IIE Transactions

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/uiie20

Genetic-algorithm-based optimal apportionment of reliability and redundancy under multiple objectives

Hong-Zhong Huang [a], Jian Qu [b] & Ming J. Zuo [b]

[a] School of Mechatronics Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China

[b] Department of Mechanical Engineering, University of Alberta, Edmonton, Alberta, Canada, T6G 2G8

Version of record first published: 28 Jan 2009.

To cite this article: Hong-Zhong Huang, Jian Qu & Ming J. Zuo (2009): Genetic-algorithm-based optimal apportionment of reliability and redundancy under multiple objectives, IIE Transactions, 41:4, 287-298

To link to this article: http://dx.doi.org/10.1080/07408170802322994

IIE Transactions (2009) **41**, 287–298

ISSN: 0740-817X print / 1545-8830 online
DOI: 10.1080/07408170802322994

Genetic-algorithm-based optimal apportionment of reliability and redundancy under multiple objectives

HONG-ZHONG HUANG[1,*], JIAN QU[2] and MING J. ZUO[2]

[1]*School of Mechatronics Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China*
E-mail: hzhuang@uestc.edu.cn
[2]*Department of Mechanical Engineering, University of Alberta, Edmonton, Alberta, Canada, T6G 2G8*
E-mail: jqu@ualberta.ca or ming.zuo@ualberta.ca

Received November 2006 and accepted June 2007

When solving multi-objective optimization problems subject to constraints in reliability-based design, it is desirable for the decision maker to have a sufficient number of solutions available for selection. However, many existing approaches either combine multiple objectives into a single objective or treat the objectives as penalties. This results in fewer optimal solutions than would be provided by a multi-objective approach. For such cases, a niched Pareto Genetic Algorithm (GA) may be a viable alternative. Unfortunately, it is often difficult to set penalty parameters that are required in these algorithms. In this paper, a multi-objective optimization algorithm is proposed that combines a niched Pareto GA with a constraint handling method that does not need penalty parameters. The proposed algorithm is based on Pareto tournament and equivalence sharing, and involves the following components: search for feasible solutions, selection of non-dominated solutions and maintenance of diversified solutions. It deals with multiple objectives by incorporating the concept of Pareto dominance in its selection operator while applying a niching pressure to spread the population along the Pareto frontier. To demonstrate the performance of the proposed algorithm, a test problem is presented and the solution distributions in three different generations of the algorithm are illustrated. The optimal solutions obtained with the proposed algorithm for a practical reliability problem are compared with those obtained by a single-objective optimization method, a multi-objective GA method, and a hybrid GA method.

Keywords: Reliability optimization, multi-objective optimization, genetic algorithms, Pareto solutions

1. Introduction

Reliability optimization is very important in system design. Usually, a single-objective optimization model is formulated. For example, system reliability may be maximized subject to resource constraints, or cost may be minimized subject to reliability requirements. To achieve the best system design, it is often desirable to simultaneously maximize system reliability and minimize resource consumption. In this case, it is better to adopt a multi-objective approach to system design (Kuo and Prasad, 2000). In solving multi-objective optimization problems, the decision maker must have a sufficient number of desirable optimal solutions to choose from. However, multiple objectives and various constraints make it difficult to simultaneously solve this type of problem.

Currently available approaches may combine multiple objectives into a single objective, treat the objectives as penalties, or apply interactive techniques to reduce the size of the optimization problem. Dhingra (1992) and Rao and Dhingra (1992) studied a reliability and redundancy allocation problem for a four-stage and a five-stage over-speed protection system, using crisp and fuzzy multi-objective optimization approaches, respectively. Li (1996) proposed a Genetic Algorithm (GA) approach for multi-objective reliability design problems maximizing the reliability and minimizing the total cost of the system. Gen and Kim (1998, 1999a, 1999b) proposed a hybrid GA approach to handle multi-objective reliability-redundancy allocation problems and compared it with the GA approach reported by Li (1996). Huang *et al*. (2004) and Huang, Tian and Zuo (2005) developed an interactive physical programming approach and applied it to a reliability and redundancy allocation problem. Huang (1997) also reported a fuzzy multi-objective optimization decision-making method for a series system. Salazar *et al*. (2006) solved constrained multiple-objective reliability optimization problems by using a second-generation multiple-objective evolutionary algorithm. Coit and Konak (2006) proposed a multiple weighted objectives heuristic for a redundancy allocation problem based on the transformation of a multiple objective

*Corresponding author

problem into a single-objective problem. For other related papers see Huang, Wu and Liu (2005), Ha and Kuo (2006), Huang *et al.* (2006), Liang and Chen (2007), Onishi *et al.* (2007) and Yun *et al.* (2007). In this paper we introduce a direct approach that uses GAs to solve multi-objective optimization problems.

GAs can be viewed as a probabilistic approach based on the principle of natural evolution. Their advantage over other methods, such as exact algorithms and heuristic algorithms, is that they can simultaneously manipulate the entire population of solutions to the optimization problem. This property makes it possible to approximate the whole Pareto frontier in a single optimization run for a multi-objective optimization problem.

Fonseca and Fleming (1995) provided an overview of GAs for multi-objective optimization. Deb *et al.* (2002) proposed a fast and elitist multi-objective GA. Horn and Nafpliotis (1993) and Horn *et al.* (1994) proposed the niched Pareto GA for multi-objective problems that is now used in many areas. Erickson and Horn (2002) applied a niched Pareto GA to simultaneously minimize remedial design cost and contaminant mass. Zheng *et al.* (2005) applied the Pareto ranking and niche formation to GA-based multi-objective time-cost optimization.

Since GAs are appropriate for high-dimension stochastic problems with many non-linearities or discontinuities, they are suited to solving reliability-based design problems. In this paper, we propose an algorithm that combines a population-based constraint handling method and a niched Pareto GA based on Pareto tournament and equivalence sharing to solve optimal reliability-redundancy allocation problems.

2. Problem statement

The reliability optimization model considered in this paper is obtained by transforming the single-objective optimization model of an over-speed protection system reported by Dhingra (1992) into a multi-objective optimization model. Its objective functions are maximizing system reliability and minimizing system cost, subject to limits on weight and volume.

Notation

R_i = reliability of a component at stage *i*;
n_i = number of redundant components at stage *i*;
R_s = system reliability;
C_s = total system cost;
W_s = total system weight;
V_s = total system volume;
N = number of stages;
$W_{\lim}$ = upper limit on weight;
$V_{\lim}$ = upper limit on volume;
w_i = the weight of each component in stage *i*;
v_i = the volume of each component in stage *i*;
α_i, β_i = constants representing the physical characteristics of each component at stage *i*;
T = operating time.

The model takes the following form:

$$\max R_s = \prod_{i=1}^{N} [1 - (1 - R_i)^{n_i}], \tag{1}$$

$$\min C_s = \sum_{i=1}^{N} \alpha_i \times \left(\frac{-T}{\ln(R_i)}\right)^{\beta_i} \times [n_i + \exp(n_i/4)], \tag{2}$$

subject to

$$W_s = \sum_{i}^{N} w_i \times n_i \times \exp(n_i/4) \le W_{\lim}, \tag{3}$$

$$V_s = \sum_{i}^{N} v_i \times (n_i)^2 \le V_{\lim}, \tag{4}$$

$$1 \le n_i \le n_{\max}, \quad R_{\min} \le R_i \le R_{\max}, \quad i = 1, 2, \ldots, N, \tag{5}$$

where $\exp(n_i/4)$ accounts for the interconnecting hardware, $n_{\max}$ represents the maximum number of components allowed at each stage and $R_{\min}$ and $R_{\max}$ the minimum and maximum reliability limits of each component. The parameters α_i and β_i provide flexibility in expressing the cost of the system as a function of the number of components n_i at each stage *i*, the mission time *T* and the component reliability level R_i at stage *i*. The adopted cost function form is as proposed by Dhingra (1992). It is adopted here for ease of comparison of the proposed algorithm with those used in Dhingra (1992). Our approach, however, is more general. When applying our approach, one should use a cost function appropriate to the specific application.

3. Multi-objective GA

GAs have been widely used to solve multi-objective optimization problems. Two factors that must be considered in such cases are fitness assignment and fitness sharing.

Fitness assignment provides a criterion for assessing the fitness of an individual solution. The Pareto ranking method, which is usually applied for this purpose, relies on the notion of dominance, that is, a better solution must have better objective function values. The concept of Pareto optimality can be used here for ranking individual solutions. A solution to a multi-objective optimization problem is called Pareto optimal if no objective can be improved without sacrificing another objective. A more formal definition of Pareto optimality is as follows (Fonseca and Fleming, 1995).

Definition 1. Without loss of generality, consider the maximization of *n* elements $f_k, \quad k = 1, \cdots, n,$ of vector function (**f**) with vector variable (**x**) in a universe (u), where:

$$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_n(\mathbf{x})). \tag{6}$$

Then, a decision vector, $\mathbf{s} \in u$, for which $\mathbf{g} = \mathbf{f}(\mathbf{s}) = (g_1, \ldots, g_n)$, dominates another decision vector, $\mathbf{t} \in u$, for which $\mathbf{h} = \mathbf{f}(\mathbf{t}) = (h_1, \cdots, h_n)$, if

$$\forall i \in \{1, \ldots, n\}, \quad h_i \leq g_i \wedge \exists i \in \{1, \cdots, n\} | \, h_i < g_i. \tag{7}$$

A decision vector is said to be Pareto optimal if it is not dominated by any other decision vectors. The set of all Pareto optimal decision vectors is called the admissible set of the problem. The corresponding set of objective vectors is called the non-dominated set.

By definition, the Pareto ranking method ranks all the individuals and then assigns a fitness value to each of them. The individuals that are non-dominated are assigned larger fitness values, so they will be copied more than the others in GA procedures.

The Pareto tournament method also uses the dominance notion. It chooses two individuals from the population at random, and then selects the better one to be included in the next generation. It is different from the Pareto ranking method in that it does not actually assign fitness values to individuals, so it is less complicated. As the commonly used pair-wise tournament method may generate more dominated solutions in the final population, an improved Pareto tournament method is used in this paper.

Other methods of fitness assignment include the weight-based method, the compromise-based method and the goal programming method (Gen and Cheng, 2000).

Fitness sharing refers to maintaining the diversity of the Pareto optimal solutions. Fitness sharing aims to provide Pareto optimal solutions over the entire non-dominated frontier. It ensures a set of diversified Pareto solutions.

The sharing scheme is designed to spread the population of individuals along the Pareto frontier by penalizing individuals that are already strongly represented in the population. There are two types of sharing: sharing in the objective space and sharing in the variable space. These approaches provide diversity of Pareto solutions in the objective space and the variable space, respectively. We calculate the following indicator for individual i (Horn and Nafpliotis, 1993):

$$m_i = \sum_{j=1}^{pop\_size} sh(d_{ij}), \tag{8}$$

where m_i is the niche count of individual i, which is an estimate of how crowded the neighborhood (niche) of individual i is; d_{ij}, usually calculated by the pth order norm (for example, $p = 2$), is the distance between individuals i and j; and $sh(\cdot)$ represents the following sharing function:

$$sh(d_{ij}) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma_{\text{share}}}\right)^2 & \text{if } d_{ij} < \sigma_{\text{share}}, \\ 0 & \text{else,} \end{cases} \tag{9}$$

where σ_{share} is the radius of the niche, specified by the user based on the minimal separation desired or expected among the solutions. The definition and the calculation method for σ_{share} are provided by Horn and Nafpliotis (1993).

Sharing usually leads to the degradation of an individual's objective fitness (f_i) by means of dividing f_i by the niche count m_i.

3.1. *The niched Pareto GA*

This section introduces the niched Pareto multi-objective GA incorporating the Pareto tournament fitness assignment and equivalence sharing proposed by Horn and Nafpliotis (1993) and Horn *et al.* (1994).

3.1.1. *Pareto tournament fitness assignment*

The tournament used here is a pair-wise tournament, and the sampling is conducted as follows. Two candidates are randomly selected from the population of solutions. A comparison set including t_{dom} individuals is also selected randomly from the population. Each of the two candidates is then compared against each individual in the comparison set. This tournament adheres to criterion 1 as outlined below.

Criterion 1:

1. If one candidate is dominated by any solution from the comparison set and the other is not, the latter is selected for reproduction.
2. If neither or both are dominated by any solution from the comparison set, then the sharing scheme given below will be used.

3.1.2. *Equivalence class sharing*

The niched Pareto GA applies a simpler method called equivalence class sharing, which does not use the degradation mechanism. The individual chosen from two candidates is the one with a smaller niche count within a predefined equivalence class region.

Figure 1 illustrates how this form of sharing works (Horn and Nafpliotis, 1993). Compared with candidate 1, which has a niche count of four, candidate 2 has a smaller niche count of one and is therefore selected as the winner.

3.2. *The constraint handling approach*

The most common approach to handling constraints in GAs is the use of penalties. Ideally, the penalty is kept as low as possible. Although conceptually simple, this approach is quite difficult to implement in practice because in most problems the exact location of the boundary between the feasible and infeasible regions is unknown *a priori* (Coello and Carlos, 2002). This paper uses the GA's population-based approach (Deb, 2000), which is a penalty function approach that does not require any penalty parameter for choosing a better solution. It uses a tournament selection

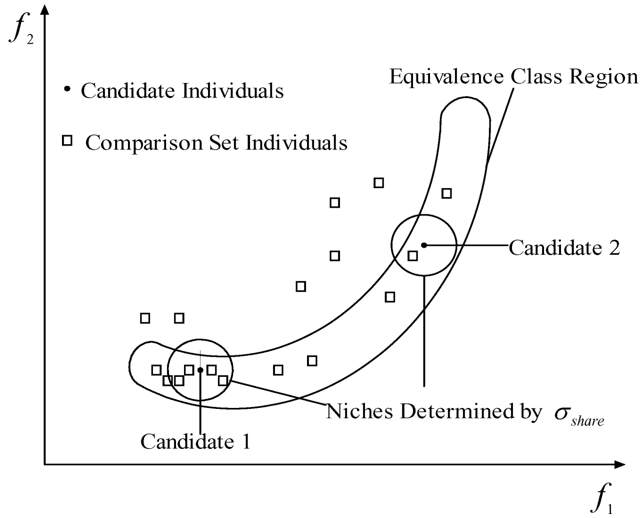

Fig. 1. Equivalence class sharing (Horn and Nafpliotis, 1993).

operator, where two solutions are compared at a time, and the tournament follows criterion 2.

Criterion 2:

1. A feasible solution is preferred to an infeasible solution.
2. Between two feasible solutions, the one having a better objective function value is preferred.
3. Between two infeasible solutions, the one having less constraint violation is preferred.

The extent of constraint violation for a solution may be judged by the number of violated constraints. The larger the number of violated constraints, the greater the number of violations of a solution.

Since the niched Pareto GA and the constraint handling approach have similar tournament-based properties, integrating them is less complicated. We propose our algorithm in the next section.

4. The proposed algorithm

The proposed algorithm uses the niched Pareto GA and the constraint handling method described earlier. It is obtained by integrating criterion 1 with criterion 2. The tasks of searching for feasible solutions and maintaining diversified Pareto solutions are simultaneously implemented in each generation. Figure 2 illustrates the proposed procedure.

In the flow chart shown in Fig. 2, P(t) and C(t) represent the parent population and the child population at generation t, and *max_gen* represents the maximum number of iterations allowed. Figure 3, which corresponds to the segment in the dashed rectangle of Fig. 2, shows the corresponding rules of criterion 3 where "Cand." denotes candidate, "Violat." denotes violation and "comp. set" denotes the comparison set.

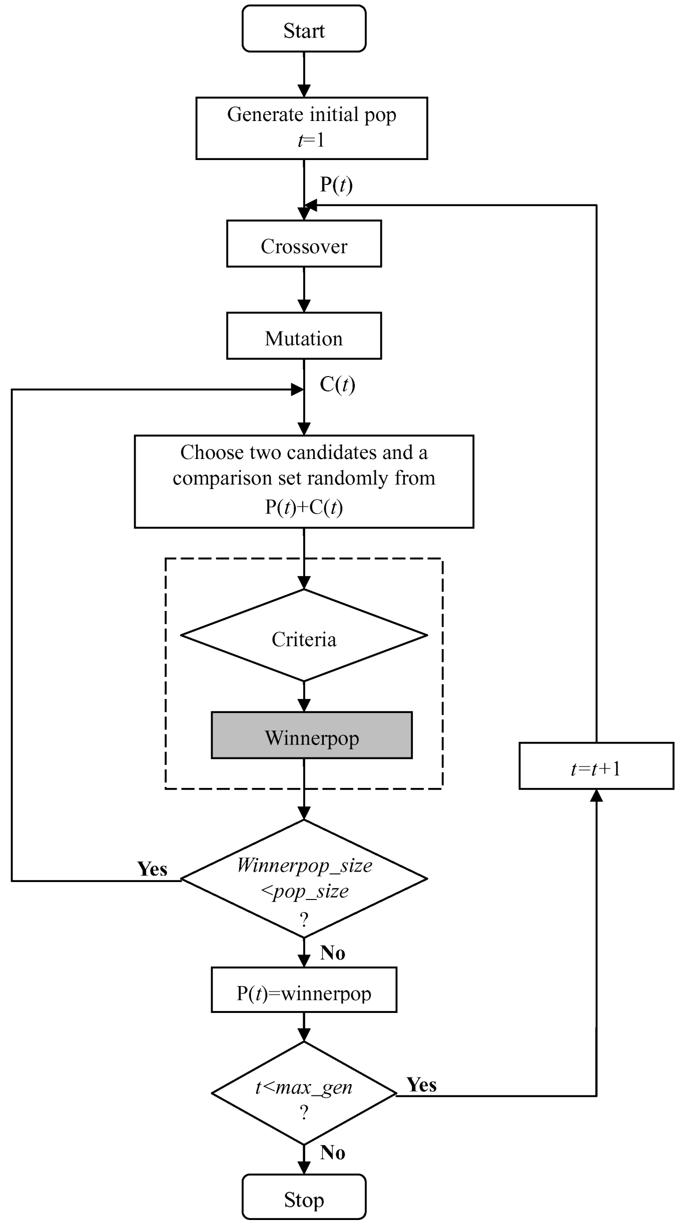

Fig. 2. Flow chart of the proposed algorithm.

Criterion 3:

1. If one candidate is infeasible and the other is feasible, the latter is incorporated into "winnerpop," where the winners are stored.
2. If both candidates are infeasible, the one having less constraint violation is incorporated into winnerpop. Otherwise, one is randomly incorporated into winnerpop.
3. If both candidates are feasible, one of five situations will arise. The corresponding strategies for incorporating candidate(s) into winnerpop are as follows:
   (i) if one candidate dominates the other, the dominating one is incorporated into winnerpop;

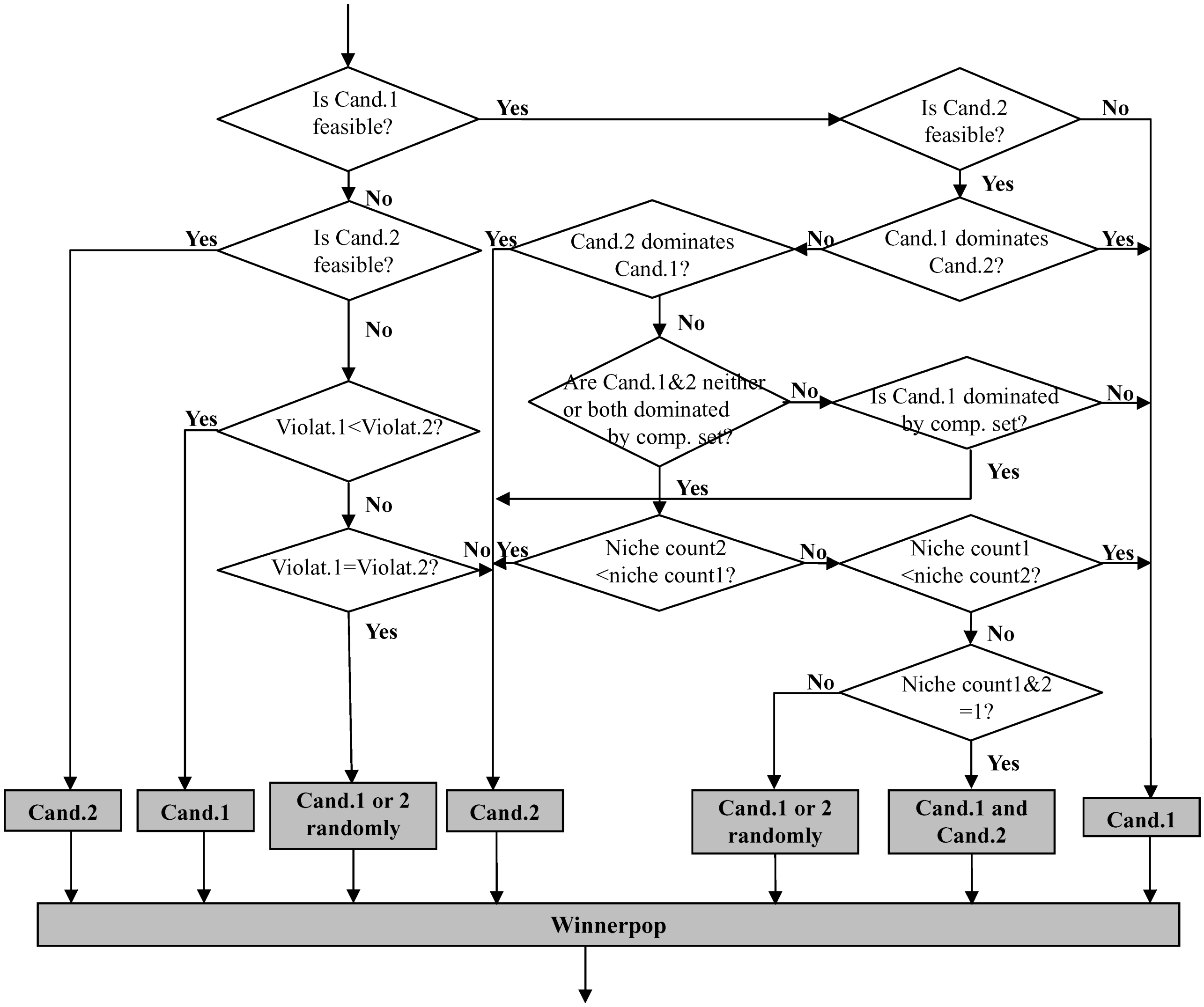

Fig. 3. Fragment in the dashed rectangle of Fig. 2.

(ii) if one of the two candidates is dominated by the comparison set, the non-dominated one is incorporated into winnerpop;

(iii) if the niche counts of the two candidates are different, the one with the smaller niche count is incorporated into winnerpop;

(iv) if both niche counts are equal to one, both are incorporated into winnerpop;

(v) if none of the above cases apply, one is randomly incorporated into winnerpop.

To obtain a set of Pareto solutions, the "initial pop" P(1) is generated at the beginning. The feasibility of individuals in the initial pop has a significant impact on the final optimality; and a lack of sufficient feasible individuals in the initial pop may cause the algorithm to fail. Of course, we can avoid this situation by running the algorithm many times. However, a more efficient method is to ensure a sufficient number of feasible individuals exist in the initial pop.

In this paper, the individuals are first randomly generated in a particular range as shown in Equation (5) and those that satisfy the constraints are directly incorporated into the initial pop. If the number of feasible individuals reaches half of the pop size, the generation of the initial pop is complete. Second, individuals in the population are processed by genetic operators such as crossover and mutation, and the child population obtained through this process is denoted by $C(t)$. The winning candidates, including $P(t)$ and $C(t)$, are selected from the mixed population according to criterion 3, and added to the winnerpop until the size of the winnerpop equals a specified population size. The winnerpop obtained at generation t is $P(t+1)$, which is used in the next generation. Solutions are improved as generations pass. After a certain number of generations specified by the user, a set of Pareto solutions is generated.

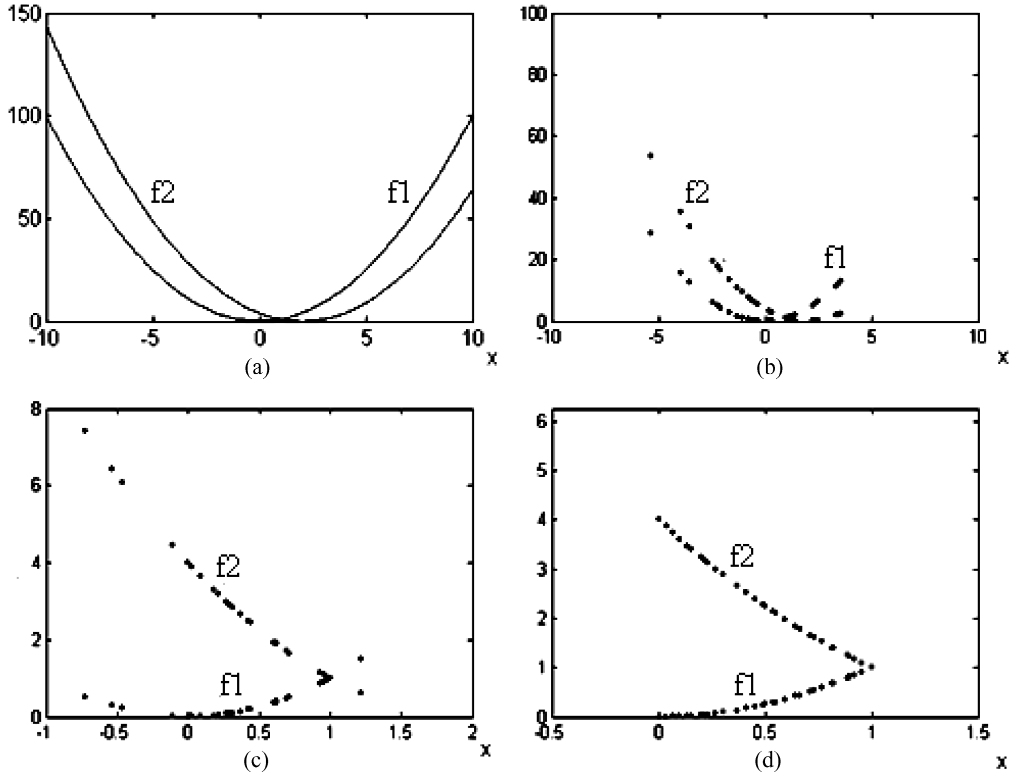

Fig. 4. (a) Objective functions of the test problem; (b) individual distribution improved by a binary tournament; (c) generation 5; and (d) generation 100.

5. A test problem

We first test the proposed algorithm with a function used by Horn and Nafpliotis (1993) and Horn *et al.* (1994) plus our own constraints. This is a simple function, with a single decision variable, the real-valued x, and two objective functions, f_1 and f_2, to be minimized:

$$\min f_1 = x^2, \tag{10}$$
$$\min f_2 = (x-2)^2, \tag{11}$$

subject to

$$g_1 = x^2 - 1 \leq 0, \tag{12}$$
$$g_2 = (x-1)^2 - 1 \leq 0. \tag{13}$$

We use real-number encoding over the range of $x \in [-10,\ 10]$, where the initial pop is generated. We plot f_1 and f_2 over this range in Fig. 4(a). It is clear that trade-offs between the two objective functions occur on the Pareto frontier. That is, for $0 \leq x \leq 2$, one of the functions decreases toward its best value while the other increases away from its best value. Because of the limits of the constraints, the range of the final Pareto solutions is reduced to $[0,\ 1]$.

We use the following parameters: population size *pop_size* $= 30$, *max_gen* $= 100$, niche size $\sigma_{\text{share}} = 0.1$ and tournament size $t_{\text{dom}} = 4$. The parameter selection principles, which are specified in Horn and Nafpliotis (1993) and Horn *et al.* (1994), are not repeated here.

The individuals are generated in the range $x \in [-10,\ 10]$ to form the initial pop. Because of the limited number of feasible solutions in this population, a simple binary tournament is used to search for the better solutions as illustrated in Fig. 4(b). Apparently, there are still many infeasible solutions but the number of feasible solutions is now adequate for our next step. As generations pass, feasible solutions will gradually occupy more positions in the population. As illustrated in Fig. 4(c), there are only five infeasible solutions left at generation 5. Figure 4(d) illustrates the population at generation 100, and there is a fairly even spread of solutions along the Pareto frontier.

Even though this test problem is an easy one, it serves to show that the proposed algorithm is effective in searching for feasible solutions and maintaining diversified Pareto solutions.

6. A reliability optimization problem

In this section we solve the reliability-redundancy allocation problem presented in Section 2, with $N = 4$, $n_{\max} = 10$, $R_{\min} = 0.5$, $R_{\max} = 1\text{–}10^{-6}$, and other parameters as specified in Table 1. We use real-number encoding in the form of chromosomes $[(R_1, n_1)(R_2, n_2)(R_3, n_3)(R_4, n_4)]$ and the following parameter values: *pop_size* $= 30$, *max_gen* $= 100$, $\sigma_{\text{share}} = 0.04$, $t_{\text{dom}} = 4$. We adopt arithmetic crossover in a crossover operation with the parameter of $P_c = 0.75$, and

Table 1. Design data for the sample problem ($W_{\rm lim} = 500$, $V_{\rm lim} = 250$, $T = 1000$)

Stage	$\alpha_j(\times 10^5)$	β_j	v_j	w_j
1	1.0	1.5	1	6
2	2.3	1.5	2	6
3	0.3	1.5	3	8
4	2.3	1.5	2	7

Table 3. Initial boundary pop and extreme solutions

	Initial boundary pop		*Extreme solutions in the final boundary pop*	
R_1	0.9036	0.59893	0.88036	0.59893
n_1	5	3	6	3
R_2	0.8566	0.61483	0.85632	0.61483
n_2	6	3	5	3
R_3	0.9153	0.62044	0.91245	0.62044
n_3	4	3	4	3
R_4	0.7515	0.58951	0.85768	0.58951
n_4	5	3	5	3
$R_{s-\max}$ R_s	0.9990	0.77613	0.99982	0.77613
C_s $C_{s-\min}$	273.0169	26.54	299.61	26.54
W_s	475.20	171.48	475.20	171.48
V_s	195.00	72.00	195.00	72.00

boundary mutation in mutation operation where a "boundary pop" is constructed consisting of two individuals having the maximal $R_s(R_{s-\max})$ and the minimal $C_s(C_{s-\min})$ of the previous population. The two members of the boundary pop are updated when an individual having a smaller C_s or a larger R_s than the old $R_{s-\max}$ or $C_{s-\min}$, respectively, arises in the current population. When mutation takes place, one of the two solutions in the boundary pop is randomly selected to replace the individual to be mutated. This process proceeds until the procedure terminates. To utilize the boundary pop, mutation should occur at least once during the whole procedure. Based on our experience, we select the mutation parameter $P_m = 0.1$.

Table 2 shows the two extreme solutions in each of the eight independent runs of the proposed algorithm for the optimization model given in Section 2, one extreme solution having the largest reliability value and the other having the lowest cost. From Table 2 we can see that all $R_{s-\max}$ values are above 0.992 01 and all $C_{s-\min}$ values are less than 27.958. This shows that the proposed algorithm provides consistently good end points of the Pareto frontier. In addition, the obtained system reliability ranges from 0.4002 to 0.9990, and the obtained system cost ranges from 17.454 to 266.17. This shows that the proposed algorithm provides a very broad Pareto frontier. One possible reason why the algorithm provides some solutions with very low system reliability (as low as 0.4002) and very high system cost (as high as 266.17) is that the model does not include a minimum reliability requirement or maximum budget for the system design.

In Table 3 we present the initial boundary pop where the members are selected from the randomly generated initial

Table 2. Solutions having maximal R_s and minimal C_s in eight independent runs

	Number							
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
$R_{s-\max}$	0.99599	0.99203	0.99861	0.99519	**0.99201**	0.99846	0.99847	**0.9990**
C_s	130.03	100.73	149.36	**266.17**	94.408	206.19	144.00	177.46
R_1	0.769	0.825	0.848	0.849	0.774	0.861	0.829	0.885
n_1	5	4	5	4	5	6	5	4
R_2	0.839	0.694	0.775	0.846	0.687	0.823	0.760	0.733
n_2	4	5	5	7	5	5	6	6
R_3	0.802	0.755	0.801	0.688	0.780	0.835	0.840	0.870
n_3	4	4	5	5	4	4	4	4
R_4	0.741	0.761	0.789	0.891	0.787	0.842	0.778	0.822
n_4	5	5	5	3	4	4	5	5
R_s	0.4590	0.6252	0.4118	0.5321	0.4076	0.6590	0.5344	**0.4002**
$C_{s-\min}$	20.462	25.195	18.743	19.657	20.557	**27.958**	**17.454**	20.98
R_1	0.645	0.694	0.644	0.613	0.641	0.691	0.577	0.635
n_1	2	2	2	2	2	2	3	2
R_2	0.681	0.589	0.647	0.632	0.563	0.675	0.576	0.605
n_2	1	2	1	2	2	2	2	2
R_3	0.688	0.644	0.616	0.658	0.655	0.710	0.661	0.627
n_3	2	2	2	2	1	2	2	1
R_4	0.620	0.630	0.619	0.576	0.657	0.667	0.549	0.644
n_4	2	3	2	2	2	2	2	2

Table 4. The solutions obtained with the proposed algorithm

Number	*[(R₁, n₁) (R₂, n₂) (R₃, n₃) (R₄, n₄)]*	R_s	C_s	W_s	V_s
1	[(0.880 36.6) (0.856 32.5) (0.912 45.4) (0.857 68.5)]	0.999 82	299.61	475.20	184
2	[(0.874 22.6) (0.851 09.5) (0.906 26.4) (0.851 72.5)]	0.999 77	279.30	475.20	184
3	[(0.865 18.6) (0.841 25.5) (0.895 16.4) (0.841 61.5)]	0.999 67	249.59	475.20	184
4	[(0.865 18.6) (0.841 25.5) (0.895 16.4) (0.841 61.5)]	0.999 63	239.98	475.20	184
5	[(0.857 26.6) (0.833 95.5) (0.886 55.4) (0.833 77.5)]	0.999 57	229.57	475.20	184
6	[0.854 82.6) (0.831 85.5) (0.884 00.4) (0.831 50.5)]	0.999 54	224.15	475.20	184
7	[(0.850 40.6) (0.828 01.5) (0.879 37.4) (0.827 36.5)]	0.999 47	214.81	475.20	184
8	[(0.837 88.6) (0.819 77.5) (0.869 37.4) (0.816 49.5)]	0.999 29	194.08	475.20	184
9	[(0.833 61.6) (0.813 67.5) (0.861 97.4) (0.812 12.5)]	0.999 16	184.82	475.20	184
10	[(0.827 74.6) (0.809 98.5) (0.857 33.4) (0.806 64.5)]	0.999 04	176.70	475.20	184
11	[(0.817 67.6) (0.801 81.5) (0.847 66.4) (0.797 47.5)]	0.998 78	163.03	475.20	184
12	[(0.816 89.6) (0.800 35.5) (0.845 76.4) (0.796 32.5)]	0.998 73	161.31	475.20	184
13	[(0.799 23.6) (0.785 60.5) (0.827 79.4) (0.780 11.5)]	0.998 09	140.99	475.20	184
14	[(0.791 27.5) (0.776 54.5) (0.817 98.4) (0.7711 4.5)]	0.997 32	125.94	418.57	173
15	[(0.781 52.5) (0.769 22.5) (0.808 13.4) (0.762 93.5)]	0.996 75	118.23	418.57	173
16	[(0.768 82.5) (0.758 36.5) (0.794 79.4) (0.750 29.5)]	0.995 77	108.17	418.57	173
17	[(0.724 97.5) (0.719 42.5) (0.748 10.4) (0.710 21.5)]	0.990 65	82.322	418.57	173
18	[(0.726 00.4) (0.721 80.4) (0.750 97.4) (0.709 96.4)]	0.977 61	66.464	293.57	128
19*	[(0.749 15.4) (0.737 74.4) (0.770 44.3) (0.734 77.4)]	0.974 49	75.020	257.40	107
20*	[(0.720 73.5) (0.719 59.4) (0.747 14.3) (0.705 17.4)]	0.968 71	66.964	296.87	116
21	[(0.692 84.5) (0.695 54.4) (0.718 44.3) (0.678 49.4)]	0.95630	57.194	296.87	116
22	[(0.669 12.4) (0.672 00.4) (0.690 53.3) (0.577 30.4)]	0.934 63	48.012	257.40	107
23	[(0.670 85.4) (0.676 83.3) (0.695 53.3) (0.657 53.3)]	0.890 68	39.612	198.61	79
24	[(0.666 18.4) (0.672 81.3) (0.690 67.3) (0.653 12.3)]	0.886 18	38.666	198.61	79
25	[(0.624 44.4) (0.637 09.3) (0.647 30.3) (0.613 22.3)]	0.840 68	31.444	198.61	79
26	[(0.637 18.3) (0.643 63.3) (0.655 25.3) (0.620 42.3)]	0.822 33	30.974	171.48	72
27	[(0.627 17.3) (0.639 58.3) (0.650 36.3) (0.616 07.3)]	0.816 19	30.293	171.48	72
28	[(0.608 21.3) (0.622 00.3) (0.630 20.3) (0.598 27.3)]	0.789 73	27.698	171.48	72
29	[(0.603 55.3) (0.618 88.3) (0.625 34.3) (0.593 86.3)]	0.782 97	27.109	171.48	72
30	[(0.598 93.3) (0.614 83.3) (0.620 44.3) (0.585 91.3)]	0.776 13	26.540	171.48	72

Note: "*" stands for dominated solutions.

population and the final boundary pop containing the two extreme solutions that have the $R_{s-\max}$ and the $C_{s-\min}$ of the final population, respectively. Table 3 shows that the individual having $R_{s-\max}$ in the initial boundary pop is replaced by one having a larger reliability value of 0.999 82 in the final population; the one having $C_{s-\min}$ in the initial boundary pop remains unchanged in the final population. Because of the improvement in at least one of the two members of the boundary pop, the proposed algorithm generates a wider range of solutions in the final population.

Using the proposed algorithm, we have obtained 30 solutions as shown in Table 4. From Table 4, we can see that 28 out of the 30 solutions are non-dominated solutions whereas the other two are dominated solutions. The 28 non-dominated solutions provide the Pareto frontier of the multi-objective optimization problem. They are plotted in Fig. 5 to show the obtained Pareto frontier.

To illustrate the performance of the proposed algorithm, we firstly compare it with the single-objective optimization method reported by Dhingra (1992). It consists of two steps because there are integer variables in the optimization problem to be solved. The first step is to use a non-linear

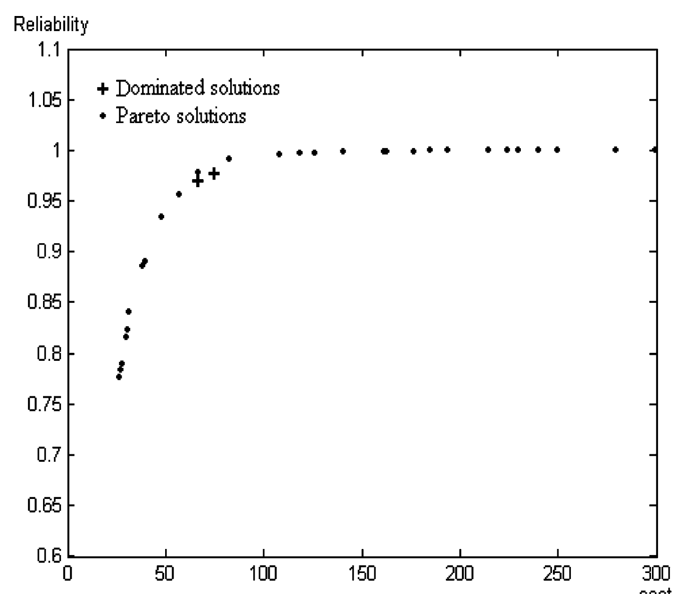

Fig. 5. Final solutions of the proposed algorithm in the objective space.

Table 5. Generating R_{lim} values for Dhingra's approach

Number of reliability ranges	*The proposed algorithm*			*Dhingra's approach*		
	Upper bound	*Lower bound*	*Number of solutions*	*Upper bound*	*Decremental value*	*Number of R_{lim} values*
1	0.999 82	0.999 04	10	0.999 82	0.0000 91	10
2	0.998 78	0.998 09	3	0.998 99	0.0004 95	3
3	0.997 32	0.990 65	4	0.997 99	0.0026 63	4
4	0.977 61	0.934 63	5	0.989 90	0.0149 75	5
5	0.890 68	0.816 19	5	0.92	0.03	5
6	0.789 73	0.776 13	3	0.79	0.02	3

constrained optimization algorithm by ignoring the integer requirements for the number of redundancies at each stage of the reliability design problem. Once a continuous optimal solution is obtained, a heuristic algorithm is used in the second step to handle the integer requirements for the number of redundancies. In our implementation of Dhingra's approach, we used the Matlab function "fmincon" in the first step of the approach. This function uses a gradient-based search algorithm. The heuristic algorithm in the second step as documented in Dhingra (1992) was coded in Matlab.

Since Dhingra's approach is for single-objective optimization problems, we need to convert one of the objective functions given in Equations (1) and (2) into a constraint. We have elected to convert the reliability objective given in Equation (1) into a constraint. To generate the Pareto frontier using Dhingra's approach, we need to specify a range of minimum system reliability values (R_{lim}) to minimize the system cost. We need to specify 30 such values to get 30 solutions so that we can compare them with those obtained with the proposed algorithm.

To specify 30 R_{lim} values in the series of single-objective optimization problems to be solved by Dhingra's approach, we used the optimal system reliability values obtained with the proposed algorithm as tabulated in Table 4 as a guide. We first divided these obtained reliability values into six ranges so that in each range there are at least three solutions generated by the proposed algorithm. Within each range, we specify a linear decrement amount in order to get the same number of solutions from Dhingra's approach as from the proposed algorithm. This is illustrated in Table 5. Based on Table 5, the specified 30 R_{lim} values are listed in the R_{lim} column in Table 6.

Trial runs of our implementation of Dhingra's approach showed that the program often diverged, especially when the specified R_{lim} value was high. We had to introduce a loop to automatically change the starting point for the algorithm to converge. In addition, this approach was unable to limit the range of the acceptable objective function values. On the other hand, the proposed algorithm does not have these disadvantages.

Using the sliding system reliability constraints, our implementation of Dhingra's approach generated the 30 solutions shown in Table 6. From Table 6, we can see that only 16 of the 30 solutions are non-dominated solutions.

The relative merit of the proposed algorithm and Dhingra's approach can be compared in two dimensions, the percentage of non-dominated solutions and the generated Pareto frontier. We desire the solutions to a multi-objective optimization problem to be non-dominated so that they can form the Pareto frontier. Thus, the percentage of non-dominated solutions is a good measure of the performance of an approach to a multi-objective optimization problem. Table 7 shows a comparison of the proposed algorithm with our implementation of Dhingra's approach. From this table, we can see that the proposed algorithm generates a much higher percentage of non-dominated solutions in comparison with our implementation of Dhingra's approach (93% versus 53.3%).

The Pareto frontier is composed of all non-dominated solutions obtained for a multi-objective optimization problem. The evenness of the solutions on the Pareto frontier is an important measure of the performance of the solution approach. Figure 6 displays the Pareto frontiers generated by our implementation of Dhingra's approach and the proposed algorithm. The solution with a system cost of 3048.54 obtained with Dhingra's approach is not displayed in Fig. 6, because the cost is too high.

Based on Fig. 6, we have the following observations.

1. The proposed algorithm provided a much smoother Pareto frontier than Dhingra's approach.
2. The solutions provided by the proposed algorithm are more evenly distributed on the Pareto frontier than Dhingra's approach.
3. When the system reliability is higher than 0.96, the Pareto frontier provided by the proposed algorithm dominates that provided by Dhingra's approach.
4. When the system reliability is below 0.96, the solutions provided by Dhingra's approach were better than those provided by the proposed algorithm. This shows that the proposed algorithm needs some further improvement.

Next, we compare the proposed algorithm with the multi-objective GA and the multi-objective hybrid GA approaches reported by Gen and Kim (1998, 1999a, 1999b).

Table 6. The solutions obtained with Dhingra's approach

Number	$[(R_1, n_1)\ (R_2, n_2)\ (R_3, n_3)\ (R_4, n_4)]$	R_{lim}	R_s	C_s	W_s	V_s
1	[(0.860 27.5) (0.959 64.3) (0.888 98.5) (0.850 47.6)]	0.999 82	0.999 85	627.93	470.66	190
2*	[(0.907 11.4) (0.970 62.3) (0.984 36.3) (0.948 74.3)]	0.999 73	0.999 76	1345.1	198.60	79
3	[(0.904 91.6) (0.947 97.4) (0.924 01.6) (0.987 06.3)]	0.999 64	0.999 99	3048.5	486.16	194
4	[(0.805 54.6) (0.910 61.6) (0.984 52.2) (0.838 77.5)]	0.999 55	0.999 60	560.37	471.22	170
5*	[(0.946 38.3) (0.968 10.3) (0.965 40.3) (0.976 51.3)]	0.999 46	0.999 76	1851.5	171.48	72
6	[(0.936 53.4) (0.883 80.4) (0.983 45.2) (0.879 40.5)]	0.999 37	0.999 50	533.68	279.01	110
7	[(0.843 09.5) (0.925 89.3) (0.907 87.4) (0.947 20.3)]	0.999 27	0.999 28	528.05	274.25	109
8*	[(0.940 54.3) (0.837 64.5) (0.847 39.6) (0.986 12.2)]	0.999 18	0.999 76	1810.3	381.01	175
9*	[(0.973 16.3) (0.630 47.8) (0.940 26.3) (0.961 48.3)]	0.999 09	0.999 36	905.94	488.04	182
10*	[(0.972 02.3) (0.689 83.7) (0.935 37.4) (0.923 58.3)]	0.999 00	0.999 24	582.80	411.24	173
11	[(0.750 06.6) (0.901 99.4) (0.785 37.5) (0.882 80.4)]	0.998 99	0.999 02	286.78	442.31	175
12	[(0.847 21.5) (0.786 12.5) (0.882 51.4) (0.777 92.5)]	0.998 50	0.998 74	155.57	418.56	173
13*	[(0.718 95.6) (0.984 85.2) (0.791 16.5) (0.811 22.5)]	0.998 00	0.998 64	1510.4	442.90	169
14*	[(0.990 75.2) (0.989 00.2) (0.966 59.6) (0.885 03.6)]	0.997 99	0.999 79	3907.8	442.92	192
15*	[(0.753 19.6) (0.954 28.2) (0.759 91.6) (0.980 99.2)]	0.995 33	0.997 13	1289.1	429.32	160
16*	[(0.609 40.7) (0.828 61.4) (0.773 42.5) (0.917 39.3)]	0.992 66	0.996 59	224.70	491.00	174
17*	[(0.625 26.6) (0.634 93.6) (0.929 57.3) (0.983 47.2)]	0.990 00	0.994 25	1282.4	396.57	143
18*	[(0.609 37.6) (0.971 11.2) (0.921 79.5) (0.772 71.4)]	0.980 00	0.992 96	610.28	396.85	151
19	[(0.704 65.5) (0.977 45.3) (0.580 55.6) (0.654 09.4)]	0.967 50	0.976 31	111.84	434.04	183
20*	[(0.953 03.2) (0.612 95.5) (0.750 00.4) (0.750 00.3)]	0.955 00	0.969 87	155.64	255.93	120
21	[(0.750 00.4) (0.500 00.6) (0.789 41.3) (0.500 00.6)]	0.942 50	0.956 19	44.64	465.61	187
22	[(0.506 44.6) (0.500 00.6) (0.874 99.3) (0.500 00.5)]	0.93	0.938 03	44.41	495.65	185
23	[(0.749 99.3) (0.500 00.5) (0.796 87.4) (0.500 00.5)]	0.92	0.922 24	39.77	351.96	157
24	[(0.500 00.5) (0.500 00.5) (0.750 00.3) (0.500 00.5)]	0.89	0.894 94	29.25	382.39	152
25	[(0.500 00.5) (0.500 00.4) (0.749 99.3) (0.500 00.5)]	0.86	0.866 07	26.97	342.91	134
26	[(0.500 00.5) (0.500 00.4) (0.500 00.6) (0.500 00.4)]	0.83	0.838 14	23.31	461.18	197
27*	[(0.500 00.4) (0.500 00.4) (0.500 00.6) (0.500 00.4)]	0.80	0.811 10	22.33	421.71	188
28	[(0.500 00.4) (0.500 00.4) (0.500 00.6) (0.500 00.4)]	0.79	0.811 10	22.33	421.71	188
29*	[(0.500 00.4) (0.500 00.3) (0.521 00.6) (0.500 00.5)]	0.77	0.785 08	22.72	440.62	192
30	[(0.500 00.4) (0.500 00.4) (0.513 24.6) (0.500 00.3)]	0.75	0.758 82	20.42	390.06	174

Note: "*" stands for dominated solutions.

These two methods will be referred to as GA and hybrid GA for simplicity. The performances of GA and hybrid GA are also shown in Table 7. From this table, we can see that the proposed algorithm provides a higher percentage of non-dominated solutions than GA and hybrid GA (93% versus 90%). Second, for the objective of system reliability maximization, we find that 12 of the 30 solutions in the final population by the proposed approach as shown in Table 4 are better than the best ones provided by hybrid GA (0.9984) and 17 of the 30 solutions from the proposed approach are better than the best ones provided by GA (0.9882). Third, the solutions from the proposed algorithm are distributed in a wider range of system reliability, namely [0.7761, 0.999 82]. The corresponding system reliability range for GA is [0.8813, 0.988 20] and for hybrid GA is [0.8667, 0.998 40]. These results lead us to believe that the proposed algorithm possesses a more powerful ability to search for non-dominated solutions in terms of the distribution range of the Pareto frontier. Even more significantly, such good results are realized in only 100 generations, as compared with the 2000 generations required by GA and hybrid GA.

Table 7. Performance comparisons

Algorithms	*The number of obtained solutions (A)*	*The number of non-dominated solutions (B)*	*Ratio (B/A) (%)*	*Generation*	R_{s-m}	R_{s-min}
Dhingra's approach	30	16	53.3	—	0.999 85	0.7588
GA	30	27	90	2000	0.988 20	0.8813
Hybrid GA	40	36	90	2000	0.998 40	0.8667
The proposed algorithm	30	28	93	100	0.999 82	0.7761

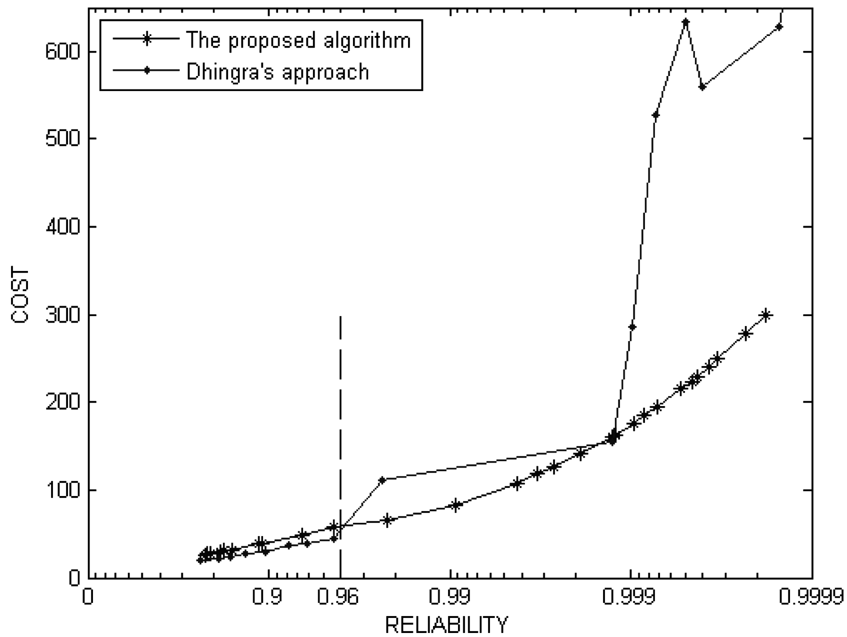

Fig. 6. Pareto frontiers from Dhingra's approach and the proposed algorithm.

7. Conclusions

In this paper, we propose a multi-objective optimization algorithm that integrates a niched Pareto GA with a population-based constraint handling approach. Our aim is to apply the algorithm in system reliability multi-objective optimization, and to obtain a population of Pareto solutions spread along the Pareto frontier. We first apply the proposed algorithm to a test problem and discover that it effectively finds and maintains the diversity of Pareto solutions. Then we use the proposed algorithm to solve a practical multi-objective optimization problem of allocating component reliability and redundancy for a four-stage series-parallel system. The optimal results are compared with those obtained through a single-objective optimization approach, multi-objective GA and multi-objective hybrid GA. We find that the proposed algorithm more effectively solves the practical multi-objective optimization problem.

Acknowledgements

The authors would like to thank the referees and the editors for their critical comments and constructive suggestions. This research was partially supported by the National Natural Science Foundation of China under contract number 50775026, the National High Technology Research and Development Program of China (863 Program) under contract number 2007AA04Z403, the Specialized Research Fund for the Doctoral Program of Higher Education of China under contract number 20060614016, and the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Coello, C. and Carlos, A. (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, **191**(11–12), 1245–1287.

Coit, D.W. and Konak, A. (2006) Multiple weighted objectives heuristic for the redundancy allocation problem. *IEEE Transactions on Reliability*, **55**(3), 551–558.

Deb, K. (2000) An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, **186**(2), 311–338.

Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002) A fast and elitist multiobjective genetic algorithm. NSGA-II. *IEEE Transactions on Evolutionary Computation*, **6**(2), 182–197.

Dhingra, A.K. (1992) Optimal apportionment of reliability & redundancy in series systems under multiple objectives. *IEEE Transactions on Reliability*, **41**(4), 576–582.

Erickson, M. and Horn, J. (2002) Multi-objective optimal design of groundwater remediation system: Application of the niched Pareto genetic algorithm. *Advances in Water Resources*, **25**(1), 51–65.

Fonseca, C.M. and Fleming, P.J. (1995) An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, **3**(1), 1–16.

Gen, M. and Cheng, R.W. (2000) *Genetic Algorithms and Engineering Optimization*, Wiley, New York, NY.

Gen, M. and Kim, J.R. (1998) Bicriteria reliability design using hybrid genetic algorithm, in *Proceedings of the Sixth Europe Congress on Intelligent Techniques and Soft Computing*, Aachen, Germany, **1**, pp. 407–412.

Gen, M. and Kim, J.R. (1999a) GA-based reliability design: state-of-the-art survey. *Computers and Industrial Engineering*, **37**, 151–155.

Gen, M. and Kim, J.R. (1999b) GA-based optimization of reliability design, in *Evolutionary Design by Computers*, P. Bentley (ed.), Morgan Kaufman, San Francisco, CA, pp.191–218.

Ha, C.H. and Kuo, W. (2006) Multi-path heuristic for redundancy allocation: the tree heuristic. *IEEE Transactions on Reliability*, **55**(1), 37–43.

Horn, J. and Nafpliotis, N. (1993) Multiobjective optimization using the niched Pareto Genetic Algorithm. Technical Report, Illinois Genetic Algorithm Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, USA.

Horn J., Nafpliotis, N., and Goldberg, D.E. (1994) A niched Pareto genetic algorithm for multiobjective optimization. in *Proceedings of the First IEEE Conference on Revolutionary Computation and the IEEE World Congress on Computational Intelligence*, IEEE, Piscataway, NJ, **1**, pp. 82–87.

Huang, H.Z. (1997) Fuzzy multi-objective optimization decision-making of reliability of series system. *Microelectronics and Reliability*, **37**(3), 447–449.

Huang, H.Z., Tian, Z. and Gu, Y. (2004) Reliability and redundancy apportionment optimization using interactive physical programming. *International Journal of Reliability, Quality & Safety Engineering*, **11**(3), 213–222.

Huang, H.Z., Tian, Z.G. and Zuo, M.J. (2005) Intelligent interactive multiobjective optimization method and its application to reliability optimization. *IIE Transactions*, **37**(11), 983–993.

Huang, H.Z., Wang, P., Zuo, M.J., Wu, W.D. and Liu, C.S. (2006) A fuzzy set based solution method for multiobjective optimal design problem of mechanical and structural systems using functional-link net. *Neural Computing & Applications*, **15**(3–4), 239–244.

Huang, H.Z., Wu, W.D. and Liu, C.S. (2005) A coordination method for fuzzy multi-objective optimization of system reliability. *Journal of Intelligent and Fuzzy Systems*, **16**(3), 213–220.

Kuo, W. and Prasad, V.R. (2000) An annotated overview of system-reliability optimization. *IEEE Transactions on Reliability*, **49**(2), 176–187.

Li, Y.X. (1996) Multi-criteria optimization techniques and its application to engineering design problems, Master's thesis, Department of Industrial and Information Systems Engineering, Ashikaga Institute of Technology, Japan.

Liang, Y.C. and Chen, Y.C. (2007) Redundancy allocation of series-parallel systems using a variable neighborhood search algorithm. *Reliability Engineering and System Safety*, **92**(3), 323–331.

Onishi, J., Kimura, S., James, R. and Nakagawa, Y. (2007) Solving the redundancy allocation problem with a mix of components using the improved surrogate constraint method. *IEEE Transactions on Reliability*, **56**(1), 94–101.

Rao, S.S. and Dhingra, A.K. (1992) Reliability and redundancy apportionment using crisp and fuzzy multiobjective optimization approaches. *Reliability Engineering and System Safety*, **37**(3), 253–261.

Salazar, D., Rocco, C.M. and Galvan, B.J. (2006) Optimization of constrained multiple-ojective reliability problems using evolutionary algorithms.*Reliability Engineering and System Safety*, **91**(9), 1083–1094.

Yun, W., Song, Y.M. and Kim, H.G. (2007) Multiple multi-level redundancy allocation in series systems.*Reliability Engineering and System Safety,* **92**(3), 308–313.

Zheng, D.X.M., Ng, S.T. and Kumaraswamy, M.M. (2005) Applying Pareto ranking and niche formation to genetic algorithm-based multiobjective time-cost optimization. *Journal of Construction Engineering and Management*, **131**(1), 81–91.

Biographies

Hong-Zhong Huang is a full professor and the Dean of the School of Mechanical, Electronic, and Industrial Engineering at the University of Electronic Science and Technology of China, Chengdu, Sichuan, China. He has held visiting appointments at several universities in Canada, USA and elsewhere in Asia. He has a Ph.D. in Reliability Engineering from Shanghai Jiaotong University, China. He has published 150 journal articles and five books in the fields of reliability engineering, optimization design, fuzzy sets theory and product development. He is a Regional Editor of the *International Journal of Reliability and Applications*, an Editorial Board Member for the *International Journal of Reliability, Quality and Safety Engineering*, *International Journal of Reliability and Quality Performance*, *International Journal of Performability Engineering*, *Advances in Fuzzy Sets and Systems* and *The Open Mechanical Engineering Journal*. He received the William A. J. Golomski Award from the Institute of Industrial Engineers in 2006. His current research interests include system reliability analysis, warranty, maintenance planning and optimization, and computational intelligence in product design.

Jian Qu is a Ph.D. student in the Department of Mechanical Engineering at the University of Alberta, Canada. He received a B.S. in 2003 and an M.S. in 2006, both in Mechanical Engineering at the Dalian University of Technology, Dalian, China. His research interests include reliability engineering and optimization.

Ming J. Zuo received an M.Sc. in 1986 and a Ph.D. in 1989, both in Industrial Engineering at Iowa State University, Ames, Iowa, USA. He is a Professor in the Department of Mechanical Engineering at the University of Alberta, Canada. His research interests include system reliability analysis, maintenance planning and optimization, signal processing and fault diagnosis. He is an Associate Editor of *IEEE Transactions on Reliability*. He is a senior member of IEEE and IIE.