NASA Technical Memorandum 107489
AIAA–97–3295

Army Research Laboratory
Memorandum Report ARL–MR–369

Parallel ALLSPD–3D: Speeding Up Combustor Analysis Via Parallel Processing

David M. Fricker
U.S. Army Research Laboratory
Lewis Research Center
Cleveland, Ohio

Prepared for the
33rd Joint Propulsion Conference and Exhibit
cosponsored by AIAA, ASME, SAE, and ASEE
Seattle, Washington, July 6–9, 1997

National Aeronautics and
Space Administration

U.S. ARMY
RESEARCH LABORATORY

PARALLEL ALLSPD-3D: SPEEDING UP COMBUSTOR ANALYSIS VIA PARALLEL PROCESSING

David M. Fricker*
U.S. Army Research Laboratory - Vehicle Technology Center, Lewis Site
Cleveland, OH

Abstract

The ALLSPD-3D Computational Fluid Dynamics code for reacting flow simulation was run on a set of benchmark test cases to determine its parallel efficiency. These test cases included non-reacting and reacting flow simulations with varying numbers of processors. Also, the tests explored the effects of scaling the simulation with the number of processors in addition to distributing a constant size problem over an increasing number of processors. The test cases were run on a cluster of IBM RS/6000 Model 590 workstations with ethernet and ATM networking plus a shared memory SGI Power Challenge L workstation. The results indicate that the network capabilities significantly influence the parallel efficiency, i.e., a shared memory machine is fastest and ATM networking provides acceptable performance. The limitations of ethernet greatly hamper the rapid calculation of flows using ALLSPD-3D.

Nomenclature

S = Speedup
E = Efficiency
N = Number of processors
T = Time
T_{wall} = wall clock or elapsed time
T_{cpu} = CPU time used by process
$_{serial}$ = serial processing with a single processor
$_{parallel}$ = parallel processing with multiple processors
ATM = Asynchronous Transfer Mode network
ethernet = Ethernet network
Re_{dia} = Reynolds Number based on diameter
T_{ref} = Reference Temperature
U_{ref} = Reference Velocity
K = Kelvin
m/s = meters/second

Introduction

ALLSPD-3D Capabilities

The ALLSPD-3D combustion code is a numerical tool developed by the Internal Fluid Mechanics Division (which is now the Turbomachinery and Propulsion Systems Division) at the NASA Lewis Research Center for simulating chemically reacting flows in aerospace propulsion systems.[1] It provides the designer of advanced engines an analysis tool that employs state-of-the-art computational technology. The code can simulate multi-phase, swirling flows over a wide Mach-number range in combustors of complex geometry. Three-dimensional, curvilinear, structured grids with multiple zones and internal obstacles give great flexibility in fitting the grid to solid bodies in the flow simulation. Various boundary conditions (multiple inlets/outlets, dilution holes, transpiration holes, periodic, symmetry, far-field, adiabatic or isothermal walls, centerline singularity) also increase the utility of ALLSPD-3D in solving complex flow simulations.

The ALLSPD-3D Computational Fluid Dynamics (CFD) code which was released in November, 1995, evolved from the two-dimensional code ALLSPD-2D (released in June, 1993). Besides extension to three dimensions, the newer code featured several improvements and enhancements, including a user-friendly Graphical User Interface (GUI), multi-platform capability (supercomputers, workstations, and parallel processors), improved turbulence and spray models, and more generalized property and chemical reactions databases. Also, eddy breakup models for turbulence-chemistry interactions were introduced. A very warmly received feature of the ALLSPD-3D version 1.0 code was the GUI for easier problem setup and post-processing.

*Engine Components Division.
Senior Member AIAA.

The ALLSPD combustion codes utilize a finite-difference, compressible flow formulation with low Mach number preconditioning of the Navier-Stokes equations. (The ALLSPD-3D code is intended only for subsonic flow simulations since it uses central-differencing for convective and viscous terms on right and left-hand sides.) Laminar or turbulent flow capability also exists, and the turbulent flows are solved using a low-Reynolds number k-ϵ turbulence model. The chemistry model can handle frozen or finite rate chemistry flows. Spray combustion is supported by a stochastic, separated flow spray model.

Need for parallelization

ALLSPD was parallelized in response to the changing computational capabilities of the major engine companies, specifically, the move from large supercomputers to small workstations. ALLSPD-3D is memory and CPU intensive for practical engineering problems. This led to the need for parallel processing on UNIX workstations such as those from HP, IBM, SGI, & Sun. However, the serial code was not to be abandoned, nor was the parallel version to be wildly divergent from the serial code. Also, the parallel code needed to be developed using parallel processing techniques readily available to the average user. Therefore, ALLSPD-3D was parallelized using the de-facto standard PVM (Parallel Virtual Machine) message passing library and with minimal modifications to the serial code.

Transferring data by message passing supplies exactly the information a process needs from its neighboring zones without requiring memory space for all of the data in all of the other zones. Because each process needs data for only its own grid zone (including those ghost cells which actually belong to neighboring zones), each process only needs enough memory for the largest zone. This reduced memory feature of parallel processing can be very beneficial with large problem sizes. Also, since each process only calculates data on its zone, the time needed to calculate a single iteration is reduced to approximately the time needed for the most numerically intensive zone. The only cost for these great benefits of parallel processing is the time it takes to transfer data between neighbors.

ALLSPD-3D Parallelization

Domain decomposition

The parallel processing in ALLSPD-3D is quite simple: the code is inherently divided in the data domain, therefore domain decomposition is used. The multiple grid zone feature provides natural dividing lines in the data for decomposing the problem onto multiple processors, i.e., each grid zone is a natural candidate for parallel processing. This also minimizes the changes to the serial code. Boundary data is exchanged between processors using the PVM message-passing library, and each processor only needs as much memory as demanded by the largest grid zone. This memory limitation is due to the lack of dynamic memory allocation in ALLSPD-3D; all array sizes are set at compile time based upon the largest grid zone since it falls within the Single Program, Multiple Data (SPMD) paradigm. SPMD can be translated as each processor running the same program as all of the other processors but with differing data.

Unfortunately, this limitation extends to the amount of data transferred between processors at the end of each iteration. The first release of ALLSPD-3D contains a design flaw which sets the amount of data to transfer using the maximum possible size of a grid zone's face regardless of how much smaller the grid face being transferred is. The maximum face size is determined at compile time, and this sets the amount of data transferred for all processors. If the size of a particular grid face to be passed to a neighboring grid zone is much smaller than the maximum possible, then a substantial penalty in communication time is taken by the transfer of unneeded information. Reducing this penalty requires code modifications to properly size the amount of data to transfer.

Message passing and PVM

The PVM (Parallel Virtual Machine) message-passing library was developed at Oak Ridge National Laboratory in Oak Ridge, Tennessee.[2] PVM was chosen because of its wide acceptance, installed user base, and portability. PVM is used in a wide variety of applications on numerous architectures and has become a de-facto standard for message-passing libraries.

The PVM library has many features including spawning of processes on a virtual machine and the communication of various message types between architectures which may have inherently different data structures. These features are used in the parallel version of ALLSPD-3D.

ALLSPD-3D version 1.0b with a minor modification was used for this study of parallel efficiency. The modification involves changing the method used to transfer data between processors. Version 1.0b (and all preceding versions) used the PVM library calls *pvmfpsend()* and *pvmfprecv()* for each flow variable to be transferred. The special version of ALLSPD-3D used for this study replaced these calls with a block of *pvmfpack()* and *pvmfunpack()* calls in conjunction with *pvmfsend()* or *pvmfrecv()* as appropriate. Note the difference of *psend()* vs. *send()* in the subroutine names.

The *pvmfpsend()* and *pvmfprecv()* calls are normally faster modes of passing messages, and the PVM documentation indicates that data sent and received will be automatically translated to native formats. The changes were made when it was discovered that the *pvmfpsend()* and *pvmfprecv()* calls did not perform automatic data type conversion between machines with different data representation formats such as Cray and SGI. Since the manuals made no mention of this fact, *pvmfpsend()* and *pvmfprecv()* were used in the original coding. However, to preserve the heterogeneous capability of ALLSPD-3D, the code changes were made. Subsequent testing revealed no degradation in parallel performance was caused by changing the method used to transfer data between processors. Thus, the use of a homogeneous workstation cluster was not affected by the modification.

Test Cases

Non-reacting transition duct

The first test case used for evaluating the parallel efficiency of ALLSPD-3D is a three-dimensional circular to rectangular transition duct with a fully turbulent, non-reacting gas mixture (air) flowing through it. This test case is one of the samples included in the ALLSPD-3D distribution and is detailed in the ALLSPD-3D user manual.[1] The fluid dynamics details are in Table 1. The single zone grid used in the baseline test case is shown in Figure 1.

To study the effect of increasing the number of processors on parallel efficiency, the baseline grid was modified for each variation. For simple speedup testing, the baseline grid was split into multiple zones of equal size with one zone per processor. To test the effects of scaling the problem with the number of processors, the baseline grid was mirrored across symmetry planes for the two and four processor cases. Then the four processor grid was refined and divided to create the eight and sixteen processor test cases. Each manipulation of the grid maintained roughly the same number of points per zone (and per processor) as the baseline test case. Thus, the two processor grid had twice as many points as the baseline while the sixteen processor grid had sixteen times as many points as the baseline. Tables 2 and 3 detail the grids used in each transition duct test case.

Re_{dia}	*195,000*
T_{ref}	*298 K*
U_{ref}	*29 m/s*

Table 1 - Transition duct flow characteristics

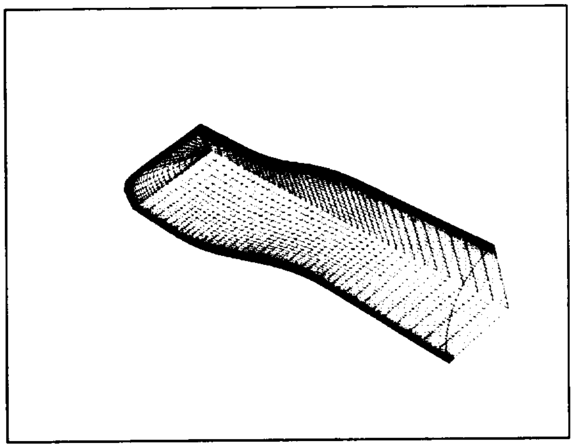

Figure 1 -Single zone grid (41x21x61=52521 points) for baseline transition duct

NUMBER OF ZONES	ZONE DIMENSIONS	POINTS PER ZONE	TOTAL NUMBER OF POINTS
1	41 x 21 x 61	52521	52521
2	41 x 21 x 31	26691	53382
4	41 x 21 x 16	13776	55104
8	21 x 21 x 16	7056	56448
16	21 x 11 x 16	3696	59136

Table 2 - Transition duct grids for simple speedup tests

NUMBER OF ZONES	ZONE DIMENSIONS	POINTS PER ZONE	TOTAL NUMBER OF POINTS
1	41 x 21 x 61	52521	52521
2	41 x 21 x 61	52521	105042
4	41 x 21 x 61	52521	210084
8	41 x 21 x 61	52521	420168
16	41 x 21 x 61	52521	840336

Table 3 - Transition duct grids for scaled speedup tests

Each test case was run with the serial and parallel versions of the code for direct comparison of the run times since the multiple zones of the grids introduce extra points for overlapping cells. These extra points preclude an accurate comparison between the run times of a single zone grid and that of a multiple zone grid. The simple tests and the scaled tests were run on the cluster of IBM RS/6000 Model 590 workstations using ethernet and ATM networking.

Reacting swirl can

The second test case used for evaluating the parallel efficiency of ALLSPD-3D is an axisymmetric swirl can combustor with a fully turbulent gas mixture (air) reacting with a methanol spray. This test case is also one of the samples included in the ALLSPD-3D distribution and is also detailed in the ALLSPD-3D user manual.[1] The fluid dynamics details are in Table 4. The single zone grid used in the baseline test case is shown in Figure 2.

Re_{dia}	*61,180*
T_{ref}	*300 K*
U_{ref}	*16 m/s*

Table 4 - Swirl can flow characteristics

Again, a single zone grid for the baseline case was manipulated to investigate the parallel efficiency with the added computational burden of chemical reactions and spray droplet tracking. The simple speedup grids were divided into equal zones with one per processor. The scaled speedup tests were performed on grids derived from their respective simple speedup test by refining them in the circumferential direction. (ALLSPD-3D calculates axisymmetric and two-dimensional cases by using periodic boundary conditions which requires only two points in the relevant direction.) Again, each manipulation of the grid maintained roughly the same number of points per zone and per processor as the baseline test case.

Tables 5 and 6 detail the grids used in each transition duct test case.

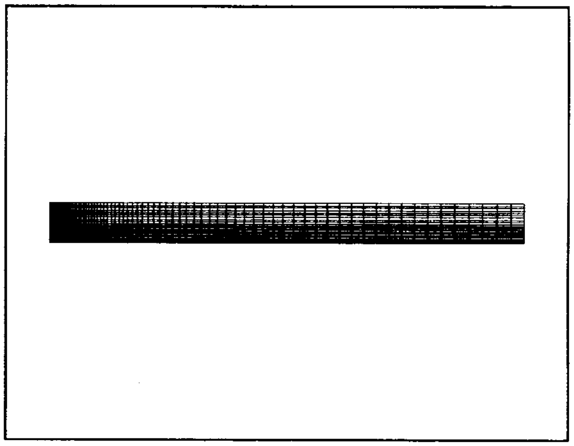

Figure 2 - Single zone grid (81x2x61=9882 points) for baseline swirl can (sparsed in radial direction for better visualization)

NUMBER OF ZONES	ZONE DIMENSION S	POINTS PER ZONE	TOTAL NUMBER OF POINTS
1	81 x 2 x 61	9882	9882
2	41 x 2 x 61	5002	10004
4	41 x 2 x 31	2542	10168
8	21 x 2 x 31	1302	10416
16	21 x 2 x 16	672	10752

Table 5 - Swirl can grids for simple speedup tests

NUMBER OF ZONES	ZONE DIMENSION S	POINTS PER ZONE	TOTAL NUMBER OF POINTS
1	81 x 2 x 61	9882	9882
2	41 x 4 x 61	10004	20008
4	41 x 8 x 31	10168	40672
8	21 x 16 x 31	10416	83328
16	21 x 32 x 16	10752	172032

Table 6 - Swirl can grids for scaled speedup tests

Again, direct comparisons for each test case were made since the multiple zones of the grids introduce extra points. The simple tests and the scaled tests were run on the shared memory, multiple processor SGI Power Challenge L workstation in addition to the cluster of IBM RS/6000 Model 590 workstations using ethernet and ATM networking.

Results

Speedup is defined as the CPU time of the serial code for a particular test case divided by the wall clock or elapsed time of the parallel code for the same test case. The parallel efficiency is the speedup divided by the number of processors.[3] Equations 1 and 2 show these definitions in a more mathematical form.

$$S = \frac{T_{cpu_{serial}}}{T_{wall_{parallel}}}$$

Equation 1 - Definition of Parallel Speedup

$$E = \frac{S}{N}$$

Equation 2 - Definition of Parallel Efficiency

All test cases were run on dedicated workstations. A cluster of sixteen IBM RS/6000 Model 590 workstations with ethernet and ATM networks and a single SGI Power Challenge L workstation with eight CPUs were used for the tests. The sixteen zone test cases were not run on the SGI Power Challenge L to keep the ratio of one grid zone per processor for all tests. The RS/6000 workstations used PVM version 3.3.10 while the SGI workstation used SGI Array version 2.0 which contains a version of PVM tuned for SGI workstations by SGI.

Each test case was run for 100 iterations and timed with the UNIX command *timex*. This number was chosen to allow for sufficient number of iterations to overshadow the start up effects such as reading in the grid but not to be so long as to preclude running all the tests within the time period allotted for dedicated usage of the computers. Once the tests were run, the timings were used to determine the parallel speedup and efficiency for each.

Simple speedup

The first advantage of parallel processing is immediately obvious in the tests of parallel speedup on the simple grids. Figure 3 shows the reduced memory needs arising from using multiple processors. The graph plots the number of processors against the normalized memory requirement for the transition duct test case run on the IBM workstations as well as the swirl can test case for compilations on the IBM and SGI workstations. The memory required was determined by the UNIX command *size* and normalized using the single processor serial code memory requirement.

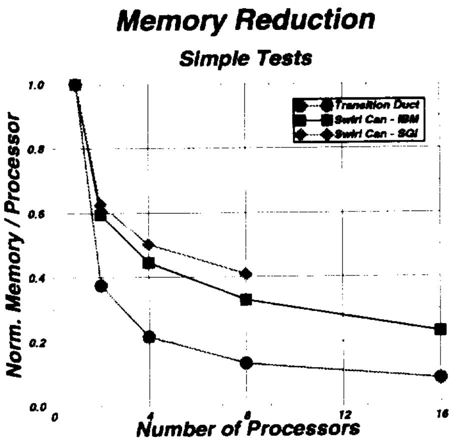

Figure 3

The transition duct shows the most dramatic memory reduction. With four processors, the per processor memory is only about 20% of the single zone test case. Thus, four workstations in parallel would need less aggregate memory than a single machine computing the problem serially because of the way ALLSPD-3D does memory management. Sixteen processors would need less than 10% of the memory needed by the single zone test case on a single CPU workstation. The swirl can test case does not show as dramatic a reduction, but the memory savings are still significant. The memory needs of the IBM and SGI executables are slightly different presumably because of differences in optimization and compiler technology. Even so, both platforms need less than half the amount of memory for each of four processors than for a single zone test on a serial processor.

The parallel speedup is the next advantage of running a test with multiple processors. Figure 4 shows the parallel speedup of the transition duct using the ethernet and ATM networks. Ideal speedup would be having the code run twice as fast with two processors, four times as fast with four processors, and so on. The graph shows that when ethernet networking is used, parallel speedup rolls off after only four processors. As a matter of fact, the turnaround time for the serial code is better than for the sixteen processor parallel code on this test. The ATM network fairs a bit better, but it rolls off at eight processors. However, the parallel code still runs faster than the serial code with ATM networking even though sixteen processors are communicating at the same time on every iteration.

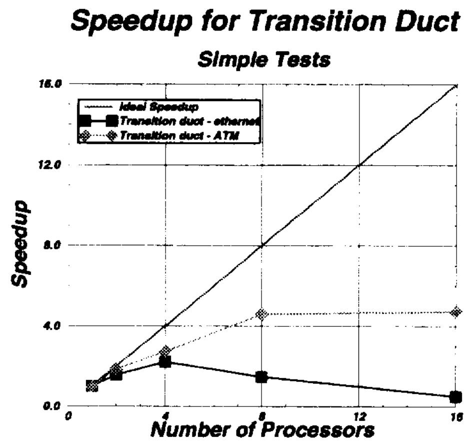

Figure 4

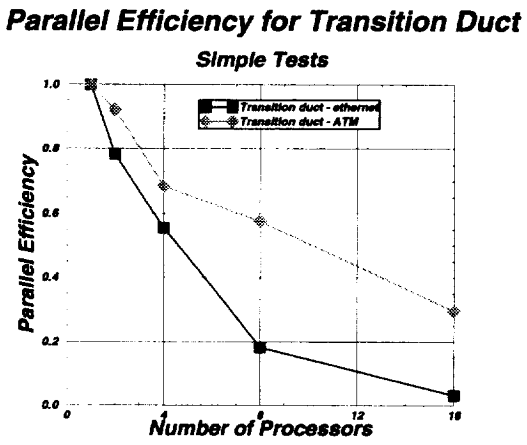

Figure 5

The parallel efficiency for these tests are plotted in Figure 5. Ideal parallel efficiency is 1.0 or 100%, i.e., two processors run twice as fast as one for the same problem. Again, the poor performance of the ethernet network shows itself. ATM networking does encounter a significant drop in parallel efficiency for sixteen processors, but the roughly 60% efficiency with only eight processors is quite acceptable.

The parallel speedup for the swirl can test cases are shown in Figure 6. In addition to the effects of networking on the speedup, we can see the effects of adding chemical reactions and spray modelling to the flow simulation. Adding these features increases the computation to communication ratio for the processors and can also cause the processors to communicate their per iteration results at slightly different times. This would help to reduce the network contention, especially for shared medium networks such as ethernet.

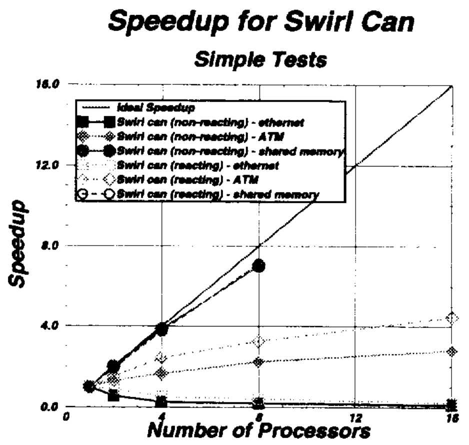

Figure 6

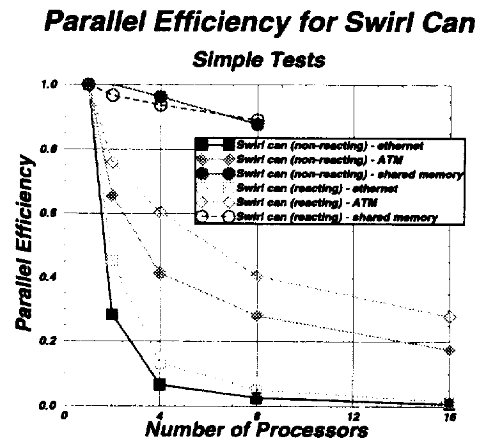

Figure 7

Again, the ethernet test runs show disappointing parallel speedup. This time, however, the ethernet is so overwhelmed by the large data transfer packets hitting the network at the same time that the serial code performs better for all cases. This is because the size of the data packets transferred after every iteration are sized on the maximum possible face. In this case, the actual amount of needed information is much smaller since the zone interfaces are J-K faces and the packets are sized by the I-K faces. The ATM network is decidedly better than the ethernet merely by having speedup values greater than one, but a maximum parallel speedup of only three or four for the sixteen processor tests is a moot improvement. The shared memory test runs on the SGI Power Challenge L workstation achieve near ideal parallel speedup. As a matter of fact, the two processor test case reaches super-linear speedup. This is most likely due to memory cache effects. In all networks the addition of chemical reactions improves the parallel speedup with the ATM network benefitting most. The shared memory run benefits least from the increase in computation to communication ratio because the shared memory "network" provides almost infinite bandwidth and almost zero latency.

The parallel efficiency for the swirl can test cases plotted in Figure 7 reflect the same trends. The ethernet tests show a marked improvement in parallel efficiency when chemical reactions are computed for the two processor case, but ethernet is still an overall poor performer for rest of the test cases. The ATM network has better overall parallel efficiency than ethernet with an almost constant improvement from the addition of chemical reactions. The shared memory version of PVM again provides the best parallel efficiency with little practical difference between having chemical reactions computed or not.

Scaled speedup

The scaled tests explored the effect of maintaining a constant computation to communication ratio for each processor on parallel speedup and efficiency. In the simple tests, the continual division of the grid into smaller pieces for each processor to work on kept decreasing the computation to communication ratio. By scaling the problem size with the number of processors, another advantage of parallel processing becomes apparent: the ability to run a large flow simulation on many workstations that would not be practical to run on a single workstation.

The parallel speedup results for the transition duct tests are plotted in Figure 8. Comparison to Figure 4 readily shows a significant improvement in speedup. The ethernet network again rolls off at four processors while the ATM network continues to speedup across the full range.

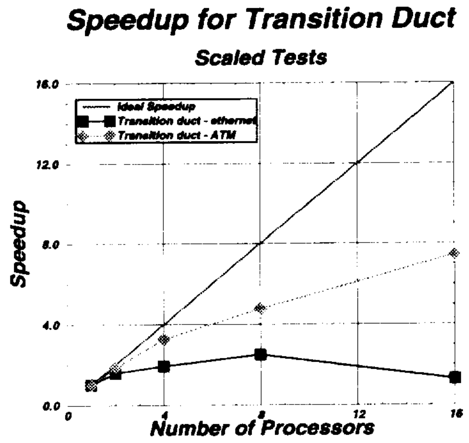

Figure 8

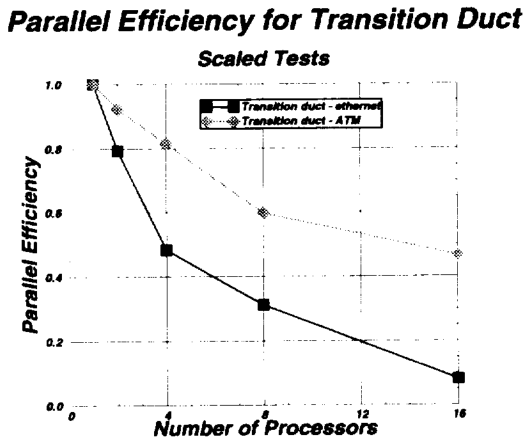

Figure 9

Figure 9 shows the parallel efficiencies plotted for the same tests. The ethernet tests show acceptable performance out to four or eight processors, and the ATM network has increased parallel efficiency all the way out to sixteen processors. This is a vast improvement compared to the efficiencies for the simple tests plotted in Figure 5.

The swirl can tests with the scaled grids shows similar improvements in parallel speedup as evidenced in Figure 10. While the ethernet network does not benefit as greatly by the increased problem size as in the transition duct tests, comparison to Figure 6 shows considerable improvement even if it is not enough to warrant running in parallel when only an ethernet is available for communication. The ATM network benefits from the scaled problem sizes with the parallel speedup almost doubling. The shared memory version is practically unaffected by the scaling except that the single workstation needs a larger amount of total memory. For all versions, the additional computational burden of chemical reactions has a constant but negligible improvement in parallel speedup.

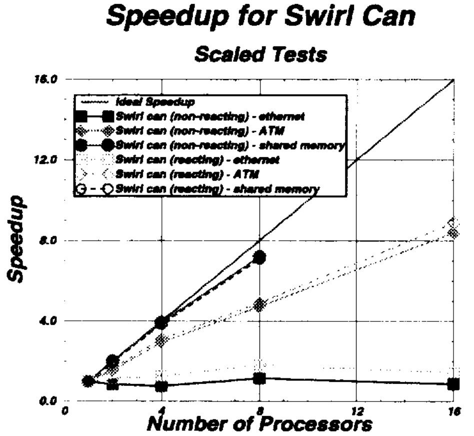

Figure 10

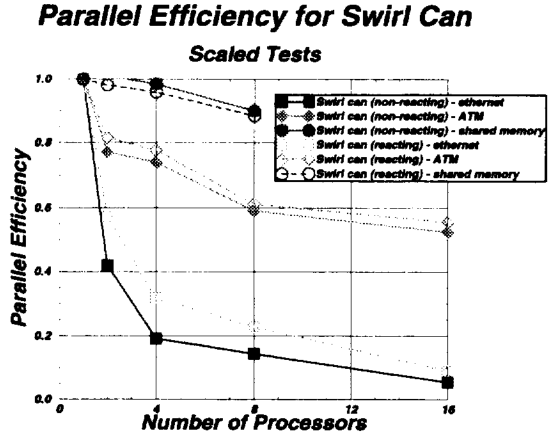

Figure 11

The parallel efficiencies for these tests are plotted in Figure 11. Comparison with Figure 7 shows improvements for the ethernet and ATM networks, but only small changes for the shared memory tests. The ATM results do show an anomaly at the two to four processor points. Currently, there is no explanation for such a drop or increase in parallel efficiency for these test cases. Again, the addition of chemical reactions to solve improves the efficiency for all communication media, but not by as significant an amount as in the simple tests.

Concluding Remarks

ALLSPD-3D can simulate flows on clusters of UNIX workstations or multiple processor workstations with shared memory using PVM for data transfer. This gives the ability to solve large problems on modest machines, but results in a communication-bound problem with limits on speedup. Faster networks alleviate the situation, but not completely. Shared memory machines provide the fastest communications but can be expensive and require enough memory for the entire problem to be solved. The network bandwidth and latency determine when adding more processors degrades turn-around time instead of improving it. Adding additional computational burdens such as chemical reactions and spray to the simulation allows more processors to be added before this breakpoint is reached. Minimizing the amount of data to be transferred is critical and is best influenced by the grid generation. When making a grid for use with ALLSPD-3D, one should keep the zones close in size and make the face sizes as small as possible. Otherwise, code modifications would be necessary to minimize the amount of data transferred.

Also, having a single source code which compiles into the serial or parallel version has resulted in the need to re-grid the test case whenever the number of processors increases. At best, this is a tedious process; at worst, all the input files for a particular test case need to be regenerated because the cell locations are different.

References

1. Chen, K.-H., Duncan, B., Fricker, D., Lee, J., and Quealy, A., "ALLSPD-3D Version 1.0a", NASA TM-107204, April 1996
2. Geist, A, Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., and Sunderam, V., PVM: Parallel Virtual Machine A User's Guide and Tutorial for Network Parallel Computing, MIT Press, 1994.
3. Ragsdale, S., Parallel Programming, McGraw-Hill, Inc., 1991, pp. 20-21.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (*Leave blank*)	**2. REPORT DATE** June 1997	**3. REPORT TYPE AND DATES COVERED** Technical Memorandum

4. TITLE AND SUBTITLE

Parallel ALLSPD-3D: Speeding Up Combustor Analysis Via Parallel Processing

5. FUNDING NUMBERS

WU-523-26-33

6. AUTHOR(S)

David M. Fricker

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Lewis Research Center
Cleveland, Ohio 44135-3191
and
U.S. Army Research Laboratory
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E-10786

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001
and
U.S. Army Research Laboratory
Adelphi, Maryland 20783-1145

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM-107489
AIAA-97-3295
ARL-MR-369

11. SUPPLEMENTARY NOTES

Prepared for the 33rd Joint Propulsion Conference and Exhibit cosponsored by AIAA, ASME, SAE, and ASEE, Seattle, Washington, July 6-9, 1997. Responsible person, David M. Fricker, organization code 5830, (216) 433-5960.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Categories 07, 61, and 64

This publication is available from the NASA Center for AeroSpace Information, (301) 621-0390.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

The ALLSPD-3D Computational Fluid Dynamics code for reacting flow simulation was run on a set of benchmark test cases to determine its parallel efficiency. These test cases included non-reacting and reacting flow simulations with varying numbers of processors. Also, the tests explored the effects of scaling the simulation with the number of processors in addition to distributing a constant size problem over an increasing number of processors. The test cases were run on a cluster of IBM RS/6000 Model 590 workstations with ethernet and ATM networking plus a shared memory SGI Power Challenge L workstation. The results indicate that the network capabilities significantly influence the parallel efficiency, i.e., a shared memory machine is fastest and ATM networking provides acceptable performance. The limitations of ethernet greatly hamper the rapid calculation of flows using ALLSPD-3D.

14. SUBJECT TERMS

Combustion; CFD; Parallel processing

15. NUMBER OF PAGES	10
16. PRICE CODE	A02

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	