Engineering Optimization
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/geno20

MULTIOBJECTIVE DESIGN OPTIMIZATION BY AN EVOLUTIONARY ALGORITHM

TAPABRATA RAY [a], KANG TAI [a] & KIN CHYE SEOW [a]

[a] Center for Advanced Numerical Engineering Simulations, School of Mechanical and Production Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore, 639798

Published online: 27 Apr 2007.

Eng. Opt., 2001, Vol. 33, pp. 399–424
Reprints available directly from the publisher
Photocopying permitted by license only

MULTIOBJECTIVE DESIGN OPTIMIZATION BY AN EVOLUTIONARY ALGORITHM

TAPABRATA RAY*, KANG TAI† and KIAN CHYE SEOW‡

Center for Advanced Numerical Engineering Simulations, School of Mechanical and Production Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798

(Received 25 January 2000; In final form 31 May 2000)

This paper presents an evolutionary algorithm for generic multiobjective design optimization problems. The algorithm is based on nondominance of solutions in the objective and constraint space and uses effective mating strategies to improve solutions that are weak in either. Since the methodology is based on nondominance, scaling and aggregation affecting conventional penalty function methods for constraint handling does not arise. The algorithm incorporates intelligent partner selection for cooperative mating. The diversification strategy is based on niching which results in a wide spread of solutions in the parametric space. Results of the algorithm for the design examples clearly illustrate the efficiency of the algorithm in solving multidisciplinary design optimization problems.

Keywords: Evolutionary algorithms; Pareto optimal solutions; Multiobjective optimization

1. INTRODUCTION

Many design optimization problems in engineering design inherently involve optimizing multiple non-commensurable and often conflicting objectives and criteria that reflect various design specifications and constraints. In the absence of any preference information among the

*e-mail: mtray@ntu.edu.sg
†Corresponding author. e-mail: mktai@ntu.edu.sg
‡e-mail: mkcseow@ntu.edu.sg

objectives, the goal of a multiobjective optimization method is to arrive at a set of Pareto optimal designs. In addition to a number of Pareto optimal designs, a widely varying set of solutions is usually required to allow the decision-maker to choose from the set.

Classical methods are not efficient for multiobjective problems as they often lead to a single solution instead of a set of optimal solutions. Multiple runs cannot guarantee a different point on the Pareto frontier each time and some methods cannot even handle problems with multiple optimal solutions. Evolutionary methods, on the other hand, maintain a set of solutions as a population during the course of the search and thus results in a set of Pareto optimal solutions in a single run. Widely differing Pareto optimal solutions can also be generated by using a diversification strategy within the evolutionary algorithm.

In this paper an evolutionary algorithm is proposed. It incorporates a Pareto ranking concept with diversification and intelligent mating among solutions that excel in the constraint and objective Pareto front. Since the mating among solutions incorporates the knowledge of constraint satisfaction and the performance in the objective function domain, convergence is expected to be faster than conventional strategies. The method does not involve any aggregation of objectives or constraints and thus the problem of scaling does not arise.

Section 2 provides a brief overview of the concepts before introducing the algorithm and its details in Section 3. The performance of the algorithm is illustrated by the four multidisciplinary design examples in Section 4.

2. BACKGROUND

The goal in a multiobjective optimization problem is to arrive at a set of Pareto optimal solutions. As identified earlier, it is necessary to provide a wide variety in the set of solutions for the decision-maker to choose from. There have been a number of approaches to arrive at a set of Pareto optimal solutions. Fonseca and Flemming [1] present an excellent review while a comprehensive recent survey of various multiobjective optimization methods has been reported by Coello

Coello [2]. Section 2.1 lists the major approaches towards multiobjective formulations and outlines their drawbacks.

Most real life design examples involve constraints. The presence of constraints significantly affects the performance of an optimization algorithm, including evolutionary search methods. There have been a number of approaches to handle constraints in the domain of mathematical programming including penalty functions and their variants, repair methods, use of decoders, separate treatment of constraints and objectives and hybrid methods incorporating constraint satisfaction methods. All these methods have limited success as they are problem dependent and require a number of additional inputs. Separate treatment of constraints from objectives is an interesting concept that eliminates the problem of scaling and aggregation. Section 2.2 outlines the major approaches towards constraint handling that are suitable for evolutionary computing models.

2.1. Multiple Objective: Pareto Based Approaches

A number of approaches to handle multiple objectives exist in literature [2, 3]. Some of these fundamental approaches that have been incorporated in evolutionary methods are discussed in a greater depth to provide a better understanding of their limitations.

Schaffer [4] came up with the concept of a Vector Evaluated Genetic Algorithm (VEGA) for multiobjective problems. VEGA is based on evaluation of objectives separately in different subpopulations followed by a migration scheme. Each objective is improved separately in different subpopulations and hence a large number of subpopulations are necessary in the presence of many objectives. A fitness evaluation mechanism that is based on a linear combination of the objectives will fail to generate Pareto optimal solutions for non-convex search spaces regardless of weights used.

Lis and Eiben [5] proposed a multisexual genetic algorithm that is based on mating restriction and multiparent crossover to arrive at a set of Pareto optimal solutions. The use of genders is another way to separate subpopulations for each objective. However, as the number of objectives increases, there is a need to have a number of subpopulations. Moreover the panmictic crossover is computationally less efficient as it requires a number of parents to generate a child.

Ishibuchi and Murata [6] introduced a combination of a weighted-sum-based evolutionary algorithm [EA] and a local search algorithm that maintains the set of Pareto optimal points separately. Since multiple objectives are transformed to a single measure by the use of random weights, all the drawbacks of objective aggregation exist though it eliminates the requirement of having a number of subpopulations as in the above two methods.

Another variation of the weighted approach is the weighted-sum-scalarization method proposed by Hajela and Lin [7]. It uses a variable set of weights to arrive at a set of solutions. Additional parameters are required for sharing and mating restrictions that may not be easy to provide.

Fonseca and Flemming [8] introduced Multiobjective Genetic Algorithm (MOGA) where the ranking of a solution is based on dominance. At the end of the ranking process there are a number of solutions with the same rank and the use of blocked fitness assignment results in a large selection pressure and premature convergence. To distribute the points evenly over the Pareto optimal region, the methodology uses a sharing mechanism in the objective function domain. As sharing is performed in the objective function space, MOGA may not be able to find solutions to problems where different Pareto optimal points correspond to the same objective function value. The algorithm is discussed in the context of unconstrained multi-objective problems.

A Niched Pareto Genetic Algorithm was proposed by Horn *et al.* [9] and it combines tournament selection and Pareto dominance. The nondominance of an individual is computed by comparing an individual with a randomly chosen population of size t_{dom}. The success of the algorithm is largely dependent on the size of t_{dom} as a small size will result in a few nondominated points in the population whereas a large size will result in premature convergence. The algorithm does not provide any mechanism or methodology to handle constraints.

Srinivas and Deb [10] proposed the Nondominated Sorting Genetic Algorithm (NSGA) which uses multi-layered classification based on nondominance. The methodology is computationally inferior when compared to the ranking scheme proposed by Fonseca and Flemming [8]. The method requires the sharing parameter as an additional input. Constraint handling methods are not incorporated in the algorithm.

Valenzuela-Rendon and Uresti-Charre [11] applied the concepts of NSGA to develop a nongenerational GA based on Pareto selection and fitness sharing. The problem is transformed to a bi-objective optimization problem by minimizing the domination count (weighted average of the number of individuals that have been dominated so far) and the minimization of the moving niche count (weighted average of the number of individuals that lie close, based on a sharing function). Additional inputs are required for weights and the sharing function. Constraint handling methods are not incorporated in the algorithm.

There have also been interesting attempts by Osyczka and Kundu [12, 13] to use distance based concepts for multiobjective optimization problems. The advantage of the method lies in the fact that it does not require an explicit sharing function. However, the method is highly sensitive to the values of penalty factor used to incorporate the constraints into the objective function and the performance is heavily dependent on the starting distance used to compare the quality of the solutions.

The general limitations of the methods discussed above include the following:

(i) No specific guidelines are outlined to incorporate constraints in the formulations.
(ii) Weighted aggregation of objectives or constraints will lead to problems of scaling, and proper identification of weights may be difficult.
(iii) Additional parameters for sharing, niche count *etc.*, may not be easy to provide. Details of sharing and niching mechanisms have been studied by Deb and Goldberg [14].
(iv) Large population size or a large number of subpopulations may be necessary for some of the approaches.
(v) There has been no attempt to incorporate parent-matching concepts during mating other than an attempt by Hinterding and Michalewicz [15].

Having identified the general limitations of the multiobjective methods discussed above, the proposed algorithm for multiobjective optimization is introduced in Section 3. The method is based on ranking of nondominated solutions in the objective and the constraint space and thus eliminates the problem of weighted aggregation or scaling. Parent matching concepts are incorporated during mating which results in mating between solutions that are good in either constraints or objectives with a hope to generate better solutions through collaborative pairing. Moreover, the diversification strategy incorporated in the algorithm distributes the points evenly over the Pareto space without any additional input for niche count or sharing function.

2.2. Constraint Handling

Evolutionary computation methods are essentially unconstrained search techniques that require a scalar measure of quality or fitness. There have been a number of approaches to handle constraints including rejection of infeasible solutions, penalty functions and their variants, repair methods, use of decoders, separate treatment of constraints and objectives and hybrid methods incorporating knowledge of constraint satisfaction. Michalewicz and Schoenauer [16] provides a comprehensive review on constraint handling methods. All the methods have limited success as they are problem dependent and require a number of additional inputs. Penalty functions using static, dynamic or adaptive concepts have been developed over the years. All of them still suffer from common problems of aggregation and scaling. Repair methods are based on additional function evaluations, while the decoders and special operators or constraint satisfaction methods are problem specific and cannot be used to model a generic constraint. Separate treatment of constraints and objectives is an interesting concept that eliminates the problem of scaling and aggregation.

Constraint handling using a Pareto ranking scheme is a relatively new concept having its origin in multiobjective optimization. Fonseca and Flemming [8] proposed a Pareto ranking scheme to handle multiple objectives. Jimenez and Verdegay [17] used a nondominated sorting GA [10] ranking scheme to deal with multiple objectives while a separate evaluation function was used for infeasible solutions. Surry *et al.* [18] applied a Pareto ranking scheme among constraints while fitness was used in the objective function space for the optimization of gas supply networks. Fonseca and Flemming [3] proposed a unified formulation to handle multiple constraints and objectives based on a Pareto ranking scheme. All the above attempts successfully eliminate the drawbacks of aggregation and scaling found in penalty function methods. Furthermore, they do not require any additional input and are problem independent. However, none of the above methods incorporate concepts of cooperative learning through parent matching which is expected to improve the efficiency of the algorithm. An interesting attempt to incorporate the knowledge of constraint satisfaction during mating was proposed by Hinterding and Michalewicz [15]. In an attempt to match *the beauty with the brains*, constraint matching was employed during partner selection. A single measure (sum of squares of violation) was used to compute a solution's infeasibility. The algorithm does not include any niching or diversification mechanism to ensure a uniform spread of points along the Pareto frontier for multiobjective problems. Moreover, a single aggregate measure of infeasibility fails to incorporate the knowledge of individual constraint satisfaction/violation and in addition leads to scalability and aggregation problems. There have also been attempts by Koziel and Michalewicz [19] to handle constrained optimization problems through the use of homomorphous mapping.

The evolutionary algorithm proposed in this work eliminates the drawbacks discussed above through the use of nondominance to handle both constraints and objectives. Since the constraints are handled separately, the true objective function is optimized and not any transformed evaluation function. The mating process within the proposed evolutionary algorithm incorporates the knowledge of every individual constraint satisfaction/violation and objective performance. Strategies to handle highly constrained and moderately constrained problems are outlined. Section 3 provides a detailed description of the algorithm.

3. PROPOSED ALGORITHM

A general constrained multiobjective optimization problem (in the minimization sense) is presented as:

$$
\begin{aligned}
\text{Minimize} \quad & \mathbf{f} = [f_1(\mathbf{x}) f_2(\mathbf{x}) \dots f_k(\mathbf{x})] \\
\text{subject to} \quad & g_i(\mathbf{x}) \geq a_i \quad i = 1, 2, \dots, q \\
& h_j(\mathbf{x}) = b_j \quad j = 1, 2, \dots, r
\end{aligned}
$$

where $\mathbf{f}$ is a vector of k objectives and $\mathbf{x}=[x_1, x_2, \ldots, x_n]$ is the vector of n design variables.

The OBJECTIVE matrix for a population of M solutions assumes the form

$$\text{OBJECTIVE} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1k} \\ f_{21} & f_{22} & \cdots & f_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ f_{M1} & f_{M2} & \cdots & f_{Mk} \end{bmatrix}$$

It is common practice to transform the equality constraints (with a tolerance δ) to a set of inequalities and use a unified formulation for all constraints:

$h_j(\mathbf{x}) \leq b_j + \delta$ which is the same as $-h_j(\mathbf{x}) \geq -b_j - \delta$ and

$$h_j(\mathbf{x}) \geq b_j - \delta$$

Thus r equality constraints will give rise to $2r$ inequalities, and the total number of inequalities for the problem is denoted by s, where $s = q + 2r$.

For each solution, $\mathbf{c}$ denotes the constraint satisfaction vector given by $\mathbf{c} = [c_1, c_2, \ldots, c_s]$ where

$$c_i = \begin{cases} 0 & \text{if the } i\text{th constraint is satisfied} \\ & \quad i = 1, 2, \ldots, s \\ a_i - g_i(\mathbf{x}) & \text{if the } i\text{th constraint is violated} \\ & \quad i = 1, 2, \ldots, q \\ b_i - \delta - h_i(\mathbf{x}) & \text{if the } i\text{th constraint is violated} \\ & \quad i = q+1, q+2, \ldots, q+r \\ -b_i - \delta + h_i(\mathbf{x}) & \text{if the } i\text{th constraint is violated} \\ & \quad i = q+r+1, q+r+2, \ldots, s \end{cases}$$

For the above c_i, $c_i = 0$ indicates that the ith constraint is satisfied, whereas $c_i > 0$ indicates the violation of the constraint.

The CONSTRAINT matrix for a population of M solutions assumes the form

$$\text{CONSTRAINT} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1s} \\ c_{21} & c_{22} & \cdots & c_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ c_{M1} & c_{M2} & \cdots & c_{Ms} \end{bmatrix}$$

A COMBINED matrix that is a combination of objective and constraint matrix assumes the form

$$\text{COMBINED} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1k} & c_{11} & c_{12} & \cdots & c_{1s} \\ f_{21} & f_{22} & \cdots & f_{2k} & c_{21} & c_{22} & \cdots & c_{2s} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{M1} & f_{M2} & \cdots & f_{Mk} & c_{M1} & c_{M2} & \cdots & c_{Ms} \end{bmatrix}$$

3.1. Pareto Ranking

From a population of M solutions, all nondominated solutions are assigned a rank of 1. The rank 1 individuals are removed from the population and the new set of nondominated solutions is assigned a rank of 2. The process is continued until every solution in the population is assigned a rank. Rank = 1 in any of the objective, constraint or combined matrices indicate that the solution is nondominated.

The Pareto rank of each solution in the population is computed individually in the OBJECTIVE, CONSTRAINT and COMBINED matrix and are stored in vectors RankObj, RankCon and RankCom respectively.

Having described the general formulation of the constraint and the objective function matrices and the concept of Pareto ranking, the pseudo code of the algorithm is introduced.

3.2. Algorithm

Initialize M solutions

Do {

Compute Pareto Ranking based on OBJECTIVE matrix to yield a vector RankObj

Compute Pareto Ranking based on CONSTRAINT matrix to yield a vector RankCon

Compute Pareto Ranking based on COMBINED matrix to yield a vector RankCom

Multiobjective Optimization Select individuals from the population if (RankCom = 1) and (Feasible) and put them into New Population

To generate the remaining members of the New Population
Do {
    Select an individual A and its partner
    Mate A with its partner
    Put parents and children into New Population
} while the population is not full.
Remove duplicate points in parametric space and shrink population
} while the maximum number of generations is not attained.

3.3. Initialization

The initialization is based on a random generation of M starting solutions using a uniform random number generator and the variable bounds (side constraints). A solution $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ is generated as follows:

$$x_i = (x_{i,\text{upper bound}} - x_{i,\text{lower bound}}) * R + x_{i,\text{lower bound}}$$

where $x_{i,\text{lower bound}}$ and $x_{i,\text{upper bound}}$ are the lower and upper bounds of the ith variable and R is a random number between 0 and 1.

3.4. Selection Probability

Individuals with RankCom = 1 corresponds to elite individuals and hence they are carried over to the next generation if they are feasible. To fill up the remaining members in the next generation, individuals are selected and subsequently mated. The probability of selection of an individual is based on the vectors RankObj, RankCon or RankCom and denoted as ProbObj, ProbCon and ProbCom respectively. As an example, the vector ProbObj is computed as follows:

The vector RankObj with element values varying from 1 to $P_{\max}$ is transformed to a fitness vector FitObj with elements varying from $P_{\max}$ to 1 using a linear scaling ($P_{\max}$ denotes the rank of the worst solution). The probability of selection ProbObj of an individual is then computed based on this fitness vector FitObj using the roulette wheel selection scheme. The process ensures that solutions that are fitter have a higher probability of being selected.

3.5. Choosing a Partner for Mating

A mating is performed between a solution A and its partner (B or C). The process of partner selection is dependent on the type of the constrained problem. Problems are classified into the following:

(i) Unconstrained problem (Objective–Objective Mating)
(ii) Moderately constrained problem (Objective–Constraint Mating)
(iii) Highly constrained problem (Constraint–Constraint Mating)

For an unconstrained problem, the selection of A, B and C is based on ProbObj. For a moderately constrained problem, selection of A is based on ProbObj while the selection of B and C is based on ProbCon. Such a mating between solutions that are *good* in objective function with that of solutions that are *good* in constraint satisfaction is analogous to mating between the *beauty and the brains*. For a highly constrained problem, selection of A, B and C is based on ProbCon. Since finding a feasible solution is quite difficult for highly constrained problems, the selection of mating partners is based on the solution's ability towards constraint satisfaction. The process of partner selection for a moderately constrained problem is outlined below for a greater understanding of the selection process.

3.5.1. Moderately Constrained: Partner Selection

Select first individual A based on ProbObj
Select potential mating candidate B based on ProbCon
Select potential mating candidate C based on ProbCon
Partner of A is either B or C, depending upon Condition 1, 2 or 3.

CONDITION 1 *If B and C are both feasible*
If RankObj_B < RankObj_C, then partner is B.
If RankObj_C < RankObj_B, then partner is C.
If RankObj_B = RankObj_C, then choose the one with the minimum adaptive niche count (explained in Section 3.6).
where, RankObj_B denotes the rank of solution B in the vector RankObj.

CONDITION 2 *If B and C are both infeasible*
If RankCon_B < RankCon_C, then the partner is B.

If RankCon_C < RankCon_B, then the partner is C.

If RankCon_B = RankCon_C, then choose the one with minimum overlapping constraint satisfaction with A (explained in Section 3.7).

CONDITION 3 *If one is feasible and the other is not.*

If B is feasible, then the partner is B.

If C is feasible, then the partner is C.

3.6. Mating

Every mating generates 3 additional solutions unlike the conventional process of crossover which generates two children. Out of the three solutions generated, one is generated by uniform crossover between A and its partner while the other two are generated using random mix and move. Every mating will place 2 parents and 3 additional solutions in the new generation. The process of random mix and move is as follows:

For $i = 1: n$

Action 1 Randomly pick A or its partner and denote it as base.
Action 2 Randomly pick a number Q for direction (< 0.5 is negative, positive otherwise)
Action 3 Randomly pick a number R between 0 and 1.

CONDITION 1 $x_{i,A} < x_{i,partner}$

A is base and $Q < 0.5$ *New variable* $= x_{i,A} - R(x_{i,A} - x_{i,lower\ bound})$

A is base and $Q \geq 0.5$ *New variable* $= x_{i,A} + R(x_{i,partner} - x_{i,A})$

Partner is base and $Q < 0.5$ *New variable* $= x_{i,partner} - R(x_{i,partner} - x_{i,A})$

Partner is base and $Q \geq 0.5$ *New variable* $= x_{i,partner} + R(x_{i,upper\ bound} - x_{i,partner})$

CONDITION 2 $x_{i,A} > x_{i,partner}$

A is base and $Q < 0.5$ *New variable* $= x_{i,A} - R(x_{i,A} - x_{i,partner})$

A is base and $Q \geq 0.5$ *New variable* $= x_{i,A} + R(x_{i,upper\ bound} - x_{i,A})$

Partner is base and $Q < 0.5$ *New variable* $= x_{i,partner} - R(x_{i,partner} - x_{i,lower\ bound})$

Partner is base and $Q \geq 0.5$ *New variable* $= x_{i,partner} + R(x_{i,A} - x_{i,partner})$

CONDITION 3 $x_{i,A} = x_{i,partner}$

$Q < 0.9$ *New variable* $= x_{i,A}$

$Q \geq 0.9$ *New variable* $= x_{i,A} + R(x_{i,upper\ bound} - x_{i,lower\ bound})$

End

The process of random mix and move will ensure that any feasible variable value can be generated even if it does not exist in either A or its partner. Generation of a large number of initial solutions to maintain all possible variable values is not considered favorable as those solutions are generated without any knowledge of the search process and adds on to a computational overhead. The proposed method as illustrated can be used with a relatively small population size as the process of generating solutions comes along with random mix and move.

3.7. Adaptive Niche Count

Adaptive niche count of a solution is the number of solutions in that population which are within the average distance metric and is computed as follows:

For $i = 1$: M

Compute the Euclidean distances between it and all other $M - 1$ solutions

Compute the average Euclidean distance

Count the number of solutions that are within the average distance

End

A solution with a small niche count as compared to another physically means that there are few solutions in the neighborhood of it. Such solutions are preferred over others and is the diversification strategy used in the algorithm. Niche count evaluation is computationally expensive. However, an evaluation of niche count may be worthwhile for problems where the evaluations of objective or constraints are themselves expensive and there is a need to make full use of such computed information to avoid parametric clustering. The proposed method does not require any additional input parameters, which also makes it easier to use.

3.8. Non-overlapping Constraint Satisfaction

The strategy is based on the philosophy that a solution is allowed to mate with another if one complements the other towards constraint satisfaction. Such a mating between the *beauty* and the *brains* is incorporated with a hope of generating solutions with better constraint satisfaction. The concept of non-overlapping constraint satisfaction is incorporated as follows:

With reference to the CONSTRAINT matrix discussed earlier, each of the solutions A, B and C has an associated constraint satisfaction vector $\mathbf{c_A}$, $\mathbf{c_B}$ and $\mathbf{c_C}$ respectively.

The sets $\{S_A\}$, $\{S_B\}$ and $\{S_C\}$ denote the set of constraints satisfied by solution A, B and C respectively. The selection of either B or C as the partner of A is based on the following condition:

If $(\{S_A\} \cap \{S_B\}) > (\{S_A\} \cap \{S_C\})$ then the partner is C.
If $(\{S_A\} \cap \{S_B\}) < (\{S_A\} \cap \{S_C\})$ then the partner is B.
If $(\{S_A\} \cap \{S_B\}) = (\{S_A\} \cap \{S_C\})$ then Partner is randomly chosen between B and C.

3.9. Population Shrinking

After each new population is full, a screening is done to remove identical points in the parametric (variable) space to give room for new and different solutions. For problems with continuous design variables, shrinking is not necessary as in real value problems, the probability of two solutions with identical parameter values is nearly zero. However, for problems with discrete and integer variables, shrinking is implemented to remove identical solutions.

4. RESULTS AND DISCUSSION

Four examples have been selected from different engineering fields to illustrate the performance of the proposed algorithm. The first two examples are related to bi-objective truss design problems. The third example is an optimal design of a vibrating platform while the last example is related to a water resource-planning problem involving five objectives.

4.1. Two Bar Truss

This problem is taken from Rao [20]. The same problem has also been solved by Cheng and Li [21]. Figure 1 illustrates the problem where structural weight (f_1) and the displacement (f_2) of joint 3 are to be minimized subject to the stress constraints on the members. The cross-sectional area A of the members and the half-distance x between the joints 1 and 2 are to be varied while keeping the truss geometry symmetric.

$$\text{Minimize} \quad f_1(\mathbf{x}) = 2\rho h x_2\sqrt{1+x_1^2}$$

$$f_2(\mathbf{x}) = \frac{Ph(1+x_1^2)^{1.5}(1+x_1^4)^{0.5}}{2\sqrt{2}Ex_1^2x_2}$$

$$\text{Subject to} \quad g_1(\mathbf{x}) = \frac{P(1+x_1)(1+x_1^2)^{0.5}}{2\sqrt{2}x_1x_2} - \sigma_0 \le 0.0$$

$$g_2(\mathbf{x}) = \frac{P(1-x_1)(1+x_1^2)^{0.5}}{2\sqrt{2}x_1x_2} - \sigma_0 \le 0.0$$

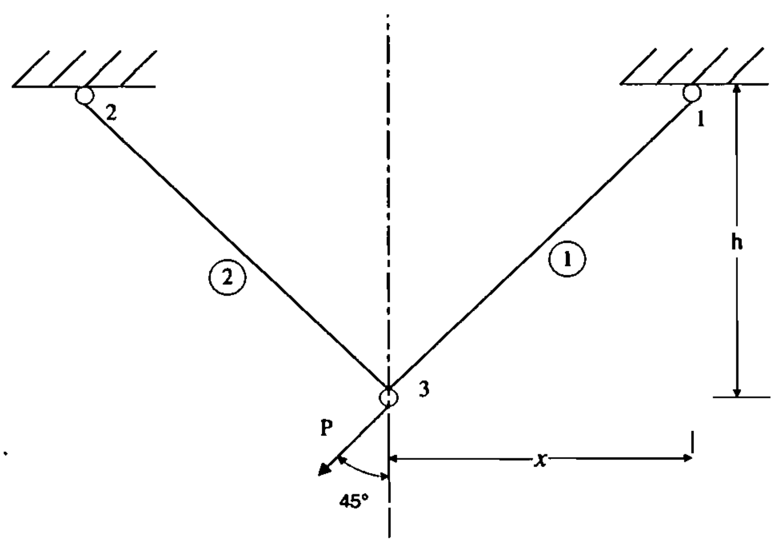

FIGURE 1 Two bar truss.

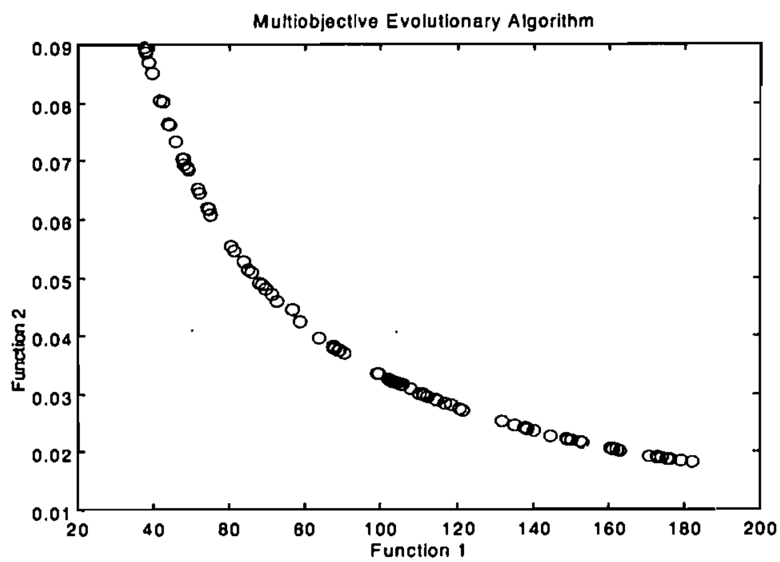

FIGURE 2 The Pareto optimal front (two bar truss).

where $\rho = 0.283\,\mathrm{lb/in^3}$, $h = 100\,\mathrm{in}$, $P = 10^4\,\mathrm{lb}$, $E = 3.0\mathrm{E}07\,\mathrm{lb/in^2}$, $\sigma_0 = 2.0\mathrm{E}04\,\mathrm{lb/in^2}$, $A_{\min} = 1.0\,\mathrm{in^2}$ (minimum cross-sectional area of the members). The transformed variables $x_1 = x/h$, $x_2 = A/A_{\min}$ are used in the formulation with the following bounds: $0.10 \leq x_1 \leq 2.25$, $0.50 \leq x_2 \leq 2.50$.

The Pareto optimal front is shown in Figure 2. It has 95 pareto optimal points obtained after 793 function evaluations. It can be observed that the points are evenly spread along the Pareto frontier.

4.2. Four Bar Truss

This problem was originally solved by Stadler [22], and has also been solved by Cheng and Li [21]. Figure 3 illustrates the problem where the structural volume (f_1) and the displacement (f_2) at joint (2) are to be minimized subject to the stress constraints on the members. The cross sectional areas of the members 1, 2, 3 and 4 are the design variables represented as x_1, x_2, x_3 and x_4 respectively. Figure 4 presents the

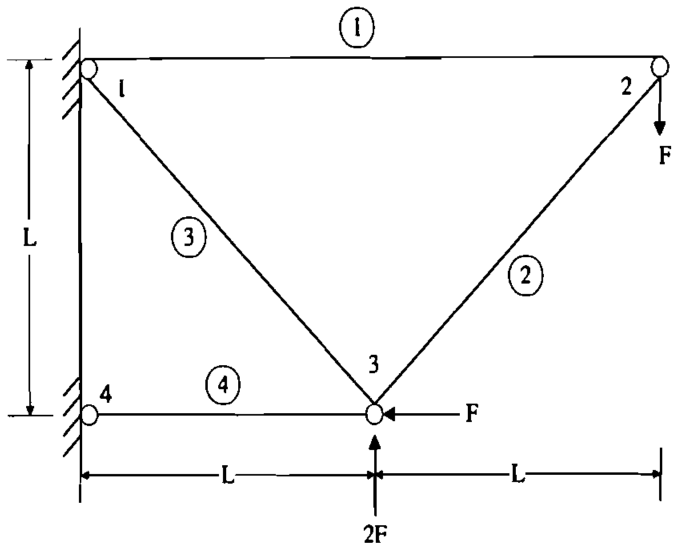

FIGURE 3 Four bar truss.

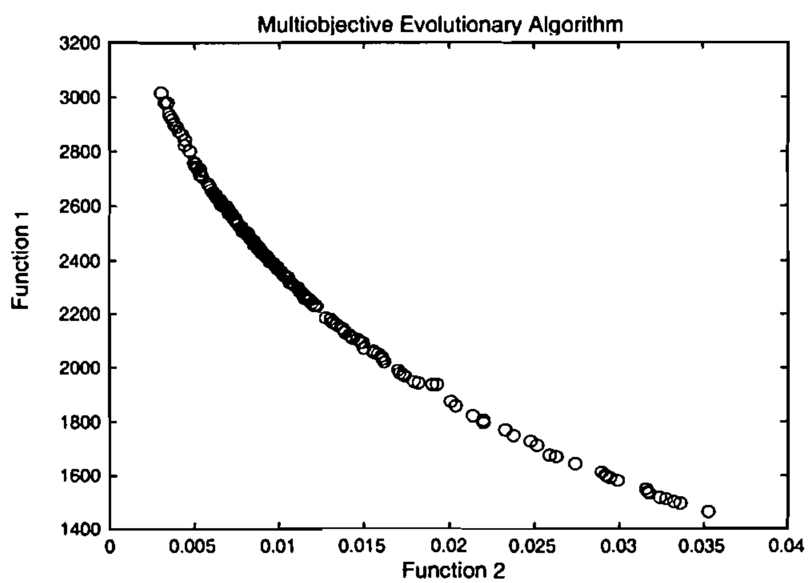

FIGURE 4 The Pareto optimal front (four bar truss).

Pareto optimal front for the problem.

$$
\begin{aligned}
\text{Minimize} \quad & f_1(\mathbf{x}) = L(2x_1 + \sqrt{2}x_2 + \sqrt{2}x_3 + x_4) \\
\text{Minimize} \quad & f_2(\mathbf{x}) = \frac{FL}{E}\left(\frac{2}{x_1} + \frac{2\sqrt{2}}{x_2} - \frac{2\sqrt{2}}{x_3} + \frac{2}{x_4}\right) \\
\text{Subject to:} \quad & F/\sigma \leq x_1 \leq 3F/\sigma \\
& \sqrt{2}F/\sigma \leq x_2 \leq 3F/\sigma \\
& \sqrt{2}F/\sigma \leq x_3 \leq 3F/\sigma \\
& F/\sigma \leq x_4 \leq 3F/\sigma
\end{aligned}
$$

where $F = 10\,\text{kN}$, $E = 2.00\text{E}05\,\text{kN/cm}^2$, $L = 200\,\text{cm}$ and $\sigma = 10\,\text{kN/cm}^2$.

Cheng and Li [21] presented the Pareto optimal curve between (2250, 0.0120) and (1700, 0.0250). The present results in Figure 4 show the extended Pareto optimal curve between (3000, 0.0025) and (1450, 0.035). The algorithm resulted in 191 Pareto optimal points after 3764 function evaluations, spreading evenly over the Pareto front.

4.3. Vibration Platform Design

This example is a modification of the vibrating platform design problem by Messac [23]. The problem was originally formulated as a single objective maximization of the fundamental frequency subject to a constraint on cost. In the present formulation, cost is considered as a second objective. The geometry and the material are to be synthesized

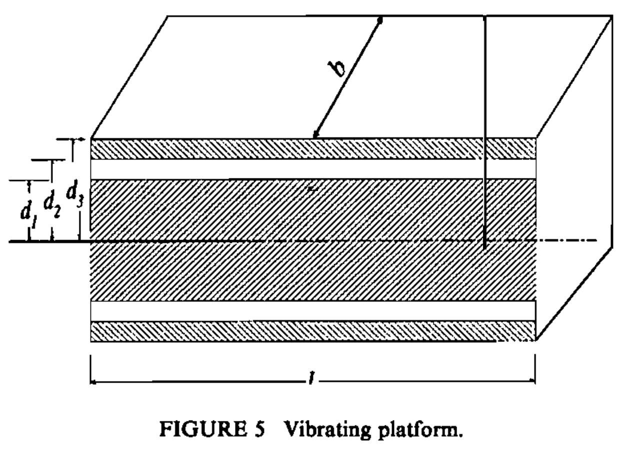

FIGURE 5 Vibrating platform.

during the process of design. Figure 5 illustrates the problem while Figure 6 presents the Pareto optimal front for the problem.

$$
\begin{aligned}
&\text{Maximize} \quad f_1(\mathbf{x}) = (\pi/(2L^2))(EI/\mu)^{1/2}\\
&\text{Minimize} \quad f_2(\mathbf{x}) = 2b[c_{M1}d_1 + c_{M2}(d_2 - d_1) + c_{M3}(d_3 - d_2)]\\
&\text{where} \quad \mathbf{x} = [L, b, d_1, d_2, d_3, M_i]\\
&\qquad \mu = 2b[\rho_{M1}d_1 + \rho_{M2}(d_2 - d_1) + \rho_{M3}(d_3 - d_2)]\\
&\qquad EI = (2b/3)[E_{M1}d_1^3 + E_{M2}(d_2^3 - d_1^3) + E_{M3}(d_3^3 - d_2^3)]\\
&\text{Subject to} \quad g_1(\mathbf{x}) = \mu L - 2800 \leq 0.0\\
&\qquad g_2(\mathbf{x}) = d_2 - d_1 \geq 0.00001\\
&\qquad g_3(\mathbf{x}) = d_3 - d_2 \geq 0.00001\\
&\qquad g_4(\mathbf{x}) = d_2 - d_1 \leq 0.01\\
&\qquad g_5(\mathbf{x}) = d_3 - d_2 \leq 0.01
\end{aligned}
$$

where $3.00 \leq L \leq 6.00$, $0.35 \leq b \leq 0.50$, $0.01 \leq d_1 \leq 0.60$, $0.01 \leq d_2 \leq 0.60$, $0.01 \leq d_3 \leq 0.60$.

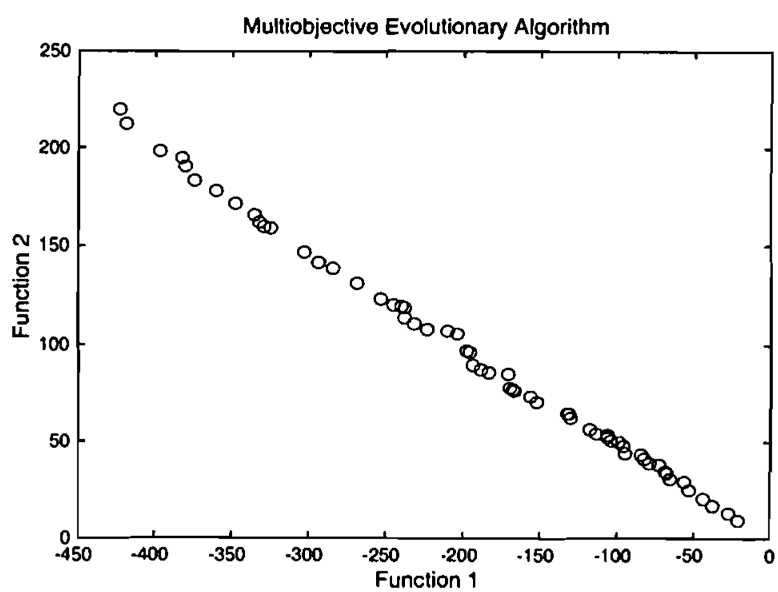

FIGURE 6 The Pareto optimal front (vibrating platform).

TABLE I Properties of material types for platform design

M_i (*Material type*)	ρ (kg/m^3)	E (N/m^2)	c (*$/volume*)
I	2,770	70.0 E09	1,500
II	100	1.6 E09	500
III	7,780	200.0 E09	800

The combinatorial variables M_i refers to the material type for layer i ($i = 1, 2, 3$). The mass density (ρ), Young's modulus (E) and the cost per unit volume (c) of each material type I, II and III are presented in Table I.

The results presented in Figure 6 indicate a wide variety of Pareto optimal solutions. The algorithm resulted in 60 Pareto optimal points in 3052 function evaluations. Some representative solutions are presented in Table II.

4.4. Water Resource Planning

The problem involves optimal planning for a storm drainage system in an urban area, described originally by Musselman and Talavage [24] and also attempted by Cheng and Li [21] The variables are x_1 = local detention storage capacity, x_2 = maximum treatment rate and x_3 = the maximum allowable overflow rate. The objective functions to be minimized are f_1 = drainage network cost, f_2 = storage facility cost, f_3 = treatment facility cost, f_4 = expected flood damage cost, and f_5 = expected economic loss due to flood. The detailed description of the problem and constraints can be obtained from Musselman and Talavage [24].

$$\text{Minimize} \quad f_1(\mathbf{x}) = 106780.37(x_2 + x_3) + 61704.67$$

$$f_2(\mathbf{x}) = 3000x_1$$

$$f_3(\mathbf{x}) = \frac{(305700)2289x_2}{[(0.06)2289]^{0.65}}$$

$$f_4(\mathbf{x}) = (250)2289\exp(-39.75x_2 + 9.9x_3 + 2.74)$$

$$f_5(\mathbf{x}) = 25\left(\frac{1.39}{x_1x_2} + 4940x_3 - 80\right)$$

TABLE II Some representative solutions of platform design

L	b	d_1	d_2	d_3	M_1	M_2	M_3	f_1	f_2
3.00162	0.35211	0.06450	0.06527	0.06859	II	I	III	−54.38	25.39
3.01181	0.35068	0.12621	0.13318	0.13754	II	III	I	−107.01	52.76
3.00551	0.35280	0.09046	0.09201	0.09383	II	III	I	−68.68	34.71
3.00523	0.35197	0.32302	0.32641	0.33303	II	I	III	−246.44	121.00

$$\text{Subject to} \quad g_1(\mathbf{x}) = \frac{0.00139}{x_1x_2} + 4.94x_3 - 0.08 \leq 1.00$$

$$g_2(\mathbf{x}) = \frac{0.000306}{x_1x_2} + 1.082x_3 - 0.0986 \leq 1.00$$

$$g_3(\mathbf{x}) = \frac{12.307}{x_1x_2} + 49408.24x_3 + 4051.02 \leq 50000.00$$

$$g_4(\mathbf{x}) = \frac{2.098}{x_1x_2} + 8046.33x_3 - 696.71 \leq 16000.00$$

$$g_5(\mathbf{x}) = \frac{2.138}{x_1x_2} + 7883.39x_3 - 705.04 \leq 10000.00$$

$$g_6(\mathbf{x}) = \frac{0.417}{x_1x_2} + 1721.26x_3 - 136.54 \leq 2000.00$$

$$g_7(\mathbf{x}) = \frac{0.164}{x_1x_2} + 631.13x_3 - 54.48 \leq 550.00$$

where $0.01 \leq x_1 \leq 0.45$, $0.01 \leq x_2 \leq 0.10$ and $0.01 \leq x_3 \leq 0.10$.

The results are scaled [$f_1/80000$, $f_2/1500$, $f_3/3000000$, $f_4/6000000$, $f_5/8000$] and the maximum and minimum values of the scaled objectives are presented in Figure 7. The lines indicate combinations of objective values for some representative solutions while Table III lists a few representative solutions.

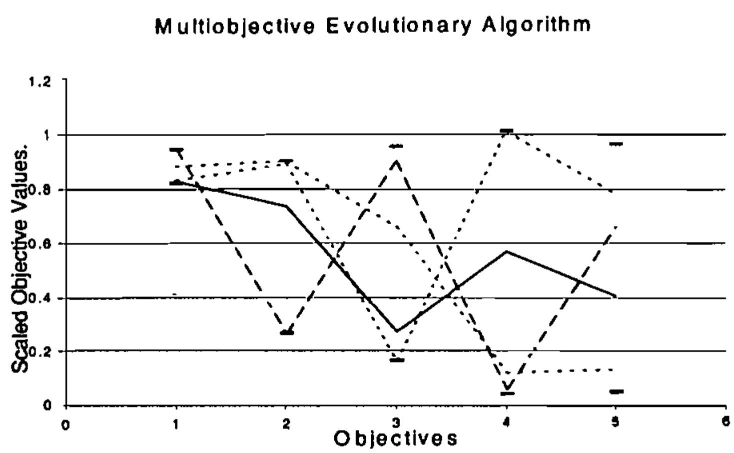

FIGURE 7 The Pareto optimal space (range of objective values) (water resource planning).

TABLE III Some representative solutions for water resource planning

x_1	x_2	x_3	f_1	f_2	f_3	f_4	f_5
0.1312	0.0942	0.0354	75550.6	393.59	268857	297434	5188.67
0.3663	0.0280	0.0142	66203.1	1099.03	797974	335489	3141.07
0.4444	0.0166	0.0280	66465.1	1333.30	474106	603903	6159.86
0.4499	0.0687	0.0149	70633.7	1349.74	196057	669173	965.80

5. SUMMARY AND CONCLUSIONS

The evolutionary algorithm for multiobjective design optimization is described in this paper. The use of a Pareto ranking concept in both the objective and the constraint domain eliminates the problems of scaling and aggregation. Moreover, the method is problem independent, can handle any computable constraint, and in addition optimizes the true objective function and not any transformed function. The results of the algorithm for the design examples clearly illustrate the capability of the algorithm to arrive at a set of Pareto optimal designs while maintaining diversity in them. The presence of intelligent mating strategies enables the algorithm to converge at pareto optimal solutions with minimal function evaluations. Niching and shrinking are computationally complex operations, and they are used to avoid evaluating too many parametrically similar designs. It is effective for most real life problems where the objectives or constraints are themselves computationally expensive and there is a need to guide the search, making the best use of all available information. The algorithm does not require additional parameters for sharing or scaling and hence is definitely attractive for generic design optimization problems involving multiple objective and constraints as highlighted by the multidisciplinary design examples.

Acknowledgments

The authors would like to acknowledge the support for this work received from the Institute of High Performance Computing and the National Science and Technology Board (RICURF research fund EMT/98/013).

References

[1] Fonseca, C. M. and Fleming, P. J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1), 1–16.
[2] Coello Coello, C. A. (1999). A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems*, 1(3), 269–308.
[3] Fonseca, C. M. and Fleming, P. J. (1998). Multiobjective optimization and multiple constraint handling with evolutionary algorithms-Part I: A unified formulation.

IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, **28**(1), 26–37.

[4] Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In: *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms*, Lawrence Erlbaum, pp. 93–100.

[5] Lis, J. and Eiben, A. E. (1997). A multi-sexual genetic algorithm for multiobjective optimization. *Proceedings of 1997 IEEE International Conference on Evolutionary Computation*, pp. 59–64.

[6] Ishibuchi, H. and Murata, T. (1996). Multiobjective genetic local search algorithm. *Proceedings of 1996 IEEE International Conference on Evolutionary Computation*, pp. 119–124.

[7] Hajela, P. and Lin, C. Y. (1992). Genetic search strategies in multi-criterion optimal design. *Structural Optimization*, **4**, 99–107.

[8] Fonseca, C. and Flemming, P. J. (1993). Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In: Forest, S. (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms*, University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, San Mateo, California, pp. 416–423.

[9] Horn, J., Nafpolitis, N. and Goldberg, D. E. (1994). A niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Computation*, **1**, IEEE Service Center, Piscataway, NJ, 82–87.

[10] Srinivas, N. and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, **2**(3), 221–248.

[11] Valenzuela-Rendon, M. and Uresti-Charre, E. (1997). A non-generational genetic algorithm for multiobjective optimization. *Proceedings of the Seventh International Conference on Genetic Algorithms*, San Fransisco, California, pp. 658–665.

[12] Osyczka, A. and Kundu, S. (1995). A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural Optimization*, **10**, 94–99.

[13] Osyczka, A. and Kundu, S. (1989). An investigation of niche and species formation in genetic function optimization. In: David Shaffer, J. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, George Mason University, Morgan Kaufmann Publishers, San Mateo, California, pp. 42–50.

[14] Deb, K. and Goldberg, D. E. (1996). A modified distance method for multicriteria optimization using genetic algorithms. *Computers in Industrial Engineering*, **30**(**4**), 871–882.

[15] Hinterding, R. and Michalewicz, Z. (1998). Your brains and my beauty: parent matching for constrained optimisation. *Proceedings of the Fifth International Conference on Evolutionary Computation*, Anchorage, Alaska, pp. 810–815.

[16] Michalewicz, Z. and Schoenauer, M. (1996). Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, **4**(1), 1–32.

[17] Jimenez, F. and Verdegay, J. L. (1998). Constrained multiobjective optimization by evolutionary algorithms. *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems* (EIS'98), University of La Laguna, Tenerife, Spain, pp. 266–271.

[18] Surry, P., Radcliffe, N. J. and Boyd, I. (1995). A multiobjective approach to constrained optimization of gas supply networks. *Lecture Notes in Computer Science*, Springer Verlag, Sheffield, UK, pp. 166–180.

[19] Koziel, S. and Michalewicz, Z. (1999). Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, **7**(1), 19–44.

[20] Rao, S. S. (1987). Game theory approach for multiobjective structural optimization. *Computers and Structures*, **26**(1), 119–127.

[21] Cheng, F. Y. and Li, X. S. (1999). Generalized method for multiobjective engineering optimization. *Engineering Optimization*, **31**, 641 – 661.

[22] Stadler, W. and Dauer, J. (1992). Multicriteria optimization in engineering: A tutorial and survey. *Structural Optimization:Status and Future, AIAA Journal*, pp. 209 – 249.

[23] Messac, A. (1996). Physical programming: effective optimization for computational design. *AIAA Journal*, **34**, 149 – 158.

[24] Musselman, K. and Talavage, J. (1980). A trade-off cut approach to multiple objective optimization. *Operations Research*, **28**(6), 1424 – 1435.