IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 21, 2019, accepted September 23, 2019, date of publication September 30, 2019, date of current version October 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2944405

Cost-Efficient and Quality of Experience-Aware Provisioning of Virtual Machines for Multiplayer Cloud Gaming in Geographically Distributed Data Centers

YONGQIANG GAO, LIN WANG, AND JIANTAO ZHOU
Inner Mongolia A. R. Engineering Laboratory of Cloud Computing and Services Software, College of Computer Science, Inner Mongolia University, Hohhot 010021, China

Corresponding author: Yongqiang Gao (gaoyongqiang@imu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61662052, in part by the Inner Mongolia Science and Technology Innovation Team of Cloud Computing and Software Engineering, and in part by the Inner Mongolia Application Technology Research and Development Funding Project.

ABSTRACT Prompted by the remarkable progress in both cloud computing and GPU virtualization, cloud gaming has become increasingly more popular in the gaming industry. In cloud gaming, games are stored and run on cloud servers and the gamers interact with games through thin clients. Cloud gaming service providers generally employ multiple geographically distributed data centers to deliver their services. The main challenge for cloud gaming service providers is to find the best tradeoff between two contradicting objectives: reducing the infrastructure operating costs and increasing the quality of player's experience. In this paper, we address a virtual machine provisioning problem for multiplayer cloud gaming with the objective of minimizing both the inter-player delay among interacting players and the electricity costs of cloud gaming service providers, while providing the good-enough response delay to gamers. We formulate the problem into a constrained multiobjective optimization problem and propose an improved grey wolf algorithm to solve the problem. The performance of our proposed algorithm are assessed by simulation experiments based on the real-world parameters. The results show the superior performance of the proposed approach in comparison with the state-of-the-art approaches applied to similar problems.

INDEX TERMS Cloud gaming, distributed data center, grey wolf algorithm.

I. INTRODUCTION

As the gaming industry matures, games became more and more complex and thus demand for the latest hardware such as multicore processors and high-end graphic cards for fluent game playing. For example, the recent release of a multiplayer first-person shooter game Call of Duty: Black Ops 4 [1], which requires minimum specifications of a dual core CPU at 3.6 GHz, a graphics card with at least 2 GB RAM, 8 GB of memory space, and 60 GB of storage space. In addition, the new game may be incompatible with the computer configuration and thus gamers have to reconfigure their computers. To make matters worse, the computer may be abandoned since it cannot meet the requirements of new games, even though it work well. These issues impose heavy burdens on gamers and thus force some potential gamers to stay away from computer games. It is urgent and essential for the game industry to seek solutions to attract more gamers.

Cloud gaming, as a flourishing gaming model, is a solution for easing the burdens on gamers, which frees gamers from downloading or installing the game and constantly upgrading their computers. In cloud gaming, games are stored and run on cloud servers and the gamers interact with games through thin clients. Specifically, the cloud servers are responsible for the interpretation of gamer input, the execution of game code, the graphics rendering, the transmission of game scenes to clients over the Internet, while the thin clients are in charge of decoding and displaying game scenes to gamers, and capturing and sending gamers' inputs to cloud servers in real time. Prompted by the remarkable progress in both virtualization and GPU technology, cloud gaming has become increasingly more popular in the gaming industry. Many companies have

The associate editor coordinating the review of this manuscript and approving it for publication was Rute C. Sofia.

started to provide cloud gaming services, such as OnLive, Gaikai, CiiNow, and Ubitus. The cloud gaming market has been forecasted to reach 6.944 billion US dollars in 2026 [2].

Cloud gaming service providers generally utilizes multiple geographically distributed data centers to deliver their services. In this paper, we consider a multiplayer cloud gaming (MCG) as illustrated in Fig. 1, which is a new form of multiplayer online gaming in the cloud computing environment. The remote game server is identical to that in traditional multiplayer online gaming and only responsible to keep consistent game states among multiple game clients. After receiving a gamer request, the matchmaking server will assign the gamer request to a certain data center and then start a rendering server or a virtual machine (VM) with specialized graphic hardware to execute the requested game. On one hand, the rendering servers appear as the ''clients'' that are connected to the remote game server to exchange game states. On the other hand, the rendering servers function as the cloud servers which are responsible for processing game graphics and logics, and then streaming encoded game frames back to the gamer via the Internet. The clients running on gamer's device are called as ''thin-clients'' which transfer gamer's command to the rendering server and play the game frames sent back by the rendering server. Many MCG systems have emerged in recent years [3].

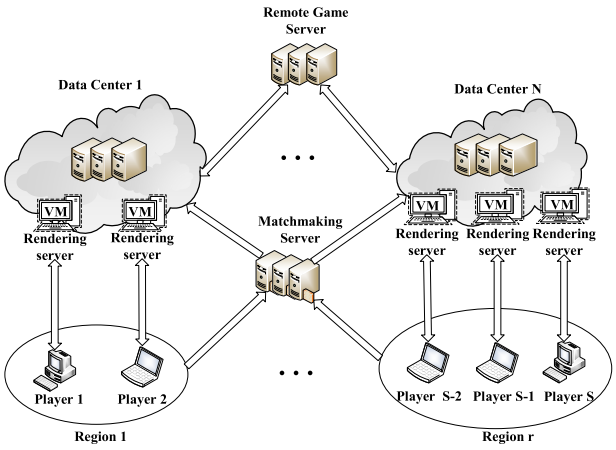

FIGURE 1. Typical multiplayer cloud gaming architecture.

The main challenge for MCG providers is to find the best tradeoff between two contradicting objectives: reducing the infrastructure operating costs and increasing the quality of player's experience (QoE). On the one hand, they need to provide enough high-performance servers that enable high QoE that lead to player satisfaction and loyalty. On the other hand, they have to reduce the operating costs as much as possible in order to increases the return on investment of cloud infrastructures. MCG demands for high QoE in terms of responsiveness and fairness. The responsiveness is assessed by the response delay, i.e. the time duration between the player issuing a command and the corresponding game frame being displayed on the screen. The fairness is evaluated by the inter-player delay, i.e. the difference of response delay among interacting players. It is worth pointing out that high inter-player delay can result in worse QoE than the high response delay [4]. The operating costs for MCG providers is mainly is determined by the electricity cost incurred due to energy consumption. The electricity costs involved in operating a large infrastructure of multiple data centers may vary depending on several factors. First, the electricity price may vary at different locations. Second, power usage effectiveness (PUE) may vary for different data centers. Finally, the power consumption of data centers may vary significantly depending on the number of active servers.

In this paper, we focus on the provisioning of the rendering servers (VMs) for MCG in geographically distributed data centers with the objective of minimizing both the inter-player delay among interacting players and the electricity costs of MCG providers, while providing the good-enough response delay to gamers. To the best of our knowledge, there is no previous work on simultaneously optimizing both QoE and electricity cost. Our main contributions are summarized as follows.

- We formulate the VM provisioning problem into a constrained multiobjective optimization problem to answer the following questions: a) how to provision VMs without violating the resource capacity and responsiveness constraints, b) how to adjust the number of active servers in each data center, c) how to fully exploit the geographical heterogeneity of electricity prices and data center PUEs and d) how to achieve a trade-off between the fairness and the electricity cost.
- We propose an improved grey wolf algorithm (MGWAM) to solve the problem. In the MGWAM, a modified social hierarchy is designed to improve its exploration and exploitation abilities. To further enhance the exploration, an adaptive mutation operator is integrated into the MGWAM. In addition, a mixed population containing both random and heuristic initial solutions is employed to accelerate the search progress.
- We conduct extensive simulations to verify the effectiveness of the proposed algorithm in the practical settings. Our simulation results show that, compared with other alternatives, our proposed algorithm can achieve lower electricity costs and better fairness, while providing the good-enough response delay to gamers.

The rest of this paper is organized as follows. The related works are summarized in Sect. II. Sect. III describes the system models and problem formulation. Our proposed MGWAM algorithm is presented in Sect. IV and Sect. V. In Sect. VI, the experimental evaluations and results are discussed. Finally, concluding remarks are given in Section VII.

II. RELATED WORK

Many efforts have been devoted to cloud gaming in recent years. Since cloud gaming is delay sensitive, a huge body of work focus on reducing the latency in cloud gaming platforms. AMIRI et al. [5] proposed a Lagrangian

Relaxation (LR) time-efficient heuristic algorithm to minimize the end-to-end latency within a cloud gaming data center. Wu et al. [6] proposed a delay-aware quality optimization framework called DAVIS to improve video streaming quality against burst packet losses in cloud-assisted real-time video delivery. Chen et al. [7] addressed the VM provisioning problem that aims at minimizing the inter-player delay, while preserving good-enough response delay experienced by players. These studies are different from our work because they focus solely on improving the performance of cloud gaming systems and thus cannot provide cost savings.

There is also some work focusing on reducing the operational cost of the cloud gaming service provides. Li et al. [8] proposed a play request dispatching algorithm to optimize the total service cost of a cloud gaming system using public cloud resources. Wang et al. [9] studied how to combine different virtual machine pricing models to serve time-varying demands at minimum cost. Li et al. [10] addressed the server provisioning problem for cloud gaming using public cloud resources, with the goal of minimizing the total server running cost and software storage cost. However, these studies only consider reducing the operational cost of the cloud gaming service provides and thus cannot provide QoE guarantees to gamers.

In addition to the above studies, there are also research efforts on minimizing the cloud gaming provider's cost while satisfying the QoE requirement of gamers. Hong et al. [11] proposed a VM placement algorithm that can minimize the operation cost of the cloud gaming platform while maintain good-enough experience. Deng et al. [3] investigated the server allocation problem for MCG with the objective of minimizing the total server rental and bandwidth cost under real-time latency constraint. Tian et al. [12] studied the selection of data centers, virtual machine allocation, and video streaming bit rate settings jointly in a multiregion multidatacenter cloud gaming system, which aims to minimize the overall service cost and ensure good-enough QoE. Different from these studies, our work focuses on minimizing both the inter-player delay among interacting players and the electricity costs of MCG providers, while providing the good-enough response delay to gamers.

III. PROBLEM FORMULATION

A. SYSTEM MODEL

Fig. 1 illustrates the system architecture of MCG platform, which includes multiple matchmaking servers, multiple gamer servers, S gamers dispersed at different regions and N geographically distributed data centers in a multielectricity-market environment. Each data center is equipped with M physical servers (PMs) and has a unique PUE value. Each physical server hosts several VMs. Each VM have diverse resource requirements, including CPU and GPU. The matchmaking server monitors system resources and implements the VM provisioning algorithm. More specifically, it is responsible for monitoring network delay, and locating and launching VMs for multiplayer game session. The information of interaction pattern among gamers can be obtained based on the locations of their avatars in the game scene. Each VM can handle the rendering and streaming workloads of up to k gamers.

B. DELAY MODEL

The QoE of game players is sensitive to response delay of MCG. Response delay mainly consists of network delay, processing delay and playout delay [13]. Network delay is essentially the network round-trip time, which can be measured by tools such as Ping. Processing delay represents the time needed by VM to process a player's input and generate the game frames, which accounts for 30% of the response delay. Playout delay is the time for the client to decode and play the received game frames, which only occupies 10% of processing delay. Because playout delay is usually fixed and occurs at the client side, we do not consider it in our model for the sake of brevity. Therefore, the response delay $RD_{p,i,j}$ of a gamer p connected to a VM running on PM j at data center i is calculated as follows:

$$RD_{p,i,j} = ND_{p,i} + PD_{i,j} + CD_i \tag{1}$$

where $ND_{p,i}$ represents the network delay between gamer p and data center i and CD_i represents the network delay between data center i and the remoting game server. $PD_{i,j}$ is the processing delay of a VM running on PM j at data center i and is defined by

$$PD_{i,j} = \alpha_1 / (1 + e^{-\alpha_2 \cdot v_{i,j} + \alpha_3}) \tag{2}$$

where α_1, α_2 and α_3 are model parameters, and $v_{i,j}$ is the number of VMs running on physical server j at data center i. This sigmoid function has been adopted by other works such [7] and [11].

C. ELECTRICITY COST MODEL

For geo-distributed data centers, the electricity cost is related to the electricity price, the amount of power consumption and the PUE. Therefore, the electricity cost C_i of a data center i can be expressed as

$$C_i = PUE_i \cdot [\sum_{j=1}^{M} P_{i,j}] \cdot b_i \tag{3}$$

where PUE_i denotes the energy efficiency of a data center i and its value is defined as a ratio of the total amount of power consumed by the entire datacenter facility (including cooling, lighting, etc.) over the power delivered to the computing equipment. Ideally, all power entering the data center is being used to power the computing equipment (PUE = 1). On average, conventional data centers have a PUE of 1.7, whereas leading industry datacenter facilities have a PUE of 1.18 [14].

In (3), b_i represents the electricity price at data center i and $P_{i,j}$ denotes the power consumed by a server j at data center i. Some research show that CPU and GPU are the major components consuming most of the system's power. In order to save power, servers are put into low-power sleep

modes when they are idle. Because today's server consumes very little power when in sleep mode, we neglect the power consumption during sleep mode. Finally, the server power consumption can be modeled as

$$P_{i,j} = \begin{cases} 0, & \textit{when being sleep mode} \\ P_{i,j}^{CPU} + P_{i,j}^{GPU} + P_{i,j}^{oth}, & \textit{otherwise} \end{cases} \quad (4)$$

where $P_{i,j}^{oth}$ denotes the power consumption of all components except CPU and GPU. $P_{i,j}^{CPU}$ and $P_{i,j}^{GPU}$ represent CPU and GPU power consumption, respectively. Inspired by the linear model in [15], [16], we can define the power consumption of these hardware components as follows

$$P_{i,j}^{CPU} = P_{i,j}^{CPU\_idle} + \left(P_{i,j}^{CPU\_max} - P_{i,j}^{CPU_{idle}}\right) \cdot U_{i,j}^{CPU} \quad (5)$$

$$P_{i,j}^{GPU} = \mu_{i,j}^{3} \cdot f_{i,j}^{GPU} \cdot U_{i,j}^{GPU} + \mu_{i,j}^{2} \cdot f_{i,j}^{GPU} + \mu_{i,j}^{1} \cdot U_{i,j}^{GPU} + \mu_{i,j}^{0} \quad (6)$$

where $\mu_{i,j}^{0}, \mu_{i,j}^{1}, \mu_{i,j}^{2}$ and $\mu_{i,j}^{3}$ are training parameters. $P_{i,j}^{CPU\_idle}$ and $P_{i,j}^{CPU\_max}$ represent CPU idle power and CPU peak power, respectively. $f_{i,j}^{GPU}$ denotes the GPU frequency. $U_{i,j}^{CPU}$ and $U_{i,j}^{GPU}$ represent the utilization of CPU and GPU, respectively. We employ the sigmoid function revealed by Hong et al [11] to compute system resource utilization and thus their formulas are defined as follows.

$$U_{i,j}^{CPU} = \beta_1 / (1 + e^{-\beta_2 \cdot v_{i,j} + \beta_3}) \quad (7)$$

$$U_{i,j}^{GPU} = \gamma_1 / (1 + e^{-\gamma_2 \cdot v_{i,j} + \gamma_3}) \quad (8)$$

where $\beta_1 - \beta_3$ and $\gamma_1 - \gamma_3$ are training parameters.

D. MULTIOBJECTIVE OPTIMIZATION MODEL

Let G be the set of gamers, E be the set of data centers and F_i be the set of servers in data center i. Let R^{CPU} and R^{GPU} be the CPU and GPU demand of each VM, respectively. Let $T_{i,j}^{CPU}$, and $T_{i,j}^{GPU}$ be the CPU and GPU capacity of each PM, respectively. The interact state between gamer i and gamer j is denoted by $I_{i,j}$. The binary variable $x_{p,i,j}$ indicates whether a gamer p is assigned to a server j at data center i. Our objective is to minimize both the inter-player delay among interacting players and the electricity costs of MCG providers, while maintaining the good-enough response delay. The VM provisioning problem can therefore be formulated as:

$$\textbf{Minimize} \ \max_{p,q \in G} \left\{ \left| D_p - D_q \right| \cdot I_{i,j} \right\} \quad (9)$$

$$\textbf{Minimize} \ \sum_{i=1}^{N} C_i \quad (10)$$

$$v_{i,j} \cdot R^{CPU} \le T_{i,j}^{CPU} \quad \forall i \in E, \ \forall j \in F_i \quad (11)$$

$$v_{i,j} \cdot R^{GPU} \le T_{i,j}^{GPU} \quad \forall i \in E, \ \forall j \in F_i \quad (12)$$

$$v_{i,j} = \lceil L_{i,j} / k \rceil \quad \forall i \in E, \ \forall j \in F_i \quad (13)$$

$$L_{i,j} = \sum_{p=1}^{S} x_{p,i,j} \quad \forall i \in E, \ \forall j \in F_i \quad (14)$$

$$\sum_{i=1}^{N} \sum_{j=1}^{M} x_{p,i,j} = 1 \quad \forall p \in G \quad (15)$$

$$D_p = \sum_{i=1}^{N} \sum_{j=1}^{M} (x_{p,i,j} \cdot RD_{p,i,j}) \quad \forall p \in G \quad (16)$$

$$D_p \le D^* \quad \forall p \in G \quad (17)$$

$$x_{p,i,j} \in \{0, 1\} \quad \forall p \in G, \ \forall i \in E, \ \forall j \in F_i \quad (18)$$

Constraints (11)-(12) avoid the resources used by VMs exceed the capability of the physical server. Constraint (13) derives the total number $v_{i,j}$ of VMs running on each server. Constraint (14) derives the total number $L_{i,j}$ of gamers allocated to each physical server. Constraint (15) ensures that each gamer is served by one physical server. Constraint (16) derives the response delay D_p of each gamer. Constraint (17) ensures that the response delay perceived by all players does not exceed the threshold D^*. Constraint (18) defines the domain of the variables of the problem. Because the proposed VM provisioning problem is NP-hard [3], it is typically impractical to make a complete enumeration of all possible solutions to find the best solutions when the amount of gamers increases to a large scale. The grey wolf algorithm has advantages of high convergence speed, few parameters and ease of implementation and it has been demonstrated to be superior or competitive to other classical metaheuristics such as genetic algorithm, particle swarm optimization algorithm and ant colony optimization algorithm. Therefore, the later section will show how to apply a grey wolf algorithm to efficiently search for good solutions in large solution spaces.

IV. THE ORIGINAL GREY WOLF ALGORITHM

Grey Wolf algorithm (GWA) is a recently developed metaheuristic search algorithm inspired by the leadership hierarchy and hunting mechanism of grey wolves in nature [17]. Grey wolves tend to live in packs. Wolf pack sizes is an average of 5–12 wolves. Fig. 2 describes the social hierarchy of the grey wolves in nature. The top of the hierarchy is occupied by alpha (α) wolf which is known as leaders of the pack. The α wolf is responsible for making decisions in a pack. The next level of hierarchy after α wolf is occupied by beta (β) wolf which helps α wolf in decision making and takes charge of the pack in absence of α wolf. The last level of the hierarchy is comprised of omega (ω) wolves which submit to all the other dominant wolves and play a role of scapegoats in the pack. The third level of the hierarchy is occupied by delta (δ) wolf which is in charge of scouting to protect and guarantee the safety of the pack. δ wolf respects to α wolf and β wolf, but dominates ω wolf. In addition to the social hierarchy of wolves, cooperative hunting is another important social behavior of grey wolves. Hunting process of grey wolves mainly consists of four phases: searching for the prey; approaching the prey; encircling the prey and attacking towards the prey.

In the mathematical model of the social hierarchy for the GWA, the fittest solution is considered as the α wolf while the second and third best solutions are termed as β and δ wolfs, respectively. The remaining solutions are assumed to be ω wolves. Hunting is guided by the α, β and δ wolfs while ω

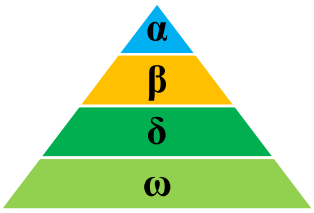

FIGURE 2. Social hierarchy of the grey wolves.

wolfs iteratively improved their location by following these three wolves.

During the hunting, the encircling behavior of grey wolves is mathematically modeled by using the following equations:

$$\vec{D} = \left|\vec{C} \cdot \overrightarrow{X_p}(t) - \vec{X}(t)\right| \tag{19}$$

$$\vec{X}(t+1) = \overrightarrow{X_p}(t) - \vec{A} \cdot \vec{D} \tag{20}$$

where t denotes the current iteration, $\vec{X}$ and $\overrightarrow{X_p}$ indicate the position vector of gray wolf and prey respectively. $\vec{A}$ and $\vec{C}$ denote coefficient vectors which are calculated as follows:

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a} \tag{21}$$

$$\vec{C} = 2\vec{r_2} \tag{22}$$

where $\vec{r_1}$ and $\vec{r_2}$ are random vectors within the range [0,1]. Approaching a prey is modeled by linearly decreasing the elements of $\vec{a}$ from 2 to 0 over the course of iteration. The vector $\vec{A}$ is utilized to balance exploration and exploitation. When $\left|\vec{A}\right| > 1$, the grey wolves are obliged to diverge from the prey to search for better prey, which stands for an exploration process. When $\left|\vec{A}\right| < 1$, the grey wolves attack towards the prey, which stands for an exploitation process.

The hunting process is usually directed by the α wolf while the β and δ wolfs can occasionally participate in hunting. In the mathematical model of hunting behavior for the GWA, the α, β and δ wolfs are supposed to have better knowledge about the potential position of prey and thus their positions are employed to update the positions of all the other (ω) wolves as shown in (23).

$$\vec{X}(t+1) = (\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3})/3 \tag{23}$$

where $\overrightarrow{X_1}$, $\overrightarrow{X_2}$ and $\overrightarrow{X_3}$ are calculated as in (24), (25) and (26) respectively.

$$\overrightarrow{X_1} = \left|\overrightarrow{X_\alpha} - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha}\right| \tag{24}$$

$$\overrightarrow{X_2} = \left|\overrightarrow{X_\beta} - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta}\right| \tag{25}$$

$$\overrightarrow{X_3} = \left|\overrightarrow{X_\delta} - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta}\right| \tag{26}$$

where $\overrightarrow{X_\alpha}$, $\overrightarrow{X_\beta}$ and $\overrightarrow{X_\delta}$ are the first three best solutions in the population at a given iteration t. $\overrightarrow{A_1}$, $\overrightarrow{A_2}$ and $\overrightarrow{A_3}$ are calculated using (21). $\overrightarrow{D_\alpha}$, $\overrightarrow{D_\beta}$ and $\overrightarrow{D_\delta}$ are calculated using (27), (28) and (29) respectively.

$$\overrightarrow{D_\alpha} = \left|\overrightarrow{C_1} \cdot \overrightarrow{X_\alpha} - \vec{X}\right| \tag{27}$$

$$\overrightarrow{D_\beta} = \left|\overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \vec{X}\right| \tag{28}$$

$$\overrightarrow{D_\delta} = \left|\overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \vec{X}\right| \tag{29}$$

where $\overrightarrow{C_1}$, $\overrightarrow{C_2}$ and $\overrightarrow{C_3}$ are calculated using (22).

V. THE PROPOSED MGWAM

The original GWA is designed mainly to handle the single objective optimization problem in the continuous domain. However, the VM provisioning problem for MCG is a discrete multiobjective combinatorial optimization problem. In order to extend the GWA algorithm for single objective optimization to solve multiobjective problems, we propose an improved multiobjective GWA algorithm to solve the VM provisioning problem for MCG. The basic steps of the proposed MGWAM algorithm are described as follows:

1. Initialize the parameters of the MGWAM algorithm
2. Generate the initial population (see Section V Part B for details)
3. Choose three best solutions from the population and set as $\overrightarrow{X_\alpha}$, $\overrightarrow{X_\beta}$ and $\overrightarrow{X_\delta}$ respectively (see Section V Part C for details).
4. Update the position of the individual by (23) (when the value computed by (23) exceed its predetermined boundaries, it is set to its boundaries.)
5. Apply the mutation operator to each individual wolf in the population with an adaptive mutation probability (see Section V Part D for details)
6. Update the population according to fast non-dominated sorting and crowding distance (see Section V Part E for details)
7. If the maximum number of iterations is reached, stop this algorithm. Otherwise, return to Step 3

The key components of the proposed MGWAM algorithm are the encoding and decoding schemes, the population initialization, the social hierarchy of grey wolves, the adaptive mutation operator and the population update strategy. The following subsections give a detailed description of these components.

A. ENCODING AND DECODING

A suitable encoding and decoding scheme can affect the performance of the MGWAM algorithm. Our goal is to select appropriate PM for each player to minimize both the inter-player delay among interacting players and the electricity costs of MCG providers while providing the good-enough response delay to gamers. To define the encoding of the problem, we need to establish the meaning and dimension of the grey wolf individual. A grey wolf individual in the proposed algorithm is illustrated as a row vector with continuous values. The dimensions of the vector are equal to the number of gamers in cloud gaming. The value of each element in the vector is a real number between *0* and the number *NP* of PMs in all the data centers. We index all PMs from different data centers using integers from *0* to *NP-1*. The core idea of the original GWA is to search the optimal solution by updating

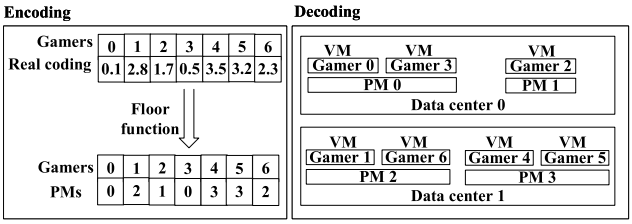

FIGURE 3. Encoding and Decoding.

the position of the grey wolf individual which is a continuous variable. However, the VM provisioning problem for MCG is a typical discrete problem. Therefore, it is a very important issue to solve the VM provisioning problem by mapping the continuous position of the grey wolf individual to the discrete assignment of players in MCG. In this paper, we adopt the floor function as a mechanism of transferring a continuous representation to an integer representation. The integer value corresponds to an index of PM and represents the PM serving the gamer defined by that element location. At the evolution stage, grey wolf individuals in continuous representation are used to search for the non-dominated solutions. At the decoding stage, grey wolf individuals in integer representation are used to determine the number of players allocated to each PM, the number of VMs running on each PM and the number of active PMs in each data center. As an example, Fig. 3 shows both the encoding scheme and the decoding scheme for the VM provisioning problem using 7 gamers, VM capacity of one gamer and 4 PMs dispersed at 2 different data centers.

B. POPULATION INITIALIZATION

In the VM provisioning problem, the search space of possible solutions is typically huge, especially when the amount of gamers increases to a large scale, which could cause the MGWAM algorithm extremely slow to converge. In order to accelerate the search progress, the initial population in our algorithm is composed of the individuals generated by different initialization approaches. Assuming that the population size is *SN*, these individuals include

- An initial individual generated by a QoE-aware heuristic algorithm, which is treated as the VM provisioning with high gaming QoE.
- An initial individual generated by a cost-aware heuristic algorithm, which is treated as the VM provisioning with low electricity cost.
- *SN-2* randomly generated individual.

The QoE-aware heuristic algorithm assigns every gamer to a server with the minimal inter-player delay among all of its eligible servers. The cost-aware heuristic algorithm allocates every player to a server in the data center with the minimum combined cost among all of its eligible datacenters. The combined cost E_d associated with each datacenter d is defined as $E_d = \mathrm{PUE}_d/PUE_{max} + b_i/b_{max}$, where PUE_{max} and b_{max} represent the maximum value of the PUE and the electricity price among all data centers respectively. We utilize these two heuristic-generated individuals to approximate two endpoints of the Pareto front. Other initial individuals are generated by randomly choosing a real number from (0, *NP*) for the element of each encoding.

C. SOCIAL HIERARCHY OF GREY WOLVES

Because of the presence of constraints, each gray wolf individual can be either feasible solution or infeasible solution. It is difficult to directly determine the α, β and δ wolfs based on Pareto dominance relationship. The constraint-domination principle proposed by Deb et al. [18] is used to handle the constraints. We utilize the fast non-dominated sorting approach to divide the population into several level ranks. All feasible solutions are ranked according to Pareto dominance relationship while all infeasible solutions are ranked according to the constraint violation value. Any feasible solution has a better rank than any infeasible solution. When multiple individuals are having the same rank, the crowding distance is employed to assess the quality of gray wolf individual.

In order to maintain an appropriate balance between exploration and exploitation, the α, β and δ wolfs are determined using a mixed operator based on binary tournament and elitist selection. In binary tournament selection, a pair of individuals are chosen at random from the population. The individual with the lower front is chosen if they are from different fronts. The solution with the higher crowding distance is chosen if they are from the same front. The number (*BTN*) of individuals selected by binary tournament is dynamic and it is defined as follows.

$$BTN = 3(1 - PR(n)^2) \tag{30}$$

where $PR(n)$ represents the progress rate in the evolutionary search and it is given by

$$PR\,(n) = NewSlo(n)\big/AllSlo(n) \tag{31}$$

where $NewSlo(n)$ denotes the number of new non-dominated solutions discovered in generation n, and $AllSlo(n)$ denotes the total number of non-dominated solutions in generation n. At the same time, the number of individuals chosen by elitist selection is also dynamic and it is given by $3 - BTN$. Elitist selection is based on individual's rank and crowding distance.

The rationale behind the mixed operator is intuitive. When $PR(n)$ is small, it means that either the generated Pareto front is approaching the true front or the search process is not discovering new solutions and wolf packs need to explore unknown areas of the search space. When $PR(n)$ is large, it means that the new solutions are being discovered and wolf packs need to perform exploitation in neighborhood of the known good solution.

D. ADAPTIVE MUTATION OPERATOR

In order to avoid losing diversity of wolf packs and reduce the risk of trapping in local optimum, an adaptive mutation operator is introduced in evolution process. The well-known inversion mutation is applied to each individual wolf in the

FIGURE 4. Inversion mutation.

population with an adaptive mutation probability. The inversion mutation is performed by choosing two positions within a solution at random and then inverts the elements between these two positions. Fig. 4 shows how the inversion mutation operates. Since the mutation probability has an important effect on the performance of the MGWAM algorithm, the inversion mutation operator adapts the mutation probability with time along the whole evolution process to maintain a balance between exploitation and exploration. The mutation probability P_m in the MGWAM algorithm is calculated using the following equation.

$$P_m = \begin{cases} h_1 \cdot \left[1 - \left(n/maxgen\right)^2\right], & 0 \le n \le w \\ h_2 \cdot \left[1 - \left(n/maxgen\right)^2\right], & w \le n \le maxgen \end{cases} \quad (32)$$

where n denotes the current iteration, $maxgen$ represents the maximum number of iterations carried out, and $w(0 \le w \le maxgen)$, $h_1(0 \le h_1 \le 1)$ and $h_2(0 \le h_2 \le 1)$ are parameters which are selected depending on the problem. In this paper, w, h_1 and h_2 are set as $maxgen/4$, 0.8 and 0.05, respectively. Unlike many other adaptive mutation operators where mutation probability decreases gradually along the evolution process, the mutation operator that we adopted focuses on producing a diverse set of solutions in the initial stage and then improving them in the later stage.

E. POPULATION UPDATE

Given an initial parent population P_t with SN individuals, the MGWAM algorithm proceeds by applying the social hierarchy, hunting and mutation operator to compute an offspring population Q_t composed of SN individuals. In order to preserve the better solutions appeared during the evolution process, the parent and offspring population after evolution are combined into a population R_t with $2{*}SN$ individuals. If the number N_f of the feasible solutions in R_t is less than SN, all feasible solutions are put into the next population and the remaining (SN-N_f) individuals are chosen from the infeasible solutions according to the constraint violation value. Otherwise, SN feasible solutions are selected from P_t as the next population according to fast non-dominated sorting and crowding distance.

VI. PERFORMANCE EVALUATION

A. EXPERIMENT SETUP

The performance of the proposed MGWAM algorithm was evaluated by conducting simulation experiments using an extension of CloudSim [19] called the GPUCloudSim toolkit [20], which is a modern framework aimed for modeling and simulating GPU-enabled data centers in cloud computing environments. It provides multilevel scheduling and provisioning of GPU resources and also adds functionalities required to support the analysis of GPU virtualization and power consumption overheads. Our experiments were conducted for two geo-distributed data center configurations containing ten and sixteen data centers. Locations of the data centers in the two configurations are chosen from major cities in the continental United States. Table 1 shows the price of electricity at different locations [21] and the PUE values of different data centers [22]. Each data center is composed of 10 homogeneous PMs with two dimensional resource capacities: CPU and GPU, and for each PM, the total resource capacities of these resources are set as 6200 MIPS and 32 vGPU with 750MHz, respectively. Each PM can host multiple VMs and each VM requires one CPU core with 1000 MIPS and 4 vGPU with 750 MHz.

TABLE 1. Data center configuration.

Data Center Location	PUE	Electricity Price ($/kW)
New York	1.69	11.40
Chicago	1.81	14.54
Denver	1.51	3.82
LA	3.03	8.25
Atlanta	2.32	7.75
Baltimore	1.47	13.01
Dallas	1.35	8.91
Indianapolis	1.69	3.29
Detroit	2.12	8.11
Las Vegas	2.63	10.25
San Francisco	1.58	3.84
Seattle	2.38	6.39
Tampa	2.04	11.88
Kansas City	1.42	6.20
Oklahoma City	1.69	10.57
Nashville	1.49	5.09

Table 2 summarizes the values of the parameters used in the various formulations for the electricity cost and inter-player delay estimation presented in Section III. Specifically, the values of the coefficients for computing the processing delay, CPU utilization and GPU utilization are taken as reported in a previous experimental study [11]. The response delay threshold is set to 500ms as used by Chen et al. [7]. The values of the coefficients for computing GPU, CPU and PM power consumption are taken as reported by Guan et al. [15] and Meisner et al. [23].

We derive the interaction pattern between players and the distribution of players from the avatar history dataset of World of Warcraft (WoW) collected by Lee et al. [24]. The network delay between players and data centers varies from 5 to 200ms, while the network delay between data centers and remote game servers from 10 to 50ms [7].

The settings for various parameters in MGWAM have a direct effect on the algorithm performance. Appropriate parameter values were determined on the basis of preliminary

TABLE 2. The related parameters used in the simulation.

Constants	Meaning
$\mu^0_{i,j} = 21.44$, $\mu^1_{i,j} = 0.09787$, $\mu^2_{i,j} = 0.00796$, $\mu^3_{i,j} = 0.0002438$	Coefficients for computing GPU power consumption
α_1 =0.6498 α_2 =0.5185 α_3 =2.6468	Coefficients for computing the processing delay of VM
β_1 = 0.9222 β_2 =0.8243 β_3 =1.2884	Coefficients for computing CPU utilization of PM
γ_1 = 0.8142 γ_2 =0.4259 γ_3 =-0.0831	Coefficients for computing GPU utilization of PM
$P^{CPU_idle}_{i,j} = 12$W $P^{CPU_max}_{i,j} = 80$W	Coefficients for computing CPU power consumption
$P^{oth}_{i,j} = 60$ W	Coefficient for computing PM power consumption
$D^* = 500$ms	Response delay threshold

experiments. The final parameter settings were as follows: the population size is 100 and the maximum number of iterations is 50.

To evaluate the algorithm performance, we take into account different problem sizes from small to large scales. The problem size is defined as a three-tuple (*k, s, t*), where *k* is the number of gamers served by each VM, *s* is the number of data centers, and *t* is the number of gamers.

B. COMPARISON OF MGWAM WITH TWO SINGLE-OBJECTIVE APPROACHES

In the first set of simulation experiments, we compared the proposed approach with two single-objective algorithms, an IDO algorithm [7] and a LCC approach. IDO is a two-phase heuristic algorithm that aims to minimize the inter-player delay between interacting players while maintaining good-enough response delay experienced by players. LCC is an extension of the LCW proposed by Deng et al. [3] with the aim of minimize the electricity cost while achieving just-good-enough response delay of gaming, in which every player is allocated to the data center with the lowest capacity wastage. if a gamer has more than one eligible datacenters providing the same lowest capacity wastage, the gamer is assigned to the one with the lowest combined cost in terms of PUE and electricity price. The optimal result obtained by the proposed algorithm is usually a set of non-dominated solutions, where each objective function value of any non-dominated solution can only be improved by degrading at least one of its other objective function values. Because of the lack of information about the preferences of objectives, we utilize a Fuzzy-based approach [25] to generate the best compromised solution from the obtained Pareto set. For each objective function f_k, a linear membership function μ_k is defined as follows:

$$\mu_k = \begin{cases} 1 & f_k \le f_k^{min} \\ (f_k^{max} - f_k)/(f_k^{max} - f_k^{min}) & f_k^{min} < f_k < f_k^{max} \\ 0 & f_k \ge f_k^{max} \end{cases} \quad (33)$$

where f_k^{min} and f_k^{max} represent the maximum and minimum values of the kth objective function among all non-dominated solutions, respectively. Correspondingly, the normalized membership function μ^i is calculated for each non-dominated solution as

$$\mu^i = \sum_{k=1}^{OB} \mu_k^i \Big/ \sum_{j=1}^{ND} \sum_{k=1}^{OB} \mu_k^i \quad (34)$$

where *OB* and *ND* represent the number of objectives functions and non-dominated solutions, respectively. The solution with the maximum membership μ^i is chosen as the best compromise solution.

Figs. 5, 6 and 7 show the maximal inter-player delay, the electricity cost and the cumulative distribution function (CDF) of response delay provided by three algorithms for various problem sizes, respectively. From these figure, we have the following results:

1. Compared to IDO, LCC performs better in terms of the electricity cost but worse in terms of the maximum inter-player delay; the reason is that it only optimizes the electricity cost and does not take the difference of response delays among players into account.
2. Compared to LCC, IDO performs better in terms of the maximum inter-player delay but worse in terms of the electricity cost; the reason is that it only optimizes the inter-player delay among players, and it does not consider the electricity cost.
3. MGWAM performs best in terms of the maximum inter-player delay and the electricity cost; the reason is that it simultaneously optimizes the inter-player delay among interacting players and the electricity costs of MCG providers.
4. All the three algorithms can maintain the response delay experienced by players below a tolerable threshold. Among these algorithms, IDO has the minimum boundary of the response delay, LCC has the medium boundary of the response delay, and MGWAM has the maximum boundary of the response delay.

C. COMPARISON OF MGWAM WITH ITS VARIANTS

To evaluate the efficiency of the improved initialization strategy, the improved social hierarchy strategy and the adaptive mutation strategy in MGWAM, we compare MGWAM with its variants, including $\text{MGWAM}_{\text{ran}}$, $\text{MGWAM}_{\text{btou}}$, and $\text{MGWAM}_{\text{fixm}}$. In the second set of simulation experiments, $\text{MGWAM}_{\text{ran}}$, $\text{MGWAM}_{\text{btou}}$ and $\text{MGWAM}_{\text{fixm}}$ denote the algorithm with the random population initialization, the algorithm with three best solutions selected using the binary tournament, and the algorithm with a fixed mutation rate 0.5, repectively. We use the hypervolume (*HV*) measure [26]

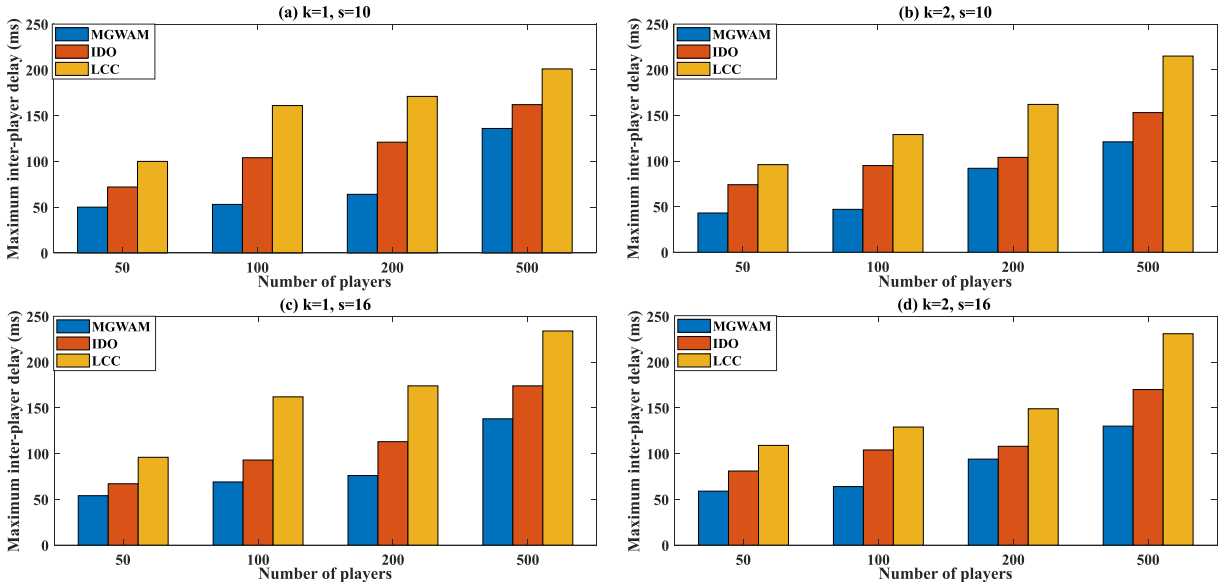

FIGURE 5. Maximum inter-player delays of three algorithms for different problem sizes.

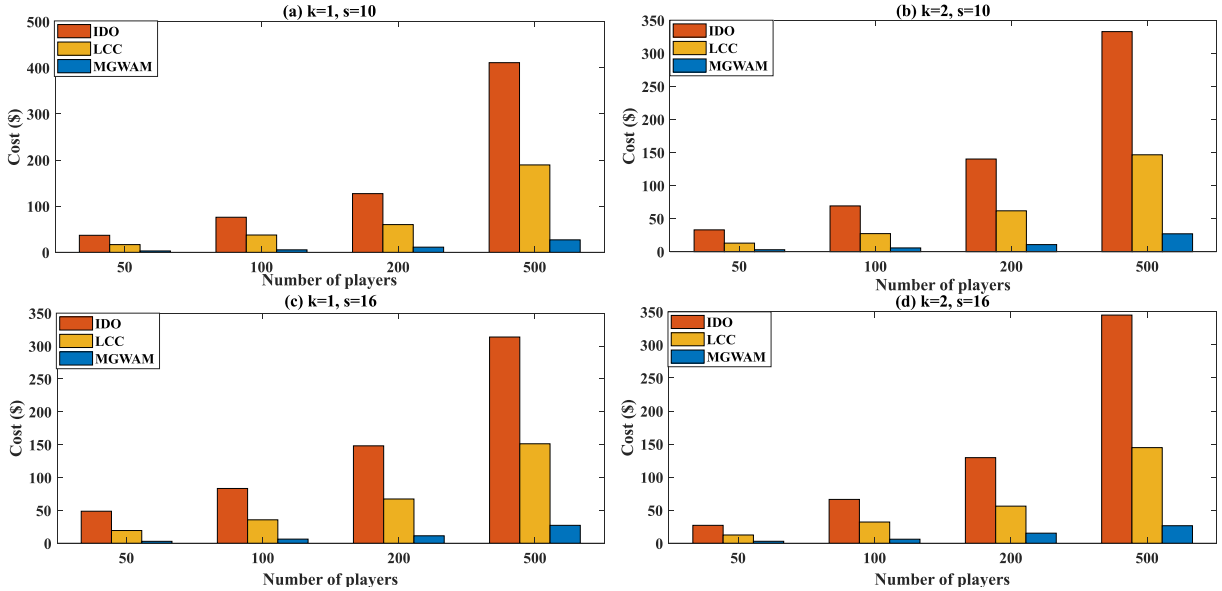

FIGURE 6. Costs of three algorithms for different problem sizes.

to assess the performance of four multiobjective optimization algorithms. The *HV* metric is defined as the volume of the dominated portion of the objective space relative to a reference point. Mathematically, for each non-dominated solution $i \in X$ found by the algorithms, a hypercube hv_i is established with the reference point and the solution i as the diagonal corners of the hypercube. The reference point can be obtained by constructing a vector of worst objective function values. The *HV* of the non-dominated solution set X is then calculated as

$$HV = \bigcup_{i=1}^{|X|} hv_i \tag{35}$$

A larger HV value is preferable as it reveals that the obtained solutions are close to the true Pareto front and also has a good distribution. For the purpose of comparison between algorithms, we use the non-dominated solutions in the merging solutions of algorithms over 10 runs as an approximation of the true Pareto front and then normalize the obtained solutions by dividing their upper bounds of the approximation. After all the solutions are normalized, a reference point (1,1) is used in the calculations of HV.

Fig. 8 shows the non-dominated solutions produced by four algorithms for various problem sizes. As can be seen

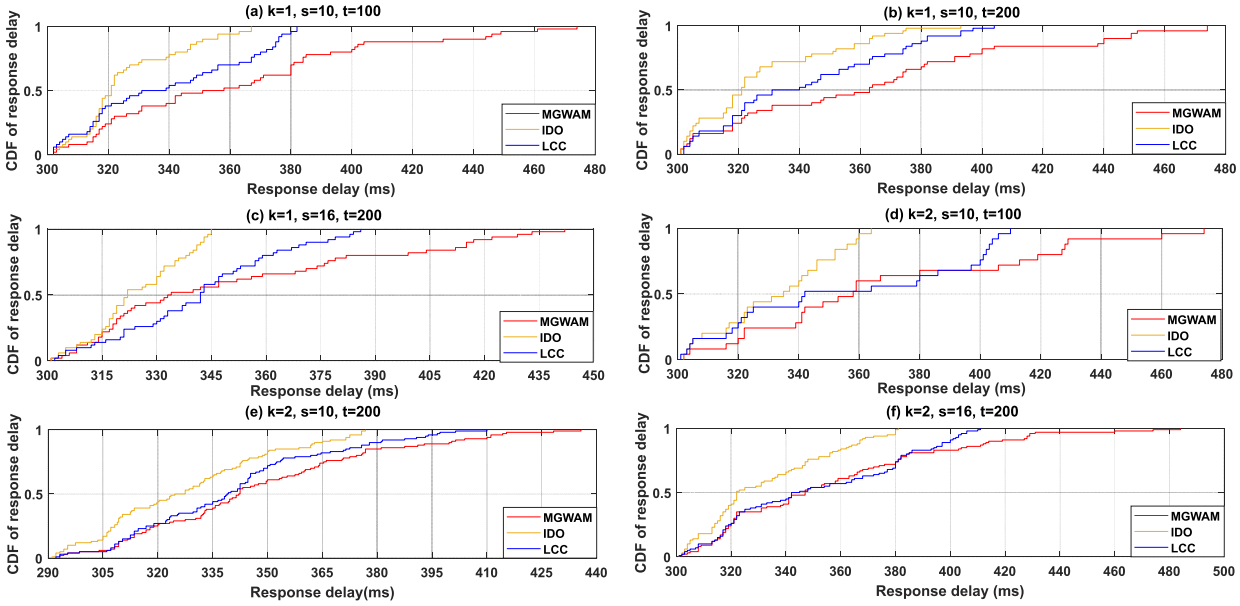

FIGURE 7. CDF of response delay under different problem sizes.

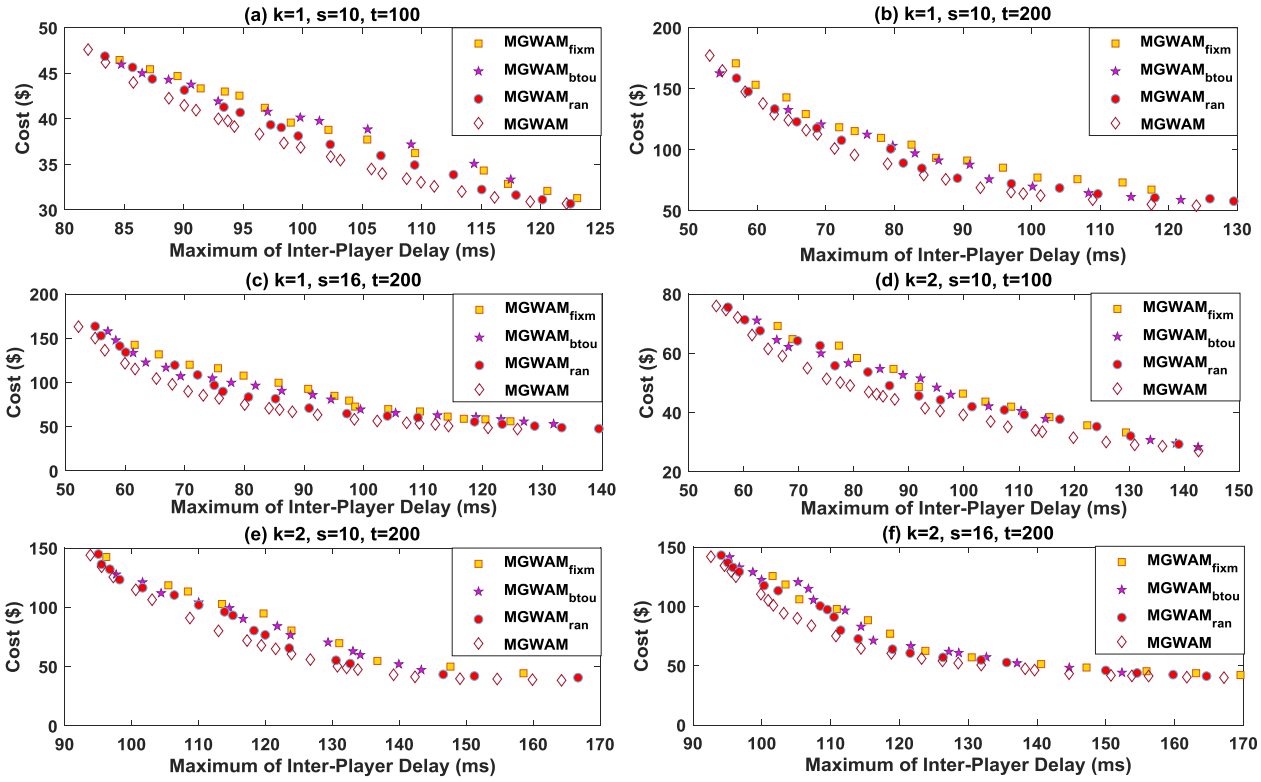

FIGURE 8. Non-dominated solutions obtained by four algorithms under different problem sizes.

in the figure, the Pareto fronts obtained by our MGWAM are superior to those obtained by the comparative algorithms. The better performance of our MGWAM compared to the comparative algorithms is also confirmed by a larger value of *HV* shown in Table 3. For a more intuitive comparison, the table also shows the performance improvement *IM* (*oth*) of MGWAM over the comparative algorithms which is defined as follows

$$IM\ (oth) = \frac{HV_{our} - HV_{oth}}{HV_{oth}} \tag{36}$$

TABLE 3. Comparison of the HV metric for three algorithms under different problem sizes.

Problem sizes	MGWAM	$MGWAM_{btou}$	$MGWAM_{fixm}$	$MGWAM_{ran}$	$IM(MGWAM_{btou})$	$IM(MGWAM_{fixm})$	$IM(MGWAM_{ran})$
(1,10,100)	0.618	0.470	0.485	0.539	31.49%	27.42%	14.66%
(1,10,200)	0.715	0.604	0.635	0.696	18.38%	12.59%	2.73%
(1,16,200)	0.724	0.673	0.572	0.709	7.58%	26.57%	2.12%
(2,10,100)	0.661	0.558	0.533	0.588	18.46%	24.01%	12.42%
(2,10,200)	0.721	0.521	0.570	0.696	38.39%	26.49%	3.59%
(2,16,200)	0.758	0.642	0.705	0.710	18.07%	7.52 %	6.76%

where HV_{oth} denotes the HV value achieved by the comparative algorithms, i.e. $MGWAM_{ran}$, $MGWAM_{btou}$, or $MGWAM_{fixm}$, and HV_{our} is the HV value achieved by MGWAM. From the table, we can also find that our MGWAM always have a better performance than $MGWAM_{ran}$, $MGWAM_{btou}$, and $MGWAM_{fixm}$ regardless of which problem sizes used in the experiment. The performance improvements achieved by our MGWAM over $MGWAM_{ran}$, $MGWAM_{btou}$, and $MGWAM_{fixm}$ can be up to 14.66%, 38.39% and 27.42%, respectively. The major reasons for the superiority performance of MGWAM are as follows: First, the search process in MGWAM is guided by three good solutions selected based on binary tournament and elitist selection, which improve the quality of population and maintain diversity of population. Second, the search operator of GWA and the adaptive mutation operator is combined to diversify the search to unvisited areas of the solution space, which give the MGWAM a better chance to explore the search space efficiently. Third, the mixed population containing both random and heuristic initial solutions can lead to high-quality solutions in a relatively small number of generations.

VII. CONCLUSION

In this paper, we model the VM provisioning problem for MCG in geographically distributed data centers as a constrained multiobjective optimization problem and present an improved grey wolf algorithm to solve the problem. The performance of our proposed algorithm is assessed through extensive simulation based on the real-world parameters. The results show that, compared with other alternatives, our proposed algorithm performs better in terms of the maximum inter-player delay and the electricity cost, while provide the good-enough response delay to gamers.

REFERENCES

[1] (May 2019). *Call of Duty: Black Ops 4*. [Online]. Available: https://www.Callofdu-ty.com/cn/zh/blackops4

[2] (2019). *Global Cloud Gaming Market Worth USD 6,944 Million By 2026*. [Online]. Available: http://zmrnewsmagazine.com/2019/04/05/global-cloud-gaming-market-worth-usd-6944-million-by-2026/

[3] Y. Deng, Y. Li, R. Seet, X. Tang, and W. Cai, "The server allocation problem for session-based multiplayer cloud gaming," *IEEE Trans. Multimedia*, vol. 20, no. 5, pp. 1233–1245, May 2018. doi: 10.1109/TMM.2017.2760621.

[4] T. Henderson, "Latency and user behaviour on a multiplayer game server," in *Proc. Int. Workshop Netw. Group Commun.*, London, U.K., 2001, pp. 1–13.

[5] M. Amiri, H. Al Osman, S. Shirmohammadi, and M. Abdallah, "Toward delay-efficient game-aware data centers for cloud gaming," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 12, no. 5s, pp. 1–19, Dec. 2016. doi: 10.1145/2983639.

[6] J. Wu, B. Cheng, Y. Yang, M. Wang, and J. Chen, "Delay-aware quality optimization in cloud-assisted video streaming system," *ACM Multimedia Comput. Commun. Appl.*, vol. 14, no. 1, Jan. 2018, Art. no. 4. doi: 10.1145/3152116.

[7] Y. Chen, J. Liu, and Y. Cui, "Inter-player Delay Optimization in multiplayer cloud gaming," in *Proc. IEEE 9th Int. Conf. Cloud Comput. (CLOUD)*, San Francisco, CA, USA, 2016, pp. 702–709.

[8] Y. Li, X. Tang, and W. Cai, "Play request dispatching for efficient virtual machine usage in cloud gaming," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 12, pp. 2052–2063, Dec. 2015. doi: 10.1109/TCSVT.2015.2450152.

[9] Q. Wang, M. M. Tan, X. Tang, and W. Cai, "Minimizing cost in IaaS clouds via scheduled instance reservation," in *Proc. Int. Conf. Distrib. Comput. Syst.*, Atlanta, GA, USA, 2017, pp. 1565–1574.

[10] Y. Li, Y. Deng, X. Tang, W. Cai, X. Liu, and G. Wang, "Cost-efficient server provisioning for cloud gaming," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 14, no. 3s, Aug. 2018, Art. no. 55. doi: 10.1145/3190838.

[11] H.-J. Hong, D.-Y. Chen, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Placing virtual machines to optimize cloud gaming experience," *IEEE Trans. Cloud Comput.*, vol. 3, no. 1, pp. 42–53, Jan. 2015. doi: 10.1109/TCC.2014.2338295.

[12] H. Tian, D. Wu, J. He, Y. Xu, and M. Chen, "On achieving cost-effective adaptive cloud gaming in geo-distributed data centers," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 12, pp. 2064–2077, Dec. 2015. doi: 10.1109/TCSVT.2015.2416563.

[13] K.-T. Chen, Y.-C. Chang, P.-H. Tseng, C.-Y. Huang, and C.-L. Lei, "Measuring the latency of cloud gaming systems," in *Proc. 19th ACM Int. Conf. Multimedia*, Scottsdale, AZ, USA, 2011, pp. 1269–1272.

[14] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: Research problems in data center networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 68–73, 2009. doi: 10.1145/1496091.1496103.

[15] H. Guan, J. Yao, Z. Qi, and R. Wang, "Energy-efficient SLA guarantees for virtualized GPU in cloud gaming," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 9, pp. 2434–2443, Sep. 2015. doi: 10.1109/TPDS.2014.2350499.

[16] R. Basmadjian, N. Ali, F. Niedermeier, H. de Meer, and G. Giuliani, "A methodology to predict the power consumption of servers in data centres," in *Proc. 2nd Int. Conf. Energy-Efficient Comput. Netw.*, New York, NY, USA, 2011, pp. 1–10.

[17] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014. doi: 10.1016/j.advengsoft.2013.12.007.

[18] T. Deb, A. Pratap, T. Meyarivan, and S. Agarwal, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002. doi: 10.1109/4235.996017.

[19] R. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Aug. 2011. doi: 10.1002/spe.995.

[20] A. Siavashi and M. Momtazpour, "GPUCloudSim: An extension of CloudSim for modeling and simulation of GPUs in cloud data centers," *J. Supercomput.*, vol. 75, pp. 2535–2561, May 2019. doi: 10.1007/s11227-018-2636-7.

[21] N. Hogade, S. Pasricha, H. J. Siegel, A. A. Maciejewski, M. A. Oxley, and E. Jonardi, "Minimizing energy costs for geographically distributed heterogeneous data centers," *IEEE Trans. Sustain. Comput.*, vol. 3, no. 4, pp. 318–331, Oct./Dec. 2018. doi: 10.1109/TSUSC.2018.2822674.
[22] S. Greenberg, E. Mills, B. Tschudi, and L. Berkeley, "Best practices for data centers: Lessons learned from benchmarking 22 data centers," in *Proc. ACEEE Summer Study Energy Efficiency Buildings Asilomar*, Pacific Grove, CA, USA, 2006, pp. 76–87.
[23] D. Meisner, B. T. Gold, and T. F. Wenisch, "PowerNap: Eliminating server idle power," in *Proc. 14th Int. Conf. Architectural Support Program. Lang. Oper. Syst.*, Washington, DC, USA, 2009, pp. 205–216.
[24] Y.-T. Lee, K.-T. Chen, Y.-M. Cheng, and C.-L. Lei, "World of warcraft avatar history dataset," in *Proc. 2nd Annu. ACM Conf. Multimedia Syst.*, San Jose, CA, USA, 2011, pp. 123–128.
[25] M. A. Abido and J. M. Bakhashwain, "Optimal VAR dispatch using a multiobjective evolutionary algorithm," *Int. J. Elect. Power Energy Syst.*, vol. 27, no. 1, pp. 13–20, Jan. 2005. doi: 10.1016/j.ijepes.2004.07.006.
[26] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999. doi: 10.1109/4235.797969.

YONGQIANG GAO received the Ph.D. degree from Shanghai Jiao Tong University (SJTU), Shanghai, China, in 2013. He is currently an Associate Professor with the College of Computer Science, Inner Mongolia University (IMU). His research interests include cloud computing, virtualization, and green computing.

LIN WANG received the B.S. degree from the Taiyuan University of Technology, in 2016. He is currently pursuing the M.S. degree with the College of Computer Science, Inner Mongolia University. His research interests include cloud gaming and artificial intelligence.

JIANTAO ZHOU received the Ph.D. degree from Tsinghua University, China, in 2005. She is currently a Professor and the Ph.D. Supervisor with the College of Computer Science, Inner Mongolia University. Her research interests include cloud computing and software engineering.