# Multi-objective Combinatorial Generative Adversarial Optimization and Its Application in Crowdsensing

Yi-nan Guo[1](✉), Jianjiao Ji[1], Ying Tan[2], and Shi Cheng[3]

[1] School of Information and Control Engineering, China University of Mining and Technology, Xuzhou 221008, Jiangsu, China
guoyinan@cumt.edu.cn

[2] Peking University, Beijing 100091, China

[3] School of Computer Science, Shaanxi Normal University, Xi'an 710119, China

**Abstract.** With the increasing of the decision variables in multi-objective combinatorial optimization problems, the traditional evolutionary algorithms perform worse due to the low efficiency for generating the offspring by a stochastic mechanism. To address the issue, a multi-objective combinatorial generative adversarial optimization method is proposed to make the algorithm capable of learning the implicit information embodied in the evolution process. After classifying the optimal non-dominated solutions in the current generation as real data, the generative adversarial network (GAN) is trained by them, with the purpose of learning their distribution information. The Adam algorithm that employs the adaptively learning rate for each parameter is introduced to update the main parameters of GAN. Following that, an offspring reproduction strategy is designed to form a new feasible solution from the decimal output of the generator. To further verify the rationality of the proposed method, it is applied to solve the participant selection problem of the crowdsensing and the detailed offspring reproduction strategy is given. The experimental results for the crowdsensing systems with various tasks and participants show that the proposed algorithm outperforms the others in both convergence and distribution.

**Keywords:** Multi-objective · Combinatorial optimization · Generative adversarial network · Participant selection · Crowdsensing

## 1 Introduction

Multi-objective combinatorial optimization problems (MOCOPs), such as traveling salesman problem, vehicle routing planning, participant selection problem of crowdsensing and so on, is to find the optimal resource assignment that takes multiple objectives into consideration [1, 2]. To address the issues, the classical linear programming, meta-heuristic search, evolutionary algorithm, and some other population-based intelligent optimization algorithms are introduced. However, with the increasing of the decision variables, the exploration ability of above-mentioned algorithms become limited due to

© Springer Nature Switzerland AG 2020

the low efficiency for generating diverse offspring by a stochastic mechanism. Many researchers [3–7] employed the handcrafted strategies that were especially designed according to the characteristics of specific problems to improve the performance. However, such problem-specific methods depends a lot on the artificial experience. The recent advances [8–11] in machine learning algorithms have shown their strong ability of helping engineers to design optimization algorithms with learning ability to solve different problems with a relatively good performance. Based on this, various studies [12–14] have tried to utilize the neurol networks to learn the useful information about the fitness landscape or the distribution of the individuals to generate the more rational offspring. But a large number of the training data must be provided for building the learning models, which is always difficult or expensive to be achieved in practice.

The Generative Adversarial Networks (GAN) proposed by Goodfellow in 2014 is able to learn high-dimensional distributions efficiently with the limited training data by the following adversarial learning mechanism [15], which consists of a generator and a discriminator [16]. The former learns the distribution $P_{data}(x)$ of a real data $x$ and generates a new sample $G(z)$ with the prior distribution $P_{prior}(x)$, while the latter works hard on identifying whether the sample is real or fake, and outputs a discriminant probability $D(x)$. More specifically, the generator tries to produce the samples as real as possible, with the purpose of decreasing the accuracy of a discriminator. By contrast, the discriminator makes an effort on enhancing its recognition ability. Both of them are trained in a minimax game manner as follows.

$$\min_{G} \max_{D} V(D, G) = E_{x \in P_{data}}[\log D(x)] + E_{z \in P_{prior}}[\log(1 - D(G(z))] \tag{1}$$

Various studies have been done on GAN-based optimization algorithms, in which the offspring is generated along the direction learned from the distribution of the better candidate solutions. Tan et al. [17] proposed a generative adversarial optimization framework for continuous optimization problems with a single objective. A new solution is produced from a candidate with the input noise by the generator, and then the discriminator predicts whether the fitness value of a generated solution is better than that of the original candidate. He et al. [15] first presented a GAN-based multi-objective evolutionary algorithm. The candidates are classified into two datasets that are labeled as real and fake, respectively. After training the generator with the above samples, the offspring is formed by it or the genetic operators with the same probability. Despite the application of GAN on continuous optimization problems, fewer works have been done on the combinatorial optimization problems. Probst [18] employed GAN to build the probability model that approximates the distribution of the candidates in the estimation of distribution algorithm. The offspring are gotten from the probability model, with the purpose of finding the optima for a scalar combinatorial optimization problem. Being different from it, a multi-objective combinatorial generative adversarial optimization algorithm (MOCGAO) is proposed for the MOCOPs in this paper. And the main contributions of this study are summarized as follows:

(1) MOCGAO takes advantage of the learning and generative abilities of GAN to generate superior solutions for the MOCOPs.

(2) A classification strategy is developed to identify the non-dominated solutions found in the evolution as the real samples for training GAN, as they provide the distribution information of the population.
(3) An offspring reproduction strategy is designed to form a new feasible solution from the decimal output of the generator for the MOCOPs.
(4) The proposed method is applied to solve the participant selection problem of the crowdsensing and the detailed offspring reproduction strategy is given.

The rest of the paper is organized as follows. In Sect. 2, the key issues of multi-objective combinatorial generative adversarial optimization are presented in detail. MOCGAO-based participant selection strategy for the crowdsensing is illustrated in Sect. 3. The experimental results are compared and further analyzed in Sect. 4. Finally, Sect. 5 concludes the whole paper and plans the topic to be researched in the future.

## 2 Multi-objective Combinatorial Generative Adversarial Optimization

According to the algorithm steps of MOCGAO shown in Algorithm 1, the initial population $P(0)$ is constructed by the randomly produced individuals and the optima of each objective obtained by the greedy algorithm. The latter ones provide the extremum for training GAN so as to speed up the convergence of the network. After training GAN by the classified individuals, the generator outputs the offspring $Q(t)$. N individuals selected from the combination of $P(t)$ and $Q(t)$ by the non-domination sort and elite selection strategy of NSGA-II, compose of the population in the next generation $P(t + 1)$. Finally, the Pareto-optimal solution is found until the termination condition is satisfied. Apparently, the key issues of the proposed method are to classify the population, train GAN and reproduce the offspring. The detailed selection strategy of NSGA-II refer to Ref. [19].

Algorithm 1: Multi-objective combinatorial generative adversarial optimization algorithm

```
Input: the population size $N$, the number of objectives $M$, the maximum termination iteration $T$
Output: $PS$
$Net$ ←Randomly initialize the GAN;
$P(0)$ ←Randomly initialize $N-M$ individuals;
$P(0)$ ←Add the optimal individual of each objective by the greedy algorithm;
For $t$=1:T do
    $X$ ← Classification($P(t)$);
    ($Q(t)$, $Net$)←Training GAN ($Net$, $X$);
    $P(t+1)$←Selection($P(t) \cup Q(t)$, $N$);
End
Obtain the Pareto-optimal solutions $PS$.
```

## 2.1 The Classification of the Population

Both the real and fake samples provide effective information for training GAN. For image processing [20–22] and text generation [23], the real data distributions of the images and texts are normally obtained in advance. However, it is difficult to know the fitness landscape, even the true Pareto-optimal solutions before the evolution of actual multi-objective optimization problems. To address the issue, He et al. [15] partitioned the population in each generation into two datasets with the same size, and the one with the better convergence and diversity is treated as real samples. However, both the non-dominate individuals and the dominated ones with the even distribution are classified to form the training data. The latter is conducive to form the GAN that can produce the offspring with better diversity, but the distribution of the worse individuals also is learned by GAN, which slows down the convergence speed.

Different from it, only the non-dominated individuals are labeled as real and employed to form the real dataset $x^r$. The rest of the population are classified into the fake dataset $x^f$. Because GAN iteratively learns only the distribution of better solutions, the generated offspring may more approximate to the true Pareto solutions.

$$x_i \in \begin{cases} x^r & x_i \text{ is a non-dominated solution} \\ x^f & \text{otherwise} \end{cases} \tag{2}$$

## 2.2 Training GAN

The generator and discriminator of MOCGAO both consist of the feedforward neural network with a single hidden layer, as shown in Fig. 1. Each node in the hidden layer employs the ReLu activation function, while the nodes in the output layer adopt the sigmoid function.

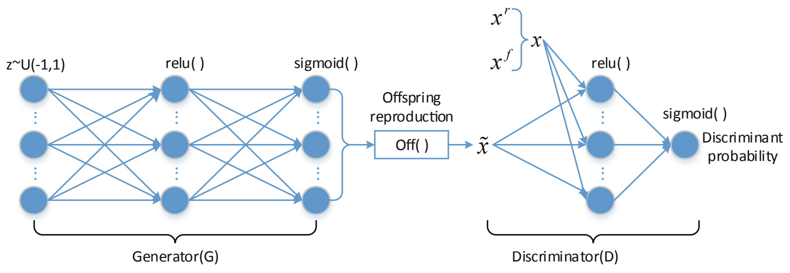


**Fig. 1.** The structure of GAN

The noise obeying the continuous uniform distribution $Z \sim U(-1, 1)$ is employed as the input signal to the generator. Following that, the output of $G(z)$ is transformed into an offspring $\tilde{x}$ by the offspring reproduction strategy $Off\,()$. The generated sample $\tilde{x}$ is utilized to calculate the loss function of the generator as follows.

$$\tilde{V}_{\mathrm{G}} = \frac{1}{N}\sum\nolimits_{i=1}^{N} \log(1 - D(\mathrm{O}\mathit{ff}\,(G(z_i)))) \tag{3}$$

Subsequently, the real samples $x^r$, the fake ones $x^f$ and the generated ones $\tilde{x}$ are all employed to train the discriminator. The comprehensive loss function is defined.

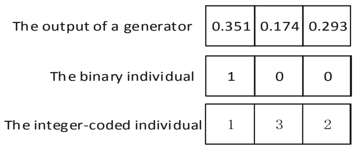
$$\tilde{V}_{\mathrm{D}} = \frac{1}{|x^r|}\sum\nolimits_{i=1}^{|x^r|} \log D(x_i^r) + \frac{1}{|x^f|}\sum\nolimits_{i=1}^{|x^f|} \log(1 - D(x_i^f)) + \frac{1}{N}\sum\nolimits_{i=1}^{N} \log(1 - D(\tilde{x}_i)) \tag{4}$$

The parameters of GAN are learned by the Adam algorithm [24], in which each parameter remains the various learning rate, instead of the traditional gradient descent method.

## 2.3 The Offspring Reproduction Strategy

For most of the combinatorial optimization problems, a solution is normally encoded by the binary or the integer. However, the outputs of the generator in GAN is a decimal between 0 and 1, which cannot represent an individual of a combinatorial optimization problem directly. To this end, a novel offspring reproduction strategy is presented, with the purpose of forming an available individual based on the decimal output of GAN.

The number of neurons in the output layer of a generator is equal to the dimension of decision variables. For a binary-coded individual, the output of each neuron is treated as the probability of each gene being set to 1. In the offspring reproduction strategy, the output of all neurons in the generator are sorted in the descending order, and the variable corresponding to the first unselected one is labeled by 1. Different from it, the possible integer-values of the decision variables are assigned to the genes corresponding to the sorted neurons in the same descending order. That is, the maximum integer-value is first set to the gene corresponding to the neuron having the maximum output. As shown in Fig. 2, the same decimal output of a generator is mapped to the different offspring in terms of the encoding scheme of an individual.



| | | | |
|---|---|---|---|
| The output of a generator | 0.351 | 0.174 | 0.293 |
| The binary individual | 1 | 0 | 0 |
| The integer-coded individual | 1 | 3 | 2 |

**Fig. 2.** An example of an offspring reproduction strategy

## 3 MOCGAO-Based Participant Selection of a Crowdsensing System

To verify the effectiveness of the proposed MOCGAO algorithm, it is applied to the participant selection problem of Crowdsensing, which allocates all the sensing tasks to suitable participants that are selected from the sensing-user set. Assume that the sensing tasks denoted as $T = \{t_1, t_2, \ldots, t_m\}$ are published on the crowdsensing system, and the mobile users expressed by $U = \{u_1, u_2, \ldots, u_n\}$ want to participate the works as executors. Suppose that each task need to be fulfilled by $\xi$ users, and each user is assigned to at most one task. Let $e_{ij}$ and $a_{ij}$ be the sensing ability and reward of the user $u_j$ for the task $t_i$, respectively. Moreover, the sensing ability is determined by the distance between the task and the user, and the reward depends on the sharing mechanism of reward for the user and his cooperative friend. More details refer to Reference [25]. Based on this, a participant selection model that maximize both the sensing quality of tasks and the reward of participants can be constructed as follow:

$$\begin{cases} \max f_1 = \sum_i \sum_j e_{ij} x_{ij} \\ \max f_2 = \sum_i \sum_j a_{ij} x_{ij} \end{cases}$$
$$s.t.\ \xi = \sum_j x_{ij}, \forall i \in [1, m]$$
$$\sum_i x_{ij} \le 1, \forall i \in [1, n] \tag{5}$$

In the above formula, $[x_{ij}]_{m \times n}$ is the Task-User matrix, as shown in Fig. 3(a). $x_{ij} = 1$ means that the task is allocated to the $j$th user. The Task-User matrix is converted to the binary vector, as shown in Fig. 3(b), with the purpose of simplifying the training process of GAN.

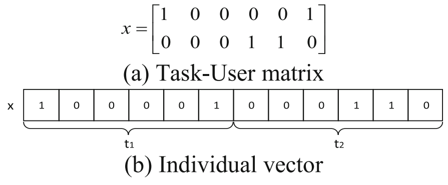

$x = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$
(a) Task-User matrix
x
1 0 0 0 0 1 0 0 0 1 1 0
t1
t2
(b) Individual vector

**Fig. 3.** An example of an individual ($m = 2, n = 6, \xi = 2$)

According to the above-mentioned encoding scheme of the participant selection model, the output of each neuron in the output layer of a generator represents the probability of a user being assigned to a task. In order to obtain the feasible offspring by the proposed offspring reproduction strategy, the number of neurons is set to $m*n$. For each task, $\xi$ unallocated users are selected to complete the task in the descending order of the corresponding probability, and the corresponding gene is set to 1.

As shown in Fig. 4, two tasks are allocated to six users in the crowdsensing system, and each task needs two participants. For the task $t_1$, the two users with the maximum probability, $u_1$ and $u_6$, are selected. The first and sixth genes are set to 1. $u_4$ and $u_5$ are chosen to carry out $t_2$. $u_6$ having a higher probability than $u_5$ is not employed by this task because of the constraint for the available users.

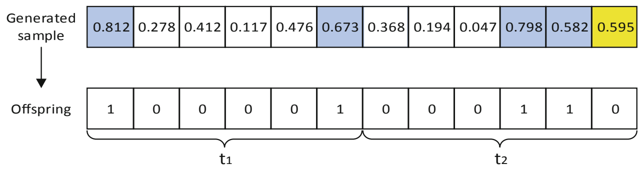


**Fig. 4.** An example of the offspring reproduction ($m = 2, n = 6, \xi = 2$)

## 4 Experimental Results and Discussion

Fully experiments are conducted to examine the performance of the proposed MOCGAO. The main experimental parameters are listed in Table 1, and the other parameters of the participant selection problem refer to paper [25]. Besides, the hypervolume (HV) and coverage(C) metrics [26] are employed to evaluate the performance of the proposed algorithm, and the best results are labeled by bold.

**Table 1.** The main parameters

| Parameter | Value |
|---|---|
| *m* (the number of tasks) | {5, 10, 15, 20} |
| *n* (the number of users) | {100, 200, 300} |
| *N* (the population size) | 100 |
| *T* (the maximum termination iteration) | 100 |
| *iter* (the training iteration) | 3 |
| $\beta_1$ (the preset constant for Adam) | 0.9 |
| $\beta_2$ (the preset constant for Adam) | 0.999 |
| $\varepsilon$ (the preset constant for Adam) | $10^{-8}$ |
| $\eta$ (the initial learning rate) | 0.001 |
| *NodeG* (the number of nodes for each layer of the Generator) | 100, 128, m*n |
| *NodeD* (the number of nodes for each layer of the Discriminator) | m*n, 128, 1 |

To verify the effectiveness of the Adam method, we compare the performances of MOCGAO with the Adam method (Adam) against one with the gradient descent method (Gradient). From the statistical results listed in Table 2, no matter how many users participate to complete the tasks, MOCGAO with the Adam method converges to the best Pareto solutions because the Adam adopts the adaptive learning rate for each parameter instead of the static on in the gradient descent method, which is more efficient for training GAN.

The rationality of the Greedy-based initialization method (Greedy) is analyzed by comparing the performance of MOCGAO with the random initialization strategy (Random), as listed in Table 3. Apparently, the extreme of each objective provides more information on the true distribution of the Pareto front, which is helpful for speeding up the training process of GAN and generating the more promising offspring. Thus, MOCGAO with the Greedy-based initialization method can found better Pareto-optimal solutions, showing the larger HV-values.

Two representative multi-objective evolutionary algorithms, including NSGA-II [19] and MOEA/D [27] are employed as the comparison algorithms. Figure 5 depicts the Pareto fronts obtained by the three compared algorithms. It can be observed that the Pareto-optimal solutions of all instances generated by NSGA-II have the worst convergence, while MOEA/D tends to generate the Pareto solutions distributed in a small

**Table 2.** Comparison of the performances for MOCGAOs with different learning methods

| m*n | HV | | C | |
|---|---|---|---|---|
| | Gradient | Adam | C(Gradient, Adam) | C(Adam, Gradient) |
| 5*100 | 6.79 | **222.76** | 0.00 | **1.00** |
| 5*200 | 23.59 | **202.64** | 0.00 | **1.00** |
| 5*300 | 17.84 | **253.20** | 0.00 | **1.00** |
| 10*100 | 64.10 | **642.33** | 0.00 | **1.00** |
| 10*200 | 105.87 | **462.18** | 0.00 | **1.00** |
| 10*300 | 97.13 | **617.41** | 0.00 | **1.00** |
| 15*100 | 132.17 | **1266.88** | 0.00 | **1.00** |
| 15*200 | 110.66 | **1882.61** | 0.00 | **1.00** |
| 15*300 | 214.91 | **2153.24** | 0.00 | **1.00** |
| 20*100 | 181.11 | **2664.65** | 0.00 | **1.00** |
| 20*200 | 163.32 | **1917.41** | 0.00 | **1.00** |
| 20*300 | 219.70 | **1873.09** | 0.00 | **1.00** |

**Table 3.** Comparison of the performances for MOCGAOs with different initialization strategies

| m*n | HV | | C | |
|---|---|---|---|---|
| | Random | Greedy | C(Random, Greedy) | C(Greedy, Random) |
| 5*100 | 222.76 | **290.47** | 0.00 | **1.00** |
| 5*200 | 202.64 | **460.19** | 0.00 | **1.00** |
| 5*300 | 253.20 | **667.13** | 0.00 | **1.00** |
| 10*100 | 642.33 | **1326.42** | 0.00 | **1.00** |
| 10*200 | 462.18 | **1480.75** | 0.00 | **1.00** |
| 10*300 | 617.41 | **2074.57** | 0.00 | **1.00** |
| 15*100 | 1266.88 | **3304.06** | 0.00 | **1.00** |
| 15*200 | 1882.61 | **3906.53** | 0.00 | **1.00** |
| 15*300 | 2153.24 | **5068.72** | 0.00 | **1.00** |
| 20*100 | 2664.65 | **5187.26** | 0.00 | **1.00** |
| 20*200 | 1917.41 | **6313.68** | 0.00 | **1.00** |
| 20*300 | 1873.09 | **7486.02** | 0.00 | **1.00** |

region, but with the better convergence. In contrast, the Pareto fronts found by the proposed MOCGAO achieve the best performance in both convergence and distribution for 11 out of 12 instances, except for the 10*100 instance. The statistical results of HV and

C summarized in Tables 4 and 5 confirm the above observations. Consequently, the proposed algorithm outperforms the other compared algorithms due to the better diversity of the offspring maintained by the improved GAN, especially for the problems with the large-scale participants.

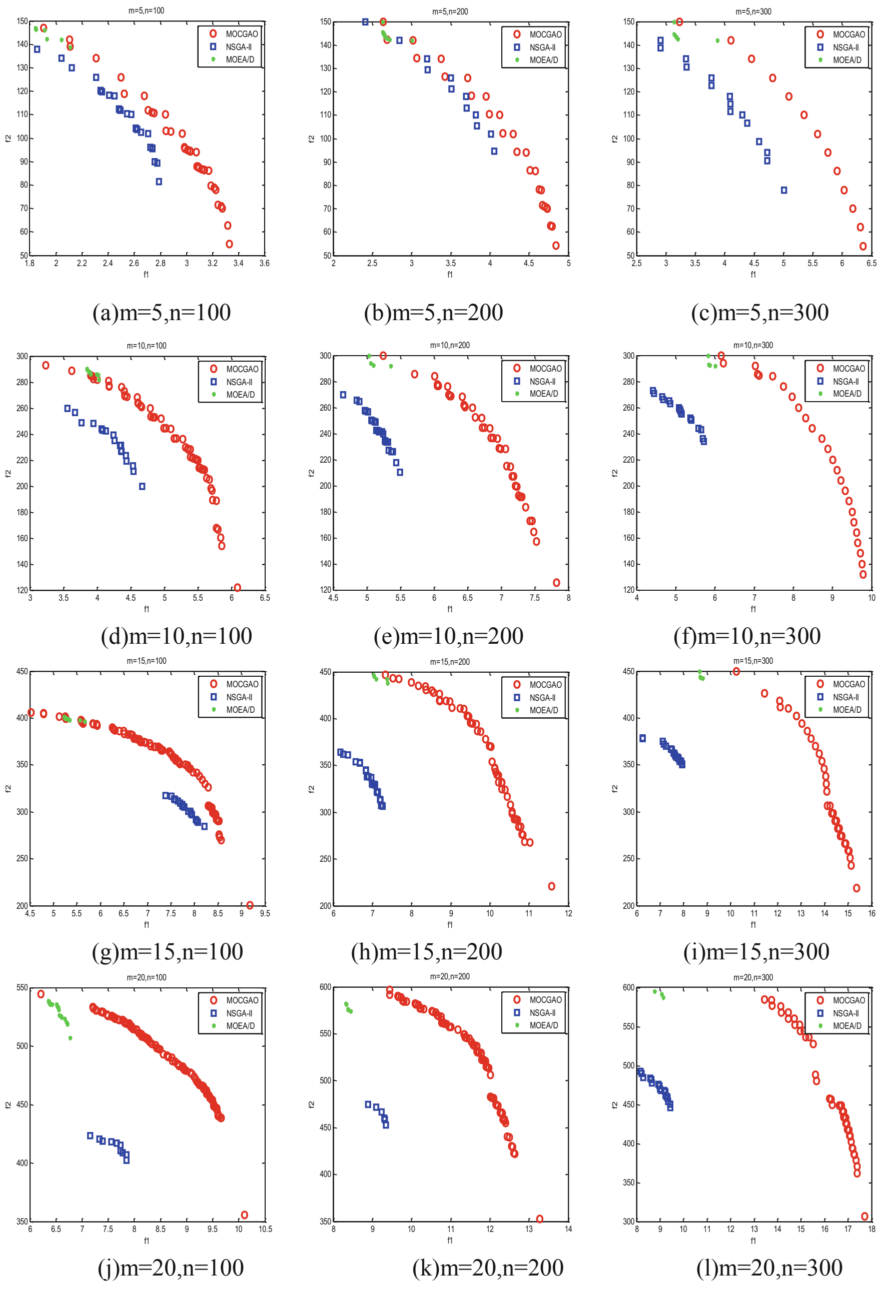


**Fig. 5.** The Pareto fronts found by different algorithms

**Table 4.** Comparison of HV for different algorithms

| Algorithms | 5*100 | 5*200 | 5*300 | 10*100 |
|---|---|---|---|---|
| NSGA-II | 255.91 | 401.69 | 555.19 | 995.15 |
| MOEA/D | 190.92 | 336.97 | 510.12 | 900.75 |
| MOCGAO | **290.47** | **460.19** | **667.13** | **1326.42** |
| Algorithms | 10*200 | 10*300 | 15*100 | 15*200 |
| NSGA-II | 1150.59 | 1436.14 | 2293.47 | 2282.35 |
| MOEA/D | 1361.30 | 1508.12 | 2021.01 | 2951.97 |
| MOCGAO | **1480.75** | **2074.57** | **3304.06** | **3906.53** |
| Algorithms | 15*300 | 20*100 | 20*200 | 20*300 |
| NSGA-II | 3050.39 | 3043.77 | 3981.00 | 4366.54 |
| MOEA/D | 3515.28 | 3144.68 | 4341.35 | 5000.99 |
| MOCGAO | **5068.72** | **5187.26** | **6313.68** | **7486.02** |

**Table 5.** Comparison of C for different algorithms

| m*n | C(NSGA-II, MOCGAO) | C(MOCGAO, NSGA-II) | **C(MOEA/D, MOCGAO)** | C(MOCGAO, MOEA/D) |
|---|---|---|---|---|
| 5*100 | 0.00 | **1.00** | 0.00 | **0.57** |
| 5*200 | 0.00 | **1.00** | 0.04 | **0.38** |
| 5*300 | 0.00 | **1.00** | 0.00 | **1.00** |
| 10*100 | 0.00 | **1.00** | **0.09** | 0.00 |
| 10*200 | 0.00 | **1.00** | 0.00 | **0.75** |
| 10*300 | 0.00 | **1.00** | 0.00 | **1.00** |
| 15*100 | 0.00 | **1.00** | 0.03 | **0.85** |
| 15*200 | 0.00 | **1.00** | 0.00 | **1.00** |
| 15*300 | 0.00 | **1.00** | 0.00 | **1.00** |
| 20*100 | 0.00 | **1.00** | 0.00 | **0.39** |
| 20*200 | 0.00 | **1.00** | 0.00 | **1.00** |
| 20*300 | 0.00 | **1.00** | 0.00 | **1.00** |

## 5 Conclusions

To overcome the weakness of the traditional evolutionary algorithms on solving multi-objective combinatorial optimization problems with the large-scale decision variables, a generative adversarial network that has the strong learning and generative abilities is introduced to construct a multi-objective combinatorial generative adversarial optimization algorithm. The extreme of each objective obtained by the greedy algorithm is combined with the randomly produced individuals to form the initial population, with the purpose of speeding up the training process of GAN. During the evolution, the optimal non-dominated solutions in the current generation are identified as the real samples, while the rest are fake. Classified solutions are employed to train GAN. More specifically, the Adam method with the adaptive learning rate is employed to update the parameters of GAN, and an offspring reproduction strategy is presented to obtain a feasible offspring from the decimal output of the generator. Following that, the proposed algorithm is utilized to solve the participant selection problem of the crowdsensing, and a detailed offspring reproduction strategy is given. The experiments are conducted on the crowdsensing system with the various tasks and participants, and the results show that the proposed algorithm superior to the others in both convergence and distribution. To combine the evolutionary operators with GAN will be our future work, with the purpose of generating the offspring with more uniform distribution.

**Acknowledgements.** This work is supported by National Natural Science Foundation of China under Grant 61973305, 61806119, National Key Research and Development Program under Grant 2016YFC0801406, and Six talent peaks project in Jiangsu Province under Grant No. 2017-DZXX-046.

## References

1. Guo, Y., Cheng, J., Luo, S., et al.: Robust dynamic multi-objective vehicle routing optimization method. IEEE/ACM Trans. Comput. Biol. Bioinform. **15**(6), 1891–1903 (2018)
2. Blum, C., Puchinger, J., Günther, R.R.: Hybrid metaheuristics in combinatorial optimization: a survey. Appl. Soft Comput. **11**(6), 4135–4151 (2011)
3. Zhang, P., Tan, Y.: Immune cooperation mechanism based learning framework. Neurocomputing **148**, 158–166 (2015)
4. Precup, R.E., David, R.C., Petriu, E.M., et al.: Grey wolf optimizer-based approach to the tuning of pi-fuzzy controllers with a reduced process parametric sensitivity. IFAC PapersOnLine **49**(5), 55–60 (2016)
5. Goli, A., Aazami, A., Jabbarzadeh, A.: Accelerated cuckoo optimization algorithm for capacitated vehicle routing problem in competitive conditions. Int. J. Artif. Intell. **16**(1), 88–112 (2018)
6. Yang, M., Omidvar, M.N., Li, C., et al.: Efficient resource allocation in cooperative co-evolution for large-scale global optimization. IEEE Trans. Evol. Comput. **21**(4), 493–505 (2017)
7. Tian, Y., Zhang, X., Wang, C., et al.: An evolutionary algorithm for large-scale sparse multi-objective optimization problems. IEEE Trans. Evol. Comput. (2019). https://doi.org/10.1109/TEVC.2019.2918140
8. Cheng, R., He, C., Jin, Y., Yao, X.: Model-based evolutionary algorithms: a short survey. Complex Intell. Syst. **4**(4), 283–292 (2018). https://doi.org/10.1007/s40747-018-0080-1
9. Guo, Y., Huan, Y., Chen, M., et al.: Ensemble prediction-based dynamic robust multi-objective optimization methods. Swarm Evol. Comput. **48**, 156–171 (2019)
10. Zhang, J., Zhan, Z., Lin, Y., et al.: Evolutionary computation meets machine learning: a survey. IEEE Comput. Intell. Mag. **6**(4), 68–75 (2011)

11. Guo, Y., Zhang, X., Gong, D., et al.: Novel interactive preference-based multi-objective evolutionary optimization for bolt supporting networks. IEEE Trans. Evol. Comput. (2019). https://doi.org/10.1109/TEVC.2019.2951217
12. Li, K., Zhang, T., Wang, R.: Deep reinforcement learning for multi-objective optimization **14**(8), 1–10 (2019)
13. Nazari, M., Oroojlooy, A., Snyder, L., et al.: Reinforcement learning for solving the vehicle routing problem. In: Advances in Neural Information Processing Systems, pp. 9839–9849 (2018)
14. Ruiz-Rangel, J., Ardila, C., Gonzalez, L.M., et al.: ERNEAD: training of artificial neural networks based on a genetic algorithm and finite automata theory. Int. J. Artif. Intell. **16**(1), 214–253 (2018)
15. He, C., Huang, S., Cheng, R.: Evolutionary multi-objective optimization driven by generative adversarial networks. arXiv:1907.04482 [cs.NE] (2019)
16. Goodfellow, I., Pouget-Abadie, J., Mirza, M., et al.: Generative adversarial nets. In: Advances in Neural Information Processing System, pp. 2672–2680 (2014)
17. Tan, Y., Shi, B.: Generative adversarial optimization. In: Tan, Y., Shi, Y., Niu, B. (eds.) ICSI 2019. LNCS, vol. 11655, pp. 3–17. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26369-0_1
18. Probst, M.: Generative adversarial networks in estimation of distribution algorithms for combinatorial optimization. arXiv:1509.09235 [cs.NE] (2015)
19. Deb, K., Pratap, A., Agarwal, S.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 180–197 (2002)
20. Ledig, C., Theis, L., Huszár, F.: Photo-realistic single image super resolution using a generative adversarial network. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4681–4690 (2016)
21. Zhang, D., Shao, J., Hu, G., Gao, L.: Sharp and real image super-resolution using generative adversarial network. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.S. (eds.) Neural Information Processing. ICONIP 2017. LNCS, vol. 10636, pp. 217–226. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70090-8_23
22. Li, R., Pan, J., Li, Z.: Single image dehazing via conditional generative adversarial network. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8202–8211 (2018)
23. Press, O., Bar, A., Bogin, B.: Language generation with recurrent generative adversarial networks without pre-training. arXiv:1706.01399 [cs.CL] (2017)
24. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv:1412.6980 [cs.LG] (2017)
25. Ji, J., Guo, Y., Gong, D., et al.: MOEA/D-based participant selection method for crowdsensing with Social awareness. Appl. Soft Comput. **87** (2019). https://doi.org/10.1016/j.asoc.2019.105981
26. Tian, Y., Cheng, R., Zhang, X., et al.: PlatEMO: a MATLAB platform for evolutionary multi-objective optimization. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2019)
27. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2008)