An Efficient Multi-objective Evolutionary Algorithm: OMOEA-II

Sanyou Zeng[1,2], Shuzhen Yao[1], Lishan Kang[3], and Yong Liu[1]

[1] Dept. of Computer Science, China University of GeoSciences, 430074 Wuhan, Hubei, P.R.China
yliu@u-aizu.ac.jp, sanyou-zeng@263.net
[2] Dept. of Computer Science, Zhuzhou Institute of Technology, 412008 Zhuzhou, Hunan, P.R.China
[3] State Key Laboratory of Software Engineering, Wuhan University 430072 Wuhan, Hubei, P.R.China

Abstract. An improved orthogonal multi-objective evolutionary algorithm (OMOEA), called OMOEA-II, is proposed in this paper. Two new crossovers used in OMOEA-II are orthogonal crossover and linear crossover. By using these two crossover operators, only small orthogonal array rather than large orthogonal array is needed for exploiting optimal in the global space. Such reduction in orthogonal array can avoid exponential creation of solutions of OMOEA and improve the performance in robusticity without degrading precision and distribution of solutions. Experimental results show that OMOEA-II can solve problems with high dimensions and large number of local Pareto-optimal fronts better than some existing algorithms recently reported in the literatures.
Keywords: evolutionary algorithms; multi-objective optimization; Pareto optimal set.

1 Introduction

Almost every real-world problem involves simultaneous optimization of several incommensurable and often competing objectives. Evolutionary algorithms have the ability to find multiple Pareto-optimal solutions in one single simulation run. They have often been used to solve multi-objective problems. Such as vector evaluated genetic algorithm (VEGA)[1], Hajela and Lins genetic algorithm(HLGA) [2], pareto-based ranking procedure(FFGA) [3], niched Pareto genetic algorithm (NPGA) [4], pareto archived evolution strategy (PAES)[5], nondominated sorting genetic algorithm (NSGA-II)[6], strength pareto evolutionary algorithm (SPEA2) [7], rMOGAxs [8], and generalized regression GA (GRGA) [9].

Orthogonal design method [10] is developed to sample a small, but representative set of combinations for experimentation to obtain good combination. Leung and Zhang incorporated orthogonal design in genetic algorithm for single objective problems[11][12], found such method was more robust and statistically sound. In [13], orthogonal design method is used in multi-objective evolutionary algorithm and developed algorithm was called OMOEA. It was showed that

OMOEA could find good solutions. But OMOEA degraded its performance on both precision and distribution of the yielded solutions for problems with strong interaction between variables, and when the number of objectives increases, the solutions yielded by OMOEA increased exponentially.

In this paper, an improved version of OMOEA (OMOEA-II) is proposed. Orthogonal design method is nested in crossover operator to select better genes as offsprings, and consequently, enhances the performance of OMOEA. Both orthogonal crossover and linear crossover are used in OMOEA-II. By combining two crossover operators, faster convergence and better solutions are obtained.

The rest of this paper is organized as follows. Section 2 briefly describes multi-objective optimization problem and orthogonal design method. Section 3 presents the proposed OMOEA-II. Section 4 shows experiment results and discussions. Finally, Section 5 concludes with a summary of the paper.

2 Preliminary

2.1 Problem Definition

Definition 1. *(**Multi-objective Optimization Problem(MOP)**) A general MOP includes a set of N parameters (decision variables), a set of K objective functions, and a set of L constraints. Objective functions and constraints are functions of the decision variables. The optimization goal is to*

$$\begin{aligned} &minimize\ \boldsymbol{y} = \boldsymbol{f}(x) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_K(\boldsymbol{x})) \\ &subject\ to\ \boldsymbol{e}(\boldsymbol{x}) = (e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), ..., e_L(\boldsymbol{x})) \leq \boldsymbol{0} \\ &where \quad \boldsymbol{x} = (x_1, x_2, ..., x_N) \in \mathcal{X} \\ &\qquad \mathcal{X} = \{(x_1, x_2, ..., x_N) | l_i \leq x_i \leq u_i, i = 1, 2, ..., N\} \\ &\qquad \boldsymbol{z} = (z_1, z_2, ..., z_N) \\ &\qquad \boldsymbol{l} = (l_1, l_2, ..., l_N) \\ &\qquad \boldsymbol{u} = (u_1, u_2, ..., u_N) \\ &\qquad \boldsymbol{y} = (y_1, y_2, ..., y_K) \in \mathcal{Y} \end{aligned} \tag{1}$$

*where $\boldsymbol{x}$ is the **decision vector**, $\boldsymbol{y}$ is the **objective vector**, $\mathcal{X}$ denotes the decision space, $\boldsymbol{z}$ is the center of the decision space, $\boldsymbol{l}$ and $\boldsymbol{u}$ are the upper bound and lower bound of the decision space, and $\mathcal{Y}$ is called the objective space.*

2.2 Orthogonal Design Methods

An example was introduced in [14] to explain the basic concept of experimental design methods. The yield of a vegetable depends on: 1) the temperature, 2) the amount of fertilizer, and 3) the pH value of the soil. These three quantities are called the factors of the experiment. Each factor has three possible levels shown in Table 1. To find the best combination of levels for a maximum yield, we can do one experiment for each combination, and then select the best one. In the above example, there are $3 \times 3 \times 3 = 27$ combinations. and hence there are 27 experiments needed. In general, when there are N factors and Q levels, there

are Q^N combinations. When N and Q are large, it may not be possible to do all Q^N experiments. Therefore, it is desirable to sample a small, but representative set of combinations for experimentation. The orthogonal design was developed for such purpose [14]. Let $L_M(Q^N)$ be an orthogonal array for N factors and Q levels, where "L" denotes a Latin square and M the number of combination of levels. It has M rows ,where every row represents a combination of levels. By applying orthogonal array $L_M(Q^N)$, we only select M combinations to be tested, where M may be much smaller than Q^N, For convenience, we denote $L_M(Q^N) = [a_{i,j}]_{M\times N}$ where the jth factor in the ith combination has level $a_{i,j}$ and $a_{i,j} \in \{1, 2, ..., Q\}$, and the corresponding yields of the M combinations by $[y_i]_{M\times 1}$, where the ith combination (experiment) has yield y_i . The following is an orthogonal array:

$$L_9(3^3) = \begin{bmatrix} 1\,1\,1 \\ 1\,2\,2 \\ 1\,3\,3 \\ 2\,1\,2 \\ 2\,2\,3 \\ 2\,3\,1 \\ 3\,1\,3 \\ 3\,2\,1 \\ 3\,3\,2 \end{bmatrix} \tag{2}$$

In $L_9(3^3)$, there are three factors, three levels per factor, and nine combination of levels. The three factors have respective levels $1, 1, 1$ in the first combination, $1, 2, 2$ in the second combination, etc. We apply orthogonal array $L_9(3^3)$ to select nine combinations to be tested. The nine combination and their yields are shown in Table 2. From the yields of the selected combinations, a promising solution can be obtained by statistical methods. Firstly, the mean value of the yield for each factor at each level is calculated, where each factor has a level with best mean value. Secondly, the combination of the best levels is chosen as promising solution. For example, the mean yields for temperature at levels 1, 2 and 3 can be calculated by averaging yields for the experiments $1-2-3$, $4-5-6$ and $7-8-9$, respectively. The mean yields at different levels for other factors can be computed in a similar manner. The mean yields are shown in Table 3. From Table 3, we can see the best levels of temperature, amount of fertilizers and pH values are 25, 150 and 8 respectively. Therefore, we regard $(25^oC, 150g/m^2, 8)$ as a promising solution. Such solution may not really be optimal. However, orthogonal design has been proven to be optimal for additive and quadratic models.

A special class of orthogonal arrays $L_M(Q^P)$, which we shall use a simple permutation method to construct, will be used in this paper, where Q is prime and $M = Q^J$, where J is a positive integer satisfying

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

Denote the jth column of the orthogonal array $[a_{i,j}]_{M\times P}$ by $\mathbf{a}_j$. Column $\mathbf{a}_j$ for $j = 1, 2, (Q^2-1)/(Q-1)+1, (Q^3-1)/(Q-1)+1, ..., (Q^{J-1}-1)/(Q-1)+1$ are

Table 1. Experimental design problem with three factors and three levels per factor.

	Factors		
Levels	Temperature	Amount of fertilizers	pH value
Level 1	$20^{o}C$	$100g/m^2$	6
Level 2	$25^{o}C$	$150g/m^2$	7
Level 3	$30^{o}C$	$200g/m^2$	8

Table 2. Nine representative combinations for experimentation and their yields based on the orthogonal array $L_9(3^3)$.

	Factors			
combination	Temperature(^{o}C)	Amount of fertilizers(g/m^2)	pH value	yield
1	1(20)	1(100)	1(6)	2.75
2	1(20)	2(150)	2(7)	4.52
3	1(20)	3(200)	3(8)	4.65
4	2(25)	1(100)	2(7)	4.60
5	2(25)	2(150)	3(8)	5.58
6	2(25)	3(200)	1(6)	4.10
7	3(30)	1(100)	3(8)	5.32
8	3(30)	2(150)	1(6)	4.10
9	3(30)	3(200)	2(7)	4.37

Table 3. The mean yields for each factor at different levels.

	Mean yield		
Level	Temperature	Amount of fertilizers	pH value
Level1	3.97	4.22	3.65
Level2	4.76	4.73	4.50
Level3	4.60	4.37	5.18

called *basic columns*. and the others are called *nonbasic columns*. The algorithm first constructs the basic columns, and then generates the nonbasic columns. The details are as follows.

Algorithm 1 *Construction of orthogonal array* $L_M(Q^P)$
//Construct the basic columns as follows:
  FOR $k = 1$ *TO* J
    $j=\frac{Q^{k-1}-1}{Q-1}+1$;
    FOR $i = 1$ *TO* Q^J
      $a_{i,j} = \lfloor \frac{i-1}{Q^{J-k}} \rfloor \ mod\ Q$;
    ENDFOR
  ENDFOR
//Construct the nonbasic columns as follows:
  FOR $k = 2$ *TO* J
    $j=\frac{Q^{k-1}-1}{Q-1}+1$;
    FOR $s = 1$ *TO* $j-1$, $t = 1$ *TO* $Q-1$
      $\mathbf{a}_{j+(s-1)(Q-1)+t} = (\mathbf{a}_s \times t + \mathbf{a}_j)\ mod\ Q$;
    ENDFOR
  ENDFOR
  Increment $a_{i,j}$ *by one for* $-1 \le i \le M$ *and* $1 \le j \le P$; ♯

The used orthogonal array in this paper is required to satisfy $P \ge N$ and M is as small as possible. That is the columns P of $L_M(Q^P)$ must be larger than the number of factors (or decision variables) in the hope of sampling small number of points (combinations) for obtaining better solution. It only needs to determine Q and J for determining $L_M(Q^P)$. $L_M(Q^P)$ is determined by solving the following minimization problem:

$$\begin{aligned} &Minimize\ \ M = Q^J \\ &Subject\ \ to\ P = \tfrac{Q^J-1}{Q-1} > N \end{aligned} \tag{4}$$

where Q is a prime and $Q \ge 3$, J is a positive integer.

$L_M(Q^P)$ is the full size of the orthogonal array, which has P columns. For a problem with N decision variables, we discard the last $P-N$ columns of $L_M(Q^P)$ and get an orthogonal array $L_M(Q^N)$.

The proposed algorithm will require the mean value of the objective at each level of each factor. Denote the objective values of the orthogonal experiments by $[y_i]_{M\times 1}$ where the objective has the value y_i at the ith combination, the mean values by $[\Delta_{k,j}]_{Q\times N}$ where the objective has the mean value $\Delta_{k,j}$ at the kth level of the jth factor, and

$$\Delta_{k,j} = \frac{Q}{M} \sum_{a_{i,j}=k} y_i \tag{5}$$

where the orthogonal array $L_M(Q^N)$ has the value $a_{i,j}$ at ith row and jth column. That is, the jth factor has level $a_{i,j}$ in the ith combination(experiment). The

objective has value y_i at the ith combination, and $\sum_{a_{i,j}=k} y_i$ implies the sum of y_i where $\forall i$ satisfy $a_{i,j} = k$. The details of the algorithm are as follows

Algorithm 2 *Calculation of mean value* $[\Delta_{k,j}]_{Q\times N}$
$[\Delta_{k,j}]_{Q\times N} = [0]_{Q\times N}$;
//Add up objective result for each factor at each level
FOR $i = 1$ *TO* M, $j = 1$ *TO* N
q=$a_{i,j}$;$\Delta_{q,j} = \Delta_{q,j} + y_i$;
ENDFOR
//Average results for each factor at each level
$[\Delta_{k,j}]_{Q\times N} = [\Delta_{k,j}]_{Q\times N} \times Q/M$ ♯

Each factor has its best level according to the mean value matrix $[\Delta_{k,j}]_{Q\times N}$, The combination $\mathbf{s}$ of the best levels is potentially a good solution. For minimization problems, it is calculated by

$$\begin{array}{l} \Delta_{k_j,j} = min\{\Delta_{1,j}, \Delta_{2,j}, ..., \Delta_{Q,j}\}, j = 1, 2, ..., N \\ \mathbf{s} = (k_1, k_2, ..., k_N) \end{array} \quad (6)$$

3 OMOEA-II

OMOEA , proposde by the author [13], is an evolutionary algorithm by introducing orthogonal design for multi-objective optimization . It is good at finding good solutions for problems with decision variables relatively independent. However, it was found there were two problems.

1. Strong interaction between variables degrades the performance of OMOEA including both precision and distribution of the yielded solutions.
2. As number of objectives increase, the yielded solutions will increase exponentially.

The idea of OMOEA-II is to replace a large orthogonal array by a small orthogonal array in order to exploit optimal efficiently relatively small space which is generated by randomly choosing individuals in the population.

3.1 Framework of OMOEA-II

Algorithm 3 *Framework of OMOEA-II*

1. *Randomly create population* $\boldsymbol{P}_0$ *with size* N_p. *Set counter* $t = 0$
2. *Execute crossover operator (Algorithm 6) on* $\boldsymbol{P}_t$ *which yields offsprings* $\boldsymbol{Q}_t$ *with size* N_p
3. $\boldsymbol{R}_t = \boldsymbol{P}_t \cup \boldsymbol{Q}_t$
4. *Execute Selection operator (Algorithm 7) on* $\boldsymbol{R}_t$ *whiche yields next population* $\boldsymbol{P}_{t+1}, t = t + 1$
5. *If stopping criterion satisfied goto Step 6, else goto Step 2*
6. *Output* $\boldsymbol{P}_t$ ♯

3.2 Crossover Operator

OMOEA-II uses two kind of multi-parents crossover operators. Parents are randomly selected from current population. Denote parents by $\mathbf{m}_1, \mathbf{m}_2$ and $\mathbf{m}_j = (m_{j,1}, m_{j,2}, ..., m_{j,N})$, $j = 1, 2$.

1 Orthogonal Crossover

Orthogonal design method is used on the subspace extended by the randomly chosen parents.

$$\begin{aligned} &\mathbf{H} = \{(x_1, x_2, ..., x_N) | l'_i \le x_i \le u'_i, i = 1, 2, ..., N\} \\ &l'_i = min\{m_{1,i}, m_{2,i}\} \quad u'_i = max\{m_{1,i}, m_{2,i}\} \end{aligned} \tag{7}$$

For $(x_1, x_2, ..., x_N)$ in $\mathbf{H}$, x_i is regarded as the ith factor. Orthogonal array is selected by Equation (4). Each factor i is parted into $Q-1$ equal portions and yields Q levels $x_{1,i}, x_{2,i}, ..., x_{Q,i}$, where the design parameter Q must be prime and $x_{q,i}$ is given by

$$\begin{aligned} &x_{q,i} = \begin{cases} l'_i & q = 1 \\ l'_i + (q-1)\delta_i & 2 \le q \le Q-1 \\ u'_i & q = Q \end{cases} \\ &where\ \delta_i = \frac{u'_i - l'_i}{Q-1} \end{aligned} \tag{8}$$

In other words, the difference between two successive levels is the same. For convenience, denote $x_j = \{x_{1,j}, x_{2,j}, ..., x_{Q,j}\}$, and call $x_{q,j}$ the qth level of the jth factor.

Algorithm 4 *Orthogonal crossover operator*

1. *Randomly choose parents* $\boldsymbol{m}_1$ *and* $\boldsymbol{m}_2$*, Construct subspace* $\boldsymbol{H}$ *according to Equation (7)*
2. *Choose an objective* k *from the* K *objectives as optimizing objective*
3. *Employ orthogonal design method on* $\boldsymbol{H}$*, where orthogonal array is determined by Equation (4)*
4. $\boldsymbol{s}_1$ *is the combination with best objective value among the* $M = Q^J$ *combinations in the orthogonal experiment.* $\boldsymbol{s}_2$ *is the potentially good solutions from Equation (6)*
5. *Output* $\boldsymbol{s}_1$, $\boldsymbol{s}_2$ *as offsprings* ♯

2 Linear Crossover

Algorithm 5 *Linear crossover operator*

1. *Randomly choose parents* $\boldsymbol{m}_1$ *and* $\boldsymbol{m}_2$
2. *For each decision variable* x_i, $i = 1, 2, ..., N$, *choose one of the two sub-steps to execute with probability* p_c *for Sub-step (a) and* $1 - p_c$ *for Sub-step (b)*
   (a) *Randomly create* a_1 *and* a_2 *with* $a_1 + a_2 = 1$ *in the range* $-0.5 \le a_1, a_2 \le 1.5$. *Let* $s_i = a_1 m_{1,i} + a_1 m_{2,i}$

(b) *Randomly create* a_1 *and* a_2 *in the range* $-0.5 \leq a_1, a_2 \leq 1.5$. *Let* $s_i = a_1(m_{1,i} - z_i) + a_2(m_{2,i} - z_i) + z_i$, *where* $\boldsymbol{z} = (z_1, z_2, ..., z_N)$ *is the center of the decision space (See Equation (1))*

3. *Output* $\boldsymbol{s} = (s_1, s_2, ..., s_N)$ *as offspring* ♯

Note that sub-step (a) is helpful for the population to converge faster to good solutions while sub-step (b) helps to diverge the population.

Algorithm 6 *Crossover operator*

1. *Empty* $\boldsymbol{Q}_t$.
2. *choose one of the two sub-steps to execute with probability* p_o *for Sub-step (a) and* $1 - p_o$ *for Sub-step (b)*
   (a) *Execute orthogonal crossover (Algorithm 4) which yields offsprings* $\boldsymbol{s}_1, \boldsymbol{s}_2$, $\boldsymbol{Q}_t = \boldsymbol{Q}_t \cup \{\boldsymbol{s}_1, \boldsymbol{s}_2\}$.
   (b) *Execute linear crossover (Algorithm 5) which yields offsprings* $\boldsymbol{s}$, $\boldsymbol{Q}_t = \boldsymbol{Q}_t \cup \{\boldsymbol{s}\}$.
3. *If the size of* $\boldsymbol{Q}_t$, *i.e.,* $|\boldsymbol{Q}_t|$, *is smaller than* N_p, *then goto Step 2, else goto Step 4.*
4. *Output* $\boldsymbol{Q}_t$. ♯

3.3 Selection Operator

Deb etc employed non-dominated sort technology in NSGA [6]. In OMOEA-II, non-dominated sort is executed by the selection operator on $\mathbf{R}_t$ where $\mathbf{R}_t = \mathbf{P}_t \cup \mathbf{Q}_t$. The details of the operator are as follows

Algorithm 7 *Selection operator*

1. *Empty* $\boldsymbol{P}_{t+1}$.
2. *Find the non-dominated set* $\boldsymbol{B}$ *of* $\boldsymbol{R}_t$. *If* $|\boldsymbol{B}| = N_p$ *then* $\boldsymbol{P}_{t+1} \Leftarrow \boldsymbol{B}$; *if* $|\boldsymbol{B}| > N_p$ *then execute cutoff operator (Algorithm 8) which eliminate* $|\boldsymbol{B}| - N_p$ *elements from* $\boldsymbol{B}$ *and assigned the reduced* $\boldsymbol{B}$ *to* $\boldsymbol{P}_{t+1}$; *if* $|\boldsymbol{B}| < N_p$ *then move* $\boldsymbol{B}$ *from* $\boldsymbol{R}_t$ *to* $\boldsymbol{P}_{t+1}$, *i.e.,* $\boldsymbol{R}_t \Leftarrow \boldsymbol{R}_t \setminus \boldsymbol{B}$ *and* $\boldsymbol{P}_{t+1} \Leftarrow \boldsymbol{P}_{t+1} \cup \boldsymbol{B}$, *repeat the process of finding the non-dominated set of reduced* $\boldsymbol{R}_t$ *and moving the non-dominated set from* $\boldsymbol{R}_t$ *to* $\boldsymbol{P}_{t+1}$ *till* $|\boldsymbol{P}_{t+1}| = N_p$.
3. *Output* $\boldsymbol{P}_{t+1}$. ♯

The goal of the cutoff operator on $\mathbf{B}$ is to enhance distribution uniformity of the reduced $\mathbf{B}$ after the cutoff. Zitzler etc. used cluster analysis to serve this goal in SPEA [7]. The cluster analysis is also employed in this paper. The details of the cutoff operator are as follows.

Algorithm 8 *Cutoff operator*

1. *Initialize cluster set* Ψ: $\Psi = \cup_{i \in \boldsymbol{B}} \{\{i\}\}$ *where each individual* $i \in \boldsymbol{B}$ *constitute a distinct cluster.*
2. *If* $|\Psi| \leq N_p$, *go to Step 5, else go to Step 3.*

3. *Calculate the distance of all possible pairs of clusters. The distance d_c of two cluster $C_1, C_2 \in \Psi$ is given as the average distance between pairs of individuals across the two clusters*

$$d_c = \frac{1}{|C_1|.|C_2|} \sum_{i_1 \in C_1, i_2 \in C_2} d(i_1, i_2)$$

where $d(i_1, i_2)$ is the distance between two individuals i_1 and i_2 (here the distance in objective space is used).

4. *Determine two clusters C_1 and C_2 with minimal distance d_c; the chosen clusters are amalgamated into a large cluster: $\Psi = \Psi \setminus \{C_1, C_2\} \cup \{C_1 \cup C_2\}$. Go to Step 2*
5. *For each cluster, select a representative individual and remove all other individuals from the cluster. We consider the centroid (the point with minimal average distance to all other points in the cluster) as the representative individual. Compute the reduced non-dominated set by uniting the representative of the clusters: $\boldsymbol{P}_{t+1} = \cup_{C \in \Psi} C$.* ♯

4 Numerical Experiments and Discussion

4.1 Test functions

Some benchmark problems are taken to test OMOEA-II. They are: FON [15], ZDT_1, ZDT_2, ZDT_3, ZDT_4, ZDT_6 [16], $DTLZ_1$, $DTLZ_3$, $DTLZ_4$, $DTLZ_6$ [17]. The number of decision variables N: $N_{FON} = 3$, $N_{ZDT_1} = N_{ZDT_2} = N_{ZDT_3} = 30$, $N_{ZDT_4} = N_{ZDT_6} = 10$, $N_{DTLZ_1} = N_{DTLZ_3} = N_{DTLZ_4} = N_{DTLZ_6} = 12$. FON , ZDT_1, ZDT_2, ZDT_3, ZDT_4, ZDT_6 have 2 objectives and $DTLZ_1$, $DTLZ_3$, $DTLZ_4$, $DTLZ_6$ have 3 ones. ZDT_1 has a convex Pareto-optimal front while ZDT_2 has a nonconvex one. ZDT_3 represents the discreteness feature, its Pareto-optimal front consists of several noncontiguous convex parts. ZDT_4 contains 21^9 local Pareto-optimal fronts and $DTLZ_3$ contains 3^9 local ones. ZDT_6 has non-uniformity of objective space. Some decision variables of $DTLZ_1$, $DTLZ_3$, $DTLZ_4$ and $DTLZ_6$ are of linkage.

The parameter settings for OMOEA-II are as follows: function evaluations 25000 (about generations 250), population size 100, orthogonal crossover probability $p_o = 0.1$, linear crossover probability $1 - p_o = 0.9$ where convergence operator probability $p_c = 0.8$ and diversity operator probability $1 - p_c = 0.2$. 10 runs are repeated for each problem. These parameters were chosen after initial experiments. They are not meant to be optimal.

4.2 Results and discussion

Metric Υ [6] was to measure the extent of convergence of known set of Pareto-optimal set. Metric Δ [6] measured the extent of spread achieved among the obtained solutions, Δ is only able to measure problems with two objectives. The

smaller both the metric Υ and Δ, the better the obtained solutions. For comparison, the metric Υ and Δ of the obtained solutions on the test problems with two objectives by SPEA, NSGA-II and OMOEA-II are shown in Tables 4 and 5, where the data for NSGA-II and SPEA were from literature [6] and both of them also had population size 100 and evolution generations 250. For problems FON, ZDT_1, ZDT_2, ZDT_3, ZDT_4, ZDT_6 with two objectives, OMOEA-II found more accurate solutions than those of SPEA and NSGA-II except the results FON. Meanwhile, OMOEA-II could keep rather smaller or comparable Δ for diversity. For the test problems with three objectives, only the simulation results of OMOEA-II are shown; the metric Υ are shown in Table 4 while the spread of the solutions are illustrated on Figure 1. As what Deb in [17] described that both NSGA-II and SPEA2 could not quit converge to true Pareto optimal on $DTLZ_3$ and $DTLZ_6$ where they evolved 500 generations, for $DTLZ_4$, they could not yield well distributed set of solutions, while OMOEA-II found solutions with both good spread and precision on these four problems (See Table 6, Figures 1)without changing parameter settings.

Table 4. Mean (first rows) and variance of the convergence metric Υ in 10 runs.

Algorithm	FON	ZDT_1	ZDT_2	ZDT_3	ZDT_4	ZDT_6
OMOEA-II	0.0032	0.00072	0.00057	0.0021	0.0032	0.000032
	0	0	0	0	0	0
NSGA-II	0.0019	0.03348	0.0724	0.1145	0.513	0.2965
	0	0.00475	0.0317	0.00794	0.118	0.0131
SPEA	0.1256	0.0018	0.00134	0.04751	7.34	0.0011
	0.00003	0.000001	0	0.00005	6.5725	0.00045

Table 5. Mean (first rows) and variance of the diversity metric Δ in 10 runs.

Algorithm	FON	ZDT_1	ZDT_2	ZDT_3	ZDT_4	ZDT_6
OMOEA-II	0.235	0.439	0.55	0.8159	0.250	0.235
	0.000283	0.0088	0.0088	0.0099	0.00044	0.0003
NSGA-II	0.378	0.39	0.43	0.7385	0.702	0.668
	0.000639	0.0019	0.0047	0.0197	0.0646	0.00992
SPEA	0.792	0.7845	0.7551	0.6729	0.7984	0.8493
	0.0055	0.0044	0.0045	0.0036	0.0146	0.0027

The population size of OMOEA in [13] is variable, and, for many problems, increases exponentially as the number of objectives increase. For problems FON, ZDT_1, ZDT_2, ZDT_3, ZDT_4, ZDT_6 with two objectives, the precision and distribution of solutions of OMOEA-II are comparable with OMOEA. But, for

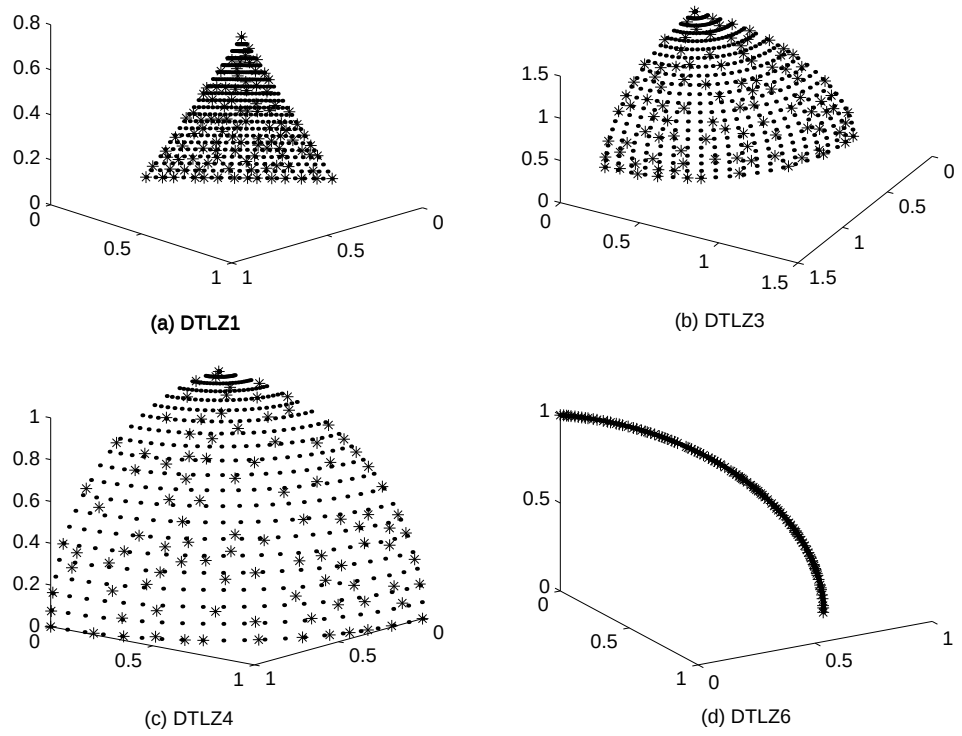

Fig. 1. Distribution of obtained solutions for $DTLZ1$, $DTLZ3$, $DTLZ4$, $DTLZ6$, where $'\cdot'$ denote true Pareto-optimal fronts and $'*'$ close-to-Pareto-optimal fronts.

$DTLZ_1$, $DTLZ_3$, $DTLZ_4$ and $DTLZ_6$, OMOEA yields no results since the population is too large, that results from the linkage variables. Since the structure of OMOEA is not much the same as common MOEAs/MOGAs, usual metric comparison is much convincible. Therefore, comparison between OMOEA-II and OMOEA was not listed.

5 Conclusion

OMOEA-II overcomes the difficulties existed in its previous version (OMOEA). Orthogonal design method is nested in crossover operator to select better genes as offsprings, and consequently, enhances result precision. The other crossover called linear crossover is used, in order to converge fast to better solutions and diverge the population for global optima by changing parameters. Experimental results show that OMOEA-II can solve problems with high dimensions,

Table 6. Mean (first rows) and variance of the convergence metric Υ

Algorithm	$DTLZ_1$	$DTLZ_3$	$DTLZ_4$	$DTLZ_6$
OMOEA-II	0.0043	0.0127	0.01	0.00078
	0	0.000001	0	0

large number of local Pareto-optimal fronts, non-uniformity or discontinuity or convex or nonconvex of global Pareto-optimal front and yield better solutions than some reported results. **Acknowledgment** This work was supported by The National Natural Science Foundation of China (No.s: 60473037, 60483081, 40275034, 60204001, 60133010) and by China Postdoctoral Science Foundation (No. 2003034505).

References

1. Schaffer, J. D. (1984).*Multiple Objective Optimization with Vector Evaluated Genetic Algorithms.* Ph. D. thesis, Vanderbilt University. Unpublished.
2. Hajela, P. and Lin, C. Y. (1992). Genetic search strategies in multicriterion optimal design. Structural Optimization 4, 99C107.
3. Fonseca, C. M. and Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms,* San Mateo, California, pp. 416C423. Morgan Kaufmann.
4. Horn, J. and Nafpliotis, N. (1993). Multiobjective optimization using the niched pareto genetic algorithm. IlliGAL Report 93005, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana, Champaign.
5. Knowles, J.D., Corne, D.W.(2000): Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evolutionary Computation 8(2), 149C172.
6. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan (2002). A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2):182-197.
7. Eckart Zitzler, Marco Laumanns, and Lothar Thiele(2002). SPEA2: Improving the Strength. Pareto Evolutionary Algorithm for Multiobjective Optimization. In K.C. Giannakoglou et al., editor, Proceedings of the EUROGEN2001 Conference, pages 95-100, Barcelona, Spain, CIMNE
8. Purshouse, R. C., Fleming, P. J. (2001). The Multi-objective Genetic Algorithm Applied to Benchmark Problems–an Analysis. Research Report No. 796. Department of Automatic Control and Systems Engineering University of Sheffield, Sheffield, S1 3JD, UK.
9. Tiwari, A., Roy, R.(2002). Generalised Regression GA for Handling Inseparable Function Interaction: Algorithm and Applications. Proceedings of the seventh international conference on parallel problem solving from nature. (PPSN VII). Granada, Spain
10. Wu, Q.(1978). On the optimality of orthogonal experimental design. Acta Math. Appl Sinica, 1(4), pp. 283-299.
11. Leung, Y. W. and Zhang, Q. (1997). Evolutionary algorithms + experimenta design methods: A hybrid apprpach for hard optimization and search problems, Res. Grant Proposal, Hong Kong Baptist Univ.

12. Jinn-Tsong Tsai, Tung-Kuan Liu, and Jyh-Horng Chou (2004). Hybrid Taguchi-Genetic Algorithm for Global Numerical Optimization, IEEE Transactions on Evolutionary Computation, 8(4),pp365-377.
13. Sanyou Y.Zeng, Lishan S.Kang, Lixing X.Ding (2004). An Orthogonal Multi-objective Evolutionary Algorithm for Multi-objective Optimization Problems with Constraints. Evolutionary Computation, 12(1), pp77-98.
14. Montgomery, D. C.(1991). Design and Analysis of Experiments. 3rd ed. New York: Wiley.
15. Carlos M. Fonseca and Peter J. Fleming(1998). Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms–Part I: A Unified Formulation. IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, 28(1):26-37.
16. Zitzler, E., Deb, K. and Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: empirical results. Evolutionary Computation 8(2): 173-195.
17. Deb. K, Lothar Thiele, Marco Laumanns and Eckart Zitzler (2001). Scalable Test Problems for Evolutionary Multi-Objective Optimization. KanGAL Report Number 2001001.