On the Bit-Complexity of Discrete Solutions of PDEs: Compact Multigrid

Victor Pan *

Computer Science Department, SUNY
Albany, NY 12222
and
Mathematics and Computer Science Department
Lehman College
CUNY, Bronx, NY 10468

John Reif **

Department of Computer Science
Duke University
Durham, NC 27706

Summary

The topic of Partial Differential Equations (PDEs) is an interesting area where the techniques of discrete mathematics and combinatorial algorithms can be brought together to solve problems which would normally be considered more properly in the domain of continuous mathematics. We investigate the bit-complexity of discrete solutions to linear PDEs. We show that for a large class of PDEs, the solution of an N point discretization can be compressed to only a constant number of bits per discretization point, without loss of information or introducing errors beyond discretization error. We show that the bit-complexity of the compressed solution is O(N) for both storage space and the total number of operations. We also show that we can compute the compressed solution by a parallel algorithm using O(log N) time and N/log N bit-serial processors, provided that all the coefficients of the PDE are bounded integers of the magnitude O(1). The best previous bounds on the bit-complexity (for both sequential time and storage space) were at least N log N; furthermore, an order of N log N bit-serial processors were required to support the O(log N) parallel time in the known algorithms. We believe this is the first case where a linear or algebraic system can be provably

* This work was supported by NSF Grant CCr-8805782 and by PSC-CUNY Awards #668541 and #669290

** This work was supported by Air Force Contract AFOSR-87-0386, ONR Contract N00014-87-K-0310, DARPA/ISTO Contract N00014-88-K-0458, ONR Contract DAAL03-88-K-0195, and NASA/CESDIS Subcontract 550-63 NAS 5-30428 URSA.

Preprint of paper appearing in International Colloquium on Automata, Languages, and Programming(ICALP 90), Warwich, England, Springer Lecture Notes in Computer Science 443, pp 612 -625, July 1990.

compressed (i.e., the bit-complexity of storage of the compressed solution is less than the solution size) and also the first case where the use of data compression provably speeds up the time to solve the system (in the compressed form).

1. Introduction

1.1 Motivation

An important area of computation (though not often investigated by theoretical computer scientists) is the approximate solution of partial differential equations (PDEs). Conventional discrete approximations of a PDE reduces it to the solution of sparse linear algebraic systems of equations, which approximate to the actual solution to the PDE. Of course, the amount of the discretization will affect the accuracy of the approximation. In general, in the well-posed systems, if N discrete approximation points have been chosen over a regular (two or three dimensional) grid, and if the PDE is linear with constant coefficients, then the solution to the linear algebraic systems frequently approximates to the solution of the PDE with an error of the order of N^{-g}, for some constant $g>0$. Thus, this approximation solution gives, at each discretization point, the first O(log N) bits of the actual value of the solution to the PDE. Such discretization errors have been very well studied by numerical analysts. This analysis work began in [Courant, Friedrichs, Lewy, 1928] and culminated in the 70's, e.g. in [Strang and Fix, 1973]. The high accuracy solution of large 2- and 3-dimensional PDEs requires the number of discretization points N to grow very large. The computation is often more limited by the storage constraints than by the time constraints (particularly, if it is desired to store the solution in the primary storage memory, without the use of the much slower secondary storage of the conventional machines). It is therefore important to investigate methods that substantially reduce this storage, by compressing the data in the solution. As we will see, this is indeed possible in some important cases and is a surprisingly fundamental property of PDEs.

1.2 The Bit-Complexity Model

Certain sequential machines (such as CRAY) were specifically designed to solve these large linear systems; their processor has the ability to do sequential arithmetic operations very quickly and also has a relatively large amount of primary storage. For such a machine, the arithmetic complexity model has generally been considered to be appropriate. However, a machine such as CRAY is capable of performing a bit-vector operation in one parallel step (for example, it may perform AND, OR or NOT operations on hundreds of bits), so that the parallel bit-complexity of such machines can also be of interest.

On the other hand, fine grained massively parallel machines (such as the CONNECTION machine) have been designed with large numbers of bit-serial processors (requiring relatively long time to execute an arithmetic operation) with very limited memory, which is generally accessed bit-serially. A complexity model for parallel algorithms must take into account both the bit-serial nature of the processors and the limited memory constraints; in particular we feel that the parallel complexity is most reasonably measured by the bit-complexity. In this model, we assume that each memory cell holds only one bit, and each processor can do a single bit-operation per step.

1.3 Previous Solutions of PDEs

The solution of the linear algebraic systems approximating PDEs can be computed by means of a number of well known and now classical algorithms. For example, for a large class of PDEs, we may apply standard linearly convergent iterative solution algorithms, such as Gauss-Siedel's, and find the solutions to the auxiliary linear systems up to the maximum accuracy of O(log N) bits at N points in O(log N) iterations, using N processors. However, each such an iteration generally requires an arithmetic operation over the O(log N)-bit numbers and hence requires at least O(log N) bit-operations per point. Thus the total work is O(N log N) arithmetic operations or $O(N \log_2 N)$ bit-operations. Various multigrid methods were first proposed by Fedorenko and Bakhvalov

in the 1960's, and then by Astrakhantzev and Brandt in the early 1970's for the solution of these linear systems approximating PDEs (see [Astrakhantzev 71], [Bakhvalov 66], [Fedorenko 64] and [Brandt 72, 77 and 84]). In [Brandt 77], it was claimed that these multigrid methods required only O(N) arithmetic operations; this was rigorously proved in [Bank, Dupont 81] (also, see [Hackbusch 77, 82 and 85], [McCormick 86] and [McCormick, Trottenberg 83]). (Actually, the multigrid methods are effective even in many cases where the classical iterative algorithms converge too slowly.) The works of [Brandt 80], [Frederickson, McBryan 87], [Chan, Saad, Schultz 86], [McBryan, Van de Velde 85, 86] all describe parallel algorithms that take O(log N) arithmetic steps using N processors, and thus use an order of N log N arithmetic operations, which is off by a factor of log N from the optimum. It follows that the known multigrid methods require a total of O(N log N) bit-operations, and at least an order N log N bit-serial processors to support an order of log N time. The bit-operation bound appears to be optimal since the binary representation of the solution occupies a total of an order of N log N bits.

1.4 Our Results

Our main goal is a rigorous study of the bit-complexity of these linear algebraic systems approximating linear PDEs. In spite of the lack of theoretical investigation into this area, we feel that the problems are fundamental in nature. In this paper we consider a class of what we call *pseudo regular* PDEs, which includes a large class of well-posed linear PDEs with constant coefficients (see the formal definitions in Section 2). We show some surprising properties of the linear algebraic systems approximating to such PDEs:

(1) their solutions can be significantly compressed to O(1) bits per solution point (which, by the factor of log N, improves the previous storage requirements) by a data structure we call the *Compact Multigrid Data Structure*. (We do not know of any previous provable results for data compression of solutions of any type linear systems or of any other algebraic systems);

(2) the compressed solutions can be very efficiently computed (both sequentially and in parallel) by an algorithm which we also call *Compact Multigrid.* The bit-complexity of our compressed solution algorithms is O(log N) time, N/log N bit-serial processors and a total of O(N) bit-operations, which is optimum since the size of the compressed solution is of the order of N. This is by the factor of log N improvement of the bounds on both sequential time and storage space of the known algorithms and by the factor of $\log^2 N$ decrease of the number of bit-serial processors supporting O(log N) time.

Note that already the log N factor is significant for even relatively small problems; for example, this factor is 10 or more for problems of size $N > 1000$, such as the 3-dimensional grid of size 10 x 10 x 10.

A bit-serial data communication required by our compact multigrid algorithm happens to be what is known as a pyramid network, consisting of a sequence of grids $L_0, L_1, \ldots, L_k$, where the grid L_i has 2^{di} points and where each node of the i–th grid is connected to its 2d neighbors in the current grid and also to the corresponding nodes of the (i+1)–th and (i–1)–th grids. It is well known that such a pyramid network can be compactly mapped to a hypercube network with the same number of nodes (within the factor of 2) such that each edge of the pyramid has a corresponding edge of a hypercube; therefore, our compact multigrid algorithm can be efficiently implemented on a hypercube network with the same complexity (within a factor of 2).

The restrictions needed for the property (1) to hold are the mild and customary bounds (2.3), (2.4) below on the discretization and extrapolation errors of the PDE; we call them the weak pseudo regularity assumptions, but even the strong pseudo regularity assumptions, required for the property (2) to hold, are still satisfied for a large class of linear PDEs. Specifically, the strong pseudo regularity includes the requirement that an iterative algorithm (such as multigrid or SSOR) for solving the auxiliary linear systems over all the grids use a constant number of steps on each grid in order to decrease the

approximation error norms linearly with the same rate for all the grids, and we also require that the linear PDEs have constant coefficients.

Given an N point solution, the compression can be done in O(log N) time using N bit-serial processors, for a total of O(N log N) amount of work, which is optimal since the input solution is of size O(N log N). A very simple decompression algorithm requires only O(log N) sequential bit-operations to access the full precision (of O(log N) bits) solution value at any discretization point.

The compressed optimum solution can be stored and can be decompressed only when its values need to be output. In many practical applications, the solutions need not be decompressed. For example, in the solution of the time dependent PDEs, the most customary solution methods perform at a discrete sequence of, say, T time steps. In each time step, a PDE is approximately solved, using an N point discretization of the PDE fixed at that time value and using the approximate solution at the previous time step (in the compressed form) as an initial approximation to the current solution. Thus the solutions at these time steps need not be decompressed, except for the solution at the final time step. Our total bit-complexity in this case, including decompression of the final solution and T calls for compact multigrid, would be $O(N(T + \log N))$ bit-operations (requiring $O(T \log N + \log^2 N)$ time using N/log N bit-serial processors). Here we need the strong pseudo regularity and, in particular, the linear convergence assumption; if it holds initially, we shall preserve it by using sufficiently small time steps.

1.5 Organization of Paper

We will specify our compression techniques for PDEs in sections 2 through 5. We will simplify our presentation, assuming, in particular, the simple lattice grids, although our results hold for more general discretization sets. In section 6 we specify some further extensions of our results, in particular, to more general discretization sets. In Appendix A we add a method for saving the storage space in the parallel solution of a general well-conditioned algebraic linear system.

2. Some Definitions and Assumptions

To simplify our presentation, we will study the linear PDEs on the unit d-dimensional cube, discretized over a family of d-dimensional lattices $L_0, L_1, \ldots, L_k$, where each point of L_j lies at the distance $h_j = 2^{-j}$ from its 2d nearest neighbors, so that there are exactly $|L_j| = N_j = 2^{dj}$ points in L_j for $j = 0,1,\ldots,k$, and the overall number of points equals $N_k = N = 2^{dk}$, where $k = (\log_2 N)/d$, provided that we identify the boundary points whose coordinates only differ by 0 or 1 from each other. (On the actual discretization grids for PDEs, all the boundary points are distinct, and so the grids contain slightly more than N_j points.)

Let $u(x)$ for $j = 1,2,\ldots,k$ denote the solution to the PDE, and let $u_j(x)$ denote the solution to

$$D_j u_j(x) = b_j,\ x \in L_j, \tag{2.1}$$

which is the linear system of the difference equations generated by the discretization of the PDE over the lattice L_j, respectively, so that $\Delta_j(x) = u(x) - u_j(x)$ for $x \in L_j$ denotes the discretization error function on L_j for $j = 1,\ldots,k$. Let $u_0(x) = 0$, and let $\hat{u}_{j-1}(x)$ for $j = 1,2,\ldots,k$ denote an extrapolation of $u_{j-1}(x)$ from L_{j-1} to L_j, obtained, say, by averaging the values of $u_{j-1}(x)$ at the appropriate points y of L_{j-1} lying near x. (More customary names in the multigrid literature are prolongation and interpolation for what we call extrapolation and averaging.) Then

$$u_j(x) = \hat{u}_{j-1}(x) + e_j(x),\ x \in L_j,\ j = 1,\ldots,k, \tag{2.2}$$

where $e_j(x)$ denotes the extrapolation error on L_j.

We will assume that the discretization and extrapolation errors satisfy the two following bounds, which we will call the assumptions of *the weak pseudo regularity* of the PDE:

$$|\Delta_j(x)| \le 2^{c-\alpha j}, \tag{2.3}$$

$$|e_j(x)| \le 2^{c-\alpha j} \tag{2.4}$$

for $x \in L_j$, $j = 1,..., k$, and for fixed $c \geq 0$ and $\alpha \geq 1$. The assumption (2.3) holds for a wide class of PDEs (see, for instance, [Ames, 77], [Lapidus and Pinder 82]), including the well posed constant coefficient linear PDEs, as well as many nonlinear PDEs. Such an assumption is routinely made in the analysis of the multigrid methods (e.g. [Brandt, 72,77, 84], [Bank and Dupont, 81]); in particular, the auxiliary grid problems are said to be "solved to the level of truncation" defined by (2.3) (see [McCormick, 87], p. 26).

As a part of the weak pseudo regularity assumption, let us further assume that $| u_j(x) | \leq 1$ for $x \in L_j$ and for all j and that every $e_j(x)$ is represented with (that is, rounded-off to) α binary bits (digits).

Let us show that the bound (2.3) implies the bound (2.4). First rewrite the bound (2.3) as follows:

$$| \Delta_j(x_j) | \leq a\, h_j^{\gamma},\ x_j \text{ member } L_j, \tag{2.5}$$

where a and γ are positive constants, and h_j is the length of a side of the mesh L_j, so that $h_{j-1} = 2h_j$ for the lattices L_j we have chosen. Let $\gamma < 1$, x_{j-1} member L_{j-1} subset L_j, x_j member L_j, and $| x_j - x_{j-1} | = h_j$. Then we deduce from (2.5) that for some Θ, $0 \leq \Theta \leq 1$, $| e_j(x_j) | = | u_j(x_j) - u_{j-1}(x_{j-1}) | \leq | u(x_j) - u_j(x_j) | + | u(x_j) - u_{j-1}(x_j) | + | u(x_j) - u(x_{j-1}) | \leq a\, h_j^{\gamma} + a\, h_{j-1}^{\gamma} + | u'(x_j + \Theta h_j) | h_j \leq a^*\, h_j^{\gamma}$, that is,

$$| e_j(x_j) | \leq a^* h_j^{\gamma} \tag{2.6}$$

where $a^* = a(1+(h_{j-1}/ h_j)^{\gamma}) + | u'(x_j+\Theta h_j) | h_j^{1-\gamma} \leq 3a + \max | u'(x) | h_j^{1-\gamma}$. The bounds (2.5) and (2.6) turn into the bounds (2.3) and (2.4) if we set $c = \log_2 a^*$, $\alpha = -\gamma (\log_2 h_j)/j$.

Remark. We may replace the bounds (2.3), (2.4) and $| u_j(x) | \leq 1$ by the bounds

$$| \, | \Delta_j(x) | \, | \leq 2^{c-\alpha j} \, | \, | u_j(x) | \, |, \ x \text{ member } L_j,$$
$$| \, | e_j(x) | \, | \leq 2^{c-\alpha j} \, | \, | u_j(x) | \, |, \ x \text{ member } L_j,$$

for a fixed victor norm, provided that $\Delta_j(x)$, $e_j(x)$, $u_j(x)$ for x member L_j are considered

as N-dimensional vectors. This modification would not change our resulting estimates for the complexity of the Compact Multigrid.

In Section 4, we will assume the *pseudo regularity,* which means the weak pseudo regularity together with the assumption that the extrapolation from L_{j-1} to L_j only requires O(1) time using N_j bit-serial processors (which is the case for the extrapolation by averaging).

In Section 5, we will assume the *strong pseudo regularity,* that is, in addition to the pseudo regularity, we will assume that

1) a fixed iterative algorithm (such as Jacobi, Gauss-Seidel, SSOR or a multigrid algorithm) for linear systems with matrices D_j uses O(1) multiplications of submatrices of D_j by vectors for every j, in order to linearly decrease, by the factor independent of j and N, the norm of the error of the approximation to the solution $u_j(x)$ of the system (2.1);

2) the entries of the matrices D_j for all j, as well as the components of b_j, are integers having magnitudes O(1), which holds if we consider the constant coefficient PDEs.

3. Compression of the Output Data

Now assume the weak pseudo regularity relations and compress approximations to all the N values of $u_k(x)$ on L_k within absolute errors of at most $2^{c-\alpha k}$, so as to economize on the storage space required.

For the straightforward fixed point binary representation of these values of $u_k(x)$, we generally need $N \lceil \alpha k - c \rceil$ binary bits.

As an alternative, let us store $u_k(x)$ on L_k in the compressed form by recursively approximating within $2^{c-\alpha j-\alpha}$ to the fixed point binary values $e_j(x)$ for $x \in L_j$, $j = 1,...,k$. The storage space of $2^d \lceil \alpha - c \rceil + \alpha(N_2+N_3+...+N_k) < 2^d \lceil \alpha - c \rceil + 2\alpha N = O(N)$ binary bits suffices for this compressed information, which means saving roughly the

factor of $k = \log_2 N$ binary bits against the straightforward representation.

It is convenient to assume the fixed point binary representation in order to estimate and to compare the numbers of binary bits used in both representations of the output, but shifting to the floating point representation would not actually require to increase these estimates.

4. Recovery of the Output from the Compressed Data

In this section, we will assume the pseudo regularity. To recover $u_k(x)$ on L_k from the compressed information given by $e_j(x)$ on L_j for $j = 1,...,k$, we start with $u_0(x) = 0$ for $x \in L_0$ and recursively, for $j = 1,...,k$, compute the values

a) $\hat{u}_{j-1}(x)$ on L_j, by extrapolation of $u_{j-1}(x)$ from L_{j-1} to L_j, and then

b) $u_j(x)$ on L_j, by applying the equations (2.2).

Both stages a) and b) are performed within the precision $2^{c-\alpha j-\alpha}$, so that the stage b) amounts to appending α binary bits of $e_j(x)$ to the available string of binary bits in the fixed point binary representation of $\hat{u}_{j-1}(x)$ for each $x \in L_j$, and the stage a) amounts to scanning the values of $u_{j-1}(x)$ on L_{j-1} and to summation of few β–bit binary numbers (where, say $\beta = O(\alpha)$) defined by the β least significant binary bits in the representation of $u_{j-1}(x)$ for appropriate x from L_{j-1}. Since $\sum_j N_j = O(N)$, the computational complexity estimates for stages a) and b) stay within the bounds stated in the introduction.

5. Computing the Compressed Solution by Compact Multigrid.

In this section, we will assume the strong pseudo regularity of the PDE. The time complexity of computing the compressed data structure is dominated by the time required to obtain the solutions $e_j(x)$ to the linear systems of equations over L_j for $j = 1,...,k$:

$$D_j e_j(x) = r_j(x), \quad x \in L_j, \tag{5.1}$$

where

$$r_j(x) = b_j - D_j\hat{u}_{j-1}(x),\ x \in L_j, \tag{5.2}$$

and the matrices D_j and the vector b_j are from the linear systems (2.1).

We will follow the routine of multigrid methods (cf. [Frederickson, McBryan, 87]) and will evaluate the vectors e_j recursively for $j = 1,...,k$. Initially, we let $u_0(x) = 0$, and at stage j, we successively compute:

a) $\hat{u}_{j-1}(x)$ (by extrapolation of $u_{j-1}(x)$ from L_{j-1} to L_j),

b) $r_j(x)$ (by using the equations (5.2)),

c) $e_j(x)$ (by solving the linear system (5.1)"to the level of truncation"),

d) $u_j(x)$ (by using the equations (2.2), as in section 4).

We may then restrict $u_{j+1}(x)$ to $u_j(x)$ for j=k–1, k–2, ..., 2 (at this stage we do not use smoothing iterations) and then recursively repeat such a V-cycle.

Part 1) of the strong pseudo regularity assumption means that the errors of the approximations to $u_j(x)$ decrease by a constant factor independent of j and N when stages a)-d) are repeated once for all j even if only O(1) iteration steps are used at stage c) for solving linear systems (5.1) for every j. Such convergence results have been proven for the customary multigrid algorithms applied to a wide class of PDEs (see [Bank, Dupont 81], [Hackbusch 77, 82 and 85], [McCormick 86], [McCormick, Trottenberg 83], [Frederickson, McBryan 87a]).

Let us estimate the time complexity of these computations, dominated by the time needed for solving the linear systems (5.1).

The size $|L_j| = 2^{dj}$ of the linear system (5.1) increases by 2^d times as j grows by 1. Even if we assume that the solution time for the system (5.1) is linear in $|L_j|$, the overall solution time for all the k such systems in terms of the number of arithmetic operations involved is less than $1/(1-2^{-d})$ times the solution time for the single system (2.1) for $j = k$, which gives us the uncompressed output values $u_k(x)$ for $x \in L_k$. The

bit-operation count is even more favorable to the solution of the systems (5.1) for all j, as opposed to the single system (2.1) for $j = k$, because the output values $e_j(x)$, satisfying the systems (5.1), are sought with the lower precision of α binary bits.

Furthermore, we solve the linear systems (5.1) by iterative methods where each step is essentially reduced to a constant number (say, one or two) multiplications of a matrix D_j or its submatrices by vectors. Due to the linear convergence assumption we made, a constant number of iterations suffices at each step j in order to compute the α desired binary bits of $e_j(x)$.

The computational cost of multiplication of D_j by a vector is $O(N_j)$ arithmetic operations for a sparse and structured discretization matrix D_j (having O(1) nonzero entries in each row). Moreover, parallel acceleration to the parallel time bound O(1) is possible using N_j processors (for we deal with a matrix-by-vector multiplications, and the matrix has O(1) nonzero entries per row). Thus we arrive at Proposition 1, whose processor bound follows similarly to the proof of Proposition 2 below.

Proposition 1. O(log N) *parallel arithmetic steps and* N/ log N *processors suffice to compute* $e_j(x)$ *for all* j, *that is, to compute the smooth compressed solution to a strongly regular PDE discretized over the lattice* L_k.

Furthermore, we only need O(1) binary bits in order to represent $e_j(x)$ for every j. Since D_j has only O(1) nonzero entries per row and since these entries are integers having magnitudes O(1), it suffices to use O(1) bits to represent $r_j(x)$. Thus, we will perform all the arithmetic operations with O(1)–bit operands and will arrive at Proposition 2.

Proposition 2. O(log N) *steps,* N/ log N *processors and* O(N) *storage space under the Boolean model of computation suffice in order to compute the compressed solution to a strongly pseudo regular PDE by using the Compact Multigrid algorithm.*

Proof. As described above, our algorithm has stages j=1,....., k = (log N)/d, where at stage j we require O(1) time for each of the $N_j = 2^{dj}$ bit-serial processors. Thus our parallel algorithm (if naively implemented), appears to take O(log N) time using N bit-serial processors. However, the first (log N)/d – log log N stages only require N/ log N bit-serial processors. Thus, at each of the last log log N stages j=(log N)/d–log log N+1,...,(log N)/d, we will slow down the computation to the time $O(\frac{2^{dj}\log N}{N})$, using only N/ log N bit-serial processors. The overall time of our resulting parallel algorithm is then still O(log N).

6. Extensions of the Results. Our results can be immediately extended to the case of more general sequences of the sets $S_0, S_1,..., S_k$ of the discretization of the PDEs, provided that each set S_j consists of $c_j\sigma^j$ points (where $0 < c < c_j < c^*$, $\sigma > 1$, c, c^* and σ are constants), and that the pseudo regularity assumptions are respectively extended to the case of the sets S_j. We also need to assume a constant degree bound 2d for all the discretization points, that is, each of them is supposed to have at most 2d neighbors: this will imply that each equation of the associated linear algebraic system has at most 2d+1 nonzero coefficients.

Finally, the presented approach can be further extended to some nonlinear PDEs, as long as our assumptions (such as (2.3) and (2.4)) hold and as long as dealing with nonlinear systems of difference equations replacing the linear systems (2.1) remains relatively inexpensive.

References

W.F. Ames, *Numerical Methods for Partial Differential Equations,* Academic Press, N.Y., 1977.

G.P. Astrakhantzev, An Iterative Method of Solving Elliptic Net Problem, *Z. Vycisl. Mat.i Mat. Fiz.,* (in Russian), Vol. 11, pp. 439-448, 1971.

K.E. Atkinson, *An Introduction to Numerical Analysis,* Wiley, 1978.

N.S. Bakhvalov, On the Convergence of a Relaxation Method under Natural Constraints on an Elliptic Operator, *Z. Vycisl. Mat.i Mat. Fiz.,* (in Russian), Vol. 6, pp. 861-883, 1966.

R. Bank and Donald J. Rose, "Analysis of a Multilevel Iterative Method for Nonlinear Finite Element Equations", *Mathematics of Computation,* Vol. 39, No 160, pp 453-465, 1982.

R. Bank and T. Dupont, "An optimal order process for solving finite element equations", *Mathematics of Computation.,* Vol. 36, pp. 35-51, 1981.

R. Bank and A. Sherman, "Algorithmic aspects of the Multi-level solution of finity element equations", in *Sparse Matrix Proceedings,* 1978, I.S. Duff and G.W. Stewart, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

A. Brandt, "Multi-Grid Solvers on Parallel Computers", ICASE Technical Report 80-23, *NASA Langley Research Center,* Hampton, VA, 1980.

A. Brandt, "Multi-level adaptive technique (MLAT) for fast numerical solutions to boundary value problems", *Proc. 3rd Int. Conf. Numerical Methods in Fluid Mechanics,* Paris, France, 1972; *Lecture Notes in Physics,* Vol. 18, pp. 82-89, Springer-Verlag, Berlin, W. Germany, 1984.

A. Brandt, "Multi-level adaptive solutions to boundary value problems", *Mathematics of Computation,* Vol. 31, pp. 333-390, 1977.

A. Brandt, *Multigrid Techniques: 1984 Guide, with Applications to Fluid Dynamics.* Available as GMD Studien Nr. 85, GMD-AIW, Postfach 1240, D-5205, St. Augustin 1, W. Germany, 1984.

T.F. Chan, Y. Saad, and M.H. Schultz, "Solving elliptic partial differential equations on hypercubes", *Hypercube Multiprocessors 1986,* SIAM, Philadelphia, PA, 1986.

R. Courant, K.O. Friedrichs, H. Lewy, "Uber die partiellen Differenzengleich-ungen der mathematischen Physik", *Math. Ann.,* Vol. 100, pp. 32-74, 1928.

R.P. Fedorenko, The Speed of Convergence of One Iteration Process, *Z. Vycisl. Mat. i Mat. Fiz. (in Russian),* Vol. 4, pp. 559-663, 1964.

G.E. Forsythe, W.R. Wasow, *Finite Difference Methods for Partial Differential Equations,* John Wiley & Sons, Inc., New York, 1960.

P.O. Frederickson and O. McBryan, "Parallel Superconvergent Multigrid", *Cornell Theory Center,* Preprint, May 1987.

P.O. Frederickson and O.M. McBryan, "Superconvergent Multigrid Methods", *Cornell Theory Center,* Preprint, May 1987a.

D. Gannon and J. van Rosendale, "Highly Parallel Multi-Grid Solvers for Elliptic PDSs: An Experimental Analysis", ICASE Technical Report 82-36, *NASA Langley Research Center,* Hampton, VA, 1982.

G.H. Golub and C.F. van Loan, *Matrix Computation,* Johns Hopkins Univ. Press, Baltimore, MD, 1983.

W. Hackbusch, "Convergence of multi-grid iterations applied to difference equations", *Mathematics of Computation,* Vol. 34, pp. 425-440, 1980.

W. Hackbusch, *Multigrid Methods and Applications,* Springer-Verlag, 1985.

W. Hackbusch, On the convergence of multi-grid iteration applied to finite element equations, Report 77-8, *Universität zu Köln,* July, 1977.

W. Hackbusch and U. Trottenberg, eds., *Multigrid Methods, Lecture Notes in Math.,* Vol. 960, Springer-Verlag, 1982.

D. Jespersen, "Multigrid methods for partial differential equations", in *Studies in Numerical Analysis,* Vol. 24, *Studies in Mathematics,* Math. Assoc. of America, 1984.

L. Lapidus and G.F. Pinder, *Numerical Solution of Partial Differential Equations in Science and Engineering,* Wiley, 1982.

L. Mansfield, "On the solution of nonlinear finite element systems", *SIAM J. Numer. Anal.,* Vol. 17, 1980, pp. 752-765.

O. McBryan and and E. Van de Velde, "Parallel Algorithms for Elliptic Equations", *Commun. Pure and Appl. Math.,* Vol. 38, pp. 769-795, 1985.

O. McBryan and E. Van de Velde, "The Multigrid Method on Parallel Processors", in *Multigrid Methods II,* ed. W. Hackbusch and U. Trottenberg, Lecture Notes in Mathematics, Vol. 1228, Springer-Verlag, Berlin, Heidelberg, 1986.

S. McCormick, editor, *Multigrid Methods,* Volume 3 of SIAM Frontiers Series, SIAM, Philadelphia, 1987.

S. McCormick, (ed.), Proceeding of the 2nd Copper Mountain Multigrid Conference, *Appl. Math. Comp.,* Vol. 19, pp. 1-372 (special issue), 1986.

S. McCormick and U. Trottenberg (eds.), Multigrid Methods, *Appl. Math. Comp.,* Vol. 13, pp. 213-474 (special issue), 1983.

D.J. Paddon and H. Holstein, eds., *Multigrid Methods for Integral and Differential Equations,* Clarendon Press, Oxford, 1985.

G. Strang and G. Fix, *An Analysis of the Finite Element Method,* Prentice-Hall, Englewood Cliffs, NJ, 1973.

P.J. Van der Houven, *Finite Difference Methods,* Mathematisch Centrum, Amsterdam, 1968.

R. Varga. *Matrix Iterative Analysis,* Prentice-Hall, Englewood Cliffs, NJ, 1962.

D. Young, *Iterative Solution of Large Linear Systems,* Academic Press, New York, 1971.

Appendix A. Space Efficient Parallel Solution of a Well-Conditioned Linear Algebraic System of Equations

In this appendix, we will give a space efficient methodology (but not a data compression technique), which we also suggest for reducing local storage in parallel solution of a general well-conditioned linear algebraic system of equations. The idea is to subdivide the original problem into the problem of parallel solution of several linear systems, with a substantial decrease of temporary storage space for the transition to each of the new linear systems.

Formally, we will proceed as follows. Given a nonsingular linear system of equations

$$Ax = b, \tag{A.1}$$

where the components of the vector b are numbers between –1 and 1 having αg binary bits in their fixed point representation, we will subdivide each such a component into g segments of α-bit binary numbers and represent b as the sum $b = \sum_{i=1}^{g} b_i$ where component j of the vector b_i for every j is defined by the i-th segment of the respective component j of b. Denote $\beta = \alpha - \lceil \log_2 \mathrm{cond}\, A \rceil$, $\mathrm{cond}\, A = \| A \| \, \| A^{-1} \|$,

$$x_i = A^{-1} b_i \tag{A.2}$$

and let x_i^* be the (hi)-bit approximation to x_i within absolute error 2^{-hi-1} in each component. According to the well-known error estimate,

$$\| x - \sum_{j=1}^{i} x_j \| \, / \, \| x \| \leq (\| b - \sum_{j=1}^{i} b_j \| \, / \, \| b \|) \, \mathrm{cond}\, A$$

(see [Atkinson 78], [Golub and van Loan 83]), and it follows that $\| x - \sum_{i=1}^{d} x_i^* \| \leq 2^{-\beta g}$.

Thus, we reduced the solution of the linear system (A.1) within the error norm $2^{-\beta g}$ to the solution of g linear systems (A.2) within $2^{-\beta i}$ for $i = 1, \ldots, g$. The main advantage of this reduction is that for the system (A.2) for each i we only need to store the α binary bits of each component of the input vector b_i and β binary bits of each component of the output vector x_i^*, whereas the storage space for the input and output of

the original system (A.1) is by g times greater than that.