Parallel Multigrid and Domain Decomposition Algorithms for Elliptic Equations

Lawrence C. Cowsar*
Alan Weiser*
Mary F. Wheeler*

Abstract. We discuss two algorithms for modeling elliptic equations on hypercubes: multigrid and domain decomposition. Both schemes have been designed for cell-centered finite differences. One of these approaches is a semicoarsening multigrid algorithm for anisotropic non-self-adjoint elliptic partial differential equations; this scheme is an extension [15] of earlier work of J. Dendy et al. [6] for symmetric problems. The second approach is a domain decomposition procedure first introduced by Glowinski and Wheeler [8] for mixed finite element methods and then later modified by Cowsar and Wheeler [3] to include multilevel acceleration.

Performance of the codes has been compared on the Intel i860 hypercube. Both procedures perform well for modeling the pressure or hydraulic head equation which arises in flow in porous media problems. In both procedures in the hypercube implementation the domain is decomposed into subdomains, with one subdomain being assigned to each processor. We discuss numerical experiments performed with these methods. In particular we emphasize scaling and parallel efficiencies.

1. Introduction. Let Ω be a rectangular or box-shaped domain in 2 or 3 dimensions with boundary Γ. The equation to be solved is the following:

\[ u(x) = -A(x)\nabla p(x) \]

in Ω,

\[ \nabla \cdot u(x) = f(x) \]

in Ω, with boundary conditions

\[ u \cdot n = 0 \]

on Γ, with the additional condition that, say,

\[ \int_{\Omega} p(x) dx = 0 \]

* Department of Mathematical Sciences, Rice University
to fix $p$ uniquely.

A physical interpretation of this equation is that of a material balance. The pressure or hydraulic head $p$ induces a velocity or flux vector $u$ with tensor coefficient matrix $A$, according to, say, Darcy's law. The divergence of the flux is equal by material balance to the sources (and sinks) $f(x)$, corresponding to well injection rates in reservoir simulation. The boundary conditions $u \cdot n = 0$ correspond to no flow into or out of the domain $\Omega$.

We solve (1) using a mixed finite element method. We choose appropriate finite dimensional spaces $V$ and $W$ such that

$$V \subset H(\Omega, \text{div}) = \{ v \in (L^2(\Omega))^D \mid \nabla \cdot v \in L^2(\Omega) \}$$

and

$$W \subset L^2(\Omega)$$

for a $D$-dimensional problem, where we require the inclusion

$$\text{div}(V) \subset W.$$ 

Then the mixed finite element method finds $U \in V$ and $P \in W$ such that

$$\int_\Omega A^{-1} U \cdot v dz = \int_\Omega P \nabla \cdot v dz - \int_\Gamma qu \cdot nds$$

for all $v \in V$ and

$$\int_\Omega \nabla \cdot U w dz = \int_\Omega f w dz$$

for all $w \in W$.

If we discretize by intervals in the coordinate directions in $x = (x_1, x_2)$ or $(x_1, x_2, x_3)$, choose $W$ to be the space of piecewise constants on the resulting set of rectangular gridblocks, and choose each component of $V$ to be the space of functions which are continuous piecewise linears in the given component direction and piecewise (discontinuous) constants in the other component directions, the lowest-order Raviart-Thomas mixed method [13] results. If we further use approximate quadrature rules corresponding to tensor product midpoint rules for all integrals except for the trapezoidal rule in the given component direction for the $\int_\Omega A^{-1} U \cdot v dz$ term, the standard block-centered finite difference or finite volume method is obtained [14,17]. The linear system is typically derived by pre-eliminating the velocities in terms of the pressures, resulting in a linear system for one unknown pressure per gridblock.

The target area we have in mind is petroleum reservoir simulation. The linear systems that arise in this area are characterized by several properties. They tend to be large (on order of tens of thousands of gridblocks) and a different system needs to be solved for each timestep (on order of thousands of timesteps). The coefficient matrices $A$ tend to be anisotropic, with much stronger coupling in the vertical than horizontal directions because of the underlying "pancake-shaped" gridblock geometry, and strongly discontinuous functions of $x$ because of the variations in the permeability of the reservoir rock. Various linear solution methods are in use in reservoir simulation, e.g., [1,7,16]. Among the methods proposed in recent reservoir simulation literature, semicoarsening multigrid [6] and domain decomposition (e.g., [12]) seem promising.

In Section 2, we describe the semicoarsening multigrid method. In Section 3, we describe the domain decomposition method. In Section 4, we compare the two methods on three test cases. In Section 5, we summarize our results.
2. The semicoarsening multigrid method. Consider a two-dimensional problem (1-4) with a discretization resulting in a nine-point operator. The resulting nine-diagonal matrix is depicted in Figure 1. Let $P$ be the permutation matrix ordering lines of unknowns red-black. The resulting red-black matrix $P^TAP$ is depicted in Figure 2.

In block form, the red-black linear system is

$$Az = b$$

or

$$
\begin{pmatrix}
A_{rr} & A_{rb} \\
A_{br} & A_{bb}
\end{pmatrix}
\begin{pmatrix}
z_r \\
z_b
\end{pmatrix} =
\begin{pmatrix}
b_r \\
b_b
\end{pmatrix}.
$$

The nonzeros in Figure 2 are labeled so that $A_{rr}$ nonzeros are labeled $e, a, b, A_{bb}$ nonzeros are labeled $r, p, q, f, c, d$, $A_{br}$ nonzeros are labeled $m, g, h, o, k, l$, and $A_{bb}$ nonzeros are
labeled $n, i, j$. One way to solve (11) is to form and solve the smaller Schur complement linear system

$$ (12) \quad A_s x_b = b_s $$

where

$$ (13) \quad A_s = A_{bb} - A_{br} A_{rr}^{-1} A_{rb} $$

and

$$ (14) \quad b_s = b_b - A_{br} A_{rr}^{-1} b_r, $$

and then backsolve

$$ (15) \quad x_r = A_{rr}^{-1} (b_r - A_{rb} x_b). $$

Now consider a two-level multigrid method for the original system, where the black unknowns are the coarse grid unknowns. The steps in such a method are:

i) smooth on the fine grid

ii) transfer the fine grid residual to the coarse grid and solve for the coarse grid correction

iii) transfer the correction back to the fine grid and add it to the current solution

iv) smooth again on the fine grid.

In the black-box multigrid framework [5] it is customary to construct the coarse grid system $A_c x_c = b_c$ as follows:

$$ (16) \quad \begin{pmatrix} T_{br} & I_{bb} \\ \end{pmatrix} \begin{pmatrix} A_{rr} & A_{rb} \\ A_{br} & A_{bb} \\ \end{pmatrix} \begin{pmatrix} T_{rb} \\ I_{bb} \\ \end{pmatrix} \begin{pmatrix} x_c \\ b_b \\ \end{pmatrix} = \begin{pmatrix} T_{br} \\ I_{bb} \\ \end{pmatrix} \begin{pmatrix} b_r \\ b_b \\ \end{pmatrix} $$

Here $b_r$ and $b_b$ denote the current residuals, and are only equal to the original right hand side if the current iterate is zero. $I_{bb}$ denotes the coarse grid identity matrix. Pure injection is used for interpolating coarse grid unknowns to black unknowns on the fine grid.

If $T_{br} = -A_{br} A_{rr}^{-1}$ and $T_{rb} = -A_{rr}^{-1} A_{rb}$, then the coarse grid system for two-level multigrid is exactly the Schur complement system $A_s x_s = b_s$. However, the Schur complement matrix is much denser than the original matrix. Fill occurs for two reasons. First, $A_{rr}^{-1}$ has dense diagonal blocks, so $T_{br}$ and $T_{rb}$ have dense diagonal blocks. Second, extra fill occurs in direction 1. For instance, the black unknowns corresponding to $(1, i, 2) = (2, 1)$ and $(4, 3)$ are both connected to the red unknown corresponding to $(1, i, 2) = (3, 2)$ via $T_{br}$ and $T_{rb}$ terms. Elimination of red terms results in direct connections between $(2, 1)$ and $(4, 3)$ in $A_s$ even though their indices differ by more than 1 in direction 1. The only way to avoid this extra fill is to allow connections in $T_{br}$ and $T_{rb}$ in direction 2 only. The locations in Figure 2 corresponding to connections in direction 2 only are the locations for diagonals $c, g, k,$ and $p$.

Some new notation must now be introduced to deal with connections in direction 2 only. Let an "$l$" (resp. "$r$") superscript appended to $A_{rb}$ or $A_{br}$ denote the result of zeroing out all entries except those corresponding to connection of a black unknown to a red unknown with a smaller (resp. larger) index in direction 2. Then $A_{rl} = A_{rb}^{l} + A_{br}^{l}$ and $A_{rl} = A_{rb}^{l} + A_{br}^{l}$. For example, in Figure 2, $A_{rb}^{l}$ may contain nonzeros on diagonals $c, d,$ and $f$; $A_{rb}^{l}$ may contain nonzeros on diagonals $p, q,$ and $r$; $A_{br}^{l}$ may contain nonzeros on diagonals $g, h,$ and $m$; and $A_{br}^{l}$ may contain nonzeros on diagonals $k, l, and o$. Similarly, let $dia_{rl}^{c}(v)$ (resp. $dia_{rl}^{c}(v), dia_{rl}^{c}(v), dia_{rl}^{c}(v)$) denote the matrix with the same nonzero structure as $A_{rb}$ (resp. $A_{rb}, A_{br}, A_{br}$) with the only nonzero entries obtained from vector $v$ and put on
the diagonal corresponding to connections of black unknowns to red unknowns with lower (resp. higher, lower, higher) index in direction 2 and with the same index in direction 1. For example, in Figure 2, \(\text{diag}^p_b(v)\) (resp. \(\text{diag}^g_b(v), \text{diag}^l_r(v), \text{diag}^r_r(v)\)) puts nonzeros in the locations of diagonal \(c\) (resp. \(p, g, k\)).

With this notation, we define the following transfer operators due to Steve Schaffer at the New Mexico Institute of Mining and Technology [4]:

\[
\begin{align*}
T^{rb} &= -(\text{diag}^l_b(A^{-1}_r A^{-1}_b e) + \text{diag}^g_r(A^{-1}_r A^{-1}_r e)) \\
T^{br} &= -(\text{diag}^l_r(e^T A^{-1}_r A^{-1}_b e) + \text{diag}^g_r(e^T A^{-1}_r A^{-1}_r e))
\end{align*}
\]

where \(e\) is the vector of all ones.

We use (17) and (18) for transfer operators. We take the initial guess for the solution to (10) on a given grid to be

\[
x_0 = \frac{b^T e}{e^T A e} e
\]

rather than zero. For our smoother we use red-black line Gauss-Seidel with lines oriented in direction 1. This completes the specification of the basic two-level two-dimensional semi-coarsening multigrid method. The adjective semi-coarsening is used because the coarse grid is only coarsened in direction 2, while usual "full coarsening" multigrid involves coarsening in both directions 1 and 2 simultaneously.

The method works particularly well for strongly anisotropic problems [15].

Now consider a three-dimensional model problem with a 15-point operator (a 3 point operator in direction 3 tensored with a 5 point operator in directions 1 and 2). Let \(P\) be the permutation matrix ordering planes of unknowns red-black.

Again the same considerations cause fill in the Schur complement matrix. Our notation to deal with connections in direction 3 is similar to the two-dimensional case.

We would again like to use transfer operators (17) and (18). However, now \(A_{rr}\) is a 5-point operator which is expensive to invert. Hence, following Schaffer [4], we use one cycle of our two-dimensional semi-coarsening multigrid method to approximately solve the \(A_{rr}\) systems needed in constructing \(T^{rb}\) and \(T^{br}\).

We use (19) for initial guesses and approximate red-black Gauss-Seidel plane relaxation for our fine-grid smoothing step, where the two-dimensional semi-coarsening multigrid method approximately solves the plane Gauss-Seidel equations. This completes the specification of the basic two-level three-dimensional semi-coarsening multigrid method.

Note that a fine grid three-dimensional 7-point operator forms 15-point operators on the coarser three-dimensional grids, and a fine grid two-dimensional 5-point operator forms 9-point operators on the coarser two-dimensional grids. Again this method works particularly well for strongly anisotropic problems [15].

We use two standard multigrid cycling approaches. The "V-cycle" (Figure 3) starts on the finest grid with a two-level method and solves the resulting coarser grid system recursively with another two level method, and so on until the coarsest grid system with one unknown is solved directly. The "FMV-cycle" or full multigrid V-cycle (Figure 4) generates an initial guess for the fine grid system by preceding the fine-grid V-cycle recursively with another V-cycle on the next-finer grid, and so on so that the very first calculation is a direct solve on the coarsest grid.

We parallelize the semi-coarsening multigrid method on the Intel i860 hypercube using domain decomposition [15]. We assign a rectangular subdomain to each processor. Processors exchange boundary information with neighboring subdomains as needed. A special
tridiagonal solver handles the line Gauss-Seidel solves — it consists of 2-level cyclic reduction coupled with burn-at-both-ends (BABEL) elimination within each subdomain [11]. We use special techniques for grids "below C-level", that is, grids that are so coarse they have no gridblocks for some processors (cf. [2,10]).

3. The domain decomposition method. We use a nonoverlapping Dirichlet domain decomposition method [3,8,9]. Assume the original linear system is symmetric and positive definite. To motivate the method, consider the grid depicted in Figure 5, with $p$'s representing pressures at the centers of grid blocks and $\lambda$'s representing pressures in thin interface blocks. Let the widths of the interface blocks shrink to zero. Then the Lagrange multipliers $\{\lambda\}$ enforce continuity of fluxes across interfaces, and the solution of the augmented system is the same as the solution of the original system without interfaces.

The resulting linear system for $p$'s and $\lambda$'s is of the form

\begin{equation}
\begin{pmatrix}
M_{pp} & M_{p\lambda} \\
M_{\lambda p} & M_{\lambda\lambda}
\end{pmatrix}
\begin{pmatrix}
p \\
\lambda
\end{pmatrix}
= 
\begin{pmatrix}
b_p \\
b_{\lambda}
\end{pmatrix}
\end{equation}

and $\lambda$ satisfies the Schur complement system

\begin{equation}
\bar{M}\lambda = \bar{b}
\end{equation}

where

\begin{equation}
\bar{M} = M_{\lambda\lambda} - M_{\lambda p} M_{pp}^{-1} M_{p\lambda}
\end{equation}

and

\begin{equation}
\bar{b} = b_{\lambda} - M_{\lambda p} M_{pp}^{-1} b_p.
\end{equation}

$\bar{M}$ is symmetric and positive definite since the original matrix is. Moreover, the condition number of $\bar{M}$ is proportional to $h^{-1}$: there exist $C_0, C_1 > 0$ such that

\begin{equation}
C_0 \|\lambda\|^2 \leq \lambda^T \bar{M} \lambda \leq C_1 \|\lambda\|^2
\end{equation}

where

\begin{equation}
\|\lambda\|^2 = \int_{\bigcup \Gamma_i} n^T A n \lambda^2 \, ds
\end{equation}

and $n$ is the outer normal to $\Gamma_i$ [3,9].
We solve the system $\bar{M}\lambda = \bar{b}$ by multigrid on interfaces. We never form $\bar{M}$ explicitly, just apply it. Our current multigrid method obtains coarse grid systems by harmonic weighting of fine grid coefficients. The injection operator used is piecewise constant averaging, and the prolongation operator used is piecewise constant assignment. The smoother used is the conjugate residual method [7], and the subgrid solver is nested factorization with Orthomin acceleration [1]. The multigrid cycle procedure used is one FMV-cycle followed by subsequent V-cycles. On the Intel hypercube, two kinds of communication between processors take place – calculation of residuals on interfaces, and communication of dot products occur across processors during computation of smoothing on the interfaces.

4. Numerical results. We made runs with three test cases on the Intel iPSC/860 using the PGF77 compiler with the .03 optimization option. The convergence tolerance was a reduction in the $L_2$ norm of the original residual by a factor of $10^6$.

Problem 1 is posed on the unit cube on a 48 by 48 by 8 grid, with $A = \text{diag}(1,1,100)$, sources $f(x) = \delta_{(1,1,1)} - \delta_{(0,0,0)}$ and boundary conditions $\mathbf{u} \cdot \mathbf{n} = 0$ on $\Gamma$. This problem represents a model with anisotropy, as would arise from a 10 to 1 ratio of horizontal to vertical grid block size in a reservoir simulation model.

Problem 2 is posed on the unit cube on a 24 by 24 by 24 grid, with $A = \text{diag}(a,a,a)$, $a=1$ except $a=.00001$ at simple flow barriers (Figure 6). The boundary conditions are $p(x_1,x_2,1) = 1$, $p(x_1,x_2,0) = 0$, and $\mathbf{u} \cdot \mathbf{n} = 0$ on the other four faces of $\Gamma$. This problem represents a model with discontinuous coefficients without a preferred direction of flow.

Problem 3 is posed on the unit square on an 80 by 80 grid, with $A = \text{diag}(a,a,a)$, $a$ ranging from .6 to 12.7 (Figure 7). The boundary conditions are $p(0,x_2) = 1$, $p(1,x_2) = 0$, and $\mathbf{u} \cdot \mathbf{n} = 0$ at $x_2 = 0,1$. This problem represents a model with more strongly discontinuous coefficients, as arises in reservoir simulation models of unfavorable mobility ratio displacement.

CPU time results are indicated in Tables 1-3. The numbers of processors in the three coordinate directions are indicated by $(n\text{proc}_1) \times (n\text{proc}_2) \times (n\text{proc}_3)$. MG, MGNC, and DD refer to the multigrid method, the multigrid method excluding CPU time for interprocessor communication, and the domain decomposition method, respectively.

The results indicate that the domain decomposition method scales much better than the multigrid method as the number of processors increases. This is because domain decomposition requires less interprocessor communication per iteration. Previous results with
FIG. 6. Problem 2 specification

FIG. 7. Problem 3 coefficient $a$

### Table 1
Problem 1 CPU times

<table>
<thead>
<tr>
<th></th>
<th>1x2x2</th>
<th>2x2x1</th>
<th>2x2x2</th>
<th>3x3x1</th>
<th>4x4x1</th>
</tr>
</thead>
<tbody>
<tr>
<td>MG</td>
<td>12.4</td>
<td>20.7</td>
<td>18.1</td>
<td>18.2</td>
<td>19.2</td>
</tr>
<tr>
<td>MGNC</td>
<td>6.5</td>
<td>8.5</td>
<td>5.6</td>
<td>4.1</td>
<td>4.1</td>
</tr>
<tr>
<td>DD</td>
<td>18.6</td>
<td>10.2</td>
<td>9.2</td>
<td>4.8</td>
<td></td>
</tr>
</tbody>
</table>

### Table 2
Problem 2 CPU times

<table>
<thead>
<tr>
<th></th>
<th>1x1x1</th>
<th>1x1x2</th>
<th>1x2x2</th>
<th>2x2x2</th>
<th>1x4x4</th>
<th>3x3x3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MG</td>
<td>13.9</td>
<td>10.8</td>
<td>11.6</td>
<td>16.9</td>
<td>11.3</td>
<td>17.5</td>
</tr>
<tr>
<td>MGNC</td>
<td>13.9</td>
<td>9.6</td>
<td>7.4</td>
<td>6.6</td>
<td>3.6</td>
<td>3.1</td>
</tr>
<tr>
<td>DD</td>
<td>34.6</td>
<td>14.0</td>
<td>7.3</td>
<td>3.9</td>
<td>3.2</td>
<td></td>
</tr>
</tbody>
</table>
the multigrid method on the Intel iPSC/2 indicated better speedups [15] - the slowdown on the iPSC/2 would be closer to the MGNC rows than the MG rows. Multigrid is faster than domain decomposition with a small number of processors, because domain decomposition reverts to nested factorization on one processor, which requires more iterations than multigrid for a fine grid. Domain decomposition requires relatively more time for the problem with more strongly discontinuous coefficients. This is predominantly due to more iterations being required by the nested factorization subgrid solver. Both methods require few outer iterations to solve all three problems.

5. Summary. A semicoarsening multigrid method and a domain decomposition method for solving elliptic equations are presented. Both methods converge fast in terms of outer iterations for anisotropic models and models with strongly discontinuous coefficients. The advantages of the multigrid method are that it is defined for nonsymmetric models, has a systematic procedure for generating transfers and coarse grid operators, has an optimal convergence rate with one processor, and seems more robust than nested factorization as a subgrid solver for problems with discontinuous coefficients. The advantages of the domain decomposition method are that it requires much less interprocessor communication per iteration, much less storage, and requires no special triadiagonal solver.

Based on these relative advantages, we propose for future investigation a hybrid method in which semicoarsening multigrid is used to determine transfer operators, coarse grid linear systems, and subgrid solvers within the context of domain decomposition with multigrid on interfaces.

REFERENCES

[7] Eisenstat, S. C., Elman, H. C., and Schultz, M. H., Block-Preconditioned Conjugate-


