Received January 2001
Revised August 2001
Accepted August 2001

Pareto-based continuous evolutionary algorithms for multiobjective optimization

Mun-Bo Shim and Myung-Won Suh
School of Mech. Engineering, The University of Sungkyunkwan, Kyoungki-Do, South Korea

Tomonari Furukawa and Genki Yagawa
Quantum Engineering and Systems Science, The University of Tokyo, Tokyo, Japan

Shinobu Yoshimura
Institute of Environmental Studies, The University of Tokyo, Tokyo, Japan

Keywords *Optimization, Algorithms*

Abstract *In an attempt to solve multiobjective optimization problems, many traditional methods scalarize an objective vector into a single objective by a weight vector. In these cases, the obtained solution is highly sensitive to the weight vector used in the scalarization process and demands a user to have knowledge about the underlying problem. Moreover, in solving multiobjective problems, designers may be interested in a set of Pareto-optimal points, instead of a single point. In this paper, Pareto-based Continuous Evolutionary Algorithms for Multiobjective Optimization problems having continuous search space are introduced. These algorithms are based on Continuous Evolutionary Algorithms, which were developed by the authors to solve single-objective optimization problems with a continuous function and continuous search space efficiently. For multiobjective optimization, a progressive reproduction operator and a niche-formation method for fitness sharing and a storing process for elitism are implemented in the algorithm. The operator and the niche formulation allow the solution set to be distributed widely over the Pareto-optimal tradeoff surface. Finally, the validity of this method has been demonstrated through some numerical examples.*

1. Introduction

Much work has been done in engineering optimization so far, but the problems tackled have only been confined to ideal and unrealistic problems, rather than real-world applications. One of the reasons for this has been that for many years only single objective functions were considered, while this is not a realistic assumption since most real-world problems have several (possibly

This study is a part of ADVENTURE project sponsored by the Japan Society for the Promotion of Science and the authors are grateful for the support provided by a grant from the Korea Science & Engineering Foundation (KOSEF) and Safety and Structural Integrity Research Center at the Sungkyunkwan University.

Engineering Computations,
Vol. 19 No. 1, 2002, pp. 22–48.

DOI 10.1108/02644400210413658

conflicting) objectives. This situation has led designers to make decisions and trade-offs based on their experience, instead of using some well-defined optimality criterion.

In a typical multiobjective optimization problem, there exists a solution space which is superior to the rest of solutions in the search space but the solution, when every objective is considered, may be inferior to other solutions in the space in one or more objectives. These solutions are known as Pareto-optimal solutions (Chankong and Haimes, 1983).

Since none of these Pareto-optimal solutions can be identified as better than others without any further consideration, the goal for multiobjective optimization is to find as many and wide Pareto-optimal solutions as possible. Once such solutions are found, it usually requires higher-level decision-making with other considerations to choose one of them for implementation. In this paper, we concentrate on the task of efficiently finding Pareto-optimal solutions.

In dealing with multiobjective optimization problems, classical optimization methods, for examples, weighted sum methods, goal programming, min–max methods etc. are not efficient, simply because most of them cannot find multiple solutions in a single run, thereby requiring them to be applied as many times as the number of desired Pareto-optimal solutions, and multiple applications of these methods do not guarantee finding widely spaced Pareto-optimal solutions.

On the contrary, the studies on evolutionary algorithms, over the past few years, have shown that these methods can be efficiently used to eliminate most of the above difficulties of classical methods (Srinivas and Deb, 1994; Fonseca and Fleming, 1993; Horn *et al.*, 1994). Since they use a population of solutions in their search, multiple Pareto-optimal solutions can, in principle, be found in one single run. However, not only they try to find several Pareto-optimal solutions but also they can solve simple problems.

In this study, Pareto-based Continuous Evolutionary Algorithms for Multi-objective Optimization (MOPCEAs) are introduced. For better efficiency and robustness, a progressive reproduction operator and a niche-formation method for fitness sharing and a storing process for elitism are implemented in these algorithms with continuous search space. The progressive reproduction operator implemented has a potential to search both local area and global one in the balance. The niche-formation method developed in this paper works better than others in multiobjective optimization technique because of considering both function space and parameter one. The storing process is a very efficient approach in multiobjective optimization to sustain the solution set found as well as to accelerate a search process. Finally, these operators allow the solution set to be distributed widely over the Pareto-optimal tradeoff surface, that is, the solution space to be found.

The paper is structured as follows: Section II introduces key concepts of multiobjective optimization and discusses the previous work of multiobjective optimization. Section III describes MOPCEAs, which is newly proposed, having continuous search space and section IV applies MOPCEAs to test problems and presents analytical results. Discussions of the results as well as summaries are given in the last section.

2. Multiobjective optimization

2.1 Statement of the problem

The multiobjective optimization problem can be defined as follows:

Find the vector $X^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ that will satisfy m inequality constraints:

$$g_i(X) \geq 0, \quad i = 1, 2, \ldots, m \tag{1}$$

p equality constraints

$$h_i(X) = 0, \quad i = 1, 2, \ldots, p \tag{2}$$

and optimize the vector function

$$F(X) = [f_1(X), f_2(X), \ldots, f_k(X)]^T \tag{3}$$

where $X = [x_1, x_2, \ldots, x_n]^T$ is the vector of decision variables. In other words, we wish to determine the particular set $x_1^*, x_2^*, \ldots, x_n^*$ which yields the optimum values of all the objective functions among the set of all points that satisfy (1) and (2).

While the single objective optimization tries to look for a single solution, multiobjective optimization derives a solution set, and this corresponds to the Pareto-optimality, which was introduced in the field of economics a century ago. Consider a problem where we have k objective functions, $f_i : R^n \rightarrow R$

$$F(X)^T = [f_1(X), f_2(X), \ldots, f_k(X)] \rightarrow \min_x \tag{4}$$

A decision vector $X_u \in R^n$ is said to be Pareto-optimal if and only if there is no vector $X_v \in R^n$ for which $v = F(X_v) = [v_1, \ldots, v_n]$ dominates $u = F(X_u) = [u_1, \ldots, u_n]$, i.e. if no other feasible solution exists which yields an improvement in one criterion without causing a deterioration of another criterion.

Figure 1 illustrates an example where points 8 and 12 satisfy Pareto-optimality. The set of all Pareto-optimal decision vectors is called the

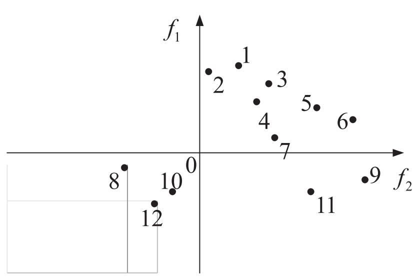


Figure 1.
Pareto-optimal set

Pareto-optimal, efficient, or admissible set of the problem. The corresponding set of objective vectors is called the non-dominated set. The notion of Pareto-optimality is only a first step towards the practical solution of a multiobjective problem, which usually involves the choice of a single compromise solution from the non-dominated set according to some preference information.

2.2 Classical method

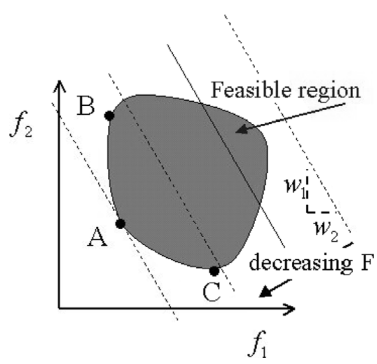
A common difficulty in multiobjective optimization is the existence of an objective conflict—none of the feasible solutions allow simultaneous optimal solutions for all objectives. In other words, optimal solutions of one objective are usually different from those of others. Thus the most favorable Pareto-optimum set are mathematically the solutions which offer the least objective conflict. Such solutions can be viewed as points in the search space which are placed from each objective. To find such points most of classical methods generally scalarize the objective vector into one objective.

Weighted sum method is probably the simplest of all calssical techniques. Multiple objective functions are combined into one overall objective function F, as follows:

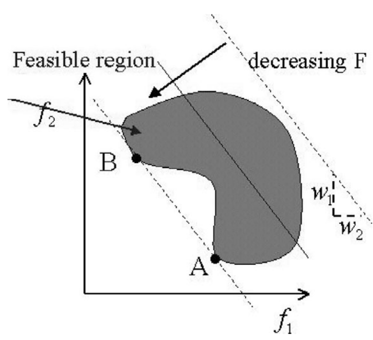
$$\text{Minimize } F = \sum_{j=1}^{k} \omega_j f_j(X) \quad \text{where } X \in \Omega, \;\; \Omega - \text{feasible region} \tag{5}$$

where ω_j is the weight used for the j-th objective function $f_j(X)$. Usually, non-zero fractional weights are used so that sum of all weights $\sum_{j=1}^{k} \omega_j$ is equal to one.

All Pareto-optimal solutions must lie in the feasible region Ω. In this method, the optimal solution is controlled by the weight vector. The working of this method is illustrated in a hypothetical problem shown in Figure 2. The figure shows the feasible search space in the function space, having two objectives.



(a) A convex Pareto-optimal front



(b) A non-convex Pareto-optimal front

Figure 2.
A convex and non-convex Pareto-optimal front

Any point inside the feasible region represents a solution X having two objective function values f_1 and f_2. With fixing a weight vector, optimizing (5) means finding a hyper-plane (a line for two objective functions) with a fixed orientation in the function space. The optimal solution is the point where the hyper-plane has a common tangent with the feasible space boundary. This solution is shown as the point A in the figure, where the line with intercepts at f_1 and f_2 axes in proportions of ω_2 and ω_1, respectively, is tangent to the feasible region. One can imagine finding other solutions such as B or C, with choosing a different weight vector and finding the corresponding common tangent point again. A collection of such solutions constitutes the Pareto-optimal set. Not only such a simple strategy is intuitively a computationally expensive method but also there is a major difficulty with such a method.

This method cannot be used to find Pareto-optimal solutions in multi-objective optimization problems having a nonconvex Pareto-optimal front. In Figure 2(a), the Pareto-optimal front is convex. However, one can also think of multiobjective optimization problems having a nonconvex Pareto-optimal front (Figure 2(b)). In Figure 2(b), finding the tangents closest to the origin with fixing a different weight vector do not give rise to finding different Pareto-optimal solutions. For every weight vector, only the solutions A or B will be found, and all other Pareto-optimal solutions within A and B cannot be found.

Although there exists a few other methods such as ϵ-perturbation method, goal programming, min-max method, most methods demand some knowledge about the problem being solved and some methods are sensitive to the shape of the Pareto-optimal front. The most profound difficulty with all these methods is that all need to be applied many times, hopefully finding one different Pareto-optimal solution each time. This makes the methods unattractive and this is one of the reasons why multiobjective optimization problems are mostly avoided in practice.

2.3 Multiobjective evolutionary algorithms

The above difficulty can be handled by using Evolutionary Algorithms (EAs). EAs seem to be especially suited to multiobjective optimization because they are able to capture multiple Pareto-optimal solutions in a single run of simulation and may exploit similarities of solutions by recombination. Several evolutionary approaches were categorized regarding plain aggregating approaches (Hajela and Lin, 1992), population-based non-Pareto approaches (Schaffer, 1985) and Pareto-based approaches (Fonseca and Fleming, 1993; Horn *et al*., 1994; Srinivas and Deb, 1994).

Aggregation methods combine the objectives into a single parameterized objective function; however, the parameters of this function are not changed for different optimization runs, but instead systematically varied during the same run. Hajela and Lin (1992), for instance, use the weighting method. Population-based non-Pareto approaches switch between the objectives during the selection phase instead of combining the objectives into a single scalar fitness value. Each time an individual is chosen for reproduction, potentially a different objective will decide which member of the population will be copied into the mating pool. For example, Schaffer (1985) proposed filling equal portions of the mating pool according to the distinct objectives. The Pareto-based approaches of calculating an individual's fitness on the basis of Pareto dominance was first suggested by Goldberg (1989). First all nondominated individuals are assigned rank one and temporarily removed from the population. Then, the next nondominated individuals are assigned rank two and so forth. Finally, the rank of an individual determines its fitness value.

The publication of the above-mentioned algorithms showed the superiority of evolutionary multiobjective optimization techniques to classical methods. However, there are demands on more efficient search algorithm for two tasks that should be achieved in a multiobjective optimization problem, that is (1) how to accomplish fitness assignment and selection, respectively, in order to guide the search toward the Pareto-optimal set; (2) how to maintain a diverse population in order to prevent premature convergence and achieve a well distributed and well spread nondominated set.

Since preservation of diversity is crucial in the field of multiobjective optimization, many multiobjective EAs incorporate niching techniques, the most frequently implemented of which is fitness sharing. Fitness sharing is based on the idea that individuals in a particular niche have to share the resources available, similar to nature. Thus, the fitness value of a certain individual is the more degraded the more individuals are located in its neighborhood. Neighborhood is defined in terms of a distance represented by the so-call niche radius σ_{share}. Sharing can be performed both in genotypic space and phenotypic space.

Several researchers have thought GAs with suitable modification in their operators have worked well to solve many multiobjective optimization problems with respect to above two tasks. However, we have felt a lack of their performance because they use the binary scheme basically (Furukawa and Dissanayake, 1993; Deb and Kumar, 1995; Furukawa and Yagawa, 1997). In addition, their operators are improved/added to guide the search toward the Pareto-optimal set and to maintain a diverse population. In the following, we describe new Pareto-based evolutionary implementation in detail.

3. Pareto-based continuous evolutionary algorithms for multiobjective optimization (MOPCEAs)

3.1 Schematic view

Pareto-based continuous evolutionary algorithms for multiobjective optimization (MOPCEAs) are characterized as follows.

(1) Continuous Search: recombination and mutation are conducted on a continuous base so that optimization problems with continuous search space can be solved efficiently, each individual thereby being represented by a real vector.

(2) Continuous Evaluation: niche formation with continuous search space is adopted in order to have well-distributed Pareto-optimal space.

(3) Historical Storage: Pareto-optimal solutions obtained by the previous generation are stored so that a concrete configuration of solution space can be seen and convergence to Pareto-optimal set can be fast.

Figure 3 shows the fundamental structure of MOPCEAs proposed by the authors.

First, a population of individuals, each represented by a continuous vector, is initially (generation $t = 0$) generated at random, i.e.

$$P_\lambda^t = \{x_1^t, \ldots, x_\lambda^t\} \in (R^n) \tag{6}$$

where λ and n represent the population size of parental individuals and the space of individual respectively. Each vector thus represents a search point, which corresponds to the phenomenological representation of individual, that is, the phenotype.

3.2 Continuous search

The definition of the recombination and mutation becomes the probabilistic distribution of the phenomenological measures accordingly. In the recombination, parental individuals breed offspring individuals by combining part of the information from the parental individuals, thereby creating new points inheriting some information from the old points. The recombination operation is then defined as

$$\begin{cases} x_\alpha^{t+1} = (1-\mu_\alpha^t)x_\alpha^t + \mu_\beta^t x_\beta^t \\ x_\beta^{t+1} = \mu_\alpha^t x_\alpha^t + \left(1-\mu_\beta^t\right)x_\beta^t \end{cases} \tag{7}$$

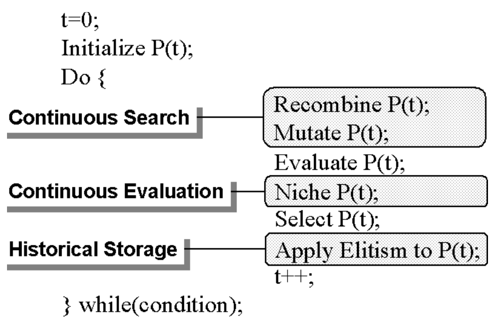


Figure 3.
Flowchart of MOPCEAs

where x_α^t and x_β^t are parental individuals at generation $t$and parameter μ_i^t may be defined by the normal distribution with mean 0 and standard deviation σ:

$$\mu_i^t = N(0, \sigma^2) \tag{8}$$

In many studies (Obayashi, 1998; Qing *et al.*, 1999), the same μ is used regardless of each parental individual, that is, the symmetric recombination. Symmetric distribution sometimes leads to good convergence for just unimodal, simple problem, while asymmetric one improves robustness of the algorithm in multiobjective optimization.

Figure 4 illustrates the difference of the recombination methods, respectively. The parental point is marked with "o" and the offspring point that is possible is marked with "*". The crossover operator in GAs uses variables coded using binary strings of size 7, respectively and the crossover point is changed from 1 to 20 bit. The symmetric and asymmetric recombination uses the normal distribution and the offspring point is selected randomly. From the above figure, we find the fact that asymmetric one is more effective because the offspring point is born suitably inside and outside the parental point, that is, both local search and global search are possible.

The mutation can also be achieved simply by implementing

$$x^{t+1} = \mathrm{rand}(x_{\min}, x_{\max}) \tag{9}$$

where $(x_{\min}, x_{\max})$ is the boundary of the independent parameters. Note that the mutation may not be necessary since it can allow individuals to alter largely with small possibility, when the coefficient μ_i^t is large.

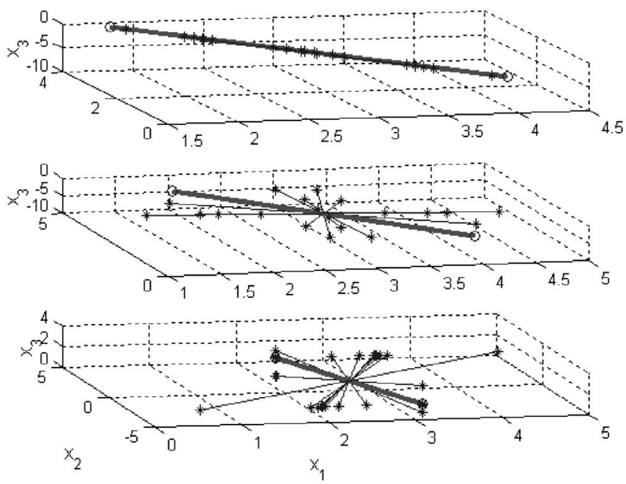


Figure 4.
Comparison of symmetric, asymmetric and genetic crossover

3.3 Continuous evaluation

The main operation of MOPCEAs to maintain a diverse population is composed of the ranking selection method used to emphasize good population in function space and the sharing method used to maintain stable sub-populations in parameter space. Besides, an elitism method is used to converge quickly. MOPCEAs emphasize the way the selection operator works. Figure 5 shows the process of the niche formation in MOPCEAs.

Before the selection is performed, the population is ranked as follows. As the Pareto-optimal set is to be found as solutions, the ranking process of individuals is composed of an elimination rule. In the rule, all the points are first concerned and the Pareto-optimal set is ranked No. 1. The points in rank No. 1 are then eliminated, and the points in No. 2 are ranked as the second Pareto-optimal set, and all the other ranked are generated stepwise in the same fashion (Goldberg, 1989). The points in rank No. *k*, *G*(*k*), are defined as

$$G(k) = P_i | \mathrm{rank}(P_i) = k, \forall i \in \{1, \ldots, \lambda\}\} \tag{10}$$

The ranked points are illustrated in Figure 6 and shadowed areas represent search areas of the points in group No. 3.

Figure 7 illustrates the evaluation of the fitness of each individual. The evaluation process starts with finding the best and worst objective function value of each objective:

```
k=1;
P'(k)=P(t);
Do {
        Find G(k) from P'(k);
        Calculate Φ(P'(k)∈G(k));
        Share P'(k)∈G(k);
        If k=1
            Assign P'(k)∈G(k) to storing set;
        Eliminate G(k) from P'(k);
        k++;
} while(P'(k)!=0);
```

Figure 5.
The process of the niche formation

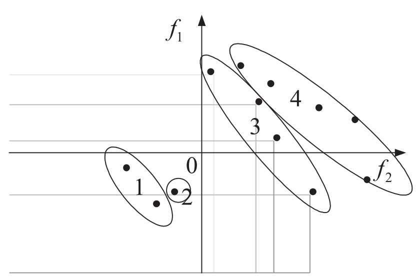


Figure 6.
Ranks of individuals

$$f_{\text{best}j} = \min\{f_j(P_i)|\forall i \in \{1,\ldots\lambda\}\} \tag{11}$$

and

$$f_{\text{worst}j} = \max\{f_j(P_i)|\forall i \in \{1,\ldots,\lambda\}\} \tag{12}$$

If we temporarily define the fitness as

$$\Phi'_j(P_i) = \frac{f_{\text{worst}j} - f_j(P_i)}{f_{\text{worst}j} - f_{\text{best}j}} \tag{13}$$

we can get the normalized conditions:

$$0 \leq \Phi'_j \leq 1 \tag{14}$$

This allows us to treat the fitness of each function with the same scale. This means the fitness value according to the relative location in function space. The fitness of each objective function is thus defined as:

$$\Phi_j(P_i) \equiv \Phi_j^{G(k)}(P_i) = \max\{\Phi'_j(P_i)|P_i \in G(k)\} \tag{15}$$

The fitness of each individual can be conclusively calculated as:

$$\Phi(P_i) = \sum_{j=1}^{m} \Phi_j(P_i) \tag{16}$$

where m is the total number of objective function and which has the range

$$0 \leq \Phi \leq m \tag{17}$$

In the ranking process above, the same fitness value is assigned to give an equal selective potential to all individuals ranked in the same group. In order to maintain a diverse population, these ranked individuals are then shared by the sharing process. Sharing methods are discussed in elsewhere (Goldberg and

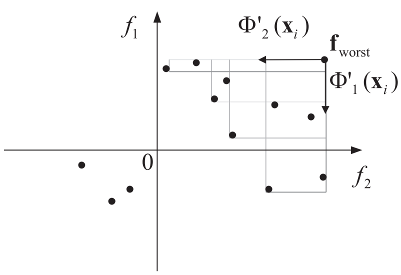


Figure 7.
Evaluation of individuals

Richardson, 1987; Deb and Goldberg, 1989). Sharing is achieved by selection operation using degraded fitness values which are obtained by dividing the scaled fitness value of an individual by a quantity proportional to the number of individuals around it in parameter space. This causes multiple optimal points to co-exist in the population. The ranked population is shared as follows. Given a set of n_k solutions in the k-th Pareto-set each having a scaled fitness value Φ_{k}, the sharing procedure computes a normalized Euclidean distance measure with another solution j for each solution $i = 1, 2, \ldots, n_k$.

$$d_{ij} = \sqrt{\sum_{p=1}^{n} \left(\frac{x_p^i - x_p^j}{x_p^u - x_p^l} \right)^2} \tag{18}$$

where n is the number of variables in the problem. The parameters x_p^u and x_p^l are the upper and lower of variable x_p. This distance d_{ij} is compared with a pre-specified parameter σ_{share} and the following sharing function value is computed (Deb and Goldberg, 1989).

$$Sh(d_{ij}) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma_{\mathrm{share}}} \right)^2 & \text{if } d_{ij} \leq \sigma_{\mathrm{share}} \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

where a pre-specified parameter σ_{share} can be calculated as follows (Srinivas and Deb, 1994):

$$\sigma_{\mathrm{share}} \approx \frac{0.5}{\sqrt[q]{n}} \tag{20}$$

where q is the desired number of distinct Pareto-optimal solutions. Although the calculation of σ_{share} depends on this parameter q in other method, that is, GAs that has an encoding operator, it has been shown that the use of above equation works in MOPCEAs. $Sh(d_{ij})$ is calculated until j is equal to n_k and niche count for i-th solution is calculated.

$$m_i = \sum_{j=1}^{n_k} Sh(d_{ij}) \tag{21}$$

The scaled fitness Φ_k of i-th solution in the k-th Pareto-set is degraded to calculate the shared fitness Φ_i^*.

$$\Phi_i^* = \frac{\Phi_k}{m_i} \tag{22}$$

This procedure is continued for all $i = 1, 2, \ldots, n_k$ and a corresponding Φ_i^* is found. Also the ranking process with the sharing method is continued until the entire population is classified into several ranks. The shared fitness obtained in the above ranking is very effective for the selection operation to maintain diversity in the population because it is considered both in function space and in parameter space. Figure 8 illustrates the evaluated fitness before and after the sharing process.

In this procedure the individuals ranked No. 1 are used so that a concrete configuration of solution space can be formed by using historical storage and convergence to the Pareto-optimal set can be fast. In the next section, this implementation is explained in detail.

3.4 Historical storage

Here, one of new approaches to multiobjective optimization, historical storage is introduced. Although single objective optimization problems may have a unique optimal solution, multiobjective optimization problems present a possibly uncountable set of solutions, that is, a solution space. Historical storage uses new techniques in order to form the solution space, that is, Pareto-optimal space.

Figure 9 shows the process of historical storage. During a niche formation, a set of Pareto-optimal solutions is determined at the current generation. The set is inserted into the select pool with Pareto-optimal solutions found so far, that is, a set of externally stored solutions, in an attempt to find nondominated solutions among them. Nondominated solutions are found by the ranking process externally. Finally, a new set of externally stored solutions participates in the elitism process. They cause convergence to Pareto-optimal set to be fast and a diverse population to be maintained.

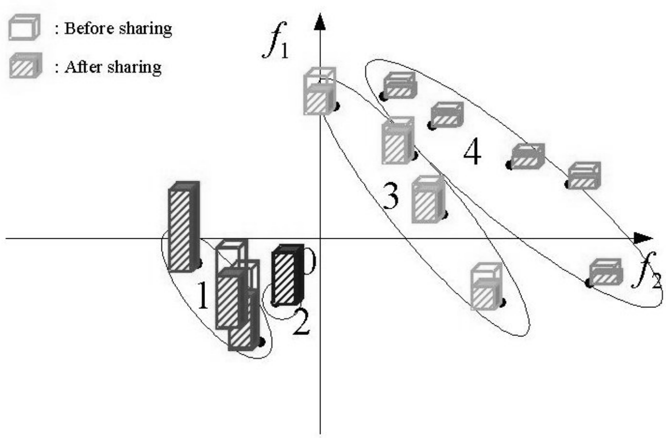


Figure 8. Comparison of the fitness before and after the sharing process

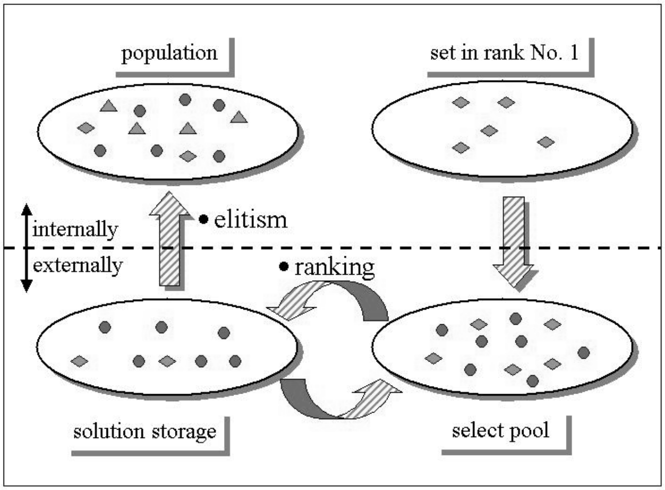


Figure 9.
The process of historical storage

The elitism operator is the policy to always include the best individual in order to prevent losing it due to sampling effects or operator disruption. In other words, this strategy can be extended to copy the best solutions to the next generation. Elitism can improve the performance of an evolutionary algorithm.

So as to grasp the configuration of the whole solution space, the resultant Pareto-optimal solutions are stored outside the loop of the evolutionary operations. This strategy allows the Pareto-optimal solutions created in the past to be kept as solutions and yield a good chance to increase the number of solutions, thus making the solution space easier to see. The storage of the solution independent of the current population also may contribute to the good distribution of the resultant solutions.

3.5 Comparison with other methods

Popular evolutionary algorithms (EAs) originally include, genetic algorithms (GAs) (Holland, 1975), consisting of binary representation of points and proportional/ranking selection and evolution strategies (ESs) (Schwefel, 1981), consisting of continuous representation of points and ranking selection. In the previous reports, those with the binary points and the ranking selection search more robustly than those with continuous points and the proportional selection at the expense of fast convergence (Hoffmeister and Back, 1992; Furukawa and Yagawa, 1995), and *vice versa*. Continuous EAs (CEAs) proposed by the authors (Furukawa and Dissanayake, 1993), incorporating continuous representation of points and proportional selection, therefore demonstrated its convergence approximately 10 times faster than that of GAs and ESs (Furukawa and Yagawa, 1997). MOPCEAs, taking over them from CEAs should also be faster than the multiobjective versions of GAs and EAs in the same manner.

Search strategy and diversity as well as fast convergence are also better than other methods. For effective diversity, niche formation is applied. Currently, most Multiple Objective Evolutionary Algorithm (MOEAs) (Hajela and Lin, 1992; Fonseca and Fleming, 1993; Horn *et al.*, 1994; Srinivas and Deb, 1994) implement niche formation for diversity. Although they always have a genotype (the encoding), they implement phenotypic one in application of sharing since in general phenotypic sharing is superior to genotypic sharing. This depends on the resolution of gene in a genetic, binary space. Therefore, what we perform sharing having a phenotype in phenotypic or continuous space is much useful in maintaining diversity along the phenotypic Pareto optimal front. For search strategy, asymmetric recombination is the better probability than other process, that is, symmetric one or genetic one because the search process has to maintain the balance of robustness and convergence in multiobjective optimization.

4. Test problems

In order to show the working of the proposed approach, a number of test problems have been solved. Although general multiobjective optimization problems are applicable, test problems having two objectives are adopted. This is because we believe that the two-objective optimization brings out the essential features of multiobjective optimization. In all test problems, MOPCEAs described in the previous section are used.

4.1 Test problem 1

We first consider a two-objective problem as follows:

$$\text{Minimize}: \begin{cases} f_1 = x_1^2 + x_2^2 \\ f_2 = (x_1 - 1)^2 + (x_2 - 2)^2 \end{cases} \tag{23}$$

Here, both x_1 and x_2 vary in the interval $[-100,100]$, respectively. We used a population size of 100 and run MOPCEAs until the convergence criterion that the number of population was equal to that of storing set was satisfied. A $\sigma_{\text{share}} = 0.158$ (equation (20) with $n = 2$, $q = 10$) is used. The Pareto-optimal solutions lie in $x_1 \in [0,1], x_2 \in [0,2]$.

Figure 10 shows the Pareto-optimal solution distribution in the $f_1 - f_2$ space. The figure shows that the solution is widely distributed.

Figure 11(a) shows the Pareto-optimal solution distribution solved by MOPCEAs without a niching formation and storing set in the $f_1 - f_2$ space, where marked with “o” is the Pareto-optimal solution and marked with “*” is the dominated solution. Figure 11(b) shows the Pareto-optimal solution distribution solved by GAs without a niching formation and storing set in the $f_1 - f_2$ space. In this case, we used a population size of 100 and both variables

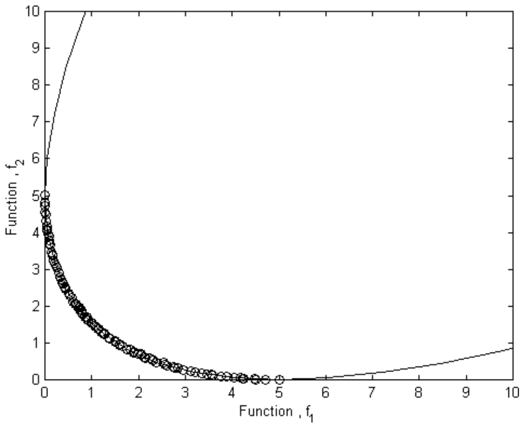
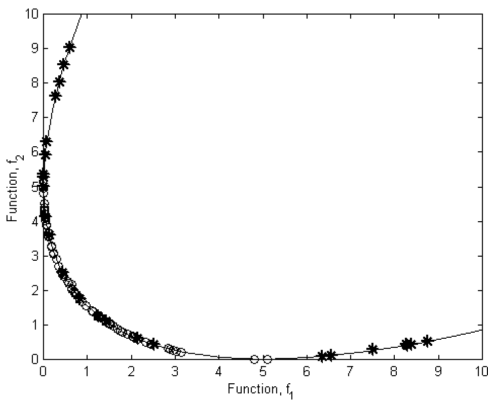
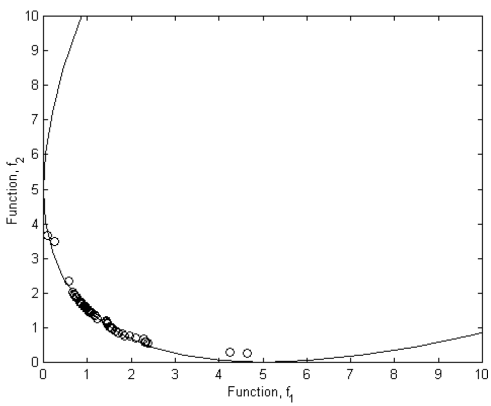


Figure 10.
Pareto-optimal solutions at generation 48 for problem 1



(a) generation 100 in MOPCEAs



(b) generation 50 in GAs

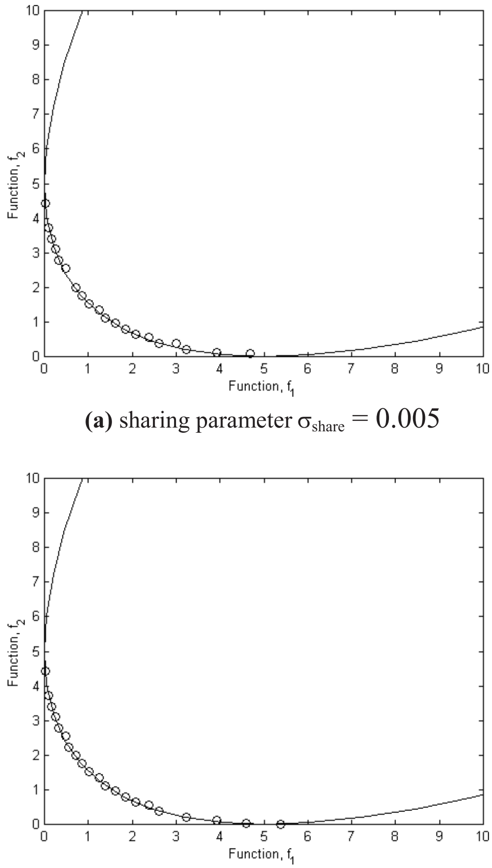
Figure 11.
Pareto-optimal solutions without niching and storing for problem 1

were coded using binary strings of size 10, respectively. Single point crossover with $p_c = 0.8$ and mutation with $p_m = 0.001$ were used. As expected, the result is not widely distributed and the fast and stable convergence is conformed. In case of GAs, the more the iteration goes on, the worse the distribution is.

Figure 12 shows the Pareto-optimal solution distribution solved by GAs with a niching formation and storing set and a different sharing parameter in the $f_1 - f_2$ space. The result is widely distributed but just several Pareto-solutions are found. Additionally, we can see the GAs are independent of the value of the sharing parameter (Srinivas and Deb, 1994).

4.2 Test problem 2

Next, we consider a multiobjective problem having a Pareto-optimal front that is discontinuous and nonconvex.



(a) sharing parameter $\sigma_{share} = 0.005$

(b) sharing parameter $\sigma_{share} = 0.5$

Figure 12.
Pareto-optimal solutions at generation 100 with sharing and storing in GAs for problem 1

$$\text{Minimize}: \begin{cases} f_1 = 10 + x^2 - 10\cos\left(\dfrac{\pi}{2}x\right) \\ f_2 = (x-4)^2 \end{cases} \tag{24}$$

A variable is initialized in the range [− 100,100]. The discontinuity and non-convexity in the Pareto-optimal region comes due to the periodicity in function. This function tests not only the algorithm's ability to find the widely distributed solutions in all discontinuous and non-convex Pareto-optimal regions but also the Pareto-solution not to be found by a classical method. The difference between symmetric recombination and asymmetric one is investigated. We used the same parameters used in problem 1 except a $\sigma_{\text{share}} = 0.05$ (equation (20) with $n = 1$, $q = 10$).

Figure 13 shows the Pareto-optimal solution distribution in the function space and in parameter space by using asymmetric recombination. The figures clearly show that the solution is widely distributed and the fast and stable convergence is conformed. On the contrary, Figure 14 shows the Pareto-optimal solution distribution in the function space by using symmetric recombination. From the above figures, we can see that the search's ability of asymmetric recombination is better than that of symmetric one in all discontinuous and non-convex Pareto-optimal regions.

4.3 Test problem 3

Next, we consider a multimodal multiobjective optimization problem with a simple two-objective having two variables $x_1 (> 0)$ and x_2:

$$\text{Minimize}: \begin{cases} f_1 = x_1 \\ f_2 = \dfrac{g(x_2)}{x_1} \end{cases} \tag{25}$$

where $g(x_2)(> 0)$ is a function of x_2 only as follows.

$$g(x_2) = 2.0 - \exp\left\{-\left(\frac{x_2 - 0.2}{0.004}\right)^2\right\} - 0.8 \times \exp\left\{-\left(\frac{x_2 - 0.6}{0.4}\right)^2\right\} \tag{26}$$

Figure 15 shows the above function for $0 \leq x_2 \leq 1$ with $x_2 \approx 0.2$ as the global minimum and $x_2 \approx 0.6$ as the local minimum solutions. The corresponding values for $g(x_2)$ function values are $g(0.6) = 1.2$ and $g(0.2) = 0.7057$, respectively. Here, both x_1 and x_2 vary in the interval [0.1,1.0] and [0,1.0], respectively. We used a population of size 60 and $\sigma_{\text{share}} = 0.158$ was used.

Figure 16 shows a run of MOPCEAs with and without the elitism process. This means the elitism process does not always work, especially in case of a

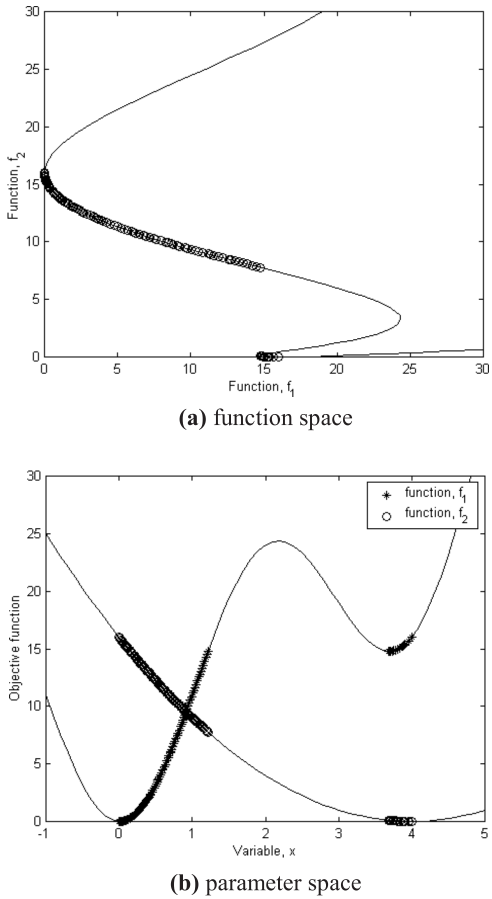


Figure 13.
Pareto-optimal solutions at generation 33 with asymmetric recombination for problem 2

multimodal multiobjective optimization problem. But MOPCEAs found a few global Pareto-optimal solutions in this process, which inform us that this result is not satisfied. Figure 16(b) shows a run of MOPCEAs at generation 441 without the elitism process. Without the elitism process MOPCEAs can find the global Pareto-optimal solution. This result shows MOPCEAs has no difficulty with a multimodal multiobjective optimization problem.

Figure 17 shows the Pareto-optimal solution solved with and without the elitism process by GAs. In this case, we used a population size of 60 and both variables are coded using binary strings of size 10, respectively. Single point crossover with $p_c = 0.8$ was chosen. Mutation with $p_m = 0.001$ was used. The parameters were the same as MOPCEAs. The result of GAs is similar to that of MOPCEAs but the convergence and the search ability is even worse.

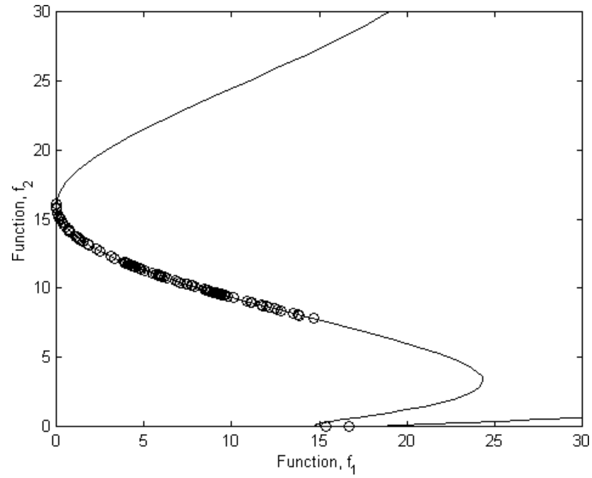


Figure 14.
Pareto-optimal solutions at generation 24 in function space with asymmetric recombination for problem 2

4.4 Test problem 4 (an engineering design)

This problem has been well studied in the context of single objective optimization (Reklaitis *et al.*, 1983) as well as in the study of multiobjective optimization (Deb and Kumar, 1995). In this problem, a beam needs to be welded on another beam and must carry a certain load F (Figure 18).

In single objective optimal design, it is desired to find four design parameters (thickness of the beam b, width of the beam t, length of weld l and weld thickness h) for which the cost of the beam is minimum. The overhang portion of the beam has a length of $L = 14$ inches and a load of $F = 6{,}000$ lb is

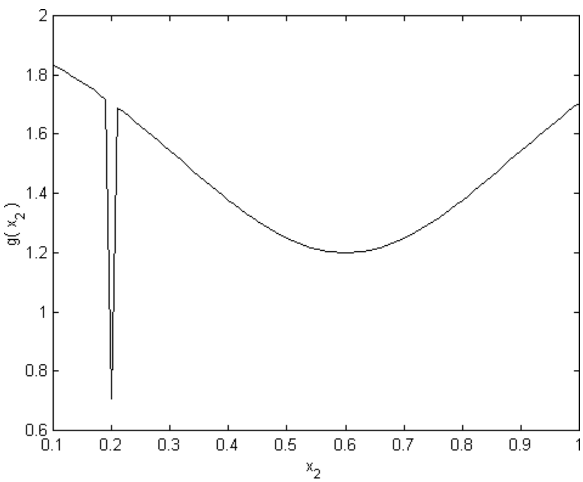
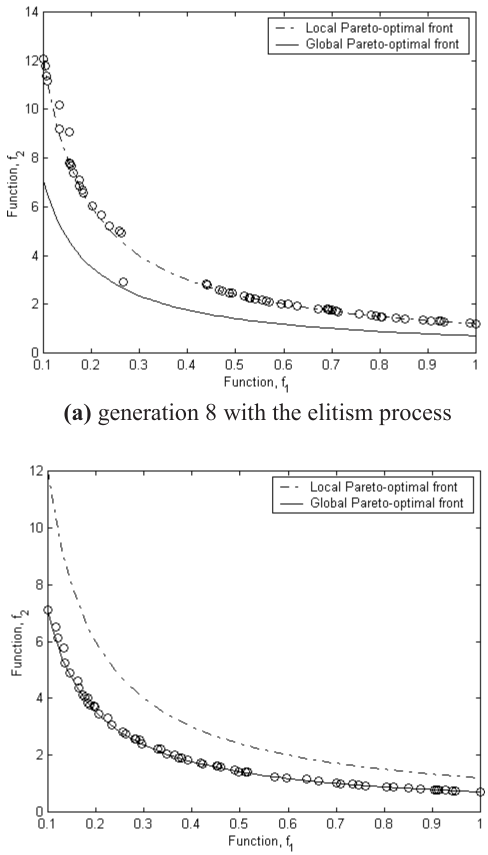


Figure 15.
The function $g(x_2)$ has a global and a local minium solution

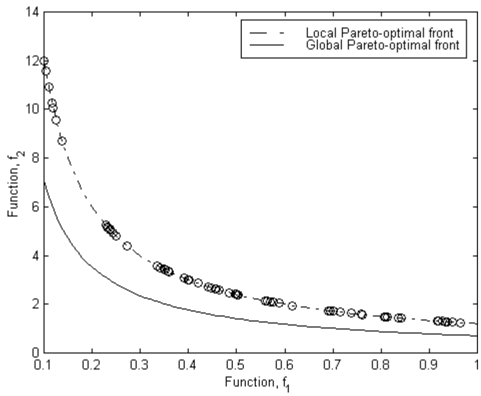


(a) generation 8 with the elitism process

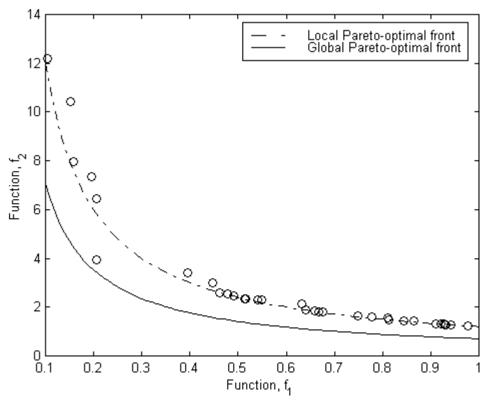
(b) generation 441 without the elitism process

Figure 16.
Pareto-optimal solutions for problem 3

applied at the end of the beam. It is intuitive that an optimal design for cost will make all four design variables to take small values. When the beam dimensions are small, it is likely that the deflection at the end of the beam is going to be large. In the parlance of mechanics of materials, this means that the rigidity of the beam is smaller for smaller dimensions of the beam. In mechanical design activities, optimal design for maximum rigidity is common. Again, a little thought will reveal that a design for maximum rigidity of the above beam will make all four design dimensions to take large dimensions. Thus, the design solutions for minimum cost and maximum rigidity (or minimum end deflection) are conflicting to each other. This kind of conflicting objective functions leads to Pareto-optimal solutions. In the following, we present the mathematical formulation of the two objective optimization problem of minimizing cost and the end deflection (Deb and Kumar, 1995).



(a) generation 24 with the elitism process



(b) generation 500 without the elitism process

Figure 17.
Pareto-optimal solutions by GAs for problem 3

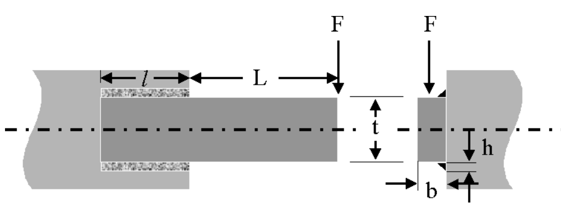


Figure 18.
The welded beam design problem. Minimizations of cost and end deflection are two objectives

$$\text{Minimize}: \begin{cases} f_1 = 1.1047h^2l + 0.04811tb(14.0 + l) \\ f_2 = \delta = \dfrac{2.1952}{t^3 b} \end{cases} \tag{27}$$

$$\text{Subject to}: \begin{cases} g_1 = 13,600 - \tau \geq 0 \\ g_2 = 30,000 - \sigma \geq 0 \\ g_3 = b - h \geq 0 \\ g_4 = P_c - 6,000 \geq 0 \end{cases}$$

There are four constraints. The first constraint makes sure that the shear stress, τ, developed at the support location of the beam is smaller than the allowable shear strength of the material (13,600 psi). Secondly normal stress, σ, developed at the support location of the beam is smaller than the allowable yield strength of the material (30,000 psi). The third constraint makes sure that thickness of the beam is not smaller than the weld thickness from a practical standpoint. As a final constraint the allowable buckling load of the beam is more than the applied load F. A violation of any of the above four constraints will make the design unacceptable. The stress and buckling terms are given as follows (Reklaitis *et al.*, 1983):

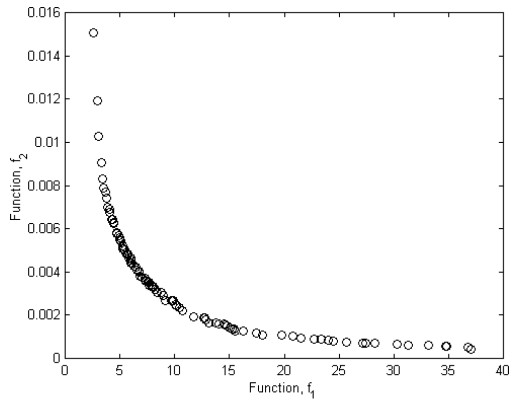
$$\tau = \sqrt{\frac{(\tau')^2 + (\tau'')^2 + (l\tau'\tau'')}{\sqrt{0.25(l^2 + (h+t)^2)}}} \tag{28}$$

$$\tau' = \frac{6,000}{\sqrt{2}hl}, \quad \tau'' = \frac{6,000(14 + 0.5l)\sqrt{0.25(l^2 + (h+t)^2)}}{2\left\{0.707hl\left(\frac{l^2}{12} + 0.25(h+t)^2\right)\right\}}$$

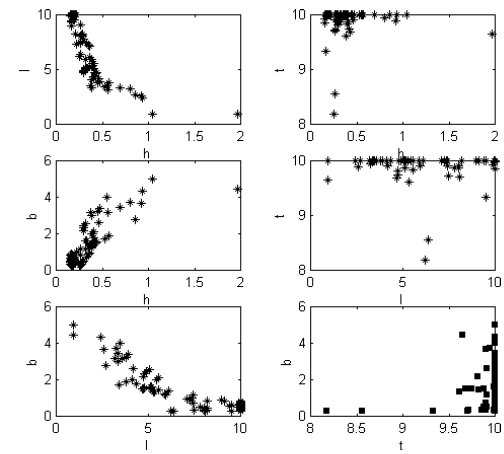
$$\sigma = \frac{504,000}{t^2 b}, \quad P_c = 64,746.022(1 - 0.0282346t)tb^3$$

The variables are initialized in the following range: $0.125 \leq h, b \leq 5.0$ and $0.1 \leq l, t \leq 10.0$. Constraints are handled using the exterior penalty function method. Penalty parameters of 100 and 0.1 are used for the first and second objective functions, respectively. We used a population size of 100 and $\sigma_{\text{share}} = 0.281$ was used.

Figure 19(a) shows that the Pareto-optimal solution after 62 satisfies the terminal criterion and has truly come the Pareto-optimal front. Figure 19(b) shows the Pareto-optimal solution after 62 in parameter space. These figures demonstrate the efficiency of MOPCEAs in converging close to the Pareto-optimal front with a wide variety of solutions. Figure 20 shows the Pareto-optimal front when a population size of 200 is used.



(a) function space

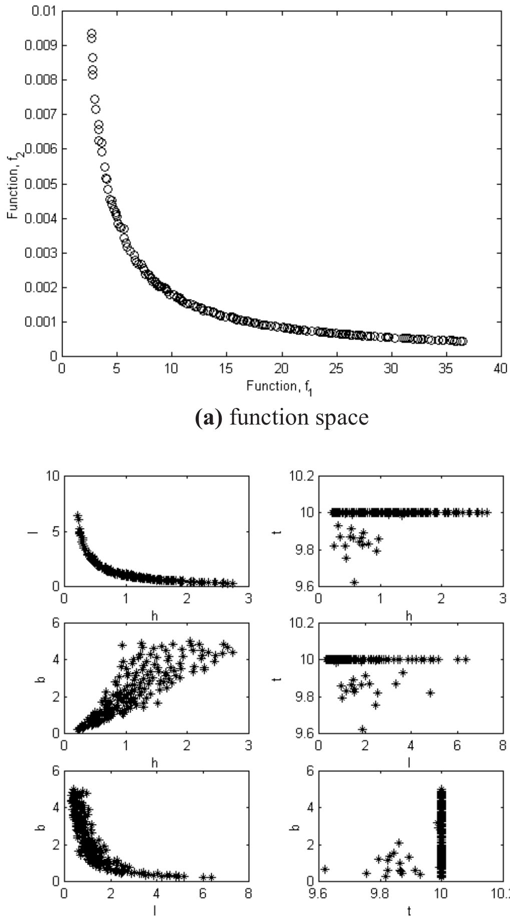


(b) parameter space

Figure 19.
Pareto-optimal solutions at generation 62

In order to investigate its efficiency of the proposed multiobjective formulation with MOPCEAs compared with others, the same problem was solved by GAs. In this case, we used a population size of 100 and both variables are coded using four case, that is, (a) binary strings of size 10, respectively (b) binary strings of size 8, respectively (c) binary strings of size 7, respectively (d) binary strings of size 5, respectively. Single point crossover with $p_c = 0.8$ and mutation with $p_m = 0.001$ were used. The parameters were the same as MOPCEAs.

Figure 21(a) and (b) show that many individuals found the Pareto-optimal solution but the effect of a niche formation did not appear. Figure 21(c) shows that several individuals found the Pareto-optimal solution but the result was not satisfied. Figure 21(d) shows that only several individuals found the wide Pareto-optimal solution and the efficiency of the search was not



(a) function space

(b) parameter space

Figure 20. Pareto-optimal solutions at generation 500

satisfactory. From Figure 21, we can observe that it is very hard to find the wide Pareto-optimal solution by GAs although it has a niche formation and the search of the Pareto-optimal solution is sensitively affected by the length of binary string. Also, we can see that the phenotypic sharing of MOPCEAs that are implemented considering the information of the fitness both in function space and in parameter space is more efficient than the sharing of GAs in a niche formation.

5. Conclusions

In this paper, Pareto-based continuous evolutionary algorithms for multi-objective optimization problem having continuous search space are introduced. The niche formation basically implements phenotypic sharing considering the

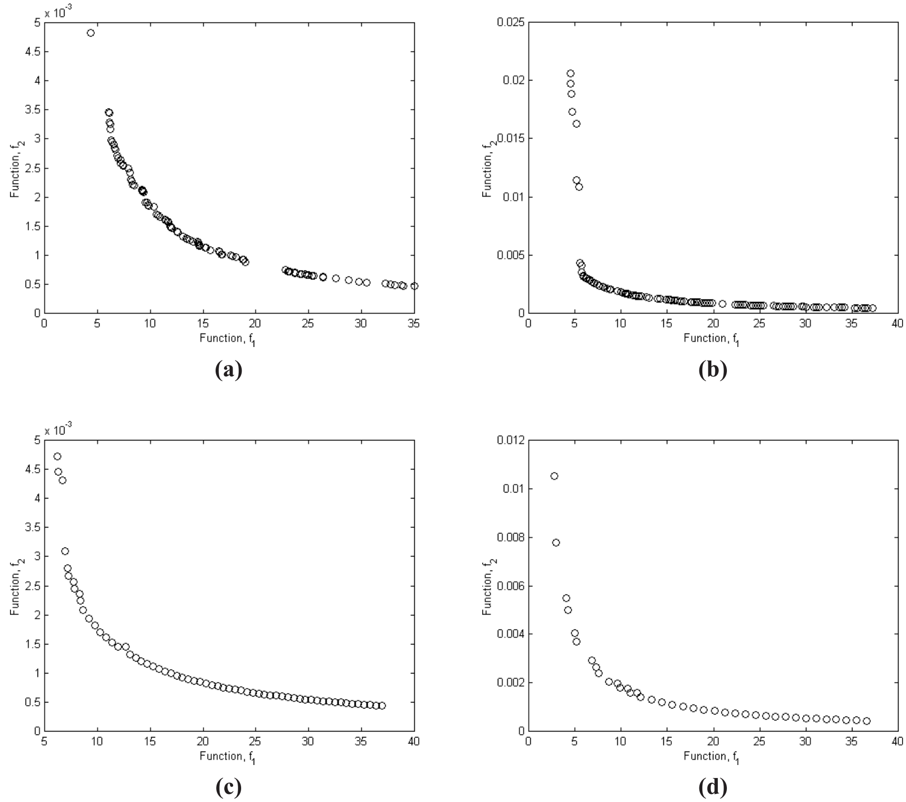


Figure 21.
Pareto-optimal solutions at generation 100

information both in function space and in parameter space and the scaled fitness is obtained dynamically. We applied the niche formation to MOPCEAs based on the continuous search method and a progressive reproduction operator and a storing process for a concrete configuration of solution space and elitism were used and the validity of this method has been tested by numerical examples.

The results of the test problems show that the search ability of MOPCEAs is much better than that of others having discrete search space. Both efficiency and robustness are dramatic and MOPCEAs are effective tools for doing multiobjective optimization in that multiple, wide Pareto-optimal solutions can be found in one single run.

Regarding future perspectives, the issue of higher-level decision-making might be subject to further examinations. In many applications, where the tradeoff surface in containing a huge number of solutions, it is essential that the EA is capable of “selecting” representative solutions. Furthermore, it might be investigated to decrease the number of function evaluation. In real-world application, it is impossible to evaluate thousands of objective function.

Other further study is the application of the technique to actual engineering problems. The technique to the crack identification (Suh *et al.*, 2000) is currently being implemented. Also, The technique to multidisciplinary design optimization of automotive engine mount system is being implemented. In case of automotive engine mount system, the models contain about 26 parameters and its determination is above the human ability. The result of these studies will be reported in further papers.

References

Baek, T. and Schwefel, H.-P. (1993), "An overview of evolutionary algorithms for parameter optimization", *International Journal of Evolutionary Computation*, Vol. 1, pp. 1-23.

Baker, J.E. (1985), "Adaptive selection methods for genetic algorithms", in, Grefenstette, J.J., (Ed.) *Proceedings of the Second International Conference on Genetic Algorithms and Their Applications*, Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 101-11.

Chankong, V. and Haimes, Y.Y. (1983), *Multiobjective Decision Making Theory and Methodology*, North-Holland, New York.

Deb, K. and Goldberg, D.E. (1989), "An investigation of niche and species formation in genetic function optimization", in, Schaffer, J.D., (Ed.) *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, pp. 42-50.

Deb, K. and Kumar, A. (1995), "Real-coded genetic algorithms with simulated binary crossover: studies on multi-modal and multi-objective problems", *Complex Systems*, Vol. 9, pp. 431-54.

Deb, K. (1999), "Multi-objective genetic algorithms: problem difficulties and construction of test problems", *International Journal of Evolutionary Computation*, Vol. 7, pp. 205-30.

Fonseca, C.M. and Fleming, P.J. (1993), "Genetic algorithms for multiobjective optimization: formulation, discussion, and generalization", in, Forrest, S., (Ed.) *Proceedings of the Fifth International Conference on Genetic Algorithms*, Morgan Kaufmann, San Francisco, CA, pp. 416-23.

Fonseca, C.M. and Fleming, P.J. (1995), "An overview of evolutionary algorithms in multi-objective optimization", *International Journal of Evolutionary Computation*, Vol. 3, pp. 1-16.

Furukawa, T. and Dissanayake, G. (1993), "Genetic algorithms using real-valued strings", in, *Proceedings of the 71st JSME Annual Meeting*, Vol. 930-71 pp. 509-10.

Furukawa, T. and Yagawa, G. (1995), "Genetic algorithms using real search space and their use for nonlinear inverse problem", *Transactions of the Japan Society of Mechanical Engineers*, Vol. 61, pp. 1409-15.

Furukawa, T. and Yagawa, G. (1997), "Inelastic constitutive parameter identification using an evolutionary algorithm with constitutive individuals", *International Journal for Numerical Methods in Engineering*, Vol. 40, pp. 1071-90.

Goldberg, D.E. (1989), *Genetic Algorithms for Search, Optimization and Machine Learning*, Addison-Wesley.

Goldberg, D.E. and Richardson, J. (1987), "Genetic algorithms with sharing for multimodal function optimization", in *Proceedings of the First International Conference on Genetic Algorithms and their Applications*, pp. 41-9.

Hajela, P. and Lin, C.-Y. (1992), “Genetic search strategies in multicriterion optimal design”, *Structural Optimization*, Vol. 4 No. 99-107, pp. 416-23.

Hoffmeister, J.H. and Back, T. (1992), “Genetic algorithms and evolution strategies: similarities and differences”. Technical Report, University of Dortmund, Germany, Sys-1/92.

Holland, J.H. (1975), *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Michigan.

Horn, J., Nafploitis, N. and Goldberg, D.E. (1994), “A niched Pareto genetic algorithm for multiobjective optimization”, in Fogel, D., (Ed.) *Proceedings of the First IEEE conference on Evolutionary Computation*, IEEE Press, Piscataway, NJ, pp. 82-7.

Obayashi, S. (1998), “Multidisciplinary design optimization of aircraft wing platform based on evolutionary algorithms”, in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, La Jolla, CA, IEEE.

Qing, A., Lee, C.K. and Jen, L. (1999), “Microwave imaging of parallel perfectly conducting cylinders using real-coded genetic algorithm”, *Journal of Electromagnetic Waves and Application*, Vol. 13, pp. 1121-43.

Reklaitis, G.V., Ravindran, A. and Ragsdell, K.M. (1983), *Engineering Optimization Methods and Applications*, John Wiley and Sons, New York, NY.

Schaffer, J.D. (1985), “Multiple objective optimization with vector evaluated genetic algorithms”, in, *Proceedings of the First International Conference on Genetic Algorithms*, Grefenstette, J.J., (Eds) Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 93-100.

Schwefel, H.-P. (1981), *Numerical Optimization of Computer Models*, John Wiley and Sons, New York, NY.

Srinivas, N. and Deb, K. (1994), “Multiobjective optimization using nondominated sorting in genetic algorithms”, *International Journal of Evolutionary Computation*, Vol. 2, pp. 221-48.

Suh, M.W., Shim, M.B. and Kim, M.Y. (2000), “Crack identification using hybrid neuro-genetic technique”, *Journal of Sound and Vibration*, Vol. 238 No. 4, pp. 617-37.

Valenzuel-Rendon, M. and Uresti-Charre, E. (1997), “A non-generational genetic algorithm for multiobjective optimization”, in *Proceedings of the Seventh International Conference on Genetic Algorithms*, Morgan Kaufmann, San Francisco, CA, pp. 658-65.

Zitzler, E. and Thiele, L. (1998), “Multiobjective optimization using evolutionary algorithms—a comparative case study”, *Parallel Problem Solving from Nature*, Vol. V, pp. 292-301.