Struct Multidisc Optim 23, 204–213 © Springer-Verlag 2002
Digital Object Identifier (DOI) 10.1007/s00158-002-0178-2

Constraint handling improvements for multiobjective genetic algorithms

A. Kurpati, S. Azarm and J. Wu

Abstract Four constraint handling improvements for Multi-Objective Genetic Algorithms (MOGA) are proposed. These improvements are made in the fitness assignment stage of a MOGA and are all based upon a "Constraint-First-Objective-Next" model. Two multiobjective design optimization examples, i.e. a speed reducer design and the design of a fleet of ships, are used to demonstrate the improvements. For both examples, it is shown that the proposed constraint handling techniques significantly improve the performance of a baseline MOGA.

Key words genetic algorithms, multiple objectives, constraint handling

1 Introduction

Real world engineering design problems have multiple objectives and constraints (Eschenauer *et al.* 1990). Often, the objectives for such problems are at least partly conflicting and the designer wants to optimize a design based on all such objectives simultaneously. In such a multiobjective design optimization problem, there is no single optimum solution to the problem. Instead, the designer obtains a set of solutions called Pareto solutions. Amongst these Pareto solutions, it is not possible to distinguish which solution is "better" over the rest of the solutions. This is because of the trade-offs that exist between various design objectives. Indeed, for any two Pareto solutions, if one solution is "better" with respect to a design objective, then it is "worse" with respect to at least one other objective. That is the reason why Pareto solutions are also called noninferior or nondominated or trade-off solutions.

In a multiobjective optimization problem, the design variables may be comprised of a mix of continuous and discrete variables. Traditionally, such problems are solved by first converting them to a series of single objective problems. Unfortunately, gradient-based optimization techniques such as DFP, BFGS, among others (Papalambros and Wilde 1988), are not applicable to these problems with such mixed variables. Stochastic techniques such as Genetic Algorithms (or GAs) (Holland 1975), simulated annealing (Van Laarhoven 1987) and tabu search (Glover and Laguna 1997) are capable of solving single objective optimization problems with mixed variables. These methods do not require computation of the derivatives to guide the optimization process. More recently, methods such as Multi-Objective Genetic Algorithms (or MOGAs) (Fonseca and Fleming 1993) have been developed that are capable of generating a Pareto solution set in a single run of the GA (as the optimizer) as opposed to solving a series of single objective optimization problems.

GAs, and thus MOGAs are essentially unconstrained optimization techniques. Hence, the way that the constraints are handled in GAs or MOGAs becomes important. A survey of the various constraint handling schemes for evolutionary algorithms can be found in the paper by Coello Coello (1999b). The most common way of handling constraints in evolutionary algorithms has been with the use of a penalty method, as shown in (1). The idea is to alter the fitness value of an individual by a penalty if it violates any of the constraints (Goldberg 1989)

$$\text{fitness}_i = f_i(x) + Q_i\,, \tag{1}$$

where the quantity fitness_i refers to the fitness of the i-th individual, $f_i(x)$ is the objective function value (to be minimized) of the i-th individual, and Q_i is a penalty function due to a constraint violation for the i-th individual.

Many approaches based on the penalty method exist for constraint handling in evolutionary algorithms (e.g. Homaifar *et al.* 1994; Joines and Houck 1994; Michalewicz and Attia 1994). Some methods define the penalty func-

Received September 14, 2000
Revised manuscript received February 15, 2001

A. Kurpati, S. Azarm and J. Wu

Department of Mechanical Engineering, University of Maryland, College Park, MD 20742–3035, USA
e-mail: azarm@eng.umd.edu

tion based on the number of constraints that are violated (Kuri and Quezada 1998). Others penalize the fitness function if the individual represents an infeasible point, without taking into account either the distance of the point from the boundary of the feasible region or the number of constraints that are violated (Narayanan and Azarm 1999). Although the use of a penalty method has been somewhat successful, the definition of a good penalty function is critical to the search capabilities of MOGAs. The main drawback of current approaches is a potentially large number of parameters that need to be defined and sensitivity of the method to these parameters. For instance, in the approach by Homaifar *et al.* (1994), the user has to define penalties according to the levels of violation for each constraint. This would imply that for problem with many constraints, the user has to define a large number of penalty parameters, which can become cumbersome. Richardson *et al.* (1989) observe that for a problem with a few constraints and a few feasible solutions, penalties that are solely functions of the number of violated constraints are not likely to produce any solutions. Some of these approaches are problem dependent. Almost all reported approaches assume that the fitness of an individual is the penalized raw objective function value. Scaling of the objective and constraint functions so that they have comparable values is not considered in these approaches to enhance the search capabilities of GAs. Most of the approaches either take into consideration the distance from the feasible region or the number of violated constraints. Both factors however can affect the performance of the GA. Coello Coello (1999b) proposes a technique that takes into account both of these factors, but the method can become cumbersome since it requires working with two populations of individuals. Better constraint handling schemes should enable the designer to obtain better solutions faster and characterize the Pareto frontier better. Such solutions also would give the designer a better idea of the design choices. Therefore, in this paper, several new constraint handling techniques specifically for MOGAs are proposed.

The rest of the paper is organized as follows. In Sect. 2, some basic concepts in multiobjective optimization are briefly reviewed to make the paper self-contained. In Sect. 3, an overview of different multiobjective GA-based optimization approaches, particularly MOGAs, is given. Four new constraint handling techniques which can be used in MOGAs are introduced in Sect. 4. Two engineering design examples are provided in Sect. 5 to demonstrate applications of the proposed constraint handling techniques. Finally, the paper is concluded with the remarks in Sect. 6.

2
Multiobjective optimization: basic concepts

A general constrained multiobjective optimization problem can be defined as in (2)

$$\text{minimize } \mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}), \ldots, f_i(\mathbf{x}), \ldots, f_m(\mathbf{x})\}$$

$$\text{subject to } \mathbf{x} \in D$$

$$D = \{\mathbf{x} : g_j(\mathbf{x}) \le 0\,,$$

$$j = 1, \ldots, J\,, \quad h_k(\mathbf{x}) = 0\,, \quad k = 1, \ldots, K\}\,, \tag{2}$$

where $\mathbf{x}$ is an $n \times 1$ design variable vector, $\mathbf{f}(\mathbf{x})$ is an $m \times 1$ vector of design objectives that are at least partly conflicting, is the j-th inequality constraint and is the k-th equality constraint. The set of design vectors that satisfies all equality and inequality constraints constitutes the feasible domain D.

Mathematically, a design solution $x^* \in D$ is said to be *Pareto optimal* if there does not exist another solution $\mathbf{x} \in D$ such that $f_i(\mathbf{x}) \le f_i(\mathbf{x}^*)$ for all $i = 1, \ldots, m$ with strict inequality for at least one i. Any other feasible solution $\mathbf{x} \in D$ with $f_i(\mathbf{x}^*) \le f_i(\mathbf{x})$ for all $i = 1, \ldots, m$, is an *inferior* solution. If each of the objective functions in (2) is individually minimized subject to the constraints defining the feasible domain D, then the components of an ideal vector (or an ideal point) $\{f_i^*, \ldots f_m^*\}$ are obtained. The *ideal point* is often used as a reference point and is the best solution that can be achieved. However, it is extremely unlikely that any optimized solution for (2) achieves an ideal point. In a minimization problem of (2), the ideal point provides a lower bound of the Pareto optimal set. In contrast, an upper bound of the Pareto set defines the components of a *nadir point*. The nadir point is given by $\{f_{i*}, \ldots, f_{*m}\}$. A *good (bad)* value for the i-th objective, i.e. f_{good} (F_{bad}), is an estimate of the i-th component of the ideal (nadir) point, f_i^* (f_{i*}).

It is common practice to scale the objective functions using f_{good} and f_{bad} using (3), so that all objectives are of the same order of magnitude and also convert the problem to a minimization type

$$f_{\text{scaled}} = \frac{(f - f_{\text{good}})}{(f_{\text{bad}} - f_{\text{good}})}\,. \tag{3}$$

This is because, prior to the scaling of the objectives using (3), if an objective is to be maximized (minimized), then f_{good} would be greater (smaller) than f_{bad}. Then, the scaled objective function values (f_{scaled}) lie in a range of [0,1], and the smaller the value of the scaled objective function, the better the solution is.

3
Multi-Objective Genetic Algorithm: MOGA

As stated earlier, when discreteness is involved in the variables of multiobjective optimization problems, GA-

based optimization approaches can be used to obtain a discrete representation of the Pareto solutions. The idea behind these techniques is to exploit the fact that GAs work with a population of candidate solutions and this population should be evolved to find as many Pareto solutions as possible rather than just one solution. There have been different approaches to incorporate multiple objectives into GAs, such as Vector Evaluated Genetic Algorithm or VEGA (Schaffer 1985), Nondominated Sorting Genetic Algorithm or NSGA (Srinivas and Deb 1994), Multi-Objective Genetic Algorithm or MOGA (Fonseca and Fleming 1993), among others. Coello Coello (1999a) gives a comprehensive survey of the various techniques for multiobjective optimization based evolutionary methods. Some methods (like VEGA) are non-Pareto based while others such as NSGA and MOGA are Pareto based. Narayanan and Azarm (1999) made some improvements to the MOGA proposed by Fonseca and Fleming (1993).

Figure 1 gives an overview of the MOGA by Narayanan and Azarm (1999). The algorithm initiates in the same way as in a conventional GA, with the generation of an initial population. For each generation, the dominant value of an individual or point in the population is calculated as follows. For a set of points in the objective space, $P = (p_1, \ldots p_{np}^-)$, the dominant value of a point p_k ($p_k \in P$) is defined as the number of all other points in the set P that dominate p_k. For example, if n points in the set P dominate the point p_k, then the dominant value of the point p_k is quantified as n. For the initial population, the individuals with zero dominant value are identified. These individuals are called the noninferior individuals. Note that, while these individuals are noninferior for the current population, they are most likely non-Pareto for the problem in an absolute sense. These noninferior individuals are given the highest rank in the current population. With the highest probability, these noninferior individuals will become parents to produce offspring, and then the process is repeated. As such, the population is gradually improved as it approaches the final population and the corresponding Pareto set for the problem.

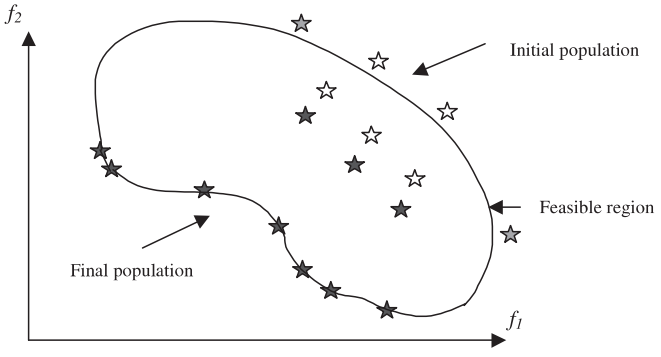

Fig. 1 Graphical illustration of MOGA

4 Constraint handling improvements for MOGA

In this section, some improvements to the way constraints are handled in a MOGA, over and beyond the methods in the literature, are discussed. The proposed improvements are used during the fitness assignment stage of the MOGA. Each proposed improvement is given an acronym. For instance, CH-NA represents the baseline constraint handling approach by Narayanan and Azarm (1999). CH-I1 stands for the proposed constraint handling improvement 1, and so on.

4.1 Assumptions

The proposed improvements are based on the guidelines obtained from numerical studies by previous researchers (Michalewicz and Attia 1994; Coello Coello 1999b) and are based on the following assumptions.

(i) In a given population, feasible solutions are preferred over infeasible solutions, i.e. feasible solutions should have a better rank than the infeasible ones.
(ii) The amount of infeasibility (or the extent of constraint violation) is an important piece of information and should not be ignored while handling constraints.
(iii) The number of violated constraints is also an important piece of information and should be taken into consideration while handling constraints.

4.2 First improvement: CH-I1

All of the proposed improvements are over and beyond the constraint handling approach of CH-NA that was reported in a MOGA implementation by Narayanan and Azarm (1999).

In CH-NA, constraints were handled in the fitness assignment stage. The following procedure was used in CH-NA to assign fitness to various individuals in a population.

Step 1. Evaluate the objective functions for every individual.
Step 2. Identify the noninferior individuals in the current population.
Step 3. Assign a low (bad) fitness value to all inferior individuals.
Step 4. Evaluate the constraints for all noninferior individuals.
Step 5. Assign a high (good) fitness value for feasible noninferior individuals.
Step 6. Assign a low (bad) fitness value for infeasible noninferior individuals.

This approach would reduce the computation cost involved, if the evaluation of constraints is computationally expensive and conducted separately from the evaluation of objectives [as shown in the (**f**)(**g**) model of Fig. 2]. This is because the constraints are not evaluated for all individuals in the population. Essentially, this approach says "**O**bjectives **F**irst **C**onstraints **N**ext" and it is referred to as the OFCN model hereafter in the paper.

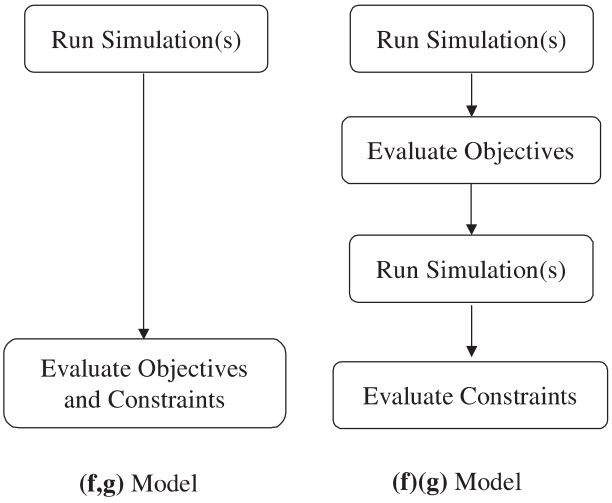

Fig. 2 Order of calculations of objective and constraint functions in two different simulation models

However, there are many real-world problems for which the computation cost involved in the evaluation of the objectives is the same as the evaluation of both objectives and constraints [i.e. a $(\mathbf{f}, \mathbf{g})$ model]. As shown in Fig. 2, in a $(\mathbf{f}, \mathbf{g})$ model, typically, the design objectives and constraints are computed by running one or more simulations. In contrast, in a (**f**)(**g**) model, one or more simulations are run to evaluate the design objectives, followed by one or more simulation runs to evaluate the constraints.

If the evaluation of the objectives and constraints follows the $(\mathbf{f}, \mathbf{g})$ model, then the evaluation of the constraints for every individual in the population does not add to the computational cost. This fact has been exploited in all of the proposed improvements. In the first improvement, CH-I1, the following procedure is used to handle constraints during the fitness assignment stage of the MOGA.

Step 1. Evaluate the constraints for every individual.
Step 2. Identify feasible and infeasible individuals in the current population.
Step 3. Assign a high (i.e. bad) rank to all infeasible individuals (i.e. rank, r, is equal to: "0.95× population size").
Step 4. Assign a moderate rank to all feasible individuals (i.e. r is equal to: "0.5× population size").
Step 5. Evaluate the objective function for all feasible individuals.
Step 6. Identify the noninferior individuals amongst feasible individuals.
Step 7. Assign a low (i.e. good) rank to feasible noninferior individuals (i.e. r is equal to: 1).
Step 8. Obtain fitness values for all individuals using (4)

$$\text{fitness}_i = C_{\max} - (C_{\max} - C_{\min})\,(r-1)/(M-1), \tag{4}$$

where $C_{\max} = 1.2$; $C_{\min} = 0.8$; r is the rank of the individual; M = population size.

The 8-step procedure listed above is based on the assumption (i) of Sect. 4.1. Essentially, the proposed approach says "Constraints First Objectives Next" and it is referred to as the CFON model hereafter in this paper. A constraint handling approach based upon CFON should be more robust than one based upon OFCN, as explained next.

CH-I1 would identify the good solutions irrespective of how the points are distributed in the objective space. This is explained with the aid of Fig. 3 which shows the distribution of individuals for a population in the objective space. The true Pareto frontier (which a MOGA seeks to detect) is shown with a dotted line in Fig. 3. Figure 3a shows the individuals that are identified as noninferior (by shaded spots) in the current population based

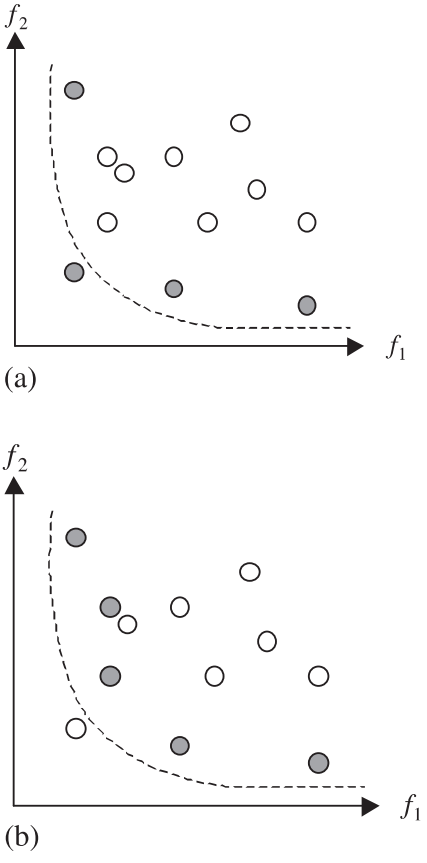

Fig. 3 (a) OFCN approach, and (b) CFON approach

on the OFCN model. Of course, the lone infeasible individual lying on one side of the true Pareto frontier would get a bad rank upon the evaluation of constraints. But, only three individuals get a good rank based on the OFCN model. All other individuals are treated equally "bad". In contrast, for the same population, Fig. 3b shows the individuals identified as noninferior using the CFON model as prescribed in this paper. Since feasibility is considered first, the CFON approach finds a better representation (more number of points) of the true Pareto frontier.

4.3 Second improvement: CH-I2

The second improvement, CH-I2, incorporates the assumption (ii) of Sect. 4.1 in addition to assumption (i). As a result, the amount of infeasibility is taken into account when constraints are handled. CH-I2 would reduce the bad fitness values of infeasible individuals in a given population by a quantity called: *factor1* (see below), depending on the extent of constraint violation. In addition to the eight steps in CH-I1, the following step is performed in CH-I2.

Step 9. Ignore steps 3 and 8 in Sect. 4.2 when calculating the fitness of all infeasible individuals, instead, obtain the fitness value of every infeasible individual using (5)

$$\text{fitness}_i = [C_{\max} - (C_{\max} - C_{\min})\,(r-1)/(M-1)] - factor1\,, \tag{5}$$

where $r = 0.8M$ and

$$factor1 = CF1 \times \left[\frac{\left[\sum_{j=1}^{J} \max(g_{j,i}(\mathbf{x}), 0) + \sum_{k=1}^{K} |h_{k,i}(\mathbf{x})|\right]}{\left(\left[\sum_{i=1}^{M}\sum_{j=1}^{J} \max(g_{j,i}(\mathbf{x}), 0) + \sum_{i=1}^{M}\sum_{k=1}^{K} |h_{k,i}(\mathbf{x})|\right]/M\right)} \right], \tag{6}$$

where $g_{j,i}$ is the j-th inequality constraint value for the i-th individual, and $h_{k,i}$ is the k-th equality constraint value for the i-th individual. The quantity $CF1$ is a correction factor with its value prescribed to be between 0.0005 to 0.015. The other parameters are defined right after (4).

In formulating (6), it is assumed that all constraints are scaled so that their values are of an order of magnitude of 1. For instance a constraint such as: $x_1x_2 - 2800 \leq 0$ can be scaled to the form $(x_1x_2/2800 - 1 \leq 0)$. If the constraints are not scaled, then different constraints would be of different orders of magnitude, and it would not be appropriate to sum up the violations. As shown in (6), the quantity: *factor1*, is computed with respect to the other individuals of the population. It is the ratio of the sum of constraint violations for the individual under consideration to the average constraint violation in the population, multiplied by a factor $CF1$.

4.4 Third improvement: CH-I3

In the third improvement, CH-I3, a strategy that takes both assumptions (iii) and (i) of Sect. 4.1 in the constraint handling is used. Instead of taking the amount of the constraint violation into account, CH-I3 makes use of the number of violated constraints. Again, in addition to the eight steps in CH-I1, the following step is performed in CH-I3.

Step 9. Ignore steps 3 and 8 in Sect. 4.2 when calculating the fitness of all the infeasible individuals, instead, obtain the fitness value of every infeasible individual using (7)

$$\text{fitness}_i = [C_{\max} - (C_{\max} - C_{\min})\,(r-1)/(M-1)]] - factor2\,, \tag{7}$$

where

$$factor2 = CF2 \left[\left(\sum_{j=1}^{J} \delta_{j,i} + \sum_{k=1}^{K} \delta_{k,i} \right) \Big/ (J+K) \right], \tag{8}$$

$\delta_{j,i}$ (or $\delta_{k,i}$) $= 1$ if g_j (or h_k) is violated for the i-th individual, $\delta_{j,i}$ (or $\delta_{k,i}$) $= 0$ otherwise.
The quantity $CF2$ is a correction factor whose value is prescribed to be between 0.0005 to 0.015. The other parameters are defined right after (4).

In (7) and (8), the quantity: *factor2*, penalizes the infeasible individuals depending on the number of constraints that the individual violates. The value of *factor2*, unlike *factor1*, does not depend on the other individuals in the population.

4.5 Fourth (hybrid) improvement: CH-I4

In this section, a new constraint handling technique, CH-I4, that seeks to combine the two independent factors, *factor1* and *factor2* described in Sect. 4.3 and 4.4, using a weighting technique is introduced. This technique takes all three assumptions in Sect. 4.1 into account. Again, in addition to the eight steps in CH-I1, the following step is performed in CH-I4. Step 9. Ignore step 3 and 8 in Sect. 4.2 when calculating the fitness of all the infeasible

solutions, instead, obtain the fitness value of every infeasible individual using (9)

$$\text{fitness}_i = \left[C_{\max} - (C_{\max} - C_{\min})\,(r-1)/(M-1)\right] - (w_1 \times \mathit{factor}1 + w_2 \times \mathit{factor}2)\,, \quad (9)$$

where $r = 0.8M$; and the quantities w_1 and w_2 are the weighting factors that are determined with the following steps. (In the following: "$a \succ b$" means "a is preferred to b".)

(i) Compute $\mathit{factor}1_{\text{avg}}$, the average of *factor1* for all infeasible individuals in the population.
(ii) Compute $\mathit{factor}2_{\text{avg}}$, the average of *factor2* for all infeasible individuals in the population.
(iii) For each individual in the population, compare *factor1* with $\mathit{factor}1_{\text{avg}}$ and *factor2* with $\mathit{factor}2_{\text{avg}}$.
Case 1. If ($\mathit{factor1} > \mathit{factor}1_{\text{avg}}$) and ($\mathit{factor2} < \mathit{factor}2_{\text{avg}}$), then ($\mathit{factor1} \succ \mathit{factor2}$). Set $w_1 = 0.75$ and $w_2 = 0.25$.
Case 2. If ($\mathit{factor}1 < \mathit{factor}1_{\text{avg}}$) and ($\mathit{factor}2 > \mathit{factor}2_{\text{avg}}$), then ($\mathit{factor}1 \prec \mathit{factor}2$). Set $w_1 = 0.25$ and $w_2 = 0.75$.
Case 3. For all other cases, both *factor1* and *factor2* are considered equally important. Set $w_1 = 0.50$ and $w_2 = 0.50$.

5 Examples

The above-mentioned constraint handling improvements are demonstrated in this section with two engineering design examples: speed reducer design and design of a fleet of ships. For both examples, the constraint handling techniques (i.e. CH-NA, and CH-I1 to CH-I4) are implemented in a MOGA (Narayanan and Azarm 1999).

5.1 Speed reducer design

This example was originally formulated by Golinski (1970) as a single-objective optimization problem. Here, the problem has been converted into a two-objective optimization problem (following Azarm *et al.* 1989 for a three-objective formulation). The example represents the design of a simple gear-box, as shown in Fig. 4, that might be used in a light airplane between the engine and propeller.

The mathematical formulation, (10), of the problem is now described. The seven design variables in the formulation are: gear face width (x_1), teeth module (x_2), number of teeth of pinion (x_3 integer variable), distance between bearings 1 (x_4), distance between bearings 2 (x_5), diameter of shaft 1 (x_6), and diameter of shaft 2 (x_7). The first design objective, f_1, is to minimize the volume. The second objective, f_2, is to minimize the stress in one of the

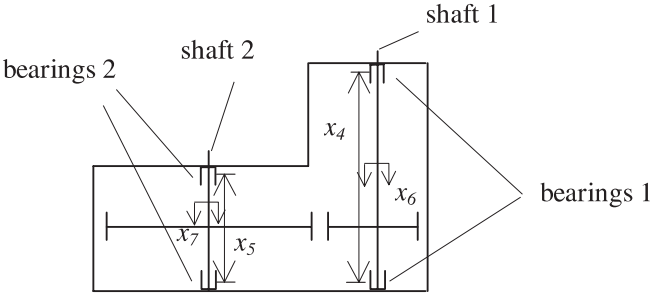

Fig. 4 A speed reducer

two gear shafts. The design is subject to a number of constraints imposed by gear and shaft design practices. An upper and lower limit is imposed on each of the seven design variables. There are 11 other inequality constraints (of which one is a constraint imposed on the first objective), as follows: g_1 is an upper bound of the bending stress of the gear tooth; g_2: upper bound of the contact stress of the gear tooth; g_3, g_4 are upper bounds of the transverse deflection of the shafts; g_5-g_7 are dimensional restrictions based on space and/or experience; g_8, g_9 are design requirements on the shaft based on experience; and g_{10}, g_{11} are constraints on stress in the gear shafts. The optimization formulation is

$$\text{minimize } f_{\text{weight}} = f_1 =$$

$$0.7854x_1x_2^2(10x_3^2/3 + 14.933x_3 - 43.0934) -$$

$$1.508x_1(x_6^2 + x_7^2) + 7.477(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)\,,$$

$$\text{minimize } f_{\text{stress}} = f_2 = \frac{\sqrt{(745x_4/x_2x_3)^2 + 1.69 \times 10^7}}{0.1x_6^3}$$

subject to

$$g_1 : \frac{1}{(x_1x_2^2x_3)} - \frac{1}{27} \le 0\,, \quad g_2 : \frac{1}{(x_1x_2^2x_3^2)} - \frac{1}{397.5} \le 0\,,$$

$$g_3 : \frac{x_4^3}{(x_2x_3x_6^4)} - \frac{1}{1.93} \le 0\,, \quad g_4 : \frac{x_5^3}{(x_2x_3x_7^4)} - \frac{1}{1.93} \le 0\,,$$

$$g_5 : x_2x_3 - 40 \le 0\,, \quad g_6 : \frac{x1}{x_2} - 12 \le 0\,,$$

$$g_7 : 5 - \frac{x_1}{x_2} \le 0\,, \quad g_8 : 1.9 - x_4 + 1.5x_6 \le 0\,,$$

$$g_9 : 1.9 - x_5 + 1.1x_7 \le 0\,, \quad g_{10} : f_1(x) \le 1300\,,$$

$$g_{11} : \frac{\sqrt{(745x_5/x_2x_3)^2 + 1.575 \times 10^8}}{0.1x_7^3} \le 1100\,. \quad (10)$$

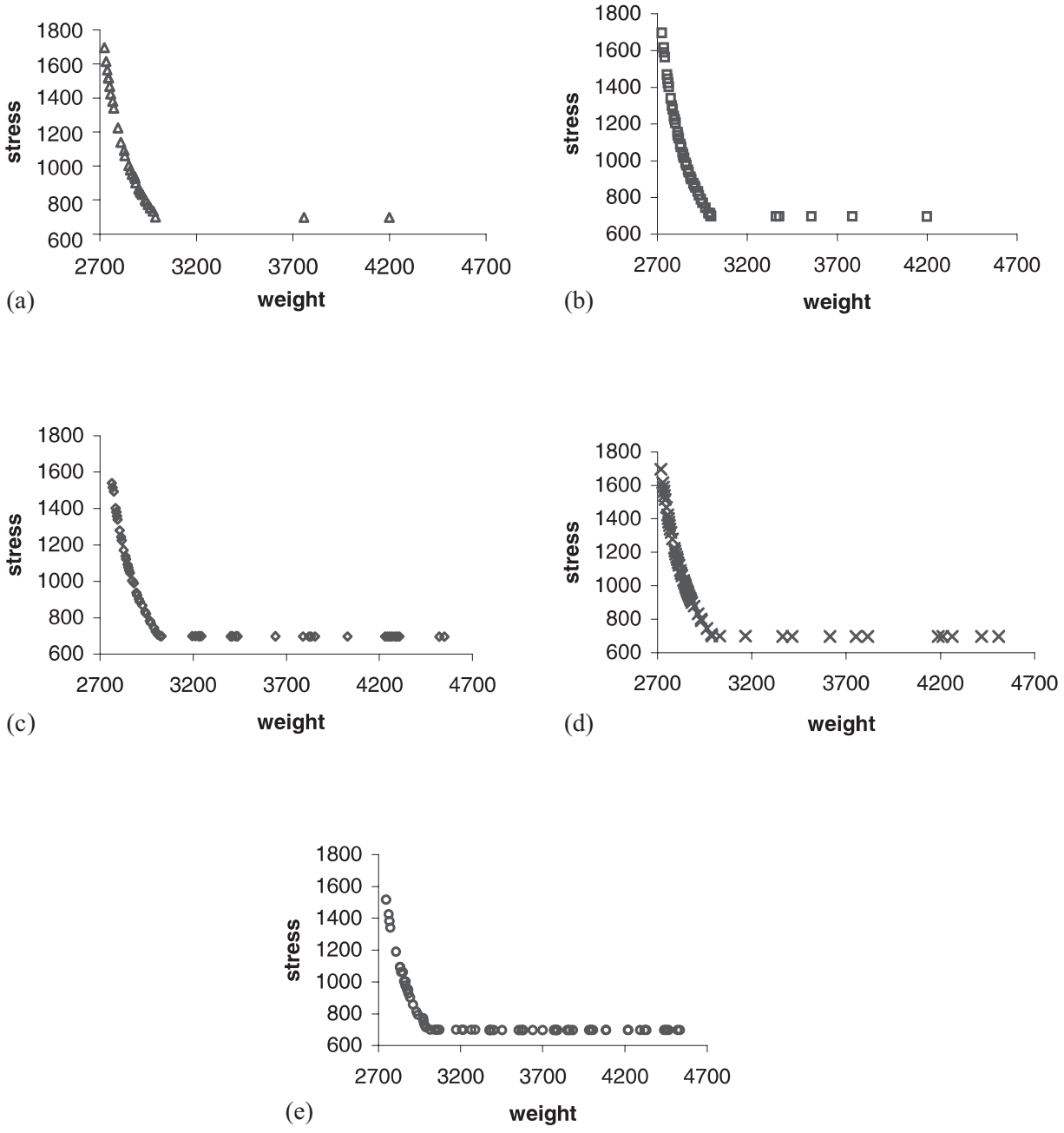

Fig. 5 Speed reducer's Pareto solutions for a MOGA with (a) CH-NA, and with the improvements: (b) CH-I1, (c) CH-I2, (d) CH-I3, and (e) CH-I4

The lower and upper limits on the seven variables are

$g_{12,13}$; $2.6 \leq x_1 \leq 3.6$, $g_{14,15}$: $0.7 \leq x_2 \leq 0.8$,

$g_{16,17}$; $17 \leq x_3 \leq 28$, $g_{18,19}$: $7.3 \leq x_4 \leq 8.3$,

$g_{20,21}$; $7.3 \leq x_5 \leq 8.3$, $g_{22,23}$: $2.9 \leq x_6 \leq 3.9$,

$g_{24,25}$; $5.0 \leq x_7 \leq 5.5$.

The Pareto solutions obtained using the five constraint handling approaches, i.e. the baseline approach, CH-NA, and the four proposed improvements, CH-I1 to CH-I4 described in Sect. 4, are shown in Fig. 5. From the results shown in Fig. 5 and Table 1, one can see that all of the proposed constraint handling improvements, i.e. CH-I1 to CH-I4, have out-performed the baseline approach of CH-NA by Narayanan and Azarm (1999) in terms of both

Table 1 Speed reducer computational cost

Function calls per Pareto point				
CH-NA	CH-I1	CH-I2	CH-I3	CH-I4
305	210	165	174	152

the computional cost (i.e. function calls per Pareto point) and closeness of the solutions to the ideal point. One can also see from Fig. 5 that the solutions produced by CH-I4 are more uniformly distributed.

5.2 Design of a fleet of ships

In this example, a fleet of ships is designed to carry oil from one port to another that is located 2900 nautical miles away. The problem was originally taken from Folkers (1973), but modified with the information taken from Manning (1956), Rao (1997), Schneekluth (1987). The objectives are: (i) to minimize the overall cost of building and operating a fleet of oil tankers, and (ii) to maximize the cargo capacity of the fleet. The first objective is given by (11)

$$\text{cost} = N\left(C_{\text{hull}} + C_{\text{machinery}} + C_{\text{fuel}}\right) , \tag{11}$$

In (11), the cost of building the hull, C_{hull}, is proportional to the amount of steel used, W_{st}, and is given by $C_{\text{hull}} = K_{st}W_{st}$, where is a constant. This formula is obtained from Schneekluth (1987). The costs of purchasing machinery and fuel are both proportional to the engine power. Hence, from the admiralty formula (Folkers 1973), $C_{\text{machinery}}$ and C_{fuel} are obtained.

The second objective is given by (12)

$$\text{capacity} = N\left(\frac{DwtV}{R} - \frac{FV^3Dwt^{2/3}}{R}\right)UW\,. \tag{12}$$

There are nine design variables. They are: (i) the number of ships in the fleet, N, (ii) the length of each ship, L, (ii) breadth, B, (iv) depth, D, (v) draft, T, (vi) deadweight, Dwt, (vii) displacement, Z, (viii) speed, V, and (ix) utilization factor, U. The constraints are categorized into two types. Type 1: operational requirements which are limits on the size of the ship and are placed due to factors like port facilities, canal width, depth, etc. The constraint g_1 represents the fact that the time of voyage at sea must be less than the total operating time. Here, $R =$ range, $O =$ loading rate, $V =$ speed of ship and $U =$ utilization factor. Type 2: ship-building constraints are imposed because of purely engineering considerations. The constraint g_2 is the Archimedes' principle. The constraint g_3 has to do with the shape of the ship and its speed. The "finer" the shape is, the faster it is. The ship should have the ability to return to equilibrium after heeling. This is specified by g_4. The constraint g_5 represents safety for the deck immersion. The ship's speed, characterized by the Froude number, is set within limits in constraints g_6 and g_7. The ship should have a minimum length to depth ratio to reduce resistance and maintain stability. The constraints g_{12} and g_{13} put limits on that. The ratios L/B, B/T, L/D, T/D and C_b have upper and lower bounds on their values to prevent an unrealistic design from being generated.

Mathematically, the formulation is shown in (13)

$$\text{minimize cost} = N\left(C_{\text{hull}} + C_{\text{machinery}} + C_{\text{fuel}}\right) ,$$

$$\text{maximize capacity} = N\left(\frac{DwtV}{R} - \frac{FV^3Dwt^{2/3}}{R}\right)UW$$

subject to

$$g_1 :\ U - \frac{RO}{RO + 2DwtV} \le 0\,,$$

$$g_2 :\ W_{st} + 0.02(V^3Z^{2/3})^{0.72} + Dwt - Z \le 0\,,$$

$$g_3 :\ \frac{Dwt}{0.3LBD} - 1.0 \le 0\,,$$

$$g_4 :\ 1.5 + 0.45D -$$

$$B\left(\frac{0.08B}{\sqrt{Cm}T} + \frac{T(0.9 - 0.3C_m - 0.1Cb)}{B}\right) \le 0\,,$$

$$g_5 :\ 0.0019L^{1.43} + T - D \le 0\,,$$

$$g_6 :\ 0.14 - \frac{V}{\sqrt{gL}} \le 0\,, \quad g_7 :\ \frac{V}{\sqrt{gL}} - 0.32 \le 0\,,$$

$$g_8 :\ 0.6 - C_b \le 0\,, \quad g_9 :\ C_b - 0.72 \le 0\,,$$

$$g_{10} :\ 4 - \frac{L}{B} \le 0\,, \quad g_{11} :\ \frac{L}{B} - 7 \le 0\,,$$

$$g_{12} :\ 10 - \frac{L}{D} \le 0\,, \quad g_{13} :\ \frac{L}{D} - 14 \le 0\,,$$

$$g_{14} :\ 2 - \frac{B}{T} \le 0\,, \quad g_{15} :\ \frac{B}{T} - 4 \le 0\,,$$

$$g_{16} :\ 0.61 - \frac{T}{D} \le 0\,, \quad g_{17} :\ \frac{T}{D} - 0.87 \le 0\,,$$

$$g_{18} :\ N \le 100\,, \quad g_{19} :\ 0.6 \le U \le 1\,,$$

$$g_{20} :\ 100 \le L \le 289.56\,, \quad g_{21} :\ 10.0 \le B \le 32.24\,,$$

$$g_{22} :\ 10.0 \le D \le 57.91\,, \quad g_{23} :\ 6.0 \le T \le 12.4\,,$$

$$g_{24} :\ V \le 25\,, \quad g_{25} :\ Dwt \le 6 \times 10^4\,,$$

$$g_{26} :\ z \le 1 \times 10^5\,,$$

$$N, L, B, D, T, V, Z, Dwt, U \ge 0\,, \tag{13}$$

where $F = 5 \times 10^{-5}$, $R = 2900$, $W = 8640$, $O = 2500$, $C_m = 0.98$, and $g = 9.82$.

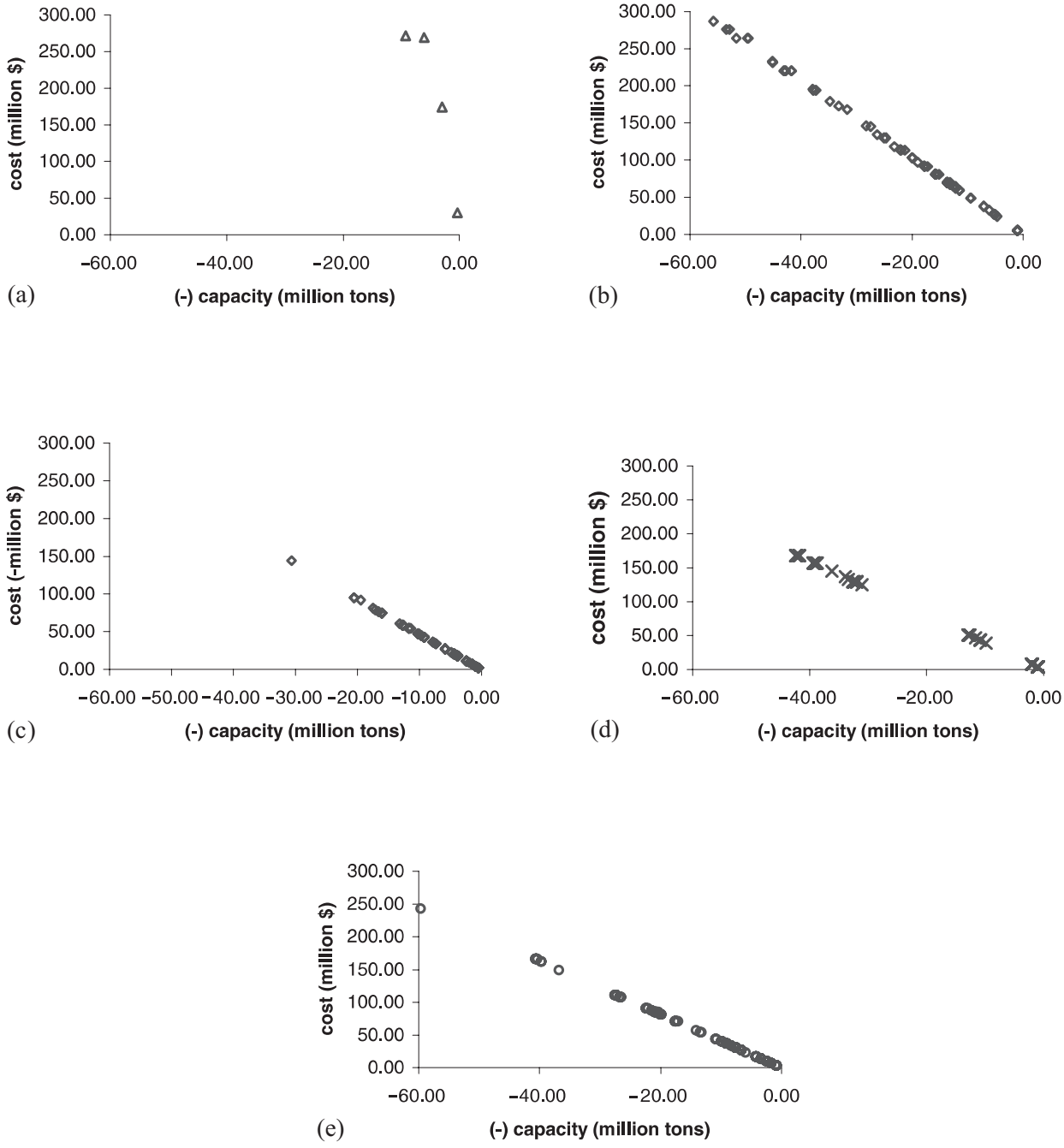

Fig. 6 Ship design's Pareto solutions for a MOGA with (a) CH-NA, and with the improvements: (b) CH-I1, (c) CH-I2, (d) CH-I3, and (e) CH-I4 Pareto solutions

The results obtained for the ship design problem are shown in Fig. 6. From the results one can see that again the suggested constraint handling approaches, CH-I1 to CH-I4, all have out-performed CH-NA in terms of the computational cost (see Table 2). Also, the proposed approaches all provide solutions that are closer to the ideal point than CH-NA. Again, One can also see from Fig. 6 that the solutions produced by CH-I4 are more uniformly distributed.

Table 2 Ship design computational cost

Function calls per Pareto point				
CH-NA	CH-I1	CH-I2	CH-I3	CH-I4
8408	263	637	438	347

6 Conclusions

Four improved constraint handling techniques, i.e. CH-I1 to CH-I4, have been proposed in this paper. A compar-

ison between the proposed improvements and a baseline approach, i.e. CH-NA by Narayanan and Azarm (1999), in terms of function calls, solution distribution and closeness to the ideal point, are presented with the help of two engineering design examples: a speed reducer design and the design of a fleet of ships.

In both examples, it was observed that all four of the proposed constraint handling approaches out-performed CH-NA. As such, it is concluded that the constraint handling approaches that are based on a "Constraint-First-Objective-Next" model generally perform better than those which are based on a "Objective-First-Constraint-Next" model. One clear advantage of the proposed constraint handling techniques is that they require the definition of only a few parameters. The improvements also show that both the amount of infeasibility and the number of violated constraints are important and should not be ignored during the constraint handling process.

Acknowledgements The work presented in this paper was supported in part by NSF (DMI-9700059) and ONR (N000-1498108492), and in part by the Maryland Industrial Partnerships and IHDIV-NSWC. Such support does not constitute an endorsement by the funding agency of the opinions expressed in the paper.

References

Azarm, S.; Tits, A.; Fan, M.K.H. 1989: Tradeoff driven optimization-based design of mechanical systems. In: *Proc. 4-th AIAA/USAF/NASA/OAI Symp. on Multidisciplinary Analysis and Optimization*, AIAA-92-4758-CP (held in Cleveland, OH), pp. 551–558

Coello Coello, C.A. 1999a: A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Int. J. Knowledge Inform. Syst* **1**, 269–308

Coello Coello, C.A. 1999b: A survey of constraint handling techniques used with evolutionary algorithms. *Technical Report Lania-RI-99-04*, Laboratorio Nacional de Informática Avanzada, Mexico

Eschenauer, H.; Koski, J.; Osyczka, A. (eds.) 1990: *Multicriteria design optimization*. Berlin, Heidelberg, New York: Springer

Folkers, J.S. 1973: Ship operation and design. In: Avriel M.; Rijckaert M.J.; Wilde D.J. (eds.) *Optimization and design*, pp. 221–226. New York: Prentice Hall

Fonseca, C.M.; Fleming, P.J. 1993: Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In: Forrest, S. (ed.) *Proc. 5-th Int. Conf. on Genetic Algorithms* (held in San Mateo, CA), pp. 416–423

Glover, F.; Laguna, M. 1997: *Tabu search*. Boston: Kluwer

Goldberg, E.D. 1989: *Genetic algorithms in search, optimization and machine learning*. New York: Addison-Wesley

Golinski, J. 1970: Optimal synthesis problems solved by means of nonlinear programming and random methods. *J. Mechanisms* **5**, 287–309

Holland, J.H. 1975: *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press

Homaifar, A.; Qi, C.X.; Lai, S.H. 1994: Constrained optimization via genetic algorithms. *Simulation* **62**, 242–254

Joines, J.; Houck, C. 1994: On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with genetic algorithms. *Proc. 1-st IEEE Conf. on Evolutionary Computation* (held in Orlando, FL), pp. 579–584

Kuri, M.A.; Quezada, C.V. 1998: A universal eclectic genetic algorithm for constrained optimization. *Proc. 6-th European Cong. on Intelligent Techniques and Soft Computing, EUFIT '98*, pp. 518–522. Aachen, Germany: Verlag Mainz

Manning, G.C. 1956: *The theory and technique of ship design*. Boston, MA: MIT Press

Michalewicz, Z.; Attia, N. 1994: Evolutionary optimization of constrained problems. *Proc. 3-rd Annual Conf. on Evolutionary Programming*, pp. 98–108. World Science

Narayanan, S.; Azarm, S. 1999: On improving multiobjective genetic algorithms for design optimization. *Struct. Optim.* **18**, 146–155

Papalambros, P.Y.; Wilde, D.J. 1988: *Principles of optimal design*. New York: Cambridge University Press

Rao S.S. 1997: *Theory and applications of optimization*. New York: Wiley Eastern

Richardson, J.T.; Palmer, M.R.; Liepins, G.; Hilliard, M. 1989: Some guidelines for genetic algorithms with penalty functions. In: *Proc. 3-rd Int. Conf. on Genetic Algorithms*, pp. 191–197. George Mason University: Mogan Kaufmann Publishers

Schaffer, J.D. 1985: Multiple objective optimization with vector evaluated genetic algorithms. In: *Genetic algorithms and their applications* (Proc. 1-st Int. Conf. on Genetic Algorithms), pp. 93–100. Lawrence: Erlbaum

Schneekluth, H. 1987: Ship design for efficiency and economy. Aachen University of Technology, Germany

Srinivas, N.; Deb, K. 1994: Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation* **2**, 221–248

Van Laarhoven, P.J.M. 1987: *Simulated annealing: theory and applications*. Dordrecht: Kluwer