of the following matrices:

(**a**)

$$A = \begin{pmatrix} 4 & 0 & 7 & 1 \\ 7 & -3 & 0 & 6 \\ 2 & 0 & 1 & 7 \\ 1 & 0 & 5 & 8 \end{pmatrix}$$

(**b**)

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 3 & 0 \\ 0 & 1 & 4 & 0 \\ 0 & 1 & 0 & 8 \end{pmatrix}$$

5. With an example show that the product of two symmetric matrices need not be symmetric.