A New Evolutionary Approach to Multiparty Multiobjective Optimization

Zeneng She[1], Wenjian Luo[1](✉), Yatong Chang[1], Xin Lin[2], and Ying Tan[3]

[1] School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, Guangdong, China
{20s151103,20s151150}@stu.hit.edu.cn, luowenjian@hit.edu.cn

[2] School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, Anhui, China
iskcal@mail.ustc.edu.cn

[3] Key Laboratory of Machine Perception (MOE), and Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China
ytan@pku.edu.cn

Abstract. Multiparty multiobjective optimization problems (MPMOPs) are a type of multiobjective optimization problems (MOPs), where multiple decision makers are involved, different decision makers have different objectives to optimize, and at least one decision maker has more than one objective. Although evolutionary multiobjective optimization has been studied for many years in the evolutionary computation field, evolutionary multiparty multiobjective optimization has not been paid much attention. To address the MPMOPs, the algorithm based on a multiobjective evolutionary algorithm is proposed in this paper, where the non-dominated levels from multiple parties are regarded as multiple objectives to sort the candidates in the population. Experiments on the benchmark that have common Pareto optimal solutions are conducted in this paper, and experimental results demonstrate that the proposed algorithm has a competitive performance.

Keywords: Multiobjective optimization · Evolutionary computation · Multiparty multiobjective optimization

1 Introduction

In the real world, there are a lot of optimization problems which have more than one objective, and these objectives are conflicted with each other. This type of optimization problems is called multiobjective optimization problems (MOPs)

This study is supported by the National Key Research and Development Program of China (No. 2020YFB2104003) and the National Natural Science Foundation of China (No. 61573327).

[1,4,11] for two or three objectives and many-objective optimization problems (MaOPs) [6] for more than three objectives.

Multiobjective evolutionary algorithms (MOEAs) have been studied for many years, such as NSGA-II [2], MOEA/D [16], SPEA2 [20] and Two_Arch2 [14]. MOEAs could find a set of optimal solutions in a run, which attracts many researchers to design efficient evolutionary algorithms for solving MOPs [12,13,19].

However, there are cases that there are multiple decision makers (DMs) and each decision maker only pays attention to some of all the objectives of MOPs. If MOEAs only search for the optimal solutions on some certain objectives for a DM, it may lead to deterioration of other objectives concerned by other DMs. If existing MOEAs are directly used to solve the MOP including all the objectives from all parties, they may result in too many Pareto optimal solutions for the MOP, but not Pareto optimal solutions for each DM's objectives.

Multiparty multiobjective optimization problems (MPMOPs) are used to express the above situation. An MPMOP often has multiple parties, and at least one party has more than one objective. MPMOPs are viewed as a subfield of multiobjective optimization problems in the viewpoints of different DMs, respectively. Although MPMOPs are regarded as multiple MOPs, they are quite different in most circumstances so that MPMOPs cannot be directly solved by the original MOEAs. That is because each DM does not consider all objectives in MPMOPs, but a few objectives which he/she cares.

A common method to solve MPMOPs is that the final optimal solutions are obtained by the complex negotiation of a third party among the Pareto optimal solutions from each party [7,8,15]. Recently, in [9], Liu *et al.* first proposed a method without negotiations to address a special class of MPMOPs with the common Pareto optimal solutions, called OptMPNDS. MPMOPs with the common Pareto optimal solutions means that there exits at least one solution that is Pareto optimal for all parties. In other words, the intersection of Pareto optimal solutions of MOPs in viewpoint of multiple DMs is not empty. OptMPNDS defines the dominance relation of solutions based on the corresponding Pareto optimal levels of multiple parties, where the levels are obtained according to the non-dominated sorting in NSGA-II [2]. In each generation, after sorting individuals by objectives preferred by each party, the common solutions with the same level of all parties are assigned to the same rank. Then, the rank of the rest individuals are set to the maximum level obtained by all parties. However, it cannot perfectly handle the situation that the individuals have the same maximum level with different other levels.

In this paper, we propose an improved evolutionary algorithm to solve the MPMOPs, called OptMPNDS2. Similar to OptMPNDS, OptMPNDS2 is also based on NSGA-II. However, OptMPNDS2 overcomes the shortcoming mentioned above. Experiments on the benchmark in [9] are conducted in this paper. From the experimental results, it can be seen that OptMPNDS2 has a more powerful performance on some MPMOPs.

The rest of this paper is organized as follows. Section 2 gives the related work. Section 3 clearly explains the proposed algorithm and Sect. 4 describes the performance metrics and shows the experiment results. Finally, we make a brief conclusion in Sect. 5.

2 Related Work

2.1 Multiparty Multiobjective Optimization Problems (MPMOPs)

An MPMOP is a particular class of multiobjective optimization problems (MOPs). MOPs are a type of optimization problems which have at least two objectives. Because MPMOPs are based on MOPs, the related concepts of MOPs are described first. Here, for convenience, an MOP is defined as a minimization problem [5]:

$$\begin{aligned} &Minimize \quad F(x) = (f_1(x), f_2(x), \dots, f_m(x)), \\ &Subject \quad to \begin{cases} h_i(x) = 0, & i = 1, \dots, n_p \\ g_j(x) \leq 0, & j = 1, \dots, n_q \\ x \in [x_{min}, x_{max}]^d \end{cases}, \end{aligned} \tag{1}$$

where f_i denotes the i-th objective function and F, combined with m objectives, denotes the vector function of objectives which should be minimized. And $h_i(x) = 0$ represents the i-th equality constraint, of which total number is n_p; $g_j(x) \leq 0$ represents the j-th inequality constraint, of which total number is n_q. x, a d-dimensional vector, stands for the decision variables that have the lower bounds x_{min} and upper bounds x_{max} in each dimension.

Dominance is a relation about decision vectors [19]. Given two decision vectors x and y, under the condition that all objectives satisfy $f(x) \leq f(y)$, if there exists one objective f_i satisfying $f_i(x) < f_i(y)$, it is said that x Pareto dominates y, denoting as $x \prec y$ [10]. Pareto optimal set (PS) is a set of Pareto optimal solutions. A decision vector x belongs to PS if and only if no solution dominates x. Pareto optimal front (PF) is a set of objectives of decision vectors in Pareto optimal set, which is formally defined as PF $= \{f = (f_1(x), f_2(x), \cdots, f_n(x)) | x \in$ PS$\}$.

There are multiple DMs in an MPMOP, where each DM focuses on different objectives and at least one DM has at least two objectives. Different from Formula (1), where $f_i(x)$ denotes one objective of the solution x, MPMOPs consider that $f_i(x)$ is a vector function, which represents all objectives of one party. Specifically,

$$f_i(x) = (f_{i1}(x), f_{i2}(x), \dots, f_{ij_i}(x)),$$

and j_i denotes the number of objectives of the i-th party. Then m becomes the number of the parties.

As shown in [9], during evolution, to compare two individuals in MPMOPs, the comparison of all objectives together is not well performed. This is because one DM only focusses on the objectives concerned by himself. Therefore, the individuals are assigned the max levels among the Pareto optimal levels obtained by all parties to compare with each other in [9].

2.2 NSGA-II

Non-dominated sorting genetic algorithm II (NSGA-II) [2] is a popular evolutionary algorithm to solve MOPs. There are two core strategies in NSGA-II, i.e., the fast non-dominated sorting and the calculation of crowding distance. The fast non-dominated sorting adopts the Pareto dominance relation to rapidly achieve the Pareto levels of all the individuals in the evolutionary population. The crowding distance describes the distribution of the individuals. If the crowding distance is large, it means that the individual locates at a region with a few individuals; while it is small, the individual is in a dense region.

In each generation of NSGA-II, the process is shown as follows. First, the crossover and mutation operators are adopted to generate offspring. Next, the parent and offspring are put together to sort the Pareto levels. Then, the crowding distances of the individuals in the same level are obtained. Finally, based on the Pareto levels and crowding distances, the population of next generation is selected from the parent population and the offspring population.

3 The Proposed Algorithm

In this section, the algorithm based on NSGA-II is proposed to solve MPMOPs. Different from traditional MOPs, in MPMOPs, we should maximize the profits of each party. That is, we should try to approach to the Pareto front of each party. In order to achieve this goal, we should optimize the objectives from each party simultaneously, but group them according to each party.

The core issue is how to evaluate a given individual. For a DM, the Pareto optimal level number L_i of the party i could be regarded as an objective value for each individual. Therefore, for total m parties, we have the following multi-objective problem:

$$Minimize \quad L = (L_1, L_2, \ldots, L_m), \tag{2}$$

where m represents the number of the parties, L_i of the party i could be obtained by non-dominated sorting function in NSGA-II. It should be noted that, when calculating L_i of the party i, only the objectives of the party i are considered. Based on Formula (2), we can sort the individuals in evolutionary population according to the standard Pareto dominance, and then the corresponding algorithm could be designed.

The pseudocodes OptMPNDS2 are described in Algorithm 1, which are described as follows.

(1) The population P_0 is initialized with population size N.
(2) The number of generation t and offspring Q_t are set as 0 and $\emptyset$, respectively.
(3) From steps 3 to 15, in the loop, the population P_t and its offspring Q_t are gathered into R_t and function MPNDS2 is applied to sort these individuals into different ranks $\mathcal{F}$. Then, sort individuals in the same rank according to crowding distance. Next the parameters t and P_t are updated for next generation. Subsequently, N individuals are picked from the best to the worst and stored in the population of the next generation P_t. Finally, the offspring Q_t is generated from P_t by both crossover and mutation operators, and the loop is repeated until the termination condition is satisfied.
(4) Return the final solutions MPS.

Algorithm 1. OptMPNDS2

```
Require: $N$ (the population size),
    $F = (F_1(x), F_2(x), \ldots, F_m(x))$ (the objective function)
Ensure: $MPS$ (the multiparty Pareto optimal solutions)
 1: Initialize population $P_0$ with size $N$ ;
 2: $t = 0, Q_t = \emptyset$ ;
 3: while The termination is not satisfied do
 4:     $R_t = P_t \cup Q_t$ ;
 5:     $\mathcal{F} = \text{MPNDS2}(N, R_t, F)$ ;
 6:     Sort $\mathcal{F}$ by crowding distance for each rank ;
 7:     $t = t + 1, P_t = \emptyset, i = 1$ ;
 8:     while $|P_t \cup \mathcal{F}_i| \leq N$ do
 9:         $P_t = P_t \cup \mathcal{F}_i$ ;
10:         $i = i + 1$ ;
11:     end while
12:     $P_t = P_t \cup \mathcal{F}_i(1 : N - |P_t|)$ ;
13:     Create the offspring $Q_t$ of $P_t$ ;
14: end while
15: $MPS =$ multiparty Pareto optimal solutions in $P_t$ ;
```

OptMPNDS2 is similar to NSGA-II except the function MPNDS2(.), and the pseudocodes of MPNDS2(.) are given in Algorithm 2.

As depicted in Algorithm 2, the key point of OptMPNDS2 is to redefine a dominance relation among individuals in the objective space $L = (L_1, L_2, \ldots, L_m)$. From the perspective of a party, the individuals are sorted by F_i (i.e., the objective function of the party i). For individuals in R_t, the Pareto level of the party i is calculated as the new "objective" of the party and stored in $\mathcal{L}(:, i)$. After all m parties are sorted, perform the non-dominated sorting with the objectives $\mathcal{L}$ again. Here, the non-dominated sorting function *NonDominatedSorting* adopted in our algorithm is the same as that in NSGA-II [18].

Algorithm 2. MPNDS2

```
Require: $N, R_t, F = (F_1(x), F_2(x), \ldots, F_m(x))$
Ensure: $\mathcal{F}$
1: $\mathcal{L} = \emptyset$ ;
2: for $i \in \{1, \cdots, M\}$ do
3:     $\mathcal{L}(:, i) = NonDominatedSorting(R_t, F_i)$ ;
4: end for
5: $\mathcal{F} = NonDominatedSorting(R_t, \mathcal{L})$ ;
```

Although both OptMPNDS2 and OptMPNDS sort individuals in terms of each party first and then sort individuals according to the levels by parties, the detail behaviors of these two algorithms are different. In OptMPNDS, the maximum levels in all parties are used to sort the individuals, while OptMPNDS2 performs the non-dominated sorting again according to the non-dominated level numbers of the individuals.

Here, we use an example to explain the difference between OptMPNDS2 and OptMPNDS. Suppose there are two individuals x and y in a bi-party multiobjective optimization, and their non-dominated level numbers are (1, 3) and (2, 3), respectively.

(1) In OptMPNDS, x and y have the same rank because both the maximum non-dominated levels are 3.
(2) In OptMPNDS2, x dominates y. For the first party, their non-dominated levels are 1 and 2, respectively. For the second party, their non-dominated levels are the same. Therefore, according to Formula (2), x dominates y.

4 Experiments

4.1 Parameter Settings

In [9], 11 MPMOP test problems with the common Pareto optimal solutions are given, and two algorithms, i.e., OptMPNDS and OptAll, are used to solve MPMOPs. OptAll just views the problems as MOPs and performs the NSGA-II to obtain solutions.

To compare with these two algorithms, OptMPNDS2 uses the same parameters as [9]. All three algorithms use the simulated binary crossover (SBX) and polynomial mutation [3], of which distribution indexes are set as 20. The rates of crossover and mutation are set to 1.0 and $1/d$, respectively, where d represents the dimension of the individuals. For each test problem, all algorithms are run 30 times to obtain the average results. In each run, the population size is set to 100 and the maximum fitness evaluations is set to $1000 * d * m$, where d and m represent the dimension of MPMOPs and the number of parties, respectively.

Inverted generational distance (IGD) [17] is an indicator to evaluate the solution quality, which measures both the convergence and uniformity of solutions. To adapt metric for MPMOPs, the work [9] slightly modifies the related concept about the distance between an individual v and a PF (denoted by P) shown as follows:

$$d(v, P) = \min_{s \in S} \sum_{i=1}^{m} \sqrt{(v_{i1} - s_{i1})^2 + \cdots + (v_{ij_i} - s_{ij_i})^2}, \tag{3}$$

where v_{ij}, as well as s_{ij}, means the j-th objective values of the i-th party, respectively. There are m parties and the i-th party has j_i objectives.

Based on Formula (3), IGD is calculated as follows:

$$IGD(P^*, P) = \frac{\sum_{v \in P^*} d(v, P)}{|P^*|}, \tag{4}$$

Where P^* represents the true PF and P represents the PF that algorithms obtained.

For IGD, the smaller value means the better performance of the algorithm, since it measures the distance between the true PF and PF obtained by the algorithm.

Considering the final population returned by an algorithm could contain some dominated solutions, the solution number (SN) [9], which represents the number of non-dominated solutions in the final population, is also used to evaluate the performance of the algorithms. The algorithm with a larger SN has a better performance.

4.2 Results

Tables 1, 2 and 3 show the IGD of the three algorithms, i.e., OptMPNDS2, OptMPNDS, OptAll. The problems from MPMOP1 to MPMOP11 are used in experiments, and the dimensions are set to 10, 30 and 50. There are the mean and standard deviation values for each problem in each row. And the best results of the same problem are labeled in the bold font. The sign "—" means that the algorithm does not obtain any solution in at least one run. And *nbr* denotes the number of problems for which the algorithm obtains the best results.

As tables depicted, OptMPNDS2 performs better than OptAll and is comparable with OptMPNDS. In Table 1, the number of the best IGD results of OptMPNDS2 reaches 4. In Table 2, OptMPNDS2 performs the best on 8 problems of 11. The performance of OptMPNDS2 is better than OptMPNDS in Table 3, where OptMPNDS2 wins 7 problems of 11.

Table 1. Mean and standard deviation of IGD for MPMOPs with 10 dimensions.

Problems	IGD		
	OptMPNDS2	OptMPNDS	OptAll
MPMOP1	**1.6345e-05 ± 6.4522e-06**	2.7894e-05 ± 1.8206e-05	—
MPMOP2	**6.7130e-05 ± 3.2436e-05**	2.7364e-04 ± 1.0637e-03	6.5179e-02 ± 7.5853e-02
MPMOP3	3.1678e-02 ± 1.5827e-02	**2.5420e-02 ± 9.2706e-03**	3.6974e-02 ± 1.1352e-02
MPMOP4	**5.6223e-02 ± 5.2352e-03**	5.7572e-02 ± 7.0402e-03	4.7859e-01 ± 8.3774e-02
MPMOP5	6.6392e-02 ± 1.1805e-02	**6.2492e-02 ± 1.1398e-02**	1.5786e-01 ± 3.3413e-02
MPMOP6	2.0289e-02 ± 3.1342e-03	**1.9709e-02 ± 2.4359e-03**	7.6749e-01 ± 2.0057e-01
MPMOP7	4.9085e-06 ± 4.0617e-06	**4.5667e-06 ± 4.1010e-06**	—
MPMOP8	**2.3038e-05 ± 3.1462e-05**	1.2262e-02 ± 4.9124e-02	7.2167e-01 ± 2.6127e-01
MPMOP9	8.3663e-02 ± 1.1029e-02	**8.2055e-02 ± 1.1601e-02**	6.1508e-01 ± 9.2426e-02
MPMOP10	5.7807e-02 ± 5.9171e-03	**5.7619e-02 ± 6.3695e-03**	3.2451e-01 ± 8.2224e-02
MPMOP11	1.8539e-02 ± 1.1730e-03	**1.8343e-02 ± 9.0481e-04**	1.5923e+00 ± 5.8241e-01
nbr	4	**7**	0

Table 2. Mean and standard deviation of IGD for MPMOPs with 30 dimensions.

Problems	IGD		
	OptMPNDS2	OptMPNDS	OptAll
MPMOP1	**1.5425e-05 ± 5.3205e-06**	1.6166e-05 ± 5.9224e-06	8.2886e-03 ± 2.8462e-03
MPMOP2	3.0351e-02 ± 6.3732e-02	**2.4442e-03 ± 1.3105e-02**	5.6115e-02 ± 5.8392e-02
MPMOP3	**1.9912e-01 ± 1.2983e-01**	2.4422e-01 ± 1.5082e-01	2.1359e-01 ± 6.1893e-02
MPMOP4	**5.2178e-02 ± 9.0699e-03**	5.3974e-02 ± 9.7020e-03	1.4187e+00 ± 7.8597e-01
MPMOP5	**4.1281e-02 ± 5.8203e-03**	4.1739e-02 ± 4.2176e-03	3.8524e-01 ± 8.7829e-02
MPMOP6	1.5643e-02 ± 7.1395e-04	**1.5277e-02 ± 9.1020e-04**	2.1889e+00 ± 1.5142e+00
MPMOP7	5.4601e-06 ± 1.7764e-06	**5.1282e-06 ± 2.9116e-06**	—
MPMOP8	**1.1977e-02 ± 4.9290e-02**	1.7650e-01 ± 1.4797e-01	1.4400e-01 ± 1.1676e-01
MPMOP9	**7.6085e-02 ± 1.3514e-02**	7.8395e-02 ± 9.6008e-03	2.1323e+00 ± 9.0104e-01
MPMOP10	**3.3262e-02 ± 2.8672e-03**	4.3365e+00 ± 2.7402e+00	6.4002e-01 ± 2.2718e-01
MPMOP11	**1.7454e-02 ± 7.3541e-04**	1.7898e-02 ± 7.7425e-04	3.0805e+00 ± 2.6421e+00
nbr	**8**	3	0

In terms of SN, from Tables 4, 5 and 6, it can be observed that OptMPNDS2 performs slightly better than OptMPNDS, and these two both are better than OptAll. All the numbers of the best SN results obtained by OptMPNDS2 for all dimensions reach 9. The numbers of the best SN results obtained by OptMPNDS are 7, 7 and 6, respectively. For OptAll, its *nbr* values for the SN metric is always 0.

In summary, OptMPNDS2 obtains more solutions, and the solutions are closer to the true PF for MPMOPs with higher dimensions.

Table 3. Mean and standard deviation of IGD for MPMOPs with 50 dimensions.

Problems	IGD		
	OptMPNDS2	OptMPNDS	OptAll
MPMOP1	**1.6188e-05 ± 5.3270e-06**	1.8747e-05 ± 6.0160e-06	6.3026e-03 ± 1.1210e-03
MPMOP2	**2.1572e-02 ± 3.3461e-02**	2.7957e-02 ± 4.8711e-02	7.6744e-02 ± 7.2050e-02
MPMOP3	5.3982e-01 ± 1.9816e-01	5.1823e-01 ± 2.0427e-01	**4.2675e-01 ± 1.0779e-01**
MPMOP4	**5.2182e-02 ± 7.6690e-03**	5.4883e-02 ± 9.0317e-03	2.0794e+00 ± 1.8092e+00
MPMOP5	**3.1668e-02 ± 2.4659e-03**	3.2240e-02 ± 2.9652e-03	4.9932e-01 ± 1.5816e-01
MPMOP6	1.4525e-02 ± 9.1710e-04	**1.4232e-02 ± 6.2776e-04**	2.6863e+00 ± 2.5453e+00
MPMOP7	8.4879e-06 ± 3.3024e-06	**7.3476e-06 ± 2.4889e-06**	—
MPMOP8	**9.2591e-02 ± 1.1908e-01**	2.4295e-01 ± 1.3750e-01	2.8748e-01 ± 1.6227e-01
MPMOP9	8.0242e-02 ± 1.3628e-02	**7.3962e-02 ± 1.0786e-02**	3.9340e+00 ± 2.5023e+00
MPMOP10	**2.5628e-02 ± 1.3457e-03**	1.1150e+01 ± 1.2323e+00	8.2081e-01 ± 3.0717e-01
MPMOP11	**1.7872e-02 ± 8.9646e-04**	1.7885e-02 ± 9.2098e-04	5.0912e+00 ± 6.7435e+00
nbr	**7**	3	1

Table 4. Mean and standard deviation of SN for MPMOPs with 10 dimensions.

Problems	SN		
	OptMPNDS2	OptMPNDS	OptAll
MPMOP1	**99.10 ± 2.09**	92.53 ± 11.14	0.03 ± 0.18
MPMOP2	**91.33 ± 6.94**	89.27 ± 7.14	6.30 ± 1.37
MPMOP3	99.87 ± 0.43	**100.00 ± 0.00**	30.20 ± 1.52
MPMOP4	**100.00 ± 0.00**	**100.00 ± 0.00**	19.70 ± 4.25
MPMOP5	99.97 ± 0.18	**100.00 ± 0.00**	29.90 ± 3.74
MPMOP6	**100.00 ± 0.00**	**100.00 ± 0.00**	6.60 ± 1.87
MPMOP7	**99.67 ± 0.80**	99.53 ± 1.25	0.00 ± 0.00
MPMOP8	**97.37 ± 1.81**	85.67 ± 8.04	2.60 ± 1.16
MPMOP9	**100.00 ± 0.00**	**100.00 ± 0.00**	21.80 ± 4.12
MPMOP10	**100.00 ± 0.00**	**100.00 ± 0.00**	10.00 ± 2.03
MPMOP11	**100.00 ± 0.00**	**100.00 ± 0.00**	4.97 ± 2.33
nbr	**9**	7	0

Table 5. Mean and standard deviation of SN for MPMOPs with 30 dimensions.

Problems	SN		
	OptMPNDS2	OptMPNDS	OptAll
MPMOP1	99.33 ± 1.47	**99.47 ± 1.50**	2.00 ± 0.00
MPMOP2	**96.13 ± 4.34**	94.00 ± 5.68	5.47 ± 1.14
MPMOP3	94.57 ± 7.99	**94.87 ± 8.76**	25.67 ± 5.45
MPMOP4	**100.00 ± 0.00**	**100.00 ± 0.00**	14.87 ± 9.87
MPMOP5	**100.00 ± 0.00**	**100.00 ± 0.00**	5.73 ± 1.72
MPMOP6	**100.00 ± 0.00**	**100.00 ± 0.00**	3.30 ± 1.09
MPMOP7	**99.87 ± 0.43**	99.80 ± 0.76	1.10 ± 0.66
MPMOP8	**99.57 ± 0.77**	98.47 ± 3.25	5.20 ± 1.10
MPMOP9	**100.00 ± 0.00**	**100.00 ± 0.00**	12.83 ± 3.87
MPMOP10	**100.00 ± 0.00**	37.30 ± 33.27	2.83 ± 0.79
MPMOP11	**100.00 ± 0.00**	**100.00 ± 0.00**	2.63 ± 1.22
nbr	**9**	7	0

Table 6. Mean and standard deviation of SN for MPMOPs with 50 dimensions.

Problems	SN		
	OptMPNDS2	OptMPNDS	OptAll
MPMOP1	**99.90 ± 0.55**	99.80 ± 0.66	2.00 ± 0.00
MPMOP2	**97.20 ± 3.70**	96.87 ± 4.34	5.17 ± 1.02
MPMOP3	84.27 ± 18.57	**87.70 ± 14.24**	23.53 ± 5.20
MPMOP4	**100.00 ± 0.00**	**100.00 ± 0.00**	19.33 ± 16.69
MPMOP5	**100.00 ± 0.00**	**100.00 ± 0.00**	3.47 ± 1.01
MPMOP6	**100.00 ± 0.00**	**100.00 ± 0.00**	3.43 ± 1.79
MPMOP7	**99.97 ± 0.18**	99.33 ± 1.63	1.33 ± 0.61
MPMOP8	**99.43 ± 1.04**	98.47 ± 3.52	4.27 ± 0.87
MPMOP9	99.97 ± 0.18	**100.00 ± 0.00**	15.23 ± 6.18
MPMOP10	**100.00 ± 0.00**	21.27 ± 7.54	2.13 ± 0.63
MPMOP11	**100.00 ± 0.00**	**100.00 ± 0.00**	3.17 ± 1.32
nbr	**9**	6	0

5 Conclusion

In this paper, we propose an evolutionary algorithm called OptMPNDS2 to solve MPMOPs. A new dominance relation of individuals in MPMOPs is defined to handle the non-dominated sorting, where the Pareto optimal level number of each party is regarded as an objective value of the individual. In the experiments, OptMPNDS2 is compared with OptMPNDS as well as OptAll. Experimental results show that the overall performance of the proposed OptMPNDS2 is better. In the future, we will establish a benchmark of the MPMOPs which have no common Pareto optimal solutions, and the benchmark will be used to evaluate the proposed algorithm.

References

1. Coello, C.C.: Evolutionary multi-objective optimization: a historical view of the field. IEEE Comput. Intell. Mag. **1**(1), 28–36 (2006)
2. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
3. Deb, K., Sindhya, K., Okabe, T.: Self-adaptive simulated binary crossover for real-parameter optimization. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp. 1187–1194 (2007)
4. Fonseca, C.M., Fleming, P.J.: An overview of evolutionary algorithms in multiobjective optimization. Evol. Comput. **3**(1), 1–16 (1995)
5. Geng, H., Zhang, M., Huang, L., Wang, X.: Infeasible elitists and stochastic ranking selection in constrained evolutionary multi-objective optimization. In: Wang, T.-D., et al. (eds.) SEAL 2006. LNCS, vol. 4247, pp. 336–344. Springer, Heidelberg (2006). https://doi.org/10.1007/11903697_43
6. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization: a short review. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp. 2419–2426. IEEE (2008)
7. Lau, R.Y.: Towards genetically optimised multi-agent multi-issue negotiations. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, pp. 35c–35c. IEEE (2005)
8. Lau, R.Y., Tang, M., Wong, O., Milliner, S.W., Chen, Y.P.P.: An evolutionary learning approach for adaptive negotiation agents. Int. J. Intell. Syst. **21**(1), 41–72 (2006)
9. Liu, W., Luo, W., Lin, X., Li, M., Yang, S.: Evolutionary approach to multiparty multiobjective optimization problems with common pareto optimal solutions. In: Proceedings of 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–9. IEEE (2020)
10. Miettinen, K.: Nonlinear Multiobjective Optimization, vol. 12. Springer Science & Business Media, Berlin (2012)
11. Tamaki, H., Kita, H., Kobayashi, S.: Multi-objective optimization by genetic algorithms: a review. In: Proceedings of IEEE International Conference on Evolutionary Computation, pp. 517–522. IEEE (1996)
12. Tian, Y., Cheng, R., Zhang, X., Jin, Y.: PlatEMO: a MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2017)
13. Trivedi, A., Srinivasan, D., Sanyal, K., Ghosh, A.: A survey of multiobjective evolutionary algorithms based on decomposition. IEEE Trans. Evol. Comput. **21**(3), 440–462 (2016)
14. Wang, H., Jiao, L., Yao, X.: Two_Arch2: an improved two-archive algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **19**(4), 524–541 (2014)
15. Zhang, C., Wang, G., Peng, Y., Tang, G., Liang, G.: A negotiation-based multi-objective, multi-party decision-making model for inter-basin water transfer scheme optimization. Water Resour. Manag. **26**(14), 4029–4038 (2012)

16. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
17. Zhang, Q., Zhou, A., Zhao, S., Suganthan, P.N., Liu, W., Tiwari, S.: Multiobjective optimization test instances for the CEC 2009 special session and competition (2008)
18. Zhang, X., Tian, Y., Cheng, R., Jin, Y.: An efficient approach to nondominated sorting for evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. **19**(2), 201–213 (2014)
19. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol. Comput. **1**(1), 32–49 (2011)
20. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm. TIK-report **103** (2001)