THE NEW CHALLENGE OF COMPUTATIONAL AEROSCIENCE

F. Ron Bailey
NASA Ames Research Center, Moffett Field, California

Douglas L. Dwoyer
NASA Langley Research Center, Hampton, Virginia

Lester D. Nichols
NASA Lewis Research Center, Cleveland, Ohio

Abstract

This paper discusses NASA's Computational Aerosciences (CAS) Project of the High Performance Computing and Communications Program (HPCCP). The project is aimed at developing advanced, multi-disciplinary simulation capabilities for aerospace vehicle and propulsion system design. It is also aimed at overcoming computational performance barriers by accelerating the development of parallel computer technology. The goals and approach of the CAS Project are described and the challenges to its implementation are addressed. Specific vehicle class simulations to be demonstrated and the principal multidisciplinary modeling approaches to be emphasized are described. The computational speed and memory requirements for representative multidisciplinary applications are estimated. Finally, the state of parallel computer technology including programing issues and the results of performance measurements are explored.

I. Introduction

The influence of computational analyses and design on the aerospace field has grown steadily and at a very rapid pace for the past 30 years. In fact, the aerospace industry has been a leader in the use of numerical simulation because it has contributed significantly to improved vehicle efficiency and performance and has aided in reducing design cost. Most simulations to date have been single discipline in nature, i.e., aero-dynamics, structures, and controls have been treated individually rather than in combination. However, in reality, the various disciplines are always present together and interact with each other, often in complicated nonlinear ways. For example, the modeling of aeroelastic behavior requires accounting for interactions between disciplines, e.g., aerodynamics and structures. A numerical simulation thus requires coupling between computational fluid dynamics (CFD) and computational structural mechanics (CSM). Many other couplings are possible when one considers the wide variety of vehicles and flight regimes encountered in modern aviation. Their numerical simulation has given rise to a new activity called "computational aeroscience" (CAS), the computational modeling of the interaction among disciplines and subsystems. Such interactions are becoming more important because of the high degree of system integration in modern aerospace vehicles and subsystems.

The rapid growth in CFD and CSM capabilities has been largely paced by two factors: (1) improvements in computational methods (principally numerical algorithms and physics models); and (2) increases in computer performance (principally computing rate and memory capacity). Figure 1 from Reference 1 shows that dramatic improvements in solution-time have resulted nearly equally from advances in computational methods and from improved computer technology. Note that in this example, computer technology improvements have increased by a factor of 1000 over 20 years, while computational methods have increased by a factor of 3000 over the same period, for an aggregate speed-up of more than one million. During the past two decades, the high computational demands of aerospace applications have been among the driving forces leading to evermore powerful supercomputers. Now, in the 1990s, it is apparent that conventional super-computers cannot sustain this high rate of advancement and it is becoming more and more apparent that future growth in computational speed will increasingly derive from parallel processing technology.

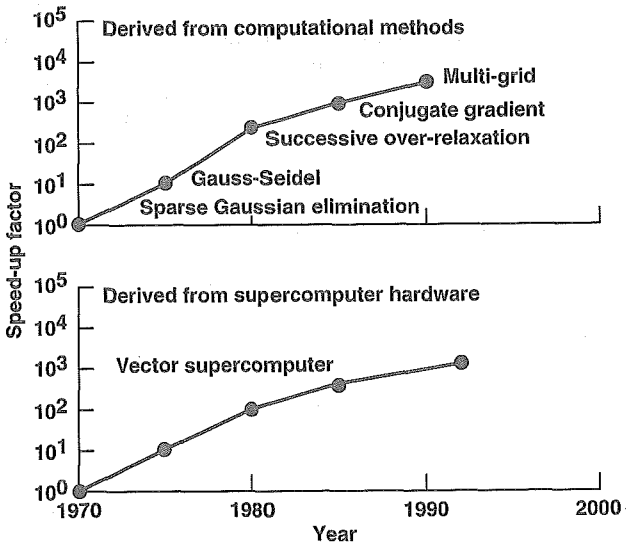


Figure 1. Performance improvement for scientific problems (Reference 1).

In the last several years a wide variety of parallel computers have become available for exploring the issues of using parallelism in scientific computing in general and CFD in particular. Most of the early computers that appeared between 1983 and 1987 were rather experimental in nature and served mainly for research investigations in areas such as algorithms, languages, and operating systems for parallel computing. In 1988 and 1989 several members of a first generation of parallel supercomputers became available. The term "supercomputer" is used here

because the Thinking Machines CM-2, the Intel iPSC/860, the NCUBE2, and other parallel computers are comparable (in their larger configurations) both in memory and peak computational speed to the performance of the most powerful conventional supercomputers, e.g., the Cray Y-MP. However, it is well known that these computers are still very deficient in their systems aspects, i.e., in their ability to handle a large number of users. Now, in 1992, we are at the threshold of a new generation of parallel supercomputers which offers considerable improvements in computational power over the previous generation as well as an improved software and user environment.

Recognition of the importance of parallel computing technology and computational methods to the advancement of science and engineering has fostered the United States government's creation of the High Performance Computing and Communications Program (HPCCP)[1-3]. HPCCP is a multi-agency effort with major participation by DARPA, NSF, DoE, and NASA and is aimed at accelerating the availability and utilization of the next generation of high performance computers and networks. An important goal of the program is to achieve a thousand-fold improvement in useful computing capability by 1996 and to enhance the range of scientific and engineering disciplines that can effectively exploit this computational capability. These goals will be realized by achieving a computational performance of 1 trillion floating-point operations per second (1 TFLOPS) on a wide range of important applications and with the development of associated system software, programming tools, and improved algorithms for a wide range of problems. The principal mechanism for accomplishing this increase will be based on improvements in hardware, systems software, and applications software utilizing massively parallel computing concepts.

HPCCP is driven by the recognition that unprecedented computational power and its creative use are needed to investigate and understand a range of scientific and engineering Grand Challenge problems. NASA has selected two Grand Challenge areas. The first area concerns the modeling and data analysis of earth and space physics phenomena. The second, which is our focus, is computational aeroscience (CAS), specifically defined as the integrated, multidisciplinary numerical simulation and design optimization of aerospace vehicle and propulsion systems throughout the flight envelope. NASA initiated the CAS Project within Ames, Langley and Lewis Research Centers to carry out the development of computational methods and computer technology necessary to advance CAS maturation. This paper briefly describes the goal, objectives, and approach of the project, then discusses the challenges confronting multidisciplinary modeling and the exploitation of parallel computers. (For a more in-depth treatment, see Holst et al. [4])

II. Computational Aeroscience Project

The overall goal of the CAS Project is to develop the necessary computational technology for the numerical simulation of complete aerospace vehicles for both design optimization and analysis throughout the flight envelope. It is recognized that accomplishing this goal requires advancing technology in numerical algorithms and computer programming, as well as computer hardware and software. Therefore, the goal is supported by four specific objectives that encompass the previously mentioned technologies:

(1) Develop advanced multidisciplinary computational models and methods that can effectively utilize massively parallel computers.

(2) Accelerate the development of advanced computing system hardware and software technologies capable of sustaining a TFLOPS performance level on multidisciplinary applications.

(3) Demonstrate and evaluate computational methods and computer system technologies for selected aerospace vehicle and propulsion systems models on scalable, parallel computing systems.

(4) Transfer the results of computational methods and computer system research to the aerospace and computer industries.

The CAS Project strategy for meeting these objectives consists of a coordinated research effort in algorithms and application codes, systems software and parallel testbed systems. Research in algorithms and application codes will focus on a selected number of problem areas, referred to as Grand Challenge applications. Code research will start on both conventional and existing massively parallel computers and will migrate to more advanced parallel computers as early in their development cycle as is practical. The advanced systems will serve as platforms for applications code research and for demonstration of well-defined pilot codes. Testbed system software from the manufacturer as well as that developed within the CAS Project and by HPCCP participants will be carefully evaluated to ensure an adequate software environment for efficient code execution. Efforts will focus on critical issues of the programming environment, compilers, and resource management. Testbed system hardware will also be evaluated by benchmark programs and the results evaluated. Results of evaluations will feed into improved and more advanced testbeds.

III. Grand Challenge Applications

Four Grand Challenge computational problem areas have been chosen for the CAS Project. These include two primary and two secondary Grand Challenges. The primary Grand Challenges are the High Speed Civil Transport (HSCT) effort and the High Performance Aircraft (HPA) effort. The two secondary Grand Challenges are the NASP-Derived Vehicle (NDV) effort and the Aerobraking effort. The four are associated with NASA efforts with supersonic transportation, high performance aircraft, air-breathing hypersonic flight, and atmospheric re-entry. They represent the entire speed regime from incompressible to hypersonic and extend into the high-altitude noncontinuum regime. In addition, they offer a broad range of important multidiscipline couplings. Finally, exploratory work has been performed with parallel computational methods in these problem areas to demonstrate feasibility of efficient parallel implementation.

The HSCT effort involves integrating the disciplines of aerodynamics, structural dynamics, combustion chemistry, and controls into a series of computational simulations of civilian supersonic cruise aircraft and its propulsion system. Emphasis within the airframe portion of this program will generally be placed in landing simulation; transonic to supersonic cruise simulation; efficient coupling of the aerodynamic, propulsion, structures, and control disciplines; and efficient implementation of multidisciplinary design and optimization. The propulsion portion of the effort will focus on enabling propulsion technologies for an environmentally sound HSCT vehicle: low NOx combustors, low noise nozzles, and efficient inlets. Multidisciplinary calculations (including complex control systems), very high temperature structural liners, and complex kinetic-aerodynamic flow strategies will be used to study ways to minimize NOx emissions and thereby minimize atmospheric ozone depletion. Additional calculations will treat complex, multi-disciplinary physics associated with source noise reduction through use of mixing nozzles.

The HPA effort will integrate the disciplines of aerodynamics, thermal ground-plane effects, engine stability, and controls into a series of computational simulations about a powered lift aircraft undergoing a transition maneuver, i. e., the transition from hover to forward flight and a fighter aircraft undergoing a low-speed, high-g turn. (Additional secondary computations associated with hover in ground effect and transonic cruise will also be performed.) This area will focus on the efficient coupling of the aerodynamic, propulsion, and control disciplines for simulation of these unsteady maneuvers. The control system for the powered-lift transition maneuver will consist of coupled aerodynamic and jet-implemented reaction systems. The control system for the high-g turn will consist of conventional aerodynamic controls, advanced forebody devices and a thrust vectoring control system. The main purpose of the propulsion effort is to develop compression system stability models that can be used in a near real-time simulation of the entire propulsion system. The focus will be on simulating engine behavior operating with highly distorted in-flow, since the selected maneuvers distort the flow into the engine and may substantially reduce engine thrust or lead to compressor system surge or stall. Coupled to the internal aerodynamic simulation, the transient behavior of a complete high performance aircraft can be studied.

The NDV effort involves efficient coupling of the disciplines of aerothermodynamics; structures; finite-rate air and combustion chemistry; and controls to simulate an air-breathing hypersonic vehicle. A finite-rate air chemistry model will be required for the highest speed portions of the simulation. In addition, the propulsion module requires the use of a hydrogen-air combustion model. Structural loads and thermal analysis will play an important part of this overall simulation effort.

The Aerobrake effort involves the efficient coupling of aerothermodynamics and finite-rate chemistry (including radiation and structures) to simulate an aerobraking vehicle during atmospheric entry. In this effort, the disciplines of aerothermodynamics and structures will play important roles. A complex chemistry model including multiple temperature scales and radiation will be required. In addition, some of the simulations will be performed at very high altitudes such that a continuum flow assumption is not valid. Under this condition, a particle simulation technique with vastly different algorithm characteristics will be used. This will provide an extreme test for the versatility of the various parallel computers to be used in the CAS Project.

IV. Multidisciplinary Modeling

The analysis of aerospace vehicles and propulsion systems is determined by complex interactions of many disciplines. These interactions can be treated in various ways depending on several factors, e.g., the characteristics of the disciplines involved, the complexity of the interaction, the accuracy desired, and the computer resources required. The CAS Project will investigate several techniques for coupling discipline variables. This section describes three principal approaches to be studied: direct coupling, matrix sensitivity, and hierarchical modeling with zooming.

Direct Coupling

Direct coupling treats the action of one discipline on another by directly coupling either at boundary conditions or at the fundamental equation level. The approach is relatively straightforward in implementation and is used for both analysis and design optimization. The simulation of wing flutter is an example of the direct coupling approach. Flutter occurs when the dynamics of the wing structure and the air stream interact to create a violent and potentially destructive oscillation of the wing. Flutter simulation involves the following computational steps:

(1) Using an initial condition or undeflected geometry and a suitable time-accurate CFD model, compute the aerodynamic loads on the wing at time level *t*.

(2) Using these aerodynamic loads and a suitable CSM model, compute the structural deflections of the wing at time level *t*.

(3) Using the wing deflections and aerodynamic solution at time level *t* and the CFD model, compute the aerodynamic loads at time level *t+dt*.

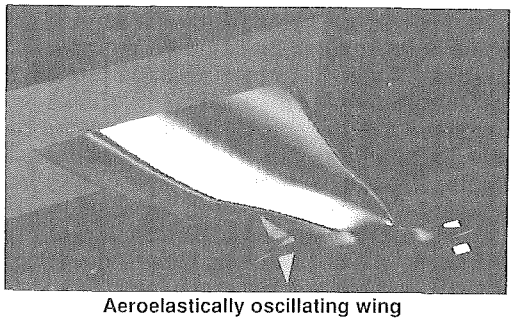
(4) Using the wing deflections at time level *t*, the aerodynamic loads at time level *t+dt* and the CSM model, compute new wing deflections at *t+dt*.

(5) Repeat steps (3) and (4) until convergence to a steady state or a periodic oscillation (flutter) is obtained.

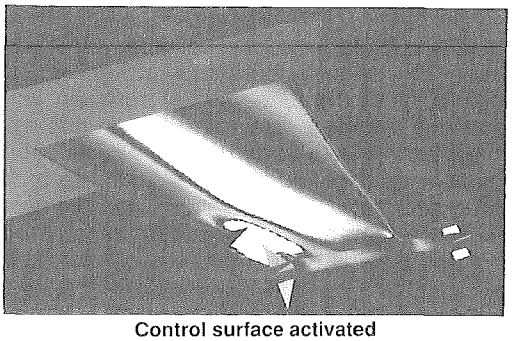
In each of these individual disciplines (i.e., aerodynamics or structural dynamics), only the surface conditions (i.e., the boundary conditions) from the other discipline are used. For the aerodynamic loads computation, only the shape of the wing surface is required. For the structural deflection computation, only the surface pressures are required. Controls could be added to this computation by simply adding a control surface coupled with an active control law integration scheme.

An example of simulation of wing flutter control is presented in Figure 2. In this work[5], the results were obtained by solving the modal structural equations of

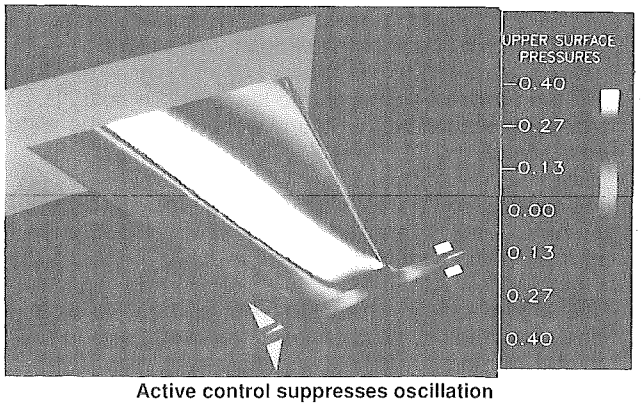
motion, a simple control law equation, and the full potential form of fluid dynamics equations, all in a fully coupled manner. Wing flutter was established with the control surface fixed (upper figure). Next, the control surface was activated (middle figure) so that its instantaneous deflections reduce and maintain the wing steady (bottom figure). The various shades of gray on the wing surface represent levels of instantaneous wing pressure.



Aeroelastically oscillating wing



Control surface activated



Active control suppresses oscillation

Figure 2. Computed pressures on a low aspect ratio wing with and without a control surface activated to suppress flutter. M_∞= 0.9, α = 0°, simulated altitude = 30,000 ft. (Guruswamy and Tu[5]).

Another example involves viscous dominated flow and concerns a numerical investigation of the tail buffet on the F-18 aircraft [6,7]. The F-18 leading-edge extension (LEX) generates a strong vortex which enhances wing lift at high angles of attack. The twin vertical tails are canted to intercept the high energy flow in the vortex to increase their effectiveness. However, at moderate-to-high angles of attack, the LEX vortex bursts and starts buffeting the tail. In the 20° - 30° range, the buffet is severe enough to cause premature structural fatigue.

As an initial step in the study of the tail buffet, the authors simplified the problem by assuming a weak coupling between the aerodynamics and structures. The solution was obtained in two parts. In the first part, a flow field was obtained for a rigid tail by solving the Reynolds-averaged Navier-Stokes (RANS) equations using a time-accurate, implicit procedure and a multizone grid. In the second part, the tail was assumed to be flexible and its aeroelastic deformation was computed using a finite element representation and the unsteady airloads as a forcing function. Figure 3 shows, at one instant in time, computed particle traces and limiting surface streamlines on the LEX, wing, and deflected leading edge flaps at a Mach number of 0.243 and angle of attack of 30.3°[6]. The computed results were qualitatively confirmed by flight test and indicate that there is unsteady LEX vortex breakdown in the vicinity of the tail. The unsteady flow field of the burst vortex generates an unsteady loading on the vertical tail. Figure 4 shows the calculated displacements of the tail

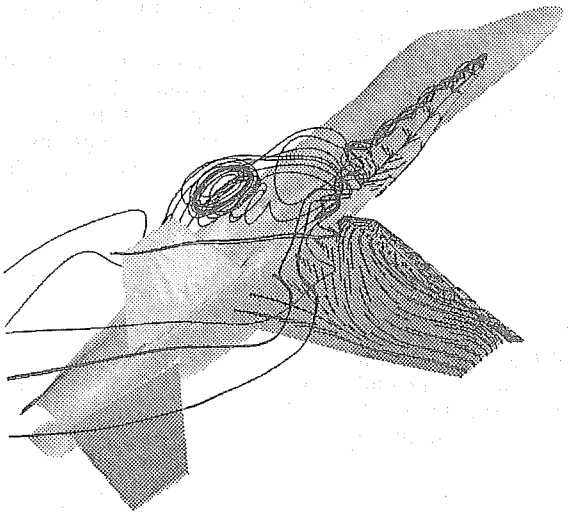


Figure 3. Computed particle traces showing LEX vortex breakdown and forward surface flow pattern, F-18 aircraft: M_∞= 0.243, α = 30.3°, Re_L = 11x10^6 (Rizk et al.[6]).

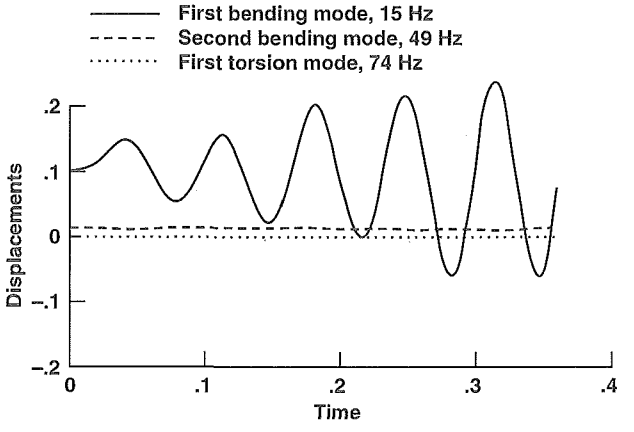


Figure 4. Vertical tail generalized displacements, F-18 aircraft: M_∞ = 0.243, α = 30.3°, Re_L = 11x10^6 (Rizk et al[7]).

due to the unsteady airloads[7]. Figure 5 shows good comparison between the predicted airload frequency and the frequency measured by sub-scale and full-scale wind tunnel tests and flight test at angles of attack of about 30°[7].

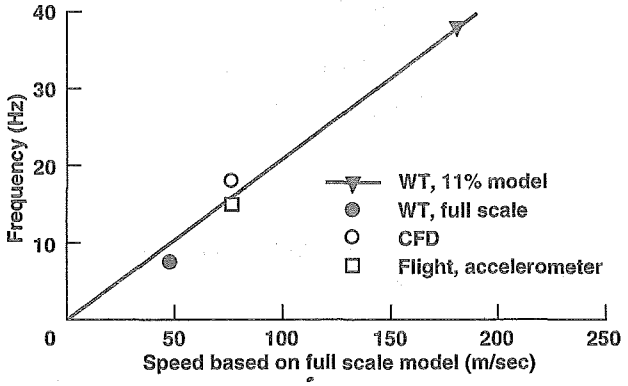
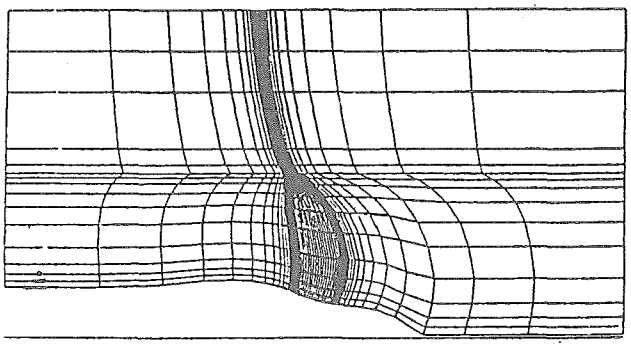
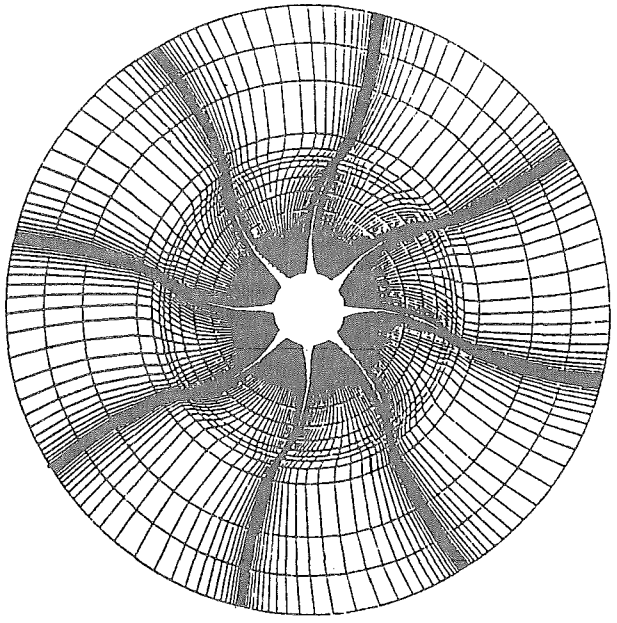


Figure 5. Comparison between computed and measured airloads frequency for F-18 aircraft vertical tail (Rizk et al.[7]).



(a) H-grid in the streamwise plane



(b) O-grid in the azimuthal plane

Figure 6. Structured mesh used in the aerodynamic calculations of a propfan (Srivastava et al.[6]).

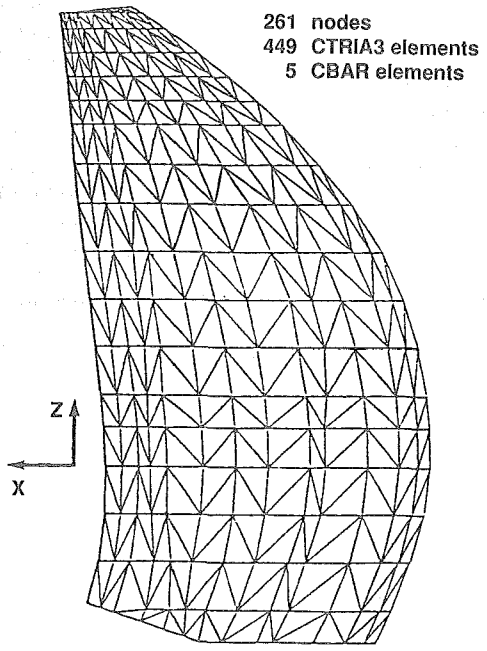
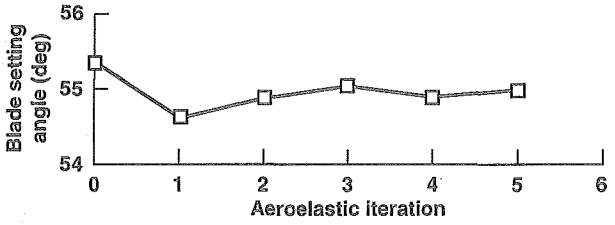
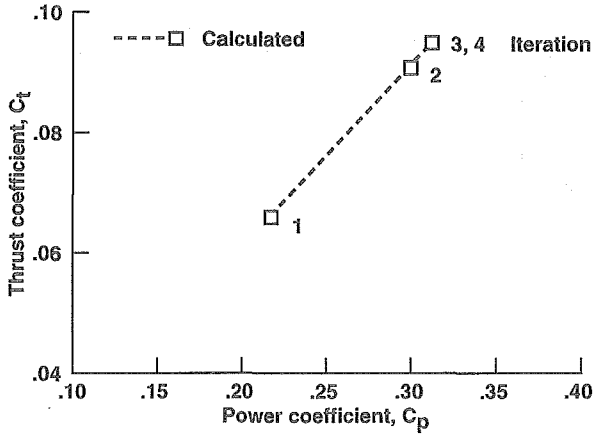


Figure 7. Finite-element unstructured grid for the structural analysis of a propfan blade (Srivastava et al.[6]).



(a) Blade setting angle at 75% span versus aeroelastic iteration



(b) Calculated thrust coefficient versus power coefficient for each aeroelastic iteration

Figure 8. Convergence history as a function of aeroelastic iteration for a static aeroelastic propfan computation (Srivastava et al.[6]).

Aeroelastic effects are also important to propulsion system performance. This is especially so in ultra-high bypass fan engines that use long, thin, highly twisted blades. Long, thin blades combined with high twist and aerodynamic loading can lead to large amplitude blade deformation. To study this problem, Strivastava et al.[8] performed a numerical simulation of a high bypass propfan by iteratively calculating the aerodynamic and structural elements of the blade. Aerodynamic loads were calculated from an Euler flow model using the structured grid shown in Figure 6. The aerodynamic loads were then used as inputs to a NASTRAN structural analysis using the unstructured grid shown in Figure 7. Spline interpolation of the aerodynamic data were used to input loads at the centroids of the structural elements. Using the new loads, NASTRAN provided an updated blade shape. The aerodynamic loads were then recalculated and the process was repeated until convergence. The convergence history of blade angle and of thrust coefficient versus power coefficients as a function of aeroelastic iteration is shown in Figure 8. The convergence rate is quite rapid and results in a final converged shape after only four iterations.

Sensitivity Matrix

The sensitivity matrix approach[9] involves the calculation of a matrix of sensitivity coefficients where the coefficients are the derivatives of the system response of interest (e.g., lift or drag) taken with respect to the design variables of interest (e.g., wing sweep or twist). For the designer, an accurate knowledge of the sensitivity derivatives of a particular system under consideration can subsequently be exploited in potentially useful ways. For example, the sensitivity can be used in approximate analysis, where if changes in a system's response is small, resulting changes in a system's response can be accurately estimated, resulting in a significant saving in computational costs. In addition, one of the most important applications of sensitivity derivatives is in design optimization and is one of the focused efforts within the CAS Project.

In general, sensitivity analysis can be classified into two categories: hierarchical and nonhierarchical. Hierarchical methods follow conventional approaches for each discipline without any communication with other disciplines except at the root level (see Figure 1 of Reference 9). This top-down procedure is not applicable for systems with lateral links between disciplines. Most modern aircraft involve configurations where cross coupling is inevitable. In such cases, it becomes necessary to use the nonhierarchical approach.

The nonhierarchical approach is illustrated in Figure 9 for sensitivity analysis of a wing with control surface. The wing may be considered as a system whose output consists of the data on aerodynamic pressures, structural deformations and active control surface deflections. Suppose that the objective is to reduce the wing-root bending stress using active controls. Aerodynamic loads are input into structural analysis, which outputs elastic deformations and corresponding stresses. Using sensitivity analysis, a signal based on the wing-root stress can then be transmitted to deflect the control surface in such a way that the wing-root stresses are reduced. During this process, several other

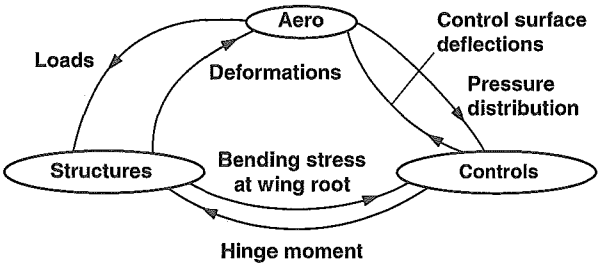


Figure 9. Graph representation of actively controlled, flexible wing as an example of a coupled system (from Reference 9).

items of information need to be computed, e.g., how to move the control surface whose deflections are coupled with both aerodynamics and structures.

Mathematically, the above process can be expressed as a set of simultaneous equations, termed "sensitivity matrix equations":

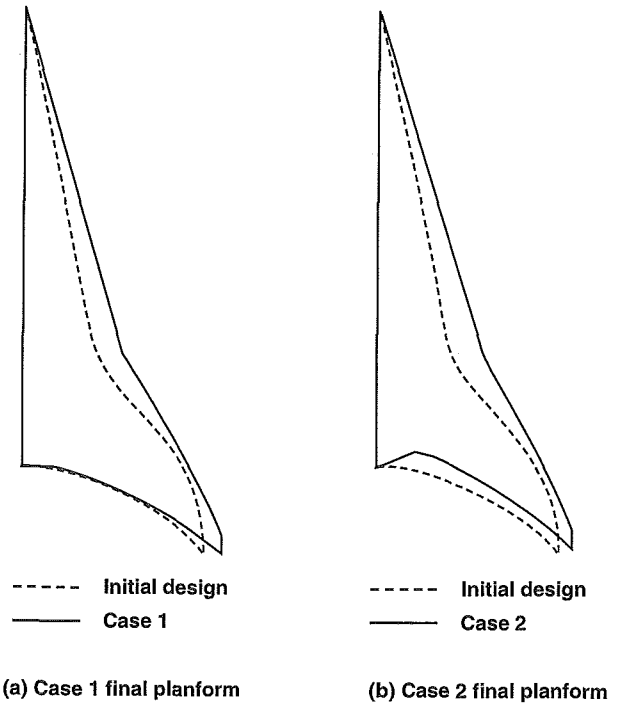
$$[C]\{G\} = \{P\}$$

In this equation, $[C]$ is a matrix of partial derivatives containing local sensitivity coefficients that relate two directly coupled disciplines. An entry in this matrix is a quantity such as the rate of change of wing-root moment with the control surface deflection. $\{G\}$ is a vector of total derivatives containing global sensitivity coefficients that relate changes in discipline parameters with respect to design variables. A typical entry of $\{G\}$ is the rate of change in the hinge moment with respect to the wing-root stress, where the wing-root stress is a design variable. Finally, $\{P\}$ is a vector of partial derivatives representing local changes of quantities within a discipline with respect to design variables. The success of this type of analysis depends on accurate and efficient computation of $[C]$ for nonlinear systems using the direct coupling method explained in the previous section.

Computational cost constraints usually dictate the sophistication of the single-discipline modes used in multidiscipline optimization. Following Hutchinson et al.[10], modeling can be defined at three levels: conceptual, preliminary and detailed design. As the design process progresses through these levels, the computational models become increasingly more sophisticated. Typically, conceptual design models are empirically based and expressed algebraically. Preliminary design models are usually linear approximations such as aerodynamic panel methods or structural plate models. Finally, detailed design models employ state-of-the-art computational methods such as RANS and finite element models (FEM).

Computational constraints currently limit multi-disciplinary optimization to the conceptual or preliminary design model stage. Hutchinson et al.[10] employed a variable-complexity modeling approach that combines conceptual and preliminary design techniques for wing optimization of the HSCT wing. Conceptual design-level algebraic equations were used to estimate aircraft weight, supersonic wave drag, friction drag, and drag due to lift. The drag due to lift and wave drag were also evaluated using more detailed, preliminary design-level

techniques. The methodology was applied to the minimization of take-off gross weight of a Mach 3.0 configuration with a range of 6500 miles. Figure 10 shows the comparison of initial and final wing planform for two cases. Case 1 employed a trailing-edge geometric constraint and Case 2 did not. Convergence was obtained in 20 cycles and resulted in initial and final design lift-to-drag ratios shown in Figure 11.



(a) Case 1 final planform (b) Case 2 final planform

Figure 10. Comparison of initial and optimized HSCT wing planforms, M_∞ = 3.0 (Hutchison et al.[10]).

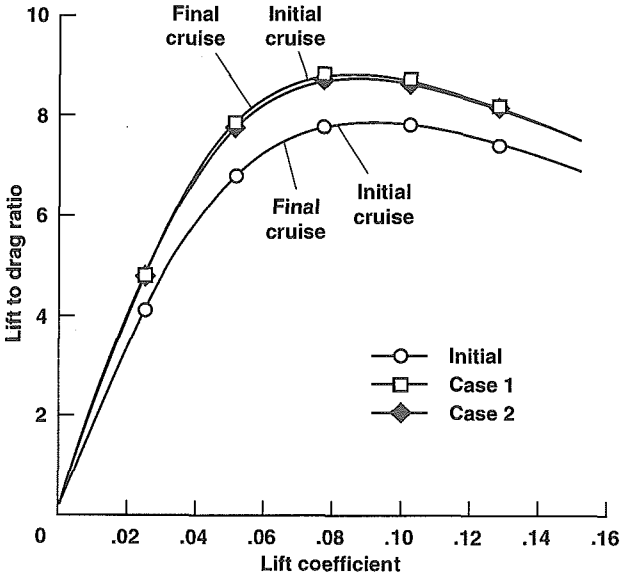


Figure 11. Lift to drag ratios for HSCT optimization, M_∞ = 3.0 (Hutchison et al.[10]).

In contrast to the above example, a detailed, design-level optimization would employ a RANS model for the aerodynamic analysis and a FEM model for the structural analysis. Figure 12 presents results of an RANS calculation in the form of a comparison of calculated and experimental lift and drag coefficients versus angle of attack for the HSCT wind tunnel model shown in Figure 13. The RANS solution was obtained from an explicit multigrid Runge-Kutta code developed at NASA Langley Research Center [11]. A singe zone grid of 688,00 points was used and the solution required 37 megawords of memory and 3.5 hours of Cray-2 time.

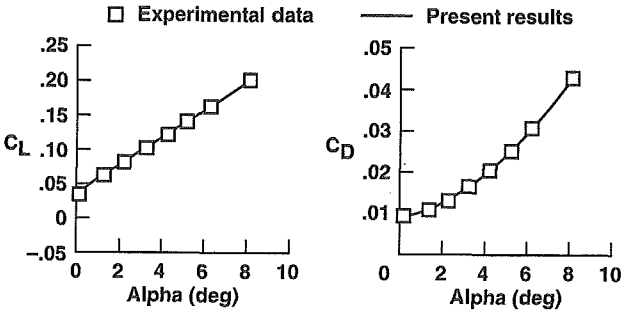


Figure 12. Comparison of force coefficients for an HSCT configuration, M_∞= 3.0 and Re_L = 6.3x10^6 (Vatsa et al.[11]).

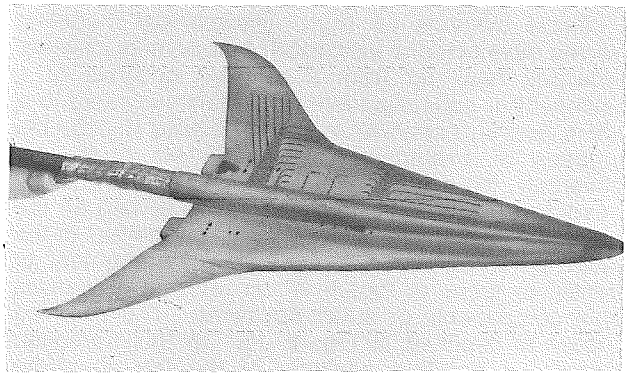


Figure 13. Photograph of the HSCT configuration used to obtain the results of Figure 10 (Vatsa et al.[11]).

A typical FEM of a HSCT configuration is shown in Figure 14 and was used to study the linear static response for various wing tip loading[12]. The linear system for this problem had 16,146 equations with a maximum semi-bandwidth of 594. Several hundred seconds are required to form and factor the stiffness matrix on a single processor of a Cray Y-MP computer. About six additional seconds are required to calculate the displacement from given loads. As is evident from this example, a typical linear, static structural mechanics problems is several orders of magnitude less computationally intensive than a typical steady, viscous CFD solution.

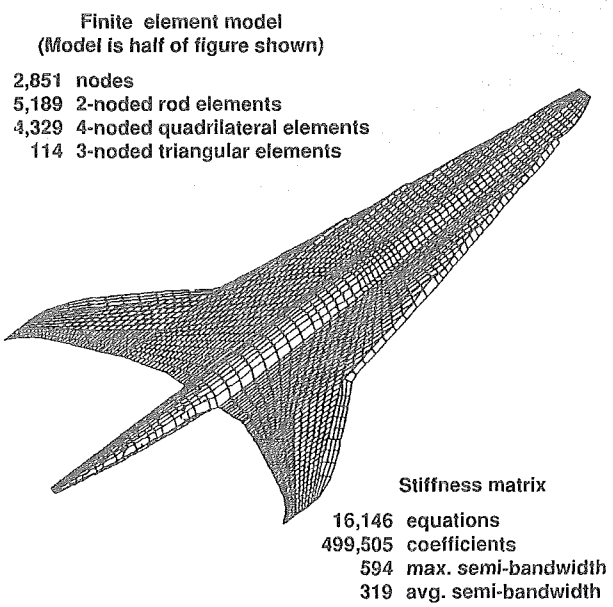


Figure 14. Finite-element model of an HSCT aircraft configuration (from Poole and Overman[12]).

Hierarchical Modeling with Zooming

Hierarchical modeling with zooming is an attempt to study the attributes of a complex system while minimizing computational cost. The concept has been developed for propulsion systems[13,14] but is applicable to vehicle systems as well. Hierarchical modeling involves developing a framework that permits physical processes resolved from a detailed analysis of a component or subcomponent to be communicated to a system analysis performed at a less detailed level for the purposes of evaluating overall system attributes. Conversely, the system analysis will provide the ability to evaluate which physical processes occurring on the component and subcomponent level are important to the system performance. This will allow the engineer or scientist to focus or "zoom in" on relevant processes within components or subcomponents. The zooming concept is depicted in Figure 15. In this particular example, a detailed analysis of the fan could be performed to study, for example, the effect of a new blade design on system performance. The inlet and compressor would be modeled at slightly lower levels of fidelity to resolve such phenomenon as inlet distortion or upstream influence of the compressor blading. The combustor, turbine, and nozzle would be modeled in less detail, perhaps to determine shaft horsepower.

The hierarchical model envisioned for propulsion simulation is characterized below:

Level 1: Engine system performance model. This model is a thermodynamic model which calculates the system efficiency based upon engine configuration and component efficiencies. It allows rapid evaluation of various engine concepts.

Level 2: Engine system dynamics and controls model. This model is a one-dimensional flow path model, with simplified structural elements, controls, and other

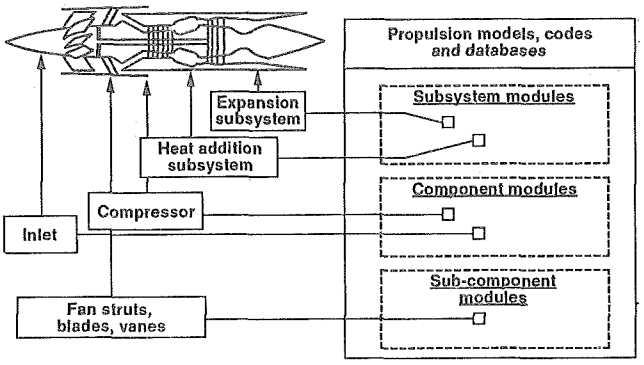


Figure 15. Illustrative framework for "zooming in" on a fan component (from Claus et al.[14]).

disciplines. It uses component performance information, design geometry information, and dynamic information in order to calculate engine thrust and weight as well as system transient response in order to analyze operability problems and devise control strategies to handle them.

Level 3: Space and/or time-averaged engine system model. This model is a two-dimensional (i.e., axisymmetric) fluid model. It utilizes axisymmetric multiple discipline models in an engine system environment in order to relate component boundary conditions (primarily input/output conditions) to overall system boundary conditions in order to simulate component interactions. The output is also the basic level about which the "zooming" process is constructed. It will be a transient model and address all problems from Level 2 in addition to providing more detailed geometry information.

Level 4: Space and/or time-averaged subsystem (or component) models. These models are three dimensional. They are multidiscipline models which are coupled in ways which are compatible with the physics of the component model but are still averaged over smaller time and space scales. These models must also be post-processed in order to connect with the Level 3 engine system model in the "zooming" process.

Level 5: Three-dimensional, time-accurate component models. This level of simulation consists of a fully three dimensional, time-accurate simulation of all physical processes on a component-by-component basis. This is the most complete level of physical approximation, representing the most complex physics, and therefore, it is the most computationally expensive level of simulation.

A typical propulsion system simulation uses a lower level analysis (such as Level 2) as a baseline "complete" engine simulation for which higher resolution, "zoomed" calculations are performed. The baseline analysis uses component performance maps to provide performance data over a wide range of operating conditions for both steady-state and transient simulations. "Zooming" then provides more accurate representations of these performance maps by conducting higher level simulations of the components.

Figure 16 presents a multiblock grid used for an axisymmetric representation of a modern turbofan

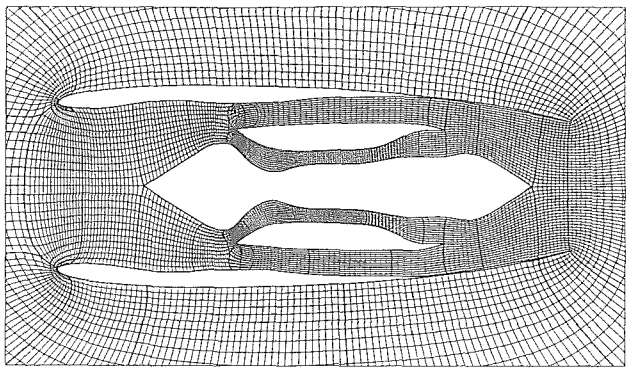


Figure 16. Multiblock grid for an axisymmetric representation of a complete engine (Stewart[15]).

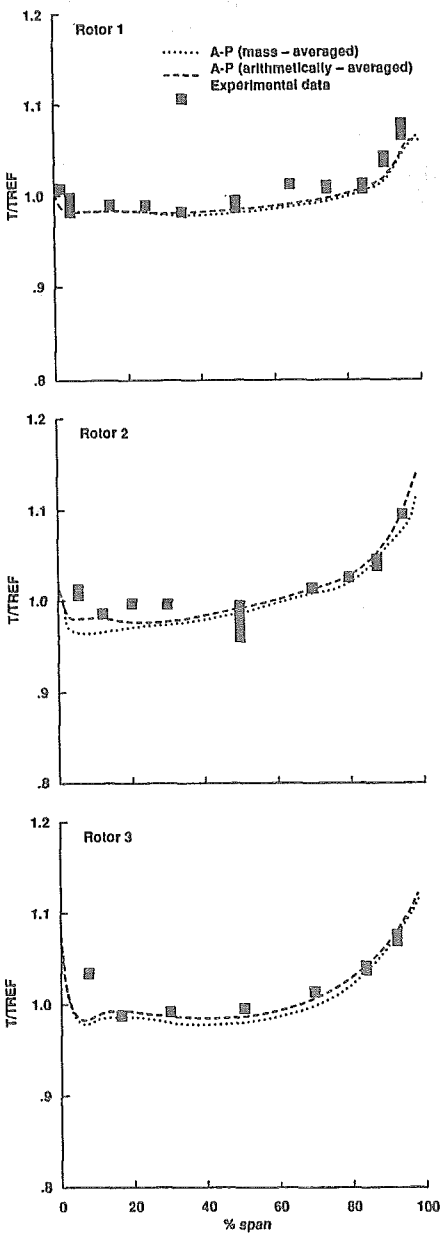


Figure 17. Comparison of computed and experimental time-averaged total temperature profiles for a three-dimensional turbomachinery computation (Mulac and Adamczyk[16]).

configuration[15] that provides the framework for a Level 3 simulation. A Level 4 example is provided by results of a three-dimensional, average-passage flow model (MSTAGE) which was used to obtain a time-averaged representation of the flow through the first four stages of an axial flow compressor[16]. Figure 17 shows good agreement between the spanwise computed and experimental total temperature profiles at the first three rotor exits. The computed results are obtained by both mass averaging and simple arithmetic averaging. The calculation used approximately 374,000 grid points and required 122 million words of memory. Two hours were required on a multitasked, eight-processor Cray Y-MP computer to reach convergence.

V. Computational Requirements

Computer speed and memory requirements for multidiscipline models depends first on the single discipline models selected and secondly on the multidisciplinary approach taken. CFD is present in nearly all CAS applications of interest and is usually the most demanding discipline for computer speed and memory. Holst et al.[4] recently estimated the computational requirements for both CFD and multidisciplinary applications. They examined a number of state-of-the-art, three-dimensional applications in which the RANS equations were solved for a steady, viscous flow using a simple algebraic turbulence model. The grids were coarse to save computer resources but adequate for attached flow. A baseline requirement for computer speed and memory was estimated from this data. The baseline computational requirements were set at 600,000 grid points, 25 megawords of run-time memory, and 10 hours of CPU time on a single processor of a Cray Y-MP computer. In addition, estimates were made for incremental increases in computer speed and memory as simulations became more complex. Factors that determined these increments included grid refinement, gas model, turbulence model, and the complexity of the flow physics, e.g., unsteadiness, flow separation, shock-wave/jet/wake/vortex interactions, etc.

Holst et al.[4] also estimated the increments in computer speed and memory needed to extend aerodynamic simulations to multidiscipline simulations. In the process, they examined several complication factors including:

(1) Computational requirements for several disciplines (aerodynamics, structural dynamics, controls, and chemistry/combustion) associated with both the airframe and propulsion elements of an aircraft must be accommodated in a single computation.

(2) Interface algorithms between the various disciplines will complicate the simulation and effectively add new computational requirements.

(3) Introduction of new physics as a result of inter-discipline interactions will create new computational requirements, e.g., flow unsteadiness caused by flexible structures or unsteady control surface deflection.

(4) Numerical optimization design applications will create significantly larger computational requirements.

Figure 18 presents a representative envelope of computer speed requirements (in floating-point operations-per-second) needed to achieve a solution within one hour (derived from the estimates of Holst et al.[4]). The left-most bar represents the estimated range of computer speed for an aerodynamics simulation alone. The bottom of the bar represents the computer speed needed for the baseline aerodynamics simulation described above. The top of the bar represents the estimated computer speed required for the extension of the baseline simulation to one that combines improved pressure and skin friction accuracy, unsteady flow, complex shock-induced boundary layer separation and Reynolds stress turbulence model. For example, such a simulation can be associated with an extreme maneuver condition at a performance boundary, but is, of course, well beyond current computer capability. The next bar to the right represents the estimated computer speed required when a finite element method structural analysis is coupled to the aerodynamic model. Successive bars to right represent, in sequence, the addition of thermal analysis; addition of control laws and deflected control surfaces; addition of thrust vectoring and jet controls; and addition of such external propulsion effects as inlets and jet plumes. Thus, Figure 18 represents an estimate of the envelop of computer speed required for a representative set of multidisciplinary models. As one goes up in each bar, the computer requirement increases due to increased aerodynamic requirements; as one goes to the right, the computer requirement increases due to the addition of disciplines. It should be noted that this is only a representative estimate and does not include all possible multidisciplinary models or all possible disciplines; real gas chemistry, for example, is not included. Computational requirements for detailed flow simulations of a complete propulsion system (e.g., turbojet engine) are not included. These requirements are difficult to estimate accurately but are generally equal to or greater than those for vehicles. Design optimization is also not included, but it can be expected to increase the computer speed required by a single analysis by a factor of 10-1000. The actual factor depends on the complexity of the simulation and the number of design perturbations selected. In general, however, one expects design optimizations to involve cruise conditions and therefore use less complex aerodynamics models.

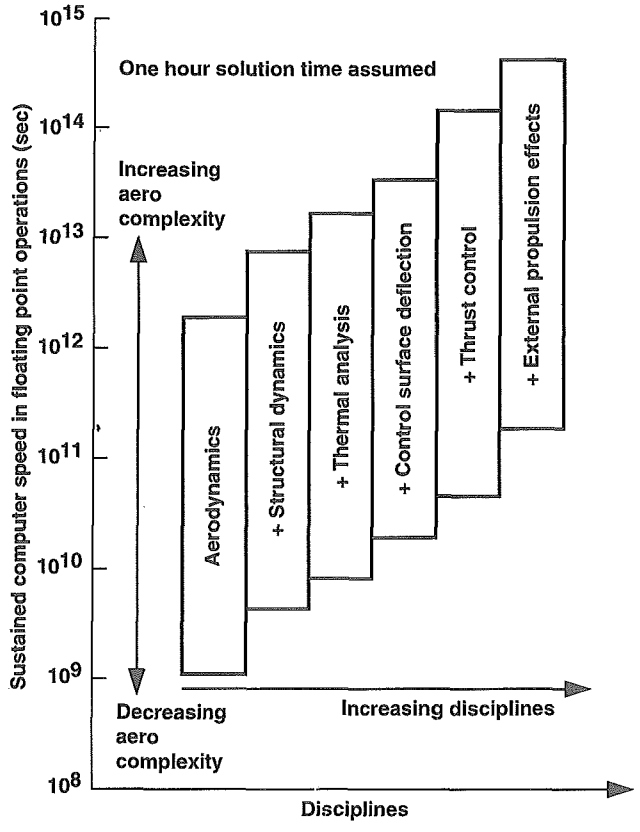


Figure 18. Computer speed estimates for representative multidisciplinary models.

The envelope of computer speed requirements spans five orders of magnitude. The lower bound of 10^9 floating-point operations-per-second (1 GFLOPS) is within the capability of today's supercomputers. However, the upper bound is well beyond any computer system planned. The CAS Project goal of 10^{12} FLOPS (1 TFLOPS) lies approximately midway between these extremes. Although the speed required for the most complex multidiscipline simulations lie beyond 1 TFLOPS, many important applications will be possible. With further improvements in algorithms the range of possible applications will expand even more.

VI. Parallel Computing

This section briefly explores parallel computers as a means of meeting the CAS Project goal of a sustained computing rate of 1 TFLOPS. This is indeed an ambitious goal as it represents 1000 times the current achievable sustained rate for CFD applications on current supercomputers such as the Cray Y-MP. It is generally accepted that sequential processing computer designs are not sufficient to meet the high-performance computing demands of the Grand Challenges. While sequential processing has the advantage of achieving near peak performance for most applications of interest, increased processing rates depend entirely on advances in circuit technology to decrease cycle time. For example, to reach 1 TFLOPS using a single sequential processor requires a cycle time of less than 3×10^{-13} seconds (assuming 3 cycles per floating-point operation). However, the current state-of-the-art in advanced gallium arsenide and silicon devices limits cycle times to about 2×10^{-9} seconds[17]. It is evident that some as yet undiscovered technology would be needed for a sequential computer to reach 1 TFLOPS.

New computer architectures have been invented to overcome circuit technology limitations. Vector computers were introduced in the mid-1970s and employed pipelining to achieve a floating-point result every cycle. These computers have produced significant performance increases for applications that have a high vector content. A more revolutionary step was the introduction of parallel processing employing the concurrent operation of multiple microprocessors. Hennessy and Jouppi[18] estimate that the CPU performance of microprocessors has improved by a rate of 1.5 to 2 times each year during the last six to seven years, whereas improvement rates for mainframes averaged about 25% per year. They quote two major

factors which have contributed to the high growth rate of microprocessor performance:

(1) The dramatic increase in the number of transistors available on a chip.

(2) Architectural advances, including the use of Reduced Instruction Set Computer (RISC) ideas, pipelining, and caches.

In Figure 19 these trends are summarized by comparing the performance of microprocessors to that of single processors on vector supercomputers of the last decade. The single processor performance of vector supercomputers within the last decade has only improved by approximately an order of magnitude[19]. In contrast, around 1987, microprocessors experienced a dramatic increase in their floating-point capabilities. If we restrict ourselves to the Intel family of microprocessors, we see an improvement from the Intel 80387 to the Intel i860 that is approximately two orders of magnitude greater. This large performance gain can be directly correlated to the performance of the corresponding parallel systems. The current Intel iPSC/860, which uses the i860 processor, is widely regarded as the first true supercomputer produced by Intel, whereas its predecessor, the iPSC/2, was at best an experimental research computer.

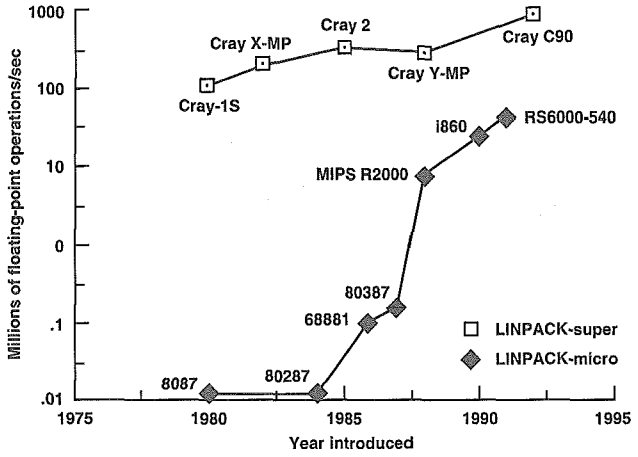


Figure 19. Comparison of supercomputer and microprocessor floating-point performance (from Reference 19).

Estimates and studies investigating future trends in microprocessor performance are highly optimistic. Gelsinger et al.[20] estimate that by the year 2000 a single chip is expected to have four processors with a combined performance of about 1 GFLOPS. Thus, the rate of microprocessor performance increases is expected to remain higher than that of traditional supercomputers. Even though these arguments do not address bandwidth-to-memory ratio, memory size, I/O devices, and other important aspects of supercomputer systems, it is believed that microprocessors will be the pacing item in the future development of massively parallel systems. (See Lundstrom[21] for a more detailed discussion.)

Massively parallel computers with enough performance to tackle realistic applications have existed only in the past few years. However, one can find numerous results in the literature. (See Holst et al.[4]) An important issue in all parallel applications is their performance relative to conventional supercomputers. To address this issue, Bailey et al.[22] developed the NAS Parallel Benchmarks, a set of programs representative of principal computational and data movement requirements of modern CFD applications. The benchmarks represent a novel approach to benchmarking in that the problems are specified in a "pencil and paper" fashion, i.e., the complete details of problem to be solved are given in a technical document. Except for a few restrictions, benchmarkers are free to select the language constructs and implementation techniques best suited for a particular system. Three benchmarks cast as "simulated" implicit CFD applications are LU, SP and BT. Each was run on a single processor of the Cray Y-MP supercomputer and on the Intel iPSC/860 and Thinking Machines CM-2 parallel computers. The LU benchmark consists of a regular-sparse block (5 x 5) lower and upper triangular system. The SP benchmark consists of multiple independent systems of non-diagonally dominant, scalar, pentadiagonal equations. Finally, the BP benchmark consists of multiple independent systems of non-diagonally document (5 x 5) block tridiagonal equations.

Table 1 shows a comparison of the performance for the three benchmarks computed with a 64^3 grid and for a two-dimensional unstructured grid flow solver[20]. The results show that the NAS Parallel Benchmarks perform somewhat slower on the parallel computers than on the Cray Y-MP whereas the unstructured grid code performs somewhat faster. One can conclude that the current generation of parallel computers performs at a level that is approximately equal to that of a single Cray Y-MP processor and, thus, they are at the low-end of the supercomputer class.

Application	System	No. of Processors	Time/Iter. (sec)	MFLOPS
NAS Benchmark LU	YMP	1	1.73	246
	iPSC/860	64	3.05	139
		128	1.90	224
	CM-2	8K	5.23	82
		16K	3.40	125
		32K	2.29	186
NAS Benchmark SP	YMP	1	1.18	250
	iPSC/860	64	2.42	122
	CM-2	8K	9.75	30
		16K	5.26	56
		32K	2.70	109
NAS Benchmark BT	YMP	1	3.96	224
	iPSC/860	64	4.54	199
	CM-2	16K	16.64	54
		32K	9.57	94
Unstructured Grid CFD Code	YMP	1	0.39	150
	iPSC/860	64	0.31	188
		128	0.19	308
	CM-2	8K	0.43	136

Table 1. Performance comparison of selected NAS Benchmarks and unstructured grid CFD code on Cray Y-MP uniprocessor, iPSC/860, and CM-2.

The main issue in massively parallel computing is designing efficient algorithms that result in high, sustained performance. (See References 23-25 for more information on performance issues.) Simply stated, an ideal algorithm is designed to keep the processors fully occupied performing calculations with minimal overhead while at the same time minimizing the total number of calculations needed to solve the problem. Clearly, since increases in computing rate are gained by performing tasks in parallel, purely sequential operations must be kept to an absolute minimum. At the same time, care must be taken to avoid using highly inefficient algorithms simply because they are highly parallel. Shadid and Tuminaro[26] performed a comparison of explicit and implicit algorithm performance on an NCUBE-2 parallel computer that illustrates this point. Their results, shown in Table 2, give the operation count, CPU time, and computing rate for four different algorithms applied to a simple convection-diffusion problem. The purely explicit Jacobi method has a computing rate (1000 MFLOPS) which is nearly six times faster than the highly implicit multigrid scheme rate (318 MFLOPS). On the other hand, the multigrid scheme time-to-convergence (6.7 seconds) is more than 300 times faster than the Jacobi method time-to-convergence (2124 seconds).

algorithm	operation (billions)	cpu time (sec)	MFLOPS
Jacobi (explicit scheme)	3820	2124	1800
Gauss-Seidel	1210	885	1365
least squares	259	185	1400
multigrid (implicit-like scheme)	2.13	6.7	318

Table 2. Computational statistics for a CFD convection-diffusion problem implemented on an NCUBE-2 computer (from Shadid and Tuminaro[26]).

Granularity is an important factor to consider in developing efficient parallel algorithms. When dividing an application into parallel pieces, granularity describes the size of each piece. Let us assume an application is broken up into n independent tasks. A task may, for example, involve the calculation of velocity at a grid point in a fluid dynamics simulation. Mapping the application across all processors will on the average result in $G = n/N$ tasks assigned to each processor where N is the total number of processors. G is called grain size and may represent, for example, velocity calculations at a cluster of grid points. For any given processor and interprocessor communication characteristics, there is a minimum grain size, G_{min}, such that if $n > NG_{min}$, one will achieve high relative performance. In other words, grain size must be large enough so that there is sufficient work to be performed by each processor to offset the relative overhead associated with interprocessor communication. The number of "grains" should also be large enough so that all processors can be usefully employed. It is apparent, then, that massively parallel computers are only well suited to large problems. The more nodes there are in the computer system, the greater the required problem size. In fact, a linear performance increase with increasing nodes can be achieved only as long as the problem size also increases linearly with the number of nodes[24,27]. An example of linear performance increase is found in the direct solution of turbulence by means of solving the full Navier-Stokes equations with no averaging or turbulence modeling. Figure 20 shows the performance of such an approach for various grid sizes as a function of the number of processors[4]. These calculations, performed on the Intel iPSC/860 and Intel Delta parallel computers, indicate that performance continues to increase at a steady rate only as long as the number of grid points also increase.

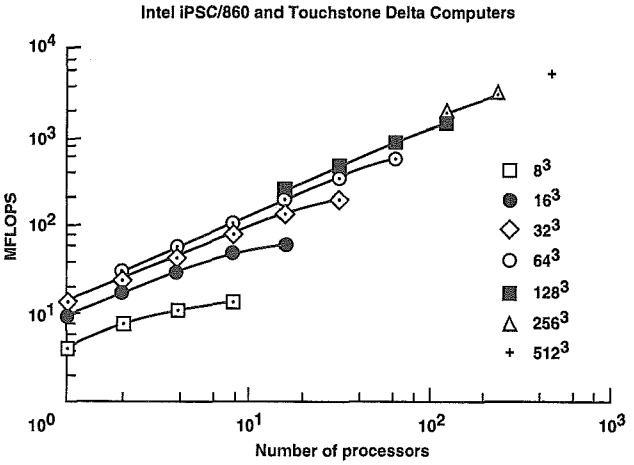


Figure 20. Execution speed vs. number of processors for a direct simulation computer code run on the Intel iPSC/860 and Intel Delta massively parallel computers (from Reference 4).

Partitioning refers to how an application is subdivided among processor nodes and is another important factor. Most mathematical models of the physical universe offer abundant opportunities for partitioning. Spatial partitioning occurs when strong interactions are localized within subdomains and when weak interaction occurs over long distances or not at all. In this widely-used strategy[28,29] a subdomain is assigned to each processor. Information is only occasionally exchanged between processors, resulting in low relative overhead. Functional parallelism is another decomposition strategy; it consists of decomposing the model into functional domains associated with different time or length scales. For example, functional domains can be assigned for each discipline within a multidisciplinary simulation model. To illustrate multidisciplinary partitioning, consider a powered-lift aircraft simulation, schematically represented in Figure 21[4]. The scheme could, for example, be used to simulate a vehicle in hover and to study its controllability. The simulation includes external aerodynamics (Navier-Stokes), a propulsion system that includes a reaction control system (RCS), and an aircraft control system that includes a routine for computing aircraft dynamics. A conceptual mapping of the simulation onto a parallel computer is shown in Figure 22. Each iteration would start with a communication step to update data to all processors. This would be followed by the parallel computation of fluid dynamics, structural dynamics, and propulsion system and control models. Each of these models would be mapped into processor modes via spatial partitioning in such a manner that idle time would be minimized. At the end of an iteration, a communication step could update

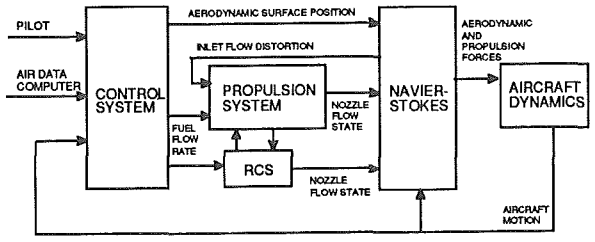


Figure 21. Schematic showing the multidiscipline interactions in a typical powered lift aircraft simulation (from Reference 4).

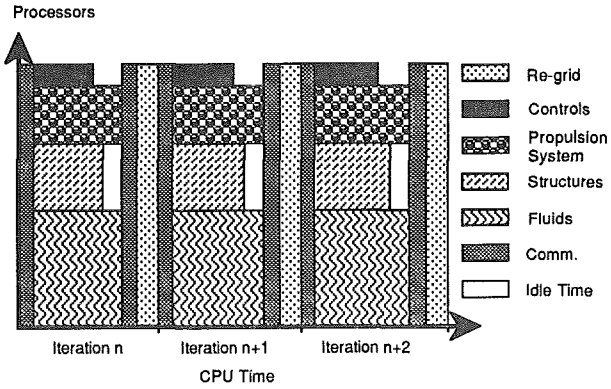


Figure 22. Schematic showing the theoretical mapping of a multidiscipline CAS problem to a functionally partitioned SIMD computer or a standard MIMD computer (from Reference 4).

information between models and a regridding step would then reflect changes in aircraft position and control surface positions. In summary, one can see that, with the increased complexity of parallelism, granularity, and partitioning, implementation of multidisciplinary models on parallel computers will be significantly more challenging than on conventional computers.

VII. Concluding Remarks

The CAS Project is a bold attempt to greatly enhance our ability to utilize computers to improve aerospace design quality and to decrease design time. Achieving this is indeed a challenge. Meeting this challenge requires the development of new multidisciplinary simulation technology and new powerful parallel computers to perform the simulations. Examples presented herein demonstrate that efforts have been underway in multidisciplinary simulation for the past several years. Results are encouraging, but progress has been hampered by the lack of computer power. Exploiting parallel computer technology is the strategy taken to overcome this limitation. The results of benchmark studies presented herein show that parallel computers have reached supercomputer status. However, the algorithm and programing challenges to exploiting their full potential are formidable. The hope is that through the coordinated research efforts of the CAS Project these barriers can be overcome.

VIII. References

1. Federal Coordinating Council for Science, Engineering, and Technology Committee on Physical, Mathematical, and Engineering Sciences, Grand Challenges: High Performance Computing and Communications, c/o National Science Foundation, Computer and Information Science and Engineering Directorate, 1800 G Street, N.W., Washington, D.C. 20550, 1991.

2. Federal Coordinating Council for Science, Engineering, and Technology Committee, A Research and Development Strategy for High Performance Computing, Executive Office of the President, Office of Science and Technology Policy, Nov. 1987

3. Federal Coordinating Council for Science, Engineering, and Technology Committee on Physical, Mathematical, and Engineering Sciences, Grand Challenges 1993 : High Performance Computing and Communications, c/o National Science Foundation, Computer and Information Science and Engineering Directorate, 1800 G Street, N.W., Washington, D.C. 20550, 1992.

4. Holst, T. L., Salas, M. D., Claus, R. W., "The NASA Computational Aerosciences Program - Toward TeraFLOPS Computing," AIAA paper 92- 0558, Jan. 1992.

5. Guruswamy, G. P., and Tu, E. L., "Transonic Aeroelasticity of Fighter Wings with Active Control Surface," Journal of Aircraft, Vol. 26, 1989, pp. 682-684.

6. Rizk, Y. M. and Gee, K., "Numerical Prediction of the Unsteady Flowfield Around the F-18 Aircraft at Large Incidence," AIAA Paper No. 91-0020, Jan. 1991.

7. Rizk, Y. M., Guruswamy, G. P., and Gee, K., "Numerical Investigation of the Tail Buffet on the F-18 Aircraft," AIAA Paper No. 92-2673, June 1992.

8. Strivastava, R., Sankar, N. L., Reddy, T. S. R., and Huff, D. L., "Application of an Efficient Hybrid Scheme for Aeroelastic Analysis of Advanced Propellers," Journal of Propulsion and Power, Vol. 7, No. 5, Sept.-Oct. 1991, pp. 767-775.

9. Sobieszczanski-Sobieski, J., "Sensitivity Analysis and Multidisciplinary Optimization for Aircraft Design: Recent Advances and Results," Journal of Aircraft, Vol. 27, No. 12, Dec. 1990, pp. 993-1001.

10. Hutchison, M., Unger, E., Mason, W., Grossman, B., and Haftka, R., "Variable-Complexity Aerodynamic Optimization of an HSCT Wing Using Structural Wing-Weight Equations," AIAA Paper No. 92-0212, Jan. 1992.

11. Vatsa, V. N., Turkel, E., and Abolhassani, J. S., "Extension of Multigrid Methodology to Supersonic/Hypersonic 3-D Viscous Flows," Proceedings of the 5th Copper Mountain Conference on Multigrid Methods, Mar. 31-Apr. 5, 1991.

12. Poole, E. L. and Overman, A. L., "Parallel Variable-Band Choleski Solvers for Computational Structural Analysis Applications on Vector Multiprocessor Supercomputers," Computing Systems in Engineering, Vol. 2, No. 2/3, 1991, pp. 183-196.

13. Nichols, L. D. and Chamis, C. C., "Numerical Propulsion System Simulation: An Interdisciplinary Approach," AIAA Paper 91- 3554, Sept. 1991.

14. Claus, R. W., Evans, A. L., Lylte, J. K., and Nichols, L. D., "Numerical Propulsion System Simulation," Computing Systems in Engineering, Vol. 2., No. 4, 1991, pp. 357-364.

15. Stewart, M.E., "Euler Solutions for an Unbladed Jet Engine Configuration," NASA TM 105332, 1991.

16. Mulac, R. A. and Adamczyk, J. J., "The Numerical Simulation of a High-Speed Axial Flow Compressor," ASME paper No. 91-GT-272, 1991.

17. Amdahl, G. M., "Limits of Expectation," The International Journal of Supercomputer Applications, Vol. 2, No. 1, Spring 1988, pp. 88-97.

18. Hennessy, J. and Jouppi, N., "Computer Technology and Architecture: An evolving Interaction," IEEE Computer, Vol. 24, No. 9, Sept. 1991, pp. 18-29.

19. Bailey, F. R. and Simon, H., "Future Directions in Computing and CFD," AIAA paper 92-2734, June 1992.

20. Gelsinger, P. P., Gargini, G. H., and Yu, A. Y. C., "Microprocessors Circa 2000, " IEEE Spectrum, Vol. 26, No. 10, 1989, pp. 43-47.

21. Lundstrom, S. F., "Supercomputing Systems - A Projection to 2000," Computing Systems in Engineering, Vol. 1, Nos. 2-4, 1990, pp. 145-151.

22. Bailey, D., Barszcz, E., Barton, J., Browning, D., Carter, R., Dagum, L., Fatoohi, R., Frederickson, P., Lasinski, T., Schreiber, R., Simon, H., Venkatakrishnan, V., and Weeratunga, S. "The NAS Parallel Benchmarks," International Journal of Supercomputer Applications, Vol. 5, No. 3, 1991, pp. 63-73.

23. Worlton, J., "Toward a Science of Parallel Computation," Computational Mechanics - Advances and Trends, AMD Vol. 75, ASME, New York, 1987, pp. 23-35,

24. Fox, C. G., and Frey, A., "High Performance Parallel Supercomputing Application, Hardware, and Software Issues for a TeraFLOPS Computer," California Institute of Technology Report C3P-451B, May 1988.

25. Denning, Peter J., "Speeding Up Parallel Processing," American Scientist, Vol. 76, July- Aug. 1988, pp. 347-349.

26. Shadid, J. N. and Tuminaro, R.S., "Iterative Methods for Nonsymmetric Systems on MIMD Machines," to appear in Proceedings of the Fifth SIAM Conference on Parallel Processing for Scientific Computing, 1991.

27. Gustafson, J. L., Montry, G. R., and Benner, R. E., "Development of Parallel Methods for a 1024- Processor Hypercube," SIAM Journal of Scientific Computing, Vol. 9, July 1988.

28. Berger, J. J., and Bokhari, S., "A Partitioning Strategy for Non-uniform Problems on Microprocessors," NASA CR 178024, Nov. 1985.29. Reed, D. A., Adams, L. M., and Patricks, M. L., "Stencils and Problem Partitionings," NASA CR 178102, May 1986.