NPAC Technical Report SCCS 738

Application of Parallel Multigrid Methods to Unsteady Flow: A Performance Evaluation

A. T. Degani[a*] and G. C. Fox[†]

[a]Northeast Parallel Architectures Center
Syracuse University
Syracuse, NY 13244, USA

ABSTRACT

The parallel multigrid time-accurate calculation of the unsteady incompressible Navier-Stokes equations is carried out using both explicit and implicit schemes. In the explicit solution method, a 'lumped' scheme is employed at the coarsest multigrid levels where all the processors solve the same problem. On the other hand, in the implicit method, in which the equations are solved in a fully-coupled mode, a 'semi-distributive' scheme is used where the effective number of active processors decreases logarithmically with each coarsening of the mesh at the coarsest levels. Both 'V' and 'W' cycles are implemented in the explicit method and the convergence rates and execution times are compared. It is demonstrated that good speedups are obtained for the implicit scheme and the slight degradation in parallel efficiency, relative to calculations performed on the finest grid, is dominated by increased convergence rates.

1. INTRODUCTION

Parallel multigrid computation offers two desirable properties for the solution of large problems: i) computational effort scales with problem size, typically of $O(N \log N)$ where N denotes the size of the problem, and ii) implementation is scalable on coarse-grained distributed machines; specifically, good speedups and parallel efficiencies are possible as the number of processors p increases for N/p large and fixed. Here the parallel implementation of the multigrid method is applied to the time-accurate calculation of the unsteady incompressible Navier-Stokes equations. The primary objective here is the evaluation of parallel multigrid methods to unsteady flow in terms of convergence rates and parallel efficiency as the size of the problem and number of processors is varied. As a first step, a regular structured two-dimensional computational domain is considered; however, since

*Alex G. Nason Research Fellow
†Director, Professor of Computer Science and Physics

a primitive-variable formulation is adopted, an extension to three dimensions is straightforward. Both explicit and implicit schemes on a staggered mesh are considered, and, in the former, the relative effectiveness of the 'V' and 'W' multigrid cycling algorithms is evaluated.

The multigrid algorithm was first coded on a uniprocessor machine in FORTRAN 77 but designed in such a fashion so as to allow the subsequent seamless transition to the development of a Single Program Multiple Data (SPMD) code with message passing. The results reported here were obtained on a 32-node CM-5 installed at NPAC. The data are distributed in a block-block layout on a two-dimensional mesh of abstract processors and, for efficient memory utilization, the local data in each processor are mapped onto a local one-dimensional array. The array of cells in each processor is augmented by a buffered boundary of one cell thickness at all multigrid levels where data from neighboring processors are stored.

The unsteady incompressible Navier-Stokes equations are given by

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \mathbf{g} - \nabla p + \frac{1}{Re} \nabla^2 \mathbf{u}, \qquad \nabla \cdot \mathbf{u} = 0, \tag{1}$$

where the equations are nondimensionalized by appropriate velocity and length scales, U_o and L_o, respectively, and the kinematic viscosity ν. Re is the Reynolds number defined as $Re = U_o L_o / \nu$.

2. EXPLICIT SCHEME

The explicit scheme adopted here is the projection method [1] in which the momentum equation is split according to

$$\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} + \mathbf{u}^n \cdot \nabla \mathbf{u}^n = \mathbf{g}^n + \frac{1}{Re} \nabla^2 \mathbf{u}^n, \tag{2}$$

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^*}{\Delta t} = -\nabla p^{n+1}, \tag{3}$$

where the superscript n denotes the evaluation of a quantity at time level n and $\mathbf{u}^*$ is an intermediate provisional value of the velocity field; this scheme is formally $O(\Delta t)$ accurate. Upon taking the divergence of equation 3 and using the continuity equation evaluated at time level $n+1$,

$$\nabla^2 p^{n+1} = \frac{1}{\Delta t} \nabla \cdot \mathbf{u}^*. \tag{4}$$

The boundary condition for the above Poisson equation for the pressure is obtained by projecting equation 3 on the boundary Γ of the computational domain (i.e. a dot product of equation 3 and the normal vector $\hat{\mathbf{N}}$ of Γ) and letting $u_\Gamma^{n+1} = u_\Gamma^*$ [1]. This yields

$$\left(\frac{\partial p}{\partial N} \right)_\Gamma^{n+1} = 0. \tag{5}$$

Note that the homogeneity of the Neumann boundary condition above is not physical but only a numerical artifice made possible due to the use of a staggered grid [1]. In

the projection method outlined above, time-stepping is accomplished in the following manner: *(i)* the provisional value of the velocity $\mathbf{u}^*$ is obtained from equation 2, *(ii)* the Poisson equation 4 is solved for the pressure subject to the boundary condition given by equation 5, and, finally, *(iii)* the velocity field at time level $n+1$ is obtained from equation 3.

The projection method is implemented on a staggered mesh where the normal velocities are defined at the midpoint of the cell faces and the pressure is defined at the cell center. All spatial derivatives are evaluated using second-order central differences. Since the no-slip condition cannot be satisfied exactly on a staggered mesh, fictitious points are defined along the boundary of the computational domain Γ. Using the prescribed value of the no-slip velocity and a fourth-order extrapolation formula, the values of the tangential velocity at the fictitious points are evaluated, thereby ensuring second-order accuracy for all spatial derivatives in equation 2.

The Poisson equation for the pressure subject to the homogeneous Neumann boundary condition is solved efficiently at each time step by employing a Correction-Scheme (CS) multigrid method which is appropriate for linear problems. At each level, the equations are relaxed by the well-known point red-black Gauss-Seidel scheme. Note that if M denotes the total number of multigrid levels, then at some level k, the 'W' cycle relaxes the equations 2^{M-k} more often than in the 'V' cycle where $k = M$ denotes the finest grid. In the scheme adopted here, the governing equation for the interior of the computational domain is relaxed twice in the forward sweep and once in the backward sweep; thus the 'V' and 'W' cycle scheme here may be denoted by $V(2,1)$ and $W(2,1)$, respectively. Furthermore, following Brandt [2], it was found effective to relax the boundary cells twice for each sweep of the interior cells.

Next, consider the idle-processor problem which occurs at a critical coarse-grid level where the total number of cells is less than the number of processors. The simplest approach is to have each processor solve the same problem at and below the critical level. In this case, denoted here as the 'lumped' scheme, it may be shown that the ideal efficiency (i.e. discounting all communication costs) is such that

$$(a)\ \ n \gg p \quad \eta_{ideal}^{-1} \ \sim \ 1 + O(\frac{1}{n}), \qquad (b)\ \ n, p \gg 1 \quad \eta_{ideal}^{-1} \ \sim \ 1 + O(\frac{p}{n}), \tag{6}$$

where $n = N/p$. It is thus hoped that this scheme will be more effective than one which attempts to distribute the reduced extent of parallelism at the coarsest grids thus incurring relatively large latency cost due to frequent transmission of small messages.

3. IMPLICIT SCHEME

A spatial and temporal second-order accurate upwind-downwind discretization scheme [3] is applied to the computation of the unsteady incompressible Navier-Stokes equations on a staggered grid. A temporal discretization of the momentum equations at the mid-time plane, i.e. at $t + \Delta t/2$, yields

$$\frac{u^{n+1} - u^n}{\Delta t} = \overline{u\frac{\partial u}{\partial x}} + \overline{v\frac{\partial u}{\partial y}} - \frac{\partial p}{\partial x}^{n+\frac{1}{2}} + \frac{1}{Re}\left[\overline{\frac{\partial^2 u}{\partial x^2}} + \overline{\frac{\partial^2 u}{\partial y^2}}\right], \tag{7}$$

$$\frac{v^{n+1}-v^n}{\Delta t} = \overline{u\frac{\partial v}{\partial x}} + \overline{v\frac{\partial v}{\partial y}} - \frac{\partial p}{\partial y}^{n+\frac{1}{2}} + \frac{1}{Re}\left[\overline{\frac{\partial^2 v}{\partial x^2}} + \overline{\frac{\partial^2 v}{\partial y^2}}\right], \tag{8}$$

where the overbar denotes the evaluation of the quantities at the mid-time plane, and the superscripts n, $n+1$ and $n+\frac{1}{2}$ indicate the values of the associated variables at times t, $t+\Delta t$ and $t+\Delta t/2$, respectively. The resulting difference equations are given by [3]

$$\mathbf{M}_+\mathbf{q} = \mathbf{F} = \mathbf{M}_-\mathbf{q}^* + \mathbf{G}, \tag{9}$$

where

$$\mathbf{q} = \begin{bmatrix} u \\ v \\ p \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} G_u \\ G_v \\ G_p \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} F_u \\ F_v \\ F_p \end{bmatrix},$$

$$\mathbf{M}_+ = \begin{bmatrix} M_+^u & 0 & D^u \\ 0 & M_+^v & D^v \\ D^u & D^v & 0 \end{bmatrix}, \quad \mathbf{M}_- = \begin{bmatrix} M_-^u & 0 & 0 \\ 0 & M_-^v & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The boundary conditions may also be written compactly as

$$\mathbf{\Lambda q} = \mathbf{\Phi}, \tag{10}$$

where $\mathbf{\Lambda}$ is the boundary operator and $\mathbf{\Phi}$ is specified [3]. In a typical solution procedure, equations 9 and 10 are relaxed alternately until convergence. For the implicit method, only the 'V' cycle is considered and the FAS multigrid algorithm, appropriate for nonlinear problems, is applied [2]. The equations are relaxed in a fully-coupled mode at each multigrid level by the PAR-SCGS algorithm [3], a parallel version of the Symmetrical-Coupled Gauss-Seidel (SCGS) algorithm [4], appropriate for distributed-memory machines using message passing. In this scheme, the four velocities at the faces of each cell and the pressure defined at the cell center are updated simultaneously in an iterative process that traverses all the cells in the computational domain.

In the implicit calculations, an alternative approach to that adopted in the explicit method is used for multigrid levels coarser than the critical level below which the number of cells is less than the number of processors [3]. Below the critical level, a group of four processors in a 2×2 grid coalesce their data and solve the same problem; in effect only $1/4th$ of the processors are active. The 'adjacent' neighbors are now a stride of 2 away in each of the coordinate directions. Similarly, at the next coarsening, a group of 16 processors solve the same problem and so on. For this algorithm, it may be shown that [3] that

$$(a)\ \ n \gg p \quad \eta_{ideal}^{-1} \ \sim \ 1 + O(\frac{1}{n}), \quad (b)\ \ n, p \gg 1 \quad \eta_{ideal}^{-1} \ \sim \ 1 + O(\frac{\log p}{n}). \tag{11}$$

Thus, the degradation in the ideal parallel efficiency is reduced to an acceptable level in comparison with the previous method for the case where both n and p are large.

Table 1
Residual tolerance error for various grid sizes

	Global Grid			
	32×32	64×64	128×128	256×256
Tol	1.0×10^{-3}	2.5×10^{-4}	6.0×10^{-5}	1.5×10^{-5}

4. RESULTS AND DISCUSSIONS

Table 1 shows the residual tolerance error that is specified for various fine grids. It is appropriate to choose the tolerance error to be of the same order of magnitude as the discretization error which for a second-order accurate scheme is of the form Kh^2 where h denotes the mesh spacing and $K \sim O(1)$. Choosing $K = 1$, the residual tolerance error is then approximately set to h^{-2}. Convergence is deemed to have occurred at each time step when the residuals of all the equations are less than the residual tolerance. In the context of multigrid methods, it is convenient to quantify the computational effort in terms of work units (WU) [5]; a work unit is the computational effort required to relax the equations at the finest grid. It may be shown that for a two-dimensional computational domain, the number of WU's required for one cycle with $M > 1$ levels of multigrids is given by

$$WU = \frac{1}{1 - 2^{\mu-3}} \left[1 - 2^{(\mu-3)(M-1)}\right] (\nu_1 + \nu_2) + \nu_1 2^{(\mu-3)(M-1)-(\mu-1)}, \tag{12}$$

where $\mu = 1, 2$ for 'V' and 'W' cycles, respectively, and ν_1, ν_2 denote the number of iterations in the forward and backward sweeps, respectively. Note that in this study it is assumed that $\nu_1 = 2$ and $\nu_2 = 1$.

The results discussed here are for the classic test case of flow in a square cavity in which the top wall is set into motion impulsively at unit speed. Consider the explicit calculations first where $Re = 10^4$ and $\Delta t = 10^{-4}$ have been chosen. Table 2 shows the number of WU's required to obtain a converged solution at each time step as a function of the number of multigrid levels and resolution of the finest grid; the results are obtained by averaging the WU's over the first 10 time steps. It may be noted that the WU's decrease more rapidly for the 'W' cycle as the multigrid levels increase beyond one as compared to the 'V' cycle. However, beyond a certain level, indicated by an asterisk, the convergence rate of the 'W' cycle plateaus and no benefit is accrued in increasing the number of levels. On the other hand, the WU's decrease monotonically as the number of levels is increased for calculations with the 'V' cycle. This trend indicates that for comparable convergence rates, the number of multigrid levels required in the 'W' cycle is typically less than that required in the 'V' cycle; in the context of parallel computation, this is significant because inefficient calculations at the coarsest levels may be avoided in a 'W' cycle. All subsequent results reported here for 'W' cyles use the number of levels indicated by an asterisk in Table 2, but, for 'V' cyles, the full complement of available multigrid levels is employed.

For the 'lumped' scheme used here for the explicit calculations, the parallel efficiency

Table 2
Work units for explicit calculations on a processor mesh of 8×4 ($Re = 10^4, \Delta t = 10^{-4}$).

Levels	Global Grid							
	32×32		64×64		128×128		256×256	
	V	W	V	W	V	W	V	W
1	110	110	-	-	-	-	-	-
2	39	39	259	259	-	-	-	-
3	13	10	71	45	749	471	-	-
4	6	*7	21	15	206	72	1753	617
5	5	7	10	*13	56	*19	456	88
6	-	-	9	13	19	19	118	*25
7	-	-	-	-	13	19	31	25
8	-	-	-	-	-	-	21	25

is obtained from

$$\eta_{par} = t_{tot}^{-1} \left[t_{comp,dist} + \frac{t_{lumped}}{p} \right], \quad t_{tot} = t_{comp,dist} + t_{lumped} + t_{comm}, \tag{13}$$

where $t_{comp,dist}$ denotes the time for the calculations at the finer levels where the problem is distributed among all the processors, and t_{lumped} denotes the calculation time at the coarser levels where all the processors solve the same problem; these quantities do not include any communication time. Rather this time is t_{comm} which denotes the overall overhead in communication. The parallel efficiency for both the 'V' and 'W' cycles obtained from equation 13 is shown in figure 1 and compared with the efficiency for calculations performed only on the finest grid. It may be noted that the degradation in the efficiency of the 'W' cycle calculations is worse than that of the 'V' cycle calculations. Thus for the specific case considered here, the simpler 'V' cycle calculations are more effective; on the other hand, as indicated by the results in Table 2, if the size of the coarsest grid is sufficiently large, the 'W' cycle is likely to be the appropriate choice [6].

Next, consider the implicit calculation which employs the FAS multigrid algorithm with a 'V' cycle. Once again, the cavity flow test case is considered with $Re = 10^4$ and $\Delta t = 10^{-2}$, and the WU's shown in Table 3 are averaged over the first 10 time steps. The residual tolerance errors are as indicated in Table 1. A substantial reduction in computational effort may be noted as the number of grid levels is increased which becomes more pronounced as the number of mesh points in the finest grid is increased.

The parallel efficiency for the implicit calculations, which uses the 'semi-distributive' scheme at the coarsest levels, is obtained from

$$\eta_{par} = t_{tot}^{-1} \left[t_{comp,dist} + \sum_{k=1}^{M_s} \frac{(t_{comp,semi})_k}{2^{2(M_s-k-1)}} \right], \quad t_{tot} = t_{comp,dist} + \sum_{k=1}^{M_s} (t_{comp,semi})_k + t_{comm}, \tag{14}$$

where M_s is the number of 'semi-distributive' levels and $(t_{comp,semi})_k$ is the computational time at the kth 'semi-distributive' level; $k = M_s$ is the finest 'semi-distributive' level. Figure 2 shows the parallel efficiency of the implicit calculations where the full complement

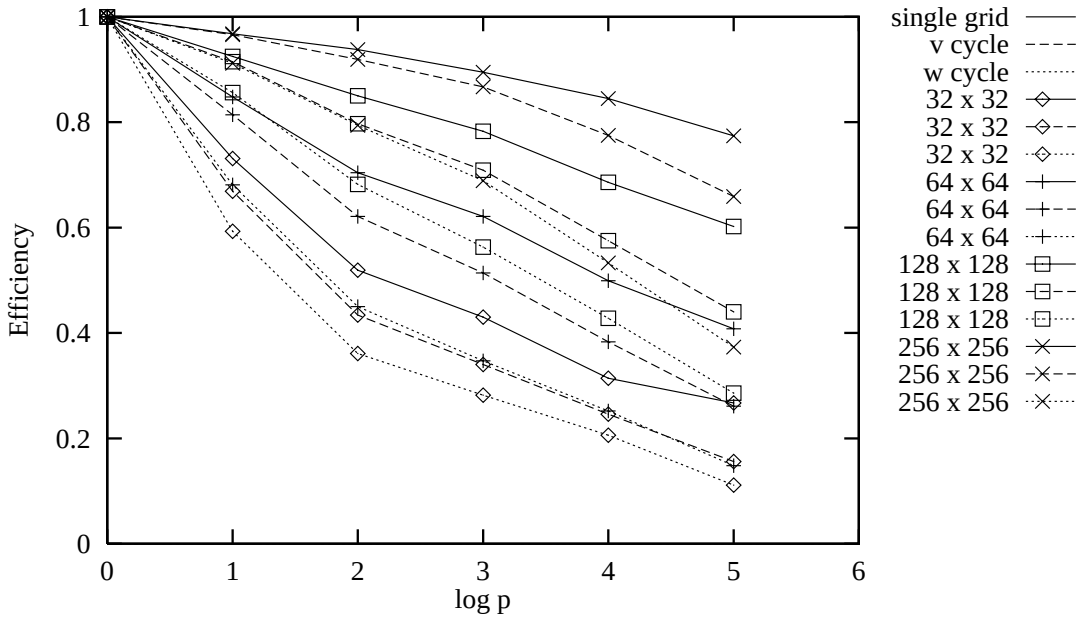

Figure 1. Variation of parallel efficiency with number of processors for single-grid and multigrid calculations for the explicit scheme.

of available multigrid levels is employed. The parallel multigrid efficiency is also compared to the efficiency of the calculations at the finest grid and it may be noted that the expected degradation in efficiency of the multigrid calculations is small for relatively large problems in comparison to calculations at the finest grid.

5. CONCLUSIONS

The parallel multigrid time-accurate calculation of the unsteady incompressible Navier-Stokes equations has been considered using both explicit and implicit methods. For both solution methods, it is clearly demonstrated that the computational effort required to obtain a converged solution at each time step reduces dramatically with increasing number of multigrid levels. Thus, although the parallel efficiency of the multigrid calculations is inferior to that possible for calculations only on the finest grid, the considerably superior convergence rates possible with the former dominates the degradation in parallel efficiency.

REFERENCES

1. R. Peyret and T. D. Taylor. *Computational Methods for Fluid Flow.* Springer-Verlag, 1983.
2. A. Brandt. *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics.* Number 85 in GMD-Studien. Gesellschaft fur Mathematik und Datenverarbeitung MBH, Bonn, 1984.

Table 3
Work units for implicit calculations on a processor mesh of 8×4 ($Re = 10^4, \Delta t = 10^{-4}$).

Levels	Global Grid			
	32×32	64×64	128×128	256×256
1	20	-	-	-
2	8	99	-	-
3	5	29	391	-
4	5	14	96	-
5	-	13	24	244
6	-	-	18	58
7	-	-	-	28

3. A. T. Degani and G. C. Fox. Parallel Computation of the Unsteady Incompressible Navier-Stokes Equations using Multigrids. In *12th AIAA CFD Conference*, AIAA Paper 95-1696, San Diego, CA, June 19-22, 1995.
4. S. P. Vanka. Block-Implicit Multigrid Solution of Navier-Stokes Equations in Primitive Variables. *Journal of Computational Physics*, 65:138–158, 1986.
5. A. Brandt. Multi-level Adaptive Solutions to Boundary-Value Problems. *Mathematics of Computation*, 31(138):333–390, 1977.
6. P. I. Crumpton and M. B. Giles. Parallel Unstructured Multigrid using OPlus. In *Parallel CFD 95*, Pasadena, CA, June 26-28, 1995.

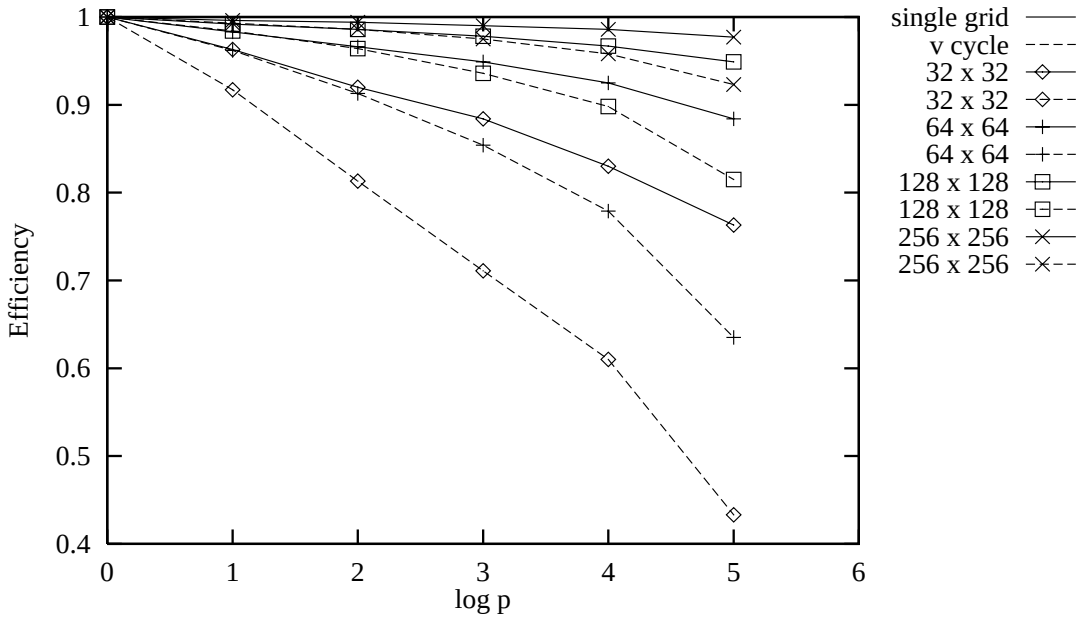

Figure 2. Variation of parallel efficiency with number of processors for single-grid and multigrid calculations for the implicit scheme.