

# Development and applications of a large scale fluids/structures simulation process on clusters

G.P. Guruswamy \*

*NASA Advanced Supercomputing (NAS) Division, Ames Research Center, MS T27B-1, Moffett Field, CA 94035-1000, United States*

Received 24 February 2005; received in revised form 20 December 2005; accepted 26 March 2006

Available online 14 July 2006

## Abstract

A modular process for efficiently solving large-scale multidisciplinary problems using single-image cluster supercomputers is presented. The process integrates disciplines with diverse physical characteristics while retaining the efficiency of individual disciplines.

Computational domain independence of individual disciplines is maintained using a meta programming approach. The process integrates disciplines without affecting the combined performance. The procedure includes an efficient load balancing scheme suitable for parallel computers. Results are demonstrated for large-scale aerospace problems. The super scalability and portability of the approach is demonstrated on several parallel supercomputers.

© 2006 Elsevier Ltd. All rights reserved.

## 1. Introduction

During the last decade, significant progress has been made in the area of supercomputing using cluster or parallel computers. This technology has started making impact on major engineering fields such as aerospace design. The aerospace community was one of the major driving forces behind supercomputing technology using serial computers and also playing a major role in adapting parallel computers for its ever-increasing computational needs. Because of the large effort required to restructure software, particularly in the area of multidisciplinary applications using high-fidelity equations, there is latency in using parallel computers in day-to-day use for analysis and design of aerospace vehicles. This paper presents a technology that leads single-image cluster supercomputing to the real-world aerospace applications.

Large-scale multidisciplinary problems are common in engineering design. They involve the coupling of many high-fidelity single disciplines. For example, aeroelasticity of large aerospace vehicles that involves strong coupling

of fluids, structures and controls is an important element in the design process [1]. It is observed that an instability due to strong fluid/structure interactions can occur soon after a launch vehicle such as the X-34 [1] gets separated from the aircraft. From the results presented in [1], it can be concluded that low-fidelity software was not adequate to completely understand the instability phenomenon which involved non-linear flows coupled with structural motions.

Methods that couple fluids and structures by using low-fidelity methods, such as the linear aerodynamic flow equations coupled with the modal structural equations, are well advanced. Although low-fidelity approaches are computationally less intensive and used for preliminary design, they are not adequate for the analysis of a system that experiences non-linear flow/structure interactions. High-fidelity equations, such as the Euler/Navier–Stokes equations (ENS) for fluids directly coupled with finite elements equations (FE) for structures are needed for accurate aeroelastic computations for which complex fluid/structure interactions are critical. Using these coupled methods, design quantities such as structural stresses can be directly computed. However, high-fidelity equations involve additional complexities from numerics such as higher-order terms.

\* Tel.: +1 650 604 6329; fax: +1 650 604 1095.

E-mail address: [guru.p.guruswamy@nasa.gov](mailto:guru.p.guruswamy@nasa.gov)

Therefore, the coupling process is more elaborate when using high-fidelity methods than it is for calculations using linear methods. Moreover, high-fidelity methods are computationally intensive and need efficient algorithms that run on parallel computers. Fig. 1 illustrates the different levels of fidelity in both fluids and structures. The arrows shown in Fig. 1 are valid for both disciplines. Higher fidelity accounts for more physics, but it makes the analysis of complex geometries more difficult. Modeling complex geometries is easier with the use of lower fidelity methods.

Significant advances have been made for single disciplines in both computational fluid dynamics (CFD) using finite-difference approaches [2] and computational structural dynamics (CSD) using finite-element methods [3]. These single discipline methods are efficiently implemented on parallel computers. For aerospace vehicles, structures are dominated by internal discontinuous members such as spars, ribs, panels, and bulkheads.

The finite-element (FE) method, which is fundamentally based on discretization along physical boundaries of different structural components, has proven to be computationally efficient for solving aerospace structural problems. The external aerodynamics of aerospace vehicles is dominated by field discontinuities such as shock waves and flow separations. Finite-difference (FD) computational methods have proven to be efficient for solving such flow problems.

Parallel methods that can solve multidisciplinary problems are still under development. Currently, there are several multidisciplinary parallel codes that solve a monolithic system of equations using unstructured grids, mostly

modeling Euler flow equations. This single computational domain approach has been in use for several years for solving fluid–structural interaction problems by using finite element approach [4]. While using the single domain approach, the main bottleneck arose from ill-conditioned matrices associated with two physical domains with large variations in stiffness properties. The drop in the convergence rate from the rigid case to the flexible case in [5] indicates the weakness of the single domain approach. As a result, a sub-domain approach is needed where fluids and structures are solved in separate domains, and solutions are combined through boundary conditions. Ref. [6] provides a comprehensive summary of such methods.

This paper presents an efficient alternative to the monolithic approach based on an independent domain approach that is suitable for massively parallel systems. Fluids and structures disciplines are interfaced through discipline-independent wrappers. Some of the materials presented earlier by the author and his co-workers are included in this paper with appropriate references to provide the necessary background.

## 2. Domain decomposition approach

A method highly suited for state-of-the-art cluster supercomputers is presented in this paper. When simulating aeroelasticity with coupled procedures, it is common to deal with fluid equations in an Eulerian reference system and structural equations in a Lagrangian system. The structural system is physically much stiffer than the fluid

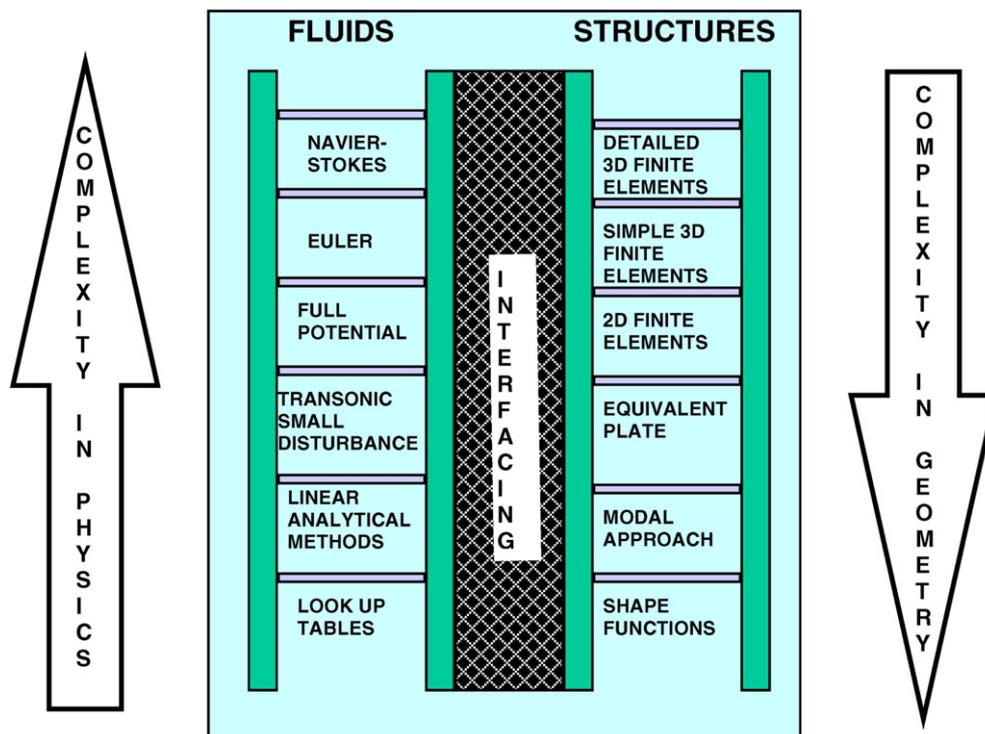


Fig. 1. The effect of fidelity on complexities in both physics and geometry of aerospace vehicles (left and right arrows are applicable for both fluids and structures).

system, and the numerical matrices associated with structures are orders of magnitude stiffer than those associated with fluids. Therefore, it is numerically inefficient or even impossible to solve both systems using a single numerical scheme.

Guruswamy and Yang [7] presented a numerical approach to solve this problem for two-dimensional airfoils by independently modeling fluids using the FD-based transonic small-perturbation (TSP) equations and structures using FE equations. The solutions were coupled only at the boundary interfaces between fluids and structures. The coupling of solutions at boundaries can be done either explicitly or implicitly. This domain-decomposition approach allows one to take full advantage of state-of-the-art numerical procedures for individual disciplines. This coupling procedure has been extended to three-dimensional problems and has been incorporated in several advanced serial aeroelastic codes such as ENS3DE [8] and ENSAERO [9], which use the Euler/Navier–Stokes equations for fluids and modal equations for structures. The main emphasis in this paper is further development of these methods for parallel computers using a highly portable and modular approach. In this effort, CSD capability is further extended to use direct finite elements in addition to the modal equations.

### 3. Parallelization effort

Significant progress has taken place in high-fidelity, single discipline codes such as NASTRAN [10] for structures, and OVERFLOW [11] for fluids. However, efforts to combine these single discipline codes into a multidiscipline code or process are still in progress. Several attempts have been made to expand single discipline codes to multidiscipline codes such as ENSAERO [9], ENS3DE [8], and STARS [12]. These codes are tightly dependent on pre-selected individual disciplines. Due to the rapid progress that is taking place in individual disciplines, freedom is needed to replace individual modules with improved ones. This requires a different approach than traditional code development.

One of the major drawbacks of using codes with high-fidelity methods is the high demand for computing resources in terms of both memory and speed. Parallel computer technology initiated new ways of solving individual disciplines with scalable performance on multiple processors. The use of the computer industry standard Message Passing Interface (MPI) [13] utility led to a successful parallel solution procedure.

In order to couple different discipline domains, communication between domains is accomplished through an interface at the end of each time step. This is achieved by creating an inter-disciplinary communicator using an MPI application programming interface (API) called `mpi_intercomm_create` [14]. For aeroelastic computations which involve fluids and structural domains, aerodynamic loads are converted into structural loads through the fluid–structure interface. Furthermore, the structural

deformation is passed to the fluid domain through this interface. Then, the surface grid is deformed according to the structural deformation. In addition, control surface deflection computed in a control domain is superimposed on the deformed surface grid.

The overall communication design is shown in Fig. 2. In using the MPI library, a communicator is used to identify a group of processors that can communicate with one another within the same group. Each group is represented by a box defined by dashed lines, as shown in Fig. 2. In this case, however, only one processor is assigned to each group for a single coupled analysis. All allocated processors have a common communicator called `mpi_comm_world` as shown in Fig. 2. The MPIAPI, `mpi_comm_create`, creates a distinct communicator, denoted as `mpirun_app_com` for each group of computational processors when it loads the executable program onto the processors. Using the `mpirun_app_com` communicator, any processor can communicate with others within a group. Communications are also defined using the MPIAPI `mpi_intercomm_create` to communicate between different discipline modules or different groups. They are denoted by solid and dashed lines with arrows, respectively.

Furthermore, the MPI library has the functionality to create a new communicator for a subset of the allocated processors. Communicators for each discipline are defined so that collective operations can be accomplished within a discipline module. Once a communicator for each discipline is defined, it is quite convenient to do a collective operation within a discipline, such as computing lift and drag coefficients. The communication design shown in Fig. 2 only explains the coupling of three different computational modules, e.g., fluids, structures, and controls. However, if needed, additional modules can be easily added to the process.

The communication design for a single coupled analysis can be further extended to perform multiple analyses concurrently. Fig. 3 shows the extension of the communication design for concurrent multiple analyses. In contrast to a single coupled analysis, several processors are assigned to each group. In this figure, each group has  $N$  processors, which is the number of different cases running concurrently. They are locally ranked from zero to  $N-1$  within a group. In the first run, the initialization data within a group are distributed from the leading processor of each group through a broadcast call using `mpirun_com` communicator. This makes it easy to distribute initial input data within a group. Once the initial data distribution is completed, each processor of a group will participate in a different analysis. For example, if  $N$  cases with different initial angles of attack are concurrently executed, each processor within a group has the same grid data of a zone but computes solutions for the different flow conditions. Within the flow domain, after solving the flow equations at every time step, each zone needs to exchange zonal boundary data with adjacent zones to advance to the next step. For this purpose, data communication is limited only among

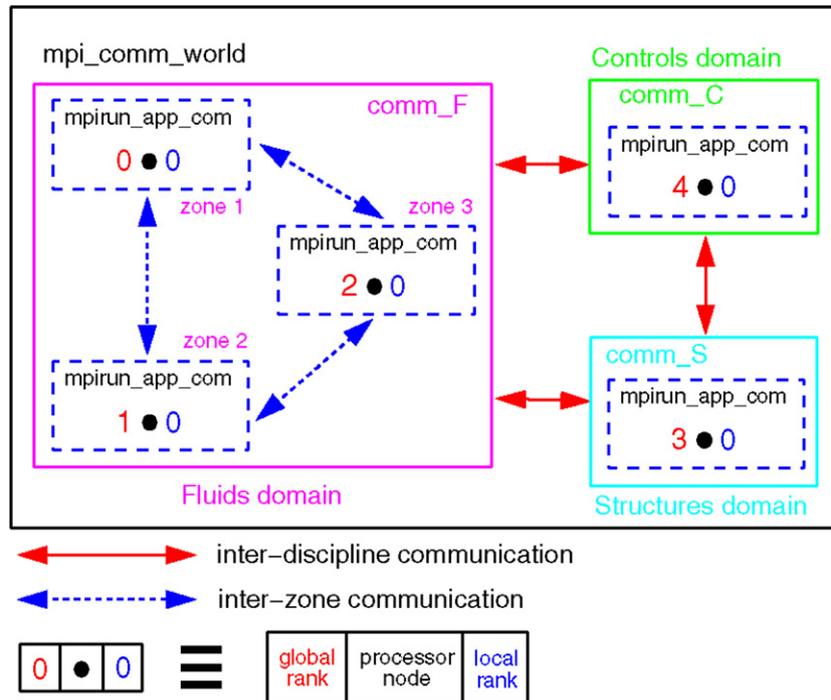


Fig. 2. Data communication design for multizonal applications on parallel computers.

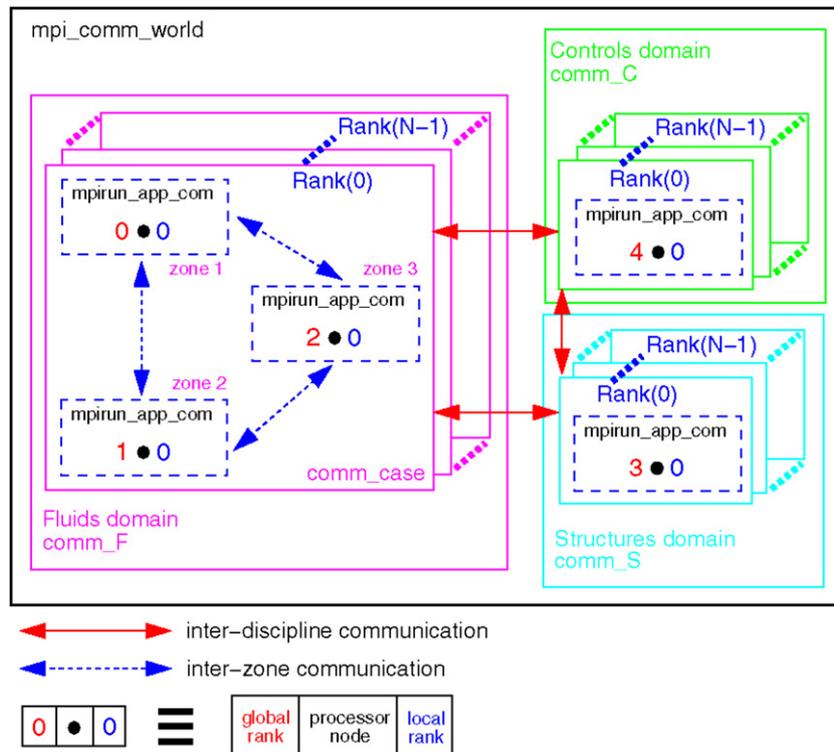


Fig. 3. Multilevel communication among fluids, structural and controls domains.

computational processors with the same local rank. In this communication strategy, each processor can distinguish itself from other processors assigned to different cases. Therefore, each processor having different local ranks can participate in different simulations. For multiple multidisci-

plinary simulations, the same communication strategy is applied for data exchange among the discipline domains. Further details of this process are described in [15]. This high-fidelity multidisciplinary analysis process along with software which includes solution modules and

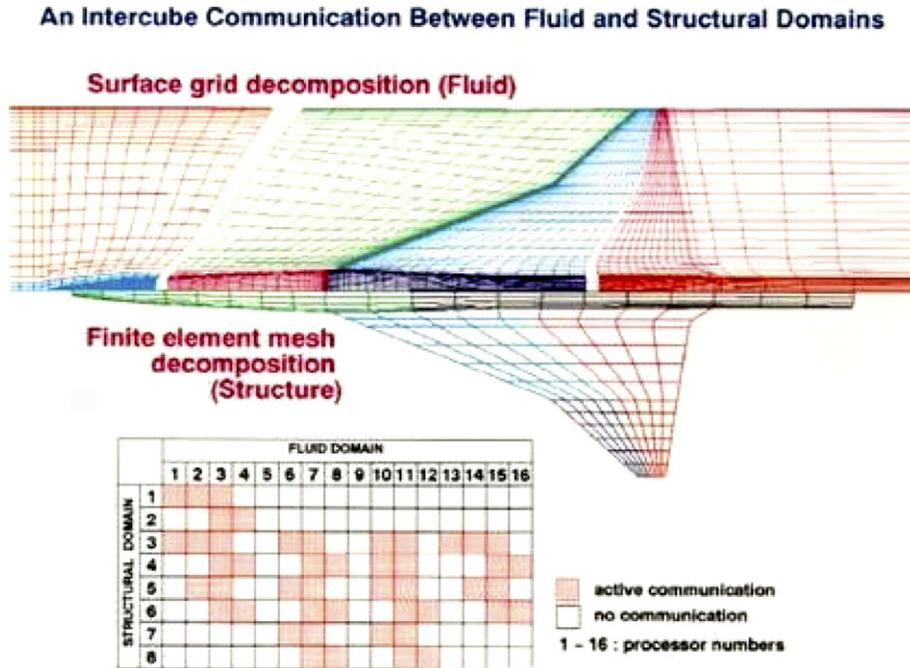


Fig. 4. Typical fluid structures communication on a parallel computer.

MPI/MPIAPI library calls is referred to as HiMAP, High Fidelity Multidisciplinary Analysis Process.

A typical fluid structure communication is illustrated in Fig. 4 for an aerospace vehicle. In this case, 16 and 8 processors are assigned to fluids and structures, respectively. The shaded areas show active communication and blank areas show no communication. Active communication takes place where fluid zones are in contact with structural zones.

#### 4. Load balancing

Efficient methods to solve fluids and structures commonly use a domain decomposition approach based on zonal grids [2]. Each zone may contain a CFD or CSD, grid specific for a component of the full configuration. To efficiently solve complex configurations with large numbers of varying size grid zones, a robust load balancing approach is needed. Load balancing can be achieved as described below.

In this work, load balancing is achieved by a zone-coalescing and partitioning approach. This parallelization approach achieves the goal of load-balanced execution provided that there are enough processors available to handle the total number of zones. One-to-one assignment of zones to processors does not guarantee an efficient use of the parallel system. The processors might be working with less than the optimal computational load and performing a lot of expensive inter-processor communications, and hence be data-starved. Both problems are alleviated by introducing a zone-coalescing and splitting capability to the parallelization scheme. In zone coalescing, a number

of zones are assigned to a single processor resulting in economy of the computational resources, and also in a more favorable communications-to-computations ratio during the execution. This method was first tried for simple configurations [16] and its general capability is shown in Fig. 5. This figure illustrates that a single zone can be split into several sub-zones or several sub-zones can be merged into a single super-zone depending on the memory available per processor.

In order to obtain maximum performance for analyzing complex configurations that involve grid zones with large bandwidth of grid sizes, a further extension of the zone coalescing-splitting approach is implemented.

A number of zones is assigned to each processor depending on its memory size. For example, it is found that an SGI Origin3000 processor can handle a maximum grid size of 500,000 pts for computations using CFD codes such as ENSAERO [9]. The assignment of a zone to a processor begins with small zones and progresses to larger zones. In this process, any zone larger than the maximum size is partitioned. The load balancing scheme used is illustrated in Fig. 6.

#### 5. Large-scale applications

The method presented here is suitable for large-scale multidisciplinary analysis. It has been tested using the Euler/Navier–Stokes-based flow solver modules such as ENSAERO [9], USM3D [17], and finite element based structures modules such as NASTRAN [10,18]. The method has been demonstrated for large-scale aeroelastic applications that required 16 million fluid grid points and

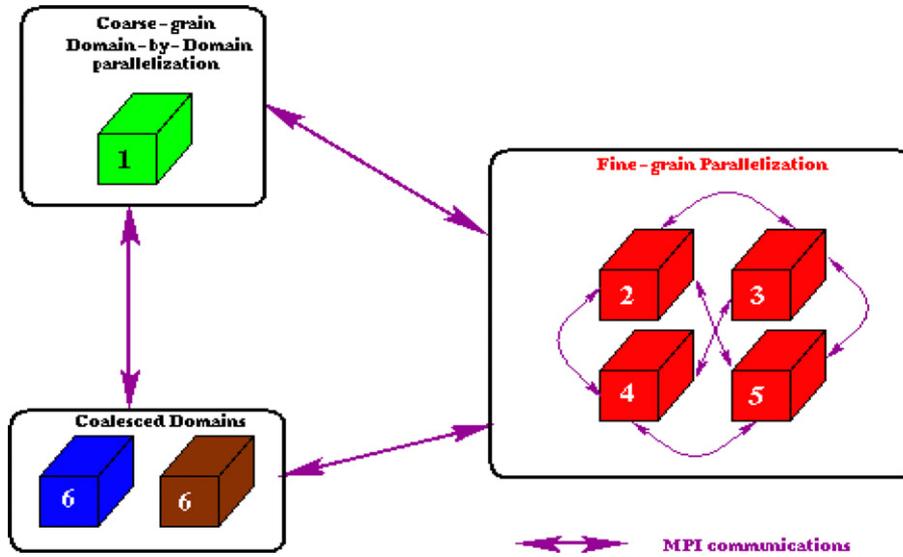


Fig. 5. Zone (Domain) coalescing-partitioning approach.

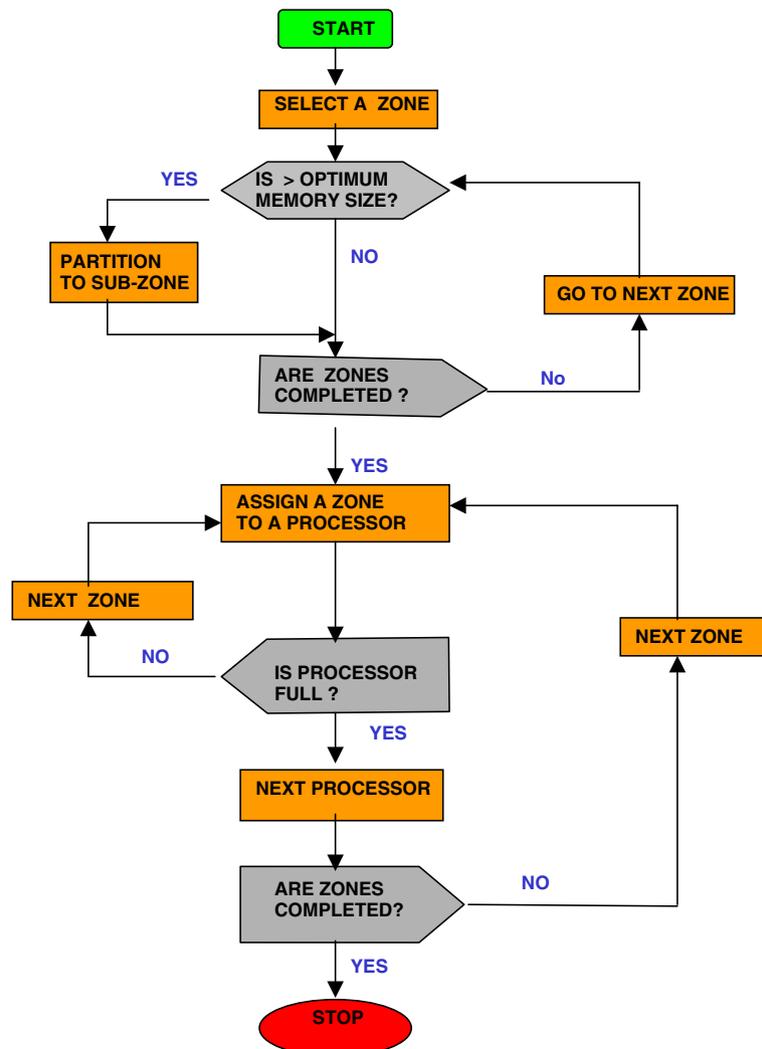


Fig. 6. The processor memory filling scheme for improving the load balancing.

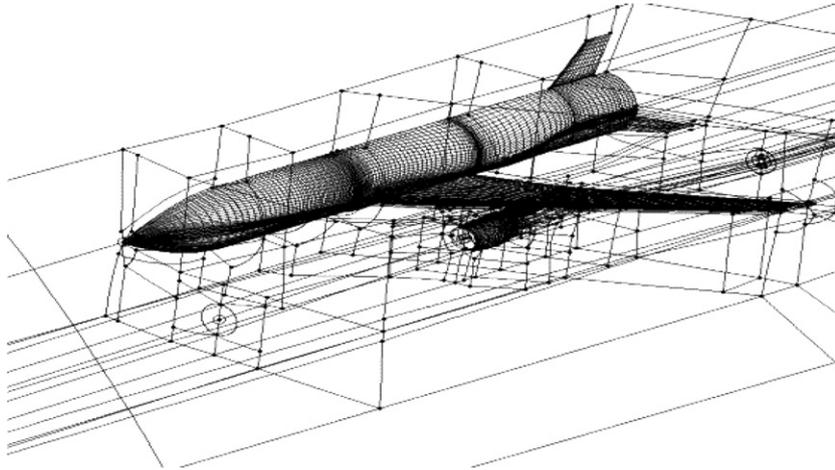


Fig. 7. Complex grid arrangement for a typical transport aircraft.

20,000 structural finite elements. Cases have been demonstrated using up to 228 processors on an IBM SP2, and 256 processors on an SGI Origin2000 computer. Typical configurations analyzed are full subsonic and supersonic aircraft.

An example of a complex multi-zone grid system is shown for a typical transport aircraft in Fig. 7. The grid system is made up of 34 zones and the number of grid points varies from 30,000 to 427,000 per zone. If each zone is assigned to a processor, the load efficiency of the processor assigned to the smallest zone will be about 7%. The load balancing scheme is applied to improve efficiency.

Fig. 8 illustrates the results of applying the load-balancing scheme to the multi-zone grid system. The dashed line shows a plot of grid size against the zone number before load balancing is applied. The solid line shows the plot of modified grid size against zones (processors) after the regrouping of zones. The number of processors needed is reduced from 34 to 28. The ratio of the minimum to maximum size per processor increased from 7% to 81%. Thus a maximum factor of increase in efficiency per processor equal to 11.6 can be achieved. Since the efficiency results from both reduction in number of processors and increase in size of the grid assigned to each processor, an overall effi-

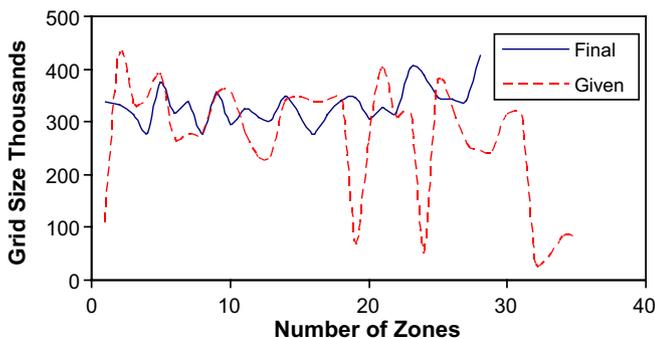


Fig. 8. Grid points per processor with and without processor filling scheme.

ciency factor,  $E$ , can be defined as the square of the ratio of number original zones to number of final zones (assigned to processors) after applying the load balancing:

$$E = \left\{ \frac{ZO}{ZN} \right\}^2 \quad (1)$$

where ZO is number zones before load balancing and ZN is the number of regrouped zones after load balancing. For the data shown in Fig. 8,  $E$  is 1.5.

Parallel computations were made on SGI's Origin2000 computer. Fig. 9 shows one of the five structural modes from the finite element computations of the selected transport aircraft. Each mode was represented by 2,100 degrees of freedom. One Origin2000 processor was assigned to the modal data. Solutions from HiMAP were obtained using ENSAERO module [9] along with a parallel MBMG (MultiBlock Moving Grid) [19] moving grid module. A typical aeroelastic solution is shown in Fig. 10. The colors represent the surface pressure coefficients. The stability and convergence of the GO3D [20] upwind algorithm in the ENSAERO module was not affected by the redistribution of patched grids to different processors.

In large-scale aerospace problems, grid topologies of the configurations are predetermined based on design needs. Grid size and number of blocks are directly related to the complexity of configuration and fidelity of equations solved. Quite often, parallel efficiency can be addressed only after the grids are designed. The procedure presented here will help in cost-effective computations.

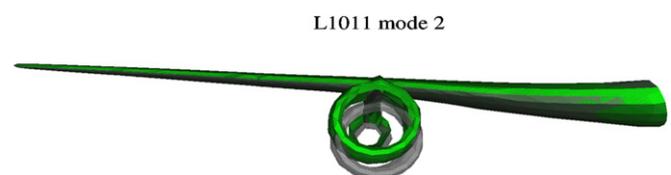


Fig. 9. Typical structural twist mode of an aircraft (black = original, green = deformed). (For interpretation of the references in color in this figure legend, the reader is referred to the web version of this article.)

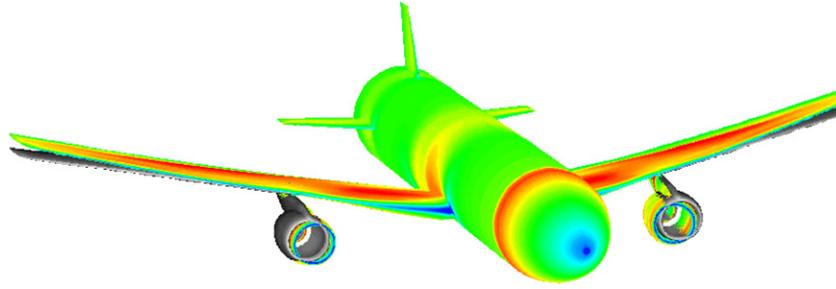


Fig. 10. Pressure coefficient map of a deformed aircraft.

Some of the results from applying methods developed here to several large aerospace problems are summarized in Fig. 11. The complexity of the problem increases significantly from a simple wing-body model to a full configuration as shown by increase in grid size and number of blocks. The present approach shows an improvement in efficiency factor  $E$  as the complexity of the configuration increases.

**6. Portability and performance**

The process developed here has been successfully ported to massively parallel processor (MPP) platforms of SGI, SUN, and IBM. The flow solver performed at a rate of 40 MFLOPS per processor on the Origin 2000 MPP platform. The super modular capability of HiMAP is demonstrated by plugging in the USM3D unstructured grid solver in place of the patched structured-grid solver and computing aeroelastic responses with minimal effort [17]. In [17], portability of this software to a workstation cluster was also demonstrated. HiMAP can also be used for the massively parallel uncoupled aeroelastic analysis [21]. A summary of results on different parallel computer systems is shown in Fig. 12. An exponential decrease in time per step is obtained on all systems. The differences in time taken per step by different computer systems for a given number of processors are due to difference in processor speeds. Almost linear scalability in performance of three-

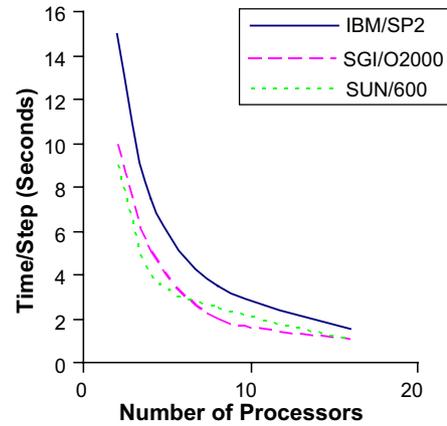


Fig. 12. Demonstration of portability and scalability.

level parallel HiMAP process was also demonstrated on a 256-node IBM SP2 MPP system [22].

The scalability of HiMAP was further improved for shared-memory configurations by implementing OpenMP parallelization per grid zone [23]. It was found that the architecture of HiMAP was also suitable for OpenMP parallelization which resulted in minimal changes to the software. Results in Fig. 13 and Table 1 based on data from [23] show the significant advantage of using OpenMP threads when efficient shared-memory hardware is available. It shows that better performance can be obtained if

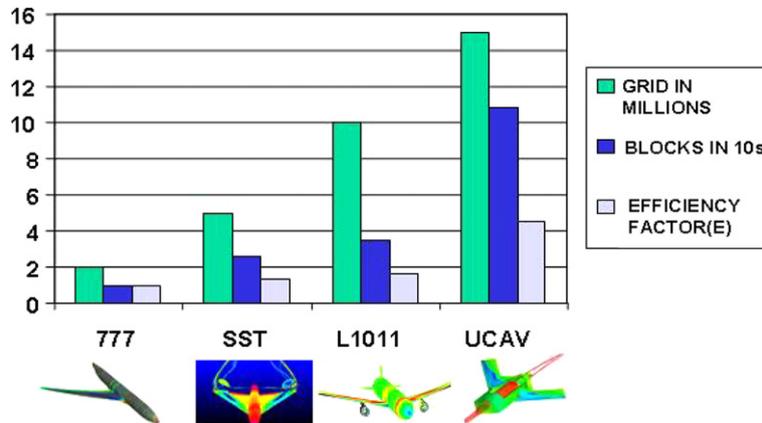


Fig. 11. Parallel efficiency factor for different complexity configurations.

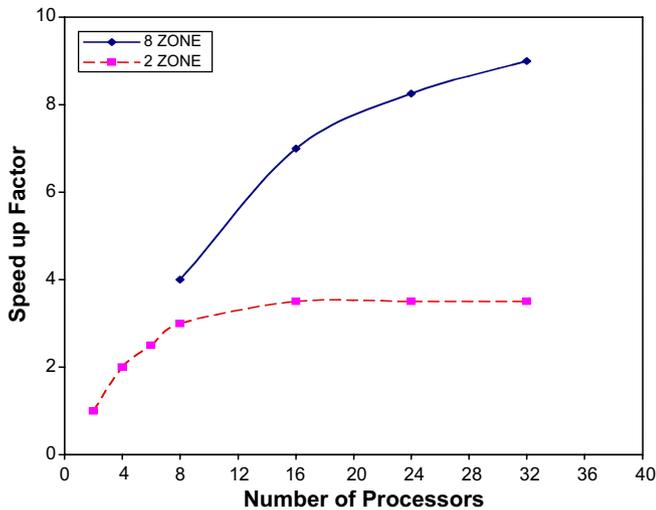


Fig. 13. Improved performance of OpenMP version [23] of HiMAP on Sun platform.

Table 1  
Percentage increase in scaling efficiency from OpenMP implementation

Processors	2 Zones	8 Zones
4	84	NA
6	70	NA
8	63	NA
16	44	79
24	32	66
32	24	66

NA: not applicable.

both MPI processes and OpenMP threads are increased. The case with eight MPI processes (8 zone case) scales better than the case with two MPI processes (2 zone case) for the same grid size. These results are obtained by introducing OpenMP threads to computationally intensive modules of HiMAP.

## 7. Conclusions

An efficient parallel process needed for computationally intensive analyses of aerospace vehicles is presented. The process can simulate aeroelasticity of aerospace vehicles using high-fidelity equations such as the Navier–Stokes equations for flows and finite-elements for structures. The process is suitable for both tightly coupled and uncoupled analysis. The process is designed to execute on massively parallel processors (MPP) and work-station clusters based on a multiple-instruction and multiple-data (MIMD) architecture. The fluids discipline is parallelized using a zonal approach while the structures discipline is parallelized using the sub-structures concept. Provision is also made to include the controls domain. Computations of each discipline are spread across processors using computer standard message passing interface (MPI) for inter-processor communications. MPI-based API is developed to run disciplines in parallel. In addition to inter-and-intra

discipline parallelizations, a massively parallel capability to run multiple parameter cases is implemented using a script system. The combined effect of three levels of parallelization is an almost linear scalability for multiple concurrent analyses that perform efficiently on MPP. Finally, this paper demonstrates an unique use of the latest parallel computer technology to the multidisciplinary analysis needed for the design of large aerospace vehicles. The scalable modular approach developed here can be extended to other fields such as bio-engineering and civil engineering.

## Acknowledgements

The author would like to thank Mark Potsdam of the US Army Aeroflightdynamics Directorate and Chansup Byun of Sun Microsystems for consultations provided in writing this paper.

## References

- [1] Blades EL, Ruth M, Fuhrmann HD. Aeroelastic analysis of the X-34 Launch Vehicle. AIAA paper April 99;1999–135.
- [2] Holst TL, Flores J, Kaynak U, Chaderjian N. Navier–Stokes computations, including a complete F-16 Aircraft. Progress in Astronautics and Aeronautics, AIAA, vol. 125, ISBN 0-930403-69-X, 1990.
- [3] Bathe KJ. Finite element procedures. Englewood Cliffs, NJ: Prentice-Hall; 1996.
- [4] Zienkiewicz OC. The finite element method. 3rd ed. McGraw Hill Book Company UK Ltd.; 1977.
- [5] Felker FF. A new method for transonic aeroelastic problems. AIAA-92-2123, 1992.
- [6] Schuster DM, Liu DD, Huttshell LJ. Computational aeroelasticity: success, progress, challenge. J Aircraft 2002;40(5):843–56.
- [7] Guruswamy GP, Yang TY. Aeroelastic time response analysis of thin airfoils by transonic code LTRAN2. Computers Fluids 1981; 9(4):409–25.
- [8] Schuster D, Vadyak J, Atta E. Static Aeroelastic analysis of fighter aircraft using a 3D Navier–Stokes algorithm. AIAA paper 90-0435, 1990.
- [9] Guruswamy GP. ENSAERO – a multidisciplinary program for fluid/structural interaction studies. Comput Syst Eng 1990;1(2–4):237–56.
- [10] NASTRAN User's manual: NASA SP-222 (08), June 1986.
- [11] Buning PG, Wong TC, Dillely AD, Pao JL. Computational fluid dynamics prediction of hyper-x stage separation aerodynamics. J Spacecraft Rockets 2001;38(6):820–6.
- [12] Gupta KK. Development of aeroelastic capability. J Aircraft 1996; 33(5).
- [13] Message Passing Interface Forum, MPI: A Message–Passing Interface Std, Univ of Tennessee, May 1994.
- [14] Fineberg S. MPIRUN: A loader for multidisciplinary and multizonal MPI applications. NAS News 1994;2(6).
- [15] Guruswamy GP, Byun C, Potsdam M, Farhangnia M. User's manual for HiMAP – a multilevel parallel multidisciplinary analysis process. NASA TM-1999-209578, September 1999.
- [16] Hatay F, Byun C, Jespersen D, Rizk Y, Guruswamy GP. A multi-level parallelization concept for high-fidelity multi-block solvers. Supercomputer 97, San Jose, November 1997.
- [17] Raj P, Goodwin SA, Sankar LN, Weed RA, Parikh PC, Bhat MK. Development and validation of a fast and accurate aeroservoelastic method on advanced parallel computing system. LG97ER0106, Lockheed Martin Report, July 1997.
- [18] Eldred L, Guruswamy GP. User's guide for ENSAERO\_FE parallel finite element solver. NASA TM-1999-208781, April 1999.

- [19] Potsdam M, Guruswamy GP. A parallel multiblock mesh movement scheme for complex aeroelastic applications. AIAA – 2001-0716.
- [20] Obayashi S, Guruswamy GP, Goojrian PM. Streamwise upwind algorithm for unsteady transonic flows past oscillating wings. AIAA J 1991;29(10):1668–77.
- [21] Byun C, Guruswamy GP. An efficient procedure for flutter boundary prediction on parallel computer. AIAA Paper 1995–1484, April 1995.
- [22] Farhangnia M, Guruswamy GP, Byun C. Performance and applications of himap on scalable computers. NASA/CP-1999 208757, January 1999.
- [23] Byun C. Porting experience with HiMAP on sun HPC environment. SUN Users meeting, Hawaii, April 2002.