Int. J. Simul. Multidisci. Des. Optim. **2**, 273-282 (2008)

DOI: 10.1051/ijsmdo/2008037

Available online at:
http://www.ijsmdo.org

MPI-enabled Shape Optimization of Panels Subjected to Air Blast Loading

V. Argod[1], A.D. Belegundu[1], A. Aziz[2], V. Agrawala[2], R. Jain[3] and S. D. Rajan[3,a]

[1] Mechanical & Nuclear Engineering, Pennsylvania State University, University Park, PA 16802, USA
[2] Research Computing & Cyberinfrastructure, Pennsylvania State University, University Park, PA 16802, USA
[3] Civil, Environmental and Sustainable Engineering, Arizona State University, Tempe, AZ 85287, USA

Received 12 June 2008, Accepted 25 September 2008

Abstract –The problem of finding the optimal shape of an aluminum plate to mitigate air blast loading is considered. The goal is to minimize the dynamic displacement of the plate relative to the test fixture, while monitoring plastic strain values, mass, and envelope constraints. This is a computationally challenging problem owing to (a) difficulty with finding optimal shapes with higher dimensional shape variations, (b) non-differentiable, non-convex and computationally expensive objective and constraint functions, and (c) difficulties in controlling mesh distortion that occur during explicit finite element analysis. An approach based on coupling LS-DYNA finite element software and a differential evolution (DE) optimizer is presented. Since DE involves a population of designs which are then crossed-over and mutated to yield an improved generation, it is possible to use coarse parallelization wherein a computing cluster is used to evaluate fitness of the entire population simultaneously. However, owing to highly dissimilar computing time per analysis, a result of mesh distortion and minimum time step in explicit finite element analysis, implementation of the parallelization scheme is challenging. Sinusoidal basis shapes are used to obtain an optimized 'double-bulge' shape.

Key words: structural optimization, shape optimization, blast loads, MPI, load balancing, differential evolution, parallel computing

Nomenclature

$\mathbf{x}$	=	design variables vector
$\mathbf{x}^{\mathbf{L}}$	=	lower limit on design variables
$\mathbf{x}^{\mathbf{U}}$	=	upper limit on design variables
FEA	=	Finite Element Analysis
$\mathbf{G}$	=	vector of x-, y-, z- coordinates of nodes in FE model
$\mathbf{q}^i$	=	i^{th} velocity field (or trial shape change) vector
w	=	z- or plate normal displacement
k	=	number of design variables
$\varepsilon_{\max}$	=	maximum plastic strain at failure
ε	=	equivalent plastic strain
M	=	total mass of the structure
$M_{\max}$	=	upper limit for the mass of the structure
T	=	thickness of the structure (plate) at any (x, y) location in the plate
$t_{\min}$	=	Minimum thickness allowed
$\mathbf{z}$	=	vector of z-coordinate of the nodes
$\mathbf{z}^{\mathbf{U}}$	=	upper limit on z-coordinate
$\mathbf{z}^{\mathbf{L}}$	=	lower limit on z-coordinate
$(\det J_j)$	=	smallest value of the Jacobian of the j^{th} hexahedral element at all the eight nodes
n_{pop}	=	Population size
n_{gen}	=	Number of generations
n_{eval}	=	Number of function evaluations
n_p	=	Number of processors

1 Introduction

Shape optimization of panels subjected to dynamic loading has unique challenges one of which is the considerable compute time required for carrying out the finite element analysis, an integral part of function evaluation [1, 2]. The analysis of metallic and composite panels subjected to both ballistic and blast load has been carried out by many researchers. Dharaneepathy and Sudhesh [3] investigate stiffener patterns on a square plate subject to blast loads modeled using Friedlander's exponential function. They demonstrate that stiffeners provide significant advantage in comparison to an unstiffened panel of same weight. They also show disadvantages of a waffle design. Hou [4] considers the effect of stiffener size on blast response. Yen, Skaags and Cheeseman [5] show that significant reduction in the maximum stress amplitude propagating within the protected components can be achieved by suitable selection of honeycomb material with proper crush strength. Icardi and Ferrero [6] study optimum fiber orientations in a laminated composite to absorb energy while maintaining stiffness. Main and Gazonas [7] have studied the effect of blast loading on sandwich plates and on cellular material sandwiched between plates. In contrast to these works, this paper uses formal shape optimization procedures in arriving at optimal shapes of solid aluminum plates. LS-DYNA is used to perform explicit finite element analysis of the structure subject to air blast loading.

While numerical optimization methodologies are fairly mature, their use on engineering design problems for dynamic loading with damage evolution, involving explicit

[a] Corresponding author: s.rajan@asu.edu

finite element analyses, is fairly limited. We use one of the more promising global optimization techniques, viz. Differential Evolution (DE). It is a stochastic direct search method utilizing the concepts developed from the broad class of evolutionary algorithms and can handle non-differentiable objective functions. DE has shown faster convergence compared to other contemporary evolutionary methods (EAs) [8, 9]. DE is similar to GA (genetic algorithms) in that a population of designs are evaluated, mutated and crossed-over a fixed number of generations while storing the best design obtained during the process. Differences between DE and GA exist in the manner of cross-over and mutation and in representing variables. Like most of the EAs, DE can also be easily parallelized since each member of the population can be evaluated individually.

In fact, use of a simpler differentiable optimizer was first attempted but without success, as the non-differentiability of the functions was identified in the process. For this particular problem, DE performs better than an in-house GA code. Response surface optimization, used successfully on crashworthiness optimization with LS-DYNA is another possible route, but has not been chosen here. That is, the route chosen here is to directly couple accurate finite element analysis with optimization. Comparison of these two approaches may be worth studying in future as noted towards the end of the paper.

There are two popular schemes for parallelization of DE (or GA for that matter) [10]. The first scheme employs the Master-Worker model. This is a simple method where the master process controls the program by assigning tasks to worker processes. A 'coarse-grained' parallelization is possible under this scenario. Typically, any function evaluation is done by workers while the master process generates new population using the worker-generated information. The second model [11], which we refer to as 'fine-grained' parallelization, divides the population into subpopulations and associates every entire subpopulation (also called islands) to a process. Each subpopulation converges to its own solution and the relevant information is then exchanged between processes – periodically, best individual from each process is moved to other process (migration). Both these schemes have been implemented and used successfully in solving various types of problems [12, 13]. In this paper, we have implemented a DE optimizer with coarse-grained parallelization.

In most parallelization schemes, the time taken by each function evaluation varies in a narrow range. However, in this work, there is a huge variation in computation time among members in the population, primarily owing to some meshes being distorted for which finite element analysis cannot be carried out and secondarily owing to differences in element thicknesses affecting the time steps used during explicit finite element analysis. Thus, within the framework of coarse-grained parallelization, two different schemes have been studied with respect to obtaining respectable speedup ratios.

The paper is divided into five sections. Section 2 deals with details of the design optimization problem. We look at the design problem statement, the finite element model required for efficient function evaluation, the shape optimization methodology and the generation of velocity field (shape changes). In Section 3, we discuss parallelization strategies needed to decrease the wall clock time required to obtain the solution. Results are presented in Section 4, where the shape of the plate is optimized with three design variables and with a larger space of nine variables. Finally, findings, limitations and future work are discussed in Section 5.

2 Problem Definition

In this paper we use a generic but specific example to illustrate our design optimization methodology for both sequential and parallel implementations. The schematics of the specific example are shown in Fig. 1. The standoff distance of the charge is taken to be 0.4064 m. It should be noted that the plate is a part of a 'grip' assembly (Fig. 2) used to model the experimental condition as explained subsequently. The design optimization problem is as follows.

Given a set of basis shapes that controls the shape and thickness of the plate, a mass limit for the structure, plastic strain limits representing fracture strength, a minimum thickness for the panel, and a geometric envelope within which the structure must lie, determine the best possible combination of these basis shapes that minimizes the deflection (at the first peak in time).

$$
\begin{array}{lll}
\text{Find} & \mathbf{G}(\mathbf{x}) & \\
\text{minimize} & \|w(\mathbf{x})\|_r & (1) \\
\text{subject to} & \varepsilon_j \le \varepsilon_{\max} & \text{for each element } j \\
 & M \le M_{\max} & \\
 & t \ge t_{\min} & \\
 & \mathbf{x}^L \le \mathbf{x} \le \mathbf{x}^U & \\
 & \det J_j(\mathbf{x}) \ge 0 & \text{for each element } j \\
 & \mathbf{z}^L \le \mathbf{z} \le \mathbf{z}^U & \text{(geometric envelope)}
\end{array}
$$

where the notation is explained at the beginning of the paper.

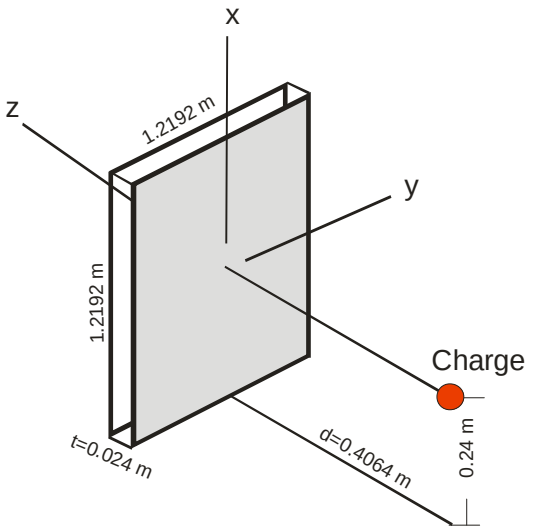


Fig. 1. Baseline structural model details

The defined problem has the following characteristics.
(a) Euclidean norm of the relative z-displacement of nodes in the plate is taken as the objective function (i.e. r = 2 in Eqn. (1)). x- and y- displacements are not significant and are

not considered. The term 'relative' is explained subsequently. The displacement is a function of time and value at first peak is monitored.
(b) Plastic strain, also a function of time, stabilizes after a certain time. This stabilized value is used in the constraint.
(c) *M* refers to the combined mass of the assembly (Fig. 2).
(d) Plate thickness is computed from nodal coordinates of the hexahedral elements used in the FE model.

2.1 FE Modeling Considerations

Our baseline design is a square plate with a grip system or assembly that holds the plate as shown in Fig. 2.

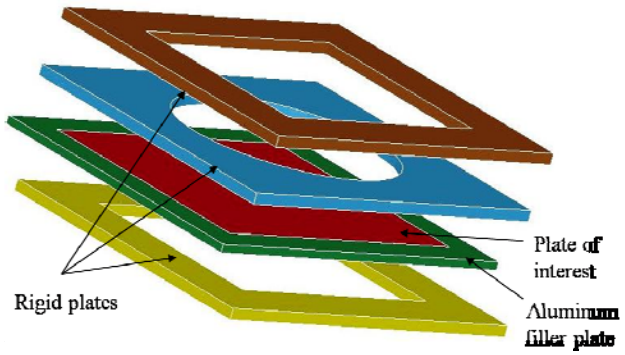


Fig. 2. Exploded view of the structural model

The grip assembly (plate of interest, filler plate and 3 rigid plates) is used in the experimental evaluation of similar panels and is included in the FE model. Appropriate boundary conditions with contact surfaces are used in the assembly. Blast load exerted is calculated using the CONWEP function in LS-DYNA computer program. Given the charge mass and the type of blast, the CONWEP function calculates the pressure values and then applies them to the appropriate surfaces. Since the pressure values are recomputed during every function evaluation, changes in the loading due to shape change are automatically calculated during the design optimization process. Details of the blast parameters can be found in Table 1.

Table 1. Blast Load Input Data

Property	Value
Equivalent mass of TNT	1 kg
Blast Location	(0, 0, -0.4064) m
Type of Burst	Air Blast (spherical charge)

The plate of interest and the surrounding filler plate are made of Aluminum 5083. In the FE model, elasto-plastic material model is used to describe the behavior (Table 2). All the parts are meshed using eight-noded solid elements. Our initial mesh convergence studies showed the FE analysis time for one complete analysis to vary between 90 s for the 4862-element model to 900 s for the 51902-element model on an Intel P4-3.6 GHz machine with 3 GB RAM. Noting that the peak response values (time required for plastic strain to stabilize at a constant value) can shift during the design optimization process due to shape changes in the FE model, the simulation time was taken conservatively to be 11 ms. The plate assembly is unrestrained; hence, the whole assembly is free to move during the blast loading. To eliminate the rigid body component in the objective function in Eqn. (1), $w(\mathbf{x})$ is taken to be the 'relative' displacement obtained by subtracting the nodal displacement at a point in the plate from the displacement of a reference node in the grip.

Table 2. Plate Material Properties

Property	Value
Mass Density	2700 kg/m^3
Young's Modulus	68.9 GPa
Poisson's Ratio	0.33
Yield Stress	225 GPa
Tangent Modulus	633 GPa
Hardening Parameter	1.0
Failure Strain	0.39

2.2 Shape Optimization Methodology

Shape changes are allowed to take place in a square portion of the model at the center of plate. This ensures that changes in plate shape do not result in changes in the grip system (see the circular hole cover plate directly above the plate of interest in Fig. 2). However, a thickness change in the plate has to be matched by an equal thickness change in the filler plate for the assembly to function. The key equation to implement shape optimization is [14, 15, 16]

$$\mathbf{G}(x) = \mathbf{G}_{original} + \sum_{i=1}^{k} x_i \, \mathbf{q}^i \qquad (2)$$

where G is vector of nodal or grid coordinates representing x-, y-, z- coordinates of all nodes in the model. Each xi is the amplitude of a permissible shape change vector, $\mathbf{q}^i$. Vectors $\mathbf{q}^i$ are generated before the start of the iterative optimization loop. $\mathbf{G}_{original}$ contains the grid coordinates of the original flat plate. The optimizer chooses $\mathbf{x}^{opt}$ so that the corresponding shape $\mathbf{G}(\mathbf{x}^{opt})$ is optimum. As $\mathbf{x}$ is iteratively changed by the optimizer and the grid point coordinates $\mathbf{G}$ are updated, the LS-DYNA input file is recreated with the updated nodal coordinates. The overall algorithm and flow of information can be written as follows.
(1) Read the FE model.
(2) Read the design model including all the $\mathbf{q}^i$ vectors (velocity field matrix).
(3) Transfer control to the optimizer,
(4) Start new generation.
(5) Optimizer calls for function evaluations for an entire population of designs.
(6) Construct the updated nodal coordinates using Eqn. (2) and recreate the FE input file.
(7) Carry out the FE analysis.
(8) Read FE results and compute the objective function and constraint values.
(9) Check stopping criteria. If not met, optimizer generates a new generation using crossover and mutation. Go to Step (5).

A pictorial diagram of the algorithm is presented in Fig. 3.

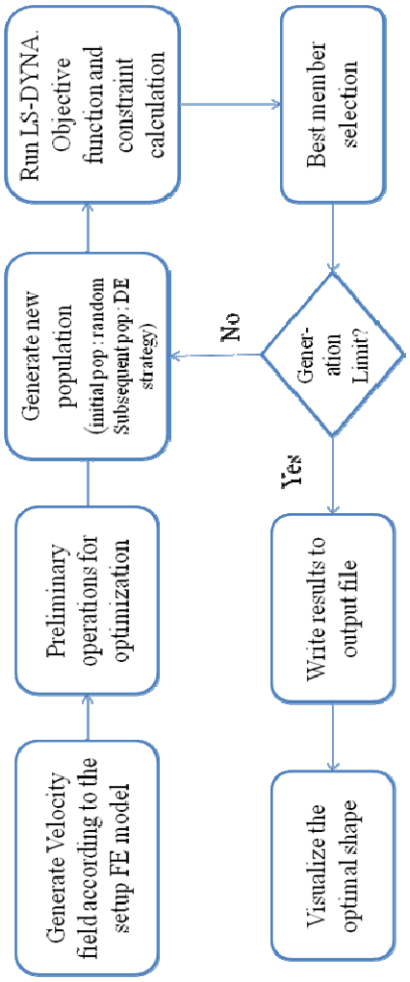


Fig. 3. Flow Diagram of the computer code for shape optimization using LS-DYNA

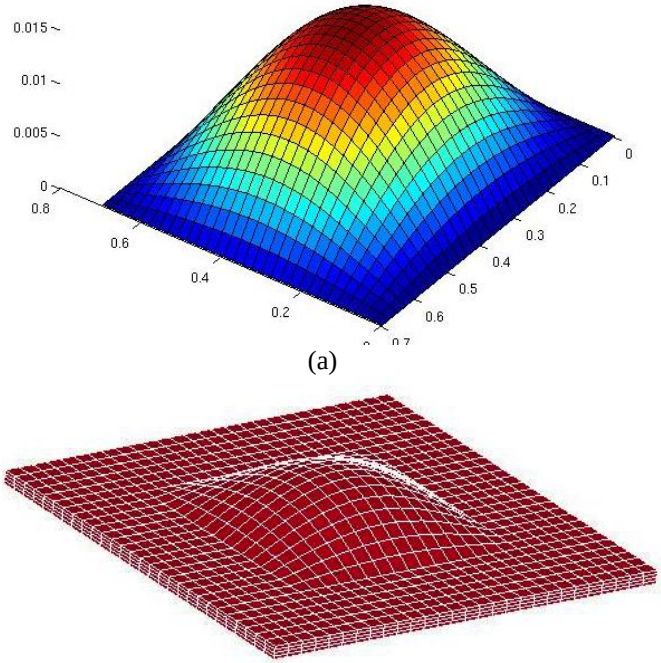
Basis Shapes (or Velocity Fields)

Sinusoidal velocity fields for the top and bottom surfaces, independently, are chosen based on

$$f(m,n) = C \sin\frac{m\pi x}{L} \sin\frac{n\pi y}{L}, \quad m,n = 1,2,3... \qquad (3)$$

where C is a suitable normalization factor. In this paper, optimization results are presented for the three different cases.

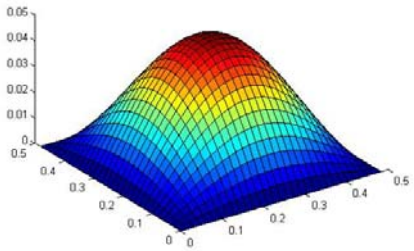
3-DV (Three Design Variables) Case: $m = n = 1$. This gives a total of three (3) *symmetric* basis shapes corresponding to $q^1 \equiv f(1,1)$ for the top surface, $q^2 \equiv f(1,1)$ for the bottom surface, and q^3 = thickness change. Specifically, q^1 represents a bulge in the shape of the top surface while bottom surface is fixed (other thru-thickness nodes are moved to preserve equal spacing), q^2 represents a bulge on the bottom surface while top surface is fixed, and q^3 = a thickness change only. The design variable vector is $x = [x_1, x_2, x_3]^T$ and the optimizer tries to determine an optimum combination of these three basis shapes. Basis shape corresponding to q^1 is illustrated in Fig. 4(a). Note that the bulge can be positive or negative depending on the sign of x_i.



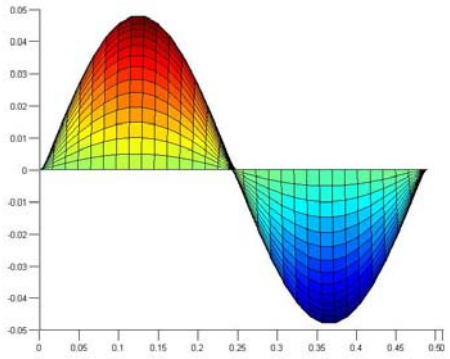
(a)

(b)

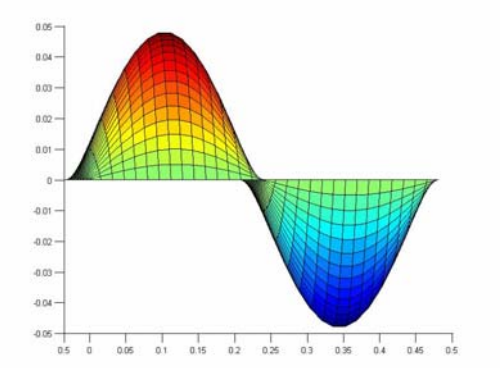
Fig. 4. (a) 3-DV basis shape corresponding to top surface bulge (exaggerated for visualization), (b) Full plate showing the domain for shape optimization of the plate



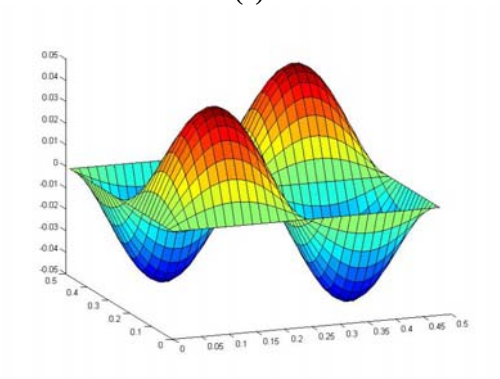
(a)



(b)



(c)



(d)

Fig. 5. Some of the basis shapes for the 9-DV problem (Note that Fig.(b) is the front view of q^2 whereas Fig.(c) is the side view of q^3)

9-DV (Nine Design Variables) Case: $m = n = 2$. This gives a total of nine (9) basis shapes corresponding to $\mathbf{q}^1 \equiv f(1,1)$, $\mathbf{q}^2 \equiv f(1,2)$, $\mathbf{q}^3 \equiv f(2,1)$, $\mathbf{q}^4 \equiv f(2,2)$ for the top surface, $\mathbf{q}^5 \equiv f(1,1)$, $\mathbf{q}^6 \equiv f(1,2)$, $\mathbf{q}^7 \equiv f(2,1)$, $\mathbf{q}^8 \equiv f(2,2)$ for the bottom surface, $\mathbf{q}^9$ = thickness change. It should be noted that unlike the 3-DV Case above, presence of unsymmetric basis shapes may lead to an unsymmetric final optimum shape. Some of the more interesting basis shapes are shown in Fig. 5.

3 Differential Evolution with Coarse Parallelization

Computational experiments showed that downhill or descent search directions did not always lead to a reduction in the objective function even for small steps when using a gradient-based optimizer. The problem was seen to be clearly non-differentiable. The stochastic Differential Evolution (DE) optimizer was chosen for this work. From a computational point of view, DE is similar to GA. An initial population of designs is generated, function evaluations are carried out for each member in the population followed by mutation and crossover operations, leading to a new generation. Salient aspects of DE are as follows.

(i) Function evaluation (here, a finite element analysis with LS-DYNA) of a population of designs can be done independently, on separate processors. Thus, a computing cluster is very attractive to reduce time (as with GA), as clearly, the most time-consuming step is the function or fitness evaluation (FEA).

(ii) The total number of analyses equals product of population size and number of generations. Thus, the total computing time for a job can be ascertained ahead of time in terms of n_p, n_{pop}, and n_{gen} (as with GA).

(iii) Real-valued design variables are stored as such, whereas in GA, a conversion to binary or other representation is involved.

(iv) The mutation and crossover operations are different than GA. Specifically, the following is one strategy used for mutation and crossover:

$$\mathbf{v} = \mathbf{x}_i + \lambda\left(\mathbf{x}_{best} - \mathbf{x}_i\right) + F\left(\mathbf{x}_{r2} - \mathbf{x}_{r3}\right) \quad (4a)$$

$$\mathbf{x}_i^{new} = \left(1-p\right)\mathbf{x}_i + p\mathbf{v} \quad (4b)$$

where $\mathbf{x}_i$ = member in the population, $r2$ and $r3$ are random numbers pointing to random members in the population, λ and F are user-defined parameters, p = probability factor, say 0.5.

In view of (i) above, a multi-processor computing cluster is used. However, different implementations have/need to be researched owing to different computing times for each member in the population, the latter occurring due to two main reasons:

(A) *Due to finite element distortion*. Specifically, each design in the population corresponds to a generally different shape or grid of nodal coordinates as per Eqn. (2). The Jacobian for each finite element is computed at the eight nodes for each hexahedral element. If it is negative at any node, then the element is distorted (non-convex) and finite element analysis (FEA) cannot and is not carried out. A suitably large objective function value is simply returned to the optimizer. Thus, the computing time is essentially zero. It is not possible to avoid this, as new member designs are obtained using a random procedure (Eqn. 4). On the other hand, if the mesh is undistorted, then typically the FEA takes, say, 5 minutes for the mesh size used in this paper. Hence, a large variation in times for each function evaluation exists depending on whether the mesh is distorted or not.

(B) *Each function evaluation involves solving differential equations in time*. The maximum time step depends on the thickness and more generally on element size and shape. The time per analysis can vary by factors of 4:1.

Two parallelization schemes have been implemented, herein called the SATR and LB. Each scheme was successively motivated by the performance of the previous scheme. Details are as given below.

Send-all-then-receive (SATR) Scheme [10]: In this approach, master process divides all the members of the population equally among all the processors. It first sends values of all the design variables associated with each member of the population to appropriate workers. Every worker, once it receives the information, carries out the function evaluation. After finishing all the function evaluation assigned, each worker sends the function values to master. Even though this is a simple scheme, due to variation in the computation time for each function evaluation, there can be unnecessary wait time for the processes that have finished their function evaluations successfully (see (3) above) or cannot evaluate due to problems with the FE model distortion. This can be illustrated with the following sample data. Let n_{pop} = 8, n_p = 4, and FEA time for each member in the population as given in

Table 3 below. The '0 min' FE time indicates distorted meshes for which no FE analysis is carried out.

Table 3. Sample Data to Illustrate Parallel Schemes

Member	1	2	3	4	5	6	7	8
FE time (min)	2	0	6	3	0	3	2	0

From Table 3 we see that a single processor will take 16 min for all 8 evaluations. With 4 processors, with P1 acting both as a master and a worker, the distribution of computations is as indicated in Fig. 6. The 8 function evaluations are distributed in two stages to the 4 processors. Thus, total time for all 8 evaluations equals 6+3 = 9 min. Thus, actual speedup is 16/9 = 1.78, much less than ideal of 4.0 which would occur if each FEA takes equal time.

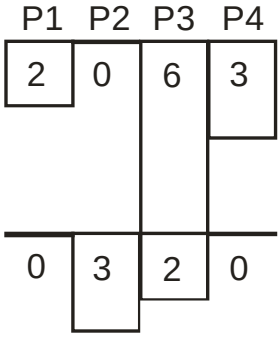


Fig. 6. SATR scheme (numbers refer to minutes, P*i* refers to processor *i*)

Load-balanced (LB) Scheme [10]: A modified scheme is used where send and receive takes place one after another depending on how the availability of the worker process. The master process sends only one individual (of the population) to each worker initially. Once the worker process completes the function evaluation and communicates the function values to the master process, the next individual in the population is sent to the worker for evaluation. In other words, the master process continues to send and receive until all members of population have been evaluated. The overall algorithm is explained next.

Steps in Master Process
(1) Broadcast details of entire population (design variable values) to all the worker processes.
(2) Set number of members already evaluated, $n_{sent} = 0$.
(3) Loop through $i = 1,\ldots,\min(n_{pop}, n_p)$.
(4) Increment n_{sent}.
(5) Ask process i to evaluate member n_{sent}.
(6) End loop
(7) Loop through all the members of the population, $j = 1,2,\ldots,n_{pop}$
(8) Receive function values from process i.
(9) If $n_{sent} < n_{pop}$, increment n_{sent} and ask process i to evaluate member n_{sent}. Else inform process i that there is no more function evaluation to conduct (**no-more-work** message).
(10) End loop.

Steps in Worker Process (only for process $i < n_{pop}$)
(1) Receive the broadcast message and store all the design variables.
(2) Loop till no-more-work message is received from the master process.
(3) Receive the population index, compute and send function values to master process.
(4) End loop.

It should be noted that only the master process executes DE and takes a very small compute time compared to the time taken for a typical function evaluation. On the sample data in Table 3, with P1 acting as the master and P2-P4 doing the FEA, Fig. 7 shows the distribution of computations. We see that maximum time is 6 min. Thus, even with one processor not sharing in FEA, the speedup is 16/6 = 2.67 > 1.78 by SATR Scheme.

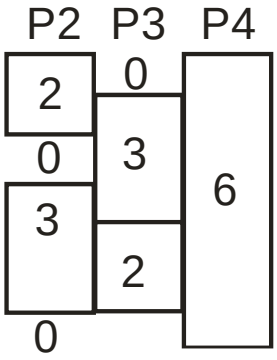


Fig. 7. LB scheme (numbers refer to minutes, P*i* refers to processor *i*)

These parallel algorithms were implemented and executed on the LION-XC cluster located at Pennsylvania State University's Research Computing and Cyberinfrastructure. The compute nodes are dual 3.0-GHz Intel Xeon 3160 (Woodcrest) Dual-Core Processors each with 16 GB of ECC RAM and with Myrinet interconnect. The program was written in FORTRAN and message passing was achieved via MPICH2 calls.

4 Numerical Results

In this section, we look at finding the optimal shapes using 3 and 9 design variables using the parallel shape optimization methodology discussed in the previous sections.

4.1 Distribution of Computing Times for Each Member of the Population

We first study the distribution of computing times for FE analysis of each member of the design population. As noted in Section 3, the unequal FEA times creates challenges to obtain good speedup. Results for the 3-DV (three design variable) problem are used here. Figure 8 shows how the iteration time for every processor varies with optimization iterations with np = 4. The length of each bar represents the time taken for each iteration. Computations for the next generation can commence only after all the processors finish function evaluations for the current generation. This slows down the progress as some processors wait for other processors to finish their computations. The iteration time for any processor is the sum of time taken by all the function evaluations allocated to that processor. Potentially, each worker can get a different number of function evaluations as the master processor tries to balance the

computational load. Next, Fig. 9 shows the distribution of the FE (LS-DYNA function evaluation) times. Bin represents the function evaluation time while frequency gives the number of evaluations in a given bin. Majority of the function evaluations required about 150 s. However, a significant number of members in the generation turn out to be invalid/distorted designs requiring almost zero compute time. Finally, in Fig. 10, the normalized computation time of each iteration is plotted against the iteration number. Departure from equal-time analyses, indicated by a horizontal line, is again evident.

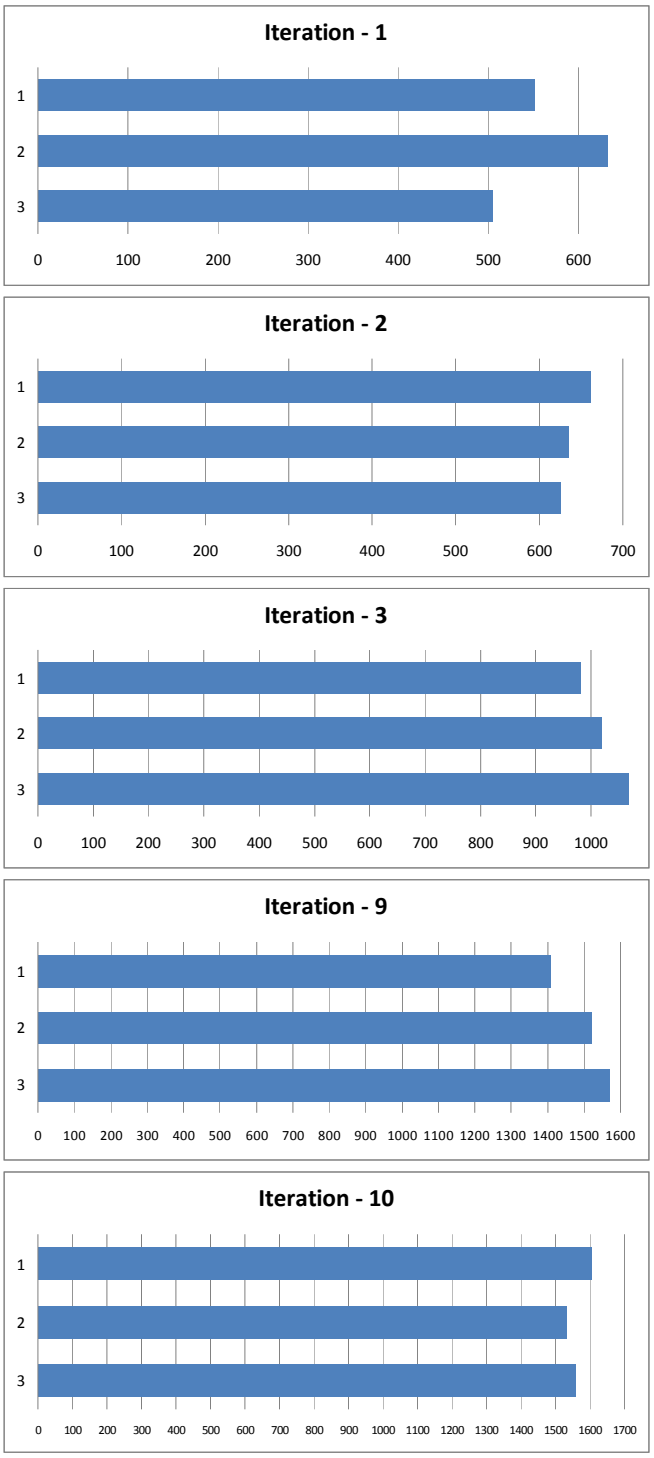


Fig. 8. Bar chart to show iteration time (in seconds for) every processor (for n_p = 4) for 3-DV case

4.2 Obtained Speedups

The SATR and LB schemes were used to solve the design optimization problem described in Section 2. Speedups obtained by the two schemes discussed in Section 3 are shown for the 3-DV and 9-DV cases, respectively, in Figs. 11, 12 and corresponding Tables 4, 5. Speedup is defined in the usual way as $S = T(n_{ref}) / T(n_p)$, where n_{ref} = number of processors used in the. In the 3-DV case in Table 4 (Fig. 11), $n_{ref} = 1$, while in the 9-DV case in Table 5 (Fig. 12), $n_{ref} = 4$ (the 9-DV problem cannot be solved with less processors in reasonable time). Note that the speedup with SATR, when n_p is very low, is better than LB since the latter uses one of the processors as a master which does not share in FEA. As is to be expected, LB overtakes SATR with increasing n_p. Compared to ideal speedup, both fall short, owing to the continuing presence of assorted computing times per FEA as explained in Section 3.

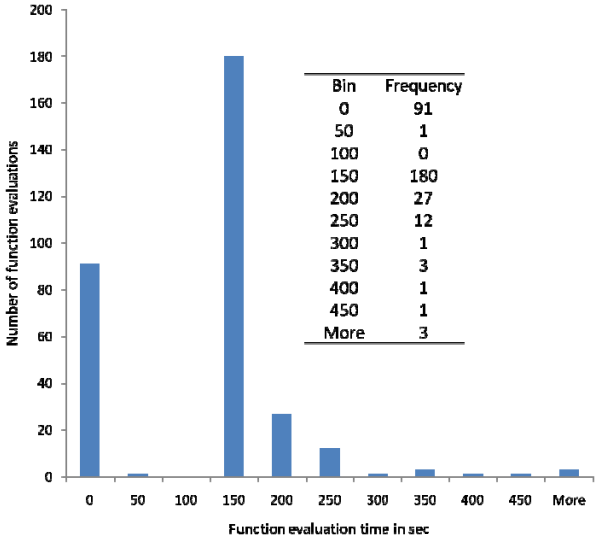


Fig. 9. Histogram of function evaluation times for 3-DV for LB Scheme

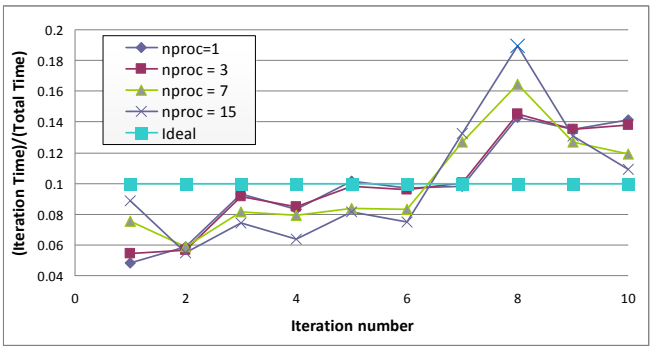


Fig. 10. Plot showing the variation of computation time of every iteration for different number of processors (SATR Scheme)

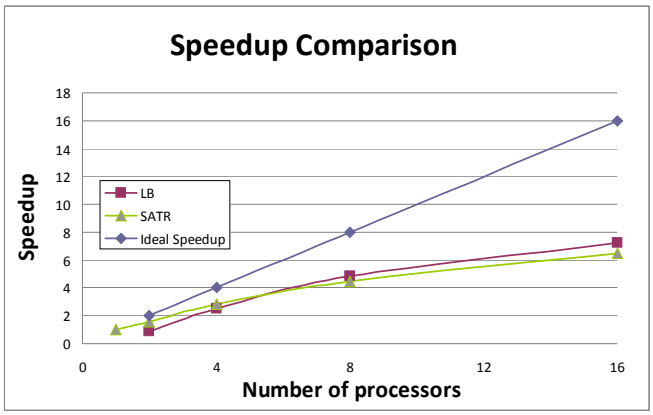


Fig. 11. Speedup comparisons of SATR and LB schemes for 3-DV case

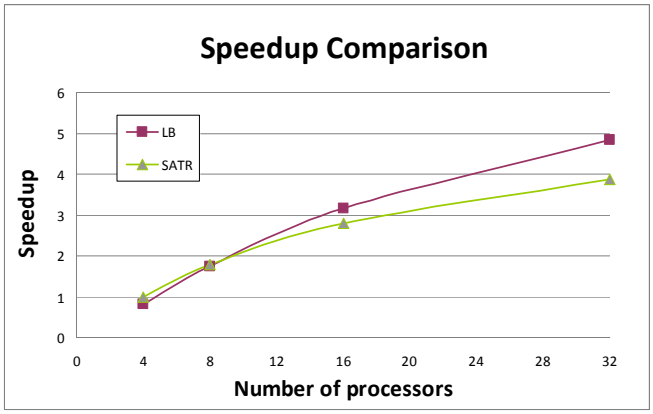


Fig. 12. Speedup comparison of SATR and LB schemes for 9-DV case

4.3 Structural Optimization Aspects

Regarding the structural aspects, the performance metrics associated with the baseline (i.e. flat plate) and optimized designs, for both 3-DV and 9-DV cases, are shown in Table 6.

Baseline and optimized plates within the grip assembly are shown in Fig. 13(a) and (b). The optimized shape turns out to be a double bulge. Overall thickness is reduced from 0.0381m to 0.0241m with bulges on both sides. There is a greater bulge towards the charge (Fig. 13(c)). Plastic strain is maximum at center for the baseline design while it is around the borders of domain for the optimized panel (Fig. 14), indicating greater utilization of material. Maximum relative displacement of optimized designs versus baseline design is shown in Fig. 15(a), where we see an 80% reduction through optimization. The displacement reaches its peak at t = 1.1 ms. The total impulse responses of optimized designs are compared to those of baseline design in Fig. 15(b). Finally, Fig. 16 shows the double bulge shape for the 9-DV problem. This is slightly unsymmetric, owing to the choice of basis shapes used (Section 2).

Limitation: An attempt was made to solve the same problem using 19 design variables (with $m = n = 3$ in Eqn. (3). However, the DE optimizer did not produce optimum solutions. Thus, more interesting shapes need to be explored through improved optimizers and velocity field generators.

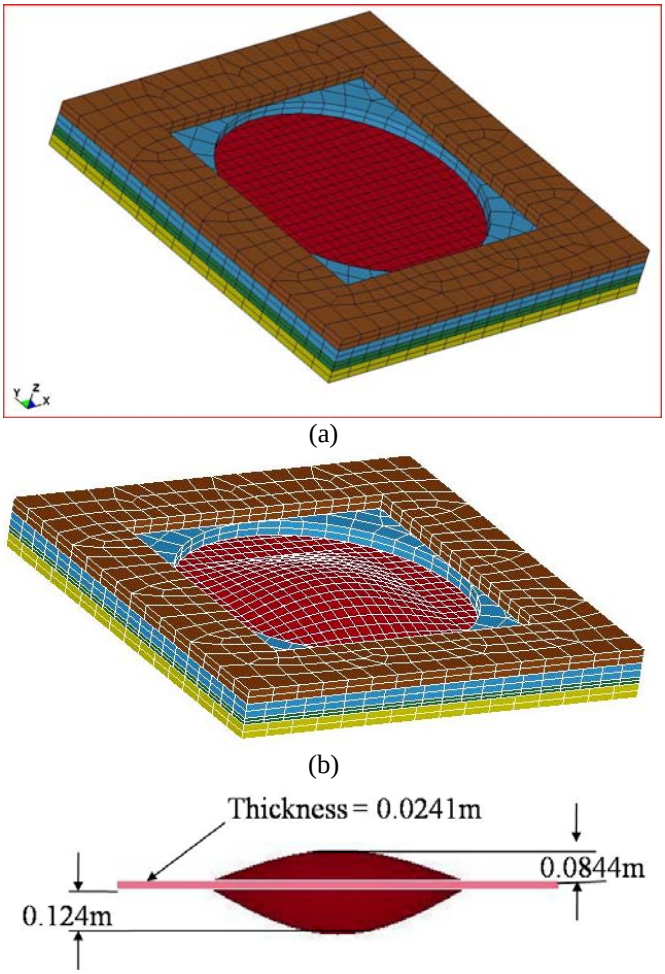


Fig. 13. (a) Baseline design (uniform thickness) (b) Optimized plate-double bulge (for large envelope case) (c) Optimized plate dimensions

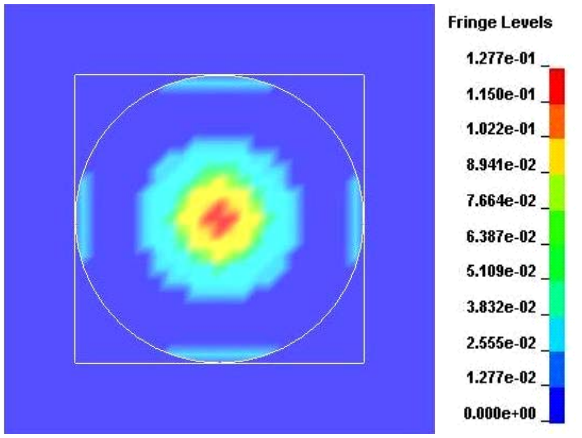
Table 4. Parallel computation for 3 DV, n_{pop} = 32, n_{gen} = 10

n_p	SATR		LB	
	Time (min)	Speedup	Time (min)	Speedup
1	487.6	1.0		
2	306.5	1.6	544.1	0.9
4	171.3	2.8	192.3	2.5
8	109.1	4.5	101.2	4.8
16	75.4	6.5	67.4	7.2

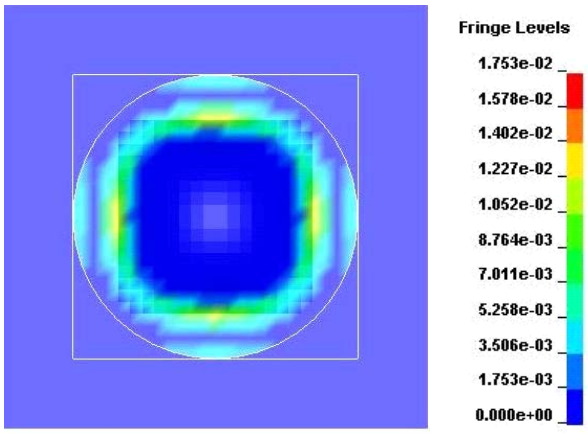
Table 5. Parallel computation for 3 DV, n_{pop} = 100, n_{gen} = 30

n_p	SATR		LB	
	Time (min)	Speedup	Time (min)	Speedup
4	1402.5	1.0	1701.0	0.8

8	784.2	1.8	806.0	1.7
16	502.24	2.8	445.0	3.2
32	362.4	3.9	290.0	4.8



(a)



(b)

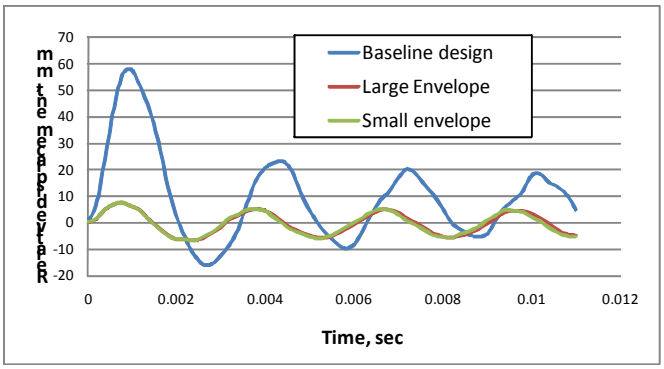
Fig. 14. Plastic strain plot for (a) baseline design (b) optimized design

Table 6. Results for Baseline and Optimized Designs

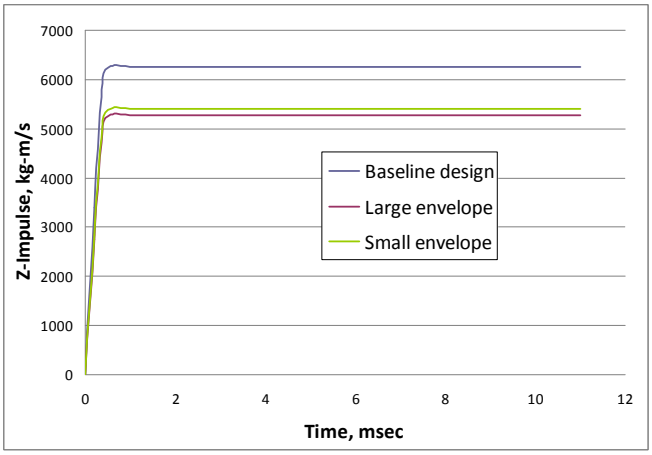
Properties	**Baseline Design**	**3-DV**	**9-DV**
Objective Function (mm)	20.51	4.49	4.28
Max Relative Displacement (mm)	58.43	7.68	7.79
Max plastic strain	0.1277	0.02	0.02
Total Mass (kg)	1872.2	1894.5	1895.3
Saturated impulse, (kg-m/s)	6254.3	5401.4	5313.6

5 Concluding Remarks

Shape optimization of a solid aluminum panel for air blast load mitigation is carried out. LS-DYNA is coupled to a stochastic DE optimizer using a modular FORTRAN code. Three and nine sinusoidal basis shapes are chosen to find the optimum shapes. The optimum shape, a combination of these basis shapes, turns out to be a double-bulge in both cases, with unequal bulges on both sides of the plate. The panel's RMS displacement which is the objective function, relative to the fixture, decreased by 80% compared to the baseline or flat plate design. Maximum plastic strain decreased as well and was well within the constraint limit. At optimum, the plastic strain was smeared, indicating better utilization of the material. Parallelization of the DE optimizer makes the design optimization approach viable. Two schemes are used to implement the coarse parallelization: A study of the two schemes SATR and LB, shows that the latter methodology is better able to handle unequal compute times per FE analysis. When attempting to go beyond 9 basis shapes (i.e. 9 design variables), the optimizer fails to provide a reasonable design. Thus, the task of finding a global optimum in such highly nonlinear, nonconvex and computationally expensive functions is challenging and improved basis shape generators and optimizers are needed in future.



(a)



(b)

Fig. 15. Comparison of (a) relative displacement (b) impulse response

Fig. 16. Optimized shape 9-DV

Acknowledgement

This paper is based on the research work supported by the U.S. Army Research Office (Proposal Number 50490-EG. Technical Monitor: Dr. Bruce LaMattina). We thank Dr. B.A. Cheeseman and Dr. C. Yen of the Army Research Lab for valuable discussions and suggestions. Partial financial and computational support from the High Performance Computing Group at Penn State under Mr. Vijay Agarwala is gratefully acknowledged.

References

1. A. D. Belegundu, V. Argod, S. D. Rajan, K. Krishnan, *Shape optimization of panels subjected to blast loading modeled with LS-DYNA*, *In* 49th AIAA/ASME/ASCE/-AHS/ASC Structures, Structural Dynamics, and Materials. AIAA 2008-2285 (2008).
2. A. D. Belegundu, V. Argod, S. D. Rajan, R. Jain, *Shape optimization of solid isotropic plates to mitigate the effects of air blast loading*, submitted to Intl J of Impact Engineering (2008).
3. M.V. Dharaneepathy, K. G. Sudhesh, *Optimal Stiffening of square plates subjected to air-blast loading*, Computers & Structures, **36**(5), 891-899 (1999).
4. Hai-Liang Hou, *Study on failure mode of stiffened plate and optimized design of structure subjected to blast load*, Explosion and Shock Waves [in Chinese], **27**, 26-33(2007).
5. C.F. Yen, R. Skaggs, B.A. Cheeseman, *Modeling of shock mitigation sandwich structures for blast protection*, The 3rd First International Conference on Structural Stability and Dynamics, Kissimmee, Florida, June 19-22 (2005).
6. U. Icardi, L. Ferrero, *Impact and blast pulse: Improving energy absorption of fibre-reinforced composites through optimized tailoring*, Proc of ESDA 2006: 8th Biennial ASME Conference on Engineering Systems Design and Analysis, Torino, Italy, July 4-7 (2006).
7. J.A. Main, G.A. Gazonas, *Uniaxial crushing of sandwich plates under air blast: Influence of mass distribution*, International Journal of Solids and Structures. **45**, 2297–2321 (2008).
8. R. Storn, *System design by constraint adaptation and differential evolution*, IEEE Transactions on Evolutionary Computation, 22-34 (1999).
9. D.K. Tasoulis, N.G. Pavlidis, V.P. Plagianakos, M.N. Vrahatis, *Parallel Differential Evolution*, Congress on Evolutionary Computation (CEC 2004), Portland, Oregon (2004).
10. S.D. Rajan, D.T. Nguyen, *Design Optimization of discrete structural systems using MPI-enabled genetic algorithm*, Structural Multidisciplinary Optimization, **27**, 1-9 (2004).
11. W. Kwedlo, *Parallelizing Evolutionary Algorithms for Clustering Data*, Springer-Verlag Berlin Heidelberg. PPAM 2005, LNCS 3911, 430-438 (2006).
12. L. Singh, S. Kumar, *Parallel Evolutionary Asymmetric Subsethood Product Fuzzy –Neural Inference System with Applications*, IEEE International Conference on Fuzzy Systems (2006).
13. C.C. Liang, M.F. Yang, P.W. Wu, *Optimum design of metallic corrugated core sandwich panels subjected to blast loads*, Ocean Engineering, **28**, 825–861 (2001).
14. A.D. Belegundu, S.D. Rajan, *A Shape Optimization Approach Based on Natural Design Variables and Shape Functions*, J. Computer Methods in Applied Mechanics and Engineering, **66**, 87-106 (1998).
15. A.D. Belegundu, T.R. Chandrupatla, *Optimization Concepts and Applications in Engineering*, Prentice-Hall, New Jersey (1999).
16. S.D. Rajan, S-W. Chin, L.Gani, *Towards a Practical Design Optimization Tool*, Microcomputers in Civil Engineering, **11**, 259-274 (1996).