TEMPORAL DOMAIN PARALLELISM: DOES IT WORK?*

ASHISH DESHPANDE†, SACHIT MALHOTRA‡, CRAIG C. DOUGLAS§, AND MARTIN H. SCHULTZ¶

Abstract. Time dependent partial differential equations are often solved using algorithms which parallelize the solution process in the spatial domain. However, as the number of processors increases, the parallel efficiency is limited by the increasing communication/computation ratio. Recently, several researchers have proposed algorithms incorporating temporal domain parallelism in order to increase efficiency. In this paper we discuss a class of such algorithms and show that this approach is not normally useful.

1. Introduction. We investigate a parallel algorithm for the numerical solution of linear, time-dependent partial differential equations (PDEs) of the form

$$\begin{cases} \frac{\partial u}{\partial t} + \mathcal{L}u = f, & x \in \Omega, \quad 0 < t < T, \\ \mathcal{B}(u) = u_b(t), & x \in \partial\Omega, \quad 0 < t < T, \\ u(x,0) = u_0(x), & x \in \Omega \end{cases}$$

where $\mathcal{L}$ is a linear elliptic spatial operator, $\mathcal{B}$ is the boundary operator, and Ω is a spatial domain with boundary $\partial\Omega$. For simplicity of presentation we choose the one dimensional problem $\Omega = (0,1)$ and Dirichlet boundary conditions $u(0,t) = u(1,t) = 0$. Our model problem in this paper will be the one dimensional heat equation.

Implicit time stepping schemes, coupled with finite difference approximations of the spatial derivatives, lead to linear systems at each time step t_k, of the form,

$$Au^k = Bu^{k-1} + b, \qquad k = 1, 2, \ldots n, \tag{1}$$

where A and B are $m \times m$ matrices (m is the number of grid points in the interior of the domain) whose elements depend on $\mathcal{L}$ and $\mathcal{B}$ and u^k is an $m-$vector containing function values u at all of the grid points at the $k'th$ time step. Starting from u^0,

* This work was supported in part by ONR Grant # N0014-91-J-1576 and by an IBM/Yale joint study.

† Yale Center for Parallel Supercomputing, Department of Computer Science, Yale University, New Haven, CT 06520. E–mail: *deshpande-ashish@cs.yale.edu.*

‡ Yale Center for Parallel Supercomputing, Department of Computer Science, Yale University, New Haven, CT 06520. E–mail: *malhotra-sachit@cs.yale.edu.*

§ IBM Research Division, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598. This author was on sabbatical at Yale University while this work was carried out. E–mail: *cdouglas@na-net.ornl.gov.*

¶ Yale Center for Parallel Supercomputing, Department of Computer Science, Yale University, New Haven, CT 06520. E–mail: *schultz-martin@cs.yale.edu.*

which is known from the initial condition, we can use an iterative algorithm such as Jacobi, Gauss-Seidel, or SOR to solve (1) sequentially for each time step:

$$u_i^k = Tu_{i-1}^k + c, \qquad i = 1, 2, \ldots, \qquad k = 1, 2, \ldots n,$$

where T is the iteration matrix and c is a vector of known values.

Usually, the entire process is spatially parallelized by splitting the domain Ω into subdomains and distributing problems on the subdomains to multiple processors [2]. At each iteration, the processors need to exchange boundary information with processors holding adjacent subdomains. As the number of processors increases, the communication/computation ratio increases making the parallel efficiency decrease. In an effort to forestall this and to allow increasing numbers of processors to be used effectively, a number of researchers have suggested algorithms which introduce time domain parallelism as well as space domain parallelism [1, 4, 5, 6].

It is claimed in [1] and [6] that a parallel scheme based on time domain parallelism will converge in exactly the same number of iterations as the sequential scheme since the two schemes have iteration matrices with equal spectral radii. In this paper, we examine this claim. We show that, for model time domain parallel schemes, the iteration matrices are defective and hence, while the spectral radius of the iteration matrix governs the asymptotic rate of convergence, it does not determine the exact number of iterations required. Hence, one cannot claim that the two schemes will converge in exactly the same number of iterations. Using a 2-norm argument, we show that the number of iterations required for convergence of the new scheme increases with the number of parallel time steps. Even if we ignore the cost of communication, the new scheme has a very low efficiency for real problems. In this paper, we provide an analysis of a class of these algorithms as well as computational experiments which demonstrate that this approach is not normally useful.

2. The Temporal Method. In [6] and [1], an iterative scheme is described for introducing temporal domain parallelism by solving the linear systems at different time steps simultaneously. The central idea is to assemble n steps of (1) into the form

$$(2) \qquad Gu \equiv \begin{pmatrix} A & & & & \\ -B & A & & & \\ & -B & \ddots & & \\ & & \ddots & A & \\ & & & -B & A \end{pmatrix} \begin{pmatrix} u^1 \\ u^2 \\ \vdots \\ u^{n-1} \\ u^n \end{pmatrix} = \begin{pmatrix} u^0 + b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix} \equiv b.$$

Equation (2) can be solved in parallel using an iterative scheme such as Jacobi or Gauss-Seidel with the red-black block ordering. If $A = D - L - U$, where D, L and U are diagonal, lower triangular, and upper triangular, then the iteration matrix T_A for solving (1) using Gauss-Seidel is

$$T_A = (D - L)^{-1}U.$$

The iteration matrix T_G for Gauss-Seidel for (2) is

$$T_G = (\bar{D} - \bar{L})^{-1}\bar{U},$$

where $\bar{D}, \bar{L}$ and $\bar{U}$ are the diagonal, lower triangular, and upper triangular parts of G. As was shown in [6], T_G is a block lower triangular matrix with diagonal blocks equal to $(D - L)^{-1}U$. Hence, the eigenvalues of T_G are the collections of all eigenvalues of all the block matrices on the main diagonal. Since all blocks on the main diagonal are equal to T_A, T_G has the same eigenvalues as T_A but with higher multiplicity. This shows that

$$\varrho(T_A) = \varrho(T_G),$$

where $\varrho(T)$ is the spectral radius of T. Hence, both iterations *appear* to converge at the same rate.

Based on the "proof" sketched above that iterative schemes (1) and (2) converge in exactly the same number of iterations, both [1] and [6] go on to demonstrate that the parallel algorithm involving both spatial and temporal dimension parallelism is far superior in terms of speedup and communication/computation ratio over algorithms employing only spatial dimension parallelism. However, in the following sections, we demonstrate that this is not valid.

3. Convergence of Iterative Methods. Any stationary linear iteration scheme can be written in the form

$$u_{k+1} = Tu_k + c$$

where T is the iteration matrix and c is a vector of known values. The error e_k in the k^{th} approximation to the solution is given by

$$e_k = T^k e_0.$$

Hence, the sequence of iterates $u_1, u_2, \ldots, u_k, \ldots$ will converge to the true solution as $k \to \infty$ if and only if

$$\lim_{k \to \infty} T^k = 0$$

since u_0, and hence e_0, is arbitrary. If the $m \times m$ matrix T has m linearly independent eigenvectors $v_s, s = 1, \ldots, m$, then it follows that

$$e_k = \sum_{s=1}^{m} c_s \lambda_s^k v_s, \tag{3}$$

where λ_s is the eigenvalue corresponding to v_s. Thus e_k will tend to 0 for an arbitrary e_0 if and only if $| \lambda_s | < 1$ for all s, i.e., if and only if $\varrho(T) < 1$, and the convergence rate behaves like ϱ^k.

While the preceding result is true in general (see [3]), this argument holds only if the matrix T has m linearly independent eigenvectors. If T is defective, i.e., lacking eigenvectors, then we may not be able to express e_0 as a linear combination of the eigenvectors and (3) may not hold. In this case, we cannot claim that the convergence rate behaves like ϱ^k except in an asymptotic sense, which may not yield any useful information. For example, consider the $m \times m$ matrix

$$T = \begin{pmatrix} 0 & & & \\ 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{pmatrix}$$

which is defective and whose eigenvalues are all equal to 0. Hence, $\varrho^k = 0$ for all $k \geq 1$. However, $\|T^k\|_2 = 1$ for $0 \leq k < m$ and $T^m = 0$. Hence, if the iteration matrix is defective it is not enough to look only at its spectral radius.

We show in Section 4 that the iteration matrix for solving (2) is defective for the block and point Jacobi and Gauss-Seidel methods with the natural ordering. Hence, the spectral radius does not give a true picture of the convergence rate and one must actually look at the norm of the k^{th} power of the iteration matrix, T^k, to determine the number of iterations it takes for the iterative scheme to converge. Since,

$$\|e_k\|_2 = \|T^k e_0\|_2 \leq \|T^k\|_2 \|e_0\|_2$$

and there exists at least one initial error vector e_0 for which the equality holds, we can only assert convergence if the norm of the error is reduced in k iterations by a factor of $\epsilon < 1$, i.e. if

$$\|T^k\|_2 \leq \epsilon < 1, \tag{4}$$

we can determine lower and upper bounds on the number of iterations for convergence by determining k such that (4) holds. We will show that the number of iterations

required to solve (2) is enough to negate any advantages of temporal domain parallelism.

In Section 5, we consider the block Jacobi and Gauss-Seidel methods and show that the 2-norm of T^k is large enough that the increase in the number of iterations required for convergence eliminates any possible speedup due to temporal domain parallelism. We present numerical results to validate this claim.

In Section 6 we address the point Jacobi method. We derive an expression for the 2-norm of T^k and show that the parallel scheme exhibits very low efficiency in most cases. We also present numerical results to validate this claim.

In Section 7 we consider the point Gauss-Seidel method. We describe numerical experiments which show that the parallel scheme using the point Gauss-Seidel method again exhibits very low efficiency contrary to the claims made in [6].

4. Defectiveness of the Iteration Matrix. Consider solving (2) using a block Jacobi iteration scheme, where each block corresponds to one time step. The resulting iteration matrix is

$$T_{BJ} = \begin{pmatrix} 0 & & & \\ A^{-1} & 0 & & \\ & \ddots & \ddots & \\ & & A^{-1} & 0 \end{pmatrix}.$$

T_{BJ} is an $mn \times mn$ matrix, where m is the order of A and n is the number of time steps we are attempting to solve simultaneously. This matrix is defective for any matrix A and all its eigenvalues are 0. In fact, it can be easily shown that T_{BJ} has only the m linearly independent eigenvectors

$$\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ e_1 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ e_2 \end{pmatrix}, \quad \ldots, \quad \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ e_m \end{pmatrix},$$

where $e_1, e_2, \ldots, e_m$ is the m-dimensional canonical basis. A similar argument shows that the iteration matrix, T_{BG}, for block Gauss-Seidel on (2) is also defective.

Let us consider the point Jacobi scheme, whose iteration matrix is

$$T_{PJ} = \begin{pmatrix} D^{-1}(L+U) & & & & \\ D^{-1} & D^{-1}(L+U) & & & \\ & \ddots & \ddots & & \\ & & & D^{-1} & D^{-1}(L+U) \end{pmatrix}.$$

To simplify the presentation, consider the block 2×2 case, which corresponds to the case of two time levels.

LEMMA 4.1. *Let $F \equiv D^{-1}(L+U)$ and $H \equiv D^{-1}$. Then the block 2×2 iteration matrix*

$$T_{PJ} = \begin{pmatrix} F & 0 \\ H & F \end{pmatrix}$$

is defective. *Proof:*T_{PJ} has the same eigenvalues as F, but with higher multiplicity. Let F have order m and eigenvalues $\lambda_1, \ldots, \lambda_m$ with corresponding eigenvectors $v_1, \ldots, v_m$. The eigenvectors of T_{PJ} are given by

$$\begin{pmatrix} F & 0 \\ H & F \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \lambda_i \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

from which it follows that either

$$x_1 = 0 \quad or \quad x_1 = v_i.$$

The choice of $x_1 = 0$ gives m linearly independent eigenvectors

$$\begin{pmatrix} 0 \\ v_1 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ v_2 \end{pmatrix}, \quad \ldots, \quad \begin{pmatrix} 0 \\ v_m \end{pmatrix},$$

and T_{PJ} is thus defective. We now show that the choice of $x_1 = v_i$ leads to a contradiction. In fact, $x_1 = v_i$ gives

$$Hx_1 + Fx_2 = Hv_i + Fx_2 = \lambda_i x_2$$

or

$$(F - \lambda_i I)x_2 = -Hv_i. \tag{5}$$

The matrix $(F - \lambda_i I)$ is singular since λ_i is an eigenvalue of F. The backward Euler scheme results in a linear system whose matrix is tridiagonal with constant diagonal elements $(1+2r)$, where $r \equiv \frac{\Delta t}{(\Delta x)^2}$. Hence, $H \equiv D^{-1} = \alpha I$, where $\alpha \equiv 1/(1+2r) =$

constant, and $F \equiv D^{-1}(L+U)$ is a tridiagonal, symmetric matrix with zero diagonal. Then, (5) becomes

$$(F - \lambda_i I)x_2 = -\alpha v_i$$

which has no solution. Indeed, taking the inner product of both sides with v_i, we get

$$< (F - \lambda_i I)x_2, v_i > \; = \; -\alpha < v_i, v_i > .$$

Since $(F - \lambda_i I)$ is symmetric, this gives

$$< x_2, (F - \lambda_i I)v_i > \; = \; -\alpha < v_i, v_i > .$$

But v_i is an eigenvector of F, so $(F - \lambda_i I)v_i = 0$ and we get

$$0 = -\alpha < v_i, v_i >$$

which is a contradiction. Hence, a solution does not exist and T_{PJ} is a defective matrix. ∎

In general, (5) will have a solution if and only if Hv_i has no projection into the nullspace of $(F - \lambda_i I)$. In practice, this is unlikely.

We state, without proof, the following theorem

THEOREM 4.2. *The iteration matrices for point Jacobi and Gauss-Seidel methods are defective.*

Hence, a spectral radius argument cannot be used to determine the number of iterations for convergence of an iterative scheme to solve (2).

5. Block Iterative Schemes. In this Section, we consider the block Jacobi and Gauss-Seidel methods. We derive lower bounds on $\|T^k\|_2$ and show that the number of iterations k required to satisfy (4) is large enough to negate the advantages of the time parallel method.

5.1. Block Jacobi Iterative Schemes.

5.1.1. Backward Euler. In order to simplify the notation in this section, we restrict our attention to backward Euler differences. In this case, the matrices in (1) are $B = I_m$ and

$$A_{m\times m} = \begin{pmatrix} 1+2r & -r & & \\ -r & 1+2r & \ddots & \\ & \ddots & \ddots & -r \\ & & -r & 1+2r \end{pmatrix},$$

where $r \equiv \frac{(m+1)^2}{n} \equiv \frac{\Delta t}{(\Delta x)^2}$. The number of timesteps being being solved for in parallel is n.

The k^{th} power of the iteration matrix is given by

$$T_{BJ}^k = \begin{pmatrix} 0 & & & & \\ \vdots & 0 & & & \\ A^{-k} & & \ddots & & \\ & \ddots & & 0 & \\ & & A^{-k} & \dots & 0 \end{pmatrix}.$$

Let t_p and t_s be the wall clock times for the parallel and serial algorithms, respectively. In the serial case the time, t_s, for n time steps is ns, where s is the serial time to solve $Ax = b$.

THEOREM 5.1. *The total wall clock time using n processors and ignoring communication time is*

$$t_p \geq \frac{1}{\pi^2} \ln\left(\frac{1}{\epsilon}\right) t_s$$

for the block Jacobi scheme applied to the backward Euler equations. **Proof:** Note that

$$\|T_{BJ}^k\|_2 = \|A^{-k}\|_2 = \frac{1}{\lambda_{min}^k(A)}.$$

A simple analysis shows that

$$\lambda_{min}(A) = 1 + 4r \sin^2\left(\frac{\pi}{2(m+1)}\right) \leq 1 + \frac{\pi^2}{n}.$$

Hence,

$$\|T_{BJ}^k\|_2 \geq [1 + \frac{\pi^2}{n}]^{-k}.$$

To be convergent, we need that

$$\|T_{BJ}^k\|_2 \leq \epsilon < 1.$$

This holds if

$$k \geq n \frac{\ln\left(\frac{1}{\epsilon}\right)}{\pi^2} = O(n).$$

The total work in aggregate in the parallel scheme is

$$W_p = k \cdot work/iteration = kns \geq \frac{sn^2}{\pi^2} \ln\left(\frac{1}{\epsilon}\right).$$

Since $t_s = sn$, the total wall clock time using n processors, *assuming no communication overhead*, is

$$t_p = \frac{W_p}{n} \geq \frac{ns}{\pi^2} \ln\left(\frac{1}{\epsilon}\right) = \frac{\ln\left(\frac{1}{\epsilon}\right)}{\pi^2} t_s. \quad \blacksquare$$

Corollary: Even if we ignore the cost of communication, the total wall clock time to solve the problem using a parallel approach is at best asymptotically the same as the serial time in spite of the larger number of processors used. In particular the parallel algorithm will actually take more wall-clock time than the serial version if $\epsilon \leq 5.17 \times 10^{-5}$.

5.1.2. Crank-Nicolson. We repeat the analysis of § 5.1.1 for the Crank-Nicolson scheme to show that the asymptotic behavior of the scheme is identical. In the Crank-Nicolson case, the matrices in (1) are

$$A_{m\times m} = \begin{pmatrix} 1+r & -r/2 & & \\ -r/2 & \ddots & \ddots & \\ & \ddots & \ddots & -r/2 \\ & & -r/2 & 1+r \end{pmatrix}$$

and

$$B_{m\times m} = \begin{pmatrix} 1-r & r/2 & & \\ r/2 & \ddots & \ddots & \\ & \ddots & \ddots & r/2 \\ & & r/2 & 1-r \end{pmatrix}.$$

The iteration matrix is

$$T_{JC} = \begin{pmatrix} 0 & & & & \\ A^{-1}B & 0 & & & \\ & A^{-1}B & \ddots & & \\ & & \ddots & 0 & \\ & & & A^{-1}B & 0 \end{pmatrix}.$$

THEOREM 5.2. *The total wall clock time for block Jacobi to solve the Crank-Nicolson equations is*

$$t_p \geq \frac{1}{\pi^2} \ln\left(\frac{1}{\epsilon}\right) t_s,$$

ignoring communication overhead. **Proof:** The norm of the iteration matrix is related to the eigenvalues of $A^{-1}B$ in the following manner:

$$\|T_{JC}^k\|_2 = \| \left(A^{-1}B\right)^k \|_2 = \lambda_{max}^k \left(A^{-1}B\right).$$

Since

$$A + B = 2I_m$$

the eigenvectors of A and B are identical. The eigenvalues satisfy

$$\lambda_i(A) + \lambda_i(B) = 2, \qquad i = 1, \ldots, m.$$

The eigenvalues of $A^{-1}B$ are given by

$$\lambda_i(A^{-1}B) = \frac{2 - \lambda_i(A)}{\lambda_i(A)} = \frac{2}{\lambda_i(A)} - 1.$$

As a result,

$$\lambda_{max}(A^{-1}B) = \frac{2}{\lambda_{min}(A)} - 1.$$

The eigenvalues of A are given by

$$\lambda_s(A) = 1 + 2r \sin^2\left(\frac{s\pi}{2(m+1)}\right), \qquad s = 1, \ldots, m.$$

Hence,

$$\lambda_{max}\left(A^{-1}B\right) \geq \frac{1 - \frac{\pi^2}{n}}{1 + \frac{\pi^2}{n}}.$$

By an analysis similar to the proof of Theorem 2, $\|T_{JC}^k\|_2 \leq \epsilon < 1$ if

$$k \geq n \frac{\ln(\frac{1}{\epsilon})}{\pi^2} = O(n). \quad \blacksquare$$

Corollary: The parallel algorithm will actually take more wall-clock time than the serial version if $\epsilon \leq 5.17 \times 10^{-5}$.

5.2. Block Gauss-Seidel Iterative Schemes. We now analyze block Gauss-Seidel methods. We state the formal result showing that the parallel methods never provide an asymptotic speedup and may be slower for sufficiently small time steps.

Before we present the results for this section, we would like to introduce some notation to simplify the presentation. The block matrix $M(A, m, n, k)$ denotes a $mn \times$

mn matrix having the following structure

$$M(A,m,n,k) = \begin{pmatrix} 0 & & & & \\ \vdots & 0 & & & \\ A & & 0 & & \\ & \ddots & & \ddots & \\ & & A & \dots & 0 \end{pmatrix}$$

where A is an $m \times m$ matrix and the $A^{'}$s lie on the k^{th} block sub-diagonal.

THEOREM 5.3. *Let t_s^{BE}, t_p^{BE}, t_s^{CN} and t_p^{CN} be the total wall clock times for the backward Euler and Crank-Nicolson (serial and parallel) variants. The total wall clock times, assuming no communication overhead, for block Gauss-Seidel are given by*

$$t_p^{BE} \geq \frac{1}{2\pi^2} \ln\left(\frac{1}{\epsilon}\right) t_s^{BE}$$

and

$$t_p^{CN} \geq \frac{1}{2\pi^2} \ln\left(\frac{\sqrt{2}}{\epsilon}\right) t_s^{CN}.$$

Proof:

We first consider the backward Euler difference approximation. The scheme suggested in [6] results in the following iteration matrix:

$$T_{GB} = \begin{pmatrix} A & & & \vdots & & & \\ & \ddots & & \vdots & & & \\ & & A & \vdots & & & \\ \dots & \dots & \dots & \vdots & \dots & \dots & \dots \\ -I & & & \vdots & A & & \\ & \ddots & & \vdots & & \ddots & \\ & & -I & \vdots & & & A \end{pmatrix}^{-1} \begin{pmatrix} & & & \vdots & 0 & & \\ & & & \vdots & I & \ddots & \\ & & & \vdots & & I & 0 \\ \dots & \dots & \dots & \vdots & \dots & \dots & \dots \\ & & & \vdots & & & \\ & & & \vdots & & & \\ & & & \vdots & & & \end{pmatrix},$$

or using the notation introduced at the beginnning of the section,

$$T_{GB} = \begin{pmatrix} 0 & M(A^{-1}, m, \frac{n}{2}, 1) \\ 0 & M(A^{-2}, m, \frac{n}{2}, 1) \end{pmatrix},$$

where A is given by

$$A_{m\times m} = \begin{pmatrix} 1+2r & -r & & \\ -r & 1+2r & \ddots & \\ & \ddots & \ddots & -r \\ & & -r & 1+2r \end{pmatrix}.$$

The rate of convergence is governed by the norm of the T_{GB}^k. Simple algebraic manipulations show that

$$T_{GB}^k = \begin{pmatrix} 0 & M(A^{-2k+1}, m, \frac{n}{2}, k) \\ 0 & M(A^{-2k}, m, \frac{n}{2}, k) \end{pmatrix}.$$

The norm of the iteration matrix is given by

$$\left\|T_{GB}^k\right\|_2 = \left(\frac{1}{\lambda_{min}(A)}\right)^{2k}$$

where

$$\lambda_{min}(A) = 1 + 4r\sin^2\left(\frac{\pi}{2(m+1)}\right) \leq 1 + \frac{\pi^2}{n}.$$

As a result

$$\left\|T_{GB}^k\right\|_2 \geq \left(1+\frac{\pi^2}{n}\right)^{-2k}.$$

To achieve a reduction in norm of the error by a factor of ϵ we require

$$\left\|T_{GB}^k\right\|_2 \leq \epsilon$$

or

$$\left(1+\frac{\pi^2}{n}\right)^{-2k} \leq \epsilon$$

which means

$$k \geq n\frac{\ln\left(\frac{1}{\epsilon}\right)}{2\pi^2} = O(n)$$

which gives us the required error bound, assuming $\epsilon < 1$.

Second, we consider the Crank-Nicolson difference approximation. An analysis similar to the one for the backward Euler case with A replaced by $M^{-1}N$ yields the desired bounds. Here M and N are given by

$$M_{m\times m} = \begin{pmatrix} 1-r & r/2 & & \\ r/2 & \ddots & \ddots & \\ & \ddots & \ddots & r/2 \\ & & r/2 & 1-r \end{pmatrix}$$

m	n	k	
		Jacobi	Gauss-Seidel
60	16	16	8
	22	22	11
	28	28	14
	34	34	17
	40	40	20
	46	43	23
	52	45	24
	58	46	24
90	16	16	8
	22	22	11
	28	28	14
	34	34	17
	40	40	20
	46	46	23
	52	52	26
	58	58	29

TABLE 1

Iterations (k) required for convergence of block methods.

and

$$N_{m\times m} = \begin{pmatrix} 1+r & -r/2 & & \\ -r/2 & \ddots & \ddots & \\ & \ddots & \ddots & -r/2 \\ & & -r/2 & 1+r \end{pmatrix}. \quad \blacksquare$$

Corollary: The parallel algorithm is slower than the serial one when

$$\epsilon \leq \begin{cases} 2.68 \times 10^{-9} & backward\ Euler \\ 3.78 \times 10^{-9} & Crank-Nicolson \end{cases}$$

Since communication time has not been factored into the above calculation, the parallel versions will in practice be slower for much larger values of ϵ.

5.3. Numerical Results. We have implemented the above block schemes on a workstation in order to illustrate and validate our claims. In the results to follow, we

counted the number of iterations required for convergence of block Jacobi and Gauss-Seidel schemes for the backward Euler method with $r \equiv \frac{\Delta t}{(\Delta x)^2} = 100$. Convergence is asserted when the norm of the residual is less than 10^{-7}.

Table 1 contains numerical results for the block Jacobi and Gauss Seidel methods applied to the backward Euler equations. The number of iterations required to converge to the desired solution increases linearly with the number of timesteps in parallel as predicted by the theory. This negates any potential benefits from parallelization.

6. Point Jacobi Iterative Schemes. In this section we consider the point Jacobi iterative method for the backward Euler method for simplicity. We consider the serial case and analyze the time it takes to solve for a single time step.

6.1. Backward Euler. For the conventional backward Euler scheme the iteration matrix is given by

$$T_{m\times m} = \frac{1}{(1+2r)} \begin{pmatrix} 0 & -r & & \\ -r & 0 & \ddots & \\ & \ddots & \ddots & -r \\ & & -r & 0 \end{pmatrix},$$

with eigenvalues given by

$$\lambda_i = \frac{1}{(1+2r)} \left[2r \cos\left(\frac{i\pi}{m+1} \right) \right], \qquad i = 1, \ldots, m.$$

The norm of the k^{th} power of the iteration matrix is given by

$$\|T^k\|_2 = \lambda_{max}^k = \frac{1}{(1+2r)^k} \left(2r \cos\left(\frac{\pi}{m+1} \right) \right)^k.$$

THEOREM 6.1. *The Jacobi iterations take approximately k iterations to converge where k is given by*

$$k = \frac{(\ln \epsilon) + \ln(1 + 1/(4r))}{\ln(2r) - \ln(1+2r)}.$$

Proof: We have

$$\|e_k\|_2 \leq \|T^k\|_2 \|e_0\|_2$$

Since $Au_{n+1} = u_n$ for the given boundary conditions and we use the solution at time t = n to initialize the iteration to solve for t = n + 1 we have

$$e_0 = u_n - A^{-1}u_n = (I - A^{-1})u_n$$

$$\|e_0\|_2 \leq \|I - A^{-1}\|_2 \|u_n\|_2 = \lambda_{max}(I - A^{-1}), \qquad \|u_n\|_2 \approx 1,$$

which reduces to

$$\|e_0\|_2 \leq \frac{4r}{1+4r} \approx 1.$$

The iteration converges after k iterations when $\|e_k\|_2 \leq \epsilon$, or,

$$\frac{4r}{1+4r}\left(\frac{2r\cos\left(\frac{\pi}{m+1}\right)}{1+2r}\right)^k \leq \epsilon.$$

This happens when

$$k \ln\left(\frac{2r\cos\left(\frac{\pi}{m+1}\right)}{1+2r}\right) = \ln\epsilon + \ln\left(1+\frac{1}{4r}\right)$$

or

$$k = \frac{(\ln\epsilon) + \ln(1+1/(4r))}{\ln(2r) - \ln(1+2r)}. \qquad \blacksquare$$

For the parallel case the iteration matrix is given by

$$T_{PJ} = \frac{1}{1+2r}\begin{pmatrix} L+U & & & \\ I & \ddots & & \\ & \ddots & \ddots & \\ & & I & L+U \end{pmatrix},$$

where

$$A = D - L - U,$$

$$L + U = V \Lambda V^t,$$

and $\|V\|_2 = 1$. The norm of the k^{th} power (assuming $k \geq n$) of the iteration matrix, which controls the rate of convergence, is given by

$$\|T_{PJ}^k\|_2 = \frac{1}{(1+2r)^k}\left\| \begin{matrix} \Lambda^k & & & & & & \\ C_1^k\Lambda^{k-1} & \Lambda^k & & & & & \\ \vdots & \ddots & \ddots & & & & \\ C_r^k\Lambda^{k-r} & & \ddots & \ddots & & & \\ \vdots & \ddots & & \ddots & \ddots & & \\ C_n^k\Lambda^{k-n} & \ldots & C_r^k\Lambda^{k-r} & \ldots & C_1^k\Lambda^{k-1} & \Lambda^k \end{matrix} \right\|_2$$

where C_i^j are the binomial coefficients.

THEOREM 6.2. *In the temporal parallel case, the point Jacobi method takes at least k iterations to converge where k is the solution to the equation*

$$\frac{C_n^k}{\lambda^n}\frac{\lambda^k}{(1+2r)^k} = \epsilon$$

with

$$\lambda = 2r\cos\left(\frac{\pi}{m+1}\right).$$

Proof: The iteration converges when $\|T^k\|_2 \leq \epsilon$. Since the norm of any matrix is greater than the largest entry in a matrix we have

$$\|T^k\|_2 \geq C_n^k \|\Lambda^{k-n}\|_2.$$

Since Λ is the matrix of eigenvalues of the original iteration matrix we have

$$\|\Lambda^{k-n}\|_2 = 2r\cos\left(\frac{\pi}{m+1}\right)^{k-n}.$$

Therefore the iteration converges when

$$\|T^k\|_2 \leq \epsilon,$$

or

$$C_n^k\left(2r\cos\left(\frac{\pi}{m+1}\right)\right)^{k-n} \leq \epsilon,$$

or when k is greater than the solution of the equation

$$\frac{C_n^k}{\lambda^n}\frac{\lambda^k}{(1+2r)^k} = \epsilon. \qquad \blacksquare$$

6.2. Numerical Results. Since the above expression for the number of iterations cannot be solved for in a simple manner we present the numerically determined values for k. We also present the number of iterations that were observed experimentally to validate our claim. In Figure 1 and Figure 2, we present graphs for the computed and experimentally observed efficiency of the parallel scheme for two different values of r. As can be easily observed, the efficiency of the parallel scheme decreases rapidly with increasing n.

This qualitative behavior is largely independent of m as can be seen from the graph in Figure 3, in which we show the efficiency of the parallel scheme for different values of m. Clearly, the efficiency does not stay constant as claimed in [6].

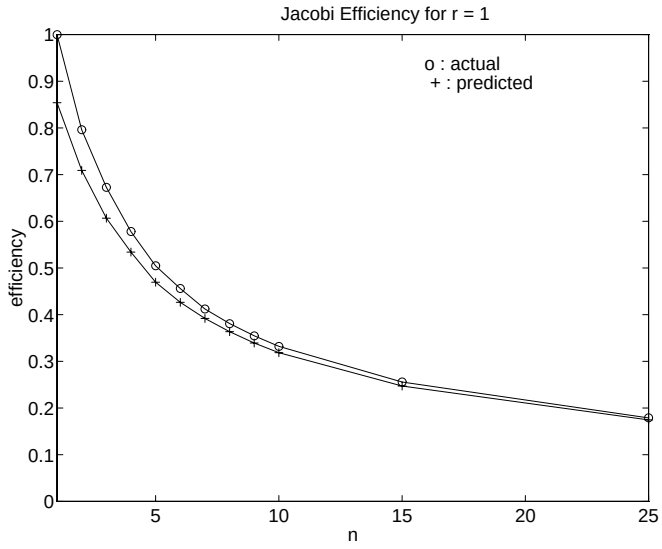

FIG. 1. Predicted vs. Actual Efficiency for m = 60 r = 1

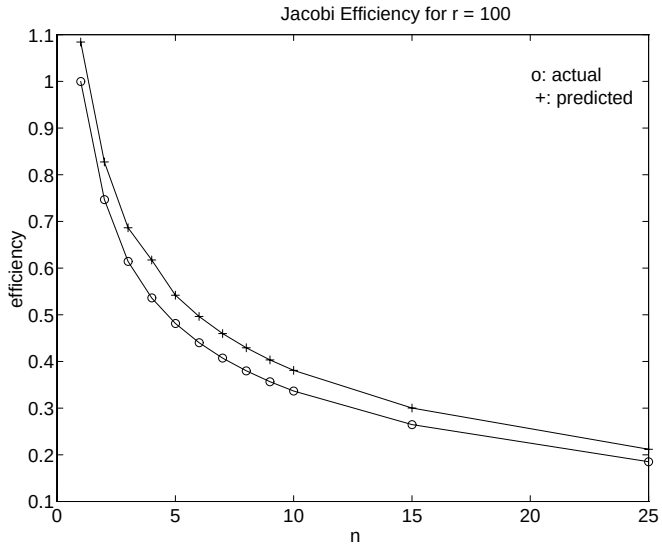

FIG. 2. Predicted vs. Actual Efficiency for m = 60 r = 100

7. Point Gauss-Seidel Iterative Schemes. In this section, we consider the point Gauss-Seidel scheme using the backward Euler method. We are unable to derive a lower bound for the 2-norm of k^{th} power of the Gauss-Seidel iteration matrix. We present numerical results to demonstrate that the efficiency of the parallel scheme goes down with increasing the number of parallel time steps contrary to the claims made

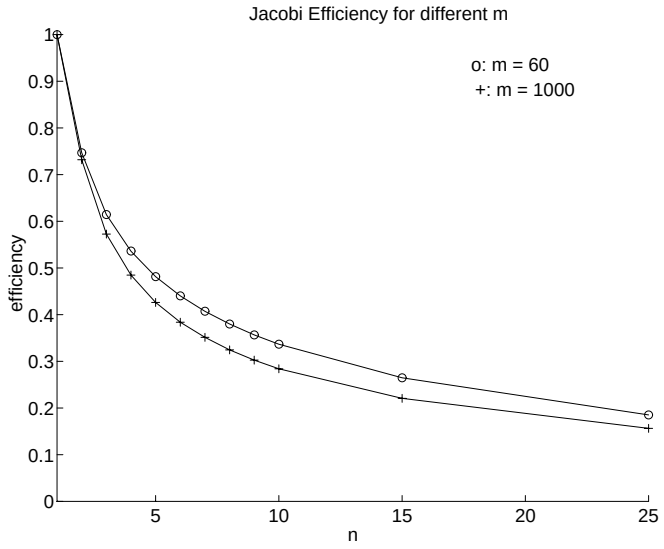

FIG. 3. Efficiency for m = 60 and m = 1000

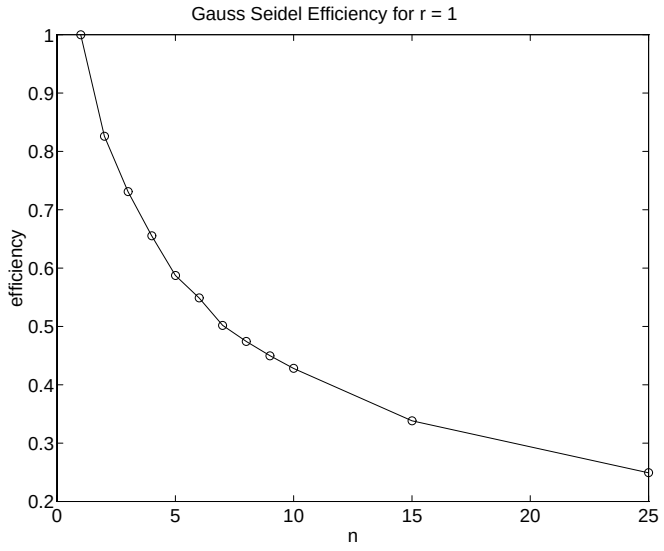

FIG. 4. Efficiency for m = 1000 r = 1

in [6] that it remains constant.

7.1. Numerical Results. In Figure 4 and Figures 5, we show the efficiency of the parallel scheme for two different values of r. As the graph shows, the efficiency decreases with increasing n.

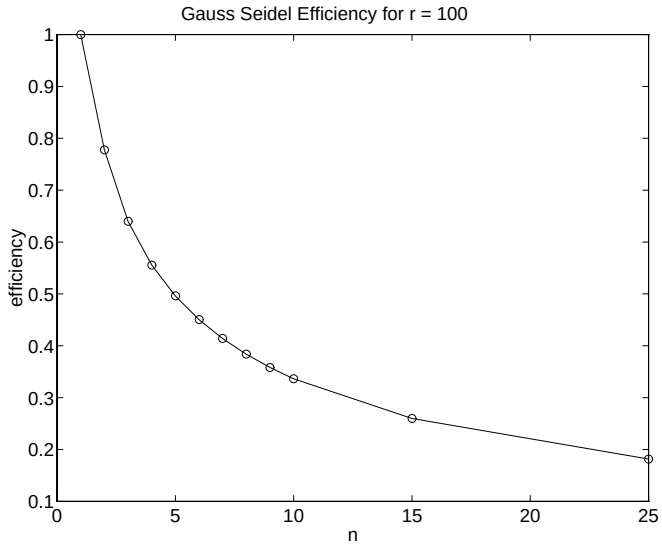

FIG. 5. Efficiency for m = 1000 r = 100

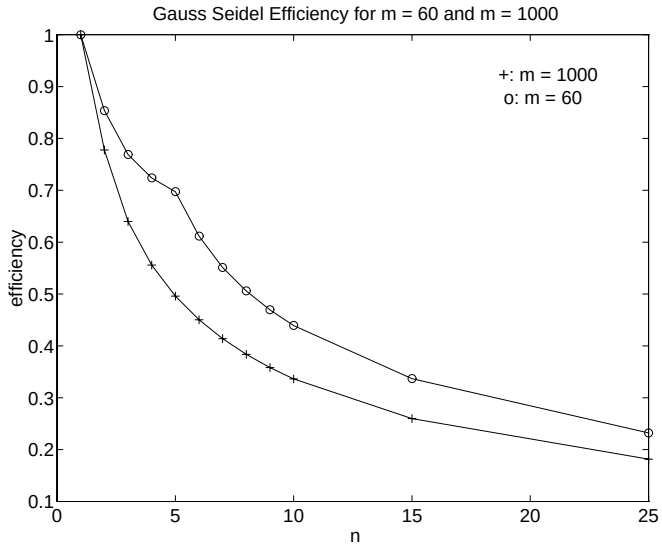

FIG. 6. Efficiency for m = 60 and m = 1000

In Figure 6, we show the behavior of the parallel scheme for two different values of m. As the graph shows, the efficiency decreases rapidly with increasing n irrespective of the value of m and does not stay constant.

8. Conclusions. We have analyzed the numerical solution of a linear, one-dimensional, time-dependent PDE using temporal domain parallelism. We have shown that the increase in the number of iterations required for the convergence of block and point Jacobi and Gauss-Seidel methods negates any advantages of temporal domain paralleism.

REFERENCES

[1] J. Saltz, *Parallel and Adaptive Algorithms for Problems in Scientific and Medical Computing*, PhD thesis, Department of Computer Science, Duke University, Durham, NC, 1985.

[2] H. Schwarz, *Uber einige abbildungsaufgaben*, Ges. Math. Abh., 11 (1869), pp. 65–83.

[3] R. Varga, *Matrix Iterative Analysis*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1962.

[4] D. Womble, *A time stepping algorithm for parallel computers*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 824–837.

[5] P. Worley, *Parallelizing across time when solving time-dependent partial differential equations*, in Proc. 5th SIAM Conf. on Parallel Processing for Scientific Computing, D. Sorensen, ed., SIAM, 1991.

[6] J. Zhu, *A new parallel algorithm for the numerical solutions of time dependent partial differential equations*, SIAM J. Sci. Stat. Comput., 40 (2001), pp. 1000–1001.