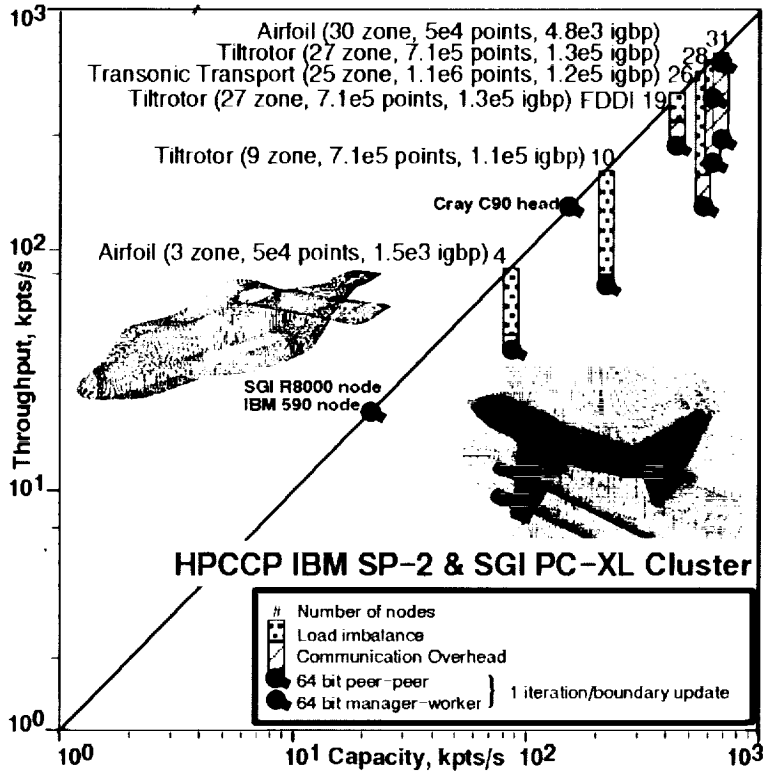
OMI
Final Report
NCC2-799

The Coupling of Fluids, Dynamics, and Controls on Advanced Architecture Computers

Christopher A. Atwood



May 1995

Overset Methods, Inc.
262 Marich Way
Los Altos, CA 94022

The Coupling of Fluids, Dynamics, and Controls on Advanced Architecture Computers

C.A. Atwood, Overset Methods Inc.

The June 1994 to May 1995 grant NCC2-799 provided for 1) the demonstration of coupled controls, body dynamics, and fluids computations in a workstation cluster environment and 2) an investigation of the impact of peer-peer communication on flow solver performance and robustness. The findings of these investigations were documented in the conference articles listed below.

The attached publication ``Towards Distributed Fluids/Controls Simulations," documents the solution and scaling of the coupled Navier-Stokes, Euler rigid-body dynamics, and state feedback control equations for a two-dimensional canard-wing. The poor scaling shown was due to serialized grid connectivity computation and Ethernet bandwidth limits. The scaling of a peer-to-peer communication flow code on an IBM SP-2 was also shown. The scaling of the code on the switched fabric-linked nodes was good, with a 2.4% loss due to communication of intergrid boundary point information. The code performance on 30 worker nodes was 1.7 μs/point/iteration, or a factor of three over a Cray C-90 head.

The attached paper ``Nonlinear Fluid Computations in a Distributed Environment," co-authored with M.H. Smith documented the effect of several computational rate enhancing methods on convergence. For the cases shown, the highest throughput was achieved using boundary updates at each step, with the manager process performing communication tasks only. Constrained domain decomposition of the implicit fluid equations did not degrade the convergence rate or final solution. The scaling of a coupled body/fluid dynamics problem on an Ethernet-linked cluster was also shown.

Publications

Atwood, C.A.,``Towards Distributed Fluids/Controls Simulations," NASA Computational Aerosciences Workshop, Santa Clara, CA, March 1995.

Atwood, C.A., and Smith, M.H.,``Nonlinear Fluid Computations in a Distributed Environment," AIAA 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1995. AIAA Paper 95-0224.

Allan, B.G., Atwood, C.A., and Packard, A.,``Control System Analysis in Nonlinear Flight Regimes," AIAA Guidance, Navigation, and Control Conference, Scottsdale, AZ, August 1994. AIAA Paper 94-3544.

TOWARDS DISTRIBUTED FLUIDS/CONTROLS SIMULATIONS

Christopher A. Atwood
Research Scientist, Overset Methods
NASA Ames Research Center, M/S 258-1
Moffett Field, California 94035-1000
atwood@nas.nasa.gov
(415)604-3974

Computational prototyping of aircraft in nonlinear regimes will require the coupling of high-order disciplines on high performance computers. This effort investigates the coupling of the Reynolds-averaged Navier-Stokes equations with six degree-of-freedom nonlinear body dynamics and controls on distributed architectures. Towards this goal, solutions obtained on a wide-node IBM SP-2, an Ethernet-linked cluster of workstations, and a Cray C90 are compared in terms of accuracy and throughput.

Physics Modelling

The coupled fluids, controls, and dynamics equations were implemented in a loosely coupled manner, as shown in Figure 1. The initialization process is performed once for each simulation, while the relative body motion cycle is completed at each time step for the serial code. During the relative body motion simulation, the integrated aerodynamic loads are passed to the body dynamics. The resultant body state is then passed to the control model, from which the new effector state is calculated by the control system. At this stage, the controller gains have been designed based on a simplified plant model or from earlier nonlinear simulations. Finally, since the relative position of the grids will have changed, the donor/receiver pairing must be re-established through a search process at each step. The OVERFLOW Reynolds-averaged Navier-Stokes code was coupled with the DCF grid communication code and Euler dynamics for this effort [1][2].

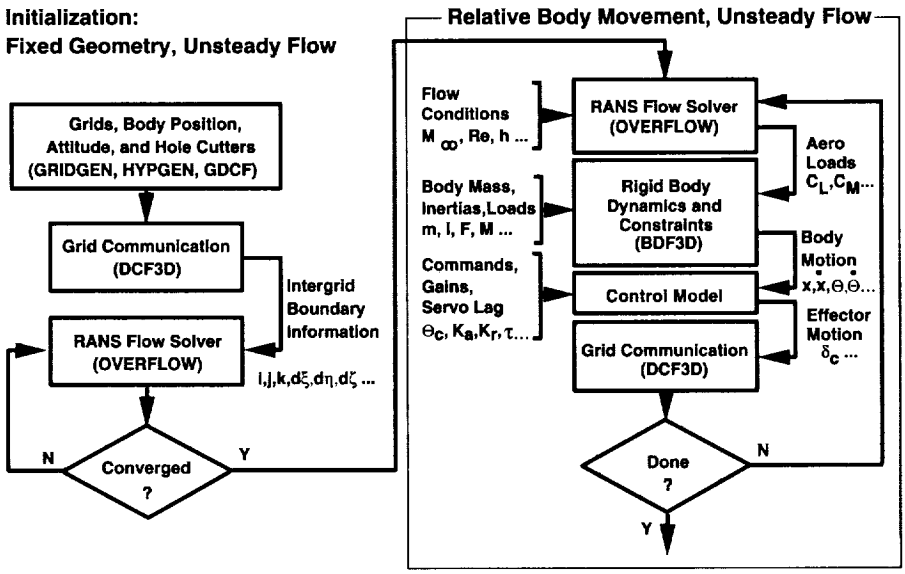


Figure 1: Coupling of Controls, Dynamics, and Fluids

Distribution Scheme

The flow solution portion of the code was distributed using domain decomposition in either a manager-worker or worker-worker communication pattern. The within-zone computation is the same for either serial or parallel versions of the code. However, the zonal boundary updates use previous iteration information, rather than the latest available quantities used by the serial solver. During the iterative solution, the dependent variables are periodically exchanged amongst connected domains using the existing Chimera format. Large zones can be further partitioned using constrained minimal surface area scheme to permit good load balance and cache behavior. This coarse grain parallelization is a rapid code port, and has advantages in terms of low communication volume and few messages per iteration since the linear solve is performed locally. However, this form of decomposition can adversely affect the convergence rate due to reduced implicitness, although Figure 2 shows that no impact was seen for a transonic foil case [3].

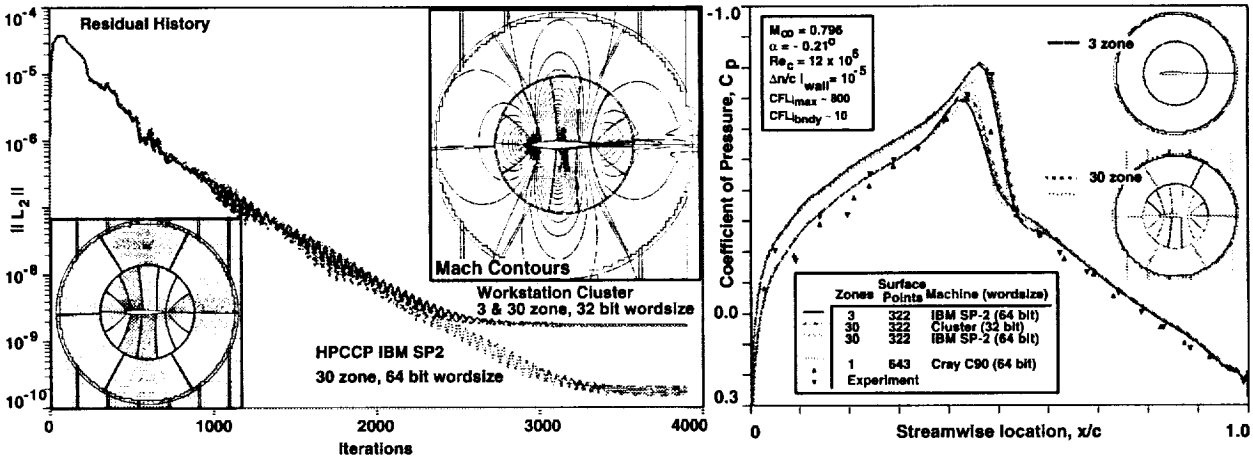


Figure 2: 64A010 Airfoil Case: Convergence History and Solution

The use of direct communication between worker processes improved code scaling on non-serialized networks, as shown in Figure 3. Performance of the HPCCP IBM SP-2, which showed computational rates of 42μs/*point/iteration* on each node, was 3.3 μs/*pt/iter* for the airfoil problem on 31 nodes with the manager-worker code. This can be contrasted to the worker-worker communication code rate of 1.7 μs/*pt/iter*, which is 97% of the load balanced peak and 84% of the ideal speedup. For the V-22 geometry, most of the communication was covered by computation, giving rates of 2.2 μs/*pt/iter* on 27 worker nodes, again leaving load imbalance as the primary contributor to the disparity from ideal speedup. By way of comparison, the throughput of serial OVERFLOW on a single-head of a Cray C90 is approximately 6 μs/*pt/iter*.

The Ethernet linked cluster of 18 workstations showed rates of 11 μs/*pt/iter* for the airfoil case, with improved performance shown when the manager process was used only to parse the boundary update messages (Ref. [3]). The performance of the code over Ethernet, a10Mbps serialized link, rapidly becomes communication bound for the transport and V-22 cases, as shown in Figure 3. For example, the 18 node V-22 case

with a boundary update at each step performs at only 50% of the load balanced level. It is expected that the use of worker-worker communication over a higher aggregate bandwidth net, such as switched and/or fast Ethernet, would significantly improve the scaling of field equation codes in general. Figure 3 also shows the improvement in computation rate gained by using infrequent boundary updates. However, a test case implemented to measure the effect of this low-communication strategy showed increased time-to-solution owing to severely degraded convergence rates (Ref. [3]).

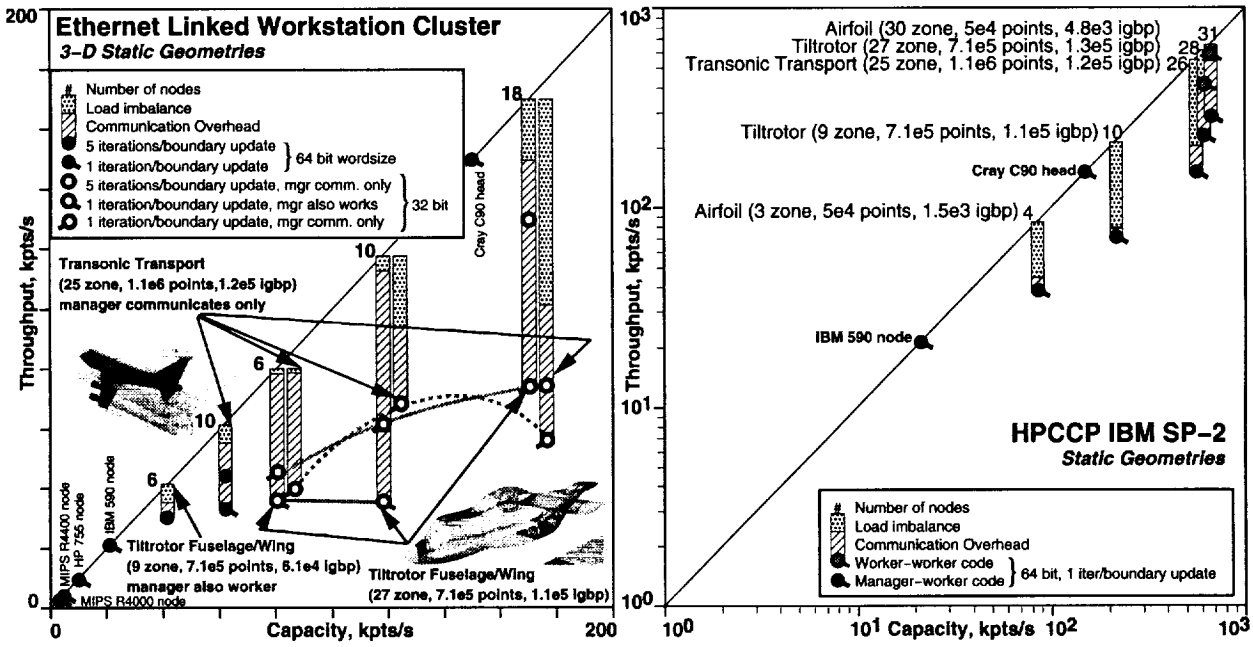


Figure 3: OVERFLOW/PVM on an Ethernet Cluster and an IBM SP-2

The grid communication code was linked to the flow solver as a separate process executable on any machine in the node pool. Although the grid communication task is performed as a single process, the donor/receiver search can be overlapped with the flow computation and/or performed infrequently at the option of the user. The effect these strategies have on throughput and accuracy is summarized in Figure 4 for two two-dimensional geometries, an airfoil under forced oscillation, and a controlled canard-wing geometry. Figure 4a demonstrates that the use of infrequent boundary updates tripled throughput for the forced airfoil case. Figure 4b shows the solutions obtained using overlapping *(lead=5)* and infrequent *(idcfrq=10)* grid communication re-establishment, with the error in attitude trajectory being 0.006 degrees.

Concluding Remarks

For this discipline-coupling effort, code development time was shortened by using message passing between high-level concurrent processes, each process reusing large amounts of serial code, while leaving potential lower-level parallelism for future linear solver schemes or compiler-level control. The use of an IBM SP-2 as a prototype tightly coupled workstation cluster has demonstrated that a highly connected node pool can surpass the throughput of a Cray C90 head for multidisciplinary applications. Given

the performance of recently introduced processors, large applications are readily feasible on workstation clusters given a high-speed interconnect. However, slow acceptance of high aggregate-bandwidth links between these powerful nodes has prevented workstation clusters from achieving full efficiency as low-cost compute servers.

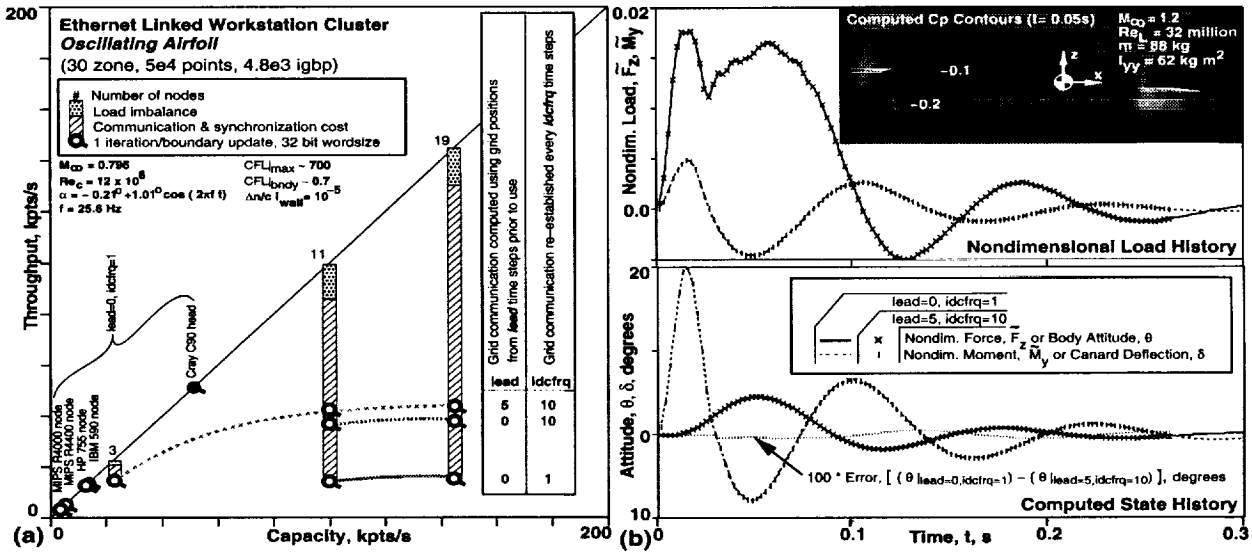


Figure 4: (a) Oscillating Airfoil and (b) Controlled Canard Wing

References

[1] Atwood, C.A., "Computation of a Controlled Store Separation from a Cavity," AIAA 32nd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1994. AIAA Paper 94-0031. http://www.nas.nasa.gov/HPCC/Pubs/atwood/mpegs.html

[2] Allan, B.G., Atwood, C.A., and Packard, A., "Control System Analysis in Nonlinear Flight Regimes," AIAA Guidance, Navigation, and Control Conference, Scottsdale, AZ, August 1994. AIAA Paper 94-3544

[3] Atwood, C.A., and Smith, M.H., "Nonlinear Fluid Computations in a Distributed Environment," AIAA 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1995. AIAA Paper 95-0224

TOWARDS DISTRIBUTED FLUIDS/CONTROLS SIMULATIONS

Christopher A. Atwood
Research Scientist, Overset Methods
NASA Ames Research Center, M/S 258-1
Moffett Field, California 94035-1000
atwood@nas.nasa.gov
(415)604-3974

Computational prototyping of aircraft in nonlinear regimes will require the coupling of high-order disciplines on high performance computers. This effort investigates the coupling of the Reynolds-averaged Navier-Stokes equations with six degree-of-freedom nonlinear body dynamics and controls on distributed architectures. Towards this goal, solutions obtained on a wide-node IBM SP-2, an Ethernet-linked cluster of workstations, and a Cray C90 are compared in terms of accuracy and throughput.

Physics Modelling

The coupled fluids, controls, and dynamics equations were implemented in a loosely coupled manner, as shown in Figure 1. The initialization process is performed once for each simulation, while the relative body motion cycle is completed at each time step for the serial code. During the relative body motion simulation, the integrated aerodynamic loads are passed to the body dynamics. The resultant body state is then passed to the control model, from which the new effector state is calculated by the control system. At this stage, the controller gains have been designed based on a simplified plant model or from earlier nonlinear simulations. Finally, since the relative position of the grids will have changed, the donor/receiver pairing must be re-established through a search process at each step. The OVERFLOW Reynolds-averaged Navier-Stokes code was coupled with the DCF grid communication code and Euler dynamics for this effort [1][2].

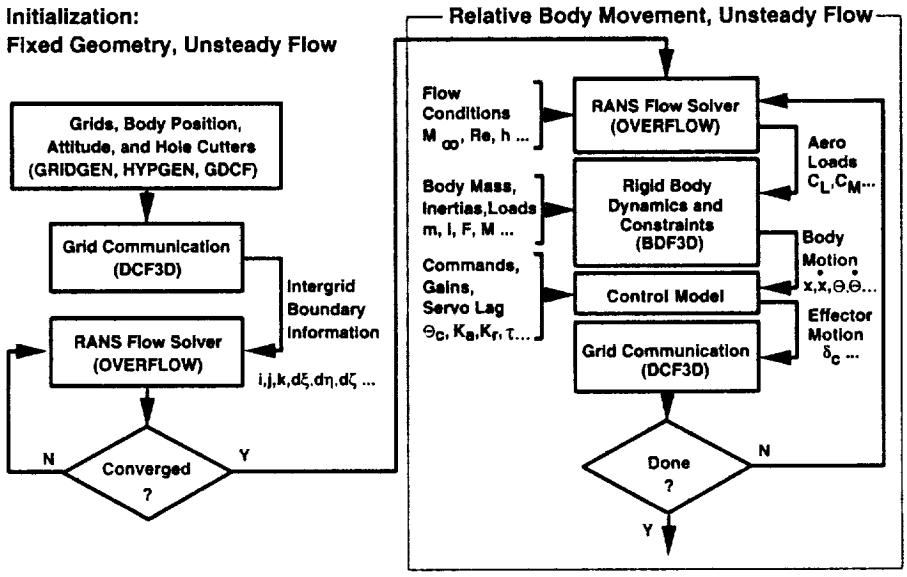


Figure 1: Coupling of Controls, Dynamics, and Fluids

Distribution Scheme

The flow solution portion of the code was distributed using domain decomposition in either a manager-worker or worker-worker communication pattern. The within-zone computation is the same for either serial or parallel versions of the code. However, the zonal boundary updates use previous iteration information, rather than the latest available quantities used by the serial solver. During the iterative solution, the dependent variables are periodically exchanged amongst connected domains using the existing Chimera format. Large zones can be further partitioned using constrained minimal surface area scheme to permit good load balance and cache behavior. This coarse grain parallelization is a rapid code port, and has advantages in terms of low communication volume and few messages per iteration since the linear solve is performed locally. However, this form of decomposition can adversely affect the convergence rate due to reduced implicitness, although Figure 2 shows that no impact was seen for a transonic foil case [3].

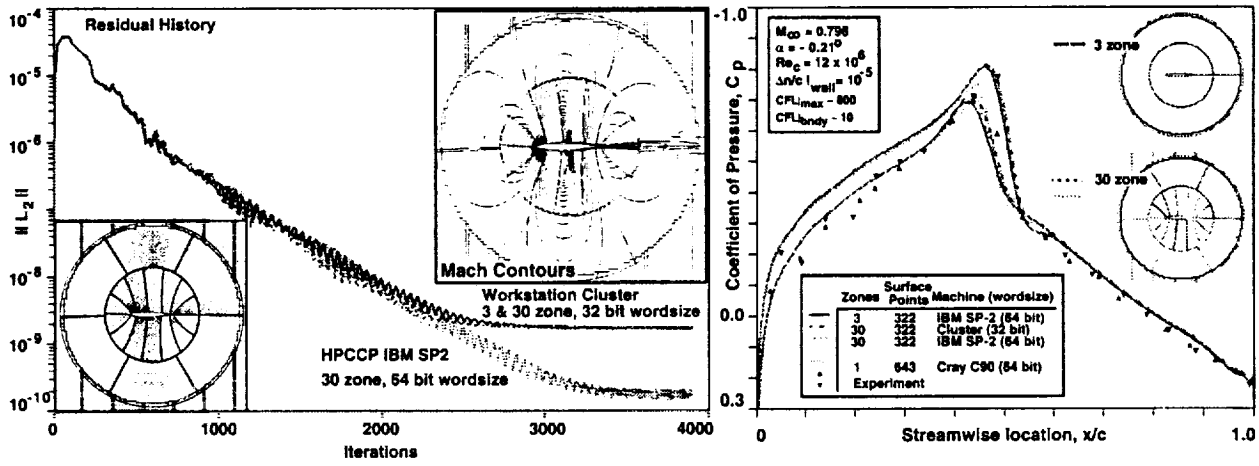


Figure 2: 64A010 Airfoil Case: Convergence History and Solution

The use of direct communication between worker processes improved code scaling on non-serialized networks, as shown in Figure 3. Performance of the HPCCP IBM SP-2, which showed computational rates of 42μs*/point/iteration* on each node, was 3.3 μs*/pt/iter* for the airfoil problem on 31 nodes with the manager-worker code. This can be contrasted to the worker-worker communication code rate of 1.7 μs*/pt/iter*, which is 97% of the load balanced peak and 84% of the ideal speedup. For the V-22 geometry, most of the communication was covered by computation, giving rates of 2.2 μs*/pt/iter* on 27 worker nodes, again leaving load imbalance as the primary contributor to the disparity from ideal speedup. By way of comparison, the throughput of serial OVERFLOW on a single-head of a Cray C90 is approximately 6 μs*/pt/iter.*

The Ethernet linked cluster of 18 workstations showed rates of 11 μs*/pt/iter* for the airfoil case, with improved performance shown when the manager process was used only to parse the boundary update messages (Ref. [3]). The performance of the code over Ethernet, a10Mbps serialized link, rapidly becomes communication bound for the transport and V-22 cases, as shown in Figure 3. For example, the 18 node V-22 case

with a boundary update at each step performs at only 50% of the load balanced level. It is expected that the use of worker-worker communication over a higher aggregate bandwidth net, such as switched and/or fast Ethernet, would significantly improve the scaling of field equation codes in general. Figure 3 also shows the improvement in computation rate gained by using infrequent boundary updates. However, a test case implemented to measure the effect of this low-communication strategy showed increased time-to-solution owing to severely degraded convergence rates (Ref. [3]).

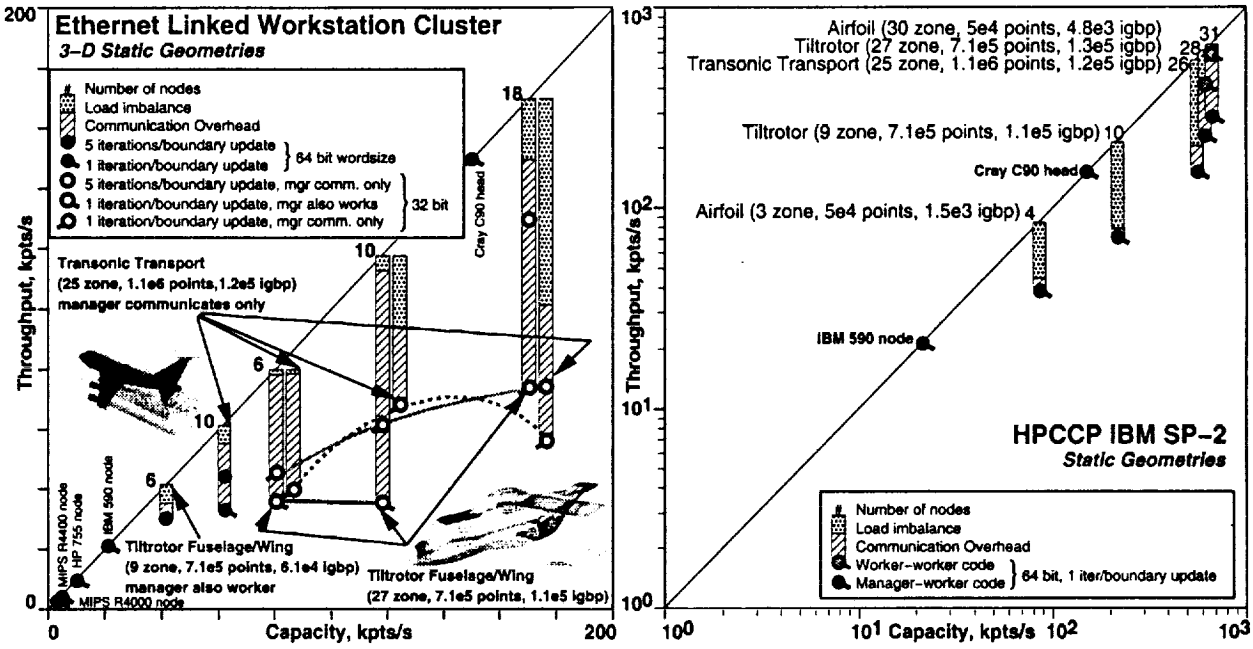


Figure 3: OVERFLOW/PVM on an Ethernet Cluster and an IBM SP-2

The grid communication code was linked to the flow solver as a separate process executable on any machine in the node pool. Although the grid communication task is performed as a single process, the donor/receiver search can be overlapped with the flow computation and/or performed infrequently at the option of the user. The effect these strategies have on throughput and accuracy is summarized in Figure 4 for two two-dimensional geometries, an airfoil under forced oscillation, and a controlled canard-wing geometry. Figure 4a demonstrates that the use of infrequent boundary updates tripled throughput for the forced airfoil case. Figure 4b shows the solutions obtained using overlapping *(lead=5)* and infrequent *(idcfrq=10)* grid communication re-establishment, with the error in attitude trajectory being 0.006 degrees.

Concluding Remarks

For this discipline-coupling effort, code development time was shortened by using message passing between high-level concurrent processes, each process reusing large amounts of serial code, while leaving potential lower-level parallelism for future linear solver schemes or compiler-level control. The use of an IBM SP-2 as a prototype tightly coupled workstation cluster has demonstrated that a highly connected node pool can surpass the throughput of a Cray C90 head for multidisciplinary applications. Given

the performance of recently introduced processors, large applications are readily feasible on workstation clusters given a high-speed interconnect. However, slow acceptance of high aggregate-bandwidth links between these powerful nodes has prevented workstation clusters from achieving full efficiency as low-cost compute servers.

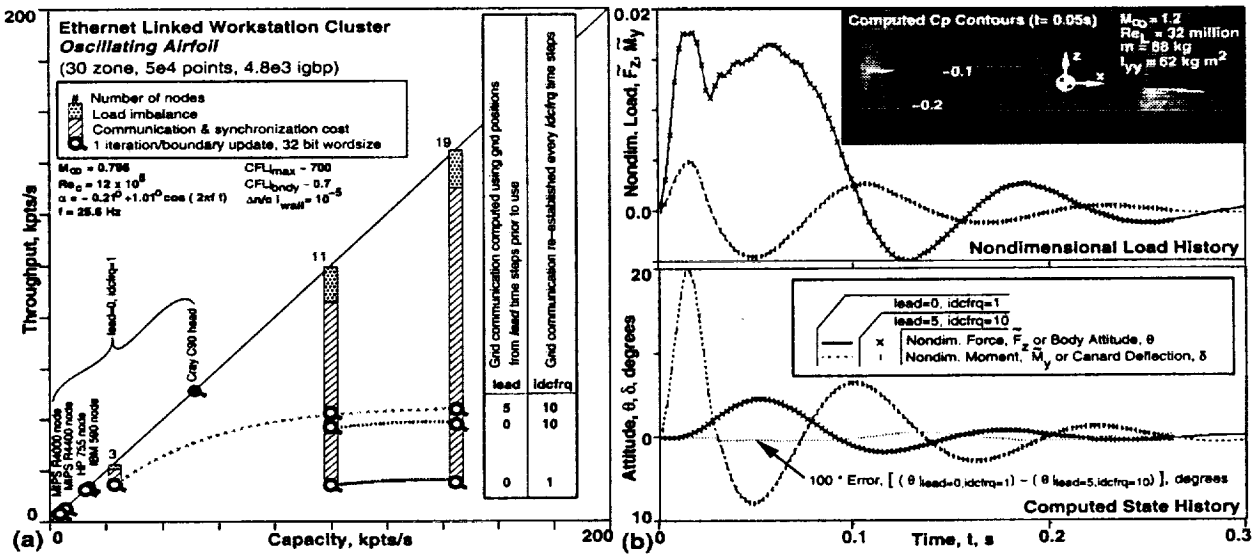


Figure 4: (a) Oscillating Airfoil and (b) Controlled Canard Wing

References

[1] Atwood, C.A., "Computation of a Controlled Store Separation from a Cavity," AIAA 32nd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1994. AIAA Paper 94-0031. http://www.nas.nasa.gov/HPCC/Pubs/atwood/mpegs.html

[2] Allan, B.G., Atwood, C.A., and Packard, A., "Control System Analysis in Nonlinear Flight Regimes," AIAA Guidance, Navigation, and Control Conference, Scottsdale, AZ, August 1994. AIAA Paper 94-3544

[3] Atwood, C.A., and Smith, M.H., "Nonlinear Fluid Computations in a Distributed Environment," AIAA 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1995. AIAA Paper 95-0224

AIAA-95-0224

Nonlinear Fluid Computations in a Distributed Environment

Christopher A. Atwood
Overset Methods
NASA Ames Research Center
Moffett Field, CA 94035-1000

Merritt H. Smith
NASA Ames Research Center
Moffett Field, CA 94035-1000

AIAA 33rd Aerospace Sciences Meeting and Exhibit

January 9-12, 1995 / Reno, NV

Nonlinear Fluid Computations in a Distributed Environment

Christopher A. Atwood*
and
Merritt H. Smith†
NASA Ames Research Center
Moffett Field, California 94035-1000

Abstract

The performance of a loosely and tightly-coupled workstation cluster is compared against a conventional vector supercomputer for the solution the Reynolds-averaged Navier-Stokes equations. The application geometries include a transonic airfoil, a tiltrotor wing/fuselage, and a wing/body/empennage/nacelle transport.

Decomposition is of the manager-worker type, with solution of one grid zone per worker process coupled using the PVM message passing library. Task allocation is determined by grid size and processor speed, subject to available memory penalties. Each fluid zone is computed using an implicit diagonal scheme in an overset mesh framework, while relative body motion is accomplished using an additional worker process to re-establish grid communication.

Introduction

It is anticipated that the high-order simulation of coupled aerodynamics, body dynamics, and controls on shared memory vector architectures will consume considerable resources.[1, 2] One approach which promises to reduce the cost of such computational prototyping is the use of distributed processor machines.

This effort investigates the use of workstation-class nodes linked by a peak 10 Megabit/s (Mbps) or 250 Mbps network. The work shown here uses a manager-worker Chimera[3] RANS solver[4] first implemented on a network of workstations[5] using the Parallel Virtual Machine (PVM) library.[9] The primary issues of concern here relate to the parallel solution of large moving body problems such as would be encountered in a controlled aircraft simulation.

In order to evaluate the performance of manager-worker computation for realistic problems, three configurations are included in the following sections. Timings for a transonic airfoil, a transonic transport,[6] and a tiltrotor[7] fuselage/wing are given. The two-dimensional airfoil case is used to characterize the effects of boundary information lagging upon convergence and throughput. Scaling effects are shown by further decomposition of the grid system while retaining the highly implicit wall-normal direction. Finally, the addition of moving-body capability is examined using an oscillating airfoil case, with differing lag levels of the intergrid boundary points (IGBP).

Approach

Flow Solver

Solutions to the Reynolds-averaged Navier-Stokes (RANS) equations were computed with the implicit diagonal scheme of Pulliam and Chaussee[8] was used, implemented in the Chimera grid framework of Steger, Dougherty, and Benek.[3] Euler implicit time marching and central second-order spatial differencing was used, with viscous wall conditions specified as no-slip, zero normal pressure gradient, and adiabatic. The Baldwin-Lomax algebraic turbulence model is used for all viscous walls. Information transfer across overset mesh boundaries was implemented using trilinear interpolation of the dependent variable vector, $Q = [\rho, \rho u, \rho v, \rho w, e]^T$. The flow solver cost is about $6.5\mu s$/cell/step on a Cray C90 head.

The flow solver Overflow-PVM has been developed from the Overflow overset grid flow solver[4] and the PVM (Parallel Virtual Machine) communication libraries. Parallelism is extracted on a coarse grid-by-grid level, and allows multiple grids to be solved on a single processor. During the run, the manager process primarily transfers flow boundary information (QBC)

*Research Scientist, Overset Methods, Inc., atwood@nas.nasa.gov. Member AIAA.

†Research Scientist, Applied Computational Aerodynamics Branch, mhsmith@nas.nasa.gov. Member AIAA.

amongst the worker nodes.

Relative grid motion is accomplished by the addition of a process to run the grid communication code, the Domain Connectivity Function (DCF).[10] The process is directly linked to the manager and worker nodes, and can be executed at a specified frequency. The grid position and speeds may be lagged a given number of steps to allow for the overlap of the flow solver with the computation of the grid communication.

Grid Decomposition

Scaling studies were performed using zone decomposition of the initial overlapping grid system. Given a user-supplied number of subdomains, the decomposition scheme generates a minimal surface area set of grids and an input file for the DCF grid communication code. A search for a nearby decompositions which result in fewer surface points is also performed. The decomposition is constrained not to divide in the ζ-direction (η in two-dimensions), which is typically the most implicit direction for wall-bounded viscous flows. This constraint also simplifies the search required by the algebraic turbulence model. The resultant set of grids contain one cell overlap with neighboring grids.

Load Balancing

A static load balancing algorithm has been developed to best match grids with processors such that idle time due to load imbalance is minimized.[5] This first implementation of Overflow-PVM initiated solution on all grids simultaneously, including the case of multiple grids on a single node. Two problems with this approach became apparent for the solution of large problems on a limited cluster. Firstly, the context switching of the UNIX operating system caused system time overhead which is wasted on an otherwise idle processor. Secondly, and more importantly, solution of multiple grids in core can consume available RAM, causing swapping of processes to disk. In order to remove or reduce these losses, a modification to the control structure was made such that only a single grid can be active on each processor of the cluster. This drastically reduces context switching between worker processes, and swapping will only take place at the beginning and end of solution on each grid. Finally, the use of multithreaded code compilation and execution on shared-memory multiprocessor, and generally multiuser, machines was implemented.

In addition to the above improvements, a static load balancing method accounting for available RAM was implemented using the integer branch and bound method of the Microsoft Excel Solver. The method allows a cluster user to input available memory on each node and estimated network bandwidth, latency, and contention for each node. The use of virtual memory is implemented as a penalty rather than a constraint. The inclusion of a memory variable in the load balancing scheme is required when cluster nodes of differing clock rates, but similar RAM, are considered. For large problems, not accounting for memory will cause the fastest CPU's to be allocated the most work units, eventually exceeding RAM capacity and inducing slow disk-paging virtual memory use.

The load-balancing algorithm for Overflow-PVM requires only node speeds (grid points/s) and available memory (grid points). For problems where grids are in relative motion, the transfer of boundary interpolant information must be periodically re-computed. If the flow solver is to be run sequentially with the grid communication, the above load balance remains unchanged. However, for cases in which DCF can be overlapped with the flow solution, current algorithm can be used by lowering the speed of the node on which DCF and the grid solvers execute to: DCF node speed=1/ [1/(p/s)+(IGBP/p) / (IGBP/s)] where p=grid points, and IGBP=intergrid boundary points.

Non-Prescribed Relative Grid Motion

Loosely coupled body and fluid dynamics problems require the integrated fluid loads to be used as the forcing function of the body dynamics, and the updated body state to be used for boundary conditions of the fluid solver. Elimination of this serial bottleneck caused by the coupling will offer increased throughput, albeit at the sacrifice of some level of accuracy. However, a large class of problems exists for which the low ratio of body natural or forced frequency over fluid motion frequency scales indicate that lagging of boundary interpolants may not cause significant reductions in accuracy.

Figure 1 depicts a comparison of the the sequential and parallel methods. The sequential technique is shown for the case of lockstep update of the IGBP after each flow solver time step and no lead time for DCF. The parallel method overlaps DCF and Overflow by starting the DCF process *lead* steps before the updated grid communication solution will be required by Overflow. The frequency at which grid communication is re-established is given by $idcfrq$. Note that the metrics are updated every step, n, based on the load integration from the previous step.

The lagging example shown in Fig. 1a computes the integrated loads acting on the body in Overflow at step $n-1$, uses these loads to compute the new body state, sends this state to the grid communication process DCF, and also uses the new body state for the flow solver step n. At time $n+1$ the Overflow manager process waits for updated grid communication to be sent to the worker processes before signalling the start of the solution.

It is anticipated that this lagging would adversely effect accuracy when the ratio of applied+aero loads

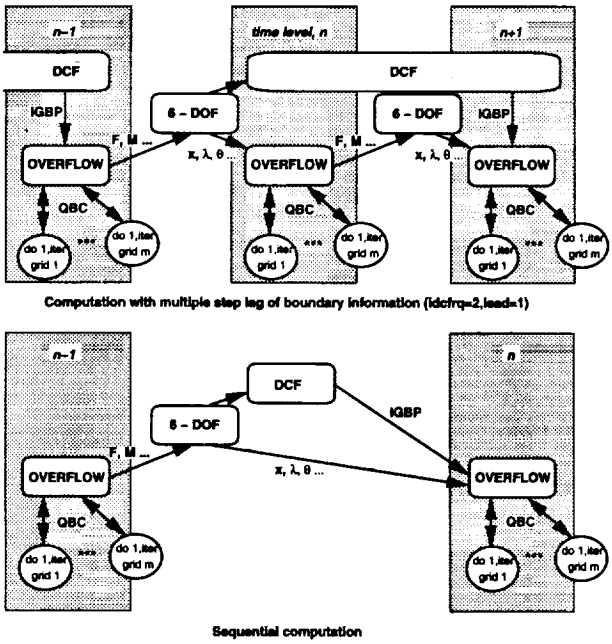


Fig. 1: Flow solver and grid communication coupling methods

to mass properties is large, i.e., accelerations are large or when fast relative grid motion exists. One disadvantage of the approach is that more total run-time nodes will be required, most dedicated to the flow solver and a few for the grid communication. However, this separation of flow solver and grid communication codes may make the coupled code more portable and maintainable because each code may be distributed the manner most appropriate.

The scheme implemented here also duplicated grid information for the DCF process, storing temporary grid and search map files in /tmp. This is done due to the requirement that the grid communication scheme generally will access all of the grids during the search for donor/receiver pairs. Accessing grid data from several nodes will incur heavy network load, competing with the flow solver nodes for the shared resource.

Results

In an effort to determine the highest throughput solution process, the Ethernet linked workstation cluster was used to evaluate the impact of lagged boundary updates and computation using 32 versus 64 bit wordsizes.

The cases were executed during low node and network use periods in single cluster-user mode on a set of cross-mounted workstations using PVM 3.2.3. No modification of the network topology was attempted, and the all of the nodes were on either of two subnets. Three gateways separated two of the pool of nodes from the master (measured using *traceroute*), with a point-to-point transmission time, shown by *ping -f* for an 8 KB datagram, of 300 ms on an otherwise idle network. This gives an effective bandwidth of 0.21 Mbps to these two distant nodes versus 0.35 Mbps measured between nodes with no intermediary gateways. For smaller datagrams, the lower collision rate increases the effective bandwidth to 8 and 2 Mbps for the near and distant nodes respectively

The peak transfer rate of Ethernet is 10 million bits per second (Mbps), although during low-use periods the observed bandwidth of the cluster is about 2-9 Mbps, which can drop to 0.2 Mbps during normal use periods. For the large packets typically used in these computations, the discrepancy between observed and peak data transfer rates can be primarily attributed to contention, PVM-required memory to buffer copies, and latency.[11]

For a typical Chimera application, approximately 10% of the total grid points are IGBP's, each of which require the dependent variable vector, Q, from another zone. In a distributed application, generally this information must be passed to another node. Overflow-PVM is implemented in a manager-worker arrangement, with the manager first parsing the grids to the worker processes, then receiving, updating, and resending large packets of boundary information at a user-specified rate. The solution from the run is collected following the final flow iteration.

Throughput timings for all of the cases were obtained either by monitoring CPU usage or by runs of 5-10 iterations, from which the zero iteration runtime was subtracted. Subtraction of start-up and wind-up costs also allowed good estimates for long-duration runs on an otherwise idle network. Data conversion (XDR) was not required for this set of nodes, and disk swapping not observed for any of the cases.

The static grid version of the code was also run on the High Performance Computing and Communication Program (HPCCP) IBM SP2 located at the Numerical Aerodynamic Simulator (NAS), using the PVMe message passing library. These cases were run in single grid per node mode, with timing numbers obtained in the same manner as on the loosely coupled cluster. Switching of interruptions did not affect timings appreciably for any of the configurations.

Steady Airfoil Case

The solution of the steady transonic flow about a 64a010 airfoil, depicted in Fig. 2, was computed to 1) quantify the effect boundary information lagging upon convergence and 2) to measure the scalability and accuracy of this domain decomposition method.

Figure 3 shows the effect on convergence of decomposing the grid from 3 zones into 30, with single cell overlap. Local time step scaling based on geomet-

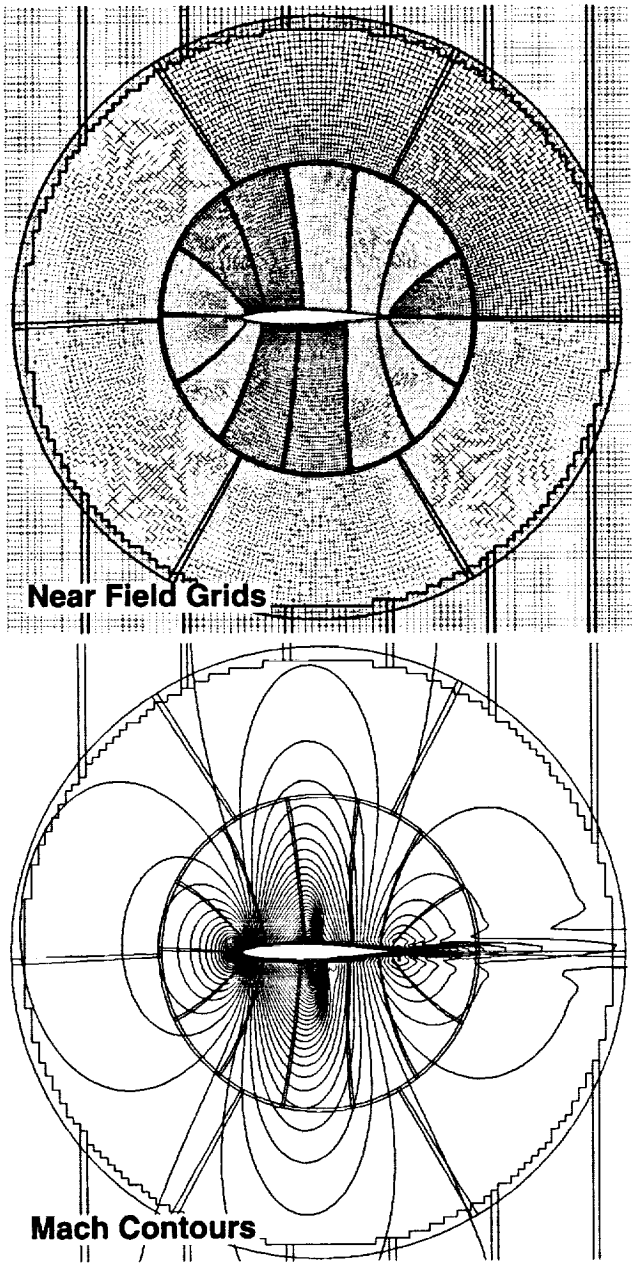


Fig. 2: 64a010 Airfoil case: grid system and Mach contours

ric considerations only was used for all of the cases: $\Delta\tau_{(i,j,k)} = \Delta\tau[1 + 0.005(1/\sqrt{V})]/[1 + 1/\sqrt{V}]$. The histories of the L2 norm for both the Ethernet linked workstation (32 bit words) and the IBM SP2 (64 bit words) are of similar trends. The differences in the high-frequency components of the history may be attributable to the elimination of fourth-order dissipation at the zone boundaries and to the inclusion of the overlapping boundary points in the computation of the residuals. The residual on the on the IBM SP2 converges an additional order of magnitude over the 32-bit wordsize cases.

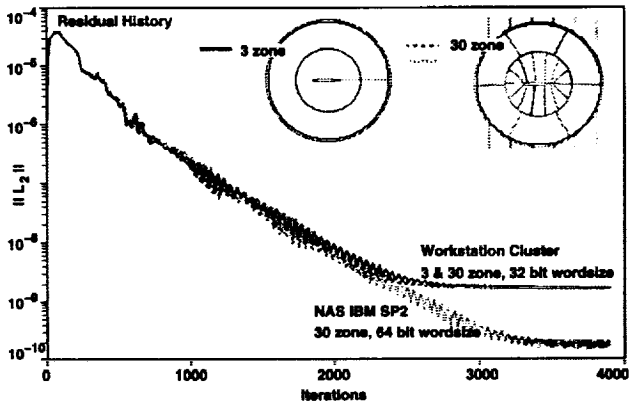


Fig. 3: 64a010 Airfoil case: Residual histories

Figure 4 compares C_p plots for the present cases compared to experiment and a high resolution single zone computation computed on a Cray C-90 (64 bit wordsize).[12] The solutions are generally in agreement, although differences exist, and are most significant near the shock.

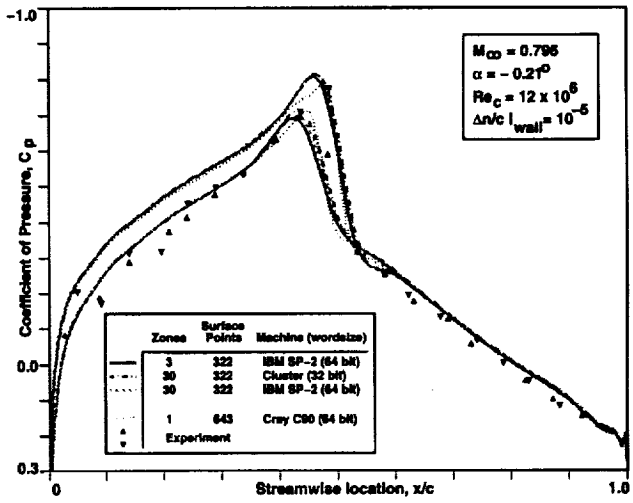


Fig. 4: 64a010 Airfoil case: Comparison of C_p vs. x/c

Figure 5 shows the speedup achieved for the foil problem over a differing number of nodes. The identity line represents ideal speedup, with speeds for single

node flow solver runs included for reference. The difference between ideal and actual speed is broken into two components. The first component contributing to lost capacity is that due to load imbalance, and is the speed the code would run given a perfect network, i.e. infinite bandwidth and zero latency. The second component contributing to the lost computational capacity is that attributable to message passing from one node to another, including time to pack (buffer copy), transmit (network latency and bandwidth), and unpack (buffer copy). The magnitude of this cost will depend both upon the number of IGBP and the ratio of surface area to volume, or IGBP/points. For this case the number of IGBP is 4827, and the total number of points is 49927. In general, the ratio of IGBP/points ranges from about 10 to 15%.

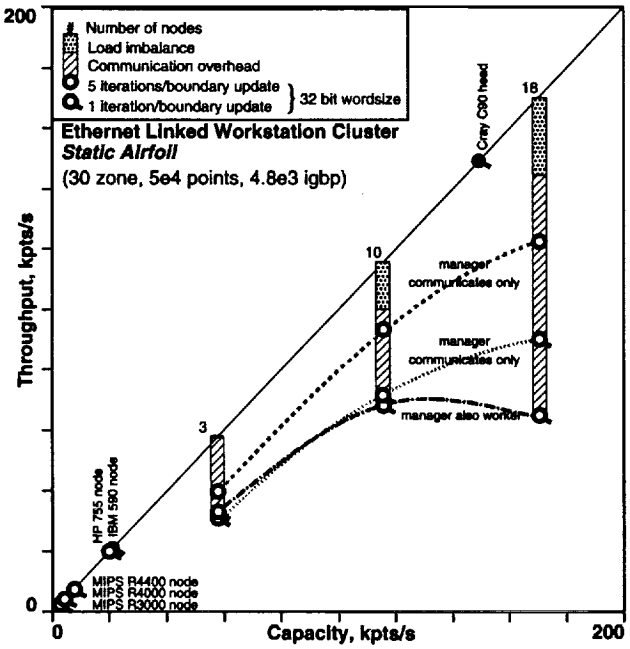


Fig. 5: 64a010 Airfoil case: Scaling study

Figure 5 also shows that it is advantageous to reserve the manager processor for communication alone for jobs where network contention is high. When the processor which contains the manager also has worker processes, the CPU is busy with the worker process when messages from completing remote processes return. The interrupted CPU then processes the incoming message before returning control to the worker process, with the concomitant adverse effect on throughput.

The difference in throughput seen between the 1 and 5 iterations per boundary Q update cases shown in Fig. 5 is indicative of the contention that is occurring for the network. Since exchange of boundary information comprises the bulk of the message traffic, decreasing this communication / computational time ratio decreases time to solution even though the number of CPU cycles is the same in either case.

An estimate of the loss in throughput due to the network can be computed from the time required to serially communicate the boundary Q information. The communication time for this geometry and precision is about 0.25 seconds: 5 words/point $\times$ 32 bits/word $\times$ 4.8 $\times$ 10^3 boundary points $\times$ 2 transmissions (one receive + one send) $\div$ 6 $\times$ 10^6 bits/second. This communication cost causes a loss in throughput of about 30 kpts/s for the 10 node case. For the 5 iteration per boundary update cases, some of this loss is covered by computation as the different processes finish with larger intervening times.

By comparison of the 5 and 1 iteration per boundary update cases, it can be seen that the cost of updating the boundary information, Q, at each step is high. For the 10 node case, the throughput would be 35% higher, provided that convergence is unaffected.

However, inspection of Fig. 6 shows the marked effect lagging of boundaries has upon convergence for the 30 zone problem. When the boundary update is lagged for a single step (iter=2) the convergence rate drops dramatically, and is monotonically worse for increased lagging. The integrated C_{l_p} includes the single cell overlap, and are presented for a comparison of the lagging levels only. Upon convergence the maximum Courant number was approximately 800 and the Courant number across the intergrid boundaries near the foil was about 10.

Figure: 7 shows the computation of the airfoil case on the IBM SP2, a tightly coupled cluster of workstations connected by a 240 Mbps peak network. The native implementation, PVMe (similar to PVM 3.2.5), of the message passing library permitted runs with only a single process per node. Although throughput for the 30 zone case exceeds that of a Cray C-90 head, the code remains communication bound, as can be seen from inspection of the 5 and 1 iteration per boundary update cases. The resultant throughput for the single update per iteration case is 40% of that achievable on an ideal network. By taking the difference in times between the perfectly load balanced case and the single iteration per boundary update, the effective bandwidth for this problem was estimated at 35 Mbps. The peak network performance for PVMe on the machine is approximately 240 Mbps for large message sizes (100 KB), dropping off to about 120 Mbps for the estimated primary message size of 8 KB for the airfoil problem.

Three-Dimensional Static Geometries

Two geometries were used to evaluate the performance of Overflow-PVM for viscous problems in three-dimensions: a tiltrotor fuselage-wing,[7] and a transonic transport.[6] Scaling of the tiltrotor geometry, shown in Figs. 8 and 9, was computed by subdivision of the 9

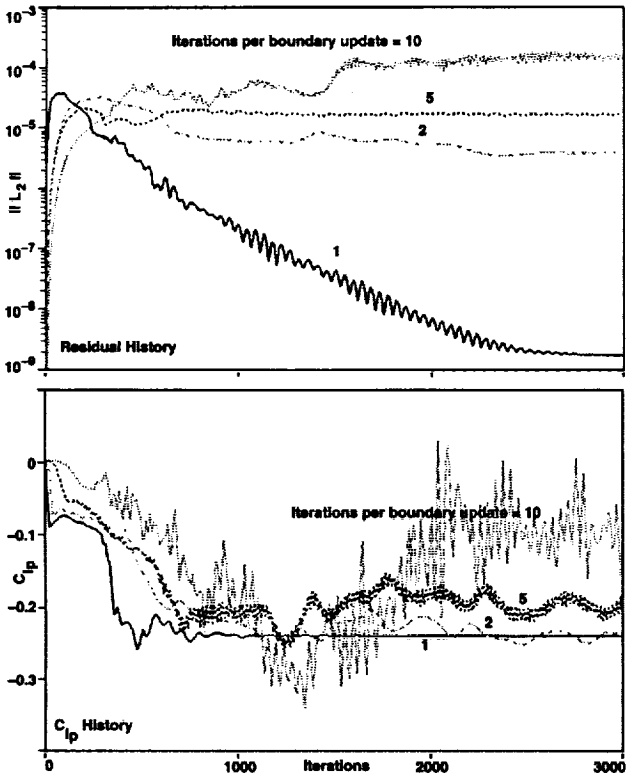


Fig. 6: 64a010 Airfoil case (30 zone): Effect of boundary lagging

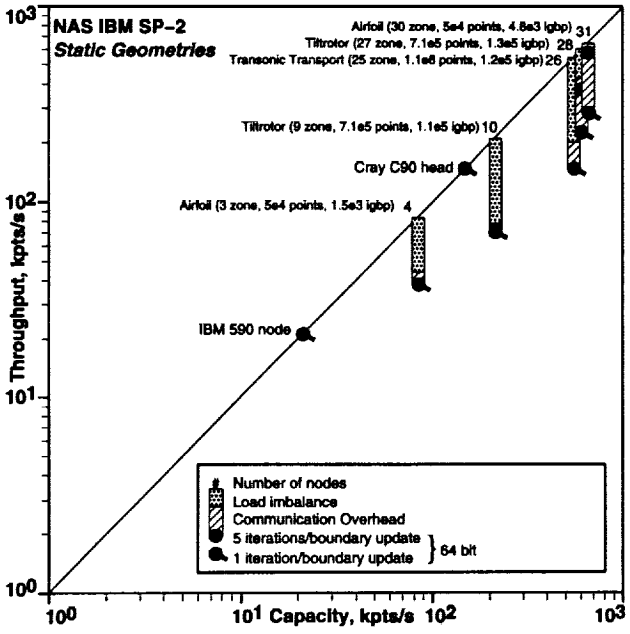


Fig. 7: NAS IBM SP2 scaling study

zone grid system into 27 zones using the same preprocessing technique as used with the airfoil. The transport case, shown in Fig. 10, did not use additional subdivision for this study, resulting in significant load imbalance for large node number cases.

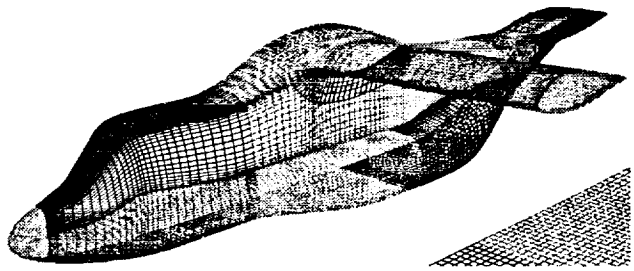


Fig. 8: Tiltrotor fuselage/wing (27 zone): decomposition of near field grids

Fig. 9: Tiltrotor fuselage/wing: developing solution

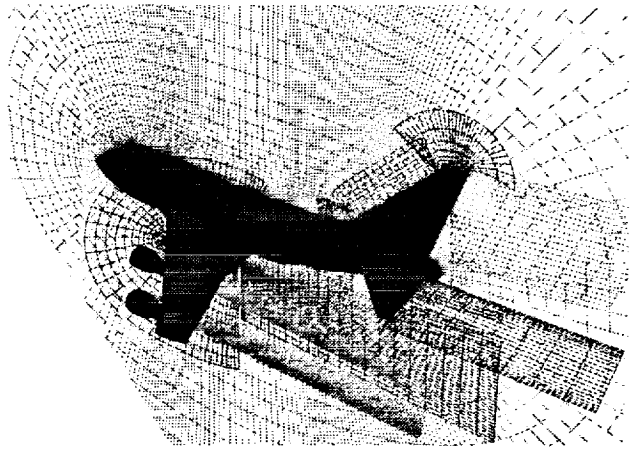


Fig. 10: Transonic transport: domain decomposition

Figure 7 shows the throughput seen on the tightly coupled cluster for these cases. Comparison of the 9 and 27 zone tiltrotor cases shows that the throughput as a fraction of capacity increases, and is higher than that of a C-90 head for the more distributed case. However, for the 9 zone configuration the loss is primarily due to load imbalance, while network associated losses dominate for the 27 zone case. Again, the 5 iteration per boundary update case is shown to exemplify the communication losses, which consume roughly half of the total time per step. The 26 node

transport case shows much of the reduction from capacity is due to load imbalance. Comparison of the 26 node transport case and the 28 node tiltrotor case shows that the load imbalance for the former partially masks the communication with computation. The 28 node tiltrotor case, with a single step per boundary Q update (QBC), shows a throughput of 51% of the load balanced capacity, while the transport case shows 80% of the throughput that would be seen on an ideal net.

A summary of the results obtained on the Ethernet linked cluster is shown in Fig. 11 for both 32 and 64 bit words. The performance of the code on single nodes is shown for 64 bit floats, with the 32 bit values from Fig. 5 applicable for these cases. The 64 bit performance of the PA-RISC and MIPS nodes used for these cases was approximately half that seen with 32 bit words. The IBM RS6000 node, which performs similarly using single or double precision, is shown for reference only.

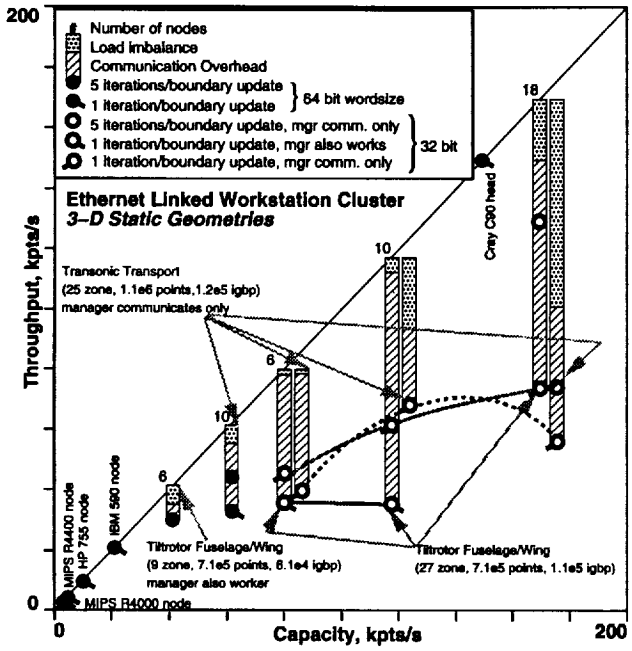


Fig. 11: Three-dimensional cases

The 64 bit transport case, which load balances to 90% efficiency, loses about 14 s per iteration to communication, implying an effective bandwidth of 5.5 Mbps. Using 5 iterations per boundary update decreases the communication overhead to 6 s. The nine zone tiltrotor case executed at 70% capacity using 5 iterations per update.

The 32 bit float tiltrotor cases given in Fig. 11 for 6 and 10 node machines again show the advantageous effect of using the manager node for communication alone. While the efficiency drops when the manager processor is allocated work according to processor speed, a positive trend is evident when the processor is allocated no grids. However, although throughput increases with increasing processors, the communication cost remains constant, with partial overlap of communication and computation due to higher load imbalance for the large machine cases. The communication cost is approximately 6 s for these tiltrotor cases, implying an effective bandwidth of 6 Mbps.

The 32 bit transport cases shown in Fig. 11 show increasing throughput from the 6 node to the 10 node case, but a negative trend from 10 to 18 nodes, despite the potentially communication covering high load imbalance. Using a 6 Mbps bandwidth, the communication cost for the 6, 10, and 18 node cases can be estimated at 6.4 s. Inspection of Fig. 11 shows that the difference between the ideal network cases and the 1 step per QBC update could be explained by 15 s /step communication cost. However, since the communication pattern is similar between the airfoil and tiltrotor cases, that is a large messages, a loss in effective bandwidth is not expected. The poor performance of the transport cases could be caused by cache misses owing to the large grids of this case, effectively lowering the load balanced throughput. The largest grids for the transport case had 1.2×10^5 points, as opposed to the 3×10^4 grids used in the tiltrotor case. However, the load balance plotted for the Fig. 11 cases uses the higher single node throughput values to uniformly represent the geometries.

Relative Grid Motion Cases

An oscillating 64a010 foil was used to measure the performance of the loosely coupled cluster for the coupled fluid/grid communication code. The grid system shown in Fig. 2 was used for this case, with the 14 foil-attached zones oscillating about the mid-chord with the remaining 16 grids fixed in inertial space. The interface between the moving and static grids is circular with uniform circumferential spacing. This topology removes effect of differing grid resolution between the zones, and provides a single relative grid speed to be evaluated for the boundary lagging cases.

The mean angle of attack was $-0.21°$ with an oscillation magnitude of $1.01°$ at a frequency of 25.6 Hz. The Reynolds number was 12×10^6 at a freestream Mach of 0.796, with a wall normal spacing of 10^{-5} of chord. A time step of $\Delta t = 10.2\mu s$ was chosen to maintain a Courant number near 0.7 across the zonal boundaries, although at the viscous wall the Courant number is approximately 700. Using this time step, the number of steps required per oscillation was approximately 3830.

The single node throughput speeds along the identity line in Fig. 12 were computed using the estimate of 10% of the grid points being IGBP, and that the grid communication is re-established at each time step. The single node computational cost of the coupled

code for large *idcfrq* would asymptotically approach about 1.1 times the static grid cases shown in Fig. 5 due to the computation of metrics at each time step. The body dynamics calculation is performed by the manager process, and adds a small computational cost at each step, and requires an additional manager to workers message of 57 floats and 9 integers. Although specified grid motion cases allow this 6-DOF computation to be distributed, the manager was allocated this work for the more general case of aerodynamically determined grid motion.

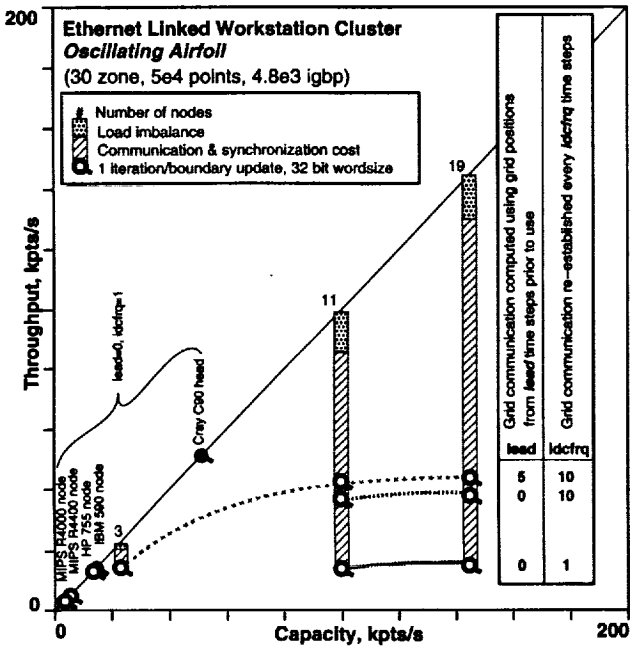


Fig. 12: Oscillating Airfoil case: Scaling study

The cluster capacity was computed from the summation of the single node flow solver capacity, for the nodes executing only the flow solver, with the capacity of the node running both Overflow workers and the DCF process. This total capacity assumes that the ideal overlap of DCF and Overflow is achieved, such that the processors are continuously busy. The losses other than load imbalance are grouped into uncovered communication time and synchronization loss, the latter caused by the Overflow and DCF processes waiting for each other to complete.

Figure 12 shows the performance scaling for three cases, all of which use the manager processor for communication only. The lowest throughput case corresponds to *lead=0, idcfrq=1*, where the grid communication is re-established following integration of the loads computed from the flow solver step. The throughput seen for the 11 node case is about 15% of the load balanced case, and inspection of the runtime CPU during use showed extensive idle time, confirming the expected synchronization bottleneck.

Communication costs are increased for moving body problems, where the boundary interpolants must be passed at intervals of *idcfrq*. The additional message volume required for moving grid capability is approximately 7 integers per IGBP for the donor/receiver indices and pointers, 3 floats per IGBP for the fractional interpolants, and 1 integer per grid point for the iblank array. For this problem, for an effective bandwidth of 6 Mbps and 32 bit integers and floats, this adds an additional 0.5 s of communication overhead at each *idcfrq* steps.

The load balance used for the 11 node *lead=0, idcfrq=1* point was based on the flow solver speeds alone, since no overlap of the flow and grid communication calculation is possible. The breakdown of the times for this point are the sum of 0.5 s for the flow computation, 2.8 s for DCF, and 0.5 s for the IGBP communication, The communication of the boundary Q array is covered by the DCF process. The throughput for the corresponding 19 node case is improved only by the reduction in flow solver compute cost from 0.5 s to 0.34 s, since synchronization losses dominate.

In an attempt to reduce the impact of the grid communication process on the throughput, the use of multiple flow solution steps per IGBP update was investigated. However, for these moving body cases the QBC and the grid metrics were updated at each step. The 11 node *lead=0, idcfrq=10* case shows that a speedup of 2.4 was achieved over the *lead=0, idcfrq=1* case. However, based on the sum of the DCF computation, IGBP communication, flow computation, and QBC communication, speedup of 3.6 should be seen. This discrepancy of 0.26 s per flow step may be caused by decreased effective bandwidth or increased computation by the DCF process due to the larger search distance. For this case the flow processes were again idled during the DCF computation and communication. Again, only a marginal increase in throughput is seen for the corresponding 19 node case.

Reduction in the idle time of the flow processors was achieved by overlapping the grid communication process with the flow processes, shown as the *lead=5, idcfrq=10* cases in Fig. 12. The 0.2 s per step difference between the *lead=0, idcfrq=10* case and the *lead=5, idcfrq=10* can be explained by the partial overlap of the DCF computation and the IGBP communication with the Overflow processes. This overlap was confirmed by inspection of runtime node usage. However, as with the *lead=0, idcfrq=10* case, the throughput seen by the coupled code was 60% of that expected based on the summation of the Overflow time and the QBC communication. Finally, the improvement seen by using 19 nodes is limited due to the loss of full overlap of the DCF and Overflow processes.

Figure 13 compares the lift and drag histories for the overlapping scheme and the reference case. The integrated loads include the single cell overlap, and are presented for a comparison of the lagging proce-

dure with the tightly coupled computation. The differences between the overlapping process, where *lead=5, idcfrq=10*, and the more tightly coupled *lead=0, idcfrq=1* case is less than 1% at peak loading for this short time history.

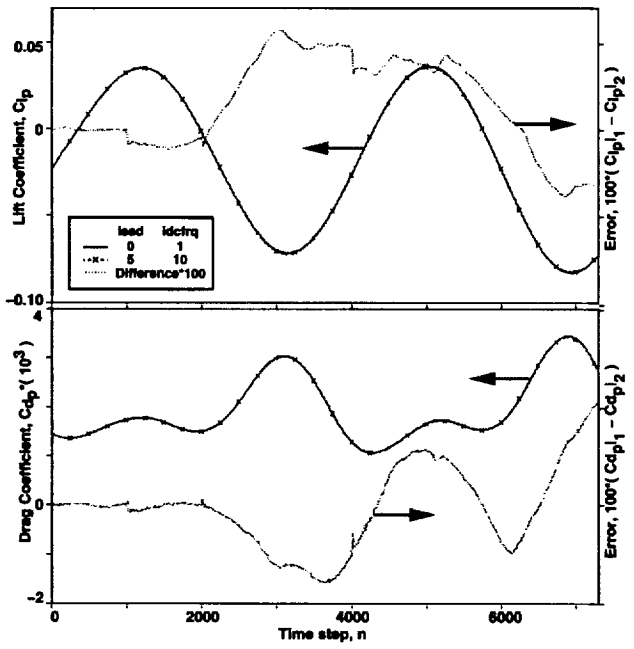


Fig. 13: Oscillating Airfoil case: Lift, C_{l_p}, and drag, C_{d_p}, history

Conclusions

The results of this research would provide the designer with a means of computationally prototyping control systems in problematic nonlinear flight regimes, potentially avoiding costly loss of life and aircraft. However, large simulations of this type will be computationally expensive, providing the impetus for the development of parallel codes.

Currently, workstation clusters using using serialized networks provide capability for solution of moderately-sized steady CFD problems for the low-communication flow solvers of the type investigated here. For the airfoil problem considered here, the highest throughput would be achieved by exchanging boundary information at each iteration, using the manager processor for communication only. The well decomposed 2- and 3-D geometries on up to 19 nodes executed at approximately 50% of cluster capacity. The Ethernet linked cluster of HP (PA-RISC) and SGI (MIPS) nodes reached half of the throughput of a Cray C-90 head. Communication became the dominant source of scaling loss for increasing number of nodes.

Porting the code to the IBM SP2 (PVMe) resulted in performance exceeding that Cray C-90 head for the 2- and 3-D cases. However, the poor effective bandwidth observed with this manager-worker code again resulted in communication dominated losses. The well load balanced cases showed efficiencies of 40% of the corresponding zero-communication case.

The relative grid motion case of the oscillating airfoil showed performance of up to 80% of the Cray benchmark on the Ethernet linked cluster, albeit by lagging of grid communication information. The lagging and infrequent updates of the grid communication caused lift and drag coefficient errors of 1% for this case. Throughput losses over the steady cases included additional message volume due to the changing grid communication, and synchronization of the grid communication and flow solver processes.

Acknowledgements

This investigation was supported by the Computational Aerosciences Branch, the High Performance Computing and Communications Project (HPCCP), the Stratospheric Observatory for Infrared Astronomy (SOFIA) Study Office, and grant NCC 2-799 to Overset Methods under Technical Monitor Dr. Terry L. Holst. Computations were performed on the NAS Cray C90, the HPCCP IBM SP2, the Distributed Cluster Facility, and Ethernet linked Silicon Graphics and Hewlett-Packard workstations at the Computational Aerosciences Branch.

References

[1] Atwood, C.A., "Computation of a Controlled Store Separation from a Cavity," AIAA Paper 94-0031, January 1994. URL http://www.nas.nasa.gov/HPCC/Pubs/atwood/mpegs.html

[2] Allan, B.G., Atwood, C.A., and Packard, A., "Control System Analysis in Nonlinear Flight Regimes," AIAA Paper 94-3544, August 1994.

[3] Steger, J.L., Dougherty, F.C., and Benek, J.A., "A Chimera Grid Scheme," Advances in Grid Generation, K.N. Ghia and U. Ghia, eds., ASME FED-Vol. 5., June 1983.

[4] Buning, P.G. and Chan, W.M., "OVERFLOW/F3D User's Manual, Version 1.5," NASA/ARC, Nov. 1990.

[5] Smith, M.H., and Pallis, J.M., "Medusa - An Overset Grid Flow Solver for Network-Based Parallel Computer Systems," AIAA Paper 93-3312, July 1993.

[6] Atwood, C.A., and Van Dalsem, W.R., "Flowfield Simulation about the Stratospheric Observatory

for Infrared Astronomy," *AIAA J. of Aircraft*, Sep. 1993, pp. 719-727.

[7] Meakin, R.L., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations," AIAA Paper 93-3350, July 1993.

[8] Pulliam, T.H. and Chaussee, D.S., "A Diagonal Form of an Implicit Approximate-Factorization Algorithm," *J. Comp. Phys.*, Vol. 39, Feb. 1981, pp. 347-363.

[9] Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., and Sunderam, V., "PVM 3 User's Guide and Reference Manual," ORNL/TM-12187, May 1993.

[10] McCann, K.M., and Meakin, R.L., "GDCF – An Interactive Approach to Domain Connectivity Among Systems of Overset Grids," AIAA Technical Note, AIAA 11th Computational Fluid Dynamics Conference, Orlando, FL, July 1993.

[11] Fatoohi, R. A. and Weeratunga, S., "Performance Evaluation of Three Distributed Computing Environments for Scientific Applications," Proceedings of Supercomputing '94, Washington DC, Nov. 1994, IEEE Computer Society Press, pp. 400-409.

[12] Meakin, R.L., "On the Spatial and Temporal Accuracy of Overset Grid Methods for Moving Body Problems," AIAA Paper 94-1925, June 1994.

AIAA-95-0224

Nonlinear Fluid Computations in a Distributed Environment

Christopher A. Atwood
Overset Methods
NASA Ames Research Center
Moffett Field, CA 94035-1000

Merritt H. Smith
NASA Ames Research Center
Moffett Field, CA 94035-1000

AIAA 33rd Aerospace Sciences Meeting and Exhibit

January 9-12, 1995 / Reno, NV

Nonlinear Fluid Computations in a Distributed Environment

Christopher A. Atwood*
and
Merritt H. Smith†
NASA Ames Research Center
Moffett Field, California 94035-1000

Abstract

The performance of a loosely and tightly-coupled workstation cluster is compared against a conventional vector supercomputer for the solution the Reynolds-averaged Navier-Stokes equations. The application geometries include a transonic airfoil, a tiltrotor wing/fuselage, and a wing/body/empennage/nacelle transport.

Decomposition is of the manager-worker type, with solution of one grid zone per worker process coupled using the PVM message passing library. Task allocation is determined by grid size and processor speed, subject to available memory penalties. Each fluid zone is computed using an implicit diagonal scheme in an overset mesh framework, while relative body motion is accomplished using an additional worker process to re-establish grid communication.

Introduction

It is anticipated that the high-order simulation of coupled aerodynamics, body dynamics, and controls on shared memory vector architectures will consume considerable resources.[1, 2] One approach which promises to reduce the cost of such computational prototyping is the use of distributed processor machines.

This effort investigates the use of workstation-class nodes linked by a peak 10 Megabit/s (Mbps) or 250 Mbps network. The work shown here uses a manager-worker Chimera[3] RANS solver[4] first implemented on a network of workstations[5] using the Parallel Virtual Machine (PVM) library.[9] The primary issues of concern here relate to the parallel solution of large moving body problems such as would be encountered in a controlled aircraft simulation.

In order to evaluate the performance of manager-worker computation for realistic problems, three configurations are included in the following sections. Timings for a transonic airfoil, a transonic transport,[6] and a tiltrotor[7] fuselage/wing are given. The two-dimensional airfoil case is used to characterize the effects of boundary information lagging upon convergence and throughput. Scaling effects are shown by further decomposition of the grid system while retaining the highly implicit wall-normal direction. Finally, the addition of moving-body capability is examined using an oscillating airfoil case, with differing lag levels of the intergrid boundary points (IGBP).

Approach

Flow Solver

Solutions to the Reynolds-averaged Navier-Stokes (RANS) equations were computed with the implicit diagonal scheme of Pulliam and Chaussee[8] was used, implemented in the Chimera grid framework of Steger, Dougherty, and Benek.[3] Euler implicit time marching and central second-order spatial differencing was used, with viscous wall conditions specified as no-slip, zero normal pressure gradient, and adiabatic. The Baldwin-Lomax algebraic turbulence model is used for all viscous walls. Information transfer across overset mesh boundaries was implemented using trilinear interpolation of the dependent variable vector, $Q = [\rho, \rho u, \rho v, \rho w, e]^T$. The flow solver cost is about $6.5\mu s$/cell/step on a Cray C90 head.

The flow solver Overflow-PVM has been developed from the Overflow overset grid flow solver[4] and the PVM (Parallel Virtual Machine) communication libraries. Parallelism is extracted on a coarse grid-by-grid level, and allows multiple grids to be solved on a single processor. During the run, the manager process primarily transfers flow boundary information (QBC)

*Research Scientist, Overset Methods, Inc., atwood@nas.nasa.gov. Member AIAA.

†Research Scientist, Applied Computational Aerodynamics Branch, mhsmith@nas.nasa.gov. Member AIAA.

amongst the worker nodes.

Relative grid motion is accomplished by the addition of a process to run the grid communication code, the Domain Connectivity Function (DCF).[10] The process is directly linked to the manager and worker nodes, and can be executed at a specified frequency. The grid position and speeds may be lagged a given number of steps to allow for the overlap of the flow solver with the computation of the grid communication.

Grid Decomposition

Scaling studies were performed using zone decomposition of the initial overlapping grid system. Given a user-supplied number of subdomains, the decomposition scheme generates a minimal surface area set of grids and an input file for the DCF grid communication code. A search for a nearby decompositions which result in fewer surface points is also performed. The decomposition is constrained not to divide in the ζ-direction (η in two-dimensions), which is typically the most implicit direction for wall-bounded viscous flows. This constraint also simplifies the search required by the algebraic turbulence model. The resultant set of grids contain one cell overlap with neighboring grids.

Load Balancing

A static load balancing algorithm has been developed to best match grids with processors such that idle time due to load imbalance is minimized.[5] This first implementation of Overflow-PVM initiated solution on all grids simultaneously, including the case of multiple grids on a single node. Two problems with this approach became apparent for the solution of large problems on a limited cluster. Firstly, the context switching of the UNIX operating system caused system time overhead which is wasted on an otherwise idle processor. Secondly, and more importantly, solution of multiple grids in core can consume available RAM, causing swapping of processes to disk. In order to remove or reduce these losses, a modification to the control structure was made such that only a single grid can be active on each processor of the cluster. This drastically reduces context switching between worker processes, and swapping will only take place at the beginning and end of solution on each grid. Finally, the use of multithreaded code compilation and execution on shared-memory multiprocessor, and generally multiuser, machines was implemented.

In addition to the above improvements, a static load balancing method accounting for available RAM was implemented using the integer branch and bound method of the Microsoft Excel Solver. The method allows a cluster user to input available memory on each node and estimated network bandwidth, latency, and contention for each node. The use of virtual memory is implemented as a penalty rather than a constraint. The inclusion of a memory variable in the load balancing scheme is required when cluster nodes of differing clock rates, but similar RAM, are considered. For large problems, not accounting for memory will cause the fastest CPU's to be allocated the most work units, eventually exceeding RAM capacity and inducing slow disk-paging virtual memory use.

The load-balancing algorithm for Overflow-PVM requires only node speeds (grid points/s) and available memory (grid points). For problems where grids are in relative motion, the transfer of boundary interpolant information must be periodically re-computed. If the flow solver is to be run sequentially with the grid communication, the above load balance remains unchanged. However, for cases in which DCF can be overlapped with the flow solution, current algorithm can be used by lowering the speed of the node on which DCF and the grid solvers execute to: DCF node speed=1/ [1/(p/s)+(IGBP/p) / (IGBP/s)] where p=grid points, and IGBP=intergrid boundary points.

Non-Prescribed Relative Grid Motion

Loosely coupled body and fluid dynamics problems require the integrated fluid loads to be used as the forcing function of the body dynamics, and the updated body state to be used for boundary conditions of the fluid solver. Elimination of this serial bottleneck caused by the coupling will offer increased throughput, albeit at the sacrifice of some level of accuracy. However, a large class of problems exists for which the low ratio of body natural or forced frequency over fluid motion frequency scales indicate that lagging of boundary interpolants may not cause significant reductions in accuracy.

Figure 1 depicts a comparison of the the sequential and parallel methods. The sequential technique is shown for the case of lockstep update of the IGBP after each flow solver time step and no lead time for DCF. The parallel method overlaps DCF and Overflow by starting the DCF process *lead* steps before the updated grid communication solution will be required by Overflow. The frequency at which grid communication is re-established is given by $idcfrq$. Note that the metrics are updated every step, n, based on the load integration from the previous step.

The lagging example shown in Fig. 1a computes the integrated loads acting on the body in Overflow at step $n-1$, uses these loads to compute the new body state, sends this state to the grid communication process DCF, and also uses the new body state for the flow solver step n. At time $n+1$ the Overflow manager process waits for updated grid communication to be sent to the worker processes before signalling the start of the solution.

It is anticipated that this lagging would adversely effect accuracy when the ratio of applied+aero loads

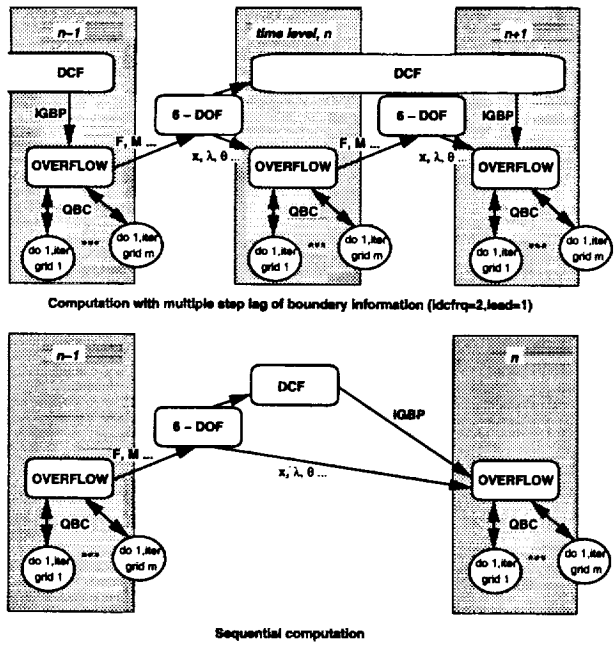


Fig. 1: Flow solver and grid communication coupling methods

to mass properties is large, i.e., accelerations are large or when fast relative grid motion exists. One disadvantage of the approach is that more total run-time nodes will be required, most dedicated to the flow solver and a few for the grid communication. However, this separation of flow solver and grid communication codes may make the coupled code more portable and maintainable because each code may be distributed the manner most appropriate.

The scheme implemented here also duplicated grid information for the DCF process, storing temporary grid and search map files in /tmp. This is done due to the requirement that the grid communication scheme generally will access all of the grids during the search for donor/receiver pairs. Accessing grid data from several nodes will incur heavy network load, competing with the flow solver nodes for the shared resource.

Results

In an effort to determine the highest throughput solution process, the Ethernet linked workstation cluster was used to evaluate the impact of lagged boundary updates and computation using 32 versus 64 bit word-sizes.

The cases were executed during low node and network use periods in single cluster-user mode on a set of cross-mounted workstations using PVM 3.2.3. No modification of the network topology was attempted, and the all of the nodes were on either of two subnets. Three gateways separated two of the pool of nodes from the master (measured using *traceroute*), with a point-to-point transmission time, shown by *ping -f* for an 8 KB datagram, of 300 ms on an otherwise idle network. This gives an effective bandwidth of 0.21 Mbps to these two distant nodes versus 0.35 Mbps measured between nodes with no intermediary gateways. For smaller datagrams, the lower collision rate increases the effective bandwidth to 8 and 2 Mbps for the near and distant nodes respectively

The peak transfer rate of Ethernet is 10 million bits per second (Mbps), although during low-use periods the observed bandwidth of the cluster is about 2-9 Mbps, which can drop to 0.2 Mbps during normal use periods. For the large packets typically used in these computations, the discrepancy between observed and peak data transfer rates can be primarily attributed to contention, PVM-required memory to buffer copies, and latency.[11]

For a typical Chimera application, approximately 10% of the total grid points are IGBP's, each of which require the dependent variable vector, Q, from another zone. In a distributed application, generally this information must be passed to another node. Overflow-PVM is implemented in a manager-worker arrangement, with the manager first parsing the grids to the worker processes, then receiving, updating, and resending large packets of boundary information at a user-specified rate. The solution from the run is collected following the final flow iteration.

Throughput timings for all of the cases were obtained either by monitoring CPU usage or by runs of 5-10 iterations, from which the zero iteration runtime was subtracted. Subtraction of start-up and wind-up costs also allowed good estimates for long-duration runs on an otherwise idle network. Data conversion (XDR) was not required for this set of nodes, and disk swapping not observed for any of the cases.

The static grid version of the code was also run on the High Performance Computing and Communication Program (HPCCP) IBM SP2 located at the Numerical Aerodynamic Simulator (NAS), using the PVMe message passing library. These cases were run in single grid per node mode, with timing numbers obtained in the same manner as on the loosely coupled cluster. Switching of interruptions did not affect timings appreciably for any of the configurations.

Steady Airfoil Case

The solution of the steady transonic flow about a 64a010 airfoil, depicted in Fig. 2, was computed to 1) quantify the effect boundary information lagging upon convergence and 2) to measure the scalability and accuracy of this domain decomposition method.

Figure 3 shows the effect on convergence of decomposing the grid from 3 zones into 30, with single cell overlap. Local time step scaling based on geomet-

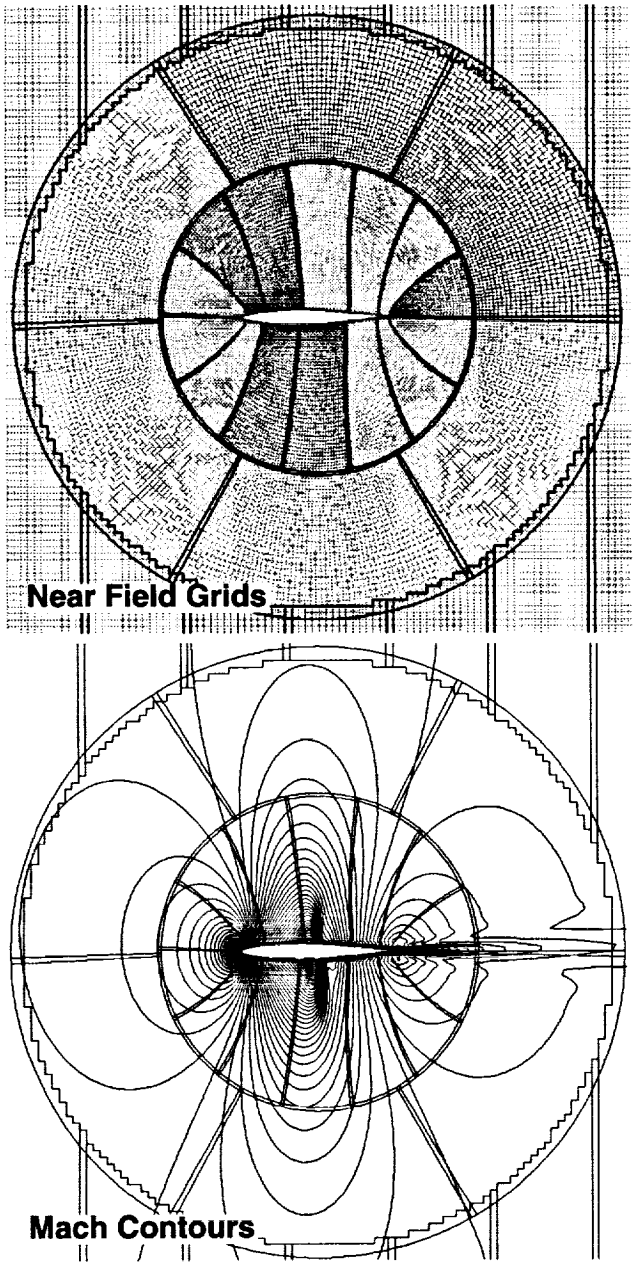


Fig. 2: 64a010 Airfoil case: grid system and Mach contours

ric considerations only was used for all of the cases: $\Delta\tau_{(i,j,k)} = \Delta\tau[1 + 0.005(1/\sqrt{V})]/[1 + 1/\sqrt{V}]$. The histories of the L2 norm for both the Ethernet linked workstation (32 bit words) and the IBM SP2 (64 bit words) are of similar trends. The differences in the high-frequency components of the history may be attributable to the elimination of fourth-order dissipation at the zone boundaries and to the inclusion of the overlapping boundary points in the computation of the residuals. The residual on the on the IBM SP2 converges an additional order of magnitude over the 32-bit wordsize cases.

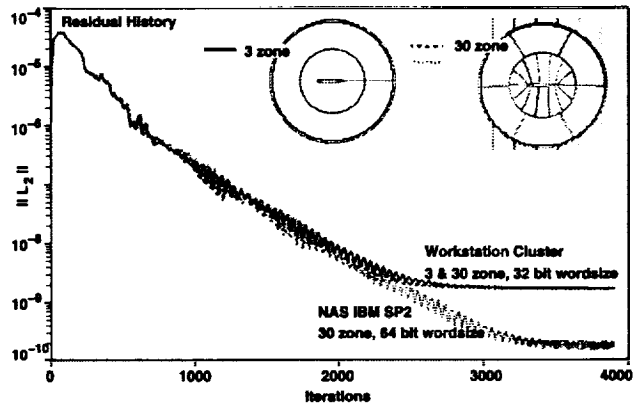


Fig. 3: 64a010 Airfoil case: Residual histories

Figure 4 compares C_p plots for the present cases compared to experiment and a high resolution single zone computation computed on a Cray C-90 (64 bit wordsize).[12] The solutions are generally in agreement, although differences exist, and are most significant near the shock.

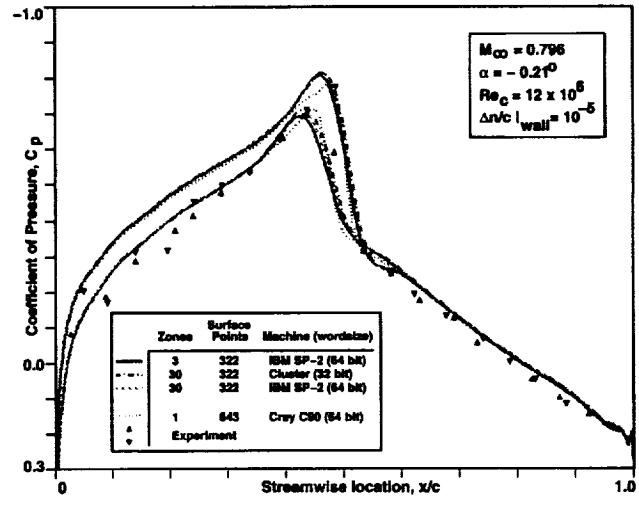


Fig. 4: 64a010 Airfoil case: Comparison of C_p vs. x/c

Figure 5 shows the speedup achieved for the foil problem over a differing number of nodes. The identity line represents ideal speedup, with speeds for single

node flow solver runs included for reference. The difference between ideal and actual speed is broken into two components. The first component contributing to lost capacity is that due to load imbalance, and is the speed the code would run given a perfect network, i.e. infinite bandwidth and zero latency. The second component contributing to the lost computational capacity is that attributable to message passing from one node to another, including time to pack (buffer copy), transmit (network latency and bandwidth), and unpack (buffer copy). The magnitude of this cost will depend both upon the number of IGBP and the ratio of surface area to volume, or IGBP/points. For this case the number of IGBP is 4827, and the total number of points is 49927. In general, the ratio of IGBP/points ranges from about 10 to 15%.

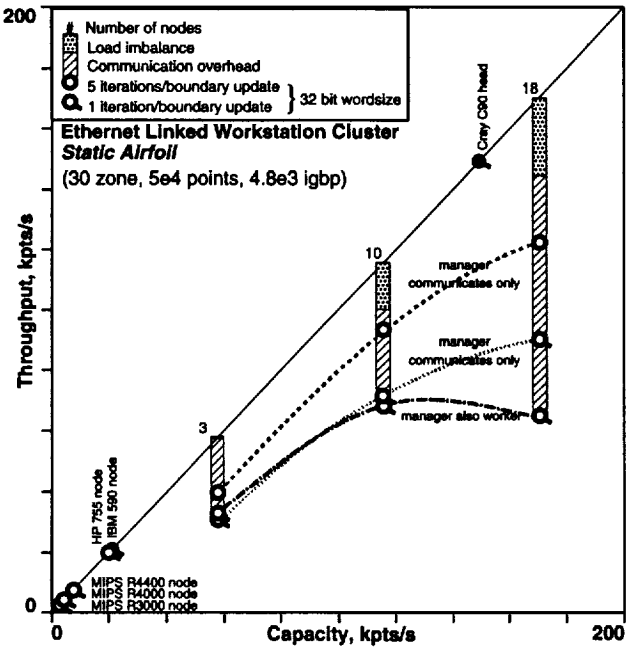


Fig. 5: 64a010 Airfoil case: Scaling study

Figure 5 also shows that it is advantageous to reserve the manager processor for communication alone for jobs where network contention is high. When the processor which contains the manager also has worker processes, the CPU is busy with the worker process when messages from completing remote processes return. The interrupted CPU then processes the incoming message before returning control to the worker process, with the concomitant adverse effect on throughput.

The difference in throughput seen between the 1 and 5 iterations per boundary Q update cases shown in Fig. 5 is indicative of the contention that is occurring for the network. Since exchange of boundary information comprises the bulk of the message traffic, decreasing this communication / computational time ratio decreases time to solution even though the number of CPU cycles is the same in either case.

An estimate of the loss in throughput due to the network can be computed from the time required to serially communicate the boundary Q information. The communication time for this geometry and precision is about 0.25 seconds: 5 words/point $\times$ 32 bits/word $\times$ 4.8×10^3 boundary points $\times$ 2 transmissions (one receive + one send) $\div$ 6×10^6 bits/second. This communication cost causes a loss in throughput of about 30 kpts/s for the 10 node case. For the 5 iteration per boundary update cases, some of this loss is covered by computation as the different processes finish with larger intervening times.

By comparison of the 5 and 1 iteration per boundary update cases, it can be seen that the cost of updating the boundary information, Q, at each step is high. For the 10 node case, the throughput would be 35% higher, provided that convergence is unaffected.

However, inspection of Fig. 6 shows the marked effect lagging of boundaries has upon convergence for the 30 zone problem. When the boundary update is lagged for a single step (iter=2) the convergence rate drops dramatically, and is monotonically worse for increased lagging. The integrated C_{l_p} includes the single cell overlap, and are presented for a comparison of the lagging levels only. Upon convergence the maximum Courant number was approximately 800 and the Courant number across the intergrid boundaries near the foil was about 10.

Figure: 7 shows the computation of the airfoil case on the IBM SP2, a tightly coupled cluster of workstations connected by a 240 Mbps peak network. The native implementation, PVMe (similar to PVM 3.2.5), of the message passing library permitted runs with only a single process per node. Although throughput for the 30 zone case exceeds that of a Cray C-90 head, the code remains communication bound, as can be seen from inspection of the 5 and 1 iteration per boundary update cases. The resultant throughput for the single update per iteration case is 40% of that achievable on an ideal network. By taking the difference in times between the perfectly load balanced case and the single iteration per boundary update, the effective bandwidth for this problem was estimated at 35 Mbps. The peak network performance for PVMe on the machine is approximately 240 Mbps for large message sizes (100 KB), dropping off to about 120 Mbps for the estimated primary message size of 8 KB for the airfoil problem.

Three-Dimensional Static Geometries

Two geometries were used to evaluate the performance of Overflow-PVM for viscous problems in three-dimensions: a tiltrotor fuselage-wing,[7] and a transonic transport.[6] Scaling of the tiltrotor geometry, shown in Figs. 8 and 9, was computed by subdivision of the 9

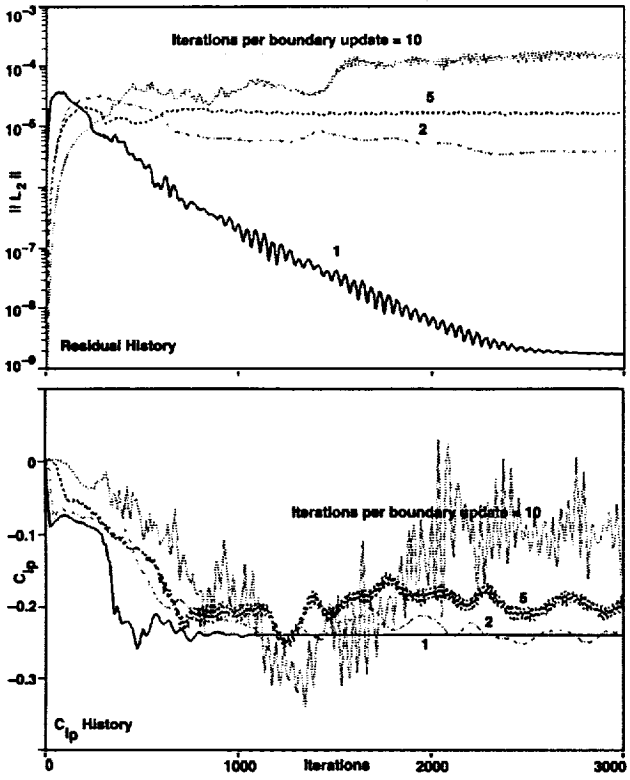


Fig. 6: 64a010 Airfoil case (30 zone): Effect of boundary lagging

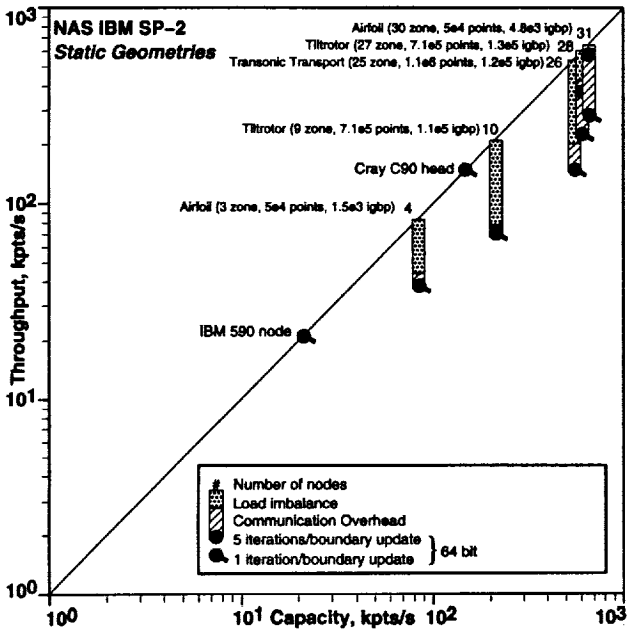


Fig. 7: NAS IBM SP2 scaling study

zone grid system into 27 zones using the same preprocessing technique as used with the airfoil. The transport case, shown in Fig. 10, did not use additional subdivision for this study, resulting in significant load imbalance for large node number cases.

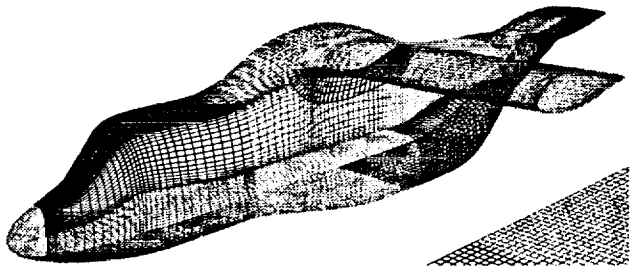


Fig. 8: Tiltrotor fuselage/wing (27 zone): decomposition of near field grids

Fig. 9: Tiltrotor fuselage/wing: developing solution

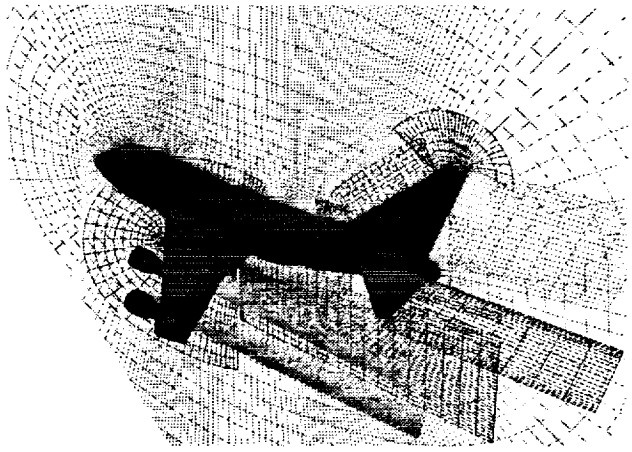


Fig. 10: Transonic transport: domain decomposition

Figure 7 shows the throughput seen on the tightly coupled cluster for these cases. Comparison of the 9 and 27 zone tiltrotor cases shows that the throughput as a fraction of capacity increases, and is higher than that of a C-90 head for the more distributed case. However, for the 9 zone configuration the loss is primarily due to load imbalance, while network associated losses dominate for the 27 zone case. Again, the 5 iteration per boundary update case is shown to exemplify the communication losses, which consume roughly half of the total time per step. The 26 node

transport case shows much of the reduction from capacity is due to load imbalance. Comparison of the 26 node transport case and the 28 node tiltrotor case shows that the load imbalance for the former partially masks the communication with computation. The 28 node tiltrotor case, with a single step per boundary Q update (QBC), shows a throughput of 51% of the load balanced capacity, while the transport case shows 80% of the throughput that would be seen on an ideal net.

A summary of the results obtained on the Ethernet linked cluster is shown in Fig. 11 for both 32 and 64 bit words. The performance of the code on single nodes is shown for 64 bit floats, with the 32 bit values from Fig. 5 applicable for these cases. The 64 bit performance of the PA-RISC and MIPS nodes used for these cases was approximately half that seen with 32 bit words. The IBM RS6000 node, which performs similarly using single or double precision, is shown for reference only.

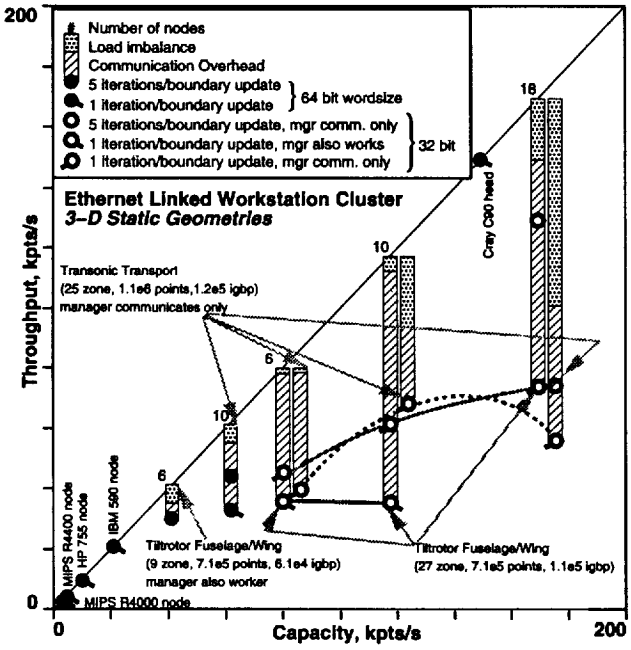


Fig. 11: Three-dimensional cases

The 64 bit transport case, which load balances to 90% efficiency, loses about 14 s per iteration to communication, implying an effective bandwidth of 5.5 Mbps. Using 5 iterations per boundary update decreases the communication overhead to 6 s. The nine zone tiltrotor case executed at 70% capacity using 5 iterations per update.

The 32 bit float tiltrotor cases given in Fig. 11 for 6 and 10 node machines again show the advantageous effect of using the manager node for communication alone. While the efficiency drops when the manager processor is allocated work according to processor speed, a positive trend is evident when the processor is allocated no grids. However, although throughput increases with increasing processors, the communication cost remains constant, with partial overlap of communication and computation due to higher load imbalance for the large machine cases. The communication cost is approximately 6 s for these tiltrotor cases, implying an effective bandwidth of 6 Mbps.

The 32 bit transport cases shown in Fig. 11 show increasing throughput from the 6 node to the 10 node case, but a negative trend from 10 to 18 nodes, despite the potentially communication covering high load imbalance. Using a 6 Mbps bandwidth, the communication cost for the 6, 10, and 18 node cases can be estimated at 6.4 s. Inspection of Fig. 11 shows that the difference between the ideal network cases and the 1 step per QBC update could be explained by 15 s /step communication cost. However, since the communication pattern is similar between the airfoil and tiltrotor cases, that is a large messages, a loss in effective bandwidth is not expected. The poor performance of the transport cases could be caused by cache misses owing to the large grids of this case, effectively lowering the load balanced throughput. The largest grids for the transport case had 1.2×10^5 points, as opposed to the 3×10^4 grids used in the tiltrotor case. However, the load balance plotted for the Fig. 11 cases uses the higher single node throughput values to uniformly represent the geometries.

Relative Grid Motion Cases

An oscillating 64a010 foil was used to measure the performance of the loosely coupled cluster for the coupled fluid/grid communication code. The grid system shown in Fig. 2 was used for this case, with the 14 foil-attached zones oscillating about the mid-chord with the remaining 16 grids fixed in inertial space. The interface between the moving and static grids is circular with uniform circumferential spacing. This topology removes effect of differing grid resolution between the zones, and provides a single relative grid speed to be evaluated for the boundary lagging cases.

The mean angle of attack was $-0.21°$ with an oscillation magnitude of $1.01°$ at a frequency of 25.6 Hz. The Reynolds number was 12×10^6 at a freestream Mach of 0.796, with a wall normal spacing of 10^{-5} of chord. A time step of $\Delta t = 10.2\mu s$ was chosen to maintain a Courant number near 0.7 across the zonal boundaries, although at the viscous wall the Courant number is approximately 700. Using this time step, the number of steps required per oscillation was approximately 3830.

The single node throughput speeds along the identity line in Fig. 12 were computed using the estimate of 10% of the grid points being IGBP, and that the grid communication is re-established at each time step. The single node computational cost of the coupled

code for large *idcfrq* would asymptotically approach about 1.1 times the static grid cases shown in Fig. 5 due to the computation of metrics at each time step. The body dynamics calculation is performed by the manager process, and adds a small computational cost at each step, and requires an additional manager to workers message of 57 floats and 9 integers. Although specified grid motion cases allow this 6-DOF computation to be distributed, the manager was allocated this work for the more general case of aerodynamically determined grid motion.

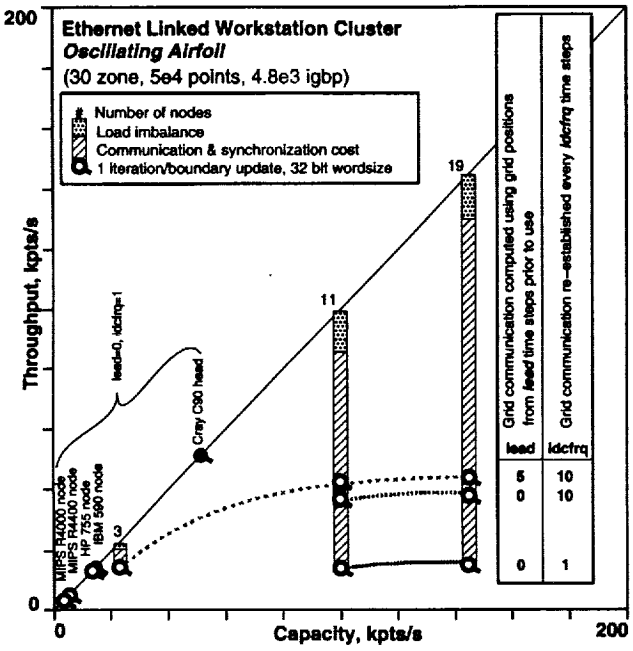


Fig. 12: Oscillating Airfoil case: Scaling study

The cluster capacity was computed from the summation of the single node flow solver capacity, for the nodes executing only the flow solver, with the capacity of the node running both Overflow workers and the DCF process. This total capacity assumes that the ideal overlap of DCF and Overflow is achieved, such that the processors are continuously busy. The losses other than load imbalance are grouped into uncovered communication time and synchronization loss, the latter caused by the Overflow and DCF processes waiting for each other to complete.

Figure 12 shows the performance scaling for three cases, all of which use the manager processor for communication only. The lowest throughput case corresponds to *lead=0, idcfrq=1*, where the grid communication is re-established following integration of the loads computed from the flow solver step. The throughput seen for the 11 node case is about 15% of the load balanced case, and inspection of the runtime CPU during use showed extensive idle time, confirming the expected synchronization bottleneck.

Communication costs are increased for moving body problems, where the boundary interpolants must be passed at intervals of *idcfrq*. The additional message volume required for moving grid capability is approximately 7 integers per IGBP for the donor/receiver indices and pointers, 3 floats per IGBP for the fractional interpolants, and 1 integer per grid point for the iblank array. For this problem, for an effective bandwidth of 6 Mbps and 32 bit integers and floats, this adds an additional 0.5 s of communication overhead at each *idcfrq* steps.

The load balance used for the 11 node *lead=0, idcfrq=1* point was based on the flow solver speeds alone, since no overlap of the flow and grid communication calculation is possible. The breakdown of the times for this point are the sum of 0.5 s for the flow computation, 2.8 s for DCF, and 0.5 s for the IGBP communication, The communication of the boundary Q array is covered by the DCF process. The throughput for the corresponding 19 node case is improved only by the reduction in flow solver compute cost from 0.5 s to 0.34 s, since synchronization losses dominate.

In an attempt to reduce the impact of the grid communication process on the throughput, the use of multiple flow solution steps per IGBP update was investigated. However, for these moving body cases the QBC and the grid metrics were updated at each step. The 11 node *lead=0, idcfrq=10* case shows that a speedup of 2.4 was achieved over the *lead=0, idcfrq=1* case. However, based on the sum of the DCF computation, IGBP communication, flow computation, and QBC communication, speedup of 3.6 should be seen. This discrepancy of 0.26 s per flow step may be caused by decreased effective bandwidth or increased computation by the DCF process due to the larger search distance. For this case the flow processes were again idled during the DCF computation and communication. Again, only a marginal increase in throughput is seen for the corresponding 19 node case.

Reduction in the idle time of the flow processors was achieved by overlapping the grid communication process with the flow processes, shown as the *lead=5, idcfrq=10* cases in Fig. 12. The 0.2 s per step difference between the *lead=0, idcfrq=10* case and the *lead=5, idcfrq=10* can be explained by the partial overlap of the DCF computation and the IGBP communication with the Overflow processes. This overlap was confirmed by inspection of runtime node usage. However, as with the *lead=0, idcfrq=10* case, the throughput seen by the coupled code was 60% of that expected based on the summation of the Overflow time and the QBC communication. Finally, the improvement seen by using 19 nodes is limited due to the loss of full overlap of the DCF and Overflow processes.

Figure 13 compares the lift and drag histories for the overlapping scheme and the reference case. The integrated loads include the single cell overlap, and are presented for a comparison of the lagging proce-

dure with the tightly coupled computation. The differences between the overlapping process, where *lead=5, idcfrq=10*, and the more tightly coupled *lead=0, idcfrq=1* case is less than 1% at peak loading for this short time history.

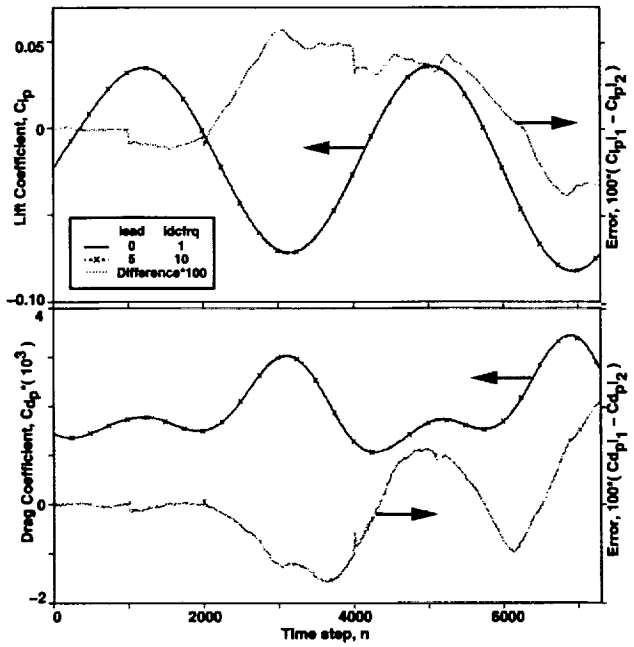


Fig. 13: Oscillating Airfoil case: Lift, C_{l_p}, and drag, C_{d_p}, history

Conclusions

The results of this research would provide the designer with a means of computationally prototyping control systems in problematic nonlinear flight regimes, potentially avoiding costly loss of life and aircraft. However, large simulations of this type will be computationally expensive, providing the impetus for the development of parallel codes.

Currently, workstation clusters using using serialized networks provide capability for solution of moderately-sized steady CFD problems for the low-communication flow solvers of the type investigated here. For the airfoil problem considered here, the highest throughput would be achieved by exchanging boundary information at each iteration, using the manager processor for communication only. The well decomposed 2- and 3-D geometries on up to 19 nodes executed at approximately 50% of cluster capacity. The Ethernet linked cluster of HP (PA-RISC) and SGI (MIPS) nodes reached half of the throughput of a Cray C-90 head. Communication became the dominant source of scaling loss for increasing number of nodes.

Porting the code to the IBM SP2 (PVMe) resulted in performance exceeding that Cray C-90 head for the 2- and 3-D cases. However, the poor effective bandwidth observed with this manager-worker code again resulted in communication dominated losses. The well load balanced cases showed efficiencies of 40% of the corresponding zero-communication case.

The relative grid motion case of the oscillating airfoil showed performance of up to 80% of the Cray benchmark on the Ethernet linked cluster, albeit by lagging of grid communication information. The lagging and infrequent updates of the grid communication caused lift and drag coefficient errors of 1% for this case. Throughput losses over the steady cases included additional message volume due to the changing grid communication, and synchronization of the grid communication and flow solver processes.

Acknowledgements

This investigation was supported by the Computational Aerosciences Branch, the High Performance Computing and Communications Project (HPCCP), the Stratospheric Observatory for Infrared Astronomy (SOFIA) Study Office, and grant NCC 2-799 to Overset Methods under Technical Monitor Dr. Terry L. Holst. Computations were performed on the NAS Cray C90, the HPCCP IBM SP2, the Distributed Cluster Facility, and Ethernet linked Silicon Graphics and Hewlett-Packard workstations at the Computational Aerosciences Branch.

References

[1] Atwood, C.A., "Computation of a Controlled Store Separation from a Cavity," AIAA Paper 94-0031, January 1994. URL http://www.nas.nasa.gov/HPCC/Pubs/atwood/mpegs.html

[2] Allan, B.G., Atwood, C.A., and Packard, A., "Control System Analysis in Nonlinear Flight Regimes," AIAA Paper 94-3544, August 1994.

[3] Steger, J.L., Dougherty, F.C., and Benek, J.A., "A Chimera Grid Scheme," Advances in Grid Generation, K.N. Ghia and U. Ghia, eds., ASME FED-Vol. 5., June 1983.

[4] Buning, P.G. and Chan, W.M., "OVERFLOW/F3D User's Manual, Version 1.5," NASA/ARC, Nov. 1990.

[5] Smith, M.H., and Pallis, J.M., "Medusa - An Overset Grid Flow Solver for Network-Based Parallel Computer Systems," AIAA Paper 93-3312, July 1993.

[6] Atwood, C.A., and Van Dalsem, W.R., "Flowfield Simulation about the Stratospheric Observatory

for Infrared Astronomy," *AIAA J. of Aircraft*, Sep. 1993, pp. 719-727.

[7] Meakin, R.L., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations," AIAA Paper 93-3350, July 1993.

[8] Pulliam, T.H. and Chaussee, D.S., "A Diagonal Form of an Implicit Approximate-Factorization Algorithm," *J. Comp. Phys.*, Vol. 39, Feb. 1981, pp. 347-363.

[9] Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., and Sunderam, V., "PVM 3 User's Guide and Reference Manual," ORNL/TM-12187, May 1993.

[10] McCann, K.M., and Meakin, R.L., "GDCF – An Interactive Approach to Domain Connectivity Among Systems of Overset Grids," AIAA Technical Note, AIAA 11th Computational Fluid Dynamics Conference, Orlando, FL, July 1993.

[11] Fatoohi, R. A. and Weeratunga, S., "Performance Evaluation of Three Distributed Computing Environments for Scientific Applications," Proceedings of Supercomputing '94, Washington DC, Nov. 1994, IEEE Computer Society Press, pp. 400-409.

[12] Meakin, R.L., "On the Spatial and Temporal Accuracy of Overset Grid Methods for Moving Body Problems," AIAA Paper 94-1925, June 1994.