AIAA JOURNAL
Vol. 39, No. 8, August 2001

Aerodynamic Shape Optimization Using Computational Fluid Dynamics and Parallel Simulated Annealing Algorithms

X. Wang* and M. Damodaran†
Nanyang Technological University, Singapore 639798, Republic of Singapore

Aerodynamic shape design using stochastic optimization methods, such as simulated annealing method to optimize objective functions evaluated by modern state-of-the-art computational fluid dynamics solvers, normally requires enormous computation time to search for the global optimal design. Aerodynamic shape optimization of internal flow systems is studied using Euler/Navier–Stokes solvers and parallel simulated annealing algorithm, which is implemented on parallel computing platforms. A variety of inverse and direct design of internal flow systems are carried out to examine the efficiency and speedup of the parallel simulated annealing algorithms. The results demonstrate that parallel simulated annealing can be a feasible global optimizer for aerodynamic shape design resulting in considerable reductions in wall-clock time on multiple processors.

Introduction

AERODYNAMIC shape design in the context of multidisciplinary optimization design (MDO) has been extensively investigated in the recent past using computational fluid dynamics (CFD) as outlined by Sobieszczanski-Sobieski and Haftka.[1] In view of the enormous costs associated with computing the objective function using stochastic optimization methods, deterministic optimization methods based on the gradient-based methods and sensitivity analysis have been favored by many researchers, such as Frank et al.[2] However, whereas deterministic methods allow the generation of an improved design, these methods do not always lead to a global optimum and often restrict the design space to conventional designs. In the traditional deterministic gradient-based methods, the design is updated iteratively in the direction of the steepest descent from the initial design guess (the hill-climbing strategy). Many variations of gradient-based methods have been widely used because the optimum obtained might be a global optimum if the objective function and constraints are differentiable and convex. In practice, however, it is very difficult to prove these properties. Deterministic optimization methods are efficient in finding the minima of continuously differentiable problems for which sufficiently accurate derivatives can be estimated at reasonable cost. The smoothness requirement and the need for derivatives are clear disadvantages in addition to that these methods locate local, rather than global, minima. Deterministic methods also expends work in constructing sensitivity analysis models in design problems using CFD. Sensitivity analysis have been used in aerodynamic shape design extensively in recent years. Unlike applications in structures and control systems, sensitivity information cannot be easily extracted from CFD codes, and this poses the main obstacle for using gradient-based methods in developing tightly coupled solution methods for MDO.

One alternative to overcome the limitations of deterministic methods is to use stochastic optimization methods, such as simulated annealing (SA), which is a robust method for searching the global minima and is easy to implement. With the advent of advanced computer architectures, it is imperative that the associated computational costs and time in the application of stochastic algorithms can be reduced by using parallel computing. The present study addresses the issue of developing robust computational methods using parallel SA (PSA) methods for global optima in the design space in conjunction with CFD methods for Euler or Navier–Stokes equations, which are used to compute the objective functions. A variety of aerodynamic shape design problems such as inverse designs and direct design problems for high-speed internal flow systems, such as nozzles and diffusers, are studied to demonstrate the feasibility of using PSA as a feasible optimization method. The study also focuses on the implementation of the PSA algorithm on a parallel computer and presents the speedup and efficiency of implementing a problem-independent PSA method on a shared memory SGI Origin 2000 architecture of a multiprocessor computer and using message passing interface (MPI) library on the machine to parallelize the optimum design of selected internal aerodynamic flow systems using advanced CFD. Although the methods developed in this study focus on a single discipline setting, the effort in principle can be extended to scenarios involving complex MDO problems.

Received 11 March 2000; revision received 24 August 2000; presented as Paper 2000-4847 at the AIAA/NASA/USAF/ISSMO 8th Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, 6–8 September 2000; accepted for publication 22 December 2000.

*Research Fellow, School of Mechanical and Production Engineering and Centre for Advanced Numerical Engineering Simulations. Member AIAA.

†Associate Professor, Division of Thermal and Fluids Engineering, School of Mechanical and Production Engineering and Centre for Advanced Numerical Engineering Simulations; mdamodaran@ntu.edu.sg. Senior Member AIAA.

PSA Algorithms

SA is a heuristic strategy for obtaining near optimal solutions and derives its name from an analogy to the annealing of solids and is based on a mathematical model of the behavior of large collections of atoms or molecules under the influence of changes in temperature. The steps of the standard SA, that is, sequential SA, algorithm, as given in Appendix A, consists of a concise description of the system's configuration, design of a random generator of moves or rearrangements of elements in a configuration, a method for determining how good a particular configuration is at any given instant, and an annealing schedule specifying how quickly the system is quenched. During the process, the best rearrangements correspond to those that lower the energy of the system. The essential features of the basic SA method for optimizing complex engineering problems are outlined by Aarts and Korst[3] and Deb.[4] Several variants based on some appropriate modifications to the basic SA, such as Ingber's adaptive SA,[5] ensemble approaches of Ruppeeiner et al.,[6] and hybrid algorithms of Desai and Patil,[7] which combine SA and local search algorithms such as genetic algorithms (GA), also offer alternative improved algorithms for searching the global optimum economically and efficiently. The main disadvantage of SA is the enormous expenditure of computational resources for objective function evaluations for large-scale engineering design problems. In view of this feature, the current investigation addresses the issue of developing robust computational methods for using PSA in conjunction with CFD, which is used to evaluate the design objective functions for the test problems that are considered in this work. Gallego et al.[8] has shown that the parallel implementation of SA not only results in speedup, but also increases the chances of searching for the global optima. Bhandarkar and Machaka[9] have discussed many different parallel schemes of SA. A systolic method was

proposed by van Laarhoven and Aarts.[10] The speculative method of Witte and Franklin[11] has shown some promise for fine-grained multiple processors but not for coarse-gained computer systems. The divisions algorithm of Aarts and Korst[3] and the three parallelization strategies of Diekmann et al.[12] are also widely used. In view of the various features of these PSA, the divisions algorithm and cluster algorithm are selected for the present study. The divisions algorithm uses a control parallelism strategy and is implemented by dividing the effort of generating a Markov main chain among the available processors. The main chain, used in sequential SA, is divided into p subchains with the same length, where p is the number of processors. Each processor works on the subchain and continues the generation of the subsequent subchain. Before updating the temperature, a new subchain is obtained by choosing the best solution from the processors. This procedure is referred to in this study as PSA1. The clustered method functions efficiently at lower temperatures as mentioned by Bhandarkar and Machaka.[9] The PSA1 algorithm can be further enhanced by incorporating the clustered method in the lower temperature regions of the annealing process, and this is referred to as PSA2 in this study. The algorithmic details for implementing PSA1 and PSA2 are outlined in Appendix B. When the acceptance ratio R_t (defined as the ratio of the number of the accepted moves to the number of all of the search moves) is lower than a certain value (range of value chosen in this study is 0.4–0.6), the values of the objective function is gathered at the end of each search step, and the solution is chosen randomly from the accepted cost functions. The implementation of the two methods, that is, PSA1 and PSA2, for the design optimization of aerodynamic shapes is described in the subsequent sections. Recent tests of PSA (PSA2), parallel GA (PGA) and parallel genetic SA (PGSA) on the minimization of test functions done by the authors[13] show that on a coarse-grained parallel computer the PSA functions better than PGA and PGSA in reducing evaluations of objective function for each processor and wall-clock time.

Flow Analysis Models for CFD

Numerical methods solving the Euler and Navier–Stokes equations are used for computing the flowfield in which the computation to steady state is initiated from a set of initial conditions by a suitable time-marching algorithm. As a variety of one, two, and axisymmetric high-speed flow systems are of interest in this study, the numerical solution of one-dimensional and two-dimensional/axisymmetric compressible flow equations at high Mach numbers forms the basic analysis tool for the flowfield inside a given configuration. The one-dimensional Euler equations are solved by a quasi-one-dimensional flow solver, which takes the form

$$\frac{\partial[S(x)\boldsymbol{Q}]}{\partial t} + \frac{\partial \boldsymbol{E}}{\partial x} - \boldsymbol{H} = 0 \tag{1}$$

where $S(x)$ is the cross-sectional area and $\boldsymbol{Q}$, $\boldsymbol{E}$, and $\boldsymbol{H}$ are the flow variables, flux vectors, and source terms, respectively. The details of the various terms and the numerical scheme solving the equations are outlined by Hoffmann and Chiang.[14] Here Eq. (1) is numerically solved by the Steger–Warming upwind scheme [first-order total variation diminishing (TVD)] for supersonic flows.

The unsteady two-dimensional/axisymmetric Euler/Navier–Stokes equations in nondimensional form and computational space take the form

$$\frac{\partial \boldsymbol{Q}}{\partial \tau} + \frac{\partial(\boldsymbol{E} - \boldsymbol{E}_v)}{\partial \xi} + \frac{\partial(\boldsymbol{F} - \boldsymbol{F}_v)}{\partial \eta} + \alpha(\boldsymbol{H} - \boldsymbol{H}_v) = 0 \tag{2}$$

where the switch $\alpha = 0$ represents two-dimensional planar flow and $\alpha = 1$ represents axisymmetric flow. $\boldsymbol{E}_v$, $\boldsymbol{F}_v$, and $\boldsymbol{H}_v$ are the viscous flux whose exact forms are outlined by Hoffmann and Chiang.[14] If the viscous fluxes are zero, then the from reduces to the two-dimensional/axisymmetric Euler equations.

These equations are solved by the lower–upper symmetric Gauss–Seidel implicit scheme proposed by Yoon and Jameson,[15] and for enhancing the convergence and improving resolution, the scheme is extended using the TVD scheme of Yee and Harten.[16] The code has been verified for a number of benchmark test problems in Ref. 17, and a simple algebraic turbulence model such as the Baldwin–Lomax turbulence model is used in the present study for viscous flows. Improved turbulence models can be used for further investigation in the future.

Aerodynamic Shape Optimization of Internal Flow Systems

In this section, the application of PSA algorithms PSA1 and PSA2 and the CFD methods outlined earlier for the design of optimal shapes of nozzles and diffusers is outlined. The design test cases, which cover a wide range of flow dimensionality and CFD models, are described briefly first and then immediately followed by results and discussions based on the computational simulation and optimization. The computations have been performed on the parallel computing environment of the SGI Origin 2000 computer system. Comparative studies done by the authors[18] show that using the MPI library[19] is more efficient than using automatic parallelization compilers[20] for implementing the PSA1 and PSA2 on parallel computers.

Design Case 1: Diffuser Shape Design

This test case is concerned with optimizing the shape of an axisymmetric diffuser for which the design flowfield condition and the pressure distribution along the centerline are defined. The aim is to find the shape of the diffuser wall or cross-sectional area distribution along the flow direction that will satisfy the defined design flow condition. The objective function that has to be minimized is expressed in normalized form as follows:

$$F(\boldsymbol{X}) = \frac{1}{\rho_0 u_0^2} \int (P - P_t)^2 \, \mathrm{d}x \tag{3}$$

where P_t is the target pressure distribution, which is specified, P is the initial or evolving pressure distribution defined along the length of the diffuser centerline, ρ_0 is the density, u_0 and P_0 are inflow velocity and pressure, respectively, which are taken as reference values for scaling flow quantities in internal flow simulations using CFD analysis and $\boldsymbol{X}$ is the vector of design variables, that is, $\boldsymbol{X} = (x_1, x_2, \ldots, x_n)$. Here the task is to minimize the square of the difference between the target and design pressure. The Mach number of the supersonic flow entering the diffuser at the inflow boundary is 1.5. This problem is a representative design problem where the objective function is nonsmooth in view of the presence of a shock wave. To start the design process, the target pressure distribution and an initial shape of the diffuser must be specified. The target pressure is that defined for a diffuser that is generated by the following distribution of cross-sectional area $S(x)$ along the length of the diffuser: $S(x) = 1.398 + 0.349 \tanh(0.8x - 4)$. Assume that the local cross-sectional area of the diffuser is a circle, then the local radius of the diffuser cross section can be computed from the local value of the cross-sectional area. The initial shape of the diffuser is that of a conical frustrum obtained by connecting the radii at the inflow and outflow boundaries with a straight line. To initiate the design minimization process, the nozzle shape must be defined in terms of design variables. The curve defining the shape of the diffuser is parameterized by Bernstein basic functions as by Faux and Pratt[21]:

$$Y = \Sigma_{i=0}^{6} \frac{6!}{(6-i)!i!} u^i (1-u)^{6-i} \boldsymbol{x}_i \tag{4}$$

where Y is the radius of diffuser and $\boldsymbol{x}_i$ denotes the position vector and taken as the design variables in the optimization process. The range $0 \le u^i \le 1$, which can be defined as x/L, where L is the interval over which the Bernstein basic function is applied. In this example corresponding to a diffuser that is 10 ft long, the first and second Bernstein functions are $u^i = x/5$ and $u^i = (x-5)/5$, respectively. For $i = 0$–6, these two functions have a total of 14 variables, with fixed radii at inlet x_0 and outlet x_{14} and at the point where the two functions are blended, $x_6 = x_7$. Hence, the diffuser shape can be defined with the 2 functions and 11 design variables. For this case, the CFD analysis is based on the quasi-one-dimensional Euler equations for inviscid nonlinear compressible flow, that is, Eq. (1). The inverse design of the diffuser is carried out using PSA algorithms PSA1 and PSA2. The target pressure and the flowfield

corresponding to the diffuser shape defined by Eq. (4) is computed on 51 uniformly distributed points along the flow direction and for the specified design values and inflow conditions using the Euler equations. PSA is used to minimize the objective function until the desired shape of the diffuser is attained. By comparing the final shape of the designed diffuser with the shape corresponding to the target pressure distribution, it is possible to make assessments on the effectiveness of the PSA.

Results and Discussion of Design Case 1

A comparison between PSA1 and PSA2 is performed to assess their feasibility and effectiveness in the diffuser shape design. For this case, the various SA tuning parameters are chosen such that the initial cooling temperature is set to a value at which the initial cooling acceptance ratio is greater than 0.95 and the main length for the cooling scheme is first tested and taken to be as short as possible on a single processor. The length of the main chain is chosen to be $80N$, where N is the number of the design variables. A constant cooling scheme, that is, $T_{k+1} = \gamma T_k$, is prescribed with $\gamma = 0.1$. Based on reported work in the literature, such as Aly et al.,[22] and experience in applying SA for aerodynamic design optimization problems, the value of γ is chosen to vary between 0.30 and 0.05. The parameter R_t has a value ranging between 0.5 and 0.6. PSA1 and PSA2 are implemented using MPI library functions. Because many multiprocessors on the SGI machine are used for the implementation of the PSA1 and PSA2 algorithms, the termination criteria to stop the program is defined such that when the objective function reaches a value that is less than $F_{\min} = 1.4E-4$ based on the collected results from all processors the computation is terminated. Figure 1a compares the initial shape and the final optimized shape of the diffuser, which is compared with the shape of the diffuser corresponding to the target pressure distribution specified for this inverse design problem. Figure 1b compares the target pressure distribution, the pressure distribution corresponding to the optimized diffuser, and the starting pressure distribution corresponding to the initial shape of the diffuser. The solid lines in Figs. 1a and 1b represent the targets that the final designs ought to converge to after the application of PSA algorithms PSA1 and PSA2. It can be seen from Figs. 1a and 1b that PSA has done a good job in carrying out the inverse design problem.

Figure 1c shows the convergence histories of the objective function as it gets minimized from its initial values to the final designed value, which should correspond to zero as a result of using sequential SA (SSA) on a single processor and the PSA algorithms PSA1 and PSA2 on 32 processors. The convergence history shown is recorded from the processor 0 for simplicity. Because the processors are running simultaneously, each processor will have the same number of evaluations of objective function and have the same objective function value at the end of each subchain, that is, at the end of loop 1 in Appendix B. As the number of processors is increased, the evaluations of objective function for each processor and the wall-clock time is reduced. It can be seen that the number of iterations to reach the final optimal design shape according to the termination criteria reduces 10-fold, if MPI is used to implement PSA instead of the SSA on the SGI machine. This shows the benefits of using MPI for this problem both in terms of reduction in the number of iterations, which saves the cost of CFD calculations for evaluating objective functions for each design iteration, and also the tremendous reduction in the turnover time.

Figure 1d compares the variation of the wall-clock time t_p with the number of processors p for PSA1 and PSA2. Figure 1e compares the variation of the number of evaluations, M_p, of the objective functions on each processor vs the number of processors for both PSA1 and PSA2. It appears that the PSA algorithm PSA1 shows efficient speedup if 1–16 processors are used in the computational task and also registers a reduction in the wall-clock time from 4 to 0.55 h. This also shows that using 32 processors will result in only marginal benefits because the wall clock time has reduced to 0.47 h. The reason for this is that, by increasing the number of processors, the length of the chain is reduced to a limit value for which the performance of PSA will no longer be enhanced as the computational overhead increases. It can also be seen that the number of function evaluation decreases for each processor if more processors are used in the computation. It can also be seen from Fig. 1e that the PSA algorithm PSA2 reduced M_p from over 6000 on single processor to around 500 for each processor while program running on 32 processors. It can be seen that on 32 processors PSA2 fares better than PSA1. The speedup ($S_p = t_p/t_1$) using PSA1 on 16 processors is 7.3 whereas that using PSA2 is about 8.6. The ideal speedup is defined by $S_{pi} = M_p/M_1$ and is compared in Fig. 1f. It can be seen that the communication overhead (obtained by subtracting real value from ideal value) of PSA2 is higher than PSA1. In view of this, all subsequent design problems are optimized using PSA2.

Design Case 2: Tunnel Wall Design

Design case 2 shows the feasibility of using PSA for solving two-dimensional shape optimization problems on multiple processors. The two-dimensional solver for the Euler equations is used for the shape design of the lower wall of a converging nozzle inside which there is high-speed compressible flow. The initial shape of the lower wall is a straight line and oriented like a wedge (see Fig. 2a). The top wall is parallel to the direction of flow along the x direction. The shape of the bottom wall, which looks like a wedge, generates a strong reflective shock wave between bottom and top walls. The task at hand is to redesign the shape of the lower wall to eliminate or weaken the shock wave. Because it is not possible to have a priori knowledge of the type of target (desired) pressure distribution P_t corresponding to a pressure distribution, which does not create a shock at the lower wall, basic theory of gasdynamics states that a compressible flow past a gradually turning wall (compressive corner flow) can produce very weak shocks. This forms the basis for the selection of the target pressure distribution by computing the compressible flow past the lower wall, which varies as $y = x^{3.5}$. For this case, the objective function is defined as

$$F(X) = \frac{1}{P_0} \int (P - P_t)^2 \, \mathrm{d}x \tag{5}$$

where P_0 is the pressure at the inflow. The integration of the difference between the target and design pressure distribution is calculated along the bottom wall line, where the shock effects are felt the strongest. The inflow Mach number is 2.2. The shape of the lower wall is represented by cubic splines as by Press et al.[23] The goal of cubic spline interpolation is to obtain an interpolation formula that is smooth in the first derivative and continuous in the second derivative, both within an interval and its boundaries. Four design variables are chosen for the design optimization, namely, two first derivatives at the inlet and outlet boundaries and two height parameters, y_1 and y_2 at $x_1 = 5.0$ and $x_2 = 8.0$, respectively. The Euler equations from Eq. (2) are solved numerically for calculating the objective function. A structured grid of 101×40 grid points is used for computing the target flowfield and the flow calculation for each evolving design shape until the final shape is attained.

Results and Discussion of Design Case 2

For this case, the SA tuning parameters appearing in the cooling schedule of SA are altered such that $\gamma = 0.15$ and the tolerance criterion is set at $F_{\min} = 4.0E-2$. The length of the main chain is $20N$. The initial shape of the lower wall of the nozzle and the computed flowfield, which contains a strong oblique shock wave that is reflected off the top wall, are shown in Fig. 2a, which shows the contours of the normalized pressure in the flowfield. This flowfield is the starting flowfield condition used for initiating the design process so that the algorithm PSA2 can find the shape of the lower wall that will eliminate the shock wave for the same flow conditions. Figures 2b and 2c, respectively, compare the optimal shape and the surface pressure distribution on the lower wall computed by the parallel PSA2 algorithm with the wall shape and pressure distributions corresponding to the initial and target flow conditions. It can be seen that the PSA2 algorithm has designed an optimal lower wall shape that has a close agreement with the target flow conditions. Figure 2d shows the contours of the normalized pressure in the flowfield corresponding to the flow past the optimized lower wall shape for the same flow conditions showing the elimination of the strong shock

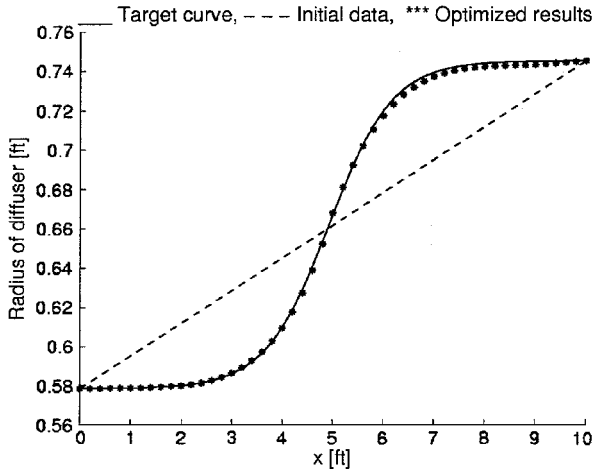

Fig. 1a Diffuser contour.

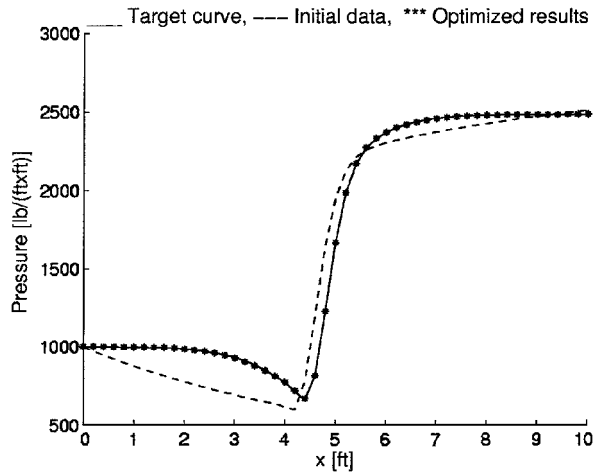

Fig. 1b Pressure distribution.

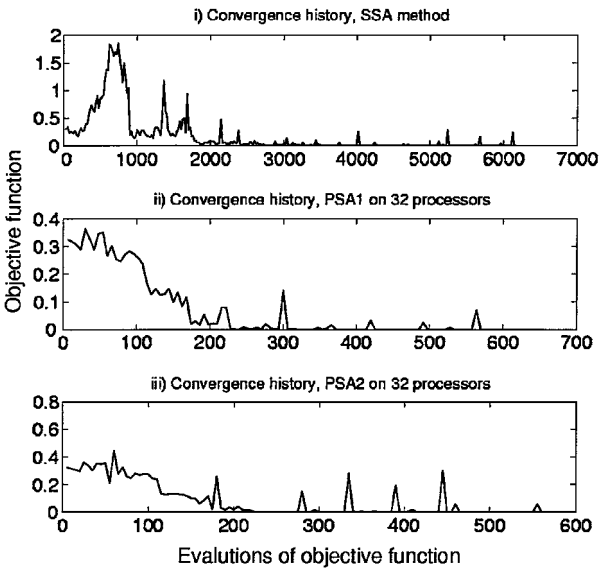

Fig. 1c Convergence history (case 1).

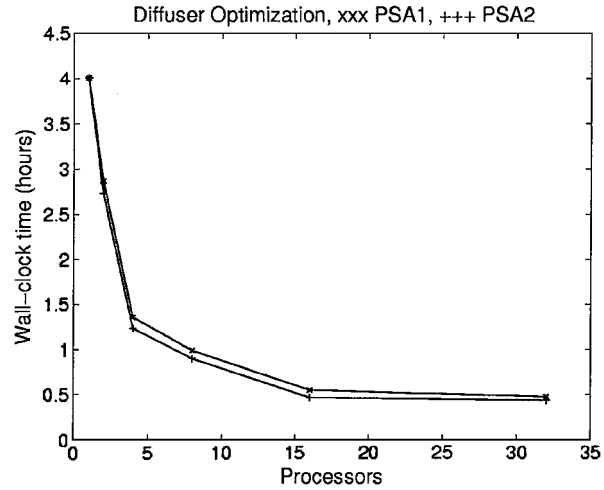

Fig. 1d Wall-clock time (case 1).

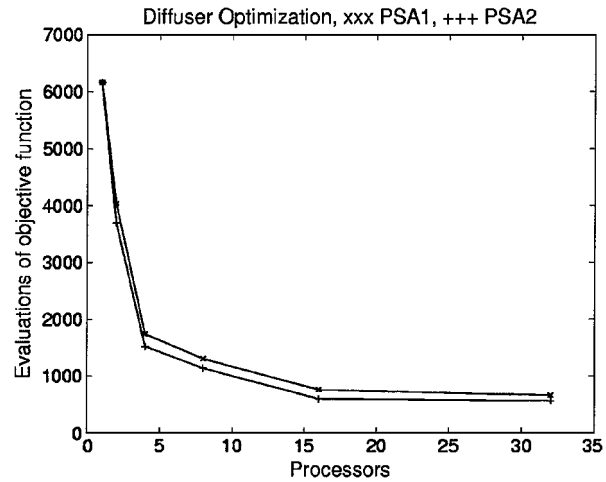

Fig. 1e Evaluations of the objective function vs number of processors (case 1).

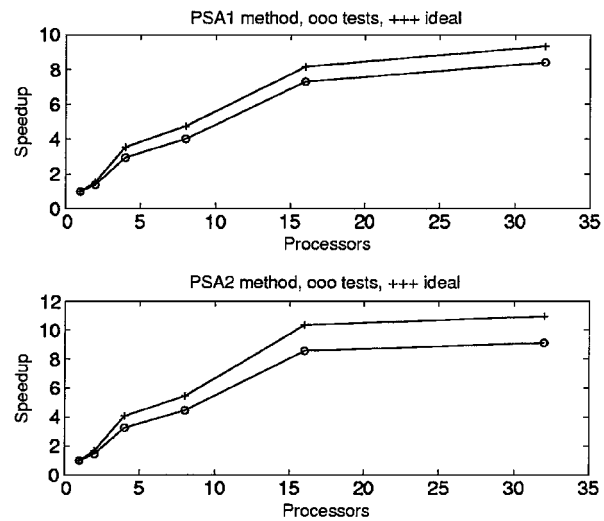

Fig. 1f Speedup for case 1.

wave from the flowfield. The computation also shows that the number of evaluations of the objective functions decreased remarkably from 653 on a single processor and on a multiprocessor for each processor to 185 on 4 processors, to 77 on 8 processors, to 41 on 16 processors, and to 33 on 30 processors. The convergence history of the optimization by PSA2 algorithm (based on information gleaned from processor 0) on 1, 4, and 16 processors is shown in Fig. 2e. Figure 2f shows the convergence history on four processors based on the information from each of the processors from processor 0 to processor 3. The minor differences that exist in the initial stages of the optimization in the convergence histories recorded in each processor result from different random searches (as a result of different random seeds) conducted in each processor. However, the final evaluation of objective function and the optimized result are the same for each processor. Figure 2g shows the wall-clock time and speedup attained from using 4, 8, 16, and 30 processors for this

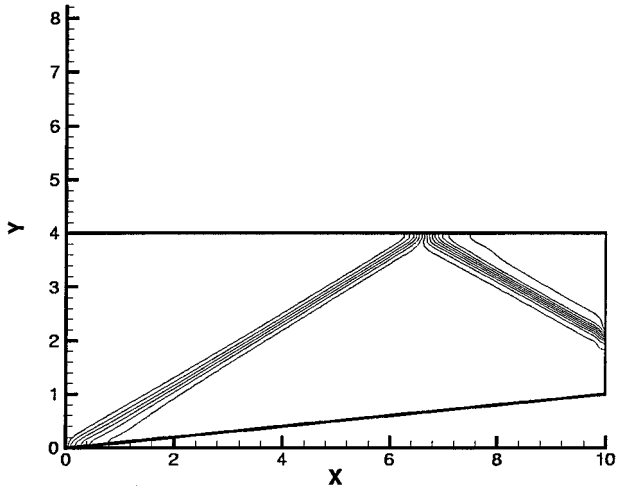

Fig. 2a Initial flowfield and shape of the lower wall (case 2).

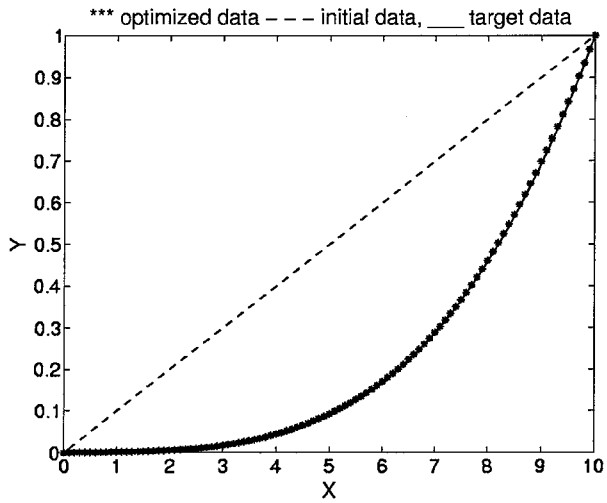

Fig. 2b Optimized shape of the lower wall (case 2).

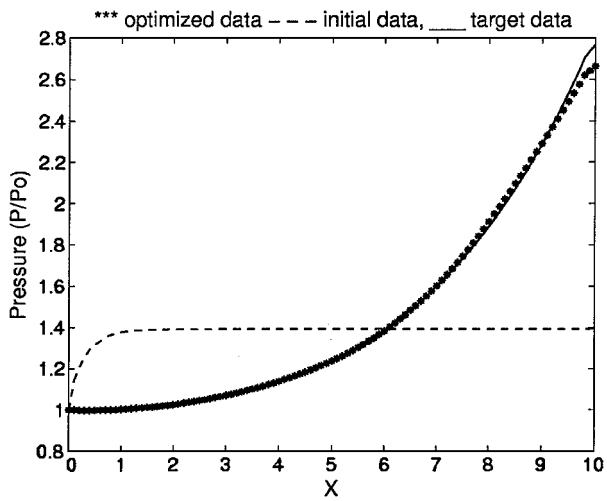

Fig. 2c Surface pressure distribution on the lower wall.

calculation. It can be seen that the wall-clock time has been reduced from 35 h on single processor to around 2.2 h on 16 processors and to 1.8 h on 30 processors. The speedup reaches 15.8 for 16 processors and 19.5 for 30 processors. Based on these observations, it can be concluded that PSA is a promising method for design optimization. It appears that the number function evaluations can be reduced to a reasonable value, which can compete with the economy offered by deterministic local methods.

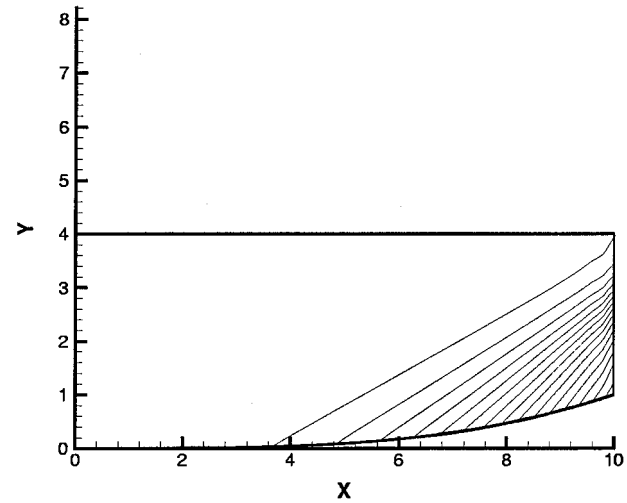

Fig. 2d Optimized flowfield and shape of the lower wall (case 2).

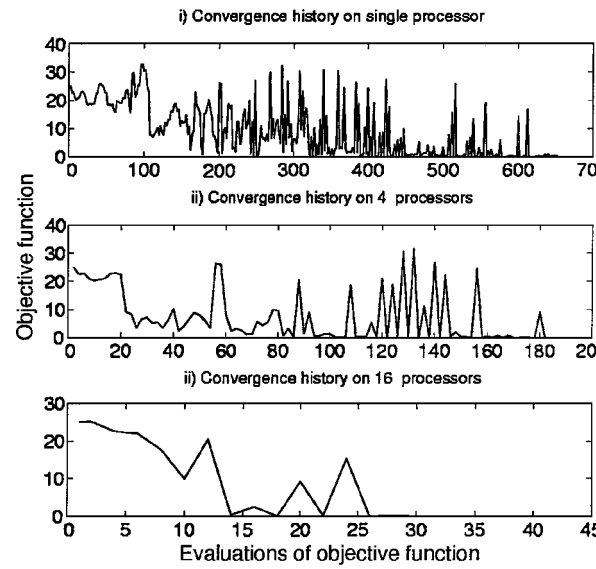

Fig. 2e Convergence history of the objective function for case 2.

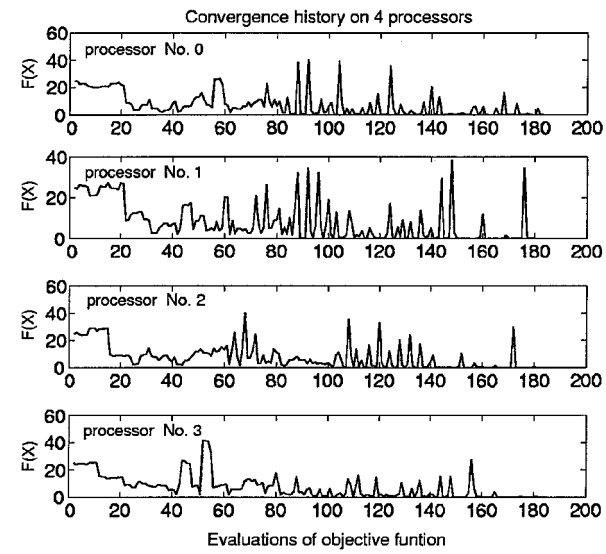

Fig. 2f Convergence history on four processors for case 2.

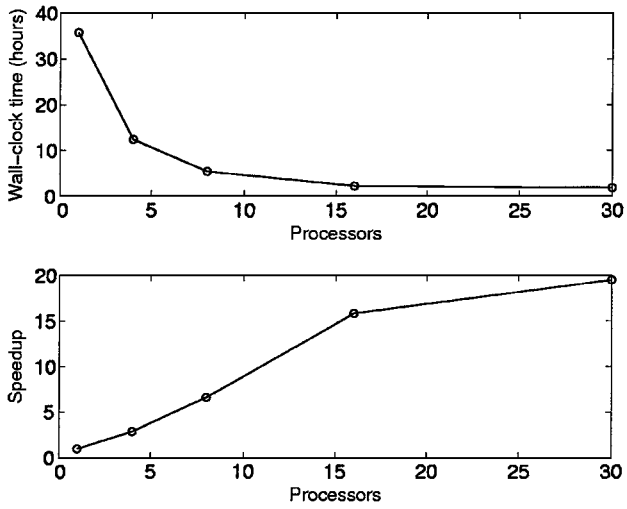

Fig. 2g Wall-clock time and speedup (case 2).

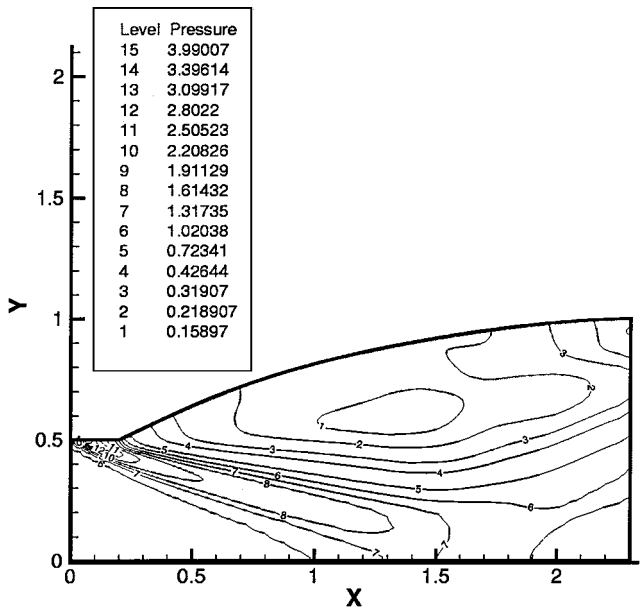

Fig. 3a Initial flowfield and shape of the nozzle (case 3).

Design Case 3: Axisymmetric Nozzle Design A

Recent applications of the Navier–Stokes (NS) solver in the design of hypersonic wind-tunnel, scramjet, ramjet, aerospike, and rocket nozzle, as by in Korte and Kumar[24] and Hussaini and Korte,[25] show remarkable advantages over traditional classical design methods based on low-order physics. Design case 3 is concerned with the design of a nozzle shape using an NS solver, which maximizes the thrust of a supersonic axisymmetric nozzle. Here the objective function is simply defined as

$$F(X) = \int (P/P_0)\,\mathrm{d}S \tag{6}$$

where between the grid points i and $i+1$, the area $\mathrm{d}S = \pi(y_{i+1}^2 - y_i^2)$. The integration is evaluated along the surface of nozzle wall. The flowfield is calculated by numerically solving the full NS equations in Eq. (2). A structured grid of 101×40 grids with clustered grids near the nozzle wall surface is used for the CFD analysis. The inflow Mach number is 4.84 and Reynolds number based on diameter of inlet is $Re = 1.35E{+}8$. The values of the radii are 0.5 m at the inlet cross section and 1.0 m at the exit. The length of nozzle is 2.52 m. The curve defining the shape of the nozzle is represented by cubic splines. Four design variables, namely, inlet expansion half-angle α_1, radii y_1 and y_2 at $x_1 = 0.5$ m and $x_2 = 1.5$ m, and outlet expansion half-angle α_2 of the nozzle wall, are used for the

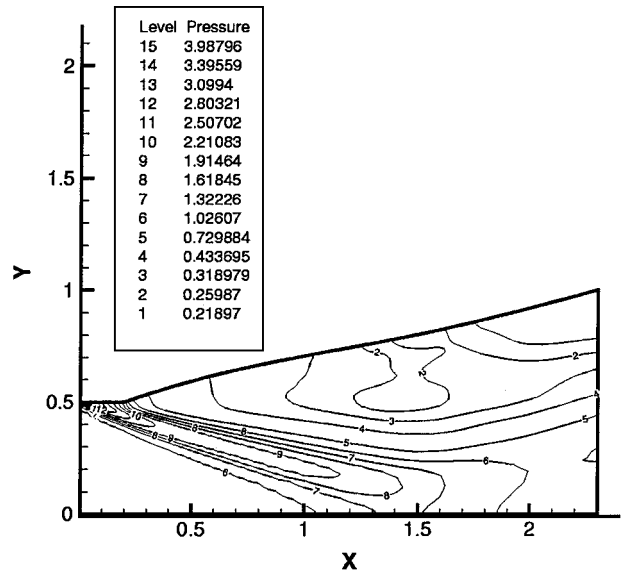

Fig. 3b Optimized flowfield and shape of the nozzle (case 3).

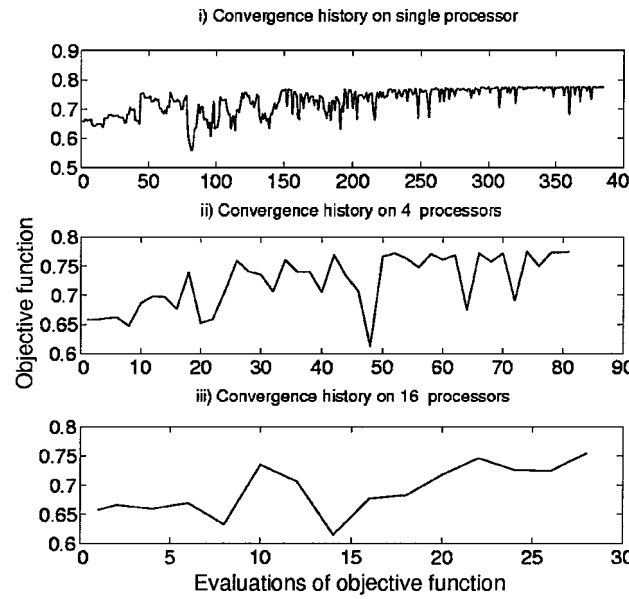

Fig. 3c Convergence history of the objective function for case 3.

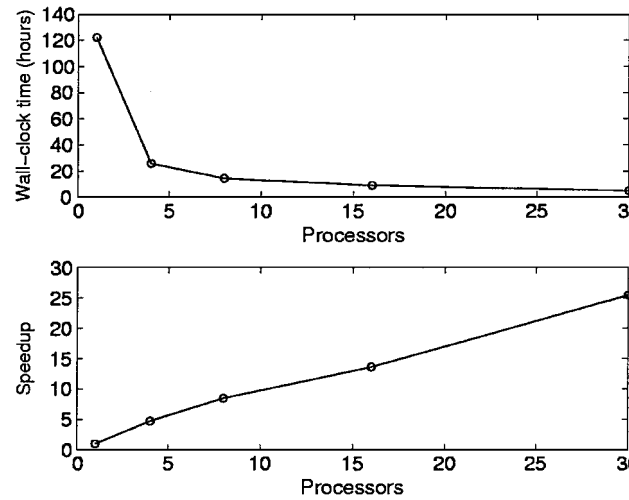

Fig. 3d Wall-clock time and speedup (case 3).

shape optimization process. Note that maximization of a function is done by the minimization of its reciprocal.

Results and Discussion of Design Case 3

The PSA2 algorithm is used for the optimization, and the termination criterion for the convergence is set such that the residual $|F(X)_{i+1} - F(X)_i| \leq 1.0E{-}3$, $i = n$ and $n+1$, is satisfied (where n is the optimization search iteration). The parameter γ for the cooling schedule is set to be 0.25, and the length of the main chain is $8N$. This case is the most computationally intensive of all three of the cases because this is direct design problem and the objective function is evaluated using the CFD solver for NS equations. For comparative purposes, the same values of the parameters $\alpha_1 = 25$ deg, $y_1 = 0.64$, $y_2 = 0.92$, and $\alpha_2 = 1.0$ deg are chosen for the optimization procedure on multiple processors.

Figure 3a shows the computed local pressure contours of the original nozzle shape, which is delivering a certain amount of thrust and corresponds to the starting flowfield for the optimization studies. Figure 3b shows the computed flowfield corresponding to the optimized nozzle shape, which has been designed from the original shape to satisfy the thrust delivery constraint. Comparison between Figs. 3a and 3b shows that the optimized nozzle has higher pressure at the entrance and a slightly lower pressure in the region near the exit, and this results in a higher thrust on the whole nozzle. The computation also shows that the number of objective function evaluations required has been reduced from 385 on a single processor and on a multiprocessor for each processor to 81 on 4 processors, to 45 on 8 processors, to 28 on 16 processors, and to 15 on 30 processors. The convergence histories of the optimization by PSA2 algorithm on 1, 4, and 16 processors are shown in Fig. 3c. The wall-clock time and speedup are shown in Fig. 3d, which shows the remarkable gains in speedup. This case further proves that satisfactory speedup can be achieved on multiple processors for both inverse design in cases 1 and 2 and direct design in this case. When compared with the optimized objective function, it can be observed that the optimized result on 16 or 30 processors is slightly better than that obtained using a fewer number of processors; the maximized objective function is 1.3990 on 30 processors and 1.3256 on 16 processors compared to 1.2906 on 8 processors, 1.2910 on 4 processors, and 1.2907 on 1 processor. The results shown in Figs. 3 are based on a grid size of 41×101.

Design Case 4: Axisymmetric Nozzle Design B

This case is the same as case 3, except that the objective function is defined as

$$F(X) = \int (\rho u^2 + P)\, \mathrm{d}S \tag{7}$$

where $F(X)$ is normalized by the inflow condition $\rho_0 u_0^2$. The integration is done on the cross-sectional area at the nozzle exit. The

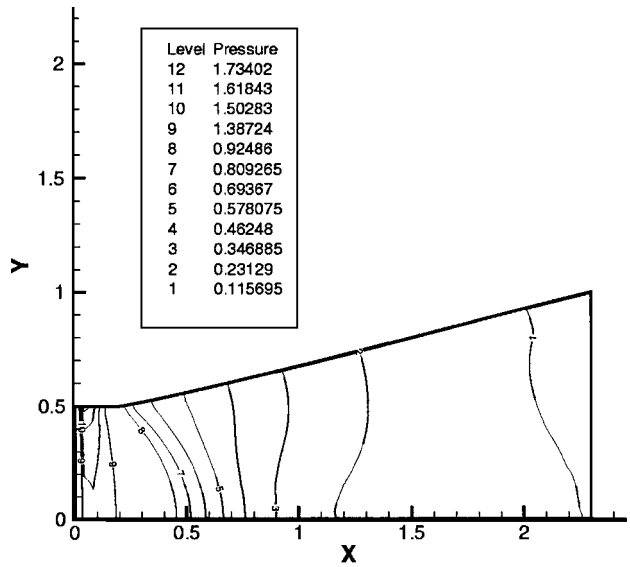

Fig. 4a Initial flowfield and shape of the nozzle (case 4).

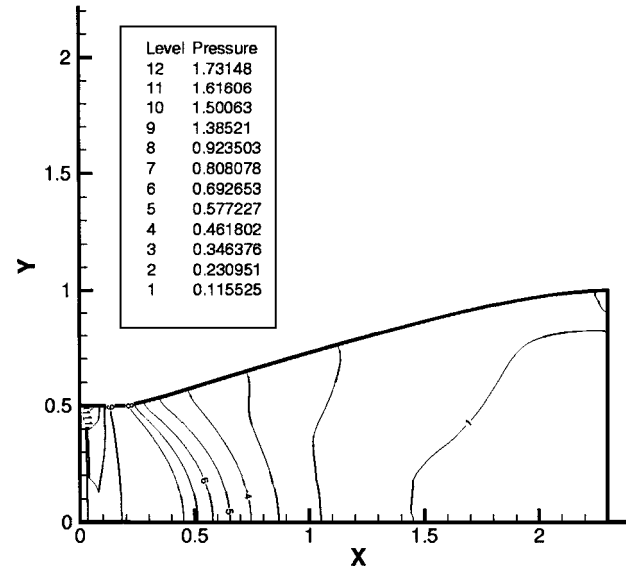

Fig. 4b Optimized flowfield and shape of the nozzle (case 4).

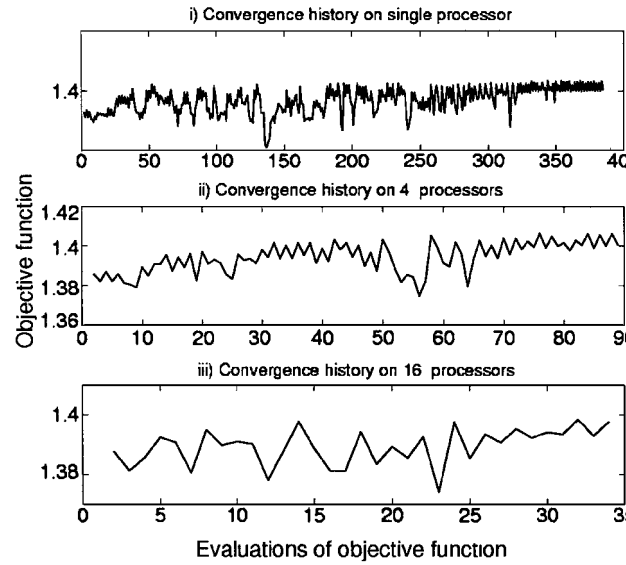

Fig. 4c Convergence history of objective function for case 4.

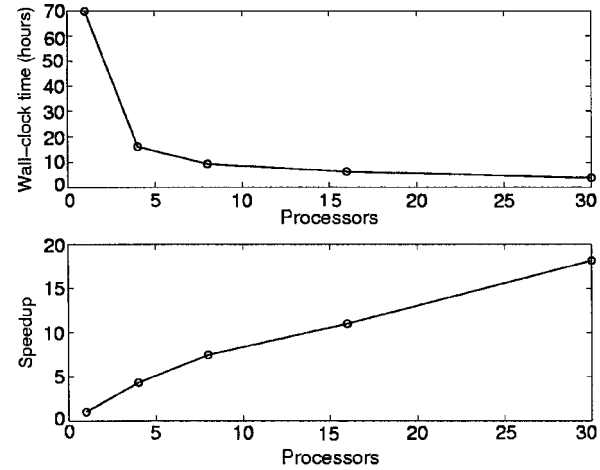

Fig. 4d Wall-clock time and speedup (case 4).

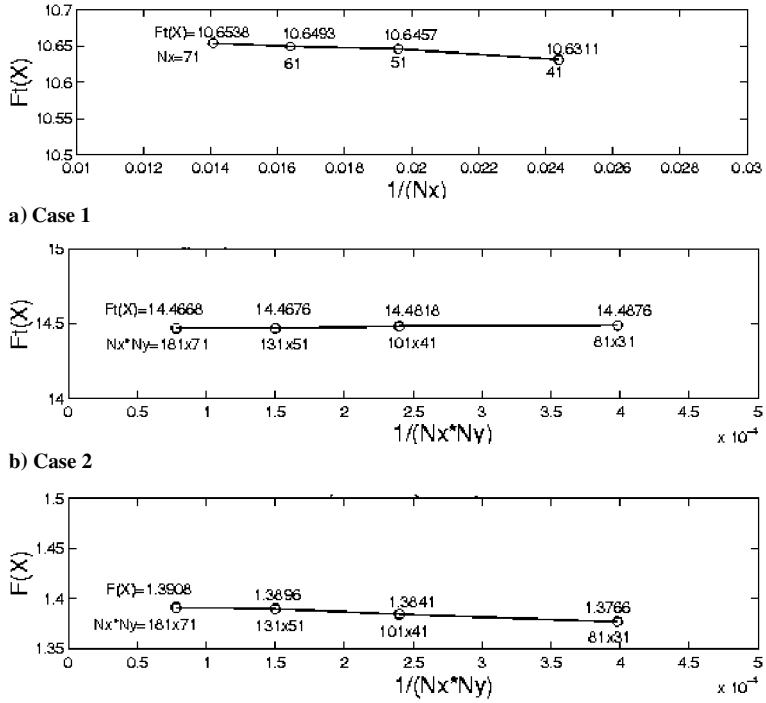

Fig. 5 Influence of grid size on the values of selected objective functions.

flowfield is simulated using the same nozzle parameters as case 3, but the inflow Mach number is 1.2 and Reynolds number based on diameter of inlet is $Re = 2.79E+7$.

Results and Discussion of Design Case 4

The termination criterion for the convergence is set such that, if the residual $|F(X)_{i+1} - F(X)_i| \leq 1.0E-4$, $i = n$ and $n+1$, is satisfied the process stops. The parameter γ and the length of the main chain are the same as in case 3.

Figure 4a shows the computed local pressure contours of the original nozzle shape, which is delivering a certain amount of thrust and corresponds to the initial condition. Figure 4b shows the computed flowfield corresponding to the optimized nozzle shape, which has been designed from the original shape to satisfy the thrust delivery constraint. Comparison of Figs. 4a and 4b shows that the optimized nozzle has a higher pressure at the entrance and a slightly lower pressure in the region near the exit, and this results in a higher thrust for the whole nozzle. The computation also shows that the number of objective function evaluations required has been reduced from 385 on a single processor and on a multiprocessor for each processor to 81 on 4 processors, to 49 on 8 processors, to 34 on 16 processors, and to 21 on 30 processors. The convergence histories of the optimization by PSA2 algorithm on 1, 4, and 16 processors are shown in Fig. 4c. The wall-clock time and speedup are shown in Fig. 4d and show remarkable gains in speedup. Note that the optimized result on 16 or 30 processors is nearly the same as that obtained using a smaller number of processors; the maximized objective function is 1.3929 on 30 processors and 1.3946 on 16 processors compared to 1.4028 on 8 processors, 1.4066 on 4 processors, and 1.4068 on 1 processor.

General Comments on the Design Test Cases

The sensitivity of the converged values of the selected objective function on the size of the computational mesh for the test cases is demonstrated by carrying out the optimization on different grid sizes. Figure 5a shows the influence of computational grid size on the convergence of the final value of the target objective function for case 1,

$$F_t(X) = \frac{1}{\rho_0 u_0^2} \int P_t \, \mathrm{d}x$$

The convergence is established under the criterion $|F_t^{m+1} - F_t^m| < 1.0E-6$ after $m+1$ iterations. Figure 5b shows the influence of the computational grid size on the convergence of the final value of the target objective function for case 2,

$$F_t(X) = \frac{1}{P_0} \int P_t \, \mathrm{d}x$$

The convergence is established under the criterion $|F_t^{m+1} - F_t^m| < 1.0E-6$ of the final value of the target objective function. Figure 5c shows the influence of computational grid size on the convergence of the value of the objective function for case 3 under the convergence criterion $|F^{m+1} - F^m| < 1.0E-6$. The results of case 3 are also valid to analyze case 4 because they have the same geometry parameters. As the grid is refined the values of the selected objective function do not show much differences, and from Fig. 5 one can extrapolate the final values as more grid points are added. However, note that the use of more grid points results in greater computational costs.

The optimized results on the multiprocessor is comparable with those obtained on single processors for cases 3 and 4. However, because the length of the subchain affects the final optimized value of the objective function, a possibility exists that optimization done on a multiprocessor might be worse than that done on a single processor. One possible way to overcome this possibility is to increase the length of the subchain at the expense of some loss in the speedup. It can be seen from the preceding test cases that, by reducing the length of the main chain and by introducing more processors, the limitation in speedup as encountered in cases 1 and 2 and the limitation in performance as encountered in case 3 will be reached eventually. Therefore, a reasonable number of processors should be chosen to maintain higher speedup (or efficiency) of computer system to achieve a satisfactory level of accuracy for the optimized results. It can also be observed from the four test cases that as the number of design variables pertaining to an optimization problem increases (11 design variables for cases 1 and 4 design variables for cases 2–4) the length of the Markov chain must be increased, and this implies that the length of the subchain is also increased, which implies that more processors may be used efficiently in design calculations to reduce wall-clock time.

Conclusions

Aerodynamic shape design of high-speed internal flow using state-of-the-art CFD methods for Euler/NS equations and parallel

SA algorithms have been implemented on multiple processors in this work. Design test cases considered in aerodynamic shape optimizations in which objective functions are evaluated with a CFD solver and shapes are represented by suitable design variables via parametric representations clearly demonstrate that the two modes of the PSA algorithm, that is, PSA1 and PSA2, can remarkably reduce the evaluations of the objective function on each processor and, therefore, reduce the calculation time (wall-clock time) for the design optimization process on a coarse-grained multiprocessor. PSA2 performs slightly better than PSA1 on multiple processors. This preliminary study illustrates that it may be premature to consider PSA as a robust approach for the design of complex design problems and that the evaluations of objective function can be reduced nearly to the same order as that obtained by deterministic methods because the efficiency of the PSA algorithms can depend on the number of design variables used to define the design space and the number of processors used. These outcomes may depend on the type of design problem that is being solved. Whereas this work demonstrates the feasibility of using a PSA algorithm for aerodynamic design work, future work based on current work can include detailed studies based on the effect of number of processors and design variables on the performance of PSA approach, the combination of adaptive SA and hybrid optimizer to further enhance the performance of PSA, and conduct more validation and application tests for the complex single/multidisciplinary problems.

Appendix A: Standard SA Procedure

Start loop (1) for given temperature T_k
  Start loop (2) for searching new solution at T_k
    with a reasonable length of search (L) obtain
      new solution and cost
    function F, ΔF = new cost - current cost
    if $\Delta F \leq 0$ or $e^{-\Delta F/T} > random()$
      accept new solution and cost function F
    end if
  Continue loop (2)
  Update $T_{k+1} = \gamma T_k$. If convergence criterion is satisfied,
    terminate loop (1)
Continue loop(1)

Appendix B: PSA Procedure

Initial multiple processors
Generate different random seeds for each processor
Start loop (1) for given temperature T_k
  Start loop (2) for searching new solution at T_k
    set the length of subchain $L_c \approx L/p$
    obtain new solution and cost function F
    ΔF = new cost - current cost
    if $\Delta F \leq 0$ or $e^{-\Delta F/T_k} >$ random()
      accept new solution and cost function F
    end if
    if the acceptance ratio $\leq R_t$
    (start PSA2 and stop PSA1)
      Randomly gather the accepted solution from
        each processor
      Reset the length $L_c = L$
    (end PSA2)
    end if
  Continue loop (2)
  (start PSA1)
  Collect information from each processor, choose
    the best cost function F
  and solutions; each processor starts from the best solution
  (end PSA1)
  Update $T_{k+1} = \gamma T_k$
  If convergence criterion is satisfied, terminate loop (1)
Continue loop(1)

References

[1]Sobieszczanski-Sobieski, J., and Haftka R. T., "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments," AIAA Paper 96-0711, Jan. 1996.

[2]Frank, P. D., Booker, A. J., Caudell, T. P., and Healy, M. J., "A Comparison of Optimization of Optimization and Search Methods for Multidisciplinary Design," AIAA Paper 92-4827, Sept. 1992.

[3]Aarts, E., and Korst J., *Simulated Annealing and Boltzmann Machines, A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley, New York, 1989, p. 12.

[4]Deb, K., *Optimization for Engineering Design, Algorithms and Examples*, Prentice–Hall of India, New Delhi, India, 1998, pp. 290–333.

[5]Ingber, L., "Adaptive Simulated Annealing (ASA): Lessons Learned," *Journal of Control and Cybernetics*, Vol. 25, No. 11, 1996, pp. 33–54 (Invited Paper in Special Issue Simulated Annealing Applied To Combinatorial Optimization).

[6]Ruppeeiner, G., Pedersen, J. M., and Salamon, P., "Ensemble Approach to Simulated Annealing," *Journal of Physics I*, Vol. 1, 1991, pp. 455–470.

[7]Desai, R., and Patil, R., "SALO: Combining Simulated Annealing and Local Optimization For Efficient Global Optimization," *Proceedings of the 9th Florida AI Research Symposium (FLAIRS'96)*, Florida Artificial Intelligence Society, Key West, FL, 1996, pp. 233–237; also Los Alamos National Lab., TR LA-UR-95-2862, Albuquerque, NM, 1995.

[8]Gallego, R. A., Alves, A. B., Monticelli, A. A., and Romero, R. "Parallel Simulated Annealing Applied To Long Term Transmission Network Expansion Planning," *IEEE Transactions on Power Systems*, Vol. 12, No. 1, 1997, pp. 181–189.

[9]Bhandarkar, S. M., and Machaka, S., "Chromosome Reconstruction from Physical Maps Using a Cluster of Workstations," *Journal of Supercomputing*, Vol. 11, No. 1, 1997, pp. 61–87.

[10]Van Laarhoven, P. J. M., and Aarts, E. H. L., *Simulated Annealing: Theory and Applications*, Kluwer Academic, Norwell, MA, 1987, pp. 139–156.

[11]Witte, E. E., and Franklin, M. A., "Parallel Simulated Annealing Using Speculative Computation," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 2, No. 4, 1991, pp. 493–493.

[12]Diekmann, R., Luling, R., and Simon, J., "Problem Independent Distribution Simulated Annealing and its Applications," *Applied Simulated Annealing*, Lecture Notes in Economics and Mathematical Systems 396, Springer-Verlag, New York, 1993, pp. 17–44.

[13]Wang, X., and Damodaran, M., "Comparative Study of Parallel Stochastic Global Optimization Algorithms," *Optimization and Engineering* (submitted for publication).

[14]Hoffmann, K. A., and Chiang, S. T., *Computational Fluid Dynamics for Engineers*, Vol. 1, Engineering Education System, Wichita, KS, 1993, p. 21.

[15]Yoon, S., and Jameson, A., "Lower–Upper Symmetric-Gauss–Seidal Method for the Euler and Navier–Stokes Equations," *AIAA Journal*, Vol. 26, No. 9, 1988, pp. 1025–1026.

[16]Yee, H. C., and Harten, A., "Implicit TVD Schemes for Hyperbolic Conservation Laws in Curvilinear Coordinates," *AIAA Journal*, Vol. 25, No. 2, 1987, pp. 266–274.

[17]Wang, X., Spiegler, E., and Timnat, Y. M., "Optimization of Ram Accelerators System Design," *Journal of Propulsion and Power*, Vol. 16, No. 2, 2000, pp. 269–277.

[18]Wang, X. and Damodaran, M., "Aerodynamic Shape Optimization Using Computational Fluid Dynamics and Parallel Simulated Annealing Algorithms," AIAA Paper 2000-4847, Sept. 2000.

[19]Pacheco, P. S., *Parallel Programming with MPI*, Morgan Kaufmann, San Francisco, CA, 1997, pp. 1–330.

[20]"Fortran 77 Programmer's Guide," SGI Technical Document 007-0711-060, SGI, Inc., Mountain View, CA, 1997.

[21]Faux, I. D., and Pratt, M. J., *Computational Geometry for Design and Manufacture*, Ellis Horwood, London, 1979.

[22]Aly, S., Marconi, F., Ogot, M., Peiz, R., and Siclari, M., "Stochastic Optimization Applied to CFD Shape Design," AIAA Paper 95-1647, Jan. 1995.

[23]Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical Recipes in Fortran: The Art of Scientific Computing*, 2nd ed., Univ. of Cambridge Press, Cambridge, England, U.K., 1997, pp. 107–122.

[24]Korte, J. J., Kumar, A., Singh, D. J., and Grossman, B., "Least-Squares/Parabolized Naviers–Stokes Procedure for Optimizing Hypersonic Wind-Tunnel Nozzles," *Journal of Propulsion and Power*, Vol. 8, No. 5, 1992, pp. 1057–1063.

[25]Hussaini, M. M., and Korte, J. J., "Investigation of Low-Reynolds-Number Rocket Nozzle Design Using PNS-Based Optimization Procedure," NASA TM 110295, Nov. 1996.

P. Givi
Associate Editor