A Bi-objective Game-based Task Scheduling Method in Cloud Computing Environment

Wanwan Guo[1], Mengkai Zhao[1], Zhihua Cui[1*], and Liping Xie[1]
[1] School of Computer Science and Technology, Taiyuan University of Science and Technology
Taiyuan, Shanxi 030024 China.
[e-mail: gww199861@163.com, taoxi_zhao@163.com, cuizhihua@gmail.com, lipingxie@163.com]
[*]Corresponding author: Zhihua Cui

Received September 21, 2022; revised October 22, 2022; accepted November 6, 2022; published November 30, 2022

Abstract

The task scheduling problem has received a lot of attention in recent years as a crucial area for research in the cloud environment. However, due to the difference in objectives considered by service providers and users, it has become a major challenge to resolve the conflicting interests of service providers and users while both can still take into account their respective objectives. Therefore, the task scheduling problem as a bi-objective game problem is formulated first, and then a task scheduling model based on the bi-objective game (TSBOG) is constructed. In this model, energy consumption and resource utilization, which are of concern to the service provider, and cost and task completion rate, which are of concern to the user, are calculated simultaneously. Furthermore, a many-objective evolutionary algorithm based on a partitioned collaborative selection strategy (MaOEA-PCS) has been developed to solve the TSBOG. The MaOEA-PCS can find a balance between population convergence and diversity by partitioning the objective space and selecting the best converging individuals from each region into the next generation. To balance the players' multiple objectives, a crossover and mutation operator based on dynamic games is proposed and applied to MaPEA-PCS as a player's strategy update mechanism. Finally, through a series of experiments, not only the effectiveness of the model compared to a normal many-objective model is demonstrated, but also the performance of MaOEA-PCS and the validity of DGame.

Keywords: Bi-objective game, cloud computing, many-objective optimization algorithms, task scheduling.

http://doi.org/10.3837/tiis.2022.11.006 **ISSN : 1976-7277**

1. Introduction

With the global cloud computing market booming, cloud computing is becoming increasingly popular due to its virtualization, hyper-scale, and high-reliability features. It has played a significant role in the growth of the digital economy and will continue to do so [1]. The rapid development of cloud computing has also led to an increasing number of enterprises choosing to go to the cloud and use it [2], which inevitably leads to an increase in demand for cloud resources. Although cloud service providers can provide abundant computing resources, the diverse requirements of tasks make it exponentially more difficult to schedule tasks in a cloud computing environment [3]. Irrational scheduling schemes can easily result in low resource utilization, serious energy wastage, and so on. Therefore, the study of task scheduling is significant and worthwhile [4].

In cloud computing, the main parties involved in scheduling are the user and the cloud service provider. And the task scheduling is essentially a mapping between the tasks submitted by users and the virtual machines owned by the cloud. In other words, users submit tasks to the cloud, and the scheduler is responsible for assigning tasks to virtual machines with different attributes according to different users' needs in terms of time and cost, etc. Of course, while satisfying users' needs, the scheduler also ensures that the resources in the cloud resource system are used in a rational manner to the maximum extent possible. Recently, considerable researches have been devoted to task scheduling in the cloud environment [5]. In order to optimize the maximum completion time required to schedule tasks, Mohamed et al. [6] employed differential evolution to enhance the moth search algorithm's lack of mining capability, while Xiong et al. [7] devised a genetic algorithm based on Johnson's rule. These articles only consider scheduling time. However, users are also concerned with objectives such as cost and task completion rates.

In order to improve the user's service experience as much as possible, Bezdan et al. [8] proposed a hybrid bat algorithm to optimize scheduling cost and time, which can deal with the lack of search capability of the traditional algorithm. To improve task scheduling performance, Zhang et al. [9] first classify tasks and then dynamically match them with virtual machines. This method minimizes user payment costs and task scheduling time. Youne et al. [10] set priorities for tasks and use the proposed RAO algorithm to find execution time optimal solutions for tasks based on user demand scheduling policies. Tong et al. [11] provided an efficient scheduling solution by exploiting the adaptive learning capability of the dual-depth Q-network to shorten the response time while guaranteeing task completion. As can be seen, these studies only design objectives from the perspective of serving users. Nevertheless, this is a rather single perspective to consider, and the optimization objectives are not comprehensive enough.

Unlike the quality of service that the user is concerned about, service providers care more about whether their resources are well utilized and whether energy is wasted [12]. Therefore, to solve the problem of excessive energy consumption of cloud resources, Hussain et al. [13] developed a two-stage scheduling approach, which can successfully cut the system's energy use while achieving the task deadline restriction. A novel energy-aware service scheduling technique was also put forth by Zhu et al. [14] with the intention of lowering energy usage. Marahatta et al. [15] solved the problem of low resource utilization and high energy consumption that easily occurs by merging heterogeneous tasks and virtual machines in a virtual trick most-available scheduling scheme. Yuan et al. [16] presented a multi-objective optimization algorithm which can optimize the profit, energy cost, and maximize the benefits of the service provider while performing all tasks.

However, it is too restrictive to design the objectives for only one of the parties involved in the task scheduling. Once scheduling has started, both the service provider and the user have their own motivations [17]. And we find that all objectives must be optimized concurrently in order to arrive at a mutually satisfactory scheduling solution. To reduce scheduling time as well as increase throughput, Attiya et al. [18] combined improved manta ray foraging optimization with the salp swarm algorithm to propose a new hybrid swarm intelligent optimization method. Hu et al. [19] formulated the scheduling problem as a non-linear mixed integer programming problem that can effectively balance the time and energy consumption between conflicts and offer a real-time scheme. To reduce time and energy usage, Emami et al. [20] introduced the new pollination strategy in the sunflower optimization algorithm to find a scheduling solution. Zade et al. [21] created a method for fuzzy-based task scheduling to optimize the total time and load balancing ratio considering security issues and energy costs. Shukri et al. [22] improved the multi-verse optimizer by saving some of the better solutions to feed back into the algorithm and using this algorithm to reduce scheduling time and improve resource utilization.

Nevertheless, these researches continue to simply take into account the conflicting between the objectives, ignoring the conflicting interests of the participants in the actual scheduling process. To address this problem, we introduce game theory. Game theory [23] is the process by which individuals or organizations choose a strategy and implement it in a strategy set, either simultaneously or sequentially, once or repeatedly, under certain rules, in order to achieve the appropriate outcome [24]. Of course, each action of a participant in a game is designed to increase their own payoff. Whereas, there may be more than one payoff function for each player in the game, so it becomes a difficult task for the players to balance their multi-objectives during the game [25].

Evolutionary algorithms [26], as a mature and highly robust and widely applicable global optimization method, excel at balancing conflicts between multiple objectives to provide decision makers with a set of solutions. Simultaneously, it has been applied in a variety of fields for the past few years, like recommendation systems [27], cloud computing [28], medical diagnosis [29], and so on. However, only 2-3 objectives can be solved using standard evolutionary algorithms, and when the number of objective dimensions increases, multi-objective optimization algorithms will generate a significant number of non-dominated solutions, which can make it tough for the algorithm to select the top solution among a bunch of solutions [30]. Therefore, designing suitable selection strategies to reduce the selection pressure is also a major problem that we need to address.

Based on the above analysis, we need to address two key challenges: 1) how the scheduling model should be designed to reflect the conflicting interests of users and service providers, and 2) how to create an appropriate algorithm to solve this model. To address these challenges, the scheduling process is formulated as a game. In the game, users and service providers sequentially change their strategies to improve and balance their two payoff functions. Then, based on the game model's characteristics, we propose a many-objective evolutionary algorithm based on a partitioned collaborative selection strategy (MaOEA-PCS), by which players can obtain a scheduling solution that can satisfy both bi-objective. The following is a summary of this paper's main contributions.

(1) The task scheduling problem is modeled as a game and we design the task scheduling model based on a bi-objective game (TSBOG). It considers both the energy consumption and resource utilization concerns of the service provider and the cost and task completion rate concerns of the user.

(2) A many-objective evolutionary algorithm based on a partitioned collaborative selection strategy is proposed to solve this TSBOG. Then we design a new crossover and mutation operators based on the dynamic game as the players' strategy update mechanism.

The rest of this paper is structured as follows. In Section 2, the basic game model and the specific payoff functions for each player are defined. Section 3 gives a many-objective evolutionary algorithm based on a partitioned collaborative selection strategy, detailing the partitioned collaborative. The simulation experiments in Section 4 show the effectiveness of the suggested algorithm and strategy. Section 5 concludes our work and provides an outlook for the future.

2. System Model

In the section, we provide a task scheduling model based on a bi-objective game and design individual decision models for each player. The model takes multiple objectives that players focus on as the payoff functions and specifies these objectives.

2.1 Basic Game Model

Fig. 1 illustrates a cloud environment consisting of users, virtual machines, and data centers, where the user generates a large number of complex computational tasks. However, many computational tasks are difficult to complete locally due to the user's limited local computing power. Therefore, users need to submit tasks to the cloud, which achieves higher computational efficiency by allocating the tasks to appropriate virtual machines. Whereas, as mentioned in the introduction, due to the diversity of task requirements, it is not straightforward to select a suitable VM for the tasks while satisfying the interests of both the user and the cloud provider. Consequently, we will apply game theory to mitigate the conflicting interests between the user and cloud service provider to maximize the players' own interests. Users and service providers take on the roles of players and adjust their game strategies against each other to determine the final scheduling solution.

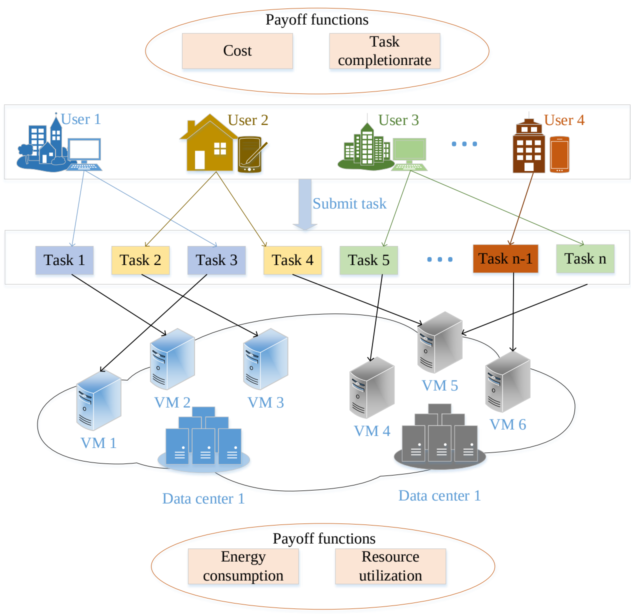


Fig. 1. The schematic of task scheduling

Now, we can define the basic game model as a triple $Y = \{W, p, (E_i)_{i \in W}\}$, where $W = \{1, 2\}$ is set of players (service provider and user), p is the scheduling strategy. And E_i is the payoff function of player i.

(1) Players: We consider the participants in the scheduling process as players, i.e., users and service providers.

(2) Strategy: In this game model, a strategy represents a scheduling solution, and the players adjust their strategies by changing the correspondence between tasks and virtual machines. Since a many-objective optimization algorithm will be used in this study to solve this game model, a chromosome is a solution, i.e., an element in the strategy set, and all the individual P's are combined to form the players' strategy set. The **Table 1** is shown below. The yellow square represents the third task being given to the eighth VM.

Table 1. Player strategy set

	1	**2**	**3**	**4**	**5**	**...**	**198**	**199**	**200**
P1	5	7	8	5	3		12	2	10
P2	...	...	...	...	...	...	...	...	...
...	7	14	9	3	3	...	1	13	8
PN	1	5	11	15	8	...	2	6	4

(3) Payoff function: Unlike single-objective games, we design models in which each player has two objectives that they want to optimize. The payoff function of service providers is expressed as U_1, while the user is U_2. The basic framework is defined as:

$$U_1 = \begin{cases} EC(p) \\ RU(p) \end{cases} \tag{1}$$

$$U_2 = \begin{cases} C(p) \\ S(p) \end{cases} \tag{2}$$

In this model, each player has two different payoff functions, and they take into account various factors to improve their own payoff. Therefore, the following section will introduce the service provider decision model and the user decision model to detail the specific manifestations of each player's payoff functions.

2.2 The Service Provider Decision Model

As a participant in the game, the service provider needs an explicit model to determine the energy consumption and resource utilization during the game process. Therefore, in this section, we will specify the two objectives of the game.

(1) energy consumption

During the scheduling process, whether the cloud service provider is performing a task or is idle, it is imperative that energy is consumed, which is a cost to the service provider, and less energy is more profitable to the service provider. Referring to [31], let E and S represent the energy consumption per time unit when the VM is executing a task or idle, respectively. Hence, the energy consumed for a complete scheduling process can be calculated as:

$$EC = \sum_{i=1}^{h} (vm_{i,exe} \times E + (t - vm_{i,exe}) \times S) \tag{3}$$

$$t = \max_{i=1}^{h} vm_{i,exe} \tag{4}$$

where t represents total scheduling time, $vm_{i,exe}$ reflects the execution time of the i -th virtual machine, and the total amount of virtual machines is h .

(2) resources utilization

As the provider of resources in the scheduling process, the extent to which VM resources are utilized is directly related to their interest. We define the resource utilization by the time spent per virtual machine performing a task, which is calculated by calculating the ratio of the execution time of each virtual machine to the total scheduling time and then taking the average. The specific formula is shown below.

$$RU = \frac{1}{h}\sum_{i=1}^{h}\frac{vm_{i,exe}}{t} \tag{5}$$

2.3 The User Decision Model

We can use three parameters to abstract and depict a task as (f_j, l_j, u_j), where f_j, l_j, u_j represents the input size, file size and output size of the j -th task, respectively. After the task is received by the cloud server, the task will be assigned to different VMs for execution. A task being assigned to VMs with different mips will affect the task's execution time and the bandwidth will affect its transmission time. Accordingly, with the bandwidth $B_{i,j}$ and mips $M_{i,j}$ of the j -th task assigned to the i -th VM, the i -th task's transmission and execution time is shown in (6) and (7), respectively.

$$t_{j,tra} = \frac{f_j + u_j}{B_{i,j}} \tag{6}$$

$$t_{j,exe} = \frac{l_j}{M_{i,j}} \tag{7}$$

Based on this basic information, we now move on to the user's payoff functions.

(1) cost

Typically, we use the transmission time and execution time of a task as a basis for evaluating the cost required to perform the task. In other words, the cost incurred by the user increases with task execution and transmission time, given a certain transmission cost P_b and an execution cost P_m . Based on this principle, one of the payoff functions in the game is derived as:

$$C = \sum_{j=1}^{n}(t_{j,tran} \times P_b + t_{j,exe} \times P_m) \tag{8}$$

(2) task completion rate

If a task can be completed before the expected time, the user will receive a higher benefit, and the user will be more satisfied. If we use k to represent the total number of tasks to achieve the desired time, the ratio of k to the overall number of tasks n can be used to calculate the task completion rate, as shown in (9).

$$S = \frac{k}{n} \times 100\% \tag{9}$$

We now have the payoff functions for each player. While the user wants to spend less money and complete tasks more quickly, the service provider wants to use resources more efficiently while consuming less energy. Therefore, we formulate the task scheduling model based on the bi-objective game (TSBOG) as follows:

$$TSBOG\begin{cases} provider\ payoff\begin{cases}\min EC=\sum_{i=1}^{h}(vm_{i,exe}\times E+(t-vm_{i,exe})\times S)\\ \max RU=\frac{1}{h}\sum_{i=1}^{h}\frac{vm_{i,exe}}{t}\end{cases}\\ user\ payofff\begin{cases}\min C=\sum_{j=1}^{n}(t_{j,tran}\times P_b+c_{j,exe}\times P_m)\\ \max S=\frac{k}{n}\times 100\%\end{cases}\end{cases} \tag{10}$$

To solve the TSBOG, we will introduce a many-objective algorithm that we developed.

3. Proposed Method

In this section, the many-objective evolutionary algorithm based on a partitioned collaborative selection (MaOEA-PCS) is initially introduced, then we provide a thorough description of the environment selection strategy. Finally, based on the model features, we create a new crossover and mutation operator as a strategy update mechanism for players.

3.1 The Framework of The MaOEA-PCS

The **Algorithm 1** demonstrates the fundamental structure of MaOEA-PCS. To start with, the procedure begins with a randomly generated population of size N and a set of reference points Z, and then the algorithm starts into iterative optimization. The initial population is used as input for matching selection, then child populations are generated by traditional genetic operators including crossover and mutation [32]. Finally, the child populations will be used as input for environment selection along with the parent populations, and a partitioned solution selection strategy is used for choosing the best solutions into the next generation. When the current number of iterations G surpasses the maximum number of iterations $G_{\max}$, MaOEA-PCS will come to an end.

Algorithm 1: The framework of the proposed MaOEA-PCS

Input: the population size N, the maximum number of iterations $G_{\max}$
Output: the optimization results
Generating the initial population P_0
Generate a set of reference points Z
While $G < G_{\max}$
  Q_t = MatingPool (P_0)
  Q_t = Crossover mutation operators（Q_t）
  $R_t = P_0 + Q_t$
  P_{t+1} = Environment selection strategy pool（R_t）
End

3.2 The Partitioned Collaborative Selection Strategy (PCS)

It is well known that the goal of environment selection is to choose individuals who own excellent convergence and diversity for the next generation, and it frequently serves as a vital role in an algorithm. Additionally, as the objective dimension increases, numerous non-dominated solutions will be generated during the optimization process, leading to an exponential increase in selection pressure as well. Thus, how to select high-quality offspring is then the question we need to address. In our design, we associate each solution with the reference vector closest to it, so that the reference vector naturally decomposes the solution space into many small spaces, ensuring a diversity of solutions. We then express convergence in terms of the distance from an individual to an ideal point, so that it is possible to choose a group of solutions with splendid convergence and diversity by simply choosing the individual with the best convergence in each small space. The **Fig. 2**. below shows a simple example of how to select a solution.

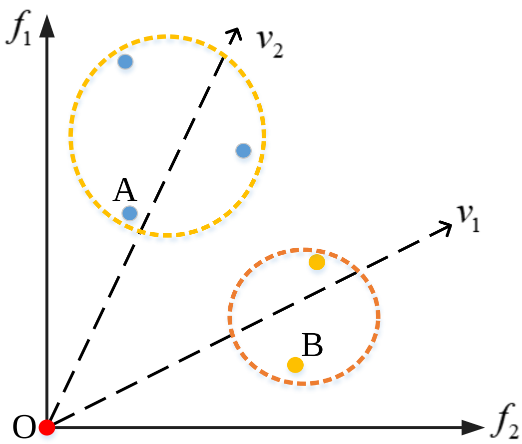


Fig. 2. The process of environment selection

The points in the yellow circle indicate individuals associated with the vector v_1 and the points in the blue circle indicate individuals associated with the vector v_2. Compared with the other points in the circle, points A and B are the closest to the ideal point O, which means they have better convergence. Hence, these two points are selected to enter the next generation.

This strategy balances the convergence and diversity of the population throughout the optimization process of the algorithm. Since the individuals assigned to the same reference vector contribute similarly to the diversity of the population, we can enhance the performance of the whole population by retaining only the individuals with the best convergence. Thus, our design improves the conflict between diversity and convergence in general.

3.3 Crossover and Mutation Operators Based on Dynamic Game（DGame）

We consider the task scheduling problem as a game and decide to solve it using an optimization algorithm. Obviously, the players change their strategies corresponding to the crossover and mutation operators. In the previous sections of the game model analysis, we designed a bi-objective function for the players of the game as their respective payoff functions, whereas, during the game, players also need a strategy update mechanism to continuously respond to the strategies of other players. Therefore, we designed a crossover and mutation operator suitable for TSBOG to obtain a mutually satisfactory solution. Finally, we apply the game-based crossover and mutation operators to the many-objective optimization algorithm to settle the TSBOG and demonstrate the effectiveness of this method in the experimental section.

Algorithm 2: The crossover operator

```
Input：the decision variables of population
Output：offspring population
Generate two random numbers a , b
Record the value of the objective functions before the crossover
For i = x   /* x is the number of cloudlet*/
    Generate a random number r
    If r < 0.9 / 2
        The a and b individuals start crossing in x gene positions
    End
    Calculate the value of the objective function
    If the player is the service provider
      If after (EC)>before (EC) && after (RU)<before (RU)
        Recovery strategy before crossover
      End
    else
      If after(C)>before(C) && after(S)<before(S)
        Recovery strategy before crossover
      End
    End
End
```

Algorithm 3: The mutation operator

```
Input：the decision variables of population
Output：offspring population
Generate a random number a   /* a is Individual mutation probability */
If a < 0.1 / 2
    Keep a record of the objective functions' value prior to the mutation
    For i = N   /* N is the population size*/
        Generate a random number b
        If b < 0.5
            Mutation at gene position i
        End
        Calculate the objective functions' value
        If the player is the service provider
            If after (EC)>before (EC) && after (RU)<before (RU)
                Recovery strategy before crossover
            End
        Else
            If after(C)>before(C) && after(S)<before(S)
                Recovery strategy before crossover
            End
        End
    End
End
```

As shown in **Algorithm 2**, after the player starts the crossover operation, it will first generate two random numbers, which represent two individuals in this crossover, and then start selecting the genetic to execute the crossover. The specific code is in lines 5-9 of Algorithm 2. Once the crossover is complete, the player needs to determine whether the individual before the crossover dominates the individual after the crossover; if it does, it means that the player's action does not give him or her a greater interest, and then the player will recover to the previous scheduling strategy. The same is true for the mutation operation. Each player performs both operations successively, and this strategy ensures that the players' actions are all in the direction of good. The whole game is manifested in the **Fig. 3** below.

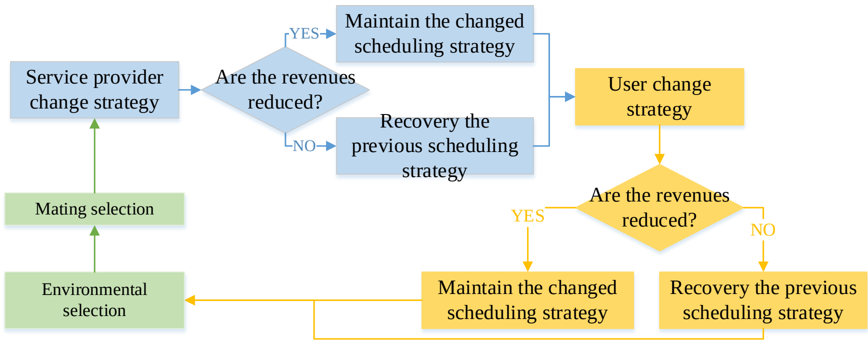


Fig. 3. The flowchart of player game

The service provider first begins to act according to Algorithms 2 and 3, and then the user chooses their own action in response based on the service provider's strategy. After both have completed their actions, the algorithm enters environment selection and mating selection until it enters the crossover and mutation operation once more.

4. Performance Evaluation

In this section, through numerical experiments as well as box plot analysis, we validated the effectiveness of the strategies and methods proposed in this study. First, we test the convergence of MaOEA-PCS and verify its performance on the test functions DTLZ1-7. In addition, we contrast MaOEA-PCS′s performance in solving TSBOG with four other many-objective optimization algorithms, as well as give evidence of the effectiveness of our proposed crossover and mutation operators based on the dynamic game and TSBOG.

4.1 Simulation Settings

To assess the efficacy of the MaOEA-PCS, we run it in PlatEMO [33] using MATLAB R2018a. The simulations were conducted on an AMD Ryzen 7 5800H, 3.20 GHz PC with 16GB RAM. For simplicity without losing generality, we consider the task scheduling environment consisting of 200 tasks, 15 virtual machines and 3 data centers. The task length, file size and output size can be described as the arithmetic progression with a first term of 1000,300,150 and a tolerance of 100,15,10, respectively. The parameters of the virtual machine are manifested below.

Table 2. The VM Parameters

parameters	value
Bw (bit)	[1024-3072]
MIPS (mips)	[400-2000]
Ram (MB)	[512-3072]
cpu_cost	[0.03-0.09]
bw_cost	[0.01-0.03]

4.2 Approaches

4.2.1 The Inverted Generational Distance (IGD)

IGD [34, 35], as a performance metric, is often used to assess algorithm diversity and convergence. It evaluates the effectiveness of an algorithm by taking points uniformly from the true Pareto front and calculating and averaging the shortest distance between these points and the true solution obtained by the algorithm. In general, a lower value of IGD, we can obtain the superior the algorithm's overall performance.

$$IGD = \frac{\sum_{i=1}^{k} |dis_i|}{k} \tag{11}$$

where k represents the number of solutions in the true Pareto front, while dis_i stands for the nearest Euclidean distance from the i-th point of the true Pareto front to known true solutions.

4.2.2 Comparison Algorithms and Parameter settings

The performance of the MaOEA-PCS will be compared with the four most advanced approaches. To make the experimental results more reliable, all key parameter settings of NSGAIII, RVEA, GrEA and HMaPSO were consistent with the original reference. Besides, the parameters when all five algorithms solve the TSBOG are set as shown in **Table 3**.

NSGAIII[36]: As a classical many-objective optimization algorithm, it can solve NP-hard problems efficiently. The algorithm improves population diversity through a reference point strategy and non-dominated ranking ensures population convergence.

RVEA[37]: To dynamically coordinate the convergence and diversity of the many-objective optimization algorithms, an angle penalty distance is provided.

GrEA[38]: Similar to the idea of reference points, the algorithm maintains a wide and even distribution of solutions through grid dominance and grid congestion.

HMaPSO[39]: The authors chose differential evolution operators, simulated binary crossover operators, and particle swarm operators as a pool of selection strategies to produce excellent solutions, achieving satisfactory results in balancing convergence and diversity.

Table 3. The Parameters Settings

parameters involved	value
The number of objectives	4
Population size	100
Crossover probability of each player	0.45
Mutation probability of each player	0.05

4.3 Performance of MaOEA-PCS

4.3.1 The Convergence of Algorithms

An experimental analysis of the number of iterations of the algorithm was carried out for the purpose of determining the MaOEA-PCS's convergence limit. The specific experimental steps were to record the objective values every 100 generations and to take the average of these objective values to plot the results as shown below.

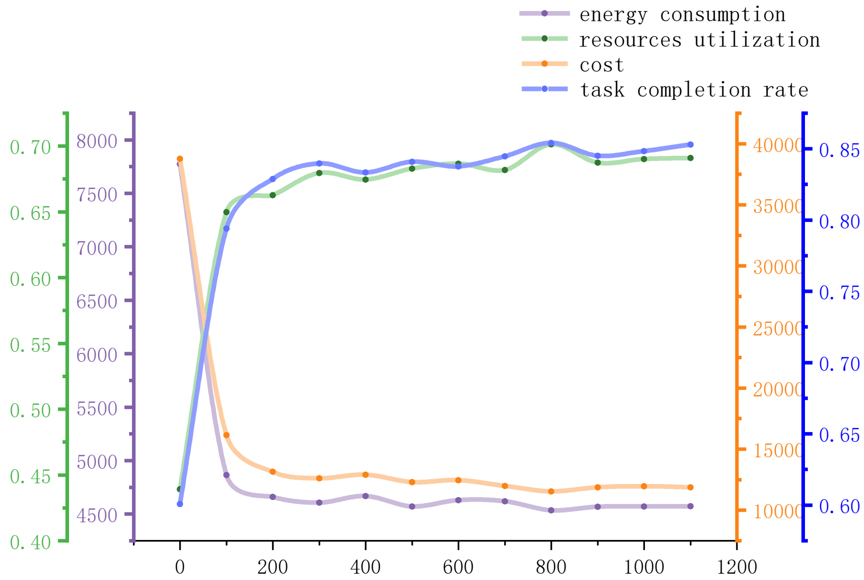


Fig. 4. Convergence of MaOEA-PCS

Distinctly, when fewer than 200 iterations have been completed, MaOEA-PCS converges more rapidly. Then, when the number of iterations ranges from 200 to 800, the rate of convergence decreases and the objective value fluctuates. And upon 800 iterations it is close to convergence and the objective value is basically unchanged. Therefore, if not otherwise mentioned, we limited the number of iterations for all methods in the following tests to 800.

4.3.2 The Results and Discussion in DTLZ Problems

To show off how well the MaOEA-PCS performs, DTLZ1-DTLZ7 are selected as the test functions for the experiments in this paper. In the experiments, we tested the selected five MaOEAs (i.e., NSGAIII, RVEA, GrEA, HMaPSO, and MaOEA-PCS) in the 4, 6, 8, 10, and 15-dimensional objective space of the benchmark problem, and the stopping criterion was set at 1,000 generations. As suggested in [40], in addition to setting the population sizes in the DTLZ test function to 120, 132, 156, 275, 135 respectively, we used the IGD values described in the previous subsection to measure the experimental results, and the **Table 4** below shows the IGD values obtained after 30 independent runs of all algorithms. The table's "+," "-," and "=" symbols signify that the comparison algorithms are better, worse, or approximately the same as those obtained by MaOEA-PCS, and the bolded fonts and gray background are the best for the different test cases.

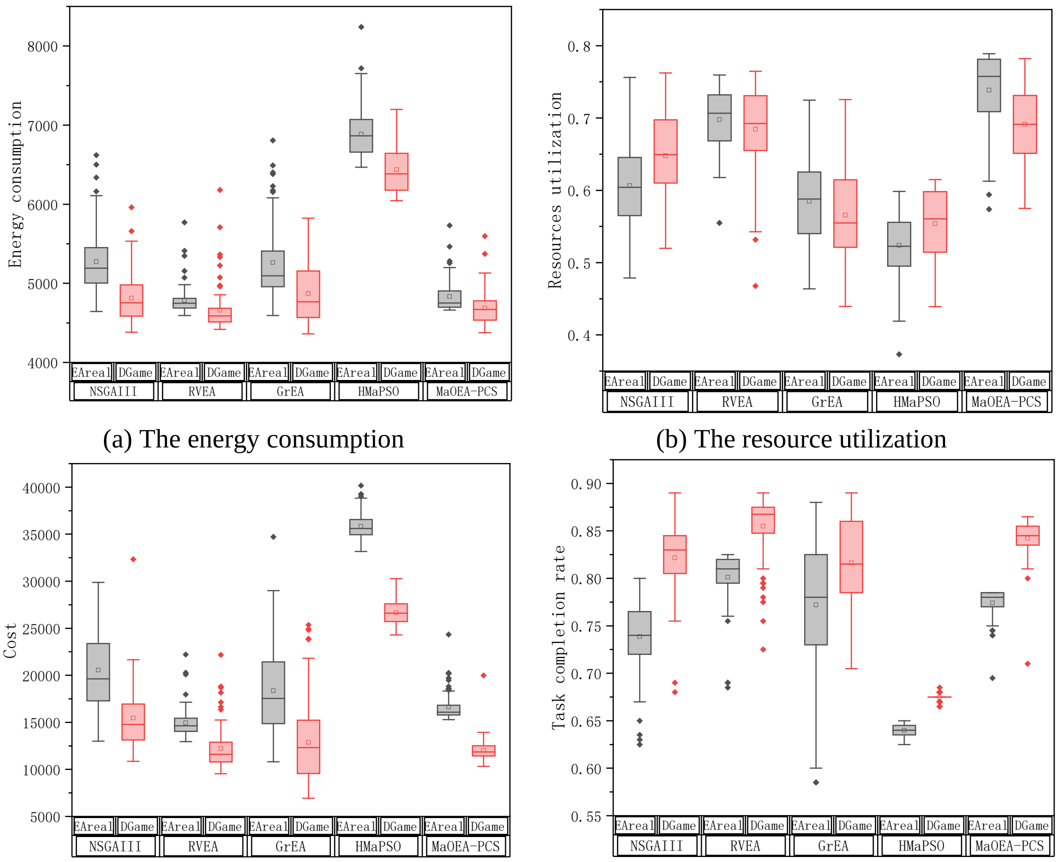
Table 4. The five algorithms' IGD results on the DTLZ

Problem	M	NSGAIII	RVEA	GrEA	HMaPSO	MaOEA-PCS
DTLZ1	4	5.6812e-1 (4.14e-1) =	6.0222e-1 (3.28e-1) =	**3.3982e-1 (2.28e-1) =**	1.1525e+0 (9.51e-1) -	4.5346e-1 (2.47e-1)
	6	1.0835e+0 (5.07e-1) -	5.2643e-1 (2.60e-1) =	1.0085e+0 (4.51e-1) -	9.7151e-1 (8.34e-1) -	**4.8753e-1 (2.29e-1)**
	8	2.1039e+0 (8.36e-1) -	5.6357e-1 (4.29e-1) =	1.4068e+0 (6.52e-1) -	1.2709e+0 (1.16e+0) -	**4.8745e-1 (2.57e-1)**
	10	3.0605e+0 (1.03e+0) -	**9.0454e-1 (3.36e-1) =**	3.4942e+0 (1.56e+0) -	1.5346e+0 (7.69e-1) -	9.4593e-1 (4.77e-1)
	15	1.0831e+0 (5.60e-1) -	5.4461e-1 (3.72e-1) -	1.2808e+1 (1.15e+1) -	1.8258e+0 (2.17e+0) -	**2.5490e-1 (7.44e-2)**
DTLZ2	4	1.2372e-1 (5.80e-4) =	**1.2355e-1 (5.77e-4) =**	1.3540e-1 (2.60e-3) -	1.4459e-1 (2.33e-3) -	1.2368e-1 (1.58e-3)
	6	2.7404e-1 (4.45e-3) -	2.5931e-1 (1.33e-3) -	2.5838e-1 (1.45e-3) -	2.6994e-1 (2.08e-3) -	**2.5281e-1 (6.50e-3)**
	8	3.7252e-1 (3.03e-2) -	**3.2916e-1 (2.60e-3) +**	3.9698e-1 (1.52e-2) -	3.7837e-1 (6.17e-2) -	3.5359e-1 (2.60e-2)
	10	5.4180e-1 (6.30e-2) -	4.2325e-1 (6.85e-3) -	4.1581e-1 (4.17e-3) -	4.5289e-1 (5.73e-2) -	**3.9641e-1 (5.49e-3)**
	15	6.6382e-1 (2.42e-2) +	**5.9405e-1 (1.15e-2) +**	6.0000e-1 (2.70e-2) +	8.4910e-1 (6.97e-2) -	8.1785e-1 (1.54e-1)
DTLZ3	4	2.1658e+1 (8.05e+0) =	2.2045e+1 (6.32e+0) =	**1.2757e+1 (4.04e+0) +**	5.7088e+1 (2.57e+1) -	2.1141e+1 (8.40e+0)
	6	4.6421e+1 (1.18e+1) -	2.2764e+1 (6.26e+0) =	3.8016e+1 (8.73e+0) -	4.5909e+1 (2.84e+1) -	**1.9921e+1 (4.58e+0)**
	8	6.8336e+1 (2.06e+1) -	**2.5207e+1 (9.94e+0) =**	7.9468e+1 (1.97e+1) -	4.0154e+1 (2.22e+1) -	2.8354e+1 (7.79e+0)
	10	1.0470e+2 (1.82e+1) -	5.0757e+1 (1.17e+1) -	1.9308e+2 (4.60e+1) -	4.5864e+1 (1.79e+1) =	**4.5071e+1 (1.09e+1)**
	15	5.3669e+1 (1.11e+1) -	1.8169e+1 (6.28e+0) =	2.2736e+2 (7.06e+1) -	4.4594e+1 (3.01e+1) -	**1.5441e+1 (6.67e+0)**
DTLZ4	4	2.2370e-1 (1.54e-1) -	**1.2415e-1 (9.03e-4) +**	2.0067e-1 (1.52e-1) -	2.0158e-1 (1.59e-1) -	1.3837e-1 (6.11e-2)
	6	3.0550e-1 (7.21e-2) -	2.7382e-1 (4.16e-2) +	**2.7192e-1 (5.98e-2) +**	4.1598e-1 (1.55e-1) -	2.8120e-1 (6.08e-2)
	8	4.3357e-1 (8.45e-2) -	3.5451e-1 (2.67e-2) -	3.6560e-1 (5.23e-3) -	6.0269e-1 (1.44e-1) -	**3.5058e-1 (2.19e-3)**
	10	5.6613e-1 (3.81e-2) -	4.6562e-1 (4.92e-3) -	4.2140e-1 (2.97e-3) -	5.7129e-1 (6.85e-2) -	**4.1179e-1 (2.43e-3)**
	15	6.9886e-1 (3.78e-2) -	6.3562e-1 (7.99e-3) -	**5.9200e-1 (7.73e-3) +**	8.0173e-1 (9.94e-2) -	5.9236e-1 (2.16e-2)
DTLZ5	4	5.6211e-2 (1.52e-2) +	1.7157e-1 (2.82e-2) =	8.4829e-2 (1.52e-2) +	**2.9225e-2 (4.19e-3) +**	1.8230e-1 (1.18e-1)
	6	2.3853e-1 (6.89e-2) -	2.0710e-1 (8.63e-2) -	1.7552e-1 (3.69e-2) =	**1.3236e-1 (6.63e-2) +**	1.6850e-1 (6.85e-2)
	8	1.8160e-1 (4.69e-2) =	3.6100e-1 (5.96e-2) -	2.6682e-1 (5.93e-2) -	5.2885e-1 (2.80e-1) -	**1.6546e-1 (4.66e-2)**
	10	1.9611e-1 (3.67e-2) -	3.5755e-1 (9.39e-2) -	3.0917e-1 (6.19e-2) -	7.1258e-1 (2.18e-1) -	**1.3221e-1 (2.61e-2)**
	15	**3.1378e-1 (5.34e-2) +**	4.4826e-1 (2.36e-1) =	5.1163e-1 (8.72e-2) -	8.8680e-1 (9.03e-2) -	3.7799e-1 (1.41e-1)
DTLZ6	4	5.3376e-1 (4.02e-1) -	2.2835e-1 (2.10e-1) -	1.7468e-1 (1.50e-1) -	**2.2971e-2 (4.82e-3) +**	1.0377e-1 (3.71e-2)
	6	3.6807e+0 (6.47e-1) -	3.9098e-1 (2.71e-1) -	5.5307e-1 (3.37e-1) -	2.8653e-1 (2.97e-1) =	**1.8411e-1 (6.57e-2)**
	8	5.3645e+0 (5.31e-1) -	5.7557e-1 (4.45e-1) -	4.4754e+0 (4.90e-1) -	6.8873e-1 (2.61e-1) -	**2.1317e-1 (1.38e-1)**
	10	6.6821e+0 (4.91e-1) -	1.5198e+0 (5.83e-1) -	3.8523e+0 (4.08e-1) -	8.2421e-1 (2.65e-1) -	**4.9896e-1 (3.42e-1)**
	15	5.2996e+0 (7.13e-1) -	9.2381e-1 (5.19e-1) -	3.5937e+0 (4.72e-1) -	8.9884e-1 (3.19e-1) -	**3.0256e-1 (1.94e-1)**
DTLZ7	4	3.0964e-1 (6.41e-2) +	4.5644e-1 (7.53e-2) =	**1.6656e-1 (9.88e-3) +**	1.7336e-1 (8.55e-2) +	4.6473e-1 (1.49e-2)
	6	8.9507e-1 (8.20e-2) +	1.0593e+0 (8.17e-2) +	5.5716e-1 (5.11e-2) +	**4.7748e-1 (1.35e-1) +**	2.4220e+0 (4.34e-1)
	8	5.2329e+0 (9.15e-1) -	1.7508e+0 (8.58e-1) +	1.2626e+0 (2.23e-1) +	**8.6508e-1 (2.01e-1) +**	3.9665e+0 (1.01e+0)
	10	1.4022e+1 (1.59e+0) -	3.4298e+0 (1.57e+0) +	7.3786e+0 (1.13e+0) -	**1.7599e+0 (6.60e-1) +**	4.1651e+0 (1.21e+0)
	15	1.7369e+1 (1.61e+0) -	4.6752e+0 (2.15e+0) =	1.8970e+1 (1.37e+0) -	**3.3381e+0 (7.85e-1) +**	4.4054e+0 (1.34e+0)

Observing the above **Table 4**, it is evident that out of the 35 test instances, the MaOEA-PCS algorithm obtained 16 optimal values, HMaPSO obtained 7, and performed exceptionally well on DTLZ7, while the classical RVEA, GrEA, and NSGAIII obtained 6, 5, and 1 optimal value in that order. On DTLZ1, MaOEA-PCS obtains 3 optima solutions and performs similarly to GrEA and RVEA in dimensions 4 and 10, indicating that this algorithm can solve such problems well in all dimensions. On DTLZ2-4, MaOEA-PCS obtained 2-3 optima solution in all 5 test instances of each problem. It is clear that the MaOEA-PCS algorithm has an obvious advantage in dealing with problems with objective dimensions of 8, 10 and 15, which may be due to the fact that our proposed PCS strategy reduces the selection pressure of evolutionary processes. Furthermore, on DTLZ5, both MaOEA-PCS and HMaPSO obtained two optimal solutions, but it can be seen that HMaPSO is more suitable for solving such problems in low dimensions, while MaOEA-PCS is more suitable for solving in high dimensions. And the MaOEA-PCS performed exceptionally well on DTLZ6, obtaining optimal solutions in dimensions 6, 8, 10 and 15, while on DTLZ7, MaOEA-PCS did not obtain any optimal solution. Based on the problem properties, it is presumed that MaOEA-PCS is weak in solving such problems with irregular Pareto front. Therefore, comparatively speaking to the other four evolutionary algorithms, the MaOEA-PCS solution put forward in this research provides a considerable advantage in handling the benchmark testing problem.

4.3.3 The Validity of DGame and TSBOG

To verify the validity of our proposed DGame, we set the operators in each of the five algorithms to run TSBOG with the simulated binary crossover (Eareal) and the DGame operator. We still use box plots to analyze the experimental results, which are shown below.



(a) The energy consumption (b) The resource utilization

(c) The cost (d) The task completion rate

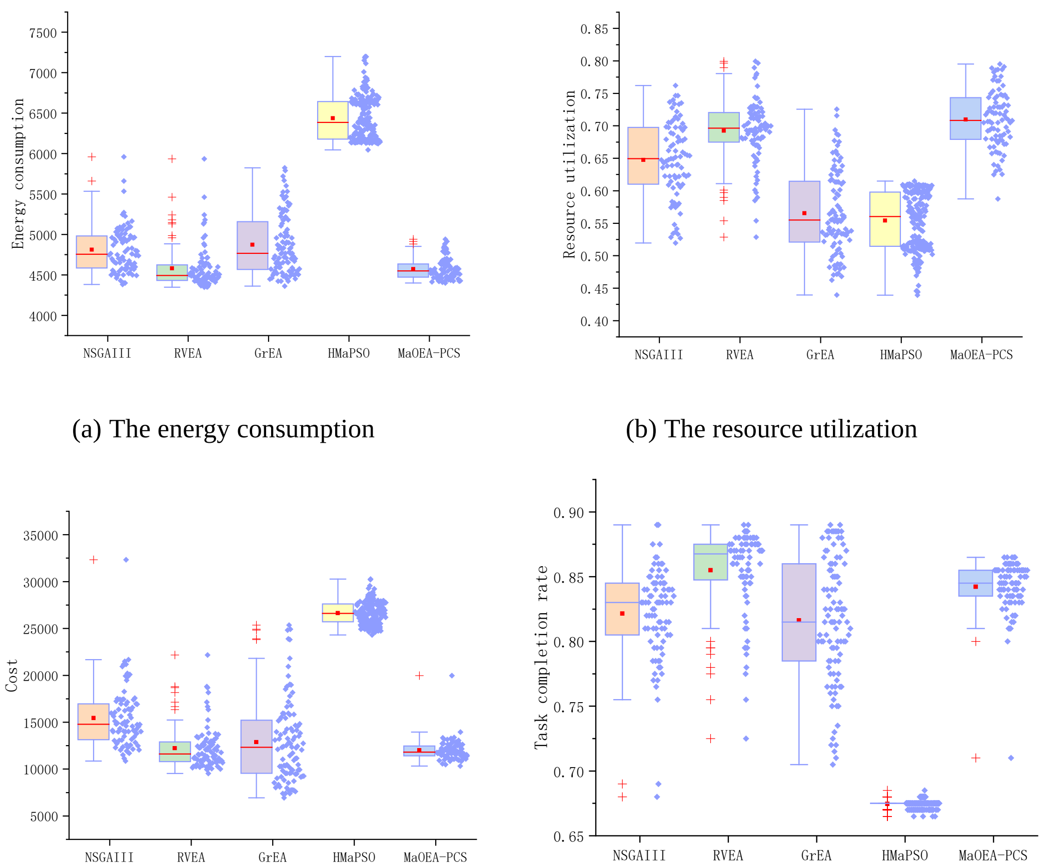
Fig. 5. The box plot comparison of two operators in four objectives.

The results achieved with the EAreal operator are shown in gray box plots, whereas the results with the DGame operator are shown in red box plots. As can be seen, all algorithms show a significant improvement in optimization on most objectives, but the magnitude of optimization on some objectives is not significant (e.g. task completion rate for the GrEA algorithm), probably because both sides of the game focus only on their own objectives during the evolutionary process, and therefore some solutions in favor of the other side are bound to be lost in the scheduling process.

Additionally, the use of different operators represents a comparison between the TSBOG and the ordinary many-objective optimization model. Based on the above analysis, we can demonstrate that the TSBOG performs better and that a better scheduling solution can be obtained for the same number of iterations.

4.3.4 The Effectiveness of The Algorithm Solving Model

To illustrate the distribution of individuals when solving TSBOG for different algorithms, **Fig. 6** shows box plots of the optimization outcomes for each objective function and the data distribution. From the figures, we can obtain the maximum, minimum, mean, median, and discrete values for this set of data.



(a) The energy consumption (b) The resource utilization

(c) The cost (d) The task completion rate

Fig. 6. The box plots result of six algorithms in four objectives

In terms of energy consumption, the overall box size of MaOEA-PCS is smaller than the other four algorithms, indicating that MaOEA-PCS has better convergence of the solution set. The median and mean are close to RVEA, which may be due to the fact that both use reference vectors to assist in solution selection. In terms of resource utilization, MaOEA-PCS has the optimal maximum, median, and mean values, indicating that PCS can obtain better optimization results. In addition, it is effortless to obtain that MaOEA-PCS performs better than the other algorithms in box size and maximum value in user cost. However, it is not as good as GrEA in terms of minimum value, but GrEA has a larger box. This may be because GrEA's grid domination approach does not obtain good convergence. In terms of task completion rate, MaOEA-PCS still has the optimal box size and minimum value, indicating that our proposed algorithm converges better. In contrast, the box for the HMaPSO algorithm is too small, indicating that the integration strategy proposed by the algorithm change affects the diversity of solutions.

In summary, TSBOG's effectiveness is proven, and MaOEA-PCS can give a better result in the DTLZ test set and TSBOG. In addition, DGame can productively improve the algorithm's optimization of the model.

5. Conclusion

In this paper, we consider the process of deciding the task scheduling strategy between users and service providers as a game, and propose a task scheduling model based on a bi-objective game, i.e. service providers focus on energy consumption and resource utilization, and users focus on cost and task completion rate. Then, we design a many-objective evolutionary algorithm based on a partitioned collaborative selection strategy to solve this model. Additionally, we redesign a crossover and mutation operator based on dynamic game as the player's strategy update mechanism. Finally, through a series of experiments we test the convergence of MaOEA-PCS and demonstrate that the algorithm performs well in the DTLZ test set as well as in solving the TSBOG model. Furthermore, we verified the performance of DGame and TSBOG by comparing the DGame operator with the traditional crossover operator.

In the future, we will further consider the real-time of task arrivals and build a dynamic multi-objective scheduling model. In addition, the extension of independent tasks to workflows is also an important issue worth investigating.

Acknowledgement

This work was supported by the Central Government Guides Local Science and Technology Development Funds (Grant No.YDZJSX2021A038), the National Natural Science Foundation of China (Grant No.61806138), Graduate Innovation Project Fund (Grant No.SY2022061).

References

[1] E. H. Houssein, A. G. Gad, Y. M. Wazery, and P. N. Suganthan, "Task Scheduling in Cloud Computing based on Meta-heuristics: Review, Taxonomy, Open Challenges, and Future Trends," *Swarm Evol. Comput.*, vol. 62, pp. 100841, Apr. 2021. Article (CrossRef Link)

[2] L. Z. Wang, G. von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, "Cloud Computing: a Perspective Study," *New Generation Computing*, vol. 28, no. 2, pp. 137-146, Jun. 2010. Article (CrossRef Link)

[3] X. Z. Kong, C. Lin, Y. X. Jiang, W. Yan, and X. W. Chu, "Efficient dynamic task scheduling in virtualized data centers with fuzzy prediction," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1068-1077, Jul. 2011. Article (CrossRef Link)
[4] Z. P. Peng, D. L. Cui, J. L. Zuo, Q. R. Li, B. Xu, and W. W. Lin, "Random task scheduling scheme based on reinforcement learning in cloud computing," *Cluster Computing-the Journal of Networks Software Tools and Applications*, vol. 18, no. 4, pp. 1595-1607, Sep. 2015. Article (CrossRef Link)
[5] Z. Y. Gao, Y. Wang, Y. F. Gao, and X. T. Ren, "Multiobjective noncooperative game model for cost-based task scheduling in cloud computing," *Concurrency and Computation-Practice & Experience*, vol. 32, no. 7, Dec. 2020. Article (CrossRef Link)
[6] M. Abd Elaziz, S. W. Xiong, K. P. N. Jayasena, and L. Li, "Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution," *Knowledge-Based Syst.*, vol. 169, pp. 39-52, Apr. 2019. Article (CrossRef Link)
[7] Y. H. Xiong, S. Z. Huang, M. Wu, J. H. She, and K. Y. Jiang, "A Johnson's-Rule-Based Genetic Algorithm for Two-Stage-Task Scheduling Problem in Data-Centers of Cloud Computing," *IEEE Trans. Cloud Comput.*, vol. 7, no. 3, pp. 597-610, Jul.-Sep. 2019. Article (CrossRef Link)
[8] T. Bezdan, M. Zivkovic, N. Bacanin, I. Strumberger, E. Tuba, and M. Tuba, "Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm," *J. Intell. Fuzzy Syst.*, vol. 42, no. 1, pp. 411-423, Dec. 2021. Article (CrossRef Link)
[9] P. Y. Zhang, and M. C. Zhou, "Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 772-783, Apr. 2018. Article (CrossRef Link)
[10] A. Younes, M. K. Elnahary, M. H. Alkinani, and H. H. El-Sayed, "Task Scheduling Optimization in Cloud Computing by Rao Algorithm," *CMC-Comput. Mater. Continua*, vol. 72, no. 3, pp. 4339-4356, Apr. 2022. Article (CrossRef Link)
[11] Z. Tong, F. Ye, B. L. Liu, J. H. Cai, and J. Mei, "DDQN-TS: A novel bi-objective intelligent scheduling algorithm in the cloud environment," *Neurocomputing*, vol. 455, pp. 419-430, Sep. 2021. Article (CrossRef Link)
[12] M. S. A. Khan, and R. Santhosh, "Task scheduling in cloud computing using hybrid optimization algorithm," *Soft Comput.*, vol. 26, pp. 13069-13079, 2022. Article (CrossRef Link)
[13] M. Hussain, L. F. Wei, A. Lakhan, S. Wali, S. Ali, and A. Hussain, "Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing," *Sustainable Computing-Informatics & Systems*, vol. 30, Jun. 2021. Article (CrossRef Link)
[14] X. Zhu, L. T. Yang, H. Chen, J. Wang, S. Yin, and X. Liu, "Real-Time Tasks Oriented Energy-Aware Scheduling in Virtualized Clouds," *IEEE Trans. Cloud Comput.*, vol. 2, no. 2, pp. 168-180, Apr. 2014. Article (CrossRef Link)
[15] A. Marahatta, S. Pirbhulal, F. Zhang, R. M. Parizi, K. K. R. Choo, and Z. Y. Liu, "Classification-Based and Energy-Efficient Dynamic Task Scheduling Scheme for Virtualized Cloud Data Center," *IEEE Trans. Cloud Comput.*, vol. 9, no. 4, pp. 1376-1390, Oct. 2021. Article (CrossRef Link)
[16] H. T. Yuan, H. Li, J. Bi, and M. C. Zhou, "Revenue and Energy Cost-Optimized Biobjective Task Scheduling for Green Cloud Data Centers," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 2, pp. 817-830, Feb. 2021. Article (CrossRef Link)
[17] H. Mahmoud, M. Thabet, M. H. Khafagy, and F. A. Omara, "Multiobjective Task Scheduling in Cloud Environment Using Decision Tree Algorithm," *IEEE Access*, vol. 10, pp. 36140-36151, Mar. 2022. Article (CrossRef Link)
[18] I. Attiya, M. Abd Elaziz, L. Abualigah, T. N. Nguyen, and A. A. Abd El-Latif, "An Improved Hybrid Swarm Intelligence for Scheduling IoT Application Tasks in the Cloud," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6264-6272, Sep. 2022. Article (CrossRef Link)
[19] B. Hu, Z. C. Cao, and M. C. Zhou, "Scheduling Real-Time Parallel Applications in Cloud to Minimize Energy Consumption," *IEEE Trans. Cloud Comput.*, vol. 10, no. 1, pp. 662-674, Jan. 2022. Article (CrossRef Link)
[20] H. Emami, "Cloud task scheduling using enhanced sunflower optimization algorithm," *Ict Express*, vol. 8, no. 1, pp. 97-100, Mar. 2022. Article (CrossRef Link)

[21] B. M. H. Zade, N. Mansouri, and M. M. Javidi, "SAEA: A security-aware and energy-aware task scheduling strategy by Parallel Squirrel Search Algorithm in cloud environment," *Expert Syst. Appl.*, vol. 176, Aug. 2021. Article (CrossRef Link)

[22] S. E. Shukri, R. Al-Sayyed, A. Hudaib, and S. Mirjalili, "Enhanced multi-verse optimizer for task scheduling in cloud computing environments," *Expert Syst. Appl.*, vol. 168, Apr. 2021. Article (CrossRef Link)

[23] R. Trestian, O. Ormond, and G. M. Muntean, "Game Theory-Based Network Selection: Solutions and Challenges," *IEEE Commun. Surv. Tutorials*, vol. 14, no. 4, pp. 1212-1231, Feb. 2012. Article (CrossRef Link)

[24] M. G. Fiestras-Janeiro, I. Garcia-Jurado, A. Meca, and M. A. Mosquera, "Cooperative game theory and inventory management," *Eur. J. Oper. Res.*, vol. 210, no. 3, pp. 459-466, May. 2011. Article (CrossRef Link)

[25] J. H. Xiao, W. Y. Zhang, S. Zhang, and X. Y. Zhuang, "Game theory-based multi-task scheduling in cloud manufacturing using an extended biogeography-based optimization algorithm," *Concurrent Engineering-Research and Applications*, vol. 27, no. 4, pp. 314-330, Oct. 2019. Article (CrossRef Link)

[26] J. Zou, Q. Y. Li, S. X. Yang, J. H. Zheng, Z. Peng, and T. R. Pei, "A dynamic multiobjective evolutionary algorithm based on a dynamic evolutionary environment model," *Swarm Evol. Comput.*, vol. 44, pp. 247-259, Feb. 2019. Article (CrossRef Link)

[27] X. J. Cai, Z. M. Hu, and J. J. Chen, "A many-objective optimization recommendation algorithm based on knowledge mining," *Inf. Sci.*, vol. 537, pp. 148-161, Oct. 2020. Article (CrossRef Link)

[28] J. L. Xu, Z. X. Zhang, Z. M. Hu, L. Du, and X. J. Cai, "A many-objective optimized task allocation scheduling model in cloud computing," *Applied Intelligence*, vol. 51, no. 6, pp. 3293-3310, Nov. 2021. Article (CrossRef Link)

[29] X. Cai, Y. Lan, Z. Zhang, J. Wen, Z. Cui, and W. S. Zhang, "A Many-objective Optimization based Federal Deep Generation Model for Enhancing Data Processing Capability in IOT," *IEEE Trans. Ind. Inf.*, vol. 19, no, 1, pp. 561-569, 2023. Article (CrossRef Link)

[30] X. J. Cai, S. J. Geng, J. B. Zhang, D. Wu, Z. H. Cui, W. S. Zhang, and J. J. Chen, "A Sharding Scheme-Based Many-Objective Optimization Algorithm for Enhancing Security in Blockchain-Enabled Industrial Internet of Things," *IEEE Trans. Ind. Inf.*, vol. 17, no. 11, pp. 7650-7658, Jan. 2021. Article (CrossRef Link)

[31] X. J. Cai, S. J. Geng, D. Wu, J. H. Cai, and J. J. Chen, "A Multicloud-Model-Based Many-Objective Intelligent Algorithm for Efficient Task Scheduling in the Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9645-9653, Jun. 2021. Article (CrossRef Link)

[32] B. Mc Ginley, J. Maher, C. O'Riordan, and F. Morgan, "Maintaining Healthy Population Diversity Using Adaptive Crossover, Mutation, and Selection," *IEEE Trans. Evol. Comput.*, vol. 15, no. 5, pp. 692-714, Oct. 2011. Article (CrossRef Link)

[33] Y. Tian, R. Cheng, X. Y. Zhang, and Y. C. Jin, "PlatEMO: A MATLAB Platform for Evolutionary Multi-Objective Optimization," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73-87, Nov. 2017. Article (CrossRef Link)

[34] P. A. N. Bosman, and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 174-188, Apr. 2003. Article (CrossRef Link)

[35] H. Ishibuchi, R. Imada, Y. Setoguchi, and Y. Nojima, "Reference Point Specification in Inverted Generational Distance for Triangular Linear Pareto Front," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 961-975, Dec. 2018. Article (CrossRef Link)

[36] K. Deb, and H. Jain, "An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems with Box Constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577-601, Aug. 2014. Article (CrossRef Link)

[37] R. Cheng, Y. C. Jin, M. Olhofer, and B. Sendhoff, "A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 773-791, Oct. 2016. Article (CrossRef Link)

[38] S. X. Yang, M. Q. Li, X. H. Liu, and J. H. Zheng, "A Grid-Based Evolutionary Algorithm for Many-Objective Optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721-736, Oct. 2013. Article (CrossRef Link)
[39] Z. H. Cui, J. J. Zhang, D. Wu, X. J. Cai, H. Wang, W. S. Zhang, and J. J. Chen, "Hybrid many-objective particle swarm optimization algorithm for green coal production problem," *Inf. Sci.*, vol. 518, pp. 256-271, May. 2020. Article (CrossRef Link)
[40] X. J. Cai, S. J. Geng, D. Wu, and J. J. Chen, "Unified integration of many-objective optimization algorithm based on temporary offspring for software defects prediction," *Swarm Evol. Comput.*, vol. 63, Jun. 2021. Article (CrossRef Link)

Wanwan Guo is currently working toward M.S. degree at computer science and technology, Taiyuan University of Science and Technology, Taiyuan, China. Her research interests include evolutionary computation, cloud computing and game theory.

Mengkai Zhao is currently working toward M.S. degree at computer science and technology, Taiyuan University of Science and Technology, Taiyuan, China. His research interests include computational intelligence, cloud computing, task scheduling.

Zhihua Cui received the PhD degree in control theory and engineering from Xi'an Jiaotong University, Xi'an, China, in 2008. He is currently a professor with the School of Computer Science and Technology, Taiyuan University of Science and Technology, China. He is the editor-in-chief of the International Journal of Bio-inspired Computation. His research interests include computational intelligence, stochastic algorithm, and combinatorial optimization.

LiPing Xie received the Ph.D. degree in control theory and control engineering from the Lanzhou University of Technology, in 2010. She is currently a Professor with the Institute of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan, China. She has published more than 30 international journal articles and conference papers. Her current research interests include swarm intelligence and swarm robotics.