Struct Multidisc Optim manuscript No.
(will be inserted by the editor)

An Efficient Multiobjective Differential Evolution Algorithm for Engineering Design

Wenyin Gong · Zhihua Cai · Li Zhu

Received: date / Accepted: date

Abstract Solving engineering design and resources optimization via multiobjective evolutionary algorithms (MOEAs) has attracted much attention in the last few years. In this paper, an efficient multiobjective differential evolution algorithm is presented for engineering design. Our proposed approach adopts the orthogonal design method with quantization technique to generate the initial archive and evolutionary population. An archive (or secondary population) is employed to keep the nondominated solutions found and it is updated by a new relaxed form of Pareto dominance, called Pareto-adaptive ϵ-dominance ($pa\epsilon$-dominance), at each generation. In addition, in order to guarantee to be the best performance produced, we propose a new hybrid selection mechanism to allow the archive solutions to take part in the generating process. To handle the constraints, a new constraint-handling method is employed, which does not need any parameters to be tuned for constraint handling. The proposed approach is tested on seven benchmark constrained problems to illustrate the capabilities of the algorithm in handling mathematically complex problems. Furthermore, four well-studied engineering design optimization problems are solved to illustrate the efficiency and applicability of the algorithm for multiobjective design optimization. Compared with NSGA-II, one of the best MOEAs available at present, the results demonstrate that our approach is found to be statistically competitive. Moreover, the proposed approach is very efficient and is capable of yielding a wide spread of solutions with good coverage and convergence to true Pareto-optimal fronts.

W.Y. Gong · Z.H. Cai · L. Zhu
School of Computer Science, China University of Geosciences, Wuhan 430074, China
Tel.: +86-15926402726
Fax: +86-27-67883716
E-mail: cug11100304@yahoo.com.cn

Keywords Engineering design · multiobjective optimization · differential evolution algorithm · orthogonal design method · Pareto-adaptive ϵ-dominance · constraint-handling method

1 Introduction

Many real-world engineering design or resources optimization involve simultaneous optimization of multiple objectives. Usually, these objectives stay in conflict with each other. As an example, in the design of an automobile an engineer may wish to maximize crash resistance for safety and minimize weight for fuel economy. Instead of finding a single solution, the multiobjective optimization methods try to produce a set of good trade-off solutions called the nondominated solutions or Pareto-optimal solutions from which the decision maker may select one.

To deal with the multiobjective optimization problems (MOPs), many mathematical programming techniques are developed since the 1950s [4]. However, mathematical programming techniques have certain limitations when tackling MOPs, such as (i) most of them can not find multiple solutions in a single run, (ii) many of them are susceptible to the shape of the Pareto front and may not work when the Pareto front is concave or disconnected, and (iii) multiple application of these methods do not guarantee finding widely different Pareto-optimal solutions. On the contrary, evolutionary algorithms (EAs) deal simultaneously with a set of possible solutions (the so-called population) which allows us to find several members of the Pareto-optimal set in a single run of the algorithm. Additionally, EAs are less susceptible to the shape or continuity of the Pareto front. Also, the use of diversity-preserving mechanisms can be added to the EAs to find widely different Pareto-optimal solutions.

Since the 1980s, several multiobjective evolutionary algorithms (MOEAs) have been proposed and applied in MOPs, such as NSGA-II [8], SPEA2 [32], DEMO [23], etc. Recently, many researchers adopted the MOEAs to solve engineering design problems [5], [12], [15], [20], and so on. These algorithms share the same purpose - searching for a *uniformly distributed*, *near-optimal*, and *near-complete* Pareto front for a given MOP. However, this ultimate goal is far from being accomplished by the existing MOEAs described in literature. In one respect, most of the MOPs are very complicated and require the computational resources to be homogenously distributed in a high-dimensional search space. On the other hand, those better-fit individuals generally have strong tendencies to restrict searching efforts within local areas because of the "genetic drift" phenomenon, which results in the loss of diversity due to stochastic sampling.

Differential evolution (DE) [26] algorithm is a novel EA for faster optimization, which mutation operator is based on the distribution of solutions in the population. And DE has won the third place at the first International Contest on Evolutionary Computation on a real-valued function testsuite. Unlike Genetic Algorithm (GA) that uses binary coding to represent problem parameters, DE is a simple yet powerful population based, direct search algorithm with the generation-and-test feature for globally optimizing functions using real valued parameters. Among the DE's advantages are its simple structure, ease of use, speed and robustness. Price & Storn [26] gave the working principle of DE with single scheme. Later on, they suggested ten different schemes of DE [27]. It has been successfully used in solving single-objective optimization problems. Hence, several researchers have tried to extend it to handle MOPs. Such as Pareto DE (PDE) [1], Pareto DE Approach (PDEA) [18], Multiobjective DE (MODE) [28], DE for Multiobjective Optimization (DEMO) [23], GDE3 [14], and ϵ-MyDE [24], which uses the ϵ-dominance [9] to get a good distribution of solutions retained. A detailed survey of multiobjective DE has been published recently [24], where the advantages and disadvantages of the most known existing multiobjective DE methods are discussed. Thus, in the present paper we do not review again this field. However, all of previous approaches generated the initial population randomly and most of them were tested in unconstrained problems. Also little previous work pays attention to solve engineering design problems.

In order to further extend DE to solve multiobjective engineering design problems efficiently, in this paper, we propose a novel multiobjective DE algorithm, called $pa\epsilon$-ODEMO, which integrates established techniques in existing EA's in a single unique algorithm. The new approach uses the orthogonal design method with quantization technique to generate the initial population. Moreover, it adopts an archive to store the nondominated solutions and employs the new Pareto-adaptive ϵ-dominance [13] to update the archive at each generation. In order to handle the constraints, a new constraint-handling method is employed, which does not need any parameters to be tuned for constraint handling. Our proposed approach is validated using seven benchmark constrained MOPs and four engineering optimization problems taken from the specialized literature, and its performance is compared against the Nondominated Sorting Genetic Algorithm II (NSGA-II) [8]. The results indicate that our approach is a viable alternative to efficient multiobjective optimization.

The remainder of this paper is organized as follows. In Section 2, we give the problem formulation of MOPs. The related work is introduced in Section 3. In Section 4, the DE algorithm is briefly described. Section 5 proposes a novel $pa\epsilon$-ODEMO to deal with MOPs and describes its main components in detail. In Section 6, we test our algorithm through a number of benchmark constrained MOPs and engineering design problems. In addition, the experiment results are compared with NSGA-II. The last section, Section 7, is devoted to conclusions.

2 Problem formulation

Without loss the generality, an MOP includes a set of n parameters (decision variables), a set of k objective functions, and a set of m constraints. Objective functions and constraints are functions of the decision variables. The optimization goal is to

$$\begin{aligned} &minimize: \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_k(\boldsymbol{x})) \\ &subject\ to: \boldsymbol{e}(\boldsymbol{x}) = (e_1(\boldsymbol{x}), \cdots, e_m(\boldsymbol{x})) \geq 0 \\ &\quad where: \boldsymbol{x} = (x_1, x_2, \cdots, x_n) \in \boldsymbol{X} \\ &\qquad\quad \boldsymbol{y} = (y_1, y_2, \cdots, y_k) \in \boldsymbol{Y} \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the decision vector, $\boldsymbol{y}$ is the objective vector, $\boldsymbol{X}$ denotes as the decision space, and $\boldsymbol{Y}$ represents the objective space. Generally, for each variable x_i it satisfies a constrained boundary

$$l_i \leq x_i \leq u_i, i = 1, 2, \cdots, n \tag{2}$$

The constraints $\boldsymbol{e}(\boldsymbol{x}) \geq 0$ determine the set of feasible solutions.

Definition 1 (*Pareto Dominance*) A vector $\boldsymbol{x} = (x_1, \cdots, x_k)$ is said to Pareto dominate another vector $\boldsymbol{y} = (y_1, \cdots, y_k)$, denoted as $\boldsymbol{x} \prec \boldsymbol{y}$, if and only if

$$\forall i \in 1, \cdots, k, x_i \leq y_i \quad \text{and} \quad \exists j \in 1, \cdots, k, x_i < y_i$$

Definition 2 (*Pareto Optimality*) A solution $\boldsymbol{x} \in \boldsymbol{X}$ is said to be Pareto optimal in $\boldsymbol{X}$ if and only if $\neg\exists \boldsymbol{y} \in \boldsymbol{X}, \boldsymbol{v} \prec \boldsymbol{u}$, where $\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_k(\boldsymbol{x})), \boldsymbol{v} = \boldsymbol{f}(\boldsymbol{y}) = (f_1(\boldsymbol{y}), \cdots, f_k(\boldsymbol{y}))$.

Definition 3 (ϵ-*dominance*) Let $\boldsymbol{f}, \boldsymbol{g} \in \Re^k$. Then $\boldsymbol{f}$ is said to ϵ-dominate $\boldsymbol{g}$ for some $\epsilon > 0$, denoted as $\boldsymbol{f} \prec_\epsilon \boldsymbol{g}$, if and only if for all $i \in \{1, \cdots, k\}$, $(1-\epsilon)f_i \leq g_i$.

Definition 4 (*Pareto-optimal set*) The Pareto-optimal set *POS* is defined as the set of all Pareto-optimal solutions, i.e., $POS = \{\boldsymbol{x} \in \boldsymbol{X} | \neg\exists \boldsymbol{y} \in \boldsymbol{X}, \boldsymbol{f}(\boldsymbol{y}) \prec \boldsymbol{f}(\boldsymbol{x})\}$.

Definition 5 (*Pareto-optimal front*) The Pareto-optimal front *POF* is defined as the set of all objective functions values corresponding to the solutions in *POS*, i.e., $POF = \{\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_k(\boldsymbol{x})) | \boldsymbol{x} \in POS\}$.

3 Related Work

3.1 ϵ-domination based MOEAs

To achieve a faster convergence in multiobjective optimization, one of the relaxed forms of Pareto dominance, ϵ-dominance proposed by Laumanns *et al.* [16], has become popular in the last few years. The ϵ-dominance acts as an archiving strategy to ensure both properties of convergence towards the Pareto-optimal set and properties of diversity among the solutions found. Deb *et al.* [9], [10] proposed a steady-state MOEA, ϵ-MOEA, based on the ϵ-dominance concept and efficient parent and archive update strategies to MOPs. The simulation results indicated that ϵ-MOEA is a good compromise in terms of convergence near to the Pareto-optimal front, diversity of solutions, and computational time. The authors concluded that the use of ϵ-dominance criterion has been found to have two advantages: (i) it helps in reducing the cardinality of Pareto-optimal region and (ii) it ensures that no two obtained solutions are within an ϵ_i from each other in the i-th objective. Later, Santana-Quintero *et al.* [24] and Cai *et al.* [2] incorporated the ϵ-dominance concept into DE method to solve MOPs. However, the above-mentioned ϵ-domination based MOEAs do not overcome the main limitation of ϵ-dominance: the loss of several nondominated solutions from the hypergrid adopted in the archive because of the way in which solutions are selected within each box [13]. In order to remedy the limitation, Hernández-Díaz *et al.* [13] proposed a new Pareto-adaptive ϵ-dominance method, called $pa\epsilon$-dominance, where different ϵ-dominance regions depending on the geometrical characteristics of the Pareto-optimal front is used. This method remedies some limitations of the original ϵ-dominance and can finds a higher number of efficient points. However, in order to use the $pa\epsilon$-dominance method, an initial Pareto front approximation, denoted by F, must be generated [13]. And the number of efficient points in F can be critical for the final performance. If F is not generated efficiently, the final performance may be very poor.

3.2 Orthogonal design method in EAs

In a discrete single objective optimization problem, when there are N factors (variables) and each factor has Q levels, the search space consists of Q^N combinations of levels. When N and Q are large, it may not be possible to do all Q^N experiments to obtain optimal solutions. Therefore, it is desirable to sample a small, but representative set of combinations for experimentation, and based on the sample, the optimal may be estimated. The orthogonal design was developed for the purpose [11]. The selected combinations are scattered uniformly over the space of all possible combinations Q^N.

Recently, some researchers applied the orthogonal design method incorporated with EAs to solve optimization problems. Leung *et al.* [17] incorporated orthogonal design in GA for numerical optimization problems and found such method was more robust and statistically sound than the classical GAs. OMOEA [29] presented by Zeng *et al.* adopted the orthogonal design method to solve MOPs. In OMOEA, it uses the orthogonal design method to generate a group of sub-niches, every sub-niche evolves at each generation. Because the orthogonal arrays (OAs) must be generated at each generation, OMOEA is very time-consuming. Cai *et al.* [2] proposed a novel multiobjective DE, ϵ-ODEMO, which uses the orthogonal design method to generate the initial population and adopts ϵ-dominance to update the archive. Experimental results indicate ϵ-ODEMO is very efficient in terms of convergence near to the Pareto-optimal front, diversity of solutions, and computational time. However, there are two limitations of ϵ-ODEMO, (i) it may lose some nondominated solutions in the final archive; and (ii) the choice of the ϵ-vector is difficult of this approach.

4 Differential Evolution Algorithm

DE algorithm [26] is a simple evolutionary algorithm that creates new candidate solutions by combining the parent individual and several other individuals of the same population. A candidate replaces the parent only if it has better fitness. This is a rather greedy selection scheme that often outperforms traditional EAs. Unlike GA that uses binary coding to represent problem parameters, DE is a simple yet powerful population based, direct search algorithm with the generation-and-test feature for globally optimizing functions using real valued parameters. It has been successfully used in solving single-objective optimization problems. Among the DE's advantages are its simple structure, ease of use, speed and robustness.

The DE algorithm in pseudo-code is shown in Algorithm 1. Where NP is size of the evolutionary population. CR is the probability of crossover operator. F is the scaling factor. rndint$(1, n)$ is a randomly chosen index $\in 1, 2, \cdots, n$

which ensures that U^i gets at least one parameter from the mutant vector. And $\mathrm{rnd}_j[0,1)$ is the j-th evaluation of a uniform random number generator from $[0,1)$. Many variants of creation of a candidate are possible. We use the DE scheme DE/rand/1/bin (see lines 6 and 13 of Algorithm 1) described in Algorithm 1 (more details on DE/rand/1/bin scheme and other DE schemes can be found in [27]).

Algorithm 1 DE algorithm with DE/rand/1/bin scheme

```
1: Generate the initial population of NP individuals P(0)
2: Evaluate the fitness for each individual in P(0)
3: while The halting criterion is not satisfied do
4:    for i = 1 to NP do
5:       Select uniform randomly r1 ≠ r2 ≠ r3 ≠ i
6:       j_rand = rndint(1, n)
7:       for j = 1 to n do
8:          if rnd_j[0,1) < CR or j = j_rand then
9:             U_j^i = X_j^{r1} + F × (X_j^{r2} − X_j^{r3})
10:         else
11:            U_j^i = X_j^i
12:         end if
13:      end for
14:      Evaluate the offspring U^i
15:      if U^i is better than X^i then
16:         X^i = U^i
17:      end if
18:   end for
19: end while
```

5 Our Proposed Approach: $pa\epsilon$-ODEMO

From the literature review, the main deficiency in the existing MOEAs lies on designing a suitable fitness assignment strategy in order to search for a *uniformly distributed*, *near-complete* and *near-optimal* approximated Pareto front for the given optimization problem. Unfortunately, these objectives are contradictory. Inspired by the ideas from the orthogonal design method successfully used in EAs ([17], [29], and [2]) and $pa\epsilon$-dominance proposed in [13], in this work, we propose an extension of DE algorithm to solve constrained MOPs, which integrates established techniques in existing EA's in a single unique algorithm. Our proposed DE algorithm is named $pa\epsilon$-ODEMO. Six crucial procedures of $pa\epsilon$-ODEMO will be discussed as follows.

5.1 Orthogonal Initial Population

As mentioned above, in order to use the $pa\epsilon$-dominance method, an initial Pareto front approximation F must be generated [13]. And the number of efficient points in F can be critical for the final performance. Obviously, the higher the number of the efficient points in F the better performance of the grid generated. However, before solving an optimization problem, we usually have no information about the location of the global minimum. It is desirable that an algorithm starts to explore those points that are scattered evenly in the feasible solution space. In our presented manner, the algorithm can evenly scan the feasible solution space once to locate good points for further exploration in subsequent iterations. As the algorithm iterates and improves the population of points, some points may move closer to the global minimum. Based on these considerations, in order to obtain an efficient F to generate the the first grid as soon as possible, we apply the quantization technique and the orthogonal design method to generate this initial archive and evolutionary population (EP).

5.1.1 Design of the orthogonal array

To design an orthogonal array (OA), in this research, we use $L_R(Q^C)$ to denote the OA with different level Q, where Q is odd and use $R = Q^J$ to indicate the number of the rows of OA, where J is a positive integer fulfilling

$$C = \frac{Q^J - 1}{Q - 1} \tag{3}$$

C denotes the number of the columns of OA in the above equation. The OA needs to find a proper J and Q to satisfy

$$\begin{aligned} &minimize: R = Q^J \\ &subject\ to: C = \frac{Q^J - 1}{Q - 1} \geq n \\ &\qquad\qquad R \geq NP \end{aligned} \tag{4}$$

where n is the number of the variables, NP is the size of the evolutionary population. In this study, we adopt the algorithm described in [17] to construct an OA. In particular, we use $L(R, C)$ to indicate the OA; and $a_{i,j}$ to denote the level of the jth factor in the ith combination in $L(R, C)$. The algorithm to generate the OA is described in Algorithm 2.

5.1.2 Quantization

For one decision variable X_j with the boundary $[l_j, u_j]$, we quantize the domain into Q levels $\alpha_1^j, \alpha_2^j, \cdots, \alpha_Q^j$, where the design parameter Q is odd and α_i is given by

$$\alpha_i^j = l_j + (i-1)(\frac{u_j - l_j}{Q-1}), 1 \leq i \leq Q \tag{5}$$

In other words, the domain $[l_j, u_j]$ is quantized $Q-1$ fractions, and any two successive levels are same as each other.

5.1.3 Generation of Initial Population

After constructing a proper OA and quantizing the domain of each decision variable, we can generate the orthogonal population (OP) which can scatter uniformly over the decision space. Recall that Algorithm 2 can only construct

Algorithm 2 Construction of Orthogonal Array

/* Construct the basic columns */
for $k = 1$ to J **do**
  $j = \frac{Q^{k-1}-1}{Q-1} + 1$
  for $i = 1$ to R **do**
    $a_{i,j} = \lfloor \frac{i-1}{Q^{J-k}} \rfloor \bmod Q$
  end for
end for
/* Construct the nonbasic columns */
for $k = 2$ to J **do**
  $j = \frac{Q^{k-1}-1}{Q-1} + 1$
  for $s = 1$ to $j-1$ **do**
    for $t = 1$ to $Q-1$ **do**
      for $i = 1$ to R **do**
        $a_{i,(j+(s-1)(Q-1)+t)} = (a_{i,s} \times t + a_{i,j}) \bmod Q$
      end for
    end for
  end for
end for
Increment $a_{i,j}$ by one for all $i \in [1, R]$ and $j \in [1, C]$

$L_R(Q^C)$, where $R = Q^J$ and J is a *smallest* positive integer fulfilling $C = \frac{Q^J-1}{Q-1} \geq n$ [see (4)]. Since the problem dimension n is given, when we execute Algorithm 2 to construct an OA with C factors, if $C > n$, then delete the last $C - n$ columns to get an array with n factors. And the remainder of the array is still an OA [17]. The algorithm for generating the OP is described in Algorithm 3, where $OP_{i,j}$ is the j-th variable of the i-th individual of OP, R is the number of rows in OA, n is the number of decision variables, and $eval$ is the current number of fitness function evaluations (NFFEs). Regularly, the number of the rows R of the OA is larger than the population size NP, so we create the initial archive with the nondominated solutions from OP first. And then we generate the initial EP from the initial archive and OP with the following procedure (more details are described in Algorithm 3). If $ar\_size \geq NP$, we select NP solutions from the initial archive randomly; or all of the ar_size solutions in the initial archive are inserted into EP, and the remainder $NP - ar\_size$ solutions are selected from OP randomly. Compared with OGA/Q [17], our approach and OGA/Q differ in two aspects with respect to the generation of initial population: (i) our approach integrates the orthogonal design method within MOEAs, and (ii) in our approach there are two population, archive and EP, we first generate the initial archive with the nondominated solutions from OP, and then the initial EP is generated from the initial archive and OP. It is worth to point out that for a given MOP and the given Q and J, the initial archive generated by Algorithm 3 is the same for all runs, so we can generate the initial archive offline to avoid the time complexity of the Algorithm 2 and Algorithm 3. Thus we do not need to generate the initial archive each run, which makes the new algorithm more efficient. Furthermore, for a given MOP, we only need to generate a minimal OA which satisfies (4), so that the parameters Q and J are very easy to be selected for different problems.

Algorithm 3 Construction of Initial Archive and Evolutionary Population

/* Construction of orthogonal population (OP) */
$eval = 0, ar\_size = 0$
for $i = 1$ to R **do**
  for $j = 1$ to n **do**
    $k = a_{i,j}$
    $OP_{i,j} = \alpha_k^j$
  end for
  Evaluate OP_i and $eval$++
end for
/* Construction of initial archive (AR) */
Find all of the nondominated solutions of OP
Insert them into the initial archive, now $ar\_size > 0$
/* Construction of initial evolutionary population (EP) */
if $ar\_size \geq NP$ **then**
  Randomly select NP solutions from AR to generate initial EP
else
  Insert all of the ar_size solutions of AR into EP
  Select the remainder $NP - ar\_size$ solutions from OP randomly
  Inert them into EP
end if

5.2 Archiving the Candidate Solutions

In the study of Zitzler *et al.* [31], it was clearly shown that elitist helps in achieving better convergence in MOEAs. In $pa\epsilon$-ODEMO, the elitism scheme is also adopted through maintaining an external archive of nondominated solutions found in evolutionary process. In order to achieve a faster convergence, in this work, the Pareto-adaptive ϵ-dominance, the so-called $pa\epsilon$-dominance proposed by Hernández-Díaz *et al.* [13], is used to update the archive. At each generation, in order to include a solution into this archive, it is compared with respect to each member already contained in the archive using $pa\epsilon$-dominance after the grid is generated. Note, however, that although ϵ can often have little meaning for test function analysis, the ϵ-dominance method is very useful: i) it allows for the desired convergence properties, and ii) it guarantees an optimal distribution of solutions [16]. Moreover, Laumanns *et al.* [16] pointed out that none of the MOEAs has a proof of convergence to the true Pareto-optimal solutions with a wide diversity among the solutions, so the ϵ-dominance method used to solve engineering problems is meaningful and reasonable compared with other MOEAs. The procedure is described as follows.

Every solution in the archive is assigned an identification array ($\boldsymbol{B} = (B_1, B_2, \cdots, B_k)^T$, where k is the total number

of objectives) as follows:

$$B_i(\boldsymbol{f}) = \left\lfloor \frac{\log(\frac{\epsilon_1^i p^{v_i} - (p^{v_i}-1) f_i}{\epsilon_1^i})}{\log(\frac{1}{p^{v_i}})} + 1 \right\rfloor \quad (6)$$

where, p controls the shape of the curve (or surface), T is the number of points desired by the decision maker, ϵ_1^i is the size of the first box for each dimension, and v_i controls the speed of variation. These parameters satisfy the following equations:

$$\begin{cases} \epsilon_1^i = \frac{(p^{v_i}-1)p^{(T-1)v_i}}{p^{Tv_i}-1} \\ (1-2^{1/p})p^{Tv_i} + 2^{1/p}p^{Tv_i/2} - 1 = 0 \end{cases} \quad (7)$$

The identification array divides the whole objective space into hyper-boxes. With the identification arrays calculated for the offspring c and each archive member a, the offspring c updates the archive as Algorithm 4 described (B_a indicates the identification arrays of solution a).

Algorithm 4 Updating the Archive

```
if B_c of the offspring ε-dominates B_a of any archive member a then
    Delete all of the dominated archive members
    Accept the offspring c
else if B_c is ε-dominated by B_a of any archive member a then
    Reject c
else
    if c shares the same grid with an archive member a then
        if c dominates a or c is closer to the grid than a then
            Delete a from the archive and accept c
        else
            Reject c
        end if
    else
        Insert c into the archive
    end if
end if
```

Using the $pa\epsilon$-dominance method, we can maintain the good properties of the original ϵ-dominance, such as ensuring both properties of convergence towards the Pareto-optimal set and properties of diversity among the solutions found in a small computation time, while overcoming the main limitations of ϵ-dominance: the loss of several non-dominated solutions from the hypergrid adopted in the archive.

5.3 Hybrid Selection Mechanism

In [16], Laumanns *et al.* concluded that the archived members are really guaranteed to be the best solutions found. As the archive is used to maintain the nondominated solutions in $pa\epsilon$-ODEMO, a key issue must be addressed is that how to allow the archive members to take part in the generating process? In ϵ-MOEA [9], [10], Deb *et al.* randomly picked a solution from the archive for mating starting from the evolutionary process. In ϵ-MyDE [24], Santana-Quintero *et al.* proposed two selection mechanisms, where the random selection and elitist selection are interleaved. At the beginning, all of the parents for mating are randomly selected from the EP to generate the offspring. When the current generations is larger than a pre-defined number, the elitist selection is used and all of the parents are randomly selected from the archive to generate the offspring. However, these two techniques have a limitation: the selection pressure is very high. Especially, in ϵ-MyDE, the main evolutionary population (EP) is not used at all when the elitist selection is adopted.

In our proposed approach, in order to regulate the selection pressure we propose a hybrid selection mechanism, which is similar to the method used in ϵ-MyDE, in which a random selection and an elitist selection are interleaved. The difference between ϵ-MyDE and our approach is that in elitist selection we only randomly choose one solution from the archive as the base parent, X^{r_1}, in DE/rand/1/bin scheme described in Algorithm 1. And the other two parents, X^{r_2} and X^{r_3}, are selected from EP randomly. We use a *selection parameter* $\lambda \in [0.1, 1.0]$ to regulate the selection pressure.

$$\text{selection} = \begin{cases} \text{random selection}, & eval < (\lambda \times Max\_eval) \\ \text{elitist selection}, & \text{otherwise} \end{cases} \quad (8)$$

where, $eval$ is the current NFFEs, and Max_eval is the maximal NFFEs pre-defined by the user. In our proposed hybrid selection mechanism, one archive solution, which is disturbed by the two randomly selected solutions from EP, is selected to take part in the generating process when the elitist selection is used. In this manner, our approach has three advantages: (i) it can guarantee to be the best solutions found; (ii) because we do not use the archive solution to guide the search at the beginning of evolutionary process, it can avoid to be misled by the inefficient archive solutions; and (iii) the solutions in EP can also guide the search in the elitist scheme.

5.4 Constraint-handling Method

Usually, most of engineering design problems have multiple constraints. So it is essential to handle the constraints to solve the engineering design problems. A considerable amount of researches on constraint handing techniques that incorporate objective(s) and constraint(s) into the fitness function of design candidates has been carried out (a good summary is given in [3]). However, many previous constraint-handling methods need to tune some parameters to balance between the objective(s) and constraint(s). In this research, we employ a new constraint-handling method proposed by

Oyama [19], which does not need any parameters to be tuned for constraint handling. This method is based on the "Constraint-First-Objective-Next" model [15], where the constraints precede the objectives because the feasibility of $\boldsymbol{x}$ is more important than minimization of $\boldsymbol{f}(\boldsymbol{x})$. The method is described as follows.

Definition 6 (*Constrained Pareto dominance*) A solution $\boldsymbol{x}_i$ is said to constrained-dominate a solution $\boldsymbol{x}_j$, if any of the following conditions is true:

1. Solutions $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are both feasible and solution $\boldsymbol{x}_i$ dominates solution $\boldsymbol{x}_j$ in objective function space.
2. Solution $\boldsymbol{x}_i$ is feasible and solution $\boldsymbol{x}_j$ is not.
3. Solutions $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are both infeasible, but solution $\boldsymbol{x}_i$ dominates solution $\boldsymbol{x}_j$ in constraint space.

where dominance in objective space is defined as Definition 1 while dominance in constraint space is defined as:

Definition 7 (*Constraint space dominance*) A solution $\boldsymbol{x}_i$ is said to dominate a solution $\boldsymbol{x}_j$ in constraint space, if both of the following conditions are true:

1. Solutions $\boldsymbol{x}_i$ is no worse than solution $\boldsymbol{x}_j$ in all constraints, *i.e.*,

$$\forall E_k(\boldsymbol{x}_i) \geq E_k(\boldsymbol{x}_j) \tag{9}$$

2. Solution $\boldsymbol{x}_i$ is strictly better than solution $\boldsymbol{x}_j$ in at least one constraint, *i.e.*,

$$\exists E_k(\boldsymbol{x}_i) > E_k(\boldsymbol{x}_j) \tag{10}$$

where $E_k(\boldsymbol{x}) = \min(0, e_k(\boldsymbol{x}))$, and $k = 1, \cdots, m$.

5.5 Handling the Constraint of the Variables

After using the DE/rand/1/bin scheme to generate a new solution, if one or more of the variables in the new solution are outside their boundaries, i.e. $x_i \notin [l_i, u_i]$, the following repair rule is applied:

$$x_i = \begin{cases} l_i + \mathrm{rnd}_i[0,1] \times (u_i - l_i) & \text{if } x_i < l_i \\ u_i - \mathrm{rnd}_i[0,1] \times (u_i - l_i) & \text{if } x_i > u_i \end{cases} \tag{11}$$

where $\mathrm{rnd}_i[0,1]$ is the uniform random variable from [0,1] in each dimension i.

5.6 Main Procedure of $pa\epsilon$-ODEMO

For an MOP, the proposed $pa\epsilon$-ODEMO algorithm works as Algorithm 5. First, the temporary orthogonal population (OP) is created using Algorithm 3, all of the nondominated solutions of OP are inserted into the initial archive (AR). And then, the initial evolutionary population (EP) is created from AR and OP. At each generation, an offspring is generated using DE/rand/1/bin scheme. The offspring replaces the parent immediately if the parent is dominated by the offspring. If the parent dominates the offspring, the offspring is discarded. Otherwise (when the offspring and parent are nondominated with regard to each other), the offspring is added to a temporary child population (CP). If the first grid is not generated, insert the offspring into the archive with usual dominance concept. Otherwise, if the first grid has been generated, insert the offspring into the archive with $pa\epsilon$-dominance concept. This step is repeated until NP number of offspring are created. After that, combine the temporary child population CP with EP, and we get a population of the size between NP and $2\times NP$. If the population has enlarged, we have to truncate it to prepare it for the next step of the algorithm.

Algorithm 5 Main procedure of the proposed $pa\epsilon$-ODEMO

```
Generate a proper OA and generate the orthogonal population OP
Create the initial archive AR_1 with the nondominated solutions of OP
Create the initial evolutionary population EP_1 from AR_1 and OP
t = 1
flag = 0
while eval < Max_eval do
    child_size = 0
    for i = 1 to NP do
        if eval < λ × Max_eval then
            Random selection
            Produce the offspring c with DE/rand/1/bin scheme
        else
            Elitist selection
            Produce the offspring c with DE/rand/1/bin scheme
        end if
        Evaluate the offspring c and eval++
        if the offspring c dominates the target parent EP_t^i then
            EP_t^i = c
        else if c is nondominated by EP_t^i then
            Add c to the child population CP
            child_size++
        else
            Discard c
        end if
        if flag == 0 then
            Update the archive with the usual dominance
        else
            Update the archive with paε-dominance
        end if
    end for
    if child_size ≠ 0 then
        Combine CP and EP_t
        Prune the mixed population using nondominated ranking method only
        Get the next evolutionary population EP_{t+1}
    end if
    if ar_size ≥ N_F and flag == 0 then
        Generate the paε-dominance grid
        flag = 1
    end if
    t++
end while
```

The truncation sorts the individuals with nondominated sorting and if the individuals in the same front we only select them randomly. This is different from NSGA-II [8] and DEMO [23] to evaluate the individuals of the same front with the crowding distance metric. Because in our approach the diversity is maintained in the archive with $pa\epsilon$-dominance. The truncation procedure keeps in EP only the best NP individuals (with regard to the nondominated sorting metric).

If the initial Pareto front approximation F is created, i.e. ar_size is larger than N_F that is the size of F, and the first grid is not generated, then we use the $pa\epsilon$-dominance concept to generate the grid (the algorithm is omitted here, interest readers can refer [13] for more details).

6 Experiments and Results

In order to validate the performance of our proposed approach, $pa\epsilon$-ODEMO, it is tested on seven constrained benchmark problems to illustrate the capabilities of the algorithm in handling mathematically complex problems. Also, four well-studied engineering design optimization problems are solved to illustrate the efficiency and applicability of the algorithm for multiobjective design optimization. Moreover, our approach is compared with NSGA-II [8], which is one of the best MOEAs available at present. For $pa\epsilon$-ODEMO, we have chosen a reasonable set of value and have not made any effort in finding the best parameter settings. We leave this task for a future study.

6.1 Parameter Settings

All approaches are run 50 independent runs with different random seeds on all test problems. The maximal NFFEs of different problems (described in the following sections) are different due to the complexity of the search space. For different approaches, the parameter settings are as follows:

- For NSGA-II [8], the simulated binary crossover (SBX) and polynomial mutation are used. The crossover probability of $p_c = 0.9$ and a mutation probability of $p_m = 1/n$ (where n is the number of decision variables). The distribution indices for crossover and mutation operators set as $\eta_c = 20$ and $\eta_m = 20$, respectively. The population size of $NP = 100$.
- For $pa\epsilon$-ODEMO, the crossover probability of $p_c = 0.9$ and the scaling factor of $F = 0.5$. The evolutionary population size NP is set to 100. The size of initial Pareto front approximation F is $N\_F = 100$[1]. The number of points desired by the decision maker of $T = 100$. The selection parameter of $\lambda = 0.9$. As suggested in [17], to generate an OA, we use $J = 2$, and $Q = 11$ if $n < 10$; otherwise, $Q = n - 1$.

[1] For a grid with a maximum capacity of 100 vectors (i.e. $T = 100$), Hernández-Díaz *et al* indicated that the best results are obtained when $75 \leq N\_F \leq 125$ [13].

6.2 Performance Metrics

In order to make a fair comparison with other MOEAs, we use two performance metrics proposed by Deb *et al.* [8] to assess the performance. The two metrics are derived from the final generation of 50 independent runs with different random seeds.

The first metric is the *Convergence metric* γ. It measures the distance between the obtained nondominated front Q and the set POF of Pareto-optimal front:

$$\gamma = \frac{\sum_{i=1}^{|Q|} d_i}{|Q|} \tag{12}$$

where d_i is the Euclidean distance (in the objective space, hereinafter) between the solution $i \in Q$ and the nearest member of POF. The lower the γ value, the better the convergence of solutions. A result of $\gamma = 0$ indicates the convergence $Q = POF$; any other value indicates Q deviates from POF.

The second metric is *Diversity metric* Δ. This metric measures the extent of spread achieved among the obtained nondominated front Q. It is desirable to get a set of solutions that spans the entire Pareto-optimal region. Δ is defined as follows:

$$\Delta = \frac{\sum_{i=1}^{k} d_i^e + \sum_{i=1}^{|Q|} |d_i - \overline{d}|}{\sum_{i=1}^{k} d_i^e + |Q|\overline{d}} \tag{13}$$

where d_i^e denotes the Euclidean distance between the i-th coordinate for both extreme points in Q and POF, and d_i measures the Euclidean distance of each point in Q to its closer point in Q. From the above definition, it is easy to conclude that $\Delta \geq 0$ and the lower the Δ value, the better the distribution of solutions. A perfect distribution, that is $\Delta = 0$, means that the extreme points of POF have been found and d_i is constant for all i.

Moreover, in order to compare the dominance relationship between two populations resulting from two different MOEAs, the coverage of two sets (C value) [30], which is a binary metric [33], is measured to show how the final population of one algorithm dominates the final population of another algorithm. The C value can be calculated as follows:

$$C(X', X'') = \frac{|a'' \in X''; \exists a' \in X' : a' \preceq a''|}{|X''|} \tag{14}$$

where $X', X'' \in X$ are two sets of objective vectors, and $a' \preceq a''$ means that a' *covers* a'' if and only if $a' \prec a''$ or $a' = a''$. Function C maps the ordered pair (X_i, X_j) to the interval $[0, 1]$, where X_i and X_j denote the final populations

resulting from algorithm i and j, respectively. The value $C(X_i, X_j) = 1$ implies that all points in X_j are dominated by or equal to points in X_i. The opposite, $C(X_i, X_j) = 0$, represents the situation when none of the points in X_j are covered by the set X_i. Note that both $C(X_i, X_j)$ and $C(X_j, X_i)$ need to be considered independently since they have the distinct meanings.

6.3 Benchmark Constrained MOPs

First, we select seven benchmark constrained MOPs (OSY, SRN, TNK, Kita, Tamaki, DTLZ8, and DTLZ9) to test the capabilities of the algorithm in handling mathematically complex problems. The details of these problems are described in Table 1, where OSY, SRN, and TNK are chosen from [6], Kita and Tamaki are chosen from [24], DTLZ8 and DTLZ9 are selected from [7]. For OSY, SRN, TNK, and Kita, they are two-objective MOPs. For Tamaki, DTLZ8, and DTLZ9, they are three-objective MOPs. The characters of these problems are briefly described as follows.

- For OSY, there are six constraints. The Pareto-optimal region is a concatenation of five regions. Every region lies on the intersection of certain constrains. Since the entire Pareto-optimal region demands an MOEA population to maintain sub-populations at different intersections of constraint boundaries, it is difficult to solve for MOEAs.
- For SRN, it is relatively easy to solve for MOEAs. The only difficulty the constraints introduce in this problem is that they eliminate some part of the unconstrained Pareto-optimal set.
- For TNK, this is a disconnected MOP, the Pareto-optimal sets are disconnected because of the constraints. Since the Pareto-optimal solutions lie on a non-linear constraint space, an optimization algorithm may find difficulty in finding a spread of solutions across the entire Pareto-optimal region.
- For Kita, this problem has disconnected, concave Pareto-optimal front, and the Pareto-optimal set is also disconnected. Note that the Pareto-optimal set is in the boundary between the feasible and the infeasible region.
- For Tamaki, this a three-objective MOP. The Pareto-optimal front is connected and forms a curved surface.
- For DTLZ8, it is a high-dimensional and three-objective problem. The Pareto-optimal front is a combination of a straight line and a hyper-plane. An optimization algorithm may find difficulty in finding solutions in both the straight line and the hyper-plane in this problem and also in maintaining a good distribution of solutions on the hyper-plane.
- For DTLZ9, it is also a high-dimensional and three-objective problem. The Pareto-optimal front is a curve with $f_1 = f_2$. The density of solutions gets thinner towards the Pareto-optimal region. The Pareto-optimal curve lies on the intersection of all 2 constraints. This feature of this problem may cause MOEAs difficulty in solving this problem.

Table 2 compares the performances of $pa\epsilon$-ODEMO and NSGA-II. The statistics shown are based on 50 independent runs with different random seeds. All the algorithms are implemented in C++ and the experiments are done on a P-IV 2.8 GHz machine with 512 MB RAM under WIN-XP platform. For the convergence metric and diversity metric, we need to know the true Pareto-optimal front for a problem. Since the benchmark constrained MOPs in Table 1 are artificial test problems, the true Pareto-optimal front is not difficult to be obtained. In our experiments we use uniformly spaced Pareto-optimal solutions as the approximation of the true Pareto-optimal front.

From Table 2, we can see that our approach is capable of converging closely to the true Pareto-optimal front, with good distribution of non-dominated solutions, for all of the benchmark constrained MOPs. With respect to the set coverage metric, $pa\epsilon$-ODEMO produces better mean values than NSGA-II for all of the seven MOPs. Also compared with the convergence metric, $pa\epsilon$-ODEMO performs better than NSGA-II for these test problems. For the diversity metric, $pa\epsilon$-ODEMO provides better mean values than NSGA-II for six out of seven test problems, NSGA-II is better than $pa\epsilon$-ODEMO on OSY. However, from Fig. 1, it can be seen that NSGA-II can not coverage the entire true Pareto front sometimes. Moreover, $pa\epsilon$-ODEMO needs less CPU time than NSGA-II. The reason are two aspects. On the one hand, for a give MOP and the given Q and J we only need to generate the initial archive one run out of 50 runs. On the other hand, as described in Algorithm 5, the crowding distance metric is not used in our approach, and hence it can eliminate the complexity of $O(k(2NP)log(2NP))$ of the crowding distance assignment and the complexity of $O(2NPlog(2NP))$ of the crowding distance sorting compared with NSGA-II [8].

In order to demonstrate the working of the algorithm, a *typical* simulation run using $pa\epsilon$-ODEMO and NSGA-II is shown in Fig. 1 and Fig. 2 for all benchmark constrained problems. The graphical illustration also clearly demonstrates that for the two-objective MOPs $pa\epsilon$-ODEMO is able to search for a *uniformly distributed*, *near-complete* and *near-optimal* approximated Pareto front. Furthermore, $pa\epsilon$-ODEMO can obtain better nondominated solutions than NSGA-II for all three-objective MOPs.

From the comparison and analysis above, we can conclude that $pa\epsilon$-ODEMO can produce competitive results compared with NSGA-II. It is able to search for a *uniformly distributed*, *near-complete* and *near-optimal* approximated Pareto front. Moreover, $pa\epsilon$-ODEMO can deal with two-

Table 1 Benchmark constrained problems used in this study, n is the number of decision variables

Problem	Max_eval	n	Domain	Objective functions	Constraints
OSY	25,000	6	$0 \leq x_1, x_2, x_6 \leq 10$	$\min f_1(\boldsymbol{x}) = -25[\sum_{i=1}^{2}(x_i-2)^2 + (x_3-1)^2 + (x_4-4)^2 + (x_5-1)^2]$	$e_1(\boldsymbol{x}) = x_1 + x_2 - 2 \geq 0$
			$1 \leq x_3, x_5 \leq 5$	$\min f_2(\boldsymbol{x}) = \sum_{i=1}^{6} x_i^2$	$e_2(\boldsymbol{x}) = 6 - x_1 - x_2 \geq 0$
			$0 \leq x_4 \leq 6$		$e_3(\boldsymbol{x}) = 2 + x_1 - x_2 \geq 0$
					$e_4(\boldsymbol{x}) = 2 - x_1 + 3x_2 \geq 0$
					$e_5(\boldsymbol{x}) = 4 - (x_3-3)^2 - x_4 \geq 0$
					$e_6(\boldsymbol{x}) = (x_5-3)^2 + x_6 - 4 \geq 0$
SRN	5,000	2	$-20 \leq x_1 \leq 20$	$\min f_1(\boldsymbol{x}) = 2 + (x_1-2)^2 + (x_2-1)^2$	$e_1(\boldsymbol{x}) = 225 - x_1^2 - x_2^2 \geq 0$
			$-20 \leq x_2 \leq 20$	$\min f_2(\boldsymbol{x}) = 9x_1 - (x_2-1)^2$	$e_2(\boldsymbol{x}) = 10 - x_1 + 3x_2 \geq 0$
TNK	20,000	2	$0 \leq x_1 \leq \pi$	$\min f_1(\boldsymbol{x}) = x_1$	$e_1(\boldsymbol{x}) = x_1^2 + x_2^2 - 1 - 0.1\cos(16\arctan(x_1/x_2)) \geq 0$
			$0 \leq x_2 \leq \pi$	$\min f_2(\boldsymbol{x}) = x_2$	$e_2(\boldsymbol{x}) = 0.5 - \sum_{i=1}^{2}(x_i-0.5)^2 \geq 0$
Kita	5,000	2	$x_1 \geq 0$	$\max f_1(\boldsymbol{x}) = x_2 - x_1^2$	$e_1(\boldsymbol{x}) = 6.5 - x_1/6 - x_2 \geq 0$
			$x_2 \geq 0$	$\max f_2(\boldsymbol{x}) = 0.5x_1 + x_2 + 1$	$e_2(\boldsymbol{x}) = 7.5 - x_1/2 - x_2 \geq 0$
					$e_3(\boldsymbol{x}) = 30 - 5x_1 - x_2 \geq 0$
Tamaki	5,000	3	$0 \leq x_i \leq 1$	$\max f_1(\boldsymbol{x}) = x_1$	$e_1(\boldsymbol{x}) = x_1^2 + x_2^2 + x_3^2 > 0$
				$\max f_2(\boldsymbol{x}) = x_2$	
				$\max f_2(\boldsymbol{x}) = x_3$	
DTLZ8	50,000	30	$0 \leq x_i \leq 1$	$\min f_1(\boldsymbol{x}) = \frac{1}{10}\sum_{i=1}^{10} x_i$	$e_1(\boldsymbol{x}) = f_3(\boldsymbol{x}) + 4f_1(\boldsymbol{x}) - 1 \geq 0$
				$\min f_2(\boldsymbol{x}) = \frac{1}{10}\sum_{i=11}^{20} x_i$	$e_2(\boldsymbol{x}) = f_3(\boldsymbol{x}) + 4f_2(\boldsymbol{x}) - 1 \geq 0$
				$\min f_3(\boldsymbol{x}) = \frac{1}{10}\sum_{i=21}^{30} x_i$	$e_3(\boldsymbol{x}) = 2f_3(\boldsymbol{x}) + f_1(\boldsymbol{x}) + f_2(\boldsymbol{x}) - 1 \geq 0$
DTLZ9	50,000	30	$0 \leq x_i \leq 1$	$\min f_1(\boldsymbol{x}) = \frac{1}{10}\sum_{i=1}^{10} x_i^{0.1}$	$e_1(\boldsymbol{x}) = f_3^2(\boldsymbol{x}) + f_1^2(\boldsymbol{x}) - 1 \geq 0$
				$\min f_2(\boldsymbol{x}) = \frac{1}{10}\sum_{i=11}^{20} x_i^{0.1}$	$e_2(\boldsymbol{x}) = f_3^2(\boldsymbol{x}) + f_2^2(\boldsymbol{x}) - 1 \geq 0$
				$\min f_3(\boldsymbol{x}) = \frac{1}{10}\sum_{i=21}^{30} x_i^{0.1}$	

Table 2 Comparison of $pa\epsilon$-ODEMO and NSGA-II on the seven benchmark constrained problems considered in the study showing the mean, and standard deviation (SD) values for set coverage metric (C, where $X1$ and $X2$ represent $pa\epsilon$-ODEMO and NSGA-II, respectively), convergence metric (γ), and spread metric (Δ). All results have been averaged over 50 independent runs. A result with **boldface** indicates better value obtained.

Problem	Statistic	C		γ		Δ		Time (s)	
		$C(X1, X2)$	$C(X2, X1)$	$pa\epsilon$-ODEMO	NSGA-II	$pa\epsilon$-ODEMO	NSGA-II	$pa\epsilon$-ODEMO	NSGA-II
OSY	Mean	**0.26040**	0.16947	**1.31837**	1.56145	0.80109	**0.74815**	**0.64000**	0.72648
	SD	0.11974	0.05659	0.16359	0.22319	0.07555	0.08818	0.02640	0.02360
SRN	Mean	**0.12900**	0.01263	**0.06575**	0.23971	**0.15080**	0.39062	**0.14470**	0.22188
	SD	0.03190	0.00935	0.00882	0.03456	0.01632	0.04516	0.00697	0.01411
TNK	Mean	**0.11880**	0.09949	**0.00146**	0.00228	**0.367440**	0.84805	**0.42648**	0.49932
	SD	0.03243	0.03350	2.28E-4	3.59E-4	0.042210	0.10055	0.02604	0.02156
Kita	Mean	**0.55318**	0.01100	**0.01964**	0.07199	**0.29728**	0.81069	**0.11532**	0.17340
	SD	0.05346	0.01093	0.00145	0.07644	0.03246	0.14839	0.00767	0.01007
Tamaki	Mean	**0.17960**	0.08153	**0.01903**	0.02027	**0.34264**	0.51283	**0.13316**	0.22682
	SD	0.04708	0.02684	0.00075	0.00249	0.01735	0.04984	0.00793	0.03209
DTLZ8	Mean	**0.35160**	0.00373	**0.00982**	0.03549	**0.34844**	0.58427	**2.03898**	2.75628
	SD	0.06052	0.00402	7.86E-4	0.00593	0.02166	0.03546	0.07374	0.18688
DTLZ9	Mean	**0.96980**	0.81995	**0.02018**	0.02530	**0.74592**	0.78633	**2.09844**	2.56438
	SD	0.04851	0.13507	0.00573	0.00356	0.02456	0.03416	0.05457	0.12258

and three-objective problems of diverse complexities; problems with low and high dimensionality, with different types of Pareto fronts (concatenation, concave, discontinuous, thin density and non-uniform spread), and with many constraints.

6.4 Different Parameter Settings

In this work, we do not make any serious attempt to find the best parameter settings for $pa\epsilon$-ODEMO. However in this section, we perform additional experiments to show the effect of a couple of different parameter settings on the performance of $pa\epsilon$-ODEMO.

First, we keep all other parameters as described in Section 6.1, but perform our approach with λ taken from $[0.0, 1.0]$ by step 0.1. Especially, when $\lambda = 0.0$, it means that starting from evolution one of the archive solutions is selected to generate the offspring. In this case, the inefficient archive member may mislead the search. However, $\lambda = 1.0$ indicates that any archive solution is not selected to generate the offspring at all. It can be observed from Fig. 3 that different λ provides different performance in terms of convergence metric and diversity metric. For all benchmark constrained MOPs, the *good* control value λ is from $[0.5, 0.9]$.

It is very interesting that apparently if $\lambda = 0.0$, $pa\epsilon$-ODEMO produces the worst performance in terms of both convergence metric and diversity metric for all benchmark MOPs . It means that the archive member mislead the search if they take part in the generating process starting from evolution. In addition, for $\lambda = 1.0$, the performance of $pa\epsilon$-ODEMO is poor for most of benchmark MOPs. It indicates that properly allowing the archive solutions to take part in

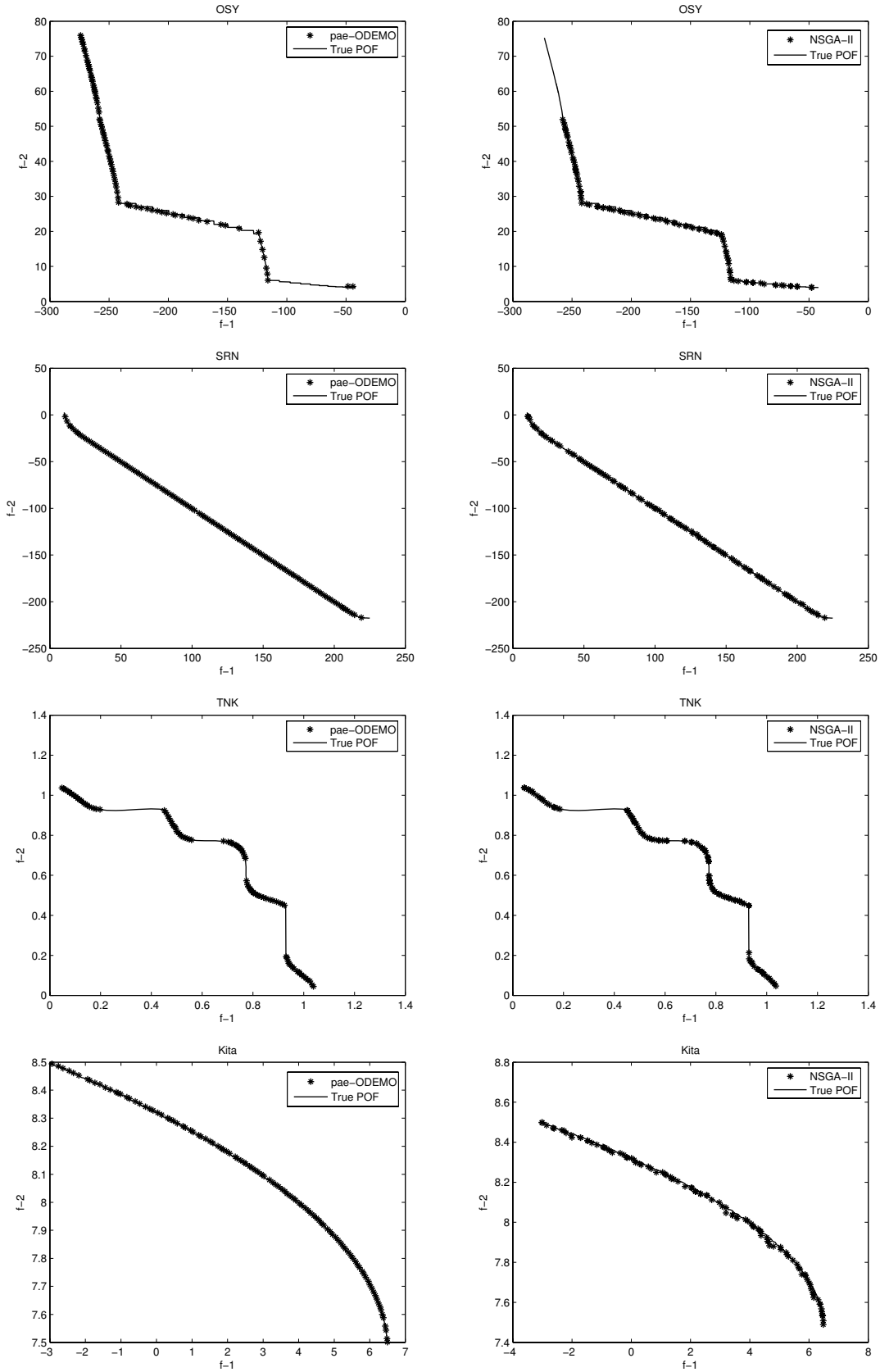

Fig. 1 Nondominated solutions of the final archive obtained by $pa\epsilon$-ODEMO and NSGA-II on two-objective MOPs. The presented fronts are the outcome of a *typical* run for $pa\epsilon$-ODEMO and NSGA-II.

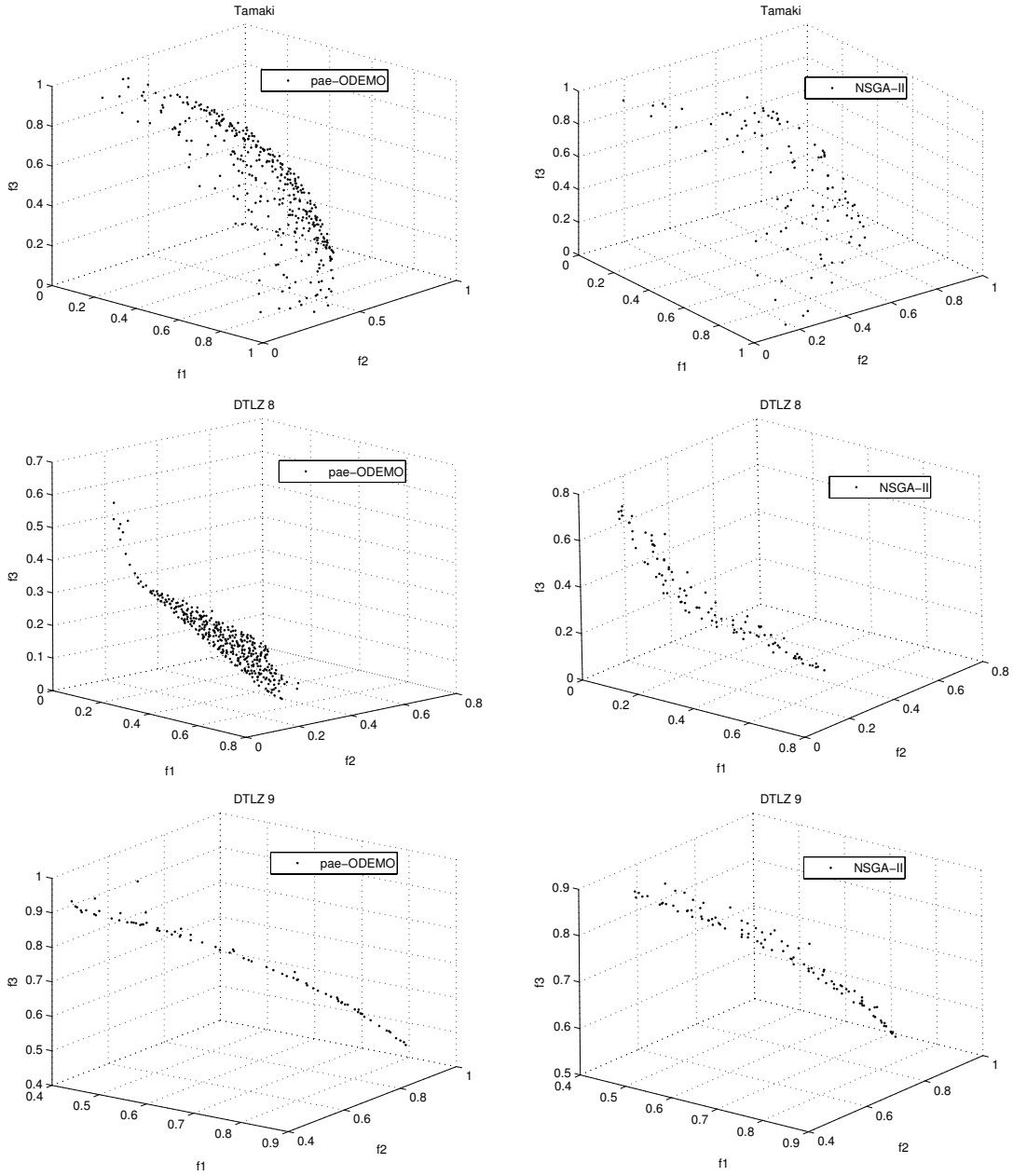

Fig. 2 Nondominated solutions of the final archive obtained by $pa\epsilon$-ODEMO and NSGA-II on three-objective MOPs. The presented fronts are the outcome of a *typical* run for $pa\epsilon$-ODEMO and NSGA-II.

the generating process can guarantee to be the best performance produced.

Second, we test different levels Q on the performance of $pa\epsilon$-ODEMO. Except for Q, all other parameters as described in Section 6.1. For OSY, SRN, TNK, Kita, and Tamaki, Q sets with 11, 17, 21, 27, 31, 37, 41, 47, 51, 57 (ticks 1 - 10 of X axis in Fig. 4 and 5). For DTLZ8 and DTLZ9, Q sets with 29, 35, 39, 45, 49, 55, 59, 65, 69, 75 (ticks 1 - 10 of X axis in Fig. 4 and 5). In addition, to validate the effect of orthogonal initial population, we generate the initial evolutionary population randomly to replace the orthogonal initial population used in $pa\epsilon$-ODEMO. The tick 0 of X axis in Fig. 4 and 5 represents the outcome of random initial population.

From Fig. 4, it can be observed that the performance of random initial population is the worst in terms of both convergence metric and diversity metric. Moreover, the effect of different levels on the performance of our approach is small. From Fig. 5, we can see that when the Q increases the running time decreases. The reason is that for the given Q and J we only need to generate the OA and the initial archive one run. Since the maximal NFFEs is fixed, when Q

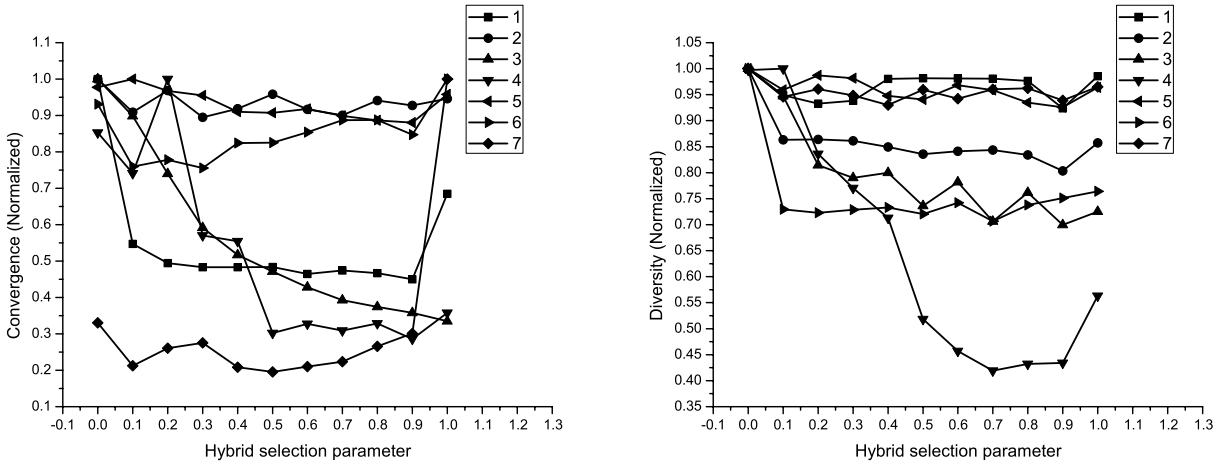

Fig. 3 Effect of different selection parameter settings on the performance of all benchmark constrained MOPs. The results were averaged over 50 independent runs. Both convergence values and diversity values for each problem are normalized. Where problems 1-7 represent OSY, SRN, TNK, Kita, Tamaki, DTLZ8, and DTLZ9, respectively, hereinafter.

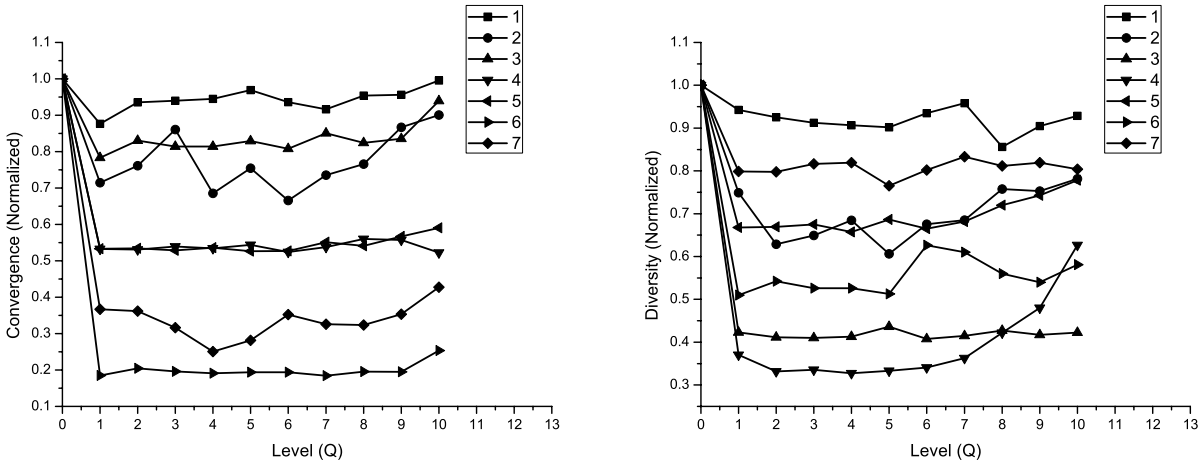

Fig. 4 Effect of different levels on the performance of all benchmark constrained MOPs. The results were averaged over 50 independent runs. Both convergence values and diversity values for each problem are normalized.

is larger, the algorithm needs more NFFEs to generate the initial archive and evolutionary population, and hence the remainder NFFEs for evolution process are smaller. In this manner, if Q is too large, the overall performance of $pa\epsilon$-ODEMO may be poor [see Fig. 4].

6.5 Engineering Design Problems

From the benchmark constrained MOPs experiments, we can see that our approach is able to handle mathematically complex problems efficiently. So this section we will use our approach to solve the engineering design problems. Four engineering design problems are selected to test the efficiency and applicability of the algorithm for multiobjective design optimization. For $pa\epsilon$-ODEMO and NSGA-II, the parameter settings are described in Section 6.1. The consistency of the algorithm was verified by running all the problems for 50 independent runs with different random seeds. However, as the performance has already been proved (see previous section), only the results for the representative sample run in 50 runs are reported here.

6.5.1 Two-bar truss design

The first problem is the two-bar truss design problem, which has been well studied by many researchers [12], [5], and

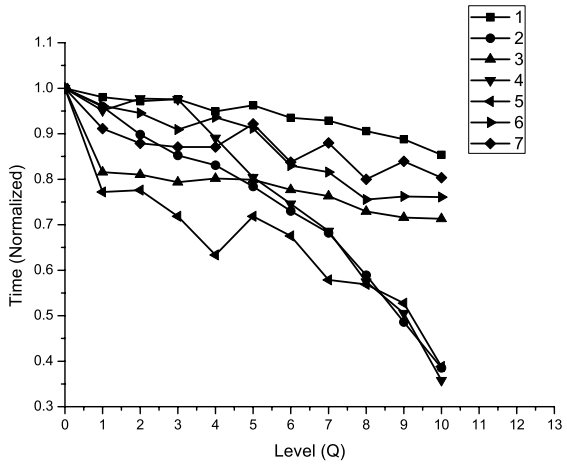

Fig. 5 Effect of different levels on the running time of all benchmark constrained MOPs. The results were averaged over 50 independent runs. The time values for each problem are normalized.

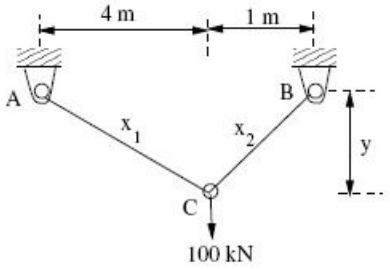

Fig. 6 The two-bar truss design problem.

[22]. The problem is shown in Fig. 6, and the mathematical description of the problem is as follows [5].

Minimize

$$\begin{cases} f_{volume} = f_1(\boldsymbol{x}) = x_1(16+y^2)^{0.5} + x_2(1+y^2)^{0.5} \\ f_{stress,AC} = f_2(\boldsymbol{x}) = 20(16+y^2)^{0.5}/(x_1 y) \end{cases} \tag{15}$$

subject to

$$\begin{cases} f_{volume} \leq 0.1 \\ f_{stress,AC} \leq 100,000 \\ f_{stress,BC} = 80(1+y^2)^{0.5}/(x_2 y) \leq 100,000 \end{cases}$$

where $1 \leq y \leq 3$, and $x_1, x_2 \geq 0$.

The maximal NFFEs of this problem is 10,000 for both $pa\epsilon$-ODEMO and NSGA-II. The nondominated solutions obtained with $pa\epsilon$-ODEMO and NSGA-II are shown in Fig. 7. From Fig. 7, it can be observed that the Pareto front of $pa\epsilon$-ODEMO is similar to the one of NSGA-II. The solutions obtained with $pa\epsilon$-ODEMO are spread in the range (0.004224m^3, 99987kPa) and (0.098645m^3, 1741.37kPa), and those obtained with NSGA-II are spread in the range (0.0043319m^3,

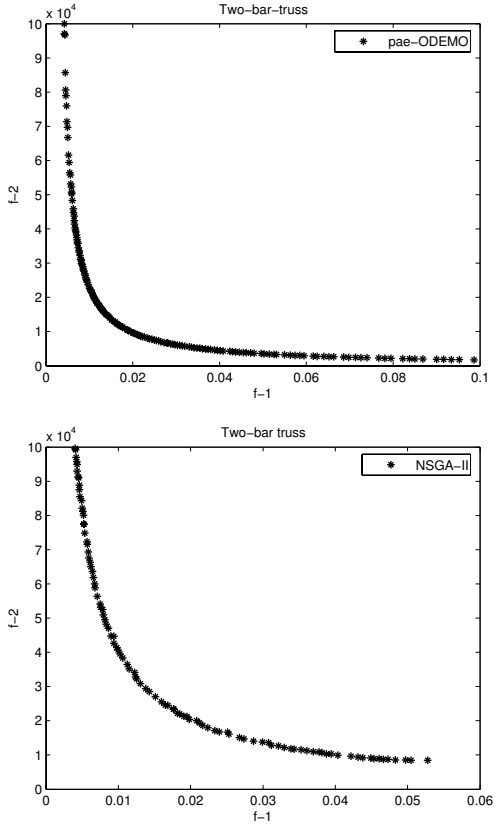

Fig. 7 The nondominated solutions obtained by $pa\epsilon$-ODEMO (top) and NSGA-II (bottom) on two-bar truss design problem.

99533.26kPa) and (0.052769m^3, 8433.61kPa). Thus $pa\epsilon$-ODEMO provides a wider spread than NSGA-II. If minimum volume is desired, our approach gives a value as low as 0.004224m^3. If minimization of stress is important, it finds a solution with stress as low as 1741.37kPa. Moreover, from Fig. 7, we can see that $pa\epsilon$-ODEMO is able to maintain a uniform distribution solutions and find a wide spread of Pareto-optimal solutions.

6.5.2 Welded beam design

The second problem is selected from [21], which is to be designed for minimum cost and minimum end deflection subject to constraints on shear stress, bending stress and buckling load (Fig. 8). The mathematical formulation of the two objective optimization problem is as follows.

Minimize

$$\begin{cases} f_1(\boldsymbol{x}) = 1.10471h^2l + 0.04811tb(14.0+t) \\ f_2(\boldsymbol{x}) = \delta(\boldsymbol{x}) = 2.1952/(t^3 b) \end{cases} \tag{16}$$

subject to

$$\begin{cases} e_1(\boldsymbol{x}) = 13,600 - \tau(\boldsymbol{x}) \geq 0 \\ e_2(\boldsymbol{x}) = 30,000 - \sigma(\boldsymbol{x}) \geq 0 \\ e_3(\boldsymbol{x}) = b - h \geq 0 \\ e_4(\boldsymbol{x}) = \mathrm{Pc}(\boldsymbol{x}) - 6,000 \geq 0 \end{cases}$$

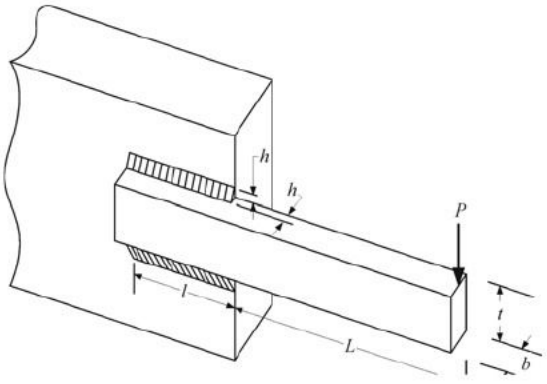

Fig. 8 The welded beam design problem.

where

$$\begin{cases} \tau(\boldsymbol{x}) = \sqrt{(\tau')^2 + (\tau'')^2 + \frac{l\tau'\tau''}{\sqrt{0.25(l^2+(h+t)^2)}}} \\ \tau' = 6,000/(\sqrt{2}hl) \\ \tau'' = \frac{6,000(14.0+0.5l)\sqrt{0.25(l^2+(h+t)^2)}}{2[0.707hl(l^2/12+0.25(l^2+(h+t)^2))]} \\ \sigma(\boldsymbol{x}) = 504,000/(t^2b) \\ \mathrm{Pc}(\boldsymbol{x}) = 64,764.022(1-0.0282346t)tb^3 \\ 0.125 \leq h, b \leq 5.0, \mathrm{and} 0.1 \leq l, t \leq 10.0 \end{cases}$$

Fig. 9 shows the non-dominated solutions obtained after 15,000 NFFEs for both $pa\epsilon$-ODEMO and NSGA-II. It can be seen that $pa\epsilon$-ODEMO is also capable of maintaining a uniform distribution of solutions. $pa\epsilon$-ODEMO found the minimal cost solution as 2.8959 units with deflection 0.0131 inches, and the minimal deflection as 0.00044 with a cost of 36.6172 units. For NSGA-II, the minimal cost is 3.0294 units for deflection of 0.0088 units, and the minimal deflection is 0.000439 with a cost of 37.4018 units. The extreme solutions are captured well in $pa\epsilon$-ODEMO. Again, it is indicated that our proposed approach is efficient and is able to find a wide variety of Pareto-optimal solutions.

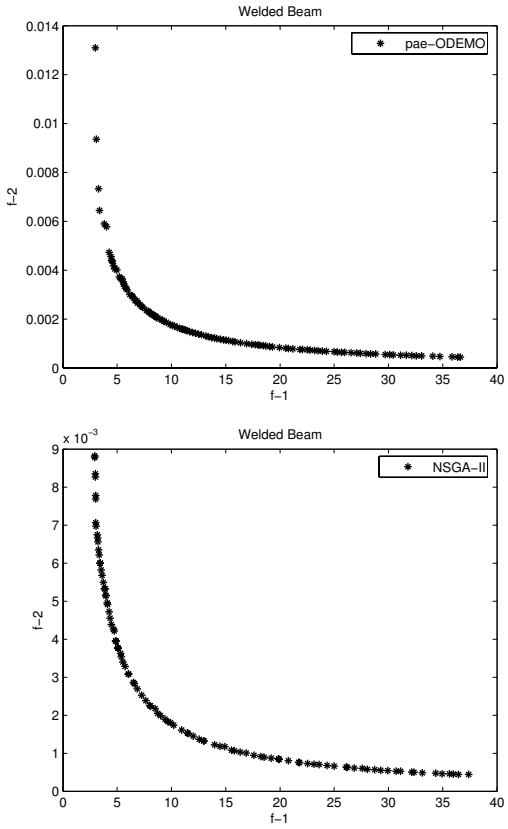

Fig. 9 The nondominated solutions obtained by $pa\epsilon$-ODEMO (top) and NSGA-II (bottom) on welded beam design problem.

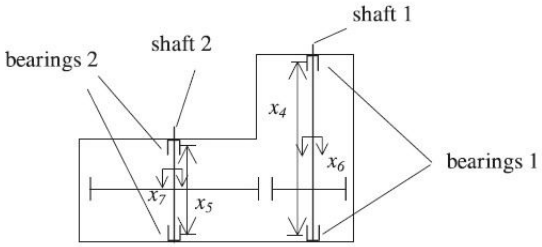

Fig. 10 The speed reducer design problem.

6.5.3 Speed reducer design

The third design problem is the speed reducer design problem shown in Fig. 10. This problem has also been studied in [5], [12], [15], [24], and so on. The problem is described as follows [5].

Minimize

$$\begin{cases} f_{weight} = f_1(\boldsymbol{x}) = \\ \quad 0.7854x_1x_2^2(10x_3^2/3 + 14.933x_3 - 43.0934) \\ \quad -1.508x_1(x_6^2 + x_7^2) + 7.477(x_6^3 + x_7^3) \\ \quad +0.7854(x_4x_6^2 + x_5x_7^2) \\ f_{stress} = f_2(\boldsymbol{x}) = \frac{\sqrt{(\frac{745x_4}{x_2x_3})^2 + 1.69\times10^7}}{0.1x_6^3} \end{cases} \quad (17)$$

subject to

$$\begin{cases} e_1(\boldsymbol{x}) = 1/27 - 1/(x_1x_2^2x_3) \geq 0 \\ e_2(\boldsymbol{x}) = 1/397.5 - 1/(x_1x_2^2x_3^2) \geq 0 \\ e_3(\boldsymbol{x}) = 1/1.93 - x_4^3/(x_2x_3x_6^4) \geq 0 \\ e_4(\boldsymbol{x}) = 1/1.93 - x_5^3/(x_2x_3x_7^4) \geq 0 \\ e_5(\boldsymbol{x}) = 40 - x_2x - 3 \geq 0 \\ e_6(\boldsymbol{x}) = 12 - x_1/x_2 \geq 0 \\ e_7(\boldsymbol{x}) = x_1/x_2 - 5 \geq 0 \\ e_8(\boldsymbol{x}) = x_4 - 1.5x_6 - 1.9 \geq 0 \\ e_9(\boldsymbol{x}) = x_5 - 1.1x_7 - 1.9 \geq 0 \\ e_{10}(\boldsymbol{x}) = 1,300 - f_2(\boldsymbol{x}) \geq 0 \\ e_{11}(\boldsymbol{x}) = 1,100 - \frac{\sqrt{(\frac{745x_5}{x_2x_3})^2 + 1.275\times10^8}}{0.1x_7^3} \geq 0 \end{cases}$$

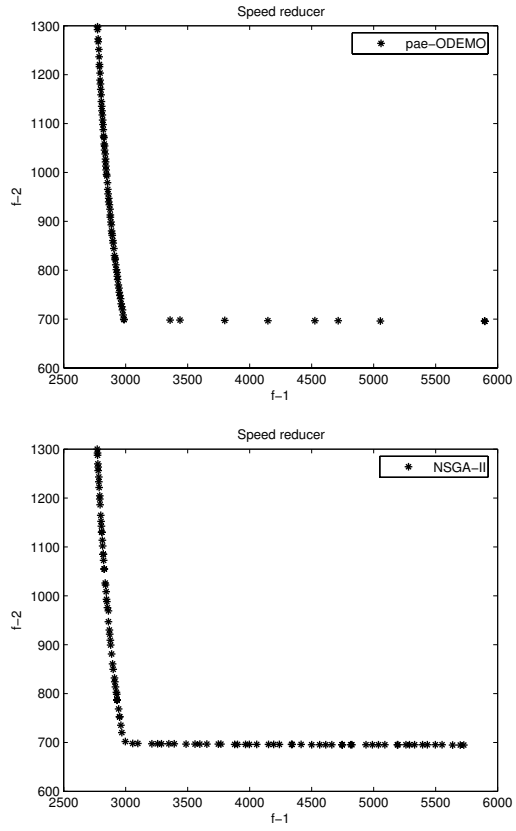

Fig. 11 The nondominated solutions obtained by $pa\epsilon$-ODEMO (top) and NSGA-II (bottom) on speed reducer design problem.

where $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4, x_5 \leq 8.3$, $2.9 \leq x_6 \leq 3.9$, and $5.0 \leq x_7 \leq 5.5$.

The Pareto fronts are presented in Fig. 11 obtained with $pa\epsilon$-ODEMO and NSGA-II after 15,000 NFFEs. $pa\epsilon$-ODEMO results in an extended Pareto curve between (2775.02, 1298.01) and (5897.4, 695.348). And the solutions obtained with NSGA-II are spread in the range (2772.08, 1299.8) and (5728.87, 696.726). Thus both have a wide variety of alternatives. The proposed $pa\epsilon$-ODEMO solutions are very competitive with NSGA-II solutions in terms of both closeness to the true optimum front and their spread. It is worth to point out that the solutions in the horizontal segment of the Pareto front are obtained with $pa\epsilon$-ODEMO because of the $pa\epsilon$-dominance adopted in $pa\epsilon$-ODEMO.

6.5.4 Disc brake design

The fourth example consists in optimizing the disc brake design problems studied by [21]. The objectives of the design are to minimize the mass of the brake and to minimize the stopping time. The variables are the inner radius of the discs, outer radius of the discs, the engaging force and the number of friction surfaces and are represented as x_1, x_2, x_3 and x_4 respectively. The constraints for the design include minimum distance between the radii, maximum length of the brake, pressure, temperature and torque limitations. The problem is a mixed, constrained, multiobjective problem. The mathematical description of this problem is as follows [21].

Minimize

$$\begin{cases} f_1(\boldsymbol{x}) = 4.9 \times 10^{-5}(x_2^2 - x_1^2)(x_4 - 1) \\ f_2(\boldsymbol{x}) = \frac{9.82 \times 10^6 (x_2^2 - x_1^2)}{x_3 x_4 (x_2^3 - x_1^3)} \end{cases} \tag{18}$$

subject to

$$\begin{cases} e_1(\boldsymbol{x}) = (x_2 - x_1) - 20 \geq 0 \\ e_2(\boldsymbol{x}) = 30 - 2.5(x_4 + 1) \geq 0 \\ e_3(\boldsymbol{x}) = 0.4 - x_3/(3.14(x_2^2 - x_1^2)) \geq 0 \\ e_4(\boldsymbol{x}) = 1 - \frac{2.22 \times 10^{-3} x_3 (x_2^3 - x_1^3)}{(x_2^2 - x_1^2)^2} \geq 0 \\ e_5(\boldsymbol{x}) = \frac{2.66 \times 10^{-2} x_3 x_4 (x_2^3 - x_1^3)}{(x_2^2 - x_1^2)} - 900 \geq 0 \end{cases}$$

where $55 \leq x_1 \leq 80$, $75 \leq x_2 \leq 110$, $1,000 \leq x_3 \leq 3,000$, and $2 \leq x_4 \leq 20$.

The maximal NFFEs of this problem is 5,000 for both $pa\epsilon$-ODEMO and NSGA-II. The nondominated solutions obtained with $pa\epsilon$-ODEMO and NSGA-II are shown in Fig. 12. It is presented from Fig. 12 that both algorithms are able to obtain a wide variety of solutions which are uniformly spread. The solutions obtained with $pa\epsilon$-ODEMO are spread in the range (0.1274, 16.6549) and (2.8684, 2.0906), and those obtained with NSGA-II are spread in the range (0.1293, 17.598) and (2.7915, 2.1127). It is interesting to note that NSGA-II found the minimal mass of the brake as 0.1293 with stopping time 17.598, this solution is dominated by the minimal mass of the brake as 0.1274 with stopping time 16.6549 obtained with $pa\epsilon$-ODEMO.

Moreover, in order to verify our approach, the four metrics (convergence metric, diversity metric, C metric, and running time) are applied in the four engineering MOPs. Note that in all the four problems we generated the true (or global) Pareto-optimal front by enumeration, using parallel processing techniques (some of these fronts require a rather high computational effort to be generated by enumeration).

Table 3 shows the results of $pa\epsilon$-ODEMO and NSGA-II. The statistics shown are based on 50 independent runs with different random seeds. From this table it can be observed that NSGA-II requires more running time than $pa\epsilon$-ODEMO for all four engineering problems, and hence it has larger time complexity. $pa\epsilon$-ODEMO is able to obtain better convergence values than NSGA-II for all engineering MOPs. In addition, for the C metric, except for Disc brake design problem, NSGA-II gives worse C values than $pa\epsilon$-ODEMO. Moreover, $pa\epsilon$-ODEMO gives smaller diversity values than NSGA-II for three engineering problems out of four. For the speed reducer design problem the diversity of $pa\epsilon$-ODEMO is worse than NSGA-II, the reason is that $pa\epsilon$-ODEMO retains only a few solutions in the horizontal segment of the Pareto-optimal front [see Fig. 11].

Table 3 Comparison of $pa\epsilon$-ODEMO and NSGA-II on the four engineering problems considered in the study showing the mean, and standard deviation (SD) values for set coverage metric (C, where $X1$ and $X2$ represent $pa\epsilon$-ODEMO and NSGA-II, respectively), convergence metric (γ), and spread metric (Δ). All results have been averaged over 50 independent runs. A result with **boldface** indicates better value obtained.

Problem	Statistic	C		γ		Δ		Time (s)	
		$C(X1, X2)$	$C(X2, X1)$	$pa\epsilon$-ODEMO	NSGA-II	$pa\epsilon$-ODEMO	NSGA-II	$pa\epsilon$-ODEMO	NSGA-II
Two-bar truss	Mean	**0.95900**	0.00240	**254.9408**	439.7384	**0.87393**	0.93725	**0.26560**	0.30660
	SD	0.01669	0.00497	48.2813	52.8976	0.05032	0.02425	0.02685	0.00876
Welded beam	Mean	**0.19620**	0.14481	**0.09169**	0.16875	**0.58607**	0.88987	**0.37520**	0.46880
	SD	0.08366	0.04019	0.00733	0.08030	0.04366	0.11976	0.01940	0.02451
Speed reducer	Mean	**0.26524**	0.05000	**2.69281**	3.0771	0.84041	**0.79717**	**0.42500**	0.46260
	SD	0.13005	0.03411	0.24051	0.10782	0.20085	0.06608	0.00671	0.00876
Disc brake	Mean	0.19800	**0.20418**	**0.05175**	0.08203	**0.61511**	0.66542	**0.13780**	0.16880
	SD	0.09348	0.09952	0.00579	0.01957	0.04581	0.01431	0.00716	0.00661

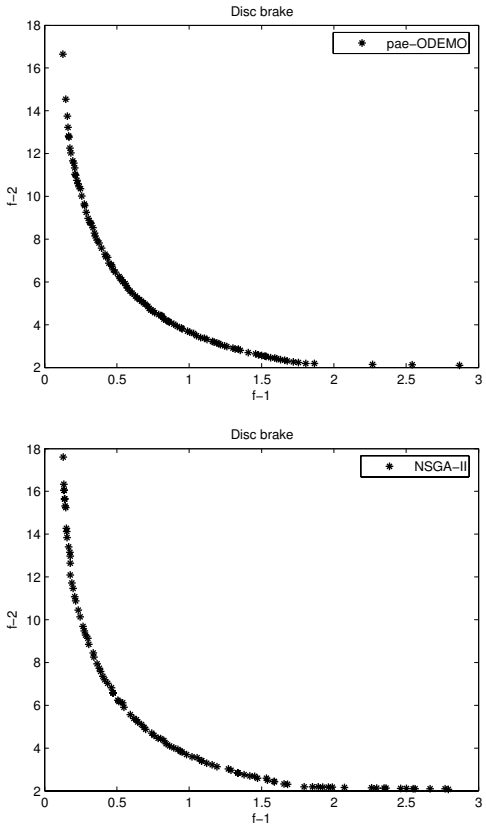

Fig. 12 The nondominated solutions obtained by $pa\epsilon$-ODEMO (top) and NSGA-II (bottom) on disc brake design problem.

From the above analysis we can see that the $pa\epsilon$-ODEMO solutions are very competitive with NSGA-II solutions in terms of both closeness to the true optimum front and their spread for all test problems used in this study. $pa\epsilon$-ODEMO does not have any difficulty in achieving a good spread of Pareto-optimal solutions for constrained multiobjective optimization. The results obtained for both benchmark constrained MOPs and engineering design problems amply demonstrate that the $pa\epsilon$-ODEMO technique can yield *efficient*, *uniformly distributed*, *near-complete* and *near-optimal* Pareto-optimal solutions for multiobjective optimization.

7 Conclusions

In this paper, an efficient multiobjective DE algorithm is presented to solve constrained MOPs. It is characterized by a) employing the orthogonal design method with quantization technique to generate the initial population, b) adopting an archive to store the nondominated solutions and employing the new Pareto-adaptive ϵ-dominance to update the archive at each generation, c) employing a new constraint-handling method to handle the constraints, and d) using a hybrid selection mechanism in which a random selection and elitist selection are interleaved in order to allow the archive solution to guide the search towards the Pareto-optimal front.

Seven benchmark constrained MOPs are chosen to test the capabilities of the algorithm in handling mathematically complex problems. And four well designed problems are selected to validate the efficiency and applicability of the algorithm for multiobjective design optimization. Compared with NSGA-II, one of the best MOEAs available at present, the results show that $pa\epsilon$-ODEMO are very competitive with NSGA-II in terms of both closeness to the true optimum front and their spread for the four all test problems. The $pa\epsilon$-ODEMO technique can yield *efficient*, *uniformly distributed*, *near-complete* and *near-optimal* Pareto-optimal solutions for all test MOPs used in this work. So, we can conclude that our approach is a viable alternative to efficient multiobjective optimization.

For the proposed $pa\epsilon$-ODEMO, we have only chosen a reasonable set of value and have not made any effort in finding the best parameter settings. Our future study consists on investigating the effect of different parameter settings on the performance of $pa\epsilon$-ODEMO.

Acknowledgements The authors would like to acknowledge the anonymous reviewers and the Editor for their useful comments and constructive suggestions. The authors also sincerely thank Dr. A.G. Hernández-Díaz for providing us with the $pa\epsilon$-MyDE code and his suggestions on our work. This work is supported by the Humanities Base Project of Hubei Province under grant No. 2004B0011 and the Natural Science Foundation of Hubei Province under grant No. 2003ABA043.

References

1. Abbass HA, Sarker R, Newton C (2001) PDE: A Pareto-frontier Differential Evolution Approach for Multi-objective Optimization Problems. In: Proceedings of the 2001 IEEE Congress on Evolutionary Computation CEC2001, pp 971 – 978
2. Cai ZH, Gong WY, Huang YQ (2007) A Novel Differential Evolution Algorithm based on ϵ-domination and Orthogonal Design Method for Multiobjective Optimization. In: Obayashi S et al (ed) Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-07), pp 286 – 301
3. Coello CAC (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 191:1245 – 1287
4. Coello CAC (2006) Evolutionary Multi-objective Optimization: A Historical View of the Field. IEEE Computational Intelligence Magazine 1:28 – 36
5. Coello CAC, Pulido G (2005) Multiobjective Structural Optimization Using a Microgenetic Algorithm. Struct Multidisc Optim 30:388 – 403
6. Deb K, Pratap A, Meyarivan T (2001) Constrained Test Problems for Multi-objective Evolutionary Optimization. In: Zitzler E et al (ed) Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO-01) 284 – 298
7. Deb K, Thiele L, Laumanns M, Zitzler E (2001) Scalable Test Problems for Evolutionary Multi-Objective Optimization. Tech. Rep. TIK-Technical Report No. 112
8. Deb K, Pratap A, Agarwal, S, Meyarivan T (2002) A Fast and Elitist Multiobjective Genetic Algorithm: NSGA - II. IEEE Transactions on Evolutionary Computation 6:182 – 197
9. Deb K, Mohan M, Mishra S (2003) Towards a Quick Computation of Well-spread Pareto-optimal Solutions. In: Fonseca CM et al (ed) Proceedings of the Second International Conference on Evolutionary Multi-Criterion Optimization (EMO-03), pp 222 – 236
10. Deb K, Mohan M, Mishra S (2005) Evaluating the ϵ-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions. Evolutionary Computation 13:501 – 525
11. Fang KT, Ma CX (2001) Orthogonal and Uniform Design (in Chinese). Science Press, Beijing, 2001
12. Farhang-Mehr A, Azarm S (2002) Entropy-based Multi-objective Genetic Algorithm for Design Optimization. Struct Multidisc Optim 24:351 – 361
13. Hernández-Díaz AG, Santana-Quintero LV, Coello CAC, et al (2006) Pareto-adaptive ϵ-dominance. Tech. Rep. Resport EVOCINV-02-2006
14. Kukkonen S, Lampinen J (2005) GDE3: The third Evolution Step of Generalized Differential Evolution. In: Proceedings of the 2005 IEEE Congress on Evolutionary Computation CEC2005, pp 443 – 450
15. Kurpati A, Azarm S, Wu J (2002) Constraint Handling Improvements for Multiobjective Genetic Algorithms. Struct Multidisc Optim 23:204 – 213
16. Laumanns M, Thiele L, Deb K, Zitzler E (2002) Combining Convergence and Diversity in Evolutionary Multi-objective Optimization. Evolutionary Computation 10:263 – 282
17. Leung YW, , Wang Y (2001) An Orthogonal Genetic Algorithm with Quantization for Global Numerical Optimization. IEEE Transactions on Evolutionary Computation 5:41 – 53
18. Madavan NK (2002) Multiobjective Optimization Using a Pareto Differential Evolution Approach. In: Proceedings of the 2002 IEEE Congress on Evolutionary Computation CEC2002, pp 1145 – 1150
19. Oyama A, Shimoyama K, Fujii K (2007) New Constraint-handling Method for Multi-objective and Multi-Monstraint Evolutionary Optimization. Transactions of the Japan Society for Aeronautical and Space Sciences 50:56 – 62
20. Rajasekaran S, Lavanya S (2007) Hybridization of Genetic Algorithm with Immune System for Optimization Problems in Structural Engineering. Struct Multidisc Optim 34:415 – 429
21. Ray T, Liew KM (2002) A Swarm Metaphor for Multiobjective Design Optimization. Engineering Optimization 34:141 – 153
22. Reddy MJ, Kumar DN (2007) An efficient Multi-objective Optimization Algorithm Based on Swarm Intelligence for Engineering Design. Engineering Optimization 39:49 – 68
23. Robič T, Filipič B (2005) DEMO: Differential Evolution for Multiobjective Optimization. In: Coello CAC et al (ed) Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization (EMO-05), pp 520 – 533
24. Santana-Quintero LV, Coello CAC (2005) An Algorithm Based on Differential Evolution for Multi-Objective Problems. International Journal of Computational Intelligence Research 1:151 – 169
25. Srinivas N, Deb K (2000) Multi-objective Function Optimization Using Nondominated Sorting Genetic Algorithms. Evolutionary Computation 2:221 – 248
26. Storn R, Price K (1997) Differential Evolution - A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces. Journal of Global Optimization 11:341 – 359
27. Storn R, Price K (2003) Home page of differential evolution. Tech. rep., online at http://www.ICSI.Berkeley.edu/~storn/code.html.
28. Xue F, Sanderson AC, Graves RJ (2003) Pareto-based Multi-objective Differential Evolution. In: Proceedings of the 2003 IEEE Congress on Evolutionary Computation CEC2003, pp 862 – 869
29. Zeng SY, Kang LS, Ding LX (2004) An Orthogonal Multiobjective Evolutionary Algorithm for Multi-objective Optimization Problems with Constraints. Evolutionary Computation 12:77 – 98
30. Zitzler E, Thiele L (1999) Multiobjective Evolutionary Algorithms: A Comparative Sase Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation 3:257 – 271
31. Zitzler E, Deb K, Thiele L (2000) Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Evolutionary Computation 8:173 – 195
32. Zitzler E, Laumanns M, Thiele L (2001) Spea2: Improving the strength pareto evolutionary algorithm. Tech. Rep. Technical Report 103, Computer Engineering and Networks Laboratory
33. Zitzler E, Thiele L, Laumanns M, Fonseca C, et al (2003) Performance Assesement of Multiobjective Optimizer: An Analysis and Review. IEEE Transactions on Evolutionary Computation 7:117 – 132