INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 24, 1449–1461 (1997)

PARALLEL FINITE ELEMENT CALCULATION OF FLOW IN A THREE-DIMENSIONAL LID-DRIVEN CAVITY USING THE CM-5 AND T3D

ANDREW YECKEL, JACOB W. SMITH, AND JEFFREY J. DERBY*

Department of Chemical Engineering and Materials Science, and Army High Performance Computing Research Center, University of Minnesota, Minneapolis, MN 55455, U.S.A.

SUMMARY

Steady flows in a three-dimensional lid-driven cavity at moderate Reynolds number are studied using various methods of parallel programming on the Cray T3D and Thinking Machines CM-5. These three-dimensional flows are compared with flows computed in a two-dimensional cavity. Solutions at Reynolds number up to 500 agree well with the experimental data of Aidun *et al.* (*Phys. Fluids A*, **3**, 2081–2091 (1991)) for the location of separation of the secondary eddy at the downstream wall. Convergence of the three-dimensional problem using GMRES with diagonal preconditioning could not be obtained at Reynolds number greater than about 500. We speculate that the source of the difficulty is the loss of stability via pitchfork and Hopf bifurcations identified by Aidun *et al.* The relative performance of various methods of message passing on the Cray T3D is compared with the data-parallel mode of programming on the CM-5. No clear advantage between machines or message-passing methods is distinguished. Ⓒ1997 by John Wiley & Sons, Ltd.

Int. J. Numer. Meth. Fluids, **24**: 1449–1461, 1997

No. of Figures: 8. No. of Tables: 2. No. of References: 32.

KEY WORDS: parallel finite element; three-dimensional; incompressible; steady; flow

1. INTRODUCTION

Massively parallel computing platforms and algorithms hold great promise for analysing a spectrum of issues involving fluid mechanics and transport phenomena in systems where three-dimensional and time-dependent behaviours predominate. Such behaviour is characteristic of many materials-processing systems, such as those employed to produce large, single crystals for photonic and electronic devices.[1–3] It is the study of these systems that drives our development of parallel algorithms.[4–8]

In this paper we aim to compare the relative performance of the Thinking Machines Corporation Connection Machine 5 (CM-5) and the Cray T3D. Both are massively parallel, distributed memory supercomputers. For this comparison we choose to compute a classical problem, a three-dimensional lid-driven cavity flow, which is representative of the non-linear flows found in many materials-processing systems of interest to us. While two-dimensional lid-driven flow has been extensively analysed using a wide variety of techniques (see e.g. References 9–17), relatively few three-

*Correspondence to: J.J. Derby, Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, MN 55455, U.S.A.

CCC 0271–2091/97/121449–13 $17.50

Received 12 July 1996
Revised 17 September 1996

dimensional analyses have been performed.[18–21] This problem choice is also prompted by experimental observations of interesting three-dimensional flow structures in this simple system.[22–24]

While the numerical analysis of two-dimensional and axisymmetric flow is routine today, understanding three-dimensional, time-dependent, incompressible fluid flows remains a daunting task via numerical analysis. A simple illustrative calculation bears out some of the challenges associated with three-dimensional computations. Consider the numerical calculation of a steady flow field over a two-dimensional domain with N grid points along each edge. We will also consider that there are five degrees of freedom at each grid point to represent three components of the velocity field, the fluid pressure field and the temperature field. Obviously there are $5N^2$ mathematical degrees of freedom in this representation. Since this is a non-linear problem, Newton iteration is commonly applied to the discrete equations that arise after application of some numerical approximation of the original equations. The Jacobian matrix that arises from this technique is unsymmetric and indefinite owing to the underlying nature of the incompressible Navier–Stokes equations. As such, direct solution methods have been gainfully applied to the resulting linear algebraic problem.[4] A simple bookkeeping scheme results in a sparse, structured Jacobian matrix which has a half-bandwidth of order $5N$. The memory requirements of this approach (storing the entire Jacobian matrix) scale as $25N^3$, and the computational effort expended using a direct solution technique, which scales as the size of the system times the half-bandwidth squared,[25] is approximately $125N^4$ floating point operations.

Applying the identical approach to a three-dimensional problem yields strikingly different results. Consider now a three-dimensional cube with N grid points per edge, yielding a total of $5N^3$ mathematical unknowns. The resulting Jacobian matrix of this system has a bandwidth of order $5N^2$. The memory requirements of this approach (again storing the entire Jacobian matrix) scale as $25N^5$, and the computational effort expended using a direct solution technique scales as $125N^7$ operations.

The implications of the different scalings of the above approach are made clear by the results computed in Table I, where several system sizes are considered. Note that two-dimensional problems with $N=50$–200 are feasible for a large vector computer of today, such as the Cray C90. However, both memory and execution effort are prohibitively large even in the $N=50$ three-dimensional

Table I. Simple scaling of computations for problem discussed in text

	$N=50$	$N=100$	$N=200$
Total number of unknows			
Two-dimensional problem	$1{\cdot}25\times10^4$	5×10^4	2×10^5
Three-dimensional problem	$6{\cdot}25\times10^5$	5×10^6	4×10^7
Memory required[a]			
Two-dimensional problem	25 Mb	200 Mb	$1{\cdot}6\times10^3$ Mb
Three-dimensional problem	62·5 Gb	2×10^3 Gb	$6{\cdot}4\times10^4$ Gb
Operations required per iteration			
Two-dimensional problem	$7{\cdot}81\times10^8$	$1{\cdot}25\times10^{10}$	2×10^{11}
Three-dimensional problem	$9{\cdot}77\times10^{13}$	$1{\cdot}25\times10^{16}$	$1{\cdot}6\times10^{18}$
Time required per iteration[b]			
Two-dimensional problem	1·56 s	25 s	400 s
Three-dimensional problem	$1{\cdot}95\times10^5$ s (2·26 days)	$2{\cdot}5\times10^7$ s (289 days)	$3{\cdot}2\times10^9$ s (101 years)

[a] Mb = megabyte = 10^6 bytes; Gb = gigabyte = 10^9 bytes.
[b] Assumes sustained calculation rate of 500 Mflops.

problem and the scaling behaviour shows the total inadequacy of this approach applied to three-dimensional problems.

Of course, the comparison above is somewhat naive. One can improve the scalings associated with memory and operations considerably by the use of more sophisticated bookkeeping techniques. In fact, Karypis and Kumar[26] have shown that renumbering to minimize fill-in during the factorization of the matrix can result in a full order-N saving in both memory and operations. However, the three-dimensional problem still scales as memory to the N^4 and operations to the N^6, which still poses formidable, if not insurmountable, challenges for numerical analysis of such problems.

Clearly a different approach is needed for the solution of three-dimensional problems. The most promising approach to these problems hinges on advances in new computer hardware, massively parallel supercomputers, and novel algorithms and implementations on such hardware, particularly iterative matrix solution techniques coupled with appropriate preconditioners and strategies to minimize communications costs. Tezduyar *et al.*[27] highlight these approaches in a recent review article; our own efforts in this area have been documented in References 5 and 6. Our approach is briefly discussed in the next sections, followed by results for the three-dimensional lid-driven flow and comparisons of implementations on the Cray T3D and the Thinking Machines CM-5.

2. FORMULATION AND IMPLEMENTATION

2.1. Governing equations

We consider steady flows governed by the three-dimensional Navier–Stokes equations written for an incompressible fluid. This approach yields the equations

$$\rho \mathbf{v} \cdot \nabla \mathbf{v} = \nabla \cdot \mathbf{T}, \tag{1}$$

$$\nabla \cdot \mathbf{v} = 0, \tag{2}$$

where $\mathbf{v}$ is the fluid velocity and $\nabla \equiv (\partial/\partial x)\mathbf{e}_x + (\partial/\partial y)\mathbf{e}_y + (\partial/\partial z)\mathbf{e}_z$ is the gradient operator (where $\mathbf{e}_i$ denotes unit co-ordinate vectors). The stress tensor $\mathbf{T}$ is given in terms of the dynamic pressure p, and the deviatoric stress τ as

$$\mathbf{T} = -p\mathbf{I} + \tau, \tag{3}$$

where the deviatoric stress of a Newtonian fluid, τ, is linearly proportional to the velocity gradient:

$$\tau = \mu(\nabla \mathbf{v} + \nabla \mathbf{v}^{\mathrm{T}}), \tag{4}$$

with μ representing the viscosity of the fluid.

While we have often been concerned with coupled flows driven by buoyancy effects,[6–8] here we consider only the driving force of the moving lid of the enclosing box. Along all surfaces of this box, no-slip velocity conditions are applied.

2.2. Galerkin finite element method

The Galerkin finite element method[28] is used to spatially discretize the above equations. Since we are interested in computing flows of only moderate intensity, stabilization techniques for the advection terms are not employed. We employ a standard mixed interpolation scheme.[29] The velocity and temperature fields are expressed as linear combinations of Lagrangian triquadratic polynomials

Φ with 27 nodes per hexahedral element, while the pressure is approximated by a sum of discontinuous linear basis functions Γ with four degrees of freedom per element:

$$\begin{bmatrix} v_x(x,y,z) \\ v_y(x,y,z) \\ v_z(x,y,z) \end{bmatrix} = \sum_{i=1}^{N} \begin{bmatrix} v_x^{(i)} \\ v_y^{(i)} \\ v_z^{(i)} \end{bmatrix} \Phi(x,y,z), \tag{5}$$

$$P(x,y,z) = \sum_{i=1}^{N_p} P^{(i)} \Gamma(x,y,z), \tag{6}$$

where N is the total number of nodes and N_p is the number of pressure unknowns.

We apply the Galerkin procedure in the standard manner to produce weak-form weighted residual equations. Boundary conditions are invoked using routine finite element procedures.[28] The weighted residual equations are evaluated numerically using 27-point Gauss quadrature on each element to yield a large set of non-linear algebraic equations which we denote as

$$\mathbf{R}(\mathbf{x}) = \mathbf{0}, \tag{7}$$

where $\mathbf{R}$ is the residual equation and $\mathbf{x}$ is the vector of unknowns comprising the complete set of velocity and pressure interpolants.

Equation (7) is solved using Newtonian iteration. An initial guess of the vector of unknowns is made, $\mathbf{x}^{(0)}$, and successive updates to the vector of unknowns are computed using the iterative scheme

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)}, \tag{8}$$

where k is the iteration counter. The update vector $\boldsymbol{\delta}^{(k)}$ is generated by solution of the linear equation set

$$\mathbf{J}(\mathbf{x}^{(k)})\boldsymbol{\delta}^{(k)} = -\mathbf{R}(\mathbf{x}^{(k)}), \tag{9}$$

where $J_{ij} = \partial R_i / \partial x_j$ are elements of the Jacobian matrix.

To terminate the Newton iteration, it is necessary to judge when a solution to equation (7) is converged. The criterion we use here is that the maximum entry in the update vector $\boldsymbol{\delta}^{(k)}$ be less than 10^{-4} (as compared with a maximum velocity of one).

2.3. Implementation

The algorithm described above is implemented on the Thinking Machines CM-5 and Cray T3D; both are distributed memory, multile-processor supercomputers. For the sake of brevity we present only the most essential aspects of the parallel implementation here. Interested readers should consult Reference 5 for more details.

In order to effectively exploit the features of these machines, individual finite element data sets are mapped to processors and the element-level components of the residual equations and Jacobian matrix (which arise from the Newton iterations) are calculated concurrently. When these computations are complete, the GMRES (generalized minimal residual) iterative scheme of Saad and Schultz[30] is used with diagonal preconditioning to solve the linear algebraic system. To take advantage of the local data structure described above, the matrix–vector multiplications of GMRES are conducted with element-level rather than global residual and Jacobian matrix elements. Therefore only the resulting update vectors need to be mapped to the global level.[5]

This approach sidesteps many of the scaling problems discussed in Section 1. By storing only elemental pieces of the Jacobian matrix, the global Jacobian need never be actually assembled, and

issues of bookkeeping and matrix structure are no longer a problem. In addition, any global Jacobian will require some zero elements to be stored; these zero elements are not needed by the iterative solver and are not stored in the local Jacobian matrices. Therefore even relatively simple-minded, elemental-based data structures lead to more efficient memory utilization for parallel methods. The operation count associated with successful iterative solution strategies scales with the effort needed in a simple matrix–vector multiply or simply the total number of non-zero elements in the Jacobian. For our element-based data structure this scaling should be of order N^3; however, exceptions to this ideal behaviour will be briefly mentioned in Section 4.

An issue of importance in any parallel implementation is that of communication overhead. The cost of communicating between the local and global levels (scattering and gathering) is a function of the positions of the elements with respect to each other on the processors. In this study we follow the simple strategy of our initial approach[5] which assigns the elements to processors in a completely arbitrary manner. While this strategy can be bettered by employing partitioning strategies,[31,32] the effect on our algorithm is not dramatic owing to the already high degree of data locality imposed by our higher-order, triquadratic finite element basis.[6]

3. RESULTS

Figure 1 shows the problem geometry and finite element mesh employed in our computations, which consists of 16,000 triquadratic elements with a total of 472,483 unknowns. Typically we solve problems with about 10^5–10^6 unknowns, so this problem size is representative of our work. The cavity length is three times the height and width. This particular cavity shape has been studied before, both experimentally[22–24] and theoretically,[18–21] so it makes a logical choice. For all the timing calculations reported here, we employed 300 GMRES iterations within each Newton iteration and have taken a total of 25 Newton iterations.

3.1. Lid-driven flow

We solved a series of steady three-dimensional flow problems using zeroth-order continuation; namely, we used the solution at a lower value of Reynolds number as an initial guess to the solution at a higher value (Reynolds number $Re \equiv VL/\nu$, where V is the velocity of the lid, L is the width of

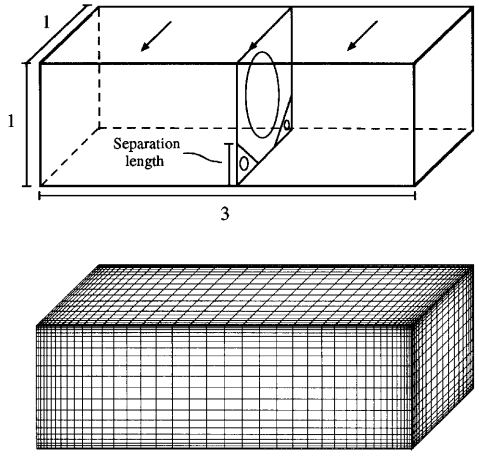


Figure 1. Domain and mesh for 3D lid-driven cavity. Arrows show direction of lid motion. Schematic diagram shows primary flow state at symmetry plane and definition of separation length for secondary eddy in downstream corner

the box and ν is the kinematic velocity of the fluid). By doing so, we were able to obtain converged solutions at *Re* up to 500. Attempts to obtain converged solutions at substantially higher values of *Re* failed, however. We also computed steady states of the two-dimensional lid-driven cavity over the same range of *Re*.

Figure 2 shows velocity vector plots of the three-dimensional flow on the symmetry plane at the centre of the cavity span (see Figure 1) at several values of *Re* (the $Re = 700$ case is not a converged solution, something we comment on shortly). Figure 3 shows velocity vector plots at various planes along the cavity span ($z = 0$ corresponds to the symmetry plane and $z = 1{\cdot}5$ corresponds to the cavity end). For comparison, Figure 4 shows velocity vector plots of the flow in a two-dimensional cavity. The flow consists of a primary vortex in the form of a cylindrical roll that spans the cavity from end to end. There also are secondary rolls caused by separation in the lower corners. Thus a two-dimensional cross-section of the flow in a three-dimensional cavity is similar to appearance to the flow in a two-dimensional cavity. Figure 3 shows that there is some departure from purely two-dimensional flow at the cavity ends, however.

Figure 5 shows a comparison of the separation length of the downstream secondary eddy (defined in Figure 1) predicted by two-dimensional and three-dimensional simulations. Also shown in the figure are estimates made by Aidun *et al.*[22] from their experiments. In each case a different method is used to estimate the location of separation. Aidun *et al.* made a visual estimate as described in their

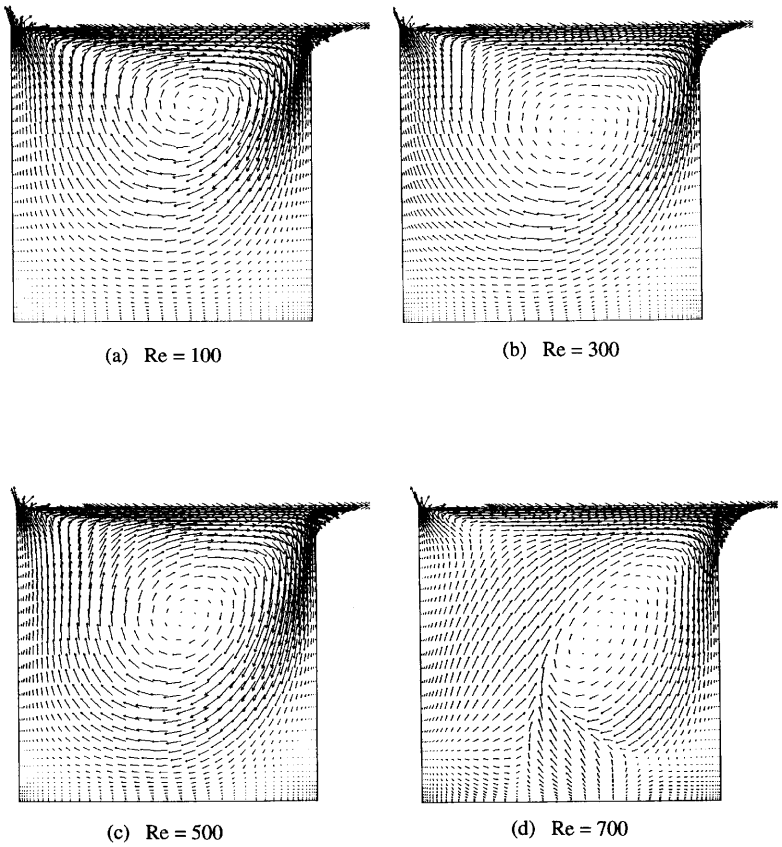


Figure 2. Velocity vector plots on symmetry plane of 3D cavity (see Figure 1) at several values of *Re*

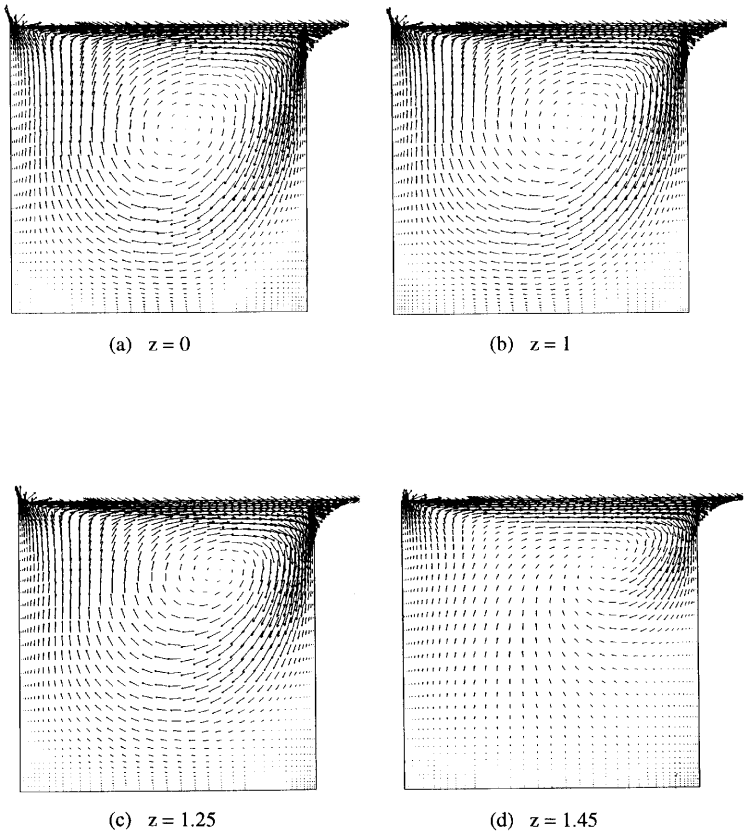
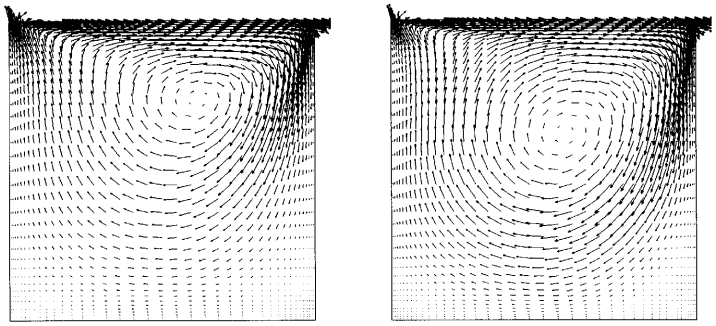


Figure 3. Velocity vector plots at various planes along cavity span ($z = 0$ corresponds to symmetry plane and $z = 1{\cdot}5$ corresponds to cavity end) at $Re = 400$

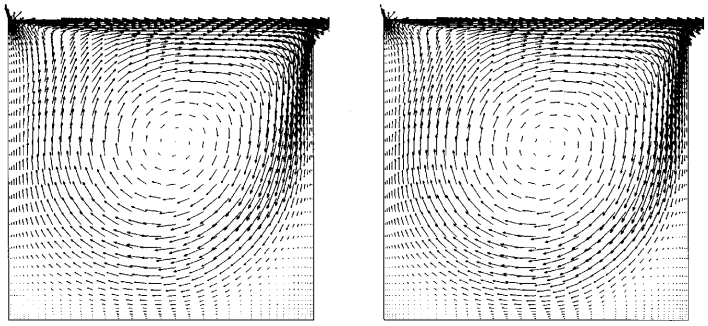
paper. In the case of the simulations we use the criterion that the vorticity passes through zero at a separation point. In the two-dimensional case we compute the vorticity using a highly accurate smoothing technique, then use a numerical method to accurately locate zero of the vorticity at the boundary (for details of this procedure, see References 16 and 17). In the three-dimensional case we measure the separation point at the symmetry plane using image plots of an unsmoothed vorticity field, a less accurate procedure. There is good agreement between our three-dimensional simulations and the experimental data, given experimental uncertainty and the approximate nature of our estimate of the separation point. There is a systematic deviation between the two-dimensional and three-dimensional simulations, but this is to be expected given the presence of end effects in the three-dimensional flow, something we discuss at length below.

At high enough *Re* a tertiary eddy appears in the corner. Figure 6 shows a comparison of the separation length of this eddy predicted by two-dimensional and three-dimensional simulations. There is a large difference between the two-dimensional and three-dimensional results. The explanation for the different is unclear, but it seems too large to attribute to approximation error. Again we presume that end effects are somehow responsible.

The comparison of two-dimensional with three-dimensional flow states is interesting, because at the limit of Stokes flow and infinite cavity span the flow on the symmetry plane of the three-dimensional cavity is identical with the flow in a two-dimensional cavity. Such is not the case if



(a) Re = 100 (b) Re = 300



(c) Re = 500 (d) Re = 700

Figure 4. Velocity vector plots in 2D cavity at several values of *Re*

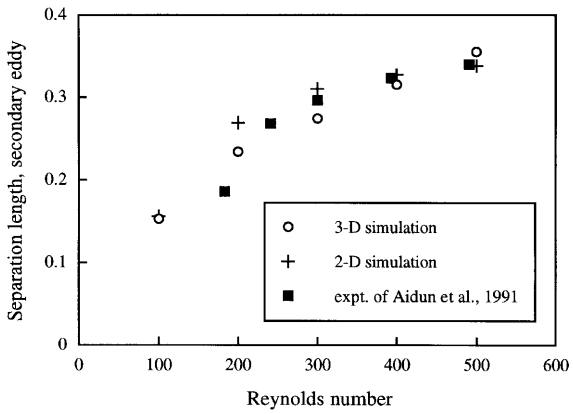


Figure 5. Separation length of downstream secondary eddy (defined in Figure 1) versus Reynolds number. Predictions of 2D and 3D simulations compared with experimental results of Aidun *et al.*[22]

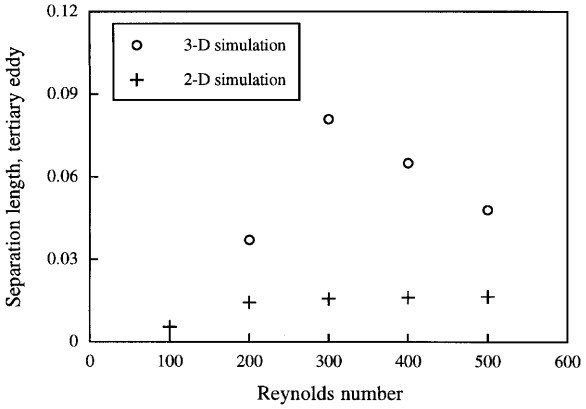


Figure 6. Separation length of downstream tertiary eddy versus Reynolds number. Predictions of 2D and 3D simulations

either the cavity span is finite or $Re > 0$. In the former case, end effects alter the flow at the symmetry plane, whereas in the latter, non-linear effects allow the possibility of other states which do not correspond to the two-dimensional flow.

To understand why the solution at the symmetry plane of the three-dimensional cavity is not necessarily a solution to the two-dimensional cavity problem, we look at equations (1) and (2). Symmetry implies that the spanwise component of velocity and the spanwise derivatives of all quantities are zero (i.e. $v_z = 0, \partial/\partial z(\cdot) = 0$). Dropping $\partial v_z/\partial z$ from equation (2) shows that the three-dimensional solution at the symmetry plane satisfies the two-dimensional continuity equation. The z-component of equation (1) reduces to $\partial^2 v_z/\partial z^2 = 0$, which is decoupled from the x- and y-components. The answer must therefore lie with the x- and y-components of equation (1). Because $v_z = 0$, the inertial terms are the same in either case. However, the viscous terms include the quantities $\partial^2 v_x/\partial z^2 = 0$ and $\partial^2 v_y/\partial z^2 = 0$, which need not be zero. In fact, the experiments of Aidun *et al.*[22] show the existence of flow states in which neither of these quantities is zero. These quantities correspond to viscous shear acting on the symmetry plane, which ultimately must be traced to the cavity ends.

A comparison of Figures 2 and 4 reveals that the flow on the symmetry plane is nearly identical with the two-dimensional cavity flow at $Re = 100$ and 300. However, at $Re = 500$ there is a significant departure between the two-dimensional and three-dimensional results. In the two-dimensional case the centre of the primary vortex moves closer to the centre of the cavity, consistent with Burggraf's[9] explanation of the tendency of this vortex to approach potential flow. In the three-dimensional case, however, the vortex centre remains at nearly the same location over the range $Re = 300$–500. The vortex becomes somewhat less circular near its centre as Re is increased, the opposite of its tendency in the two-dimensional case. Presumably the difference is a consequence of end effects.

At first thought we might surmise that the importance of end effects should diminish at higher Re because of the presumption that end effects are confined to a boundary layer. However, it appears that the influence of the cavity ends is convected towards the symmetry plane to some extent by the transverse flow and that this effect becomes more important at higher Re. This view is supported by the observation that the vortex centre is higher up and further downstream near the cavity ends that it is at the symmetry plane, as shown in Figure 3. Thus the tendency of the vortex centre to move closer to the centre of the symmetry plane at higher Re is counteracted by the convected influence of the ends to push the vortex centre higher up and further downstream.

Aidun *et al.* find that the flow state shown in Figures 2 and 3 is stable at low values of *Re* and persists at *Re* up to about 825. They also show that competing states which have Taylor–Görtler-like vortices coexist at $Re < 825$, but the actual state that is observed depends on the start-up history of the experiment. In all our calculations we have obtained converged solutions only in the case of cavity-spanning cylindrical rolls. We suspect that the difficulty in obtaining a converged solution at *Re* much above 500 is at least partly a result of the present of bifurcations at nearby values of *Re* (we rule out the possibility that the mesh is under-refined at these modest values of *Re*, based on extensive studies in the two-dimensional cavity at high Reynolds number). In the $Re = 700$ case of Figure 2 it appears that the solution is undergoing a dramatic rearrangement to another flow state, but as mentioned earlier, this solution is not converged to our satisfaction, so we can only speculate. In the future we hope to investigate this issue further by integrating the time-dependent Navier–Stokes equations.

3.2. Code performance

A schematic diagram detailing the relative costs of different communications schemes on the T3D is shown in Figure 7. Three approaches are considered. PVM (Parallel Virtual Machine) is a cross-platform software package that enables message passing across heterogeneous networks (but also can be implemented on single multiprocessor machines). MPI (Message Passing Interface) is a platform-independent standard for communications across distributed memory systems (including heterogeneous networks and multiprocessor machines). SHMEM (SHared MEMory library) is a communications library specific to Cray multiprocessor machines.

Something to consider in evaluating different message-passing schemes is the extent to which the scheme is optimized to take advantage of machine architecture and underlying native languages. PVM, for instance, is a software implementation of message passing that restricts the extent to which communications can be optimized to take advantage of a specific machine. MPI, on the other hand, is a communications standard that leaves implementation details to the machine vendor, thereby enabling better optimization than PVM. Likewise, SHMEM is a vendor-specific implementation, again allowing for better optimization than PVM.

In the light of these observations it is not surprising that SHMEM outperforms the other methods. Somewhat surprising is that PVM performs slightly better than MPI, the opposite of what we expected. Our tests were conducted with an unsupported implementation of MPI, however, which we surmise was not well optimized on the T3D. Despite these differences, all three message-passing schemes perform adequately. Even though MPI is not the fastest scheme in our tests, it seems a good choice because it is portable and its implementation is more flexible than PVM, with greater potential efficiency.

A comparison of the algorithmic times for the code implemented on the two machines is shown in Table II. The comparison is made between the CM-5 using the data-parallel mode and the T3D using SHMEM, the fastest scheme tested on that machine. Calculating the element-level Jacobian matrix

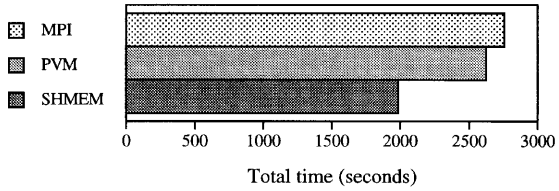


Figure 7. Comparison of solution times using various message-passing schemes on T3D (256 processors, 25 Newton iterations, 300 GMRES iterations per Newton iteration)

Table II. Comparison of T3D and CM-5 implementations (256 processors, 25 Newton iterations, 300 GMRES iterations per Newton iteration)· All times in seconds

Programme task	T3D	CM-5
Calculation of Jacobian matrix and vector of residuals	172	180
Solution of linear equation set via GMRES		
Matrix–vector multiplications	159	205
Gather/scatter communications	1032	505
Total time	1985	1132

and vector of residuals takes nearly the same amount of time on both machines. Matrix–vector multiplications in GMRES take slightly longer on the CM-5, but the difference of 20% is modest. By far the biggest difference is in the gather/scater calls, which account for the bulk of the communications cost. Here the CM-5 outperforms the T3D by a factor of two in our tests, but we emphasize that no effort was made to optimize buffer preparation and referencing (included in the gather/scatter timings); optimization potentially could reduce this performance gap.

The numbers in Table II do not tell the whole story, since the programming methods being compared are quite different. Message passing on the T3D using SHMEM, while not portable, is more nearly so than data-parallel programming on the CM-5. A well-optimized MPI library might perform nearly as well as SHMEM, and although perhaps slower than a data-parallel code on the CM-5, would be completely portable.

Besides portability and efficiency, we are concerned with how the methods scale with the number of processors. Such a comparison is shown in Figure 8. Code performance scales reasonably well on both machines. Doubling the number of processors from 256 to 512 gives a reduction of 40% in total time on the CM-5 and 35% on the T3D, compared with a theoretical maximum reduction of 50%. The slightly better scaling on the CM-5 is mostly due to better scaling of communications cost, which again is bought at the expense of portability.

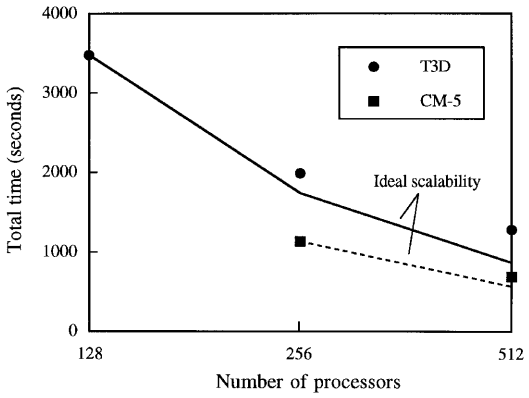


Figure 8. Scaling of solution times with number of processors on CM-5 and T3D (25 Newton iterations, 300 GMRES iterations per Newton iteration)

4. CONCLUDING REMARKS

We did not find a dramatic difference in speed between the CM-5 and T3D in our comparison. The CM-5 was nearly twice as fast overall, but most of the difference was in communications cost, bought at the expense of portability. By far a more important issue is the rate of convergence of preconditioned GMRES. Using diagonal preconditioning, GMRES converges slowly, or not at all, when solving incompressible flows, especially interior flows with much structure. Slow convergence clouds the issues surrounding operation count discussed earlier. As mentioned in Section 2.3, the operation count for iterative solution of equation (9) scales as N^3. The total operation count also scales as the total number of GMRES iterations, however. Should a great many GMRES iterations be required to obtain convergence, the method becomes unattractive.

Explicit use of the continuity equation is part of the problem. Continuity residuals give a zero entry on the diagonal of the Jacobian, which makes strict diagonal preconditioning indefinite. To fix this problem, we use a small number in place of zero for these entries in the diagonal preconditioner. The rate of convergence is quite sensitive to the value of this *ad hoc* parameter, however.[5] Use of stabilized methods such as PSPG and Galerkin least squares obviates the need to use an *ad hoc* parameter.[27] Our preliminary tests show that these methods converge faster than the conventional Galerkin method using diagonal preconditioning when applied to interior incompressible flows such as the lid-driven cavity.

It is clear that the current generation of massively parallel supercomputers makes three-dimensional flow calculations feasible. The report by Tezduyar *et al.*[27] of solving several complicated three-dimensional flows holds out great promise that such calculations will become routine within a few years. Much work remains to be done in the area of preconditioning, however, particularly preconditioners for incompressible flows which can be parallelized efficiently.

ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation under grant DMR-9058386. Computational resources were provided by the University of Minnesota Supercomputer Institute and the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory co-operative agreement DAAH04-95-2-0003/contract DAAH04-95-C-0008, the content of which does not necessarily reflect the position or policy of the government, and no official endorsement should be inferred.

REFERENCES

1. J. J. Derby, 'Theoretical modeling of Czochralski crystal growth', *MRS MRS Bull.*, **XIII**(10), 29–35 (1988).
2. J. Derby, S. Kuppurao, Q. Xiao, A. Yeckel and Y. Zhou, 'Large-scale numerical modeling of bulk crystal growth from the melt and solution', in J. P. van der Eerden and O. S. L. Bruinsma (eds), *Science and Technology of Crystal Growth*, Kluwer, Dordrecht, 1995, pp. 111–122.
3. J. J. Derby, 'Macroscopic transport processes during the growth of single crystals from the melt', in J. P. van der Eerden and O. S. L. Bruinsma (eds), *Science and Technology of Crystal Growth*, Kluwer, Dordrecht, 1995, pp. 97–110.
4. J. J. Derby, S. Brandon, A. G. Salinger and Q. Xiao, 'Large-scale numerical analysis of materials processing systems: high-temperature crystal growth and molten glass flows', *Comput. Methods Appl. Mech. Eng.*, **112**, 69–89 (1994).
5. A. G. Salinger, Q. Xiao, Y. Zhou and J. J. Derby, 'Massively parallel finite element computations of three-dimensional, time-dependent, incompressible flows in materials processing systems', *Comput. Methods Appl. Mech. Eng.*, **119**, 139–156 (1994).
6. Q. Xiao, A. G. Salinger, Y. Zhou and J. J. Derby, 'Massively parallel finite element anlaysis of coupled, incompressible flows: a benchmark computation of baroclinic annulus waves', *Int. j. numer. methods fluids*, **21**, 1007–1014 (1995).
7. Q. Xiao and J. J. Derby, 'Three-dimensional melt flows in Czochralski oxide growth: high-resolution, massively parallel, finite element computations', *J. Cryst. Growth*, **152**, 169–181 (1995).

8. Q. Xiao, S. Kuppurao, A. Yeckel and J. J. Derby, 'On the effects of ampoule tilting during vertical Bridgman growth: three-dimensional computations via a massively parallel, finite element method', *J. Cryst. Growth*, **167**, 292–304 (1996).
9. O. R. Burggraf, 'Analytical and numerical studies of the structure of steady separated flows', *J. Fluid Mech.*, **24**, 113–151 (1966).
10. U. Ghia, K. N. Ghia and C. T. Shin, 'High-*Re* solutions for incompressible flow using the Navier–Stokes equations and a multi-grid method', *J. Comput. Phys.*, **48**, 387–411 (1982).
11. R. Schreiber and H. B. Keller, 'Driven cavity flows by efficient numerical techniques', *J. Comput. Phys.*, **49**, 310–333 (1983).
12. P. M. Gresho, S. T. Chan, R. L. Lee and C. D. Upson, 'A modified finite element method for solving the time-dependent, incompressible Navier–Stokes equations. Part 2: Applications', *Int. j. numer. methods fluids*, **4**, 619–640 (1984).
13. T. J. R. Hughes, W. K. Liu and A. Brooks, 'Finite element analysis of incompressible viscous flows by the penalty function formulation', *J. Comput. Phys.*, **30**, 1–60 (1979).
14. J. H. Morrison and M. Napolitano, 'Efficient solutions of two-dimensional incompressible steady viscous flows', *Comput. Fluids*, **16**, 119–132 (1988).
15. J. L. Sohn, 'Evaluation of FIDAP on some classical laminar and turbulent benchmarks', *Int. j. numer. methods fluids*, **8**, 1469–1490 (1988).
16. A. Yeckel, 'Multiple solutions in the lid-driven cavity', *J. Comput. Phys.*, submitted.
17. A. Yeckel, 'Flow turnaround in blocked channels at low Reynolds number', *J. Fluid Mech.*, submitted.
18. J. Kim and P. Moin, 'Application of fractional-step method to incompressible Navier–Stokes equations', *J. Comput. Phys.*, **59**, 308–323 (1985).
19. C. J. Freitas, R. L. Street, A. N. Findikakis and J. R. Koseff, 'Numerical simulation of three-dimensional flow in a cavity', *Int. j. numer. methods fluids*, **5**, 561–575 (1985).
20. C. J. Freitas and R. L. Street, 'Non-linear transient phenomena in a complex recirculating flow: a numerical investigation', *Int. j. numer. methods fluids*, **8**, 769–802 (1988).
21. C.-Y. Perng and R. L. Street, 'Three-dimensional unsteady flow simulations: alternative strategies for a volume-averaged calculation', *Int. j. numer. methods fluids*, **9**, 341–362 (1989).
22. C. K. Aidun, N. G. Triantafillopoulos and J. D. Benson, 'Global stability of a lid-driven cavity with throughflow: flow visualization studies', *Phys. Fluids A*, **3**, 2081–2091 (1991).
23. J. R. Koseff and R. L. Street, 'Visualization studies of a shear driven three-dimensional recirculating flow', *J. Fluid Eng.*, **106**, 21–29 (1984).
24. J. R. Koseff and R. L. Street, 'The lid-driven cavity flow: a synthesis of qualitative and quantitative observations', *Trans. ASME*, **106**, 390–398 (1984).
25. G. Dahlquist and A. Björck, *Numerical Methods*, transl. by N. Anderson, Prentice-Hall, Englewood Cliffs, NJm 1974.
26. G. Karypis and V. Kumar, 'A fast and high quality multilevel scheme for partitioning irregular graphs', *Tech. Rep. TR 95-035*, Department of Computer Science, University of Minnesota, 1995; also available on WWW at URL http://www.cs.umn.edu/~karypis/papers/mlevel_serial.ps.
27. T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, V. Kalro and M. Litke, 'High performance computing techniques for flow simulations', in *Solving Large-Scale Problems in Mechanics*, Wiley, New York 1996.
28. T. J. R. Hughes, *The Finite Element Method*, Prentice-Hall, Englewood cliffs, NJ, 1987.
29. P. M. Gresho, 'Contribution to Von Karman Institute lecture series on computational fluid dynamics: advection–diffusion and Navier–Stokes equations', *UCRL-92275*,, Lawrence Livermore National Laboratory, Livermore, CA, 1985.
30. Y. Saad and M. H. Schultz, 'GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems', *SIAM J. Sci. Stat. Comput.*, **7**, 856–869 (1986).
31. Z. Johan, 'Data parallel finite element techniques for large-scale computational fluid dynamics', *Ph.D. Thesis*, Department of Mechanical Engineering, Stanford University, 1992.
32. A. Pothen, H. D. Simon and L. Wang, 'Spectral nested dissection', *Tech. Rep. RNR-92-003*, NASA Ames Research Center, Moffet Field, CA, 1992.

