Multi-objective Optimization in Cloud Brokering Systems for Connected Internet of Things

Teerawat Kumrai, Kaoru Ota, Mianxiong Dong, Jay Kishigami, and Dan Keun Sung

***Abstract*—Currently, more than 9 billion things are connected in the Internet of Things (IoT). This number is expected to exceed 20 billion in the near future, and the number of things is quickly increasing, indicating that numerous data will be generated. It is necessary to build an infrastructure to manage the connected things. Cloud computing has become important in terms of analysis and data storage for IoT. In this paper, we consider a cloud broker, which is an intermediary in the infrastructure that manages the connected things in cloud computing. We study an optimization problem for maximizing the profit of the broker while minimizing the response time of the request and the energy consumption. A multi-objective particle swarm optimization (MOPSO) is proposed to solve the problem. The performance of the proposed MOPSO is compared with that of a genetic algorithm and a random search algorithm. The results show that the MOPSO outperforms a well-known genetic algorithm for multi-objective optimization.**

***Index Terms*—Internet of Things, Cloud Broker, Particle Swarm Optimization, Multi-objective Optimization.**

I. Introduction

INTERNET of Things (IoT) is an Internet technology that connects machines and tools with each other via the Internet or wireless technologies, e.g., Wi-Fi, Radio-Frequency Identification (RFID), Bluetooth Low Energy (BLE), ZigBee and so on, as shown in Figure 1. The things and data that are connected in IoT outnumber the world population in 2011. Currently, 9 billion things are connected, and this number could exceed 24 billion by 2020 [1]. A lot of data will be generated in the future because the number of things is rapidly increasing. Therefore, big data and cloud computing will play important roles in terms of analysis and data storage. Cloud computing (CC) is a service delivered through data centers, which are based on virtualization technologies.

CC provides and delivers the service to clients or users on demand [2]. The clients could use software, systems, and computing resources of the cloud service provider through the Internet anywhere and anytime, as shown in Figure 2. Moreover, some clients who use a mobile device (e.g., smartphone and tablet) could detect and use the computing resources on other mobile devices through device-to-device (D2D) communications [3]. The service in the CC is classified into the following three categories. The first service is Software as a Service (SaaS) in which clients can run software through the Internet without installing and maintaining the software. The second service is Platform as a Service (PaaS) to provide a platform allowing clients to develop, run, and manage applications virtualization servers. The third service is Infrastructure as a Service (IaaS) in which organizations can outsource support operations, including hardware, storage, servers, and networks. Service providers will own equipment, work responsibilities, and maintenance by clients, who have the option to pay by actual usage.

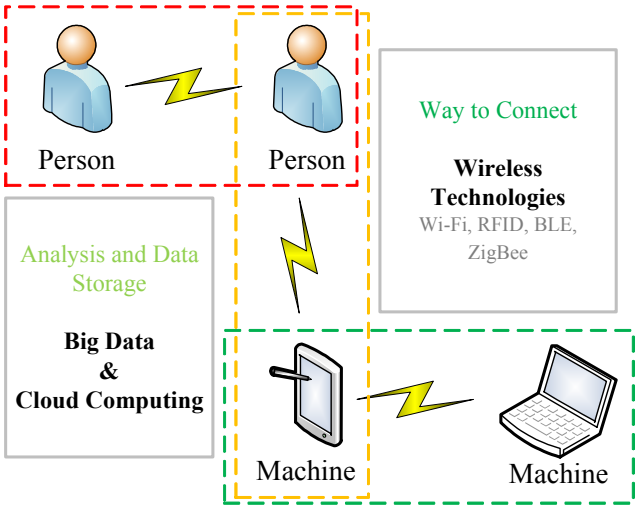


Fig. 1. Internet of Things: Things can be connected and communicated with each other via wireless technologies. Big data and cloud computing will play important roles in terms of analysis and data storage.

IoT in the future poses a challenge whereby a huge number of things and data are connected through the Internet. An infrastructure should be developed to manage and support this quantity of connected things. To overcome this challenge, we utilize an intermediary of IaaS in CC to manage the connected things and data in IoT. In CC, a request from clients sometimes can be submitted to the cloud through the intermediary, which resides between the clients and cloud service providers. We use it (called a cloud broker hereafter) to optimize resource selection in CC. Cloud brokering matches the requests from clients with offers provided by the service providers. In a sense, the cloud broker is expected to simply find the best deal between the clients and service providers with the maximum profit. On the other hand, the clients are expected to minimize

T. Kumrai, K. Ota, M. Dong and J. Kishigami are with Muroran Institute of Technology, Japan. Email: {15096013, ota, mxdong, jay}@mmm.muroran-it.ac.jp.

Dan Keun Sung is with Korea Advanced Institute of Science and Technology (KAIST), Korea. Email: dksung@kaist.ac.kr.

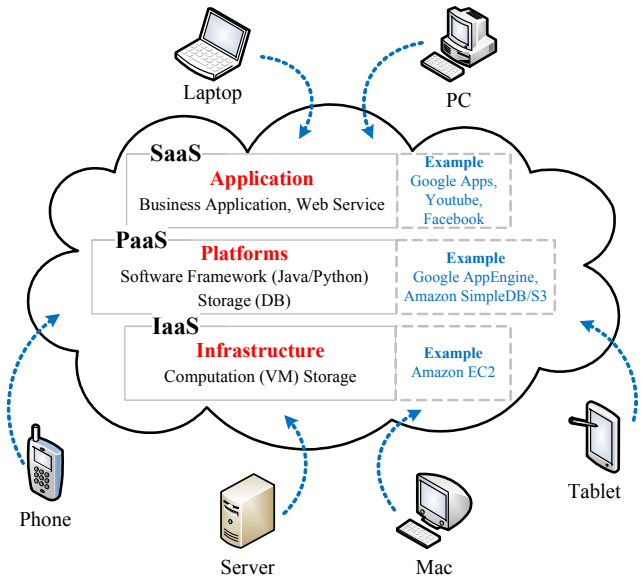


Fig. 2. Cloud Computing Architecture

the response time when they submit their requests to the service providers. In addition, reducing energy consumption in CC is an important issue because of a rapid growth in cloud services. Thus, the problem that we address in this paper is how to minimize the response time of requests and the total energy consumption in CC while maximizing the profit of the cloud broker.

Many studies focused on cloud brokering in CC [4]–[6]. In [4], A multi-objective genetic algorithm and a greedy heuristic algorithm were proposed to optimize the energy consumption, CO2 emissions, and deployment cost. A new framework, CLOUD Resource Broker, was proposed in [5]. The proposed framework is integrated with a particle swarm optimization-based resource allocation scheme and a deadline-based job scheduling. The objectives are to minimize the execution time and cost and maximize the number of jobs that are completed within a deadline. This framework was compared with a genetic algorithm (GA), a rank-based allocation mechanism and ant colony optimization. In [6], the authors proposed a new evolutionary algorithm (EA) to maximize the broker profit of virtual machine subletting in CC. The algorithm used is a parallel hybrid EA and was compared with a greedy heuristic algorithm by using veritable data from the cloud providers. However, neither method solved the response time, energy consumption or the profit of the cloud broker simultaneously. Additionally, they did not propose a particle swarm optimization (PSO) scheme, which converges easily and has a lower complexity than a genetic algorithm (GA). Thus, the main motivation behind this research is to formulate, design and develop the PSO for the cloud brokering system to reduce the response time of requests and the total energy consumption in CC while increasing the profit of the cloud broker.

Therefore, we investigate the response time of requests, the energy consumption, and the profit of the broker. We consider a cloud brokering problem as a multi-objective optimization problem with three objectives: minimizing the response time of requests from users, minimizing the energy consumption in CC, and maximizing the profit of the broker. We use a multi-objective particle swarm optimization (MOPSO) scheme to find the optimal solution for this multi-objective optimization problem because MOPSO can be used to find a Pareto-optimal solution for the cloud brokering.

The main contributions of this paper can be summarized as follows:

- We investigate a cloud brokering problem in a cloud IoT system by considering the energy consumption, broker profit, and response time.
- A novel optimization problem is formulated to solve how to maximize the broker profit and to minimize the energy consumption of the system and response time of users. We propose a PSO scheme to solve a single-objective problem and also design a MOPSO scheme to solve a multi-objective problem.
- The MOPSO is proposed to solve the formulated problems. We take extensive simulations to evaluate the MOPSO. The performance of the proposed MOPSO is compared with that of a well-known GA and a random search algorithm.

In this paper, we review the related work in Section II. Then, we describe a system model and propose an optimization problem in Section III. The standard PSO and MOPSO are described in Section IV. The performance of the proposed methods is evaluated through computer simulations in Section V. Finally, we draw conclusion in SectionVI.

II. Related Work

Cloud computing resource management problems such as scheduling, load balancing and mapping are NP-hard. It implies that there are no optimal algorithms to solve these problems. However, these problems have already been addressed by several methods of computational intelligence such as the EA, ant colony optimization, and fuzzy and neural networks. In this section, we describe the related work on the resource management optimization problem in CC.

Several studies proposed EAs for solving the resource management optimization problems in CC [4]–[6]. In [4], the authors investigated a scheme to reduce energy consumption, which is an important problem in CC. They proposed a multi-objective GA and a greedy heuristic algorithm. The proposed algorithms are used to optimize the energy consumption, CO2 emissions and deployment cost. A performance comparison of the MOGA and the greedy heuristic was presented. The MOGA outperforms the greedy heuristic in terms of energy consumption and CO2 emissions.

The EAs have been used to find optimal solutions of various optimization problems in a wireless sensor network (WSN) [7]–[9]. A GA was proposed to optimize two objectives in [7]. The objectives consist of the connectivity of sensor nodes and the coverage in a k-covered hotspot area. In [8], the authors proposed a genetic algorithm for solving a node placement problem in WSN. The algorithm involves

maximizing the coverage and the network lifetime. *EVOLT* was proposed in [9]. It is an EA with new genetic operators that are *aging*, *age-based crossover*, *aged-based mutation*, and *fitness-based crossover*. The *EVOLT* was proposed to solve the problem using high-dimensional quality of service (QoS) optimization for power grid communication networks.

Several studies proposed a novel algorithm/framework for solving optimization problems in wireless networks [10], [11]. Recently, a complex alliance strategy with multi-objective optimization of coverage was proposed in [10]. It is a novel algorithm that could improve the effectiveness of sensor node coverage and the network lifetime in WSNs. The authors also presented a scheme to calculate coverage expectation and proportionality. In [11], the authors focused on an optimal forwarding problem in mobile ad-hoc networks (MANETs). A novel game-theoretic framework was proposed for examining the optimal forwarding problem of a two-hop f-cast relay algorithm in MANETs. The proposed algorithm indicates the relationship between the forwarding behaviors and the final throughput capacity.

Several studies proposed a novel model/framework for solving problems in cloud computing [12]–[16]. CSAM-IISG was proposed in [12]. It is an imperfect information Stackelberg game with hidden Markov model (HMM) for a cloud resource allocation model in a cloud computing environment. The proposed model can increase the profit of both the service providers and the applicant. In [13], the authors proposed a service framework and a pricing strategy for a multi-cloud environment. The proposed framework can provide streaming big data computing service and maximize the profits of the multi-cloud intermediary. The profits of the proposed pricing strategy are higher than other pricing strategies. Recently, a cloud-centric multi-level authentication as a service was proposed in [14]. The proposed approach is for secure public safety networks in the cloud and IoT devices. Scalability and time constraints are considered to demonstrate the effectiveness of the approach. In [15], a multi-cloud architecture called MCES was proposed. Smart evacuation services are deployed in multi-cloud providers. This system can tolerate pressure more than single cloud service under emergency environment. In [16], an online-deduplication mechanism based on energy efficiency storage system was proposed for virtual machines (VM) storage. The authors also designed a deduplication selection algorithm to minimize the storage energy consumption. The proposed mechanism can reduce both of the redundant data blocks without service interruption and the energy consumption.

Moreover, several studies on a Fiber-Wireless (FiWi) network focusing on the network energy efficiency, scalability and reliability were reviewed in [17]. The authors summarized the progress in radio-over-fiber and radio-and-fiber based on FiWi networks, QoS provisioning schemes, energy efficiency schemes, scalability-improving technique, reliability-enhancing schemes, and industry standardization activities and new trends for future work. In [18], a novel location recognition approach was proposed. The proposed approach includes supporting location-aware services, especially check-in services. Global and hybrid positioning systems are used in the proposed method. The authors also proposed the concept of radio FingerPrint for their method.

A new EA, a parallel hybrid EA was proposed in [6]. The objective is to maximize the broker profit in the cloud system. The performance of the algorithm was compared with that of a greedy heuristic algorithm. The results show that the proposed algorithm outperforms the greedy heuristic algorithm in terms of the profit values. A framework called CLOUD Resource Broker was proposed in [5] with particle swarm optimization-based resource allocation and deadline-based job scheduling. The objectives are to minimize the execution time and cost and maximize the number of jobs that are completed within a deadline. The performance of the proposed framework was compared with that of a genetic algorithm (GA), ant colony optimization (ACO) and rank-based allocation (RBA) mechanism. The results show that the proposed framework is able to complete jobs within the deadline.

A grouping genetic algorithm (GGA) was proposed in [19]. The GGA works with fuzzy multi-objective evaluation. The authors proposed a control system that consist of two levels. The system manages the mapping of workloads to virtual machines (VMs) and VMs to physical resources. They considered the minimization of wasted resources, power consumption and the temperature of a cloud system as three optimization objectives. The performance of the proposed GA was compared with that of two algorithms, and four bin-packing algorithms and two single-objective approaches are used.

PSO was proposed for solving various problems in CC in [20], [21]. In [20], the authors proposed an adaptive power-aware virtual machine provisioner as a novel metascheduler. They used self-adaptive particle swarm optimization (SAPSO) to solve a virtual machine placement problem. They presented a performance comparison of standard PSO, multi-ensemble particle swarm optimization and SAPSO using five experiments. In the first experiment, they presented a performance comparison of detecting and tracking an optimal target server. Next, they analyzed the number of failures in VM provisioning. The rate of failure in VM provisioning with fixed and variable evaporation factors was analyzed using the third experiment. They demonstrated the impact of exploiting power-saving states along with dynamic voltage frequency scaling (DVFS) in VM provisioning. Finally, a performance comparison of the power trade-offs was presented. [21] proposed a PSO-based heuristic scheme to minimize computation and communication costs for workflow scheduling in cloud environments. The performance of the proposed algorithm is compared with that of a greedy best resource selection algorithm.

Several studies proposed ant colony optimization for solving various problems in CC [22]–[24]. In [22], the authors investigated a virtual machine placement problem in a CC environment. They proposed a multi-objective ant colony algorithm for improving the power efficiency and resource utilization. The performance algorithm is compared with that of a multi-objective genetic algorithm and two single-objective algorithms. The results show that the proposed algorithm is better than the other algorithms. A new cloud scheduler

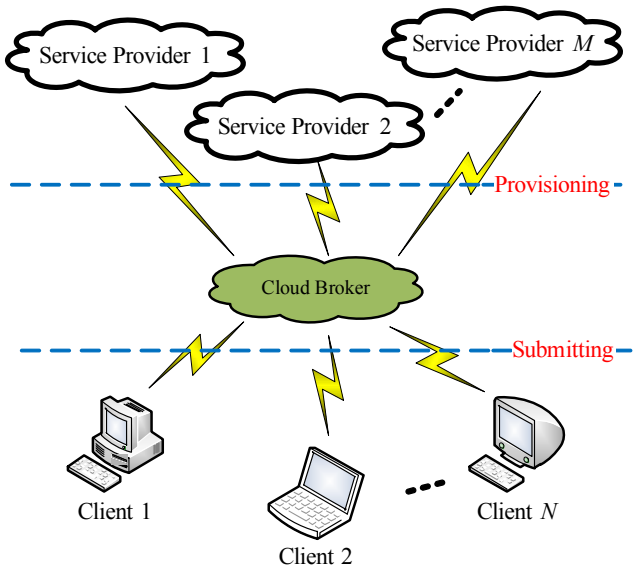


Fig. 3. Cloud Brokering Model. The model consists of N clients, M cloud service providers and one cloud broker.

based on ant colony optimization was proposed in [23] for executing parameter sweep experiments in clouds. They also formulated a problem as minimizing the weighted flowtime and makespan. In [24], the authors investigated a workload placement problem in CC. They proposed a multi-dimensional bin-packing problem and ant colony optimization to compute the placement dynamically. They compared the performance of their algorithm with that of a greedy algorithm such as First-Fit Decreasing.

In this paper, we apply multi-objective particle swarm optimization to a cloud brokering system. We consider it as an optimization problem with the following three objectives: maximizing the profit of the cloud broker and minimizing the response time of requests and the energy consumption. More details are presented in Sections III and IV.

III. Problem Statement

In this section, we describe the problem statement of cloud brokering in CC. The cloud model in this paper is an IaaS. As shown in Figure 3, our model consists of N clients, one cloud broker and M cloud service providers. The cloud broker has to find the best configuration between the clients and the cloud service providers. We use the set $U = u_1, ..., u_N$ to denote N clients and the set $S = s_1, ..., s_M$ to denote M service providers in the model. Each service provider has a limited capacity for handling the requests from clients and the total number of handling requests in the service provider needs to be greater than the number of requests from the a client. In order to describe the process of service providers, we introduce a binary variable b_{ij} with $i = 1, ..., N$ and $j = 1, ..., M$ as follows:

$$b_{ij} = \begin{cases} 1 & , if\ s_j \text{ handles the request from } u_i \\ 0 & , otherwise \end{cases} \quad (1)$$

Clients are expected to complete their jobs in a minimal time when they submit requests to the cloud broker and service providers. Therefore, we consider the response time of requests from clients. We set L_{ij} as the latency between client i and service provider j. It can be measured as $L_{ij} = CT - AT$, where CT is the current time and AT is the arrival time of a request from client i at service provider j. When the service provider receives a request from a client, the service provider has to spend time T_j to execute the request. Thus, the first objective is the minimization of the response time (RT) of requests. It is formulated as follows:

$$RT = \sum_{i=1}^{N} \sum_{j=1}^{M} b_{ij}(L_{ij} + T_j) \quad (2)$$

A client submits his/her request to a service provider through the cloud broker. The cloud broker manages and finds the best solution for the client's satisfaction. Meanwhile, the broker is expected to make a profit from the task. Therefore, the profit of the broker is considered as the second objective. We set P_i as the price from client i and C_j as the cost of service provider j. Thus, the second objective is the maximization of the profit (P) of the cloud broker, formulated as follows:

$$P = \sum_{i=1}^{N} \sum_{j=1}^{M} b_{ij}(P_i - C_i) \quad (3)$$

To execute the request from the client, the service provider has to complete the job with minimal energy consumption. Thus, we consider the energy consumption to be an important issue in CC as an objective. We assume that E_j is the energy consumption that service provider j used to execute the job. The total energy consumption of all the service providers is formulated as follows:

$$E = \sum_{i=1}^{N} \sum_{j=1}^{M} b_{ij} \cdot E_j \quad (4)$$

The last objective is to minimize the total energy consumption of the system. With the above three objectives, we consider the optimization problem of the cloud broker as a single-objective and multi-objective optimization problems. We describe the single-objective optimization problem in Section III-A and describe the multi-objective optimization problem in Section III-B.

A. Single-Objective Optimization Problem

In the beginning, we consider the three objectives as a single-objective optimization problem, and we state the utility function of the cloud broker as follows:

$$U = \omega_1 RT + \omega_2 E - \omega_3 P \quad (5)$$

where ω_1, ω_2 and ω_3 are the weighting factors of the response time of requests, the profit of the cloud broker and the total energy consumption of the system, respectively. The sum of weights is equal to one ($\omega_1 + \omega_2 + \omega_3 = 1$). RT is the response time of requests of the system that is calculated using Eq. (2). E is the total energy consumption of the system that is calculated using Eq. (4). P is the profit of the cloud broker that

is calculated using Eq. (3). Hence, the optimization problem of the cloud broker is to minimize the utility function as follows:

$$\underset{U,S}{\text{Minimize}}\ U = \omega_1 RT + \omega_2 E - \omega_3 P \tag{6}$$

$$\text{Subject to}\ \sum_{i=1}^{N} x_{ij} \le A_j \tag{7}$$

$$\sum_{j=1}^{M} b_{ij} \ge R_i \tag{8}$$

$$A_j, R_i \ge 0 \quad ; i = 1, ..., N; j = 1, ..., M \tag{9}$$

where R_i is the number of requests from client i and A_j is the capacity of the service provider j. Constraint (7) indicates whether the total number of requests from clients to service provider j is less than or equal to the capacity of service provider j. Next, constraint (8) indicates whether the total number of requests from client i to all service providers is greater than or equal to the number of requests from client i. Finally, constraint (9) indicates whether the number of requests and the capacity are positive values.

However, it is very difficult to find the best value for the weights ω_1, ω_2 and ω_3. Thus, we consider and describe the three objectives as a multi-objective optimization problem in the next subsection.

B. Multi-objective Optimization Problem

To address the assignment problem of the weight values in the single-objective problem, we consider the cloud brokering problem as a multi-objective optimization problem and state the optimization problem of the cloud broker as follows:

$$\underset{U,S}{\text{Minimize}}\ RT = \sum_{i=1}^{N}\sum_{j=1}^{M} b_{ij}(L_{ij} + T_j) \tag{10}$$

$$\underset{U,S}{\text{Minimize}}\ E = \sum_{i=1}^{N}\sum_{j=1}^{M} b_{ij} \cdot E_j \tag{11}$$

$$\underset{U,S}{\text{Maximize}}\ P = \sum_{i=1}^{N}\sum_{j=1}^{M} b_{ij}(P_i - C_i) \tag{12}$$

$$\text{Subject to}\ \sum_{i=1}^{N} x_{ij} \le A_j \tag{13}$$

$$\sum_{j=1}^{M} b_{ij} \ge R_i \tag{14}$$

$$A_j, R_i \ge 0 \quad ; i = 1, ..., N; j = 1, ..., M \tag{15}$$

where R_i is the number of requests from client i and A_j is the capacity of the service provider j. Constraints (13), (14) and (15) have the same meaning as the constraints (7), (8) and (9), respectively.

Moreover, the authors in [25] show that the scheduling problem is a NP-hard problem. Therefore, we can conclude that the cloud brokering problem is NP-hard. The NP problem could be solved by using a meta-heuristic algorithm (evolutionary algorithm). In this paper, we use multi-objective particle swarm optimization to solve the cloud brokering problem. The details of PSO and MOPSO are described in the next section.

IV. Particle Swarm Optimization

This section describes the process of particle swarm optimization (PSO). PSO seeks the Pareto-optimal schedule for the cloud brokering system. The cloud brokering system consists of N clients, one cloud broker and M service providers. The objectives are to minimize the response time of requests, to minimize the energy consumption and to maximize the profit of the cloud broker. Our proposed system operates for a smart grid in terms of energy consumption monitoring, management and efficiency. The PSO runs on a cloud broker. PSO performs its optimization process to find the best configuration between the clients and the cloud service providers. Three objective values and the set of solutions are provided to the decision makers when the PSO is finished.

A. A Single-Objective PSO

PSO was designed by Kennedy, Eberhart and Shi in 1995 [26] by imitating the motion of a flock of birds. Thus, it is a population-based optimization tool. The PSO can be applied to solve various optimization problems. The population or swarm in PSO represents the set of potential solutions. Each individual or a particle in the population represents a solution position in the search space. Particles move repeatedly through the d-dimensional space to find a new position with the best fitness value.

The particles in the swarm are represented by a position vector $\vec{x_l} = (x_{l1}, x_{l2}, ..., x_{lk})$ with the l-th particles in k-dimensions. In this case, we assume that the position vector is the set of N clients and the set of M service providers, as $\vec{x} = (u_1, ..., u_N, s_1, ..., s_M)$. The velocity vector is represented by $\vec{v_l} = (v_{l1}, v_{l2}, ..., v_{lk})$. The best position of the particle and the best position of the swarm are represented by $\vec{p_l} = (p_{l1}, p_{l2}, ..., p_{lk})$ and $\vec{g_l} = (g_{l1}, g_{l2}, ..., g_{lk})$, respectively. The PSO starts by randomly generating particles to form an initial swarm. At each iteration t, the PSO computes the updated velocity vector as follows:

$$v_{lk}(t+1) = w v_{lk}(t) + c_1 r_1[p_{lk} - x_{lk}(t)] + c_2 r_2[g_{lk} - x_{lk}(t)] \tag{16}$$

where c_1 and c_2 are the learning factors called the coefficient of the self-recognition component and the coefficient of the social component, respectively. w is an inertia weight. r_1 and r_2 are the random numbers that are uniformly distributed in the interval 0 to 1.

After calculating the updated velocity, the positions of the particles are updated as follows:

$$x_{lk}(t+1) = x_{lk}(t) + v_{lk}(t+1) \tag{17}$$

where l is the number of particles and k denotes the dimension of the particles. The PSO terminates its optimization processes when the number of iterations reaches the maximum limit or the minimum objective function error is satisfied.

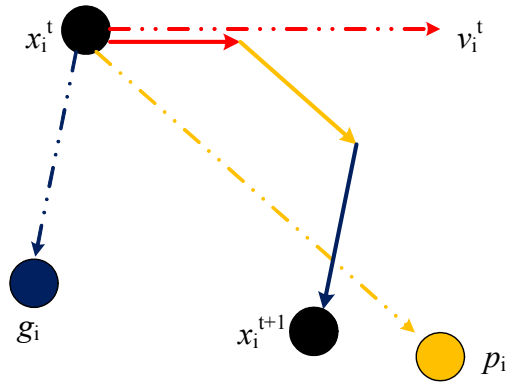


Fig. 4. Example of position updates and velocity in particle swarm optimization

Figure 4 shows an example of position updates and velocity in particle swarm optimization. The pseudocode for the particle swarm optimization is shown in Algorithm 1.

Algorithm 1 Particle Swarm Optimization

main
- Initialize the parameters of the particle swarm
- Randomly set the position and velocity of all particles
- Set each particle's pBest to the particle position
- Set gBest to the randomized particle position

repeat
  for each *particle*
    do
      - Calculate the particle's updated velocity by Eq. (16)
      - Calculate the particle's position by Eq. (17)
      - Calculate the fitness value by Eq. (6)
      if Fitness of the new position > pBest **then** - Set pBest as a new position
  - Select the particle position with the best fitness value as gBest
until the termination criterion is met

B. Multi-objective PSO

We here consider a multi-objective optimization problem. Then, the standard PSO needs to be modified for solving the problem. The multi-objective particle swarm optimization (MOPSO) was presented in [27]. The MOPSO seeks to find a set of solutions called the Pareto set.

In the beginning, the initial swarm is randomly generated. Then, a set of gBest is initialized by using non-dominated particles from the swarm. The set of gBest is stored in an external archive. At each iteration, a gBest is selected, and the positions of the particles are updated. The turbulence operators are applied in MOPSO after updating the position. The set of gBest is updated after all the processes of all the particles have finished.

The MOPSO terminates its processes when the number of iterations reaches the maximum limit or the minimum objective function error is satisfied. Algorithm 2 shows the pseudocode for the MOPSO. The processes are marked with ***italicized and bold text*** that show the difference from the standard PSO.

Algorithm 2 Multi-objective Particle Swarm Optimization

main
- Initialize the parameters of the particle swarm
- Randomly set the position and velocity of all particles
- Set each particle's pBest to the particle position
- Set gBest to the randomized particle position
- ***Initialize the set of gBest in an external archive***

repeat
  for each *particle*
    do
      - ***Select gBest***
      - Calculate the particle's updated velocity by Eq. (16)
      - Calculate the particle's position by Eq. (17)
      - Calculate the fitness value by Eqs. (10), (11) and (12)
      - ***Turbulence operators***
      if Fitness of the new position > pBest **then** - Set pBest as a new position
  - Select the particle position with the best fitness value as the gBest
  - ***Update the set of gBest in an external archive***
until the termination criterion is met
Report results in the external archive

V. Performance Evaluation

In this section, we present the setup of our simulation. Then, we analyze the performance of the MOPSO for the cloud brokering system that optimizes the three objectives described in Section III.

We compare the performance of the MOPSO with that of GA and random search algorithm.

A. Simulation Setup

It is assumed that the simulated cloud brokering is used to find the best deals between 5 clients and 7 service providers with the maximum profit for the cloud broker and minimum response time and minimum energy consumption of the cloud. There are three types of instances: small, large and extra-large. The average execution time and instance prices depend on the instance type. Table I lists the set of parameters used in the cloud brokering system.

TABLE I
Cloud Brokering Parameters

Parameter	Value
Instance type	small, large, extra-large
Average execution time(s)	980 ± 71, 616 ± 61, 697 ± 13
Hourly price	0.1, 0.125, 0.143
Instance prices	Amazon EC2 pricing history

In our simulations, we used jMetal to execute the MOPSO, GA and random search algorithm. jMetal is a simulator for multi-objective optimization problems with meta-heuristics. It utilizes an object-oriented JAVA-based framework. The simulation results are compared to those of a well-known existing GA for multi-objective optimization and the random search algorithm. In this paper, we use a non-dominated sorting genetic algorithm-II (NSGA-II) [28] as the GA for multi-objective optimization because of the low computational requirements, parameter-less sharing and elitist approach of the NSGA-II ($O(mN^2)$, where N is the population size and m is the number of objectives). The simulated binary crossover (SBX) [29] is applied in NSGA-II. The average results of 10 runs of each scenario are compared. The simulation configurations of MOPSO are set as follows: 100 swarms, 100 archives, 250 iterations, and a mutation rate of $1/n$. The simulation configurations of the NSGA-II are set as follows: 100 populations, 250 iterations, a mutation rate of $1/n$, and a crossover rate of 0.9. The summary of the simulation configurations is presented in Table II.

TABLE II
THE SIMULATION CONFIGURATIONS

Configuration	MOPSO	NSGA-II
Number of iterations	250	250
Population Size	-	100
Swarm Size	100	-
Archive Size	100	-
mutation rate	$1/n$	$1/n$
crossover rate	-	0.9
degree of SBX crossover	-	15
degree of polynomial mutation	20	20

B. Simulation Results

In this section, we divide the simulation results into three parts: C-metric, the results of the single-objective optimization problems that are solved using single-objective particle swarm optimization, and the results of the multi-objective optimization problem that are solved using the MOPSO, GA and random search algorithm. In the first part, the C-metric shows the performance of the solutions of each algorithm. Next, we show a comparison of the weight value of the single-objective optimization problem in the second part. Finally, we show a comparison of the best solutions in each iteration of the three algorithms: MOPSO, GA and random search.

1) C-metric: In these simulations, we use C-metric [30] as a performance metric, which represents how individuals obtained from one algorithm outperform the individuals from another. $C(A, B)$ represents the C-metric between algorithm A and B. $C(A, B)$ is calculated as follows:

$$C(A, B) = |\{b \in B \mid \exists a \in A : a \succ b\}|/|B|, \tag{18}$$

where operator $\succ$ denotes the dominating algorithm (e.g., $a \succ b$ represents individual a dominating individual b). Thus, if $C(A, B) = 0$, there is no individual in B that is dominated by an individual in A. On the other hand, if $C(A, B) = 1$, there is at least one individual in A that dominates all individuals in B.

TABLE III
C-METRIC

$C(A, B)$	min	avg	max
$C(MOPSO, NSGA-II)$	0.04	0.82	1
$C(NSGA-II, MOPSO)$	0.16	0.73	0.83
$C(MOPSO, RandomSearch)$	0.75	0.87	1
$C(RandomSearch, MOPSO)$	0.26	0.50	0.75
$C(NSGA-II, RandomSearch)$	0.75	0.78	0.82
$C(RandomSearch, NSGA-II)$	0.34	0.54	0.75

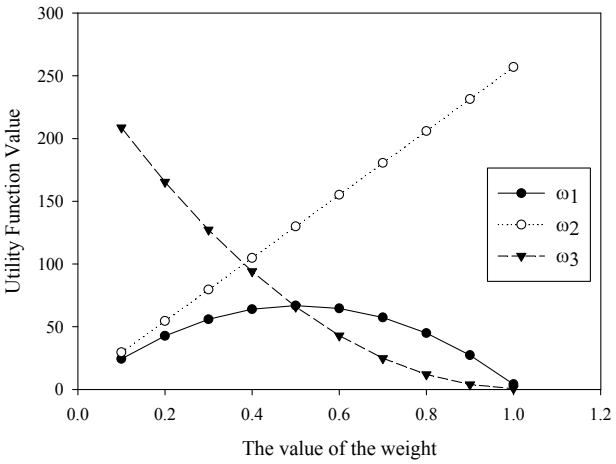


Fig. 5. The utility function value of each weight value

Table III shows the C-metric at iteration 250. The result shows that in terms of the minimum, maximum and average of 10 independent runs, the MOPSO contributes more to the non-dominated frontier than the NSGA-II and random search algorithm. The maximum of $C(MOPSO, NSGA-II)$ is 1, which implies that there is at least one individual in $MOPSO$ that dominates all individuals in $NSGA-II$. In addition, the maximum of $C(MOPSO, RandomSearch)$ is 1, which implies there is at least one individual in $MOPSO$ that dominates all individuals in $RandomSearch$.

2) Single-Objective Optimization Problem: In this paper, we described the three objectives in the cloud brokering system as a single-objective optimization problem and single-objective PSO to solve the problems in Subsection III-A and IV-A, respectively, because it is difficult to find the best weight value. Thus, we compared the results from the single-objective PSO with three different weight values (ω_1, ω_2 and ω_3). In our simulations, the weight values are formulated as follows:

$$\omega_1 = random(\lambda)/\lambda \tag{19}$$

$$\omega_2 = (1 - \omega_1)(\omega_1) \tag{20}$$

$$\omega_3 = 1 - \omega_1 - \omega_2 \tag{21}$$

where λ is a random number that is uniformly distributed as a positive number.

Figure 5 shows the best utility function value of each weight value. Figures 5 shows the comparison of the best utility function value that has weight values ranging from 0.1 to 1.0.

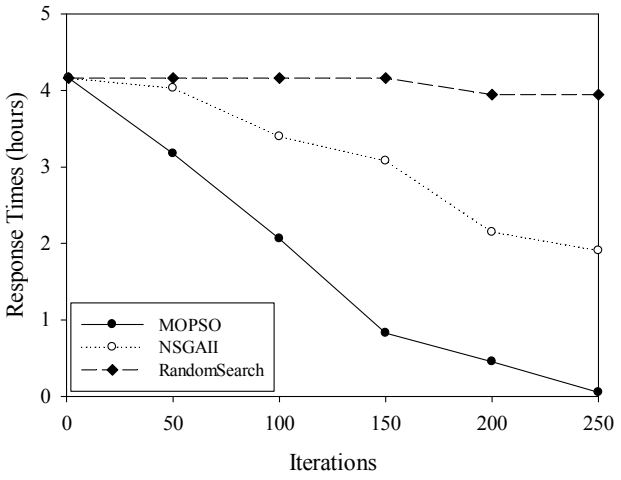


Fig. 6. The response time at the end of each iteration

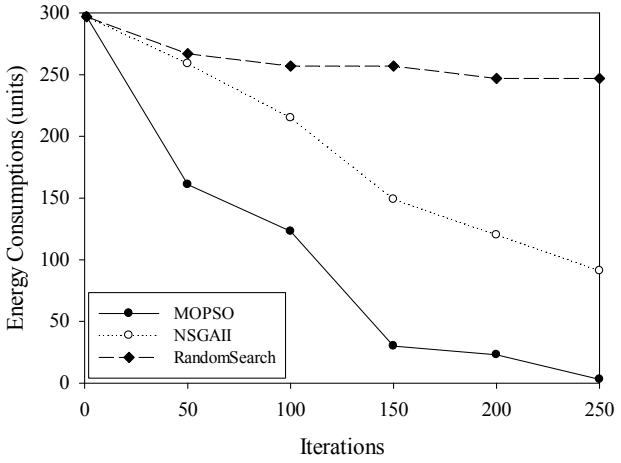


Fig. 7. The energy consumption at the end of each iteration

The results show that the utility function value in the single-objective optimization depends on the weight value. We can choose the weight value according to our objective; however, we cannot conclude that it is the best weight value for the problem.

3) Multi-objective Optimization Problem: We describe the three objectives in the cloud brokering system as a multi-objective optimization problem and multi-objective PSO to solve the problems in Subsection III-B and IV-B, respectively. In these simulation results, we present a comparison of the best results of three algorithms: MOPSO, GA and random search algorithm. The simulation setups of each algorithm are described in Subsection V-A.

Figures 6, 7 and 8 show the best results of each scenario based on 10 independent runs. Figures 6, 7 and 8 show the response time, the energy consumption and the profit of the cloud broker at the end of each iteration, respectively. The results show that the MOPSO contributes to the optimality of

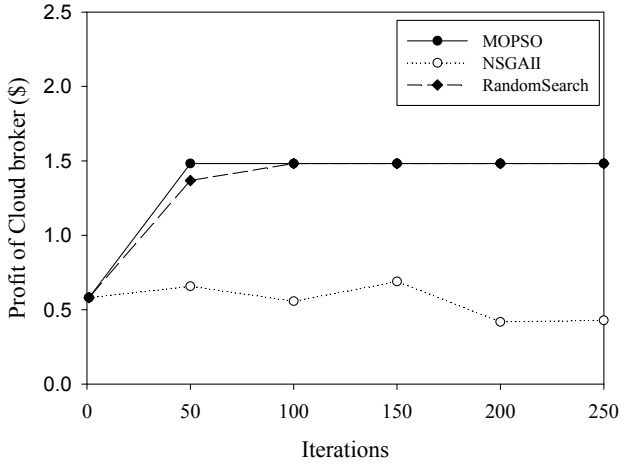


Fig. 8. The profit of the cloud broker at the end of each iteration

the response time, the energy consumption and the profit of the cloud broker more than the NSGA-II and random research algorithm. However, the profit of the cloud broker from the random search algorithm is similar to the resultsts from the PSO. It is possible that the random processes in the random search algorithm lead to high values for the three objectives.

Moreover, PSO and GA are similar in terms of population-based search approaches and depend on information sharing among members (particle or individual) in the population to enhance their processes. GA has a high ability for global searching. However, there are many parameters (e.g., population size, crossover rate and mutation rate) and many operators (e.g., crossover, mutation and selection operator). Thus, the complexity of the GA is higher, and the convergence is slower. On the other hand, PSO has a high global searching ability as well, but it could be implemented simply without too many parameters and operators. Thus, it converges fast. However, it is possible that the solutions of PSO may fall into local optima.

VI. Conclusion

In this paper, we proposed a multi-objective particle swarm optimization (MOPSO) scheme for cloud brokering to find the appropriate connections between clients and service providers to optimize the energy consumption of service providers, the profit of the cloud broker and the response time of requests from clients. Extensive simulations have been conducted, and the results demonstrate that the MOPSO is able to find appropriate sets of solutions for cloud brokering. We compared the performance of the MOPSO with that of a well-known genetic algorithm (NSGA-II) and a random search algorithm. The results show that the proposed algorithm successfully reduces the response time and the energy consumption of the system and increases the profit of the cloud broker better than the NSGA-II and the random search algorithm. In future work, we will conduct experiments under large-scale and well-known cloud simulations (e.g., CloudSim, GreenCloud, and

iCanCloud). Additionally, we will take into account the complexity of the MOPSO as a parameter of energy consumption and find the operators to reduce the complexity of MOPSO.

Acknowledgment

This work is partially supported by JSPS KAKENHI Grant Number 26730056, 16K00117, 15K15976, JSPS A3 Foresight Program.

Appendix
List of Parameters

Notation	Meaning
N	The number of clients
M	The number of cloud service providers
U	The set of N clients
S	The set of M cloud service providers
b_{ij}	A binary variable with client i and service provider j
RT	The response time
P	The profit of the cloud broker
E	The energy consumption
L_{ij}	The latency between client i and service provider j
T_j	The execution time of the request at service provider j
P_i	The price from client i
C_j	The cost of service provider j
E_j	The energy consumption of service provider j used to execute the job
R_i	The number of requests from the client i
A_j	The capacity of service provider j
v_{lk}	The velocity vector at the l-th particle and the k-th dimension
x_{lk}	The position vector at the l-th particle and the k-th dimension
p_{lk}	The best position of the l-th particle and k-th dimension
g_{lk}	The best position of the swarm at the l-th particle and the k-th dimension
w	An inertia weight
c_1	the coefficient of the self-recognition component
c_2	the coefficient of the social component
r_1, r_2	A random number that is uniformly distributed in the interval 0 to 1.
ω_1	The weight of the response time of requests
ω_2	The weight of the profit of the cloud broker
ω_3	The weight of the total energy consumption of the system
μ	The number of individuals in the population
d'	The number of individuals that dominate X_i
λ	A random number that is uniformly distributed as a positive number.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] R. Buyya, J. Broberg, and A. M. Goscinski, *Cloud computing: principles and paradigms*. John Wiley & Sons, 2010.

[3] J. Liu, N. Kato, J. Ma, and N. Kadowaki, "Device-to-device communication in LTE-advanced networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 1923–1940, 2014.

[4] Y. Kessaci, N. Melab, and E.-G. Talbi, "A pareto-based metaheuristic for scheduling hpc applications on a geographically distributed cloud federation," *Cluster Computing*, vol. 16, no. 3, pp. 451–468, 2013.

[5] T. S. Somasundaram and K. Govindarajan, "Cloudrb: A framework for scheduling and managing high-performance computing (hpc) applications in science cloud," *Future Generation Computer Systems*, vol. 34, pp. 47–65, 2014.

[6] S. Iturriaga, S. Nesmachnow, B. Dorronsoro, E.-G. Talbi, and P. Bouvry, "A parallel hybrid evolutionary algorithm for the optimization of broker virtual machines subletting in cloud systems," *IEEE 2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, pp. 594–599, 2013.

[7] K. Yildirim, T. Kalayci, and A. Ugur, "Optimizing coverage in a k-covered and connected sensor network using genetic algorithms," in *Proceedings of the 9th WSEAS International Conference on Evolutionary Computing*. World Scientific and Engineering Academy and Society (WSEAS), 2008.

[8] M. Le Berre, F. Hnaien, and H. Snoussi, "Multi-objective optimization in wireless sensors networks," in *IEEE International Conference on Microelectronics (ICM)*, 2011.

[9] P. Champrasert, J. Suzuki, and T. Otani, "Evolutionary high-dimensional QoS optimization for safety-critical utility communication networks," *Natural Computing*, vol. 10, no. 4, pp. 1431–1458, 2011.

[10] Z. Sun, Y. Zhang, Y. Nie, W. Wei, J. Lloret, and H. Song, "Casmoc: a novel complex alliance strategy with multi-objective optimization of coverage in wireless sensor networks," *Wireless Networks*, pp. 1–22, 2016.

[11] J. Liu, X. Jiang, H. Nishiyama, R. Miura, N. Kato, and N. Kadowaki, "Optimal forwarding games in mobile ad hoc networks with two-hop f-cast relay," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 11, pp. 2169–2179, 2012.

[12] W. Wei, X. Fan, H. Song, X. Fan, and J. Yang, "Imperfect information dynamic stackelberg game based resource allocation using hidden markov for cloud computing," *IEEE Transactions on Services Computing*, 2016.

[13] H. Li, M. Dong, K. Ota, and M. Guo, "Pricing and repurchasing for big data processing in multi-clouds," in *IEEE Transactions on Emerging Topics in Computing*. 2016, 10.1109/TETC.2016.2517930.

[14] I. Butun, M. Erol-Kantarci, B. Kantarci, and H. Song, "Cloud-centric multi-level authentication as a service for secure public safety device networks," *IEEE Communications Magazine*, vol. 54, no. 4, 2016.

[15] M. Dong, H. Li, K. Ota, L. T. Yang, and H. Zhu, "Multicloud-based evacuation services for emergency management," *IEEE Cloud Computing*, vol. 1, no. 4, pp. 50–59, 2014.

[16] H. Li, M. Dong, X. Liao, and H. Jin, "Deduplication-based energy efficient storage system in cloud environment," *The Computer Journal*, vol. 58, no. 6, pp. 1373–1383, 2015.

[17] J. Liu, H. Guo, H. Nishiyama, H. Ujikawa, K. Suzuki, and N. Kato, "New perspectives on future smart fiwi networks: Scalability, reliability and energy efficiency," *IEEE Communications Surveys & Tutorials*, 2015.

[18] I. Bisio, R. L. C. Pan, F. Lavagetto, M. Marchese, A. Sciarrone, C. Fra, and M. Valla, "Smartphone-based automatic place recognition with wi-fi signals for location-aware services," in *2012 IEEE International Conference on Communications (ICC)*, 2012.

[19] J. Xu and J. A. Fortes, "Multi-objective virtual machine placement in virtualized data center environments," in *2010 IEEE/ACM Int'l Conference on Green Computing and Communications (GreenCom) & Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, 2010.

[20] R. Jeyarani, N. Nagaveni, and R. V. Ram, "Design and implementation of adaptive power-aware virtual machine provisioner (APA-VMP) using swarm intelligence," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 811–821, 2012.

[21] S. Pandey, L. Wu, S. M. Guru, and R. Buyya, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments," in *2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, 2010.

[22] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *Journal of Computer and System Sciences*, vol. 79, no. 8, pp. 1230–1242, 2013.

[23] C. Mateos, E. Pacini, and C. G. Garino, "An ACO-inspired algorithm for minimizing weighted flowtime in cloud-based parameter sweep experiments," *Advances in Engineering Software*, vol. 56, pp. 38–50, 2013.

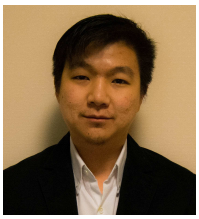
[24] E. Feller, L. Rilling, and C. Morin, "Energy-aware ant colony based workload placement in clouds," in *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*. IEEE Computer Society, 2011.

[25] M. R. Garey and D. S. Johnson, "Computers and intractability: a guide to the theory of np-completeness. 1979," *San Francisco, LA: Freeman*, 1979.

[26] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

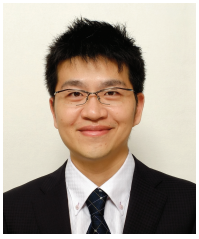
[27] M. Reyes-Sierra and C. C. Coello, "Multi-objective particle swarm optimizers: A survey of the state-of-the-art," *International journal of computational intelligence research*, vol. 2, no. 3, pp. 287–308, 2006.

[28] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
[29] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Son, 2001.
[30] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.



Teerawat Kumrai received a B.S. in Mathematics and M.Eng. in Computer Engineering from Chiang Mai University, Thailand. He is currently a doctoral student in the Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. His research interests include wireless sensor networks, vehicular ad-hoc networks, participatory sensing, smart grid, optimization techniques and cloud computing.

Kaoru Ota was born in Aizu Wakamatsu, Japan. She received M.S. degree in Computer Science from Oklahoma State University, USA in 2008, B.S. and Ph.D. degrees in Computer Science and Engineering from The University of Aizu, Japan in 2006, 2012, respectively. She is currently an Assistant Professor at the Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. From March 2010 to March 2011, she was a visiting scholar at the University of Waterloo, Canada. She was also a Japan Society of the Promotion of Science (JSPS) research fellow with the Kato-Nishiyama Lab at the Graduate School of Information Sciences at Tohoku University, Japan, from April 2012 to April 2013. She joined the JSPS A3 foresight program as a primary researcher in 2011, which is supported by the Japanese, Chinese and Korean governments. She was a Guest Editor of IEEE Wireless Communications, IEICE Transactions on Information and Systems and serves as Editor of Peer-to-Peer Networking and Applications (Springer), Ad Hoc & Sensor Wireless Networks, International Journal of Embedded Systems (Inderscience) and Journal of Cyber-Physical Systems. Her research interests include wireless sensor networks, vehicular ad hoc networks, and ubiquitous computing.



Mianxiong Dong received his B.S., M.S. and Ph.D. in Computer Science and Engineering from The University of Aizu, Japan. He is currently an Associate Professor at the Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. Before joining Muroran-IT, he was a Researcher at the National Institute of Information and Communications Technology (NICT), Japan. He was a JSPS Research Fellow with the School of Computer Science and Engineering, the University of Aizu, Japan, and was a Visiting Scholar with the BBCR group at the University of Waterloo, Canada supported by the JSPS Excellent Young Researcher Overseas Visit Program from April 2010 to August 2011. Dr. Dong was selected as a Foreigner Research Fellow (a total of 3 recipients all over Japan) by NEC C&C Foundation in 2011. Dr. Dong's research interests include wireless sensor networks, vehicular ad hoc networks, software-defined networks, big data and cloud computing. He is the Best Paper Award Winner of IEEE HPCC 2008, IEEE ICESS 2008, ICA3PP 2014, GPC 2015 and IEEE DASC 2015. Dr. Dong serves Editor of IEEE Communications Surveys and Tutorials, IEEE Network, IEEE Wireless Communications Letters, IEEE Access, Cyber-Physical Systems (Taylor & Francis), leading guest editor of ACM Transactions on Multimedia Computing, Communications and Applications (TOMM), IEEE Transactions on Computational Social Systems (TCSS), Peer-to-Peer Networking and Applications (Springer), and the Symposium Chair of IEEE GLOBECOM 2016. Dr. Dong is currently a research scientist with the A3 Foresight Program (2011-2016) funded by Japan Society for the Promotion of Sciences (JSPS), NSFC of China, and NRF of Korea.

Jay Kishigami received his B.S. and M.S. in Physics from The Hokkaido University, Japan in 1980 and a Ph.D. degree in electronic engineering from The Hokkaido University, Japan, in 1989. He is currently a Professor at Muroran Institute of Technology, Hokkaido, Japan. He was a Professor at the University of Tokyo for five years. He is also serving as Adjunct Professor at UTAR. He joined UTAR in 2012 as a professor at the faculty of engineering and science. He has published more than 100 papers and presentations and 80 patents. He has been a member or leader of more than 20 government committees. He is a senior advisor to NTT, sits on the advisory board of W3D and is a Board member of MMHL. He worked at NTT Lab and NTT America for more than 30 years. He is a member of IEICE Japan and IPSJ Japan and is a Distinguished Speaker of IEEE, committee at ARIB, INSTAC and TV Anytime Forum. He won the Minister of Internal Affairs and Communication Award in 2011. His research interests include fundamental informatics, statistical science, computer system network, media informatics/data base, and fundamental engineering.

Dan Kuen Sung received the B.S. degree in electronics engineering from Seoul National University in 1975 and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin, in 1982 and 1986, respectively. Since1986, he has been with the faculty of the Korea Advanced Institute of Science and Technology (KAIST), Daejon, Korea, where he is currently a Professor with the School of Electrical Engineering. From 1996 to 1999, he was the Director of the Satellite Technology Research Center (SaTReC), KAIST. He had served as Division Editor of the Journal of Communications and Networks from 1998-2007. He also had served as Editor of IEEE Communications Magazine from 2002-2011. His research interests include mobile communication systems and networks, with special interest in resource management, smart grid communication networks, energy networks, machine-to-machine communications, WLANs, WPANs, traffic control in wireless & wired networks, performance and reliability of communication systems, and microsatellites. Dr. Sung is a member of the National Academy of Engineering of Korea. He was the recipient of the 1992 National Order of Merits, the Dongbaek Medal for successfully developing, launching, and operating the first Korean satellite in Korean history, the 1997 Research Achievement Award, the 2000 Academic Excellence Award, the 2004 Scientist of the Month from the Ministry of Science and Technology and the Korea Science and Engineering Foundation, and the 2013 Haedong Academic Grand Award from the Korean Institute of Communications and Information Sciences.