IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 18, 2018, accepted September 5, 2018, date of publication September 17, 2018, date of current version October 12, 2018.

*Digital Object Identifier 10.1109/ACCESS.2018.2870132*

# A New Adaptive Data Center Resource Provisioning Scheme Based on the Dual-Level Cooperative Game Approach

**SUNGWOOK KIM**
Department of Computer Science, Sogang University, Seoul 04107, South Korea
e-mail: swkim01@sogang.ac.kr

This work was supported by the Ministry of Science and ICT, South Korea, under the Information Technology Research Center Support Program under Grant IITP-2018-2018-0-01799 supervised by the Institute for Information Communications Technology Promotion.

**ABSTRACT** With the popularity of 5G network technology, cloud computing paradigm in data center (DC) networks has drawn increased attention. DC is a pool of computational and storage resources interconnected using a communication network. To handle the growing demands of cloud computing services, resources in DC should be used efficiently. In this paper, we design a new DC resource provisioning scheme based on the cooperative game theory. Using the matching game model and Mood value, we propose new resource allocation algorithms, which attempt to equalize users' satisfactions with respect to an effective distribution of the DC resource. Due to the competitive and coordinative DC environments, our proposed game approach can adaptively respond to the current cloud computing service requests. The main novelty possessed by our resource provisioning scheme is to capture the dynamics of DC network operations with the consideration of efficiency and fairness. Finally, we show the advantages of our proposed scheme through computer simulations. Comparing with other existing protocols, performance evaluations demonstrate that our game-based approach can outperform the existing state-of-the-art methods in terms of DC resource usability, service success ratio and fairness. In addition, we provide the guidance on the future research direction including other issues.

**INDEX TERMS** Matching game theory, cloud computing paradigm, mood value, data center, cooperative resource provisioning.

## I. INTRODUCTION

The advent of cloud computing paradigm has given rise to new and exciting prospects for 5G network technology. It can flexibly lease processing, storage, and network resources on-demand, according to users' temporal needs. Nowadays, cloud computing is evolving as everything as a service on the network service [1]. As cloud computing paradigm is becoming mainstream, data center (DC) must serve an ever-growing demand for computation and storage resources. The term 'data center' can be interpreted as an offsite storage center that consists of servers and other equipment needed to keep the stored data accessible both virtually and physically. Underpinning cloud computing is a DC infrastructure, which is maintained and managed by local as well as global operators and typically offers as-a-services to corporate and individual customers over the Internet [1]–[3].

DCs are the crucial part of any cloud computing service. Effective operation of DCs will not only help to reduce service costs but also to deliver high quality cloud services to customers. To efficiently exploit the DC infrastructure, the operation of DCs is becoming ever more challenging. One major problem for DC operations is lack of performance guarantee, which includes both resource limitations and unpredictable cloud service demands. To solve this problem, we should find the right balance between the conflicting aims of keeping service costs down, reducing energy consumption, improving quality of service (QoS), and enhance resource usability. However, it is a complex and difficult work under a dynamically changing DC environment [2], [4]–[6], [23], [24].

Virtualization is an enabling technique to facilitate the sharing of resources. To meet constantly growing cloud

service requirements, this technology bridges the foreground commercial usage and the background resource management on underlying DC infrastructures. To provide the adaptive DC capability, it is an essential step to logically isolate computing, storage, and network resources, and these resources should be freely configured on-demand. For the high flexibility, predictable performance, reliability and controllability, virtual resource pools using virtualization technology act cooperatively and collaborate with each other while fundamentally altering assumptions about the physical locations of resources [7], [8].

DC virtualization encompasses a broad range of tools, technologies and processes that enable a DC to operate and provide cloud computing services. It not only helps in optimal DC infrastructure utilization, but also in reducing DC capital and operational costs. As virtualization technology becomes the mainstream way in DCs, the effective and efficient placement of virtual machines (VMs) becomes an important issue. VMs, each of which acts like a real machine with an operating system, are created on underlying physical resources. Traditionally, VMs are used to partition and share DC resources while allowing the secure and isolated co-existence of multiple VMs on the same physical resources. VMs are well studied before the appearance of DCs, dated back to 1970's. However, traditional VM implementations have been considered in the relatively static environment, which is in stark contrast to the dynamic nature of DCs [7], [9].

Individual DC enables the coexistence of multiple VMs while allowing each VM to be independently implemented and managed with topology constraints. Therefore, the DC resources can be scheduled with fine-granularity, which improves resource utilization significantly. To meet service requirements such as lower cost, better scalability, and additional management flexibility, each VM runs independently with a proprietary DC resource. Therefore, accommodating a number of VM placements requires an efficient mapping of VMs onto the DCs. This VM placement is to select a suitable DC to deploy each newly-created VM in runtime. Recently, the VM placement process has been the basic key issue in DC virtualization and has a large solution space with a high computational complexity [9], [10].

Nowadays, cooperative game theory becomes a hot research topic and has received a generous concern. Cooperative games are games where groups of players may enforce cooperative behaviors. Therefore, players choose their strategies through a consensus decision-making process. For designing fair and efficient cooperative game strategies, some solution concepts have been proposed. In this study, we adopt the fundamental ideas of matching game and Mood value, and prove that they are very powerful tools for the VM placement and DC resource distribution problems. Under widely dynamic and diversified DC operations, key challenges of matching game and Mood value can satisfy certain reasonable principles while ensuring rational behaviors [11], [12].

Inspired by the above discussion, we have developed a new dual-level cooperative game model while ensuring good global properties. We take a novel step towards providing an effective tradeoff between global optimality and practicality. At the first-level, VMs find out the finest DC by considering its computation constraints. According to the matching game, we find an effective way to match each VM with a specific DC so that no unmatched pair can later find out that they can both do better by matching each other. Therefore, each VM can be placed on the most adaptable DC. At the second-level, each DC distributes its CPU capacity to its corresponding VMs using the Mood value. When placing VMs on a DC, different VMs require different amounts of CPU resource capacity to meet such requirements. If the available resource in the DC is not enough to fully satisfy VM demands, DC resource distribution becomes a challenging problem. By using the Mood value, which has been developed as a cooperative resource allocation rule, we attempt to equalize the satisfaction of each VM. With the respect of dual-level cooperative game approach, we can analyze and solve effectively the VM placement and DC resource distribution problems, going beyond existing approaches that do not explore the setting where a judicious mixture of coordination and collaboration is often advantageous in competitive environments.

### A. RELATED WORK

Considerable state-of-the-art research has been conducted on the design of resource provisioning schemes for DCs. Reference [13] investigates a new game theory based consolidation method of VMs in DCs with system constraints. First, this scheme predicts the future load values of resources to timely adjust the load for the higher degree of load balance. Second, physical resources are grouped according to the number of VMs and a pre-processing algorithm is used to decide VM migrations. Finally, the developed game based method selects the physical resources to get the optimal energy consumption. The main novelty of this scheme is to reduce energy consumption as well as balance loads without unnecessarily increasing the number of VM migrations.

The main idea in [2] is that a simplified model of the core scheduling issues should be taken into account during the VM placement. Usually, ignoring the scheduling of cores during VM placement leads to an over-simplification that may cause a suboptimal VM placement. As an ideal scheme, the approach in [2] identifies a constraint programming to formulate complex constraints, and defines the VM placement as a global optimization problem with a well-defined objective function. By incorporating different heuristics, it enables a balance between solution quality and running time. Finally, a simulation-based empirical analysis shows that the proposed approach can deliver significantly better results compared to a typical non-multicore-aware heuristic protocols [2].

Reference [14] proposes a new energy and QoS aware VM placement optimization algorithm to add the global QoS

guarantee while considering the VM placement optimization problem. To effectively solve the VM placement optimization problem, this scheme improves the particle swarm optimization technique by redefining its parameters and operators, and adopts a local fitness-first strategy to update the particle position by eliminating the assumption of server homogeneity. With a tradeoff between energy consumption and global QoS guarantee, the scheme in [14] designs a novel energy-aware virtual machine placement for a tree-like datacenter network, which is adopted by the commercialized DC. Experimental results show that the proposed approach can reduce energy consumption while still satisfying the global QoS guarantee [14].

This scheme [15] proposes a novel Markov chain model, which allows a system administrator to explicitly set a QoS goal. For a known stationary workload and a given state configuration, the control policy obtained from the Markov model optimally solves the host overload detection problem in the online setting by maximizing the mean intermigration time, while meeting the QoS goal. This underlying analytical model allows a derivation of an optimal randomized control policy for any known stationary workload and a given state configuration. And then, a new control algorithm is developed for the problem of host overload detection as a part of dynamic VM consolidation. In addition, this algorithm handles unknown nonstationary workloads using the multi-size sliding window workload estimation technique. The experimental study has shown that the proposed scheme is efficient in handling multiple mixed heterogeneous real-world workloads [15].

Authors in the work [16] mainly discuss the communication cost optimization problem based on service oriented VM placement, and develop the Service Oriented VM Placement (SOVMP) scheme. They propose a service-oriented VM placement strategy in DCs and a genetic algorithm to solve service oriented VMs placement optimization and communication cost problem. The proposed VM placement strategy is also suitable for the intelligent computing platform of IoT back end. Such service oriented concept can increase resource utilization rate and reduce the communication cost between VMs. In the part of resource utilization rate, they put emphasis on the communication cost between VMs for better use of resource. Under the situation of limited resources, their service oriented VM placement strategy based on the optimal configuration for different types of VMs can be fully optimized to achieve the minimum communication overhead [16].

In [17], the Many-Objective Virtual Machine Placement (MOVMP) scheme is proposed to model a many-objective VM placement method. Most existing studies do not comprehensively consider all objectives, instead only considering some of them. This scheme considers multiple control issues such as energy consumption, resource utilization, load balance and robustness, and these objectives are simultaneously considered in practical scenarios. To address these issues, energy efficient evolutionary algorithm is developed; it is a high performance algorithm for many-objective problems. The MOVMP scheme also considers actual requirements of cloud providers and offers practical and good experience for end users. Experimental results show that the MOVMP scheme can improve the performance of the VM placement problem in terms of energy saving, load balance, and robustness [17].

Some earlier studies [2], [13]–[17] have attracted considerable attention while introducing unique challenges in handling the VM placement problems in DCs. In this paper, we demonstrate that our cooperative game based scheme significantly outperforms these existing SOVMP [16] and MOVMP [17] schemes.

### B. CONTRIBUTION

Driven by service requirements, the number of VMs has increased so that new challenges to the resource allocation technology of DCs have emerged. As more or all VMs on a DC are performing computational tasks, there is a strong possibility of CPU capacity contention; it will reduce the QoS. To satisfy the growing service requirements, developing a flexible and dynamic resource distribution protocol has become a major challenge. In this work, we focus on the DC resource provisioning problem for heterogeneous VMs. According to the cooperative game solution, we design a fair-efficient resource distribution scheme in order to strike the appropriate performance balance among contradictory requirements. Our dual-level game approach can strike an appropriate performance balance between efficiency and fairness. Although several VM placement control schemes for DCs have been proposed, there is little work that has considered the most proper combination of the efficiency and fairness requirements. Inspired by the above discussed challenges, our main contributions are i) to develop a simple matching algorithm and Mood value implementation algorithm, ii) to explore the sequential interaction of matching and resource distribution algorithms to jointly design an integrated dual-level game model, and iii) to strike an appropriate performance balance.

### C. ORGANIZATION

The rest of the paper is organized as follows. In Section II, we address our proposed VM placement and DC resource allocation algorithms. Based on the matching game model, VMs are dynamically placed in appropriate DCs. And then, physical resources in the DC are dynamically allocated for VMs according to the Mood value. In particular, this section provides fresh insights into the main features of our game-based approach, and shows the main steps of the proposed scheme to increase readability. In Section III, we present the experimental simulation results and performance analysis. By comparing with existing SOVMP [16] and MOVMP [17] schemes, we validate the performance superiority of the proposed scheme. Finally, our conclusions are summarized in Section IV. In this section, we also offer recommendations for future work direction and discuss the

remaining open challenges in this research area along with possible solutions.

## II. THE PROPOSED DC RESOURCE PROVISIONING ALGORITHMS

In this section, we apply the key design principles to address the VM placement and the DC resource distribution problems. To solve these problems, we present a novel dual-level cooperative game model, which consists of the many-to-one matching algorithm and the Mood value based DC resource distribution algorithm.

### A. CLOUD SYSTEM ARCHITECTURE AND DUAL-LEVEL GAME MODEL

In this study, we consider the cloud computing system infrastructure with multiple DCs and VMs platforms. Cloud computing service, i.e., Infrastructure as a service (IaaS), is characterized by resource demands, and the corresponding VM is created with resource allocated by the DC. Usually, important resource to execute VMs is the DC's CPU capacity for computations. As the number of VMs increases, the resource contentions also increase. Under the dynamic changing cloud computing environments, some DCs may be underload, idle, or resource-intensive; that is, the resources in DCs are not being utilized effectively. Therefore, the use of static VM placement algorithm and DC resource distribution algorithm will tend to cause the resource waste or inadequacy, which results in system inefficiency and load imbalance inter-DCs and intra-DC [13], [18]. To tackle these control problems, we propose a dual-level cooperative game model. At the first phase, VMs are placed on appropriate DCs. At the second phase, the limited resource in each DC is distributed fair-efficiently to placing VMs. To provide a well-balanced solution, we characterize game model $\mathbb{G} = \left\{\{\mathbb{D}, \mathbb{V}\}, \left\{S_{DC_i}, S_{VM_j}\right\}, \left\{U_i^{DC}, U_j^{VM}\right\}, \mathcal{K}, \mathbb{S}, t\right\}$ at each time period $t \in \mathcal{T}$ of gameplay;

- Game players: the sets of DCs and VMs. We suppose that there are n DCs, i.e., $\mathbb{D} = \{DC_1, \ldots, DC_n\}$ and at some time, that there are m VMs, i.e., $\mathbb{V} = \{VM_1, \ldots, VM_m\}$.
- Strategy: The strategy of the $DC_{1\le i\le n}$ $\left(S_{DC_i}\right)$ is a feasible resource distribution to ensure VMs' operations. The strategy of the $VM_{1\le j\le m}$ $\left(S_{VM_j} = \left\{M_{VM_j}, m_{VM_j}\right\}\right)$ is a set of requested resource amounts for the cloud service where $M_{VM_j}$ and $m_{VM_j}$ are the maximum and minimum DC's resource requests, respectively.
- Utility: $U_i^{DC}$ and $U_j^{VM}$ are the utility functions of $DC_i$ and $VM_j$, respectively.
- Tasks: $\mathcal{K}$ is the set of randomly generated cloud service tasks $\mathcal{K} = \{\mathcal{K}_1 \ldots \mathcal{K}_8\}$; each $\mathcal{K}$ has its own $m_{VM}$ and $M_{VM}$.
- Feasible matching tuples: $\mathbb{S}$ is the set of all possible matching pairs.
- $\mathcal{T} = \{1 \ldots t, t+1 \ldots\}$ is a time, which is represented by a sequence of time steps with imperfect information.

**TABLE 1. The notations for symbols and functions.**

| Notation | Explanation |
|---|---|
| DC | data center |
| VM | virtual machine |
| $\mathbb{D}$ | set of DCs |
| $\mathbb{V}$ | set of VMs |
| $S_{DC_i}$ | the strategy of the $DC_i$ |
| $S_{VM_j}$ | the strategy of the $VM_j$ |
| $\mathcal{K}$ | the set of randomly generated cloud service |
| $\mathbb{S}$ | the set of all possible matching pairs |
| $t$ | a sequence of time step |
| $U_i^{DC}$ | utility function of $DC_i$ |
| $U_j^{VM}$ | utility function of $VM_j$ |
| $\theta$ | the control parameter for $U_i^{DC}$ |
| $\delta$, $\beta$ | the control parameters for $U_j^{VM}$ |
| $\mathfrak{M}^{DC_i}$ | the total CPU capacity for the $DC_i$ |
| $\phi_{DC_i}^t$ | the $DC_i$'s allocated CPU capacity for running VMs at the time $t$ |
| $x_j^{VM}$ | the assigned CPU computation capacity for the $VM_j$ |
| $\Theta_{DC_i}^{VM_j}$ | the pair preference of $DC_i$ and $VM_j$ |
| $\omega$ | a weight coefficient factor for each preference |
| $\Gamma_{\mathcal{Y}}^{\mathbb{D}}$ | a function for many-to-one matching among DCs and VMs |
| $\mathbb{S}(DC, VM)$ | the matching set of $(DC, VM)$ pairs |
| $\mathcal{Q}(\mathbb{S}(\mathbb{D}\cup\mathbb{V}))$ | the total sum of all $\Theta_{DC}^{VM}$ values within the $\mathbb{S}((\mathbb{D}\cup\mathbb{V}))$ |
| $\mathcal{N}$ | the claimants of the bankruptcy situation |
| $v$ | the characteristic function of each coalition |
| $E$ | an estate that has to be divided among the claimant set $\mathcal{N}$ |
| $d_{C_i}$ | the $C_i$'s demand |
| $\boldsymbol{d}$ | the claim vector of all claimants |
| $\Upsilon_{C_i}$ | the actually assigned resource division for the $C_i$ |
| $QoE_{C_i}^{\mathcal{N}}$ | the $C_i$'s quality of experience |
| $\mathfrak{Z}_{C_i}^{M}$ | the maximal right for the $C_i$ |
| $\mathfrak{Z}_{C_i}^{m}$ | the minimal right for the $C_i$ |
| $\mathbb{V}(S)$ | the resource distribution vector for the coalition $S$ |
| $\mathcal{S}(\mathcal{N}, \mathbb{V}(S))$ | the dissatisfaction degree of coalition $S$ with $\mathbb{V}(S)$ |
| $Mv(\mathcal{N}, C_i)$ | the Mood value for the $C_i$ |

Table 1 lists the notations used in this paper. Based on their own expectations, each DC and VM focus on its own workload and computation outcome, respectively. From the DC's viewpoint, utility function $\left(U^{DC}\right)$ is determined based on the currently working load. To avoid the working overhead, the smaller computation load is the better way for each DC. From the VM's viewpoint, utility function $\left(U^{VM}\right)$ is derived from the outcome gained by the cloud computing service; it is defined according to application's characteristics. Usually, the more DC resource is used, the higher the satisfaction is obtained. In the proposed game model, the utility functions of $DC_i$ and $VM_j$, e.g., $U_i^{DC}$ and $U_j^{VM}$, are defined as follows;

$$\begin{cases} U_i^{DC}\left(\phi_{DC_i}^t, \mathfrak{M}^{DC_i}\right) = \cos\left(\theta \times \dfrac{\phi_{DC_i}^t}{\mathfrak{M}^{DC_i}}\right) \\ U_j^{VM}\left(x_j^{VM}, M_{VM_j}\right) = 1 - \dfrac{\delta}{1 + \exp\left(\beta \times \dfrac{x_j^{VM}}{M_{VM_j}}\right)} \end{cases} \tag{1}$$

where $\theta$, $\delta$ and $\beta$ are the control parameters. $\mathfrak{M}^{DC_i}$ is the total CPU capacity for the $DC_i$ and $\phi_{DC_i}^t$ is the $DC_i$'s allocated CPU capacity for running VMs at the time $t$. $x_j^{VM}$ is the assigned CPU computation capacity for the $VM_j$ where $m_{VM_j} \le x_j^{VM} \le M_{VM_j}$.

### B. MANY-TO-ONE MATCHING ALGORITHM FOR VMs AND DCs

As an instance of cooperative game theory, we use the matching game theory to design the VM placement algorithm.

In 1962, Gale and Shapley developed a many-to-one matching model for the college admission problem [12]. In the many-to-one matching game, game players on each side have preferences over players on the other side, and have enough information to rank players on the other side. Therefore, player in one side tries to be matched to the other player in opposite side so as to satisfy both players as much as possible [12]. In this paper, we formulate a cooperative game to solves the matching problem between the two sets: DCs and VMs. The main objects of each DC and VM are to host a set of VMs, and to select a specific DC, respectively, while maximizing their payoffs. To coordinate VMs and DCs while maximizing the system efficiency, DCs and VMs can exchange the information such as their strategies, and make a set of matching pairs. In particular, a VM can enter a bilateral contract with one DC while a DC can provide its resource to multiple VMs. Therefore, we can capture the matching between DCs and VMs as a many-to-one matching.

To decide the effective matching pairs, one of major concerns is to estimate a pair preference. Using $U_i^{DC}$ and $U_j^{VM}$ functions, the pair preference of $DC_i$ and $VM_j$ $\left(\Theta_{DC_i}^{VM_j}\right)$ is defined as similar as the one used with the weighted-multiplication-of-objective-functions method. Based on the decision science, this approach deals with the properties of utility functions [19]. At time $t$, $\Theta_{DC_i}^{VM_j}$ is estimated as follows;

$$\Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right) = \left(\sum_{i=1}^{2}\left((\omega_i \times f_i)+1\right)^2\right)^{\frac{1}{2}}, \quad \text{s.t.,} \begin{cases} f_1 = U_i^{DC}\left(\phi_{DC_i}^t, \mathfrak{M}^{DC_i}\right) \\ f_2 = U_j^{VM}\left(x_j^{VM}, M_{VM_j}\right) \end{cases} \tag{2}$$

where $0 < \omega < 1$ is a weight coefficient factor for each preference where $\omega_2 = 1 - \omega_1$. For the stable matching, the $\Theta_{DC}^{VM}$ allows one to transform multi-objective fun ctions into a mono-objective one. The main goal of our matching game is to find the optimal set of pairs, which maximize the total sum of all pairs' $\Theta_{DC}^{VM}$ values. To satisfy this goal, DCs and VMs are put into $\mathbb{D}$ and $\mathbb{V}$ sets in random order, and each VM in the set $\mathbb{V}$ in turn examines the DCs in the counterpart set $\mathbb{D}$. DCs have preferences over VMs, and DCs have enough information to rank VMs. Therefore, each DC in one side tries to be matched to the other VM in opposite side so as to satisfy both DC and VM as much as possible. In the main loop of the matching algorithm, DCs find out the most suitable VMs according to (3).

$$MSP\left(DC_i, VM_j\right) = \left(\Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right) \mid VM_j \in \mathbb{V} \text{ and } \times\left(\phi_{DC_i}^t + x_j^{VM}\right) \le \mathfrak{M}^{DC_i}\right) \tag{3}$$

At the first-level game, $x_j^{VM}$ is assumed as the $m_{VM_j}$ of $VM_j$. To get the stable pairs, our developed matching algorithm identifies all potential changes and swaps throughout matching procedure execution. For example, if a DC picks the counterpart VM in the already matched pair, swapping can occur on the basis of MSP values; the new match by swapping should be higher MSP value than its present matching pair.

*Definition 1:* A many-to-one matching among DCs and VMs is a function $\Gamma_{\mathbb{V}}^{\mathbb{D}}$: $\mathbb{D} \bigcup \mathbb{V} \rightarrow \mathbb{S}\left(\left(DC_i, VM_j\right)\right)_{DC_i \in \mathbb{D}, VM_j \in \mathbb{V}}$ where $\mathbb{S}\left((DC, VM)\right)$ is the matching set of $(DC, VM)$ pairs.

The outcome of our matching game is stable when no VM will incur a higher MSP value from changing or swapping its matched DC unilaterally. Let $\mathbb{Q}\left(\mathbb{S}\left(\mathbb{D} \bigcup \mathbb{V}\right)\right)$ denote the total sum of all $\Theta_{DC}^{VM}$ values within the $\mathbb{S}\left(\left(\mathbb{D} \bigcup \mathbb{V}\right)\right)$. A stable outcome of the matching game is the feasible strategy profile $\mathbb{S}\left(\left(\mathbb{D} \bigcup \mathbb{V}\right)\right)^*$ such that for

$$\begin{aligned} &\mathbb{Q}\left(\mathbb{S}\left(\mathbb{D} \bigcup \mathbb{V}\right)\right) \\ &\ge \mathbb{Q}\left(\left(\Gamma_{\mathbb{V}}^{\mathbb{D}}\left(\mathbb{D} \bigcup VM_j\right)\right) \bigcup \left(\mathbb{S}\left(\mathbb{D} \bigcup \mathbb{V}_{-VM_j}\right)\right)\right) \\ &\quad \text{s.t., } \Gamma_{\mathbb{V}}^{\mathbb{D}}\left(\mathbb{D} \bigcup VM_j\right) \notin \mathbb{S}\left(\mathbb{D} \bigcup \mathbb{V}\right) \text{ and } VM_j \in \mathbb{V} \end{aligned} \tag{4}$$

In general, a stable outcome may not exist in a matching game [20]. Therefore, we should prove the existence of a stable outcome for the many-to-one matching game of DCs and VMs by constructing an exact potential function, which can express the payoff changes when all players change their strategies. The existence of a potential function is the characteristic of a potential game. An important feature of a potential game is that potential game has been shown to always converge to a Nash equilibrium when the best response dynamics is performed [12].

*Proposition 1:* The proposed the DC-VM matching game is potential game.

*Proof:* First, consider the situation that a VM simply changes its corresponding DC. We assume that $\left(DC_i, VM_j\right)$ and $\left(DC_k, VM_j\right)$ are feasible paring elements and $\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j}\left((DC, VM)\right)$ represents the $\mathbb{S}\left((DC, VM)\right)$ but the $DC_i$ and $DC_k$ are swapped by the $VM_j$. Based on these notations, we can define the pair set $\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j}\left((DC, VM)\right)$ like as $\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j}\left((DC, VM)\right) = \left(DC_k, VM_j\right) + \mathbb{S}\left((DC, VM)\right)_{DC \in \mathbb{D}, VM \in \mathbb{V} \setminus VM_j}$. For the matching game model in the proposed scheme, the potential function can be expressed as follows:

$$\Psi\left(\mathbb{S}\left(\left(DC_i, VM_j\right)\right)_{DC_i \in \mathbb{D}, VM_j \in \mathbb{V}}\right) = \sum_{VM_j \in \mathbb{V}}\left(MSP\left(DC_i, VM_j\right) \mid DC_i \in \mathbb{D}\right) \tag{5}$$

With the pair $\left(DC_i, VM_j\right)$ in $\mathbb{S}\left((DC, VM)\right)$, the change in the potential function by the $DC_i$ and $DC_k$ swapping by the $VM_j$ is

$$\begin{aligned} &\Psi\left(\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j}\left((DC, VM)\right)\right) \\ &= \Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right) \\ &\quad + \sum_{VM_r \in \mathbb{V} \setminus VM_j}\left(DC_l \in \mathbb{D} \mid MSP\left(DC_l, VM_r\right)\right) \\ &\quad \text{s.t., } \left(\phi_{DC_k}^t + x_j^{VM}\right) \le \mathfrak{M}^{DC_k} \end{aligned} \tag{6}$$

The goal is to assign each VM's matching value that is perfectly aligned with the global potential function in (5). To satisfy this goal, we should show that each VM's matching value can capture the VM's marginal contribution to the potential function. In our matching game, the $VM_j$'s matching value change by the swapping of $DC_i$ and $DC_k$ $\left(\mathrm{X}\left(DC_i \longleftrightarrow DC_k | VM_j\right)\right)$ is defined as;

$$\begin{aligned}&\mathrm{X}\left(DC_i \longleftrightarrow DC_k | VM_j\right)\\&= \left(\Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right) - \Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right)\right) \quad (7)\end{aligned}$$

$\mathrm{X}\left(DC_i \longleftrightarrow DC_k | VM_j\right)$, provided that all other VMs without $VM_j$ collectively still stay $\mathcal{S}'(DC_l, VM_r)_{DC_l \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j}$, is

$$\begin{aligned}&\mathrm{X}\left(DC_i \longleftrightarrow DC_k | VM_j\right)\\&= \left(\Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right) - \Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right)\right)\\&= \left(\Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right) - \Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right)\right)\\&\quad + \left(\sum_{DC_l \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j}\left(\Theta_{DC_l}^{VM_r}\left(t, \phi_{DC_l}^t, x_r^{VM}\right)\right)\right.\\&\qquad \left. - \sum_{DC_l \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j}\left(\Theta_{DC_l}^{VM_r}\left(t, \phi_{DC_l}^t, x_r^{VM}\right)\right)\right)\\&= \left(\Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right)\right.\\&\qquad \left. + \sum_{VM_r \in \mathbb{V}\setminus VM_j}\left(\Theta_{DC_l}^{VM_r}\left(t, \phi_{DC_l}^t, x_r^{VM}\right)\right)\right)\\&\quad - \left(\Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right)\right.\\&\qquad \left. + \sum_{VM_r \in \mathbb{V}\setminus VM_j}\left(\Theta_{DC_l}^{VM_r}\left(t, \phi_{DC_l}^t, x_r^{VM}\right)\right)\right)\\&= \Psi\left(\mathbb{S}\left(\left(DC_i, VM_j\right)\right)_{DC_i \in \mathbb{D}, VM_j \in \mathbb{V}}\right)\\&\quad - \Psi\left(\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j}(DC, VM)\right) \quad (8)\end{aligned}$$

■

Second, consider the swapping situation. We assume that $\left(DC_i, VM_j\right)$ and $(DC_k, VM_l)$ pairs are in the $\mathbb{S}(DC, VM)$ and $\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j \longleftrightarrow VM_l}((DC, VM))$ represents the $\mathbb{S}((DC, VM))$ but the $DC_i$ and $DC_k$ are swapped in the pairs $(DC_i, VM_l)$ and $\left(DC_k, VM_j\right)$. Therefore, we can define the pair set $\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j \longleftrightarrow VM_l}((DC, VM)) = (DC_i, VM_l) + \left(DC_k, VM_j\right) + \mathbb{S}'\left(\left(DC_d, VM_r\right)\right)_{DC_d \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j, VM_l}$. The change in the potential function by the $VM_j$ and $VM_l$ swapping is

$$\begin{aligned}&\Psi\left(\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j \longleftrightarrow VM_l}((DC, VM))\right)\\&= \Theta_{DC_i}^{VM_l}\left(t, \phi_{DC_i}^t, x_l^{VM}\right) + \Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right)\\&\quad + \sum_{VM_r \in \mathbb{V}\setminus VM_j, VM_l}(MSP(DC_l, VM_r) \mid DC_d \in \mathbb{D}),\\&\text{s.t.,}\ \left(\phi_{DC_i}^t + x_l^{VM}\right) \le \mathfrak{M}^{DC_i} \text{ and } \left(\phi_{DC_k}^t + x_j^{VM}\right) \le \mathfrak{M}^{DC_k} \quad (9)\end{aligned}$$

The matching value change by the swapping of $DC_i$ and $DC_k$ $\left(\mathrm{X}\left(DC_i \longleftrightarrow DC_k | VM_j \longleftrightarrow VM_l\right)\right)$ is defined as;

$$\begin{aligned}&\mathrm{X}\left(DC_i \longleftrightarrow DC_k | VM_j \longleftrightarrow VM_l\right)\\&= \left(\Theta_{DC_i}^{VM_l}\left(t, \phi_{DC_i}^t, x_l^{VM}\right) + \Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right)\right)\\&\quad - \left(\Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right) + \Theta_{DC_k}^{VM_l}\left(t, \phi_{DC_k}^t, x_l^{VM}\right)\right)\\&= \left(\Theta_{DC_i}^{VM_l}\left(t, \phi_{DC_i}^t, x_l^{VM}\right) + \Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right)\right)\\&\quad - \left(\Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right) + \Theta_{DC_k}^{VM_l}\left(t, \phi_{DC_k}^t, x_l^{VM}\right)\right)\\&\quad + \left(\sum_{DC_d \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j, VM_l}\left(\Theta_{DC_d}^{VM_r}\left(t, \phi_{DC_d}^t, x_r^{VM}\right)\right)\right.\\&\qquad \left. - \sum_{DC_d \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j, VM_l}\left(\Theta_{DC_d}^{VM_r}\left(t, \phi_{DC_d}^t, x_r^{VM}\right)\right)\right)\\&= \left(\Theta_{DC_i}^{VM_l}\left(t, \phi_{DC_i}^t, x_l^{VM}\right) + \Theta_{DC_k}^{VM_j}\left(t, \phi_{DC_k}^t, x_j^{VM}\right)\right.\\&\qquad \left. + \sum_{DC_d \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j, VM_l}\left(\Theta_{DC_d}^{VM_r}\left(t, \phi_{DC_d}^t, x_r^{VM}\right)\right)\right)\\&\quad - \left(\Theta_{DC_i}^{VM_j}\left(t, \phi_{DC_i}^t, x_j^{VM}\right) + \Theta_{DC_k}^{VM_l}\left(t, \phi_{DC_k}^t, x_l^{VM}\right)\right.\\&\qquad \left. + \sum_{DC_d \in \mathbb{D}, VM_r \in \mathbb{V}\setminus VM_j, VM_l}\left(\Theta_{DC_d}^{VM_r}\left(t, \phi_{DC_d}^t, x_r^{VM}\right)\right)\right)\\&= \Psi\left(\mathbb{S}\left(\left(DC_i, VM_j\right)\right)_{DC_i \in \mathbb{D}, VM_j \in \mathbb{V}}\right)\\&\quad - \Psi\left(\mathbb{S}'_{DC_i \longleftrightarrow DC_k | VM_j \longleftrightarrow VM_l}((DC, VM))\right) \quad (10)\end{aligned}$$

■

Based on the best response update to determine a stable outcome, our many-to-one matching algorithm can be executed to converge to a stable outcome. In the proposed scheme, DCs and VMs are responsible for the information exchange in an online fashion. Our matching algorithm consists of three phases: initiation, matching and swapping phases. In the below procedure, lines 1 and 4 describe the initiation phase for the VMs and DCs. The loop from lines 5 and 14 describes the matching and swapping phases. Through this repeated matching loop, a VM can match at most one DM.

### C. MOOD VALUE BASED DC RESOURCE DISTRIBUTION ALGORITHM

After the many-to-one matching among VMs and DCs, each DC distributes its limited resource to the corresponding VMs.

**Algorithm 1** Matching Game Model Procedure

VMs and DCs' Matching Game Algorithm

**Input:** VMs' resource requests and DCs' resource capacities.
**Output:** stable mapping pairs between VMs and DCs
**1:** Each VM collects all the information of DCs.
**2:** Individual VMs calculate their preferences for DCs using Eq.(1)-(3), and build their own preference lists.
**3:** Each DC collects all the information of VMs.
**4:** Individual DCs calculate their preferences for VMs using Eq.(1)-(3), and build their own preference lists.
**5:** Each VM proposes to its most preferred DC according to its preference list.
**6: Repeat**
**7: If** (DC can accept all VMs within its capacity constraint $\mathfrak{M}^{DC}$) **then**
**8:** that DC temporarily holds all the matching VMs.
**9: Else**
**10:** according to the DC's preference list, the DC selects the most preferred VMs without violating the capacity constraint.
**11:** based on the DC's decision, the requested VM can be rejected or accepted by swapping.
**12: End If**
**13:** Not matched VM proposes to its next most preferred DC according to its preference list.
**14: Until** no VM can change the current matching solution.
**15:** Confirm the stable matching pairs

DC resource allocation becomes a challenging problem when the available resource is limited and not enough to fully satisfy VMs' maximum demands $\left(\sum M_{VM}\right)$. In such situations, resource allocation algorithms need to ensure the efficiency for the resource usability and the fairness among VMs. Under awareness about the available resource amount and VMs' demands, our major goals are i) defining the level of justice in the resource distribution and ii) formulating a model to allocate a quantitative resource amount for each VM, and iii) finding the best set-value solution in terms of efficiency and fairness. To satisfy these goals, we are also inspired by cooperative game theory.

A classical set-value solution for n-player cooperative games is the core, which is defined as the set of allocations that cannot be improved upon by any coalition. The core, sometimes it is called as a cooperative Nash equilibrium, is useful to obtain the stability condition of the coalitional cooperative game. However, the core is always well-defined, but can be empty. In addition, the fairness concept would still prevent a stable cooperation in some situations where the core is non-empty. Therefore, a new solution that provides the most preferable distribution strategy is required [11], [12]. In 2017, F. Fossati et al proposed the main concept of Mood value [11]. With the core, the Mood value is a strong set-value solution concept based on the fair distribution of total gains to the players, assuming that they all collaborate. Therefore, by using the Mood value approach, we can attempt to equalize each VM's satisfaction with respect to an effective distribution of the DC resource.

At the second-level game, we develop a novel DC resource allocation algorithm based on the Mood value. Generally, the resource distribution problem for $n$ VMs leads to the similar conflict situation as in the money division problem for $n$ claimants. Therefore, the DC resource allocation algorithm is analogous to the bankruptcy game model. The bankruptcy game assumes that a company becomes bankrupt and this company owes money to $n$ money claimants; the money is needed to be divided among these claimants. This conflicting situation also introduces an $n$ player cooperative game where the players are seeking for the equilibrium point to divide the money [12], [21]. The bankruptcy game is defined as $G(\mathcal{N}, v)$ where $\mathcal{N} = \{C_1 \dots C_i \dots C_n\}$ represents the claimants of the bankruptcy situation and $v$ is the characteristic function that associates to each coalition ($S$) of claimants: $2^{\mathcal{N}} \rightarrow \mathbb{R}$ where $v(\emptyset) = 0$ [11], [12], [21]:

$$v(S) = \max\ (0, E - \sum_{C_i \in \mathcal{N} \setminus S} d_{C_i}) \quad \text{s.t., } S \subset \mathcal{N} \setminus \{\emptyset\}, \quad E < \sum_{C_i \in \mathcal{N}} d_{C_i} \text{ and } d_{C_i} \in \boldsymbol{d} \quad (11)$$

where $E \geq 0$ is an estate that has to be divided among the claimant set $\mathcal{N}$ and $d_{C_i}$ is the $C_i$'s demand. $\boldsymbol{d} \in \mathbb{R}_+^{|\mathcal{N}|}$ is the claim vector of all claimants. The important properties of bankruptcy game are i) super-additivity, ii) supermodular and iii) Pareto optimality.

$$\begin{cases} i)\ v(S_1) + v(S_2) \leq v\left(S_1 \bigcup S_2\right), \\ \quad \text{s.t., } \forall S_1, S_1 \subseteq \mathcal{N} \text{ and } S_1 \bigcap S_2 = \emptyset \\ ii)\ v(S_1) + v(S_2) \leq v\left(S_1 \bigcap S_2\right) + v\left(S_1 \bigcup S_2\right), \\ \quad \text{s.t., } \forall S_1, S_1 \subseteq \mathcal{N} \\ iii)\ \sum\limits_{C_i \in \mathcal{N}} \Upsilon_{C_i} = E, \\ \quad \text{s.t., } 0 \leq \Upsilon_{C_i} \leq d_i \end{cases} \quad (12)$$

where $\Upsilon_{C_i}$ means the actually assigned resource division for the $C_i$. The restriction (iii) implies that no claimant gets more than he claims or less than zero and that the total amount E is divided among the claimants. Based on the bankruptcy game model, we can re-define the DC resource distribution problem. Simply, we can assume the $VM_i$ as the $C_i$ and $E$ is the DC's $\mathfrak{M}^{DC}$. $\mathcal{N}$ is the set of VMs, which are placed in the DC, and $d_{C_i}$ is the $M_{VM_i}$. The $C_i$'s quality of experience $\left(QoE_{C_i}^{\mathcal{N}}\right)$, which is a measure of the delight or annoyance of the $C_i$'s experience with the resource distribution, is defined as

$$QoE_{C_i}^{\mathcal{N}} = \frac{\Upsilon_{C_i} - \mathfrak{Z}_{C_i}^m}{\mathfrak{Z}_{C_i}^M - \mathfrak{Z}_{C_i}^m}, \quad \text{s.t., } \begin{cases} \mathfrak{Z}_{C_i}^M = v(\mathcal{N}) - v(\mathcal{N} \setminus C_i) \\ \mathfrak{Z}_{C_i}^m = v(C_i) \end{cases} \quad (13)$$

where $\mathfrak{Z}_{C_i}^M$ and $\mathfrak{Z}_{C_i}^m$ are the maximal and minimal right for the $C_i$, respectively. If $\sum_{C_i \in \mathcal{N}} d_{C_i} \leq E$, the $QoE_{C_i}^{\mathcal{N}}$ is 1. Therefore, the $C_i$ is measured to be completely satisfied when the $C_i$ gets $\mathfrak{Z}_{C_i}^M$ and extremely unsatisfied when he gets $\mathfrak{Z}_{C_i}^m$, i.e., the $QoE_{C_i}^{\mathcal{N}}$ is 0 [11].

By using the game-theoretic interpretation, the dissatisfaction degree for fairness can be defined. According to the gain of the player from the cooperation, it is defined as the ratio of the loss incurred by the complementary coalition to the loss incurred by the current coalition itself. Mathematically, the dissatisfaction degree of coalition $S$ with the resource distribution vector $\mathbb{V}(S)$ $(\mathcal{S}(\mathcal{N}, \mathbb{V}(S)))$ is formulated as follows [11], [22];

$$\begin{aligned} \mathcal{S}(\mathcal{N}, \mathbb{V}(S)) &= \frac{\mathbb{V}(\mathcal{N} \backslash S) - \nu(\mathcal{N} \backslash S)}{\mathbb{V}(S) - \nu(S)} \\ &= \frac{\nu(\mathcal{N}) - \mathbb{V}(S) - \nu(\mathcal{N} \backslash S)}{\mathbb{V}(S) - \nu(S)} \end{aligned} \tag{14}$$

The dissatisfaction degree of the coalition $S$ quantifies its desire to leave that coalition. If the $\mathbb{V}(S)$ is the same as the $\nu(S)$, the desire of disrupting the coalition $S$ is maximum. If the $\mathbb{V}(S) = (\nu(\mathcal{N}) - \nu(\mathcal{N} \backslash S))$, the desire of disrupting the coalition $S$ is zero. If the $\mathbb{V}(S) > (\nu(\mathcal{N}) - \nu(\mathcal{N} \backslash S))$, i.e., $\mathcal{S}(S, \mathbb{V}) < 0$, all the members in the coalition $S$ are hyper-enthusiastic to cooperate each other while maintaining the $S$ coalition. Based on the relationship of $\mathcal{S}(S, \mathbb{V}(S))$, $QoE_{C_i}$ in the equation (13) can be re-defined as follows [11];

$$\begin{aligned} QoE_{C_i}^{\mathcal{N}} &= \frac{\Upsilon_{C_i} - \mathfrak{Z}_{C_i}^m}{\mathfrak{Z}_{C_i}^M - \mathfrak{Z}_{C_i}^m} = \frac{\Upsilon_{C_i} - \nu(C_i)}{\mathfrak{Z}_{C_i}^M - \nu(C_i)} \\ &= \frac{\mathbb{V}(C_i) - \nu(C_i)}{\nu(\mathcal{N}) - \nu(\mathcal{N} \backslash C_i) - \nu(C_i)} \end{aligned} \tag{15}$$

According to (14), $\mathcal{S}(\mathcal{N}, \mathbb{V}(C_i))$ can also be re-defined given by;

$$\begin{aligned} &\mathcal{S}(\mathcal{N}, \mathbb{V}(C_i)) \\ &= \frac{\nu(\mathcal{N}) - \mathbb{V}(C_i) - \nu(\mathcal{N} \backslash C_i)}{\mathbb{V}(C_i) - \nu(C_i)} \\ &= \frac{\nu(\mathcal{N}) - \mathbb{V}(C_i) - \nu(\mathcal{N} \backslash C_i) - (\nu(C_i) - \nu(C_i))}{\mathbb{V}(C_i) - \nu(C_i)} \\ &= \frac{\nu(\mathcal{N}) - \nu(\mathcal{N} \backslash C_i) - \nu(C_i)}{\mathbb{V}(C_i) - \nu(C_i)} - \frac{\mathbb{V}(C_i) - \nu(C_i)}{\mathbb{V}(C_i) - \nu(C_i)} \\ &= \frac{\nu(\mathcal{N}) - \nu(\mathcal{N} \backslash C_i) - \nu(C_i)}{\mathbb{V}(C_i) - \nu(C_i)} - 1 = \frac{1}{QoE_{C_i}^{\mathcal{N}}} - 1 \\ &\Longleftrightarrow QoE_{C_i}^{\mathcal{N}} = \frac{1}{(\mathcal{S}(\mathcal{N}, \mathbb{V}(C_i)) + 1)} \end{aligned} \tag{16}$$

*Definition 2:* Using the bankruptcy game model characterized by $G(\mathcal{N}, v)$, the Mood value is a set-value solution for each $C_i \in \mathcal{N}$. The Mood value for the $C_i$ $(Mv(\mathcal{N}, C_i))$ is defined as follows [11];

$$Mv(\mathcal{N}, C_i) = \left\{ \mathbb{V}(C_i) \mid QoE_{C_i}^{\mathcal{N}} = QoE_{C_j}^{\mathcal{N}} \text{ and } \forall C_i, C_j \in \mathcal{N} \right\} \tag{17}$$

Due to the relation between the $QoE_C$ and $\mathcal{S}(\mathcal{N}, \mathbb{V}(C))$, the Mood value solution provides the same dissatisfaction degree for each $C$. In other words, equalizing the dissatisfaction degree of each $C$ means to equalize the mood of each $C$. Therefore, the fairness idea behind the Mood value is that each $C$ has the same $\mathcal{S}(\mathcal{N}, \mathbb{V}(C))$ and the resource distribution rule to equalize the $\mathcal{S}(\mathcal{N}, \mathbb{V}(C))$ is the $Mv(\mathcal{N}, C)$ for each $C$. In this study, we design a new DC resource distribution algorithm based on the Mood value. First, each individual DC ensures the $m_{VM}$ request amounts for all the placed VMs. And then, $\nu(C_i)$ is simply calculated based on the $U_i^{VM}\left(x_i^{VM}, M_{VM_i}\right)$ according to (1). Finally, the remaining DC's resource is distributed for the placed VMs using the Mood value.

### D. MAIN STEPS OF PROPOSED DUAL-LEVEL RESOURCE CONTROL ALGORITHM

Recently, DCs play an important role to deliver high quality cloud computing services to end users. For the effective operation of DCs, the limited resource is virtualized to support different application tasks. Virtualization adds flexibility to DC management but that flexibility comes at the expense of control overhead. This paper considers a new method to share fair-efficiently the limited DC resource based on the matching game and Mood value. In order to implement our dual-level game model, we logically partition the time-axis into equal intervals of length unit_time. Every unit_time, our proposed scheme is repeatedly executed through a step-by-step online manner. The main steps of the proposed DC resource provisioning algorithm are given next.

*Step 1:* At the initial time, cloud computing system parameters and application tasks are characterized by the simulation scenario and Table 2.

**TABLE 2. System parameters used in the simulation experiments.**

| Cloud service types | Minimum CPU request | Maximum CPU request | Service Duration |
|---|---|---|---|
| $\mathcal{K}_1$ | $m_{VM}$ = 50MHz/unit_time | $M_{VM}$ = 200MHz/unit_time | 30 unit_time |
| $\mathcal{K}_2$ | $m_{VM}$ = 80MHz/unit_time | $M_{VM}$ = 300MHz/unit_time | 50 unit_time |
| $\mathcal{K}_3$ | $m_{VM}$ = 100MHz/unit_time | $M_{VM}$ = 500MHz/unit_time | 40 unit_time |
| $\mathcal{K}_4$ | $m_{VM}$ = 150MHz/unit_time | $M_{VM}$ = 750MHz/unit_time | 25 unit_time |
| $\mathcal{K}_5$ | $m_{VM}$ = 200MHz/unit_time | $M_{VM}$ = 1GHz/unit_time | 60 unit_time |
| $\mathcal{K}_6$ | $m_{VM}$ = 250MHz/unit_time | $M_{VM}$ = 1.2GHz/unit_time | 35 unit_time |
| $\mathcal{K}_7$ | $m_{VM}$ = 350MHz/unit_time | $M_{VM}$ = 1.5GHz/unit_time | 30 unit_time |
| $\mathcal{K}_8$ | $m_{VM}$ = 450MHz/unit_time | $M_{VM}$ = 3GMHz/unit_time | 20 unit_time |

| Parameter | Value | Description |
|---|---|---|
| m | 10 | the number of DCs |
| $\theta$ | 1.5 | the control parameter of DC's utility function |
| $\delta, \beta$ | 2 , 3 | the control parameters of VM's utility function |
| $\mathfrak{M}^{DC}$ | 10GHz | the maximum CPU capacity of DC |
| $x_j^{VM}$ | $m_{VM_j} \leq x_j^{VM} \leq M_{VM_j}$ | the assigned CPU computation capacity for the $VM_j$ |
| $\omega_1, \omega_2$ | 0.7, 0.3 | weight coefficient factors for each preference |

*Step 2:* Users generates sequentially application tasks and related VMs are dynamically formed. At that time, DCs are waiting to execute the generated VMs. Each VM has maximum and minimum DC's resource requests, i.e., $M_{VM}$ and $m_{VM}$.

*Step 3:* The payoffs of each DC and VM are defined according to (1). Based on these payoffs, the pair preference of each DC-VM is calculated using the equation (2).

*Step 4:* Based on the proposed the DC-VM matching algorithm, we can get the stable many-to-one matching pairs.

*Step 5:* After obtaining the stable matching pairs, we can categorize VMs; each DC can have multiple placed VMs. If the total sum of the placed VMs' $M_{VM}$ amounts is less than the DC's available resource $\left(\mathfrak{M}^{DC}\right)$, each $M_{VM}$ is fully allocated for the corresponding VM; $QoE$ is 1 and all VMs are completely satisfied.

*Step 6:* If the total sum of the placed VMs' $M_{VM}$ amounts is larger than the $\mathfrak{M}^{DC}$, all the placed VMs are ensured their $m_{VM}$, preferentially. And then, the remaining DC's resource is distributed to the placed VMs based on the Mood value.

*Step 7:* Based on the step-by-step repeated game process, the VMs and DCs interact with each other in an online manner.

*Step 8:* Every unit_time, our dual-level game model is executed sequentially to provide a fair-efficient DC resource distribution solution.

*Step 9:* Under the dynamic cloud computing system environment, our game based control approach is constantly self-monitoring the current system conditions; proceeds to Step 2 for the next dual-level game iteration.

## III. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our proposed protocol. In order to evaluate the implementation of the proposed model, the performance analysis is performed by comparing the existing SOVMP scheme [16] and MOVMP [17] schemes. To ensure a fair comparison, the following simulation assumptions and cloud computing system platform are used.

- There are ten DCs, i.e., $\mathbb{D} = \{DC_1, \ldots, DC_{10}\}$, and multiple VMs, which are generated based on the Poisson process; the generation rate range $\lambda$ is varied from 0 to 3.
- VMs can be categorized as eight different task groups, i.e., $\{\mathcal{K}_1, \ldots, \mathcal{K}_8\}$. Each $\mathcal{K}$ is specified according to the minimum and maximum computation requirements. They are generated with equal probability.
- The time-axis is partitioned by equal intervals of length, called unit_time, and our dual-level game model is executed at each unit_time.
- For computation simplicity, the DC's resource is specified in terms of basic computation units (BCUs), where one BCU is the minimum amount of resource distribution, e.g., 50MHz in our system, by using the Mood value method.
- System performance measures obtained on the basis of 100 simulation runs are plotted as functions of the offered task generation rate.
- For simplicity, we assume the absence of physical obstacles in the experiments.

To demonstrate the validity of our proposed method, we measured the normalized DC resource usability, cloud service success ratio, and fairness among VMs in the cloud system. Table 2 shows the system parameters used in the simulation.

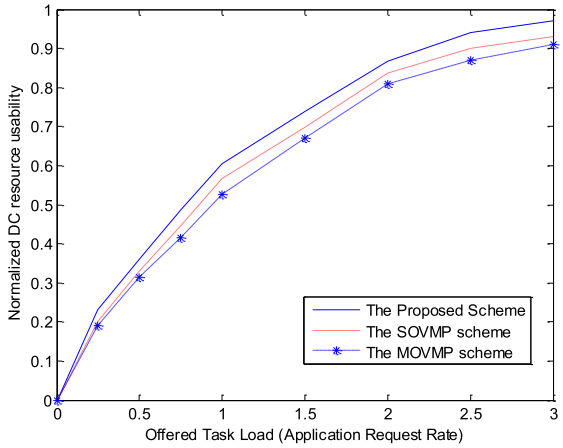



**FIGURE 1.** Normalized DC resource usability.

Figure 1 compares the DC resource usability of each scheme. In this study, the DC resource is the CPU computation capacity. To estimate the total system efficiency, the DC resource usability is a key factor in the cloud service operation. All schemes exhibit a similar trend; however, the proposed scheme outperforms the existing methods from low to high task generation intensities. By using a dual-level cooperative game paradigm, VMs in our scheme are adaptively placed in the most adaptable DC and get the DC resource efficiently; it can improve the DC resource usability than other SOVMP and MOVMP schemes.

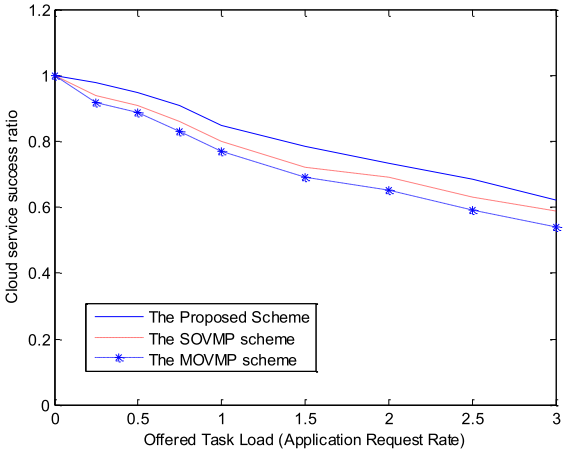



**FIGURE 2.** Cloud service success ratio.

Figure 2 compares the cloud service success ratio. In this study, each application task for cloud service is formed as a VM. Therefore, the cloud service success ratio can be thought as a task complete probability to the total number of generated tasks. Based on the cooperative game features, VMs and DCs in our scheme are dynamically matched

and the DC's available resource is distributed effectively to its corresponding VMs. Therefore, the gain in this performance evaluation criterion is a result of our scheme's self-adaptability and real-time effectiveness. As you see in Fig.2, the proposed scheme attains superior cloud service success ratio to other schemes, which were designed as one-sided protocols and do not respond to current cloud system conditions.

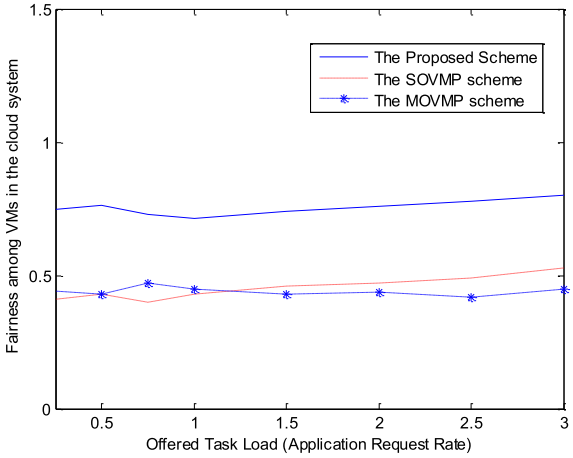



**FIGURE 3.** Fairness among VMs in the cloud system.

The curves in Figure 3 indicate the fairness among VMs in the cloud system. In this study, the evaluation of fairness is measured based on the concept of Jain's index, which has been frequently used to measure the fairness of system resource allocations [11]. Usually, the Jain's index is bounded between $1/n$ and 1 where n is the total number of resource requesters. The maximum fairness is measured when all the requesters obtain the same fraction of demand and the minimum fairness is measured when it exists only one requester that receives all the resource. The Jain's index has the good properties - Population size independence, Scale and metric independence, Boundedness and Continuity [11]. According to the feature of Mood value, each DC fairly distributes its remaining resource in a distributed online manner while reflecting VMs' features.

Under widely different and diversified task generation intensities, our dual-level cooperative game approach can obtain synergistic and complementary characteristics for cloud services. It is an effective and suitable way to operate practically the cloud computing system. Therefore, the simulation results shown in Figures 1 to 3 demonstrate that the proposed scheme can attain an appropriate fair-efficient system performance, something that the existing SOVMP [16] and MOVMP [17] schemes cannot offer.

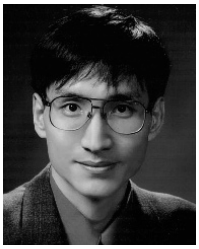
## IV. SUMMARY AND CONCLUSIONS

Virtualization is the enabling technique to facilitate the sharing of limited system resource, with high flexibility, predictable performance, reliability and controllability. Nowadays, DC virtualization has been the subject of recent research interests. In particular, with the large amount of cloud computing services, how to place VMs efficiently to available DCs has become an essential topic in 5G networks. In this study, we develop a new DC resource provisioning scheme for cloud computing services. By considering the current resource constraints, the major challenge is to ensure global QoS metrics for different service requests. In contrast to existing protocols in this research area, we present two cooperative game formulations by adopting the main concepts of matching game model and Mood value. Using the step-by-step interactive dual-level game process, we can effectively provide the DC resource for different VMs while rewarding a tradeoff between service fairness and system efficiency. Under widely different and dynamic cloud computing situations, the proposed scheme adaptively responds to the current DC situations and find an adaptable solution for the cloud system operations. Through extensive simulation experiments, we can confirm that the proposed scheme is a more robust and efficient method than other existing protocols. As a part of future work, we plan to extend the presented approach with other resource dimensions such as energy consumptions and I/O devices. In addition, we will focus on the VM migration issue among DCs for load balancing. Furthermore, we would like to enhance the presented approach with more advanced game models by adding bargaining solutions and mechanism designs. It can improve the system robustness. Finally, we would like to take a research opportunity to address the differential privacy and deep learning algorithms in the VM placement problems.

## REFERENCES

[1] R. Cziva, S. Jouët, D. Stapleton, F. P. Tso, and D. P. Pezaros, "SDN-based virtual machine management for cloud data centers," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 2, pp. 212–225, Jun. 2016.
[2] Z. Á. Mann, "Multicore-aware virtual machine placement in cloud data centers," *IEEE Trans. Comput.*, vol. 65, no. 11, pp. 3357–3369, Nov. 2016.
[3] Y. Liu, Y. Sun, J. Ryoo, S. Rizvi, and A. V. Vasilakos, "A survey of security and privacy challenges in cloud computing: Solutions and future directions," *J. Comput. Sci. Eng.*, vol. 9, no. 3, pp. 119–133, 2015.
[4] T. H. Duong-Ba, T. Nguyen, B. Bose, and T. T. Tran, "A dynamic virtual machine placement and migration scheme for data centers," *IEEE Trans. Services Comput.*, to be published.
[5] K. Kim, S. Uno, and M. Kim, "Adaptive QoS mechanism for wireless mobile network," *J. Comput. Sci. Eng.*, vol. 4, no. 2, pp. 153–172, 2010.
[6] S. Kim, S. Park, Y. Kim, S. Kim, and K. Lee, "VNF-EQ: Dynamic placement of virtual network functions for energy efficiency and QoS guarantee in NFV," *Cluster Comput.*, vol. 20, no. 3, pp. 2107–2117, 2017.
[7] J. Duan and Y. Yang, "A load balancing and multi-tenancy oriented data center virtualization framework," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 8, pp. 2131–2144, Aug. 2017.
[8] C. Zhu, "Data-compression-based resource management in cloud computing for biology and medicine," *J. Comput. Sci. Eng.*, vol. 10, no. 1, pp. 21–31, 2016.
[9] X. Li, Z. Qian, S. Lu, and J. Wu, "Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center," *Math. Comput. Model.*, vol. 58, nos. 5–6, pp. 1222–1235, 2013.
[10] H. B. Yedder, Q. Ding, U. Zakia, Z. Li, S. Haeri, and L. Trajkovic, "Comparison of virtualization algorithms and topologies for data center networks," in *Proc. IEEE ICCCN*, Jul. 2017, pp. 1–6.
[11] F. Fossati, S. Moretti, and S. Secci, "A mood value for fair resource allocations," in *Proc. IFIP Netw. Conf. Workshops*, Jun. 2017, pp. 1–9.
[12] S. Kim, *Game Theory Applications in Network Design*. Hershey, PA, USA: IGI Global, 2014.

[13] L. Guo, G. Hu, Y. Dong, Y. Luo, and Y. Zhu, "A game based consolidation method of virtual machines in cloud data centers with energy and load constraints," *IEEE Access*, vol. 6, pp. 4664–4676, 2018.
[14] S. Wang, A. Zhou, C.-H. Hsu, X. Xiao, and F. Yang, "Provision of data-intensive services through energy- and QoS-aware virtual machine placement in national cloud data centers," *IEEE Trans. Emerg. Topics Comput.*, vol. 4, no. 2, pp. 290–300, Apr. 2016.
[15] A. Beloglazov and R. Buyya, "Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 7, pp. 1366–1379, Jul. 2013.
[16] Y.-H. Chen and C.-Y. Chen, "Service oriented cloud VM placement strategy for Internet of Things," *IEEE Access*, vol. 5, pp. 25396–25407, 2017.
[17] X. Ye, Y. Yin, and L. Lan, "Energy-efficient many-objective virtual machine placement optimization in a cloud computing environment," *IEEE Access*, vol. 5, pp. 16006–16020, 2017.
[18] B.-H. Park, Y. Kim, B.-D. Kim, T. Hong, S. Kim, and J. K. Lee, "High performance computing: Infrastructure, application, and operation," *J. Comput. Sci. Eng.*, vol. 6, no. 4, pp. 280–286, 2012.
[19] Y. Collette and P. Siarry, *Multiobjective Optimization*. New York, NY, USA: Springer, 2004.
[20] S. Bahrami, V. W. S. Wong, and J. Huang, "Demand response for data centers in deregulated markets: A matching game approach," in *Proc. IEEE Int. Conf. Smart Grid Commun.*, Oct. 2017, pp. 344–350.
[21] D. Niyato and E. Hossain, "A cooperative game framework for bandwidth allocation in 4G heterogeneous wireless networks," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2006, pp. 4357–4362.
[22] D. Gately, "Sharing the gains from regional cooperation: A game theoretic application to planning investment in electric power," *Int. Econ. Rev.*, vol. 15, no. 1, pp. 195–208, 1974.
[23] J. So and H. Byun, "Load-balanced opportunistic routing for duty-cycled wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 1940–1955, Jul. 2017.
[24] E. C. Park, D. Y. Kim, C. H. Choi, and J. So, "Improving quality of service and assuring fairness in WLAN access networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 4, pp. 337–350, Apr. 2007.



**SUNGWOOK KIM** received the B.S. and M.S. degrees in computer science from Sogang University, Seoul, South Korea, in 1993 and 1995, respectively, and the Ph.D. degree in computer science from Syracuse University, Syracuse, NY, USA, supervised by P. K. Varshney, in 2003. He has held faculty positions with the Department of Computer Science, Chung-Ang University, Seoul. In 2006, he returned to Sogang University, where he is currently a Professor with the Department of Computer Science and Engineering, and he is a Research Director of the Network Research Laboratory. His research interests include resource management, online algorithms, adaptive quality-of-service control, and game theory for network design.