Dynamic Resource Prediction and Allocation for Cloud Data Center Using the Multiobjective Genetic Algorithm

Fan-Hsun Tseng, Xiaofei Wang, *Member, IEEE*, Li-Der Chou, *Member, IEEE*, Han-Chieh Chao, *Senior Member, IEEE*, and Victor C. M. Leung, *Fellow, IEEE*

***Abstract*—In order to optimize the resource utilization of physical machines (PMs), the workload prediction of virtual machines (VMs) is vital but challenging. Most of existing literatures focus on either resource prediction or allocation individually, but both of them are highly correlated. In this paper, we propose a multiobjective genetic algorithm (GA) to dynamically forecast the resource utilization and energy consumption in cloud data center. We formulate a multiobjective optimization problem of resource allocation, which considers the CPU and memory utilization of VMs and PMs, and the energy consumption of data center. The proposed GA forecasts the resource requirement of next time slot according to the historical data in previous time slots. We further propose a VM placement algorithm to allocate VMs for next time slot based on the prediction results of GA. In our simulation-based analysis, the optimal solution for resource prediction under stable and unstable utilization tendency is found by the proposed GA. The prediction result is superior to the previous proposed Grey forecasting model. Results show that the proposed VM placement algorithm not only increases the average utilization level of CPU and memory but also decreases the energy consumption of cloud data center.**

***Index Terms*—Cloud data center, genetic algorithm (GA), multiobjective optimization (MOO), resource allocation, resource prediction.**

Manuscript received April 20, 2016; revised September 4, 2016, January 19, 2017, and June 18, 2017; accepted June 18, 2017. Date of publication July 21, 2017; date of current version May 2, 2018. This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 104-2917-I-008-008, Grant 104-2221-E-197-014, and Grant 105-2221-E-197-010-MY2. *(Corresponding author: Han-Chieh Chao.)*

F.-H. Tseng is with the Department of Technology Application and Human Resource Development, National Taiwan Normal University, Taipei 106, Taiwan (e-mail: fanhsuntseng@ieee.org).

X. Wang is with the Tianjin Key Laboratory of Advanced Networking, School of Computer Science and Technology, Tianjin University, Tianjin 300350, China, and also with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: xiaofeiwang@ieee.org).

L.-D. Chou is with the Department of Computer Science and Information Engineering, National Central University, Taoyuan 320, Taiwan (e-mail: cld@csie.ncu.edu.tw).

H.-C. Chao is with the Department of Electrical Engineering, National Dong Hwa University, Hualien 974, Taiwan, the Department of Computer Science and Information Engineering and the Department of Electronic Engineering, National Ilan University, Yilan 260, Taiwan, the College of Mathematics and Computer Science, Wuhan Polytechnic University, Wuhan 430023, China, and the School of Information Science and Engineering, Fujian University of Technology, Fuzhou 350118, China (e-mail: hcc@mail.ndhu.edu.tw).

V. C. M. Leung is with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: vleung@ece.ubc.ca).

Digital Object Identifier 10.1109/JSYST.2017.2722476

I. Introduction

CLOUD computing has become a popular research issue because its high scalability, flexibility, and cost-efficiency satisfy emerging computation requirements [1]. It integrates the features of grid computing [2] with the capabilities of a virtualization technique. New communication patterns on the basis of computer applications are combined with Internet communication between multiple computers in data centers [3]. Remote host provides cloud services so that users can share data and collaborate tasks in clouds. Cloud computing provides with massive computing [4], sharing storage [5], green computing [6], load balancing [7], and scalable service with optimal construction and management cost [8].

The requirement and utilization of virtual machines (VMs) and physical machines (PMs) influence resource allocation. Researchers investigate how to predict the resource utilization of cloud data center [9]–[12]. In general, history data are utilized to forecast the future tendency of resource requirement. The ultimate goal of resource prediction is accurate so that providers can allocate resources based on the precise prediction results. The purpose of resource prediction and allocation not only eliminates unneeded resource wastage but also satisfies incoming user demands with appropriate resource budget. However, most of existing literatures focus on prediction or allocation separately. The two issues should be considered and solved jointly due to embroiled relation between them.

In this paper, we formulate a multiobjective optimization (MOO) problem of resource allocation for cloud data center. We propose a multiobjective genetic algorithm (GA) to forecast resource utilization in next time slot, which includes central processing unit (CPU) and memory utilization, and energy consumption. The VM placement algorithm is further proposed for maximizing resource utilization and minimizing energy consumption. It allocates VMs based on the resource prediction results of the proposed GA. This paper aims to predict resource demands accurately for minimizing the energy consumption of cloud data center.

The rest of this paper is organized as follows. Section II introduces main issues of resource prediction and allocation for data center and compares related works. In Section III, the MOO problem of resource allocation in data center is formulated. The prediction model and VM placement algorithm are proposed in Section IV. Section V illustrates the proposed mechanism with different resource utilization tendencies then discusses prediction and allocation results. Finally, conclusion of this work is drawn in Section VI.

II. Related Works

Resource prediction and allocation for cloud data center have been widely studied in the literature, e.g., workload modeling [13], workload characterization and prediction [14], predictive model for workload forecasting [15], and traffic reduction [16]. In [17], Dabbagh *et al*. have surveyed the challenges of resource prediction, consolidation, and allocation for clouds. Energy efficiency issues and power management techniques have been discussed. Zohar *et al*. have proposed predictive ACKs (PACK) for cloud users in [18]. The PACK aims to offload the traffic redundancy of cloud servers to end clients. In [19], Di *et al*. have considered both workload and hostload prediction errors to drive the upper bound of task length. The proposed adaptive algorithm with divisible resource and payment budget decreases the cloud task execution length.

In [20], Di *et al*. have utilized Bayesian model to predict host load for achieving the defined service-level agreement. The authors used a monthly statistical data of Google data center as historical data, which is an enormous amount of training data and process. Prevost *et al*. have predicted the network loads of cloud data center by using stochastic model and neural network in [21]. However, the authors only investigated resource prediction rather than both of prediction and allocation in our work. In [22], the Grey forecasting model (or simply Grey model throughout this paper) has been utilized to predict VM workloads in data center. The Grey model accurately forecasts the increased and decreased tendency of VM workloads, but it is unsatisfied with forecasting fluctuant VM workloads.

The proposed algorithm in [23] reserved resources for media streaming applications and reduced the monetary cost of resource allocation in cloud. In [24], Hossain *et al*. have proposed three consolidation schemes to decrease migration energy overhead in enterprise data centers. In [25], VM placement problem for decreasing energy consumption was studied. Goudarzi *et al*. have utilized dynamic programming and local search to place multiple copies of VMs. In [26] and [27], the authors proposed auction-based and truthful greedy mechanisms for dynamic VM provisioning and allocation. The goal in dynamic VM allocation is the same, but methodologies are different. Heuristic and greedy-based algorithms were proposed in [26] and [27]. We utilized metaheuristic GA to predict resource requirements in advance, and then proposed VM placement algorithm to maximize resource utilization and to minimize energy consumption. These studies only focus on resource allocation, which is unlike we consider dynamic resource prediction and allocation at the same time.

A few of existing literatures [28]–[30] have tackled resource prediction and allocation for cloud data center at the same time. In [28], Xiao *et al*. have proposed a new concept *skewness* to combine the unevenness in multiple resources utilization on a server. The resource utilization of VM is predicted based on the past external behaviors of VMs, and the VM consolidation is achieved by using the skewness metric to combine VMs with different resource characteristics. The authors showed that CPU load prediction is achieved but the memory prediction. Moreover, the proposed prediction method may fail to accurately forecast the turning points of resource utilization. Last, the heuristic-based algorithm is different to our metaheuristic algorithm. In [29], Huang *et al*. have proposed the M-convex VM consolidation engine to balance power cost and network cost during VM consolidation process. The resource utilization of VM is predicted based on the kernel density estimation (KDE) method, and the VM consolidation is solved based on a greedy strategy. However, the resource prediction in simulation result is unclear that how the KDE method forecasts resource usage accurately. Moreover, the authors focused on server cost and network cost, which is unlike the CPU and memory considered in our paper. Last, the used greedy VM consolidation strategy is different to the proposed GA based on the metaheuristic method in our work. In [30], Dabbagh *et al*. have proposed a framework to predict VM resource usage and place VMs according to prediction results. The resource prediction is obtained by the Wiener filter prediction approach and the proposed heuristic algorithm consolidates VMs. With experimental results, the proposed method may be unable to predict few overloads, although the number of unpredicted overloads is low. Furthermore, the authors did not show that how the heuristic algorithm predicts the CPU and memory utilization in their experiments. Last, the defined integer linear programming problem is different to the MOO problem in our work, also the heuristic-based algorithm and our metaheuristic GA. A comparison of related works is given in Table I.

In this paper, we proposed multiobjective GA to predict resource demand dynamically and to minimize energy consumption by maximizing the resource utilization of each active PM. Three objectives of the paper are listed as follows.

1) *Dynamic resource prediction*. Both CPU and memory utilization should be predicted accurately by the proposed GA, no matter what kind of utilization tendency is.
2) *Resource utilization maximization*. The resource utilization of an active PM should be maximized by using the proposed VM placement algorithm; thus, the number of active PMs can be minimized.
3) *Energy conservation*. The number of active PMs should be minimized for saving power consumption.

The contributions of this paper are summarized as follows.

1) A MOO problem in resource allocation for cloud data center is formulated while maximizing both CPU and memory of each active PM and minimizing the energy consumption of data center.
2) The proposed GA accurately predicts the CPU and memory utilization in next time slot, no matter the utilization tendency is stable raise and fall tendency or unstable fluctuation tendency.
3) The proposed VM placement algorithm reallocates VMs for the next time slot based on the forecast result of the proposed GA. By using the VM placement algorithm, the resource utilization of each active PM is maximized, and the number of active PMs is minimized.

III. Problem Formulation

A. System Architecture

The system architecture is captured in Fig. 1. Network infrastructure is composed of routers, switches, and PMs. Herein, VMs are nested in PMs. To monitor and forecast resource utilization, a management node is deployed in one separate PM. According to collected history data, the management node predicts and reallocates resource requirements for next time slot. The used system architecture can be applied to several well-known data center structures, e.g., tree structure, fat-tree, VL2, and BCube. Both of VL2 and BCube are adopted

TABLE I
COMPARISON OF RELATED WORKS

Metrics \ Literatures	Grey model [22]	Skewness [28]	M-convex [29]	Heuristic [30]	Proposed GA
Optimization method offered	X	X	O	X	O
Resource utilization prediction	O	O	O	O	O
Different tendencies of resource utilization	X	O	X	X	O
VM migration considered	O	O	O	O	O
Energy conservation	O	O	X	O	O
Dynamic resource allocation	X	O	X	O	O
Numbers of VMs and PMs	75 VMs in 15 PMs	1400 VMs in 140 PMs	1052 VMs in 550 PMs	12000 PMs	500 VMs in 100 PMs
Resources considered	CPU, memory	CPU, memory, network workload	CPU, memory	CPU, memory	CPU, memory, energy
Approach proposed	Heuristic	Heuristic	Integer programming	Heuristic	Meta-heuristic

The symbol "O" means supported and the symbol "X" means unsupported. Abbreviations: VM: Virtual Machine; PM: Physical Machine; GA: Genetic Algorithm.

The symbol "O" means supported and the symbol "X" means unsupported. Abbreviations: VM: virtual machine; PM: physical machine; GA: genetic algorithm.

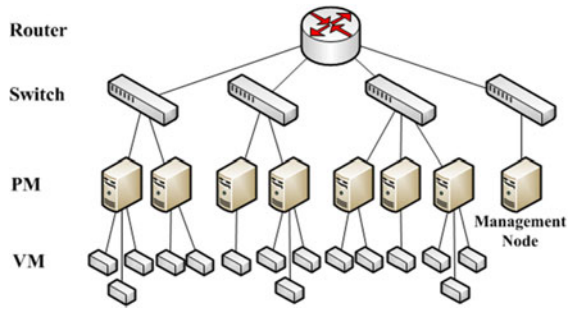


Fig. 1. System architecture.

and investigated in Microsoft's data center. These two data center structures are layered with layer 3 routers (include core routers and access routers) and layer 2 switches (include L2 aggregation switches and L2 switches). The fat-tree structure is also similar to these two structures, but it has a more concrete definition on link connection such as ary and pods. Because the system architecture in this work is a tree-based and layered architecture, it can be applied to these structures with a slight modification of the original structure.

B. MOO Problem

VM provisioning in reality should consider many objects, such as the CPU and memory utilization of VMs and PMs, construction cost, and energy cost of data center. Network operators expect to maximize the utilization of active PMs for minimizing the energy consumption of data center. In order to solve the trade-off problem, a MOO problem is formulated. The used variables and their definitions are described in Table II.

We define a binary decision variable $\mathbb{p}_j$ where $\mathbb{p}_j = 1$ if PM p_j is active and $\mathbb{p}_j = 0$ if PM p_j is shutdown. Then, another binary decision variable $\mathbb{v}_{i,j}$ is defined, where $\mathbb{v}_{i,j} = 1$ if VM v_i is allocated to PM p_j, and $\mathbb{v}_{i,j} = 0$ otherwise. Three object functions are investigated in this work, which are CPU utilization, memory utilization, and energy consumption. Assume that there are m VMs and n PMs in data center. The variable C^{avg} is the average CPU utilization of n PMs, which is formulated as follows:

$$C^{\text{avg}} = \frac{\sum_{j=1}^{n} \mathbb{p}_j p_j^{\text{cpu}}}{n} \tag{1}$$

TABLE II
DEFINITION OF SYMBOLS

Variable	Definition
$V = \{v_1, \ldots, v_m\}$	Set of VMs
$P = \{p_1, \ldots, p_n\}$	Set of PMs
$\mathbb{p}_j$	Binary decision variable of PM status
$\mathbb{v}_{i,j}$	Binary decision variable of VM allocation
$v_{i,j}^{\text{cpu}}$	CPU utilization of VM v_i in PM p_j
$v_{i,j}^{\text{mem}}$	Memory utilization of VM v_i in PM p_j
$v_{i,j}^{\text{eng}}$	Energy consumption of VM v_i in PM p_j
$v_{i,j}^{\text{bytes per request}}$	The average number of bytes used per request
$v_{i,j}^{\text{maximum bytes}}$	The maximum bytes allocated to VM v_i in PM p_j.
p_j^{cpu}	CPU utilization of PM p_j
p_j^{mem}	Memory utilization of PM p_j
p_j^{eng}	Energy consumption of PM p_j
$T_{p_j}^{\text{cpu}}$	CPU time of PM p_j
p_j^{idle}	Power consumption of PM p_j in idle state
p_j^{max}	Maximum power consumption of PM p_j
$p_j^{\text{CPU clock cycle}}$	The number of clock cycles of PM p_j
$p_j^{\text{clock rate}}$	The clock rate of PM p_j
C^{avg}	Average CPU utilization of n VMs in m PMs
C^{max}	Maximum CPU utilization of PM
M^{avg}	Average memory utilization of n VMs in m PMs
M^{max}	Maximum memory utilization of PM
E	Total energy consumption of all PMs in data center
E^{max}	Maximum energy consumption of data center

where p_j^{cpu} is the CPU utilization of PM p_j that ranges from 0 to 1. It is derived from

$$p_j^{\text{cpu}} = \mathbb{p}_j \left(\frac{\sum_{i=1}^{m} \mathbb{v}_{i,j}\, v_{i,j}^{\text{cpu}}}{m} \right) \tag{2}$$

where $v_{i,j}^{\text{cpu}}$ is the CPU utilization of VM v_i in PM p_j. It is calculated as follows:

$$v_{i,j}^{\text{cpu}} = \mathbb{p}_j \mathbb{v}_{i,j} \left(\frac{v_{i,j}^{\text{active}}}{T_{p_j}^{\text{cpu}}} + \frac{v_{i,j}^{\text{idle}}}{T_{p_j}^{\text{cpu}}} \right) * 100\%. \tag{3}$$

The variables $v_{i,j}^{\text{active}}$ and $v_{i,j}^{\text{idle}}$ represent the CPU time of VM v_i in PM p_j with active state and idle state, respectively. The variable $T_{p_j}^{\text{cpu}}$ stands for the CPU time of PM p_j, which is calculated as follows:

$$T_{p_j}^{\text{cpu}} = \mathbb{p}_j \left(\frac{p_j^{\text{CPU clock cycle}}}{p_j^{\text{clock rate}}} \right) \tag{4}$$

where $p_j^{\text{CPU clock cycle}}$ is the number of clock cycles of PM p_j and $p_j^{\text{clock rate}}$ is the clock rate as well as the clock frequency of PM p_j.

The variable M^{avg} is the average memory utilization of n PMs, which is formulated as follows:

$$M^{\text{avg}} = \frac{\sum_{j=1}^{n} \mathbb{p}_j p_j^{\text{mem}}}{n} \tag{5}$$

where p_j^{mem} is the memory utilization of PM p_j. It is derived from

$$p_j^{\text{mem}} = \mathbb{p}_j \left(\frac{\sum_{i=1}^{m} \mathbb{v}_{i,j} v_{i,j}^{\text{mem}}}{m} \right) \tag{6}$$

where $v_{i,j}^{\text{mem}}$ is the memory utilization of VM v_i in PM p_j. It is calculated as follows:

$$v_{i,j}^{\text{mem}} = \mathbb{p}_j \mathbb{v}_{i,j} \left(\frac{v_{i,j}^{\text{bytes per request}}}{v_{i,j}^{\text{maximum bytes}}} \right) * 100\%. \tag{7}$$

The notation $v_{i,j}^{\text{bytes per request}}$ is the average number of used bytes per request, and the notation $v_{i,j}^{\text{maximum bytes}}$ is the maximum bytes allocated to VM v_i in PM p_j.

The variable E is the total energy consumption of n PMs in cloud data center, which is formulated as follows:

$$E = \sum_{j=1}^{n} \mathbb{p}_j p_j^{\text{eng}} \tag{8}$$

where p_j^{eng} is the total energy consumption of PM p_j. It is calculated as follows:

$$p_j^{\text{eng}} = \mathbb{p}_j \left(\sum_{i=1}^{m} \mathbb{v}_{i,j} v_{i,j}^{\text{eng}} \right) \tag{9}$$

where $v_{i,j}^{\text{eng}}$ is the energy consumption of VM v_i in PM p_j. It is calculated as follows:

$$v_{i,j}^{\text{eng}} = \mathbb{p}_j \, \mathbb{v}_{i,j} \left[p_j^{\text{idle}} + \left(p_j^{\max} - p_j^{\text{idle}} \right) * v_{i,j}^{\text{cpu}} \right]. \tag{10}$$

The variable p_j^{idle} is the power consumption of PM p_j in idle state, and the variable $p_j^{\max}$ is the maximum power consumption of PM p_j. Although we assume that the server energy is linear proportionality with its CPU utilization in this work, the proposed framework can be applied to other server power models, e.g., the convex model of quadratic server energy in [31].

The MOO problem of resource allocation in data center is defined as follows:

$$G(x) = \begin{cases} g_1(x) = g_{\text{cpu}}(x) = \max \; C^{\text{avg}} \\ g_2(x) = g_{\text{memory}}(x) = \max \; M^{\text{avg}} \\ g_3(x) = g_{\text{energy}}(x) = \min \; E \end{cases} \tag{11}$$

subject to

$$\forall i, j, m, n \in \mathbb{N} \tag{12}$$

$$0 \le p_j^{\text{cpu}} = \mathbb{p}_j \left(\frac{\sum_{i=1}^{m} \mathbb{v}_{i,j} v_{i,j}^{\text{cpu}}}{m} \right) < C^{\max}$$
$$\forall i \in \{1 \ldots m\}; \quad \forall j \in \{1 \ldots n\} \tag{13}$$

$$0 \le p_j^{\text{mem}} = \mathbb{p}_j \left(\frac{\sum_{i=1}^{m} \mathbb{v}_{i,j} v_{i,j}^{\text{mem}}}{m} \right) < M^{\max}$$
$$\forall i \in \{1 \ldots m\}; \quad \forall j \in \{1 \ldots n\} \tag{14}$$

$$0 \le p_j^{\text{eng}} = \mathbb{p}_j \left(\sum_{i=1}^{m} \mathbb{v}_{i,j} v_{i,j}^{\text{eng}} \right) < E^{\max}$$
$$\forall i \in \{1 \ldots m\}; \quad \forall j \in \{1 \ldots n\} \tag{15}$$

$$\sum_{j=1}^{n} \sum_{i=1}^{m} \mathbb{v}_{i,j} = 1, \mathbb{v}_{i,j} \in \{0, 1\} \quad \forall i \in \{1 \ldots m\};$$
$$\forall j \in \{1 \ldots n\} \tag{16}$$

$$0 \le v_{i,j}^{\text{cpu}}, p_j^{\text{cpu}}, v_{i,j}^{\text{mem}}, p_j^{\text{mem}} \le 1 \quad \forall i \in \{1 \ldots m\};$$
$$\forall j \in \{1 \ldots n\} \tag{17}$$

$$0 \le C^{\text{avg}}, \; C^{\max}, \; M^{\text{avg}}, M^{\max} \le 1. \tag{18}$$

The objective functions in (11) are to maximize the average CPU and memory utilization of PMs, and to minimize the total energy consumption of data center. Constraint (12) guarantees that these numbers are natural number. In constraints (13) and (14), the summation of CPU and memory utilization of VMs in an active PM should be equal to the CPU and memory utilization of the PM, but less than its maximum CPU and memory capacity. It equals to zero if PM p_j is shutdown. In constraint (15), the summation of energy consumption of VMs in an active PM is equal to the energy consumption of the PM, and it is restricted to less than the maximum energy consumption of data center. If PM p_j is shutdown, the energy consumption equals to zero. Constraint (16) guarantees that one VM is allocated and placed to one PM only. The CPU and memory utilization of each VM and PM are restricted to the range from 0 to 1 in constraint (17). Constraint (18) guarantees both of the average and maximum CPU and memory utilization is greater than or equal to 0 and less than or equal to 1.

IV. Proposed Algorithms

A. Resource Prediction by GA

There are many metaheuristic algorithms for solving an MOO problem, such as simulated annealing (SA) [32], Tabu search (TS) [33], ant colony optimization (ACO) [34], particle swarm optimization (PSO) [35], and GA [36]. SA is a physical process that changes molecule metal by heating and cooling. The object is to get new molecular arrangement. However, SA is easy to fall into local search so that it is inappropriate for finding the optimal solution. TS is provided with memory ability, which records previous move and maintains Tabu list. TS is greater and superior to SA because it has lower probability of acquiring regional optimal solution. However, excessive memory requirements may lead to miss the optimal solution. ACO depends on the quantity

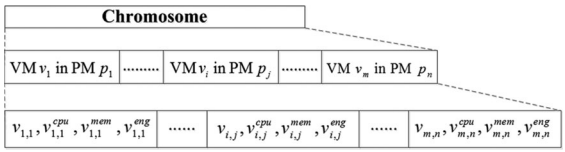


Fig. 2. Designed chromosome.

of pheromones to solve optimization problems. However, it is not suitable for recursively predicting the resource utilization of data center. In [37], Tsai *et al*. have done a complete survey work of metaheuristic scheduling methods for clouds. PSO is suitable for solving continuous optimization problems but discrete problems. We consider that the population-based GA is more appropriate to forecast resource utilization. In addition, the multiobjective GA provides better solutions than single objective GA. Since the defined problem in this work is an MOO problem, we utilize the concept of multiobjective GA to solve the problem.

We solve the MOO problem of resource allocation for data center based on the prediction results of the proposed GA. Crossover operation facilitates the obtainment of optimal solution, and mutation operation prevents falling to regional optimal solution. The offspring inherits the gene of superior chromosome in crossover operation, and eliminates poor chromosomes through competition. GA is more efficient and faster than the Brute force search in terms of search time and solution space. The acquired solution of GA after a complete evolution process usually outperforms heuristic algorithms.

The designed chromosome is captured in Fig. 2. Chromosomes are designed as VM placement results of data center in a time slot. In encoding operation, the resource utilization of each VM encapsulates chromosome every two minutes. The designed chromosome is composed of m substrings since there are m VMs in data center. Each substring notes that VMs are under the jurisdiction of which PM. Then, each substring describes the resource utilization of VM including its CPU utilization, memory utilization, and energy consumption. According to placement result and resource status, the fitness value of each chromosome can be calculated.

The fitness function of a chromosome is designed to fit in with the *survival of the fittest*. It represents the deviation between prediction and reality, which is designed as follows:

$$F_c\ (t) = \alpha * f_{\text{cpu}}\,(t) + \beta * f_{\text{mem}}\,(t) + \gamma * f_{\text{eng}}\,(t) \tag{19}$$

where

$$f_{\text{cpu}}\ (t) = |C^{\text{avg}}\,(t') - C^{\text{avg}}\,(t)| \tag{20}$$

$$f_{\text{mem}}\ (t) = |M^{\text{avg}}\,(t') - M^{\text{avg}}\,(t)| \tag{21}$$

$$f_{\text{eng}}\ (t) = \frac{|E\,(t') - E\,(t)|}{E^{\max}} * 100\% \tag{22}$$

$$\alpha + \beta + \gamma = 1. \tag{23}$$

The fitness function of a chromosome in time slot t is denoted as $F_c(t)$, which is composed of CPU utilization $f_{\text{cpu}}(t)$, memory utilization $f_{\text{mem}}(t)$, and energy consumption $f_{\text{eng}}(t)$. Parameters α, β, and γ are the weights of three metrics. To ensure the importance of three metrics is the same, we set α, β, and γ as the same weight to guarantee the fairness of them, i.e., $(\alpha, \beta, \gamma) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Note that we can adjust one of these parameters to a lower weight if the importance of this metric is lower than others. In (20) and (21), the deviations of CPU and memory utilization between time slot t' and time slot t are evaluated. The deviation of energy consumption between time slot t' and t is evaluated in (22). Note that a smaller fitness value represents the better chromosome, and vice versa. It implies the smaller prediction error between predicted and realistic resource utilization. Therefore, it is provided with the higher probability of being an offspring as well as a prediction result during the evolution process.

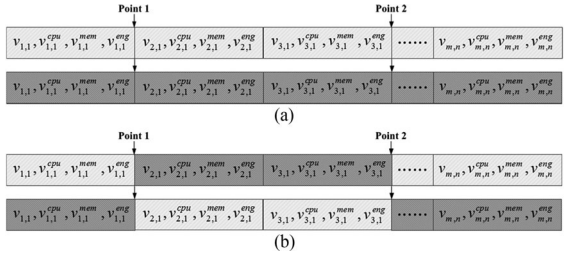


Fig. 3. Two-point crossover operation. (a) Two chromosomes before crossover operation. (b) Offspring after crossover operation.

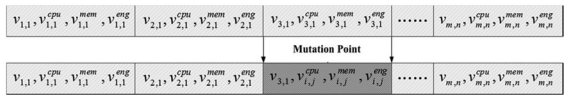


Fig. 4. Mutation operation.

In crossover operation, parent chromosomes mate with each other and produce child chromosomes as well as offspring. Two-point crossover approach with crossover probability p_c is utilized, which is illustrated with Fig. 3. Note that these two points are selected randomly. In Fig. 3(a), two parent chromosomes in different time slots exchange their substrings within these two points. Then, offspring is generated and captured in Fig. 3(b). After crossover, superior offspring is selected based on its fitness value. A chromosome with the smaller fitness value represents that it has the better prediction accuracy in terms of CPU utilization, memory utilization, and energy consumption. Therefore, the more accurate resource prediction result can be acquired.

GA may fall into local optimal solutions when the fitness values of parent chromosomes are unfavorable. A chromosome is able to mutate itself with mutation probability p_m in mutation operation, so that regional solutions can be avoided. It not only increases the diversity of chromosome but also generates new string. An example of mutation operation is illustrated in Fig. 4. In the example, the information of substring 3 is mutated. The VM v_3 in PM p_1 mutates itself with the information of VM v_i in PM p_j for generating a better offspring. To guarantee the fairness of mutation operation, mutated points and information are selected arbitrarily.

The flowchart of the proposed GA for resource prediction is shown as Fig. 5. In the beginning, the information of resource utilization in previous time slot is regarded as historical data. Then, these historical data are collected into the mating pool. Note that we expect to achieve dynamic resource prediction and allocation; GA is continuously executed in the management node. Two chromosomes are randomly selected to be the parent chromosomes. In crossover operation, parent chromosomes mate with each other under crossover probability p_c. If the fitness value of offspring is better than parent chromosomes, it will be put into group pool as a new generation. Otherwise, GA executes mutation operation with mutation probability p_m. If the fitness value of offspring is better than parent strings after

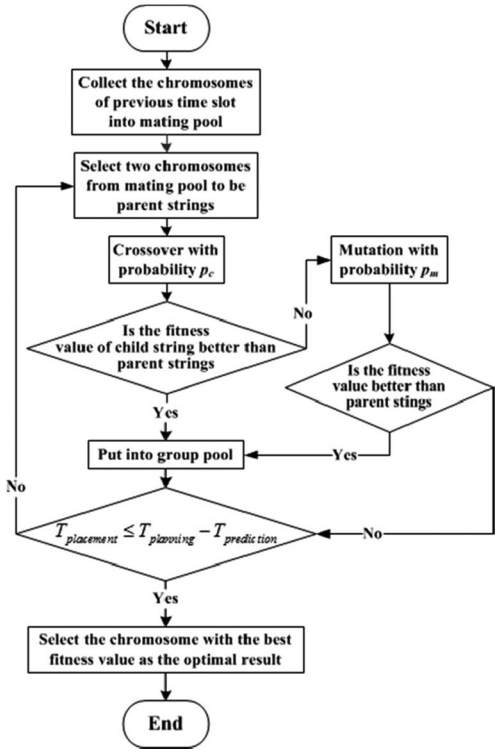


Fig. 5. Flowchart of the proposed GA.

mutation operation, it will be put into group pool as a new generation. Once an offspring is placed into the group pool, the time condition of placement process should be examined.

The time complexity of GA is mainly decided by its encoding, selection, fitness calculation, crossover and mutation operations with GA's generation number and population size, that is $\Theta(\text{Generation} * \text{Population} * \Theta(\text{Fitness}) * (\Theta(\text{Selection}) + \Theta(\text{Crossover}) + \Theta(\text{Mutation})))$. The time complexity of the proposed GA is computed by $\Theta(GN\Theta(Nm)(\Theta(N\ m \log m) + p_c * \Theta(Nm) + p_m * \Theta(m)))$, where G is the number of generation, N is the population size, m is the number of substring, p_c and p_m are crossover and mutation probability. Since a worse prediction result as well as a result with higher fitness value is discarded by tournament selection, the time complexity of selection operation can be reduced to $O(Nm)$. We do acknowledge that the time complexity of the proposed GA is higher than other heuristic algorithms, i.e., the Grey model compared in simulations. Although the proposed GA has higher computation complexity, it is executed and is restricted within the time of a time slot. With a time slot, the time for executing the proposed framework is denoted as T_{slot}, which is defined as follows:

$$T_{\text{slot}} = T_{\text{prediction}} + T_{\text{placement}} \tag{24}$$

where $T_{\text{prediction}}$ is the time for predicting resource utilization and $T_{\text{placement}}$ is the time for VM placement after resource prediction. If the time for VM placement $T_{\text{placement}}$ is less than or equal to $T_{\text{slot}} - T_{\text{prediction}}$, GA pauses its evolution process and selects an offspring with the smallest fitness value from group pool as the optimal prediction result. After that, the management node terminates the GA process and starts to execute the VM placement algorithm. Otherwise, GA continues the evolution process until it reaches the time for VM placement. It should be noticed that although GA terminates its evolution process when the time approaches VM placement, but history data collection is uninterrupted. In addition, the proposed GA does not uninterruptedly search the Pareto front of each time slot since the time for resource prediction $T_{\text{prediction}}$ is limited and restricted to not exceed the time slot T_{slot}.

Algorithm 1: VM Placement Algorithm.

Input: V, P, $C^{\max}$, $M^{\max}$, $E^{\max}$
01 $\Omega_1 = v_m, \Omega_2 = p_n, \Delta = \phi$
02 **while** $\Omega_2 \neq \phi$
03 **repeat**
04 $y = f(j) = \max_{j \in \{1 \ldots n\}} [(C^{\max} - p_j^{\text{cpu}}) + (M^{\max} - p_j^{\text{mem}})]$
05 $y^* = \arg\max f(j)$
06 $S = f_d(\Omega_2, p_j, y^*)$
07 $\Delta = \Delta \cup S[1]$
08 **if** $(p_{S[n]}^{\text{cpu}} < C^{\max})$ and $(p_{S[n]}^{\text{mem}} < M^{\max})$
09 $v_{i,j} = v_{i,n}$
10 $p_{S[1]} = \phi$
11 **end if**
12 **until** $v_{i,j} = \Delta$
13 **end while**

B. VM Placement Algorithm

After using GA to forecast resource utilization, the VM placement algorithm is proposed to reallocate VMs for maximizing resource utilization. The procedure of the proposed VM placement algorithm is shown in Algorithm 1. In lines 4–7, the difference in CPU and memory utilization between the maximum value of data center and PM p_j is sorted in a descending order list S. For example, the CPU and memory utilization of PM p_1 are 20% and 30%, and PM p_2 are 50% and 60%. The PM p_1 is ahead of PM p_2 in the descending order list S. The PM with minimum CPU and memory utilization is put in the first order. Besides, let notation $S[i]$ be the ith order in list. If two PMs have the same degree, the orders of them in the ordered list are arbitrary. In lines 8–11, VMs in the PM with minimum utilization are migrated to the PM in the last order of list S. Herein, the resource utilization of PM $p_{S[n]}$ should not exceed the maximum CPU utilization $C^{\max}$ and memory utilization $M^{\max}$, which is guaranteed in line 8. If the resource utilization of PM exceeds resource limitation, the remaining VMs will be migrated to a PM with the second-last order of list S. Once VMs in a PM are fully migrated to other PMs, the PM is shutdown for decreasing the energy consumption of data center.

According to the prediction result of GA, the VM placement algorithm aims to maximize the average of CPU and memory utilization and minimize the total energy consumption of data center with the least active PMs. The placement algorithm is executed at the end of each time slot after prediction. It should be finished and accomplished before next time slot.

V. Simulation Results

A. Simulation Setup and Parameters

The prediction result of the proposed GA is compared against the result of Grey model [22]. We have designed two tendencies of resource utilization, which are stable raise and fall (or simply stable tendency throughout this paper) and unstable fluctuation

TABLE III
SIMULATION PARAMETERS

Parameter	Value
Number of PM	20
Number of VM	100
Number of time slot	30
Crossover rate (p_c)	0.9
Mutation rate (p_m)	0.1
Time slot (T_{slot})	1 (hour)
Prediction time ($T_{\text{prediction}}$)	50 (minute)
Placement time ($T_{\text{placement}}$)	10 (minute)

(or simply unstable tendency throughout this paper). Prediction results are evaluated with performance metrics in terms of execution time and evolutionary generation. Simulation parameters are captured in Table III. The simulation-based analysis was conducted on a PC with 2.93 GHz Intel(R) i7 CPU and 8 GB of memory running Microsoft Windows 7 and using MATLAB [38] to construct the proposed GA in a data center with 20 PMs and 100 VMs. Various research works [39]–[41] have pointed out that the cost of VM migration is a vital issue to resource allocation in cloud environments. Resource allocation too often will lead to unnecessary migration cost. Therefore, we consider that the resources of VMs and PMs are allocated hourly in this work. There are 30 time slots and each of them is an hour. In addition, each time slot is divided into 50 min for resource prediction and 10 min for VM placement. A higher crossover rate accelerates to generate new chromosomes, and a higher mutation rate has a higher opportunity of avoiding locally optimal solutions [42]. To enrich the diversity of planning results and facilitate the escape in a local optimum, the probabilities of crossover and mutation are set to 0.9 and 0.1, respectively. The setting is familiar to GA in simulation [43], [44].

B. Resource Prediction Result

The execution time and generation of GA are investigated when slot is timed from 0 to 30. The results of execution time under stable and unstable tendencies are captured in Fig. 6(a) and (b), and results of generation under stable and unstable tendencies are captured in Fig. 6(c) and (d). Because evolutionary algorithms need historical data for training evolution process, GA collects information of VMs and PMs in the first and second time slot. GA is able to find optimal solution within 50 min before the twenty time slot, but it needs more time for resource prediction after the 20 time slot. At that time, GA is constrainedly terminated due to time limitation. The remaining 10 min in any slots are reserved for VM placement process. In Fig. 6(c) and (d), it can be observed that the generation of GA increases when the number of time slot increases. This is attributed to the fact that plenty of historical data lead to the greater number of generation in mating pool. In addition, GA under unstable tendency yields the larger number of generation due to the greater diversity of chromosomes in different time slots.

The solution obtained by GA for resource prediction under stable and unstable tendency is captured in Fig. 7(a) and (b), respectively. In these two figures, yellow dots represent normal solutions as well as offspring and red dots represent the optimal solutions in 30 time slots. It can be observed that the number of prediction results under stable tendency is less than unstable tendency. In addition, the optimal prediction results of stable tendency almost aggregate and align in a line, and the optimal prediction results of unstable tendency are distributed in solution space. This is attributed to the fact that unstable tendency has the greater diversity of chromosomes.

The prediction results of GA and Grey model are investigated when slot is timed from 0 to 30. The prediction results of resource utilization and energy consumption under stable raise and fall tendency are captured in Fig. 8. The average CPU and memory utilization of PMs are depicted as Fig. 8(a) and (b), and the energy consumption of data center is depicted as Fig. 8(c). In stable tendency, realistic data stably increase in the first half of time slots, and stably decrease in the last half-time slots. It can be observed that GA yields more accurate prediction results compared to the Grey model. The prediction results of GA almost approximate realistic data in terms of three objective functions. Grey model predicts the future tendency based on existing data tendency. Therefore, it leads to critical prediction error at the turning point of tendency. Moreover, the prediction error will become worse once it misses the tendency of realistic data. Because Grey model only utilizes the data array of previous few time slots, its prediction error is unable to correct in future time slots. The GA is capable of predicting resource utilization in next time slot, and achieves prediction error less than 5% compared to realistic data.

The prediction results of unstable fluctuation tendency are captured in Fig. 9. The prediction results of average CPU utilization and memory utilization are shown in Fig. 9(a) and (b). The real resource utilization in cloud data center is similar to the unstable fluctuation tendency. Because the relevance between historical data of unstable tendency is smaller, it is more difficult to predict resource utilization in a fluctuation tendency. That is why resource prediction and allocation are of vital importance. It can be observed that both of GA and Grey model have obvious prediction error under unstable tendency compared to the stable tendency. However, GA yields the smaller prediction deviation compared to Grey model. It is attributed to the evolution process of GA on the basis of historical data in mating pool. The prediction result of energy consumption is captured in Fig. 9(c). It can be observed that GA accurately predicts the energy consumption of data center, but the Grey model fails to catch the fluctuation after the fourth time slot and becomes worse. Unlike the Grey model, the prediction result of the proposed GA tightly follows the realistic data. The accurate prediction results significantly facilitate and enhance the performance of VM allocation and provisioning.

C. VM Placement Results

The placement result of the proposed VM placement is investigated. The used simulation parameters are the same with Table II. There are 100 VMs allocated within 20 PMs during 30 time slots. The VM placement algorithm is executed in the last 10 min of each time slot. According to previous time slots, GA predicts the resource utilization in next time slot. The VM placement algorithm allocates VMs for maximizing the CPU and memory utilization of active PMs and minimizing the energy consumption of cloud data center in next time slot.

The performance of the VM placement algorithm under stable tendency in different time slots is captured in Fig. 10. The average CPU and memory utilization of stable tendency after

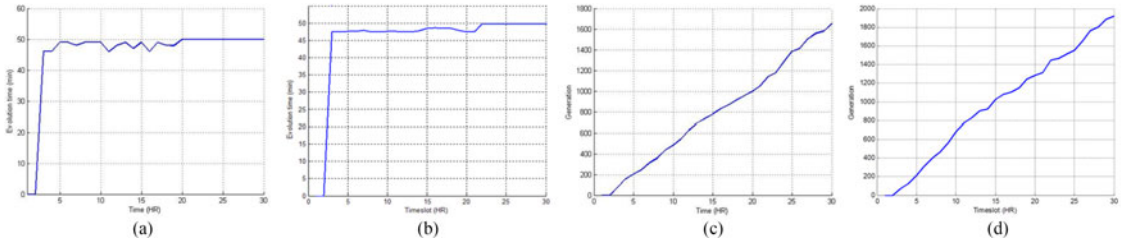


Fig. 6. Execution time and generation of GA in the different time slots. (a) Execution time of stable tendency. (b) Execution time of unstable tendency. (c) Generation of GA in stable tendency. (d) Generation of GA in unstable tendency.

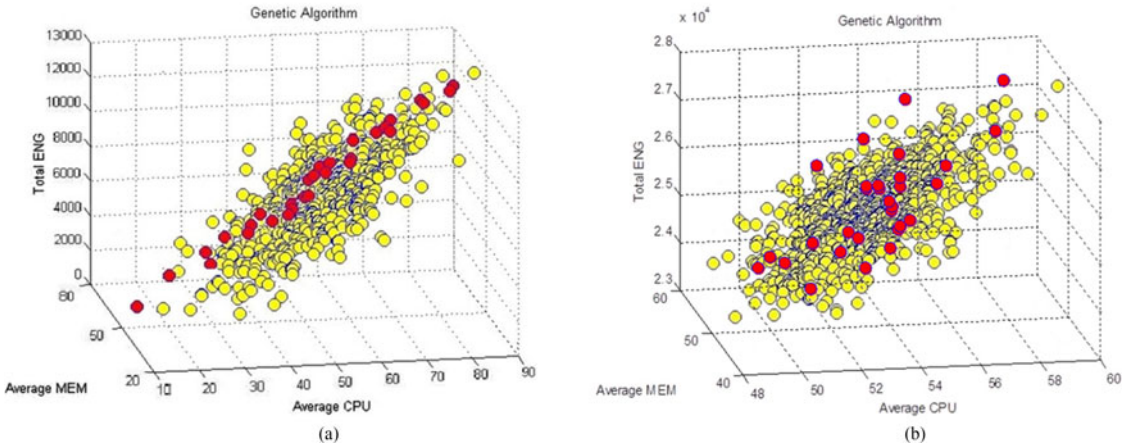


Fig. 7. Optimal solution obtained by GA for resource prediction. (a) Solutions obtained by GA under stable tendency. (b) Solutions obtained by GA under unstable tendency.

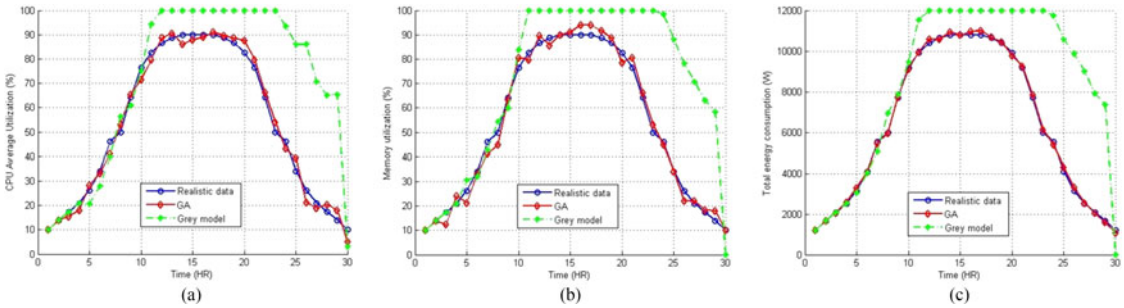


Fig. 8. Prediction results of the GA and Grey model under stable raise and fall tendency. (a) CPU utilization. (b) Memory utilization. (c) Energy consumption.

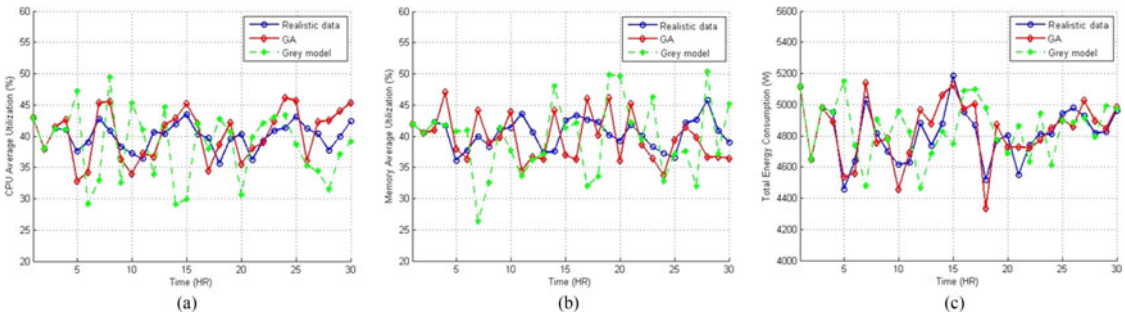


Fig. 9. Prediction results of the GA and Grey model under unstable fluctuation tendency. (a) CPU utilization. (b) Memory utilization. (c) Energy consumption.

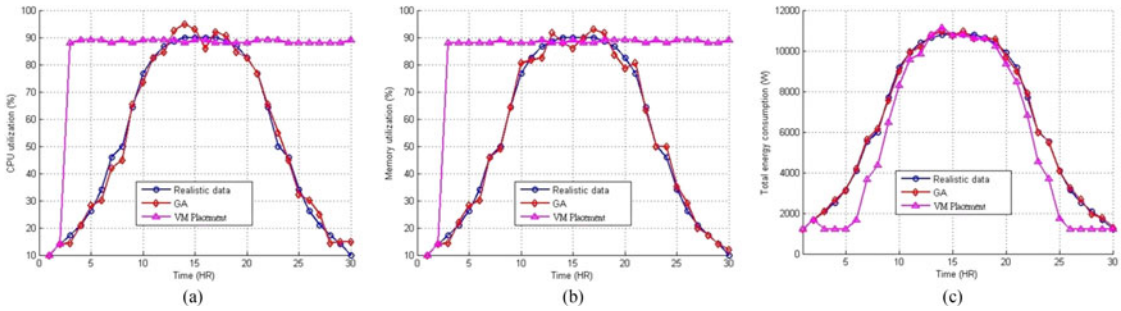


Fig. 10. Performance of VM placement results under stable raise and fall tendency. (a) CPU utilization. (b) Memory utilization. (c) Energy consumption.

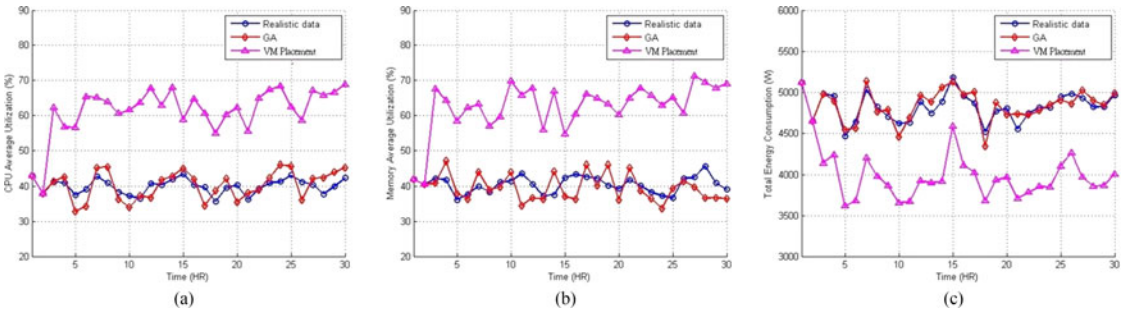


Fig. 11. Performance of VM placement results under unstable fluctuation tendency. (a) CPU utilization. (b) Memory utilization. (c) Energy consumption.

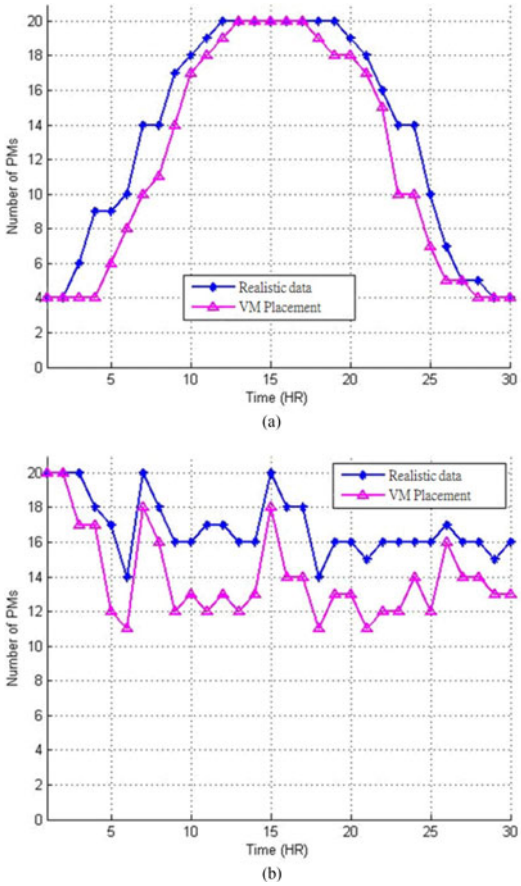


Fig. 12. Number of active PMs after VM placement. (a) Number of active PMs in the stable tendency. (b) Number of active PMs in the unstable tendency.

VM placement are shown as Fig. 10(a) and (b), and the energy consumption is shown as Fig. 10(c). It can be observed that VM placement algorithm improves both of CPU and memory utilization to 90% after the third time slot. This is attributed to the fact that the proposed GA predicts resource utilization accurately. The VM placement algorithm is capable of migrating VMs from a PM with low utilization to a PM with high utility, so that the average of CPU and memory utilization can be improved. In addition, the PM with low resource utilization is shut down after all VMs are migrated. Therefore, the energy consumption of data center can be reduced. The proposed VM placement algorithm achieves the higher average resource utilization of PMs and the lower energy consumption of data center than others schemes.

The performance of the VM placement algorithm under unstable tendency in different time slots is captured in Fig. 11. The CPU and memory utilization of unstable tendency after VM placement are shown as Fig. 11(a) and (b). Both of CPU and memory utilization are almost increased to 60% after the third time slot. In Fig. 11(c), the energy consumption of data center can be reduced after resource allocation by the VM placement algorithm. It can be observed that the placement result varies with the tendency of realistic data owing to the accurate prediction of GA. The difficult resource allocation problem of unstable utilization can be solved by the proposed approach.

The number of active PMs after VM placement is investigated. The results of stable and unstable tendency are shown in Fig. 12(a) and (b). It can be observed that VM placement algorithm efficiently decreases the number of active PMs in both resource tendencies. The lower energy consumption is attributed to the fact that less active PMs with higher resource utilization are achieved. The improvement in number of active PMs under unstable tendency is more obvious than that under stable

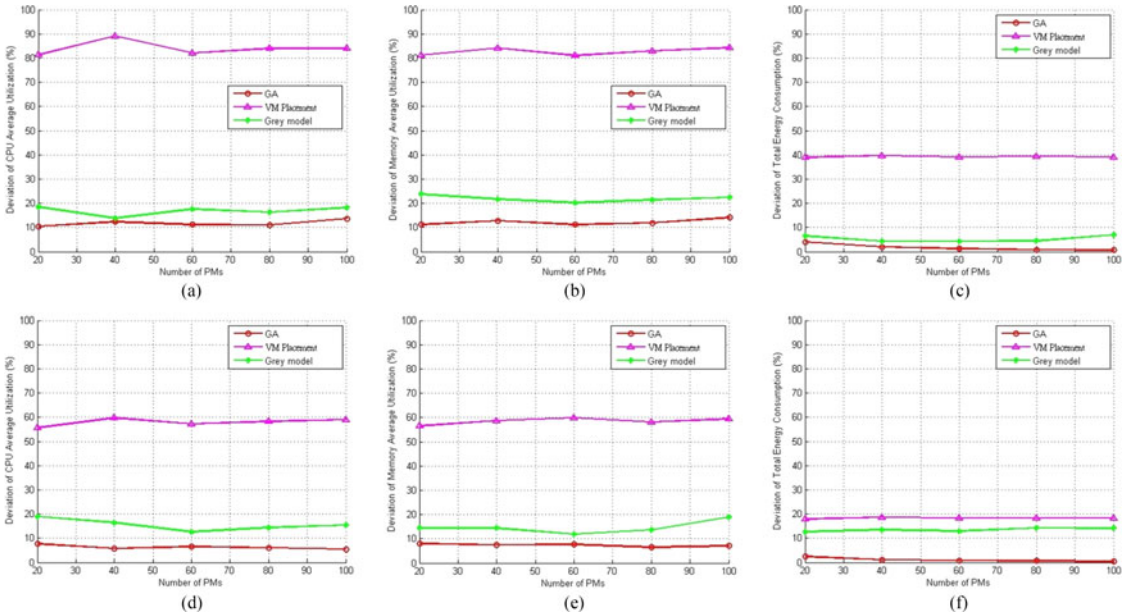


Fig. 13. Prediction and placement deviation under unstable tendency with different proportion of PMs to VMs. (a) CPU utilization with ratio 1:3. (b) Memory utilization with ratio 1:3. (c) Energy consumption with ratio 1:3. (d) CPU utilization with ratio 1:5. (e) Memory utilization with ratio 1:5. (f) Energy consumption with ratio 1:5.

tendency. Because the average resource utilization of unstable tendency is lower than the stable tendency. The lower average utilization implies that there is enough room for resource allocation. In other words, the proposed VM placement algorithm has superior achievement when the resource utilization of data center is lower.

The prediction deviation of GA and Grey model with different ratios of PMs to VMs is investigated while varying the number of PMs from 20 to 100. In Fig. 13(a)–(c), the ratio of PMs to VMs is 1:3. Therefore, the minimum scale is 60 VMs in 20 PMs and the maximum scale is 300 VMs in 100 PMs. The proportion of PMs to VMs is 1:5 in Fig. 13(d)–(f). With the ratio of PMs to VMs is 1:5, the minimum scale is 100 VMs in 20 PMs and the maximum scale is 500 VMs in 100 PMs. The benchmark as well as deviation equals to zero is the numerical data of realistic resource utilization and energy consumption. It can be observed that GA yields the smaller prediction deviation than the Grey model with any ratio of PMs to VMs and the deviation decreases when the ratio increases. It means that the prediction accuracy of GA is superior to the Grey model. In addition, the prediction deviation of GA becomes smaller while increasing the ratio of PMs to VMs. This result is mainly attributed to the fact that there are more VMs in a higher proportion of PMs to VMs; more VMs lead to more historical data for resource prediction. Therefore, GA is capable of finding a better solution through the evolution process. It is also observed that the VM placement algorithm with the higher ratio of PMs to VMs yields higher resource utilization and lower energy consumption compared to the results of the lower proportion of PMs to VMs. Because there is more space for VM allocation in the lower ratio, the proposed VM placement algorithm is capable of increasing resource utilization and decreasing energy consumption. In summary, the proposed GA is suitable to allocate heterogeneous resources of a large-scale data center.

The deviation of the proposed VM placement algorithm from the optimal placement result is captured in Fig. 14. The optimal placement result is obtained by using the brute-force search

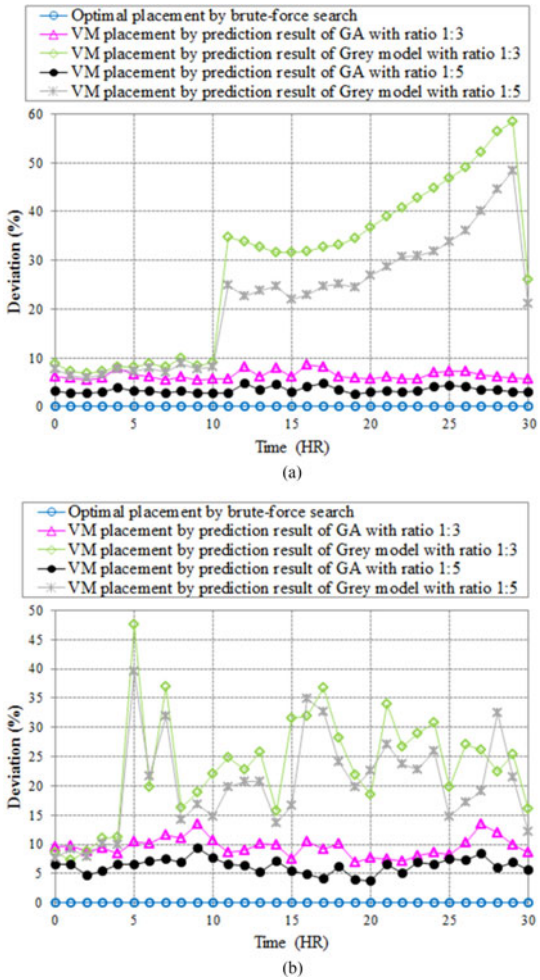


Fig. 14. Deviation of the proposed algorithm from the optimal solution. (a) Deviation under stable raise and fall tendency. (b) Deviation under unstable fluctuation tendency.

method based on the resource utilization of historical data. The brute-force search method exhaustively searches all possible VM placement results and selects a solution with the best fitness value as optimal solution. We adopt this placement result to be the benchmark as well as the line at deviation equals to 0%. The placement result under stable raise and fall utilization tendency is captured in Fig. 14(a), and the unstable fluctuation tendency is in Fig. 14(b). The placement results with different ratios of PMs to VMs are also captured, i.e., 1:3 and 1:5. To investigate the impacts of prediction errors on the VM placement results, the VM placement algorithm is tested based on two different prediction results, i.e., the prediction results of the proposed GA and the Grey model.

It can be observed that the VM placement algorithm with GA's prediction result has less deviation from the optimal result, which is better than that of the VM placement algorithm with Grey model's prediction result. This is attributed to the fact that the proposed GA yields the better prediction accuracy compared to the Grey model. Therefore, the VM placement result with GA's prediction is more close to the optimal placement result. It can also be observed that the placement result with a higher ratio has the lower deviation compared to the result with the ratio 1:3. This is mainly attributed to the fact that there are more VMs can be optimized by the proposed GA. In other words, more VMs result in a higher probability of shutting down more PMs. In addition, more VMs lead to a larger quantity of historical data for resource prediction, thereby the proposed GA yields the higher prediction accuracy. Last, it can be observed that the placement result under stable raise and fall tendency has the lower deviation from optimal result than that of the result under unstable fluctuation tendency. This is attributed to the fact that both of the proposed GA and the Grey model yield worse prediction accuracy under unstable resource utilization tendency. Even though it is hard to predict the resource utilization under unstable tendency, the placement result with GA's prediction still approaches the optimal result due to the better prediction result of the proposed GA.

VI. Conclusion

Targeting the challenging issues of cloud resource optimization, we have proposed a new prediction approach based on GA for enhancing prediction accuracy in cloud data center. We have also proposed a VM placement algorithm for improving the average of resource utilization and reducing the energy consumption of data center based on the prediction results from GA. Simulation results showed that the proposed GA is superior in prediction accuracy to the Grey model in terms of CPU utilization, memory utilization, and energy consumption no matter in stable or unstable utilization tendency. In addition, the proposed VM placement algorithm improves the resource utilization of PMs and decreases the energy consumption of data center with less active PMs. We have also showed that the proposed approach under lower utilization yields more obvious improvements on resource utilization and energy conservation. Moreover, the prediction approach becomes more accurate in higher ratio of PMs to VMs due to the greater quantity and diversity of historical data. In the future, we expect to test the proposed GA with the Google data center traces to verify its prediction accuracy. In addition, we intend to compare the proposed GA against other metaheuristics, and realize the proposed approach to realistic cloud environments.

Acknowledgment

The authors would like to thank J.-J. Jheng for her valuable comments and suggestions to prepare this manuscript.

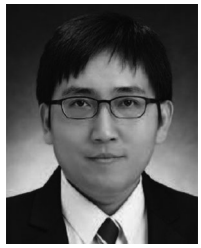
References

[1] M. Armbrust *et al.*, "A view of cloud computing," *ACM Commun.*, vol. 53, no. 4, pp. 50–58, Apr. 2010.
[2] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Proc. Grid Comput. Environ. Workshops*, Nov. 2008, pp. 1–10.
[3] M. F. Bari *et al.*, "Data center network virtualization: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 909–928, May 2013.
[4] A. Bahga and V. K. Madisetti, "Analyzing massive machine maintenance data in a computing cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 10, pp. 1831–1843, Oct. 2012.
[5] Y.-M. Chu, N.-F. Huang, and S.-H. Lin, "Quality of service provision in cloud-based storage system for multimedia delivery," *IEEE Syst. J.*, vol. 8, no. 1, pp. 292–303, Mar. 2014.
[6] F.-H. Tseng, C.-Y. Chen, L.-D. Chou, H.-C. Chao, and J.-W. Niu, "Service-oriented virtual machine placement optimization for green data center," *Mobile Netw. Appl.*, vol. 20, no. 5, pp. 556–566, Oct. 2015.
[7] C.-C. Lin, H.-H. Chin, and D.-J. Deng, "Dynamic multiservice load balancing in cloud-based multimedia system," *IEEE Syst. J.*, vol. 8, no. 1, pp. 225–234, Mar. 2014.
[8] F. Legillon, N. Melab, D. Renard, and E.-G. Talbi, "Cost minimization of service deployment in a public cloud environment," *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops Ph.D. Forum*, May 2013, pp. 491–498.
[9] Z. Zheng, X. Wu, Y. Zhang, M. R. Lyu, and J. Wang, "QoS ranking prediction for cloud services," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1213–1222, Jun. 2013.
[10] W. A. Tan, Y. Sun, G. Z. Lu, and T. Wang, "A trust service-oriented scheduling model for workflow applications in cloud computing," *IEEE Syst. J.*, vol. 8, no. 3, pp. 868–878, Sep. 2014.
[11] G. Sun, V. Anand, D. Liao, C. Lu, X. Zhang, and N.-H. Bao, "Power-efficient provisioning for online virtual network requests in cloud-based data centers," *IEEE Syst. J.*, vol. 9, no. 2, pp. 427–441, Jun. 2015.
[12] C.-W. Tsai and J. J. P. C. Rodrigues, "Metaheuristic scheduling for cloud: A survey," *IEEE Syst. J.*, vol. 8, no. 1, pp. 279–291, Mar. 2014.
[13] A. Ganapathi, Y. Chen, A. Fox, R. Katz, and D. Patterson, "Statistics-driven workload modeling for the cloud," in *Proc. IEEE Int. Conf. Data Eng. Workshops*, Mar. 2010, pp. 87–92.
[14] A. Khan, X. Yan, S. Tao, and N. Anerousis, "Workload characterization and prediction in the cloud: A multiple time series approach," in *Proc. IEEE Netw. Oper. Manage. Symp.*, Apr. 2012, pp. 1287–1294.
[15] N. Roy, A. Dubey, and A. Gokhale, "Efficient autoscaling in the cloud using predictive models for workload forecasting," in *Proc. IEEE Int. Conf. Cloud Comput.*, Jul. 2011, pp. 500–507.
[16] X. Wang, Z. Sheng, S. Yang, and V. C. M. Leung, "Tag-assisted social-aware opportunistic device-to-device sharing for traffic offloading in mobile social networks," *IEEE Wireless Commun.*, vol. 23, no. 4, pp. 60–67, Aug. 2016.
[17] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment," *IEEE Netw.*, vol. 9, no. 2, pp. 56–61, Mar./Apr. 2015.
[18] E. Zohar, I. Cidon, and O. Mokryn, "PACK: Prediction-based cloud bandwidth and cost reduction system," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, pp. 39–51, Feb. 2014.
[19] S. Di, C.-L. Wang, and F. Cappello, "Adaptive algorithm for minimizing cloud task length with prediction errors," *IEEE Trans. Cloud Comput.*, vol. 2, no. 2, pp. 194–207, Apr.–Jun. 2014.
[20] S. Di, D. Kondo, and W. Cirne, "Host load prediction in a Google compute cloud with a Bayesian model," in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, Nov. 2012, pp. 1–11.
[21] J. J. Prevost, K. Nagothu, B. Kelley, and M. Jamshidi, "Prediction of cloud data center networks loads using stochastic and neural models," in *Proc. Int. Conf. Syst. Syst. Eng.*, Jun. 2011, pp. 276–281.
[22] J.-J. Jheng, F.-H. Tseng, H.-C. Chao, and L.-D. Chou, "A novel VM workload prediction using Grey forecasting model in cloud data center," in *Proc. Int. Conf. Inf. Netw.*, Feb. 2014, pp. 40–45.
[23] A. Alasaad, K. Shafiee, H. M. Behairy, and V. C. M. Leung, "Innovative schemes for resource allocation in the cloud for media streaming applications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 1021–1033, Apr. 2015.

[24] M. M. Hossain, J.-C. Huang, and H.-H. S. Lee, "Migration energy-aware workload consolidation in enterprise clouds," in *Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2012, pp. 405–410.
[25] H. Goudarzi and M. Pedram, "Energy-efficient virtual machine replication and placement in a cloud computing system," in *Proc. IEEE Int. Conf. Cloud Comput.*, Jun. 2012, pp. 750–757.
[26] S. Zaman and D. Grosu, "A combinatorial auction-based mechanism for dynamic VM provisioning and allocation in clouds," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 129–141, Jul.–Dec. 2013.
[27] M. M. Nejad, L. Mashayekhy, and D. Grosu, "Truthful greedy mechanisms for dynamic virtual machine provisioning and allocation in clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 2, pp. 594–603, Feb. 2015.
[28] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1107–1117, Jun. 2013.
[29] Z. Huang and D. H. K. Tsang, "M-convex VM consolidation: Towards a better VM workload consolidation," *IEEE Trans. Cloud Comput.*, vol. 4, no. 4, pp. 415–428, Oct.–Dec. 2016.
[30] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "An energy-efficient VM prediction and migration framework for overcommitted clouds," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2016.2564403.
[31] C.-H. Hsu and S. W. Poole, "Revisiting server energy proportionality," in *Proc. Int. Conf. Parallel Process.*, Oct. 2013, pp. 834–840.
[32] S. Bandyopadhyay, S. Saha, U. Mauilk, and K. Deb, "A simulated annealing-based multiobjective optimization algorithm: AMOSA," *IEEE Trans. Evol. Comput.*, vol. 12, no. 3, pp. 269–283, Jun. 2008.
[33] J.-Q. Li, Q.-K. Pan, and Y.-C. Liang, "An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems," *Comput. Ind. Eng.*, vol. 59, no. 4, pp. 647–662, Nov. 2010.
[34] C.-F. Tsai, C.-W. Tsai, and C.-C. Tseng, "A new and efficient ant-based heuristic method for solving the traveling salesman problem," *Expert Syst.*, vol. 20, no. 4, pp. 179–186, Sep. 2003.
[35] C.-W. Tsai, K.-W. Huang, M.-C. Chiang, and C.-S. Yang, "A fast particle swarm optimization for clustering," *Soft Comput.*, vol. 19, no. 2, pp. 321–338, Feb. 2015.
[36] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[37] C.-W. Tsai and J. P. C. Rodrigues, "Metaheuristic scheduling for cloud: A survey," *IEEE Syst. J.*, vol. 8, no. 1, pp. 279–291, Mar. 2014.
[38] *MATLAB*. 2014. [Online]. Available: http://www.mathworks.com/products/matlab/
[39] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migration in clouds: A performance evaluation," in *Proc. Int. Conf. Cloud Comput.*, Dec. 2009, pp. 254–265.
[40] W. Dargie, "Estimation of the cost of VM migration," in *Proc. Int. Conf. Comput. Commun. Netw.*, Aug. 2014, pp. 1–8.
[41] O. Rana, "The costs of cloud migration," *IEEE Cloud Comput.*, vol. 1, no. 1, pp. 62–65, May 2014.
[42] S. A. Stanhope and J. M. Daida, "Optimal mutation and crossover rates for a genetic algorithm operating in a dynamic environment," in *Proc. Int. Conf. Evol. Program. VII*, Mar. 1998, pp. 693–702.
[43] X. Wu *et al.*, "Retracted: Scheduling optimization of the RFID tagged explosive storage based on genetic algorithm," in *Proc. Int. Conf. Grid Pervasive Comput.*, 2013, pp. 358–366.
[44] N. R. Siriwardene and B. J. C. Perera, "Selection of genetic algorithm operators for urban drainage model parameter optimization," *Math. Comput. Model.*, vol. 44, nos. 5/6, pp. 415–429, Sep. 2006.

Fan-Hsun Tseng received the Ph.D. degree in computer science and information engineering from National Central University, Taoyuan, Taiwan, in 2016.

He is currently with Assistant Professor with the Department of Technology Application and Human Resource Development, National Taiwan Normal University, Taipei, Taiwan. His research interests include software-defined radio, sensor network applications, cloud computing, LTE, and LTE-Advanced networks.



Xiaofei Wang (M'13) received the M.S. and Ph.D. degrees from the School of Computer Science and Engineering, Seoul National University, Seoul, South Korea, in 2008 and 2013, respectively.

He is currently a Professor with the School of Computer Science and Technology, Tianjin University, Tianjin, China. He was a Postdoctoral Fellow with the School of Electrical and Computer Engineering, The University of British Columbia, Canada, from 2014 to 2016. He has authored or co-authored more than 60 papers in the areas including cooperative cell caching, social-aware multimedia service in cloud computing, and traffic offloading in mobile networks.

Dr. Wang was the recipient of the IEEE ComSoc Fred W. Ellersick Prize in 2017.

Li-Der Chou (M'95) received the M.S. and Ph.D. degrees in electronic engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan, in 1991 and 1995, respectively.

He is currently a Distinguished Professor with the Department of Computer Science and Information Engineering and the Director of the Computer Center, National Central University, Taoyuan, Taiwan. He also serves as the Director of the Board of the Taiwan Network Information Center. He was the Deputy Director General of the National Center for High-Performance Computing, Taiwan, from 2013 to 2016. He is the holder of 5 U.S. and 16 Taiwan patents. His research interests include SDN/NFV/SFC, vehicular networks, network management, broadband wireless networks, and Internet services, and he has authored or co-authored more than 200 papers in these areas.

Dr. Chou was the recipient of seven Best Paper Awards and four Excellent Paper Awards from international and domestic conferences. He was also a recipient of two Gold Medal Awards and four Silver Medal Awards in international invention shows held in Geneva, Switzerland, Moscow, Russia, London, U.K., and Taipei, Taiwan.

Han-Chieh Chao (SM'04) received the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1989 and 1993, respectively.

He is currently a Professor with the Department of Electrical Engineering, National Dong Hwa University, Hualien, Taiwan, where he also serves as the President. He is also with the Department of Computer Science and Information Engineering and the Department of Electronic Engineering, National Ilan University, Yilan, Taiwan, College of Mathematics and Computer Science, Wuhan Polytechnic University, Wuhan, China, and the Fujian University of Technology, Fuzhou, China.

Dr. Chao serves as the Editor-in-Chief for the *Institution of Engineering and Technology Networks*, the *Journal of Internet Technology*, the *International Journal of Internet Protocol Technology*, and the *International Journal of Ad Hoc and Ubiquitous Computing*. He is a Fellow of the Institution of Engineering and Technology (Institution of Electrical Engineers) and a Chartered Fellow of the British Computer Society.

Victor C. M. Leung (S'75–M'89–SM'97–F'03) is a Professor of electrical and computer engineering and Holder of the TELUS Mobility Research Chair with the University of British Columbia (UBC), Vancouver, BC, Canada. He has co-authored more than 1000 technical papers in archival journals and refereed conference proceedings, several of which have won Best Paper Awards. His research is in the areas of wireless networks and mobile systems.

Dr. Leung is a Fellow of the Royal Society of Canada, Canadian Academy of Engineering, and Engineering Institute of Canada. He is serving on the Editorial Boards of the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, IEEE WIRELESS COMMUNICATIONS LETTERS, and several other journals. He has provided leadership to the Technical Program Committees and Organizing Committees of numerous international conferences. He was the recipient of the 1977 APEBC Gold Medal, Natural Sciences and Engineering Research Council of Canada Postgraduate Scholarships from 1977 to 1981, a 2012 UBC Killam Research Prize, and an IEEE Vancouver Section Centennial Award.