Research on Task Scheduling for Internet of Things Cloud Computing Based on Improved Chicken Swarm Optimization Algorithm

Shizheng Liu[1], Xuan Chen[1] and Feng Cheng[2,*]

[1]*Zhejiang Industry Polytechnic College, Shaoxing, Zhejiang, 312000, China*
[2]*Southwest Jiaotong University, Chengdu, Sichuan, 611756, China*
E-mail: chengfeng2013 @swjtu.edu.cn
[*]*Corresponding Author*

Received 15 September 2023; Accepted 02 November 2023

Abstract

Aiming at the shortcomings of long completion time and high consumption cost of cloud computing batch task scheduling in IoT, an Improved Chicken Swarm Optimization Algorithm (ICSO) for task scheduling in cloud computing scenarios is proposed. Specifically, in order to solve the problems of slow convergence and falling into local optimum of the chicken swarm optimization algorithm, we adopt the nonlinear decreasing technique of the rooster and the weighting technique of the hen, optimize the following coefficients of the chicks, and apply ICSO to cloud computing task scheduling. In simulation experiments, we conducted a large number of experiments using four standard benchmark functions with different number of tasks and the results show that ICSO algorithm reduces 25.8%, 9.3%, 8.8%, 7.5% in small task time compared to CSO, DCSO, GCSO, ABCSO in large task

Journal of ICT Standardization, Vol. 12_1, 21–46.
doi: 10.13052/jicts2245-800X.1212

time by 30.8%, 8.3%, 7.8%, 6.3%, 11.8%, 10.3%, 8.8%, 7.5% savings in small task cost and 25.8%, 11.2%, 10.8%, 9.3% savings in large task cost. This method effectively reduces task scheduling time and cost consumption. Meanwhile, we tested it in combination with an IoT-based cloud platform and achieved very satisfying Results.

Keywords: Cloud computing, task scheduling, chicken swarm optimization, Internet of Things.

1 Introduction

In recent years, major technological innovations have occurred in sensing, computing, and communication technologies. In particular, the rapid development of wireless communication technology has made it possible to access devices, software, or control systems that are deployed anywhere, anytime. And the Internet of Things is a combination of these important technologies. However, due to the capacity constraints of IoT devices themselves, such as limited memory, CPU, battery capacity, etc., this will make them unsuitable for tasks that require large computational resources and are energy consuming on the device side. Cloud computing [1] provides resources such as computing, storage or operating environment, and the correspond-ing tasks are run in the cloud and the results are sent to the IoT devices, which not only reduces the energy consumption of the IoT devices, but also expands the functions of the IoT devices by utilizing the powerful computing and storage capabilities of the cloud computing, which is no longer limited to its own hardware configuration [2].

Task scheduling is an important aspect of cloud computing, which involves the rational allocation of resources and efficient execution of tasks. More and more organizations and individuals are migrating their computing tasks to the cloud, making the research and practice of cloud computing task scheduling especially critical. The goal of cloud computing task scheduling [3] is to reasonably allocate tasks to the resources of cloud service providers and optimize the resource utilization and task completion time, while the task scheduling algorithms need to consider several factors, such as the priority of the tasks, the availability of the resources, the dependencies of the tasks, and so on. Therefore, how to efficiently perform cloud computing task scheduling is a key bottleneck that needs to be solved by implementing an adaptive scheduling strategy that balances the load of the entire platform and minimizes resource consumption while ensuring service quality.

It is well known that cloud computing task scheduling task is a typical NP problem [4]. In order to solve this problem, there are two methods studied by scholars, one is used to study the real-time processing tasks, and the other is used to study the batch processing tasks. The more representative algorithm of the former is the deep reinforcement learning algorithm [5], and the latter is the metaheuristic algorithm [6]. The chicken swarm optimization (CSO) algorithm [7] is one such novel intelligent metaheuristic algorithm, which is inspired by the behavioral characteristics and foraging behavior of chickens. This algorithm obtains the optimal solution by simulating the collaborative and competitive behavior of multiple chickens while foraging. It also has the advantages of fewer control parameters and greater stability than other algorithms.

We use the chicken swarm optimization algorithm for cloud computing task scheduling under IoT in this paper. However, in order to solve the problem that the chicken swarm optimization algorithm has the problem of fast convergence and easy to fall into the local optimum, we propose an improved chicken swarm optimization algorithm (ICSO), which is improved in three aspects: the non-linear decreasing based idea is used in the update of the rooster's position, the weighting idea is used in the update of the hen's position, and the adaptive idea is used in the following coefficients of the chick's position. ICSO is used for cloud computing task scheduling in IoT, and simulation experiments verify the performance of this paper's algorithm in cloud computing task scheduling.

Therefore, the main contributions of the work in this paper are as follows: (1) We propose a cloud computing task scheduling model based on optimization time and cost; (2) We propose an improved Chicken swarm Optimization Algorithm (ICSO), which improves the efficiency of task scheduling by employing advanced optimiza-tion techniques, and by improving the convergence speed and accuracy as compared to the traditional CSO-based approach. (3) We evaluated the ICSO algorithm. The experimental results show that the algorithm can achieve superior performance in terms of completion time and cost consumption.

This paper is organized as follows: we present the current state of the art in cloud computing task scheduling research in Section 2, and in Section 3, we introduce the time and cost based task scheduling model. In Section 4, we describe the process of improving the chicken swarm Optimization Algorithm in detail. In Section 5, we test the performance of the algorithm, and analyze the algorithm for task scheduling under IoT cloud computing. In Section 6, we summarize the full paper.

2 Related Research

Currently, the main research on cloud computing scheduling problem is categorized into task scheduling and VM-host mapping [8]. In task scheduling, the bandwidth, storage, cost and time are different for each task. Therefore, the goal of scheduling is the need to obtain the appropriate VM resources, which has a direct relationship to the impact of both time and cost efficiency of task scheduling [9].

The current research on cloud computing task scheduling mainly focuses on two aspects: schedule of real-time tasks and schedule of batch tasks. The former case is mainly addressed by deep reinforcement learning, in which the intelligent agent can interact with the cloud environment and learn an optimal scheduling policy based on the optimization objectives [10]. Although all the approaches are shown to be efficient and can output other real-time scheduling algorithms such as round-robin, they are not suitable for handling tasks in a batch way.

In comparison the DL based algorithms, the most commonly used algorithms for scheduling batch tasks is metaheuristics [11]. For instance, Bezdan et al. [12] pro-posed an improved bat algorithm for cloud computing task scheduling and presented a new cloud computing task scheduling model. The simulation results showed that the algorithm has better performance in terms of virtual machine load and task completion time, but further optimization of cost is needed. Chen et al. [13] proposed the use of a whale algorithm for cloud computing task scheduling and constructed a task scheduling model based on virtual machines, cost, and time. The simulation results showed that the algorithm has good results in terms of virtual machines, completion time, and task scheduling. Mangalampalli et al. [14] used a cat swarm algorithm for cloud computing task scheduling, and the simulation results showed that the algorithm has good performance in terms of scheduling efficiency. Velliangiri et al. [15] proposed a genetic algorithm based on electronic search strategy, and the simulation results showed that this algorithm outperforms traditional metaheuristic algorithms such as GA, PSO, etc. in terms of completion time and cost consumption in cloud computing. Abualigah et al [16] proposed a hybrid ant lion optimization algorithm based on elite differential evolution for cloud computing task scheduling, and the simulation results showed better results in resource utilization maximization and time minimization. Muthulakshmi et al. [17] proposed a cloud computing task scheduling strategy using an artificial swarm algorithm and optimized the task scheduling effect by using an improved swarm algorithm.

The other approach to improve scheduling is the combination of multiple metaheuristic algorithms, as seen in studies such as Manikandan et al.'s hybrid method using artificial bee colony and whale optimization algorithm [18], Chen et al. proposed ACO and PSO algorithms for cloud computing task scheduling [19]. Additionally, there are also many customized heuristics which can scheduling cloud tasks in an efficient way, such as on workflows [20], and deep learning applications [21]. Although using a hybrid algorithm with two or more metaheuristics or customized heuristics can also achieve good performance in cloud computing task scheduling, but it may come at the cost of increased algorithm complexity. Moreover, some customized algorithms have also been proposed to address scheduling problems in cloud [22, 23], but their performance could be not robust as metaheuristics. In comparison, in this work, we focus on utilizing a single metaheuristic algorithm, the CSO algorithm, which can effectively enhance task scheduling in cloud computing. More specifically, we are aiming to further improve its performance through effective algorithm optimization.

3 Task Scheduling Model

Performing cloud computing task scheduling in IoT is mainly to optimize the scheduling of tasks distributed on various IoT devices to maximize system efficiency, reduce energy consumption, and meet real-time requirements. In IoT, various tasks that require cloud computing task processing are collected, and these tasks are categorized according to priority, real-time requirements, etc., and cloud service resources are reasonably configured according to the requirements of these tasks. Therefore, the essence of designing a well-performing cloud computing task scheduling model is how to maximize the benefit of the relationship between the VMs required for a task, the cost of the task, and the time consumption. In order to better represent our proposed algorithm and achieve better results in task scheduling, we choose the task scheduling model proposed in the literature [24] as a model reference, and we use time and cost as the main factors to study the optimization objective of task scheduling under IoT.

We assume that the set of tasks is $Task = \{task_1, task_2, \ldots task_m\}$, the total number of tasks is m, $task_i$ represents the ith task, the set of virtual machines is $Vm = \{vm_1, vm_2, \ldots vm_n\}$, the number of virtual machines is n. Therefore, the correspondence between tasks and virtual machines is represented by matrix R, as shown in Equation (1), r_{ij} is 1 means task i is assigned to virtual machine j, r_{ij} is 0 means task i is not assigned to the

corresponding virtual machine

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ r_{m1} & r_{m2} & \cdots & r_{mn} \end{bmatrix} \tag{1}$$

(1) Task completion time and cost

Task completion time is the whole process of the task from the time it is submitted until the task is executed on the virtual machine, i.e., the time the task is transferred to the virtual machine, the time the task waits for the virtual machine and the time the task is executed on that virtual machine. Equation (2) represents the transfer time, Equation (3) represents the task wait time, and Equation (4) represents the task execution time. Thus, the total task completion time is the maximum completion time of the executed tasks in each virtual machine, and the maximum completion time of each virtual machine comes from the maximum time that each task finishes completing individually.

$$TT = \begin{bmatrix} tt_{11} & tt_{12} & \cdots & tt_{1n} \\ tt_{21} & tt_{22} & \cdots & tt_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ tt_{m1} & tt_{m2} & \cdots & tt_{mn} \end{bmatrix} \tag{2}$$

$$WT = \begin{bmatrix} wt_{11} & wt_{12} & \cdots & wt_{1n} \\ wt_{21} & wt_{22} & \cdots & wt_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ wt_{m1} & wt_{m2} & \cdots & wt_{mn} \end{bmatrix} \tag{3}$$

$$ET = \begin{bmatrix} et_{11} & et_{12} & \cdots & et_{1n} \\ et_{21} & et_{22} & \cdots & et_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ et_{m1} & et_{m2} & \cdots & et_{mn} \end{bmatrix} \tag{4}$$

$$TaskFinshTime = \max\{\max\{tt_{ij} + et_{ij} + wt_{ij}\}\} \tag{5}$$

In Equation (2), tt_{ij} denotes the time it takes for task i to get to virtual machine j, i.e., $tt_{ij} = task_{i\_input\_size}/vm_{j\_bw}$, where $task_{i\_input\_size}$ denotes the amount of data for task i and vm_{j_bw} denotes the virtual machine communication bandwidth. In Equation (3), wt_{ij} denotes the time that task i waits for virtual machine j, i.e., $wt_{ij} = \sum_{u \in TaskQuene_j} et_{uj}$, where $u \in$

$TaskQuene_j$ denotes the queue of unexecuted tasks on virtual machine j, u is the task number, and et_{uj} denotes the execution time of task u on virtual machine j. In Equation (4) et_{ij} denotes the execution time of task i on virtual machine $j, et_{ij} = task_{i\_length}/vm_{j\_compute}$, $task_{i\_length}$ denotes the size of the instruction, and $vm_{j\_compute}$ denotes the computing power of the virtual machine whose value is composed of the computing power of multiple virtual machines.

The cost in task scheduling under cloud computing mainly considers the cost required for task execution in the virtual machine shown in Equation (6) and the cost required during task transfer shown in Equation (7). The total cost of task completion is the sum of the above two costs, as shown in Equation (8)

$$CostExecute_{ij} = et_{ij} \times vm_{j\_execute\_cost} \tag{6}$$

$$CostBW_{ij} = tt_{ij} \times vm_{j\_bw} \times vm_{j\_bw\_unit\_cost} \tag{7}$$

$$TotalCost = \sum_{j=1}^{n} \sum_{i=1}^{m} (CostExecute_{ij} + CostBW_{ij}) \tag{8}$$

In Equation (6), $vm_{j\_execute\_cost}$ denotes the cost per unit time of the virtual machine i.e. $vm_{j\_execute\_cost} = vm_{j\_mips} \times vm_{j\_core\_num} \times vm_{j\_mips\_unit\_cost}$, vm_{j_mips} denotes the individual CPU computing power, $vm_{j\_core\_num}$ denotes the number of CPUs of the virtual machine, and $vm_{j\_mips\_unit\_cost}$ denotes the $mips$ cost of the virtual machine. In Equation (7), vm_{j_bw} denotes the communication bandwidth of a VM and $vm_{j\_bw\_unit\_cost}$ denotes the unit bandwidth of a VM.

(2) Time and cost functions

The time and cost functions in this paper are the time affiliation function and the cost affiliation function, respectively. The time affiliation function is shown in Equation (9), and the cost affiliation function is shown in Equation (10).

$$Time(I) = \begin{cases} (TaskFinisTime - Time_{deadline})/ & TotalCost \\ TotalCost & > Time_{deadline} \\ (Time_{deadline} - TaskFinisTime)/ & TotalCost \\ Time_{deadline} & \leq Time_{deadline} \end{cases} \tag{9}$$

$$Cost(I) = \begin{cases} (TotalCost - Cost_{expert})/ & TotalCost \\ TotalCost & > Cost_{expect} \\ (Cost_{expert} - TotalCost)/ & TotalCost \\ Cost_{expert} & \leq Cost_{expect} \end{cases} \tag{10}$$

In Equations (9)–(10), $Time_{deadline}$ denotes the as of time for all tasks and $Cost_{expect}$ denotes the expected cost for all tasks.

4 Intelligent Task Scheduling Based on ICSO Algorithm

4.1 Chicken Swarm Optimization Algorithm

The core idea of the swarm optimization optimization algorithm is to simulate the process of finding food by the swarm optimization behavior. It divides the whole swarm optimization into several sub-groups according to different hierarchies and foraging in the swarm optimization, and each sub-group mainly consists of one rooster, several hens and chicks. Since there are different hierarchies, there is some competition between different swarm optimizations. To better describe the algorithm, we describe the individuals and behaviors in the algorithm. The individual with the best fitness value is called a rooster, the individual with the worst fitness value is a chick, and the individual with moderate fitness value is a hen in the swarm optimization algorithm. The hen can randomly select any subgroup. The relationship between the hen and the chicks is not fixed and remains unchanged once established and changes only after renewal. In each subgroup, all individuals within the group are searching for food around the rooster.

Set the population size of the swarm optimization as N, the spatial dimension as D, and the location of individual i in the tth iteration in the jth dimension as $x_i^j(t)$. The positions of the roosters, hens and chicks are indicated as follows.

(1) Position of rooster

The rooster is the best adaptation value in each subgroup and is in the leading position in the subgroup, with the following position updates.

$$x_i^j(t+1) = x_i^j(t) \times (1 + Randn(0, \sigma^2)) \tag{11}$$

$$\sigma^2 = \begin{cases} 1 & if\ f_i \leq f_k \\ \exp\left(\dfrac{f_k - f_i}{|f_i| + \varepsilon}\right) & otherwise \end{cases} \tag{12}$$

In Equations (11)–(12), $Randn(0, \sigma^2)$ denotes a normal distribution with mean 0 and standard deviation σ, ε is a constant that ensures that the denominator is not zero, and f_i and f_k denote the objective function values of the current i, k roosters, respectively.

(2) Position of hens

The hen is the most numerous individual in the subgroup. During foraging, the hen mainly searches for food near the rooster, and the position of the hen is updated with the following formula.

$$x_i^j(t+1) = x_i^j(t) + c_1 \times rand \times (x_{r1}^j(t) - x_i^j(t)) + c_2 \times rand \times (x_{r2}^j(t) - x_{i,j}(t)) \tag{13}$$

$$c_1 = \exp((f_i - f_{r1})/abs(f_i) + \varepsilon) \tag{14}$$

$$c_2 = \exp(f_{r2} - f_i) \tag{15}$$

In Equations (13)–(15), $rand$ is a random number with a value of 0.5, r_1 is the rooster corresponding to the ith hen itself, r_2 is any randomly selected individual among the roosters and hens in the whole swarm optimization, and $r_1 \neq r_2$.

(3) Position of chicks

In each subgroup, the chicks will forage with the hen's side, and the position is updated as follows

$$x_i^j(t+1) = x_i^j(t) + F \times (x_m^j(t) - x_i^j(t)) \tag{16}$$

In Equation (16), m means the i chicks corresponding to the hen, F is the following factor, that the chicks follow the hen in search of food.

4.2 Improved Chicken Swarm Optimization Algorithm

Like most of the meta-heuristic algorithms characteristics, we optimize the swarm-ing optimization algorithm for the drawbacks such as its tendency to fall into local optimums, which leads to slow convergence, and so on, we optimize it in the follow-ing three directions:

(1) Rooster position update based on nonlinear decreasing

In the swarm optimization algorithm, the position of the individual rooster is very important, which determines the effect of the optimal value of the algorithm's solution, but it is found from the update of the individual position that

the position of the rooster lacks timely comparison with the global individual solution, which makes the local and global solutions of the algorithm cannot be balanced in a timely and effective manner, and in order to improve this situation, we introduce a nonlinear learning factor formula in the update of the rooster's position.

$$a(t) = t \times \frac{\ln w_{\max} - \ln w_{\min}}{t_{\max}} \tag{17}$$

$$w(t) = e^{-a(t)} \tag{18}$$

In Equations (17)–(18), t denotes the current number of iterations, $t_{\max}$ denotes the maximum number of iterations, $w_{\max}$ and $w_{\min}$ are the maximum and minimum values of the learning factors, respectively. $w(t)$ is the learning factor, so the position of the individual rooster is updated as follows.

$$\begin{aligned} x_i^j(t+1) = {} & x_i^j(t) \times (1 + Randn(0, \sigma^2)) \\ & + w(t) \times (x_{best}(t) - x_i^j(t)) \end{aligned} \tag{19}$$

(2) Individual hens based on weighting

In the chicken swarm optimization algorithm, all chicken individuals are aggregated at the location with the best fitness value in the search space of choice, which can easily lead to the loss of diversity in the later population, so to avoid this situation, we introduce a hen individual with a weighted center, which can participate in the competition with the whole swarm optimization individuals for the location of the global best individual and help the algorithm to jump out of the local best point. The update formula for this weighted central individual is as follows.

$$x_{center}^t = \sum_{i=1}^{H} c_i^t x_i^t \tag{20}$$

$$c_i^t = \frac{f_i^t}{\sum_{i=1}^{H} f_i^t} \tag{21}$$

In Equations (20)–(21), x_i^t denotes the position of individual hen i at the tth iteration, f_i^t denotes the fitness value of individual hen i, c_i^t is the normalized fitness value as a weighting factor after the tth iteration, x_{center}^t

denotes the position of the weighting center in the hen population at the tth iteration, and H is the number of hens. Therefore, the updated expression for the hen position is as follows:

$$x_i^j(t+1) = x_{center}^t + c_1 \times rand \times (x_{r1}^j(t) - x_i^j(t)) + c_2 \times rand \times (x_{r2}^j(t) - x_{i,j}(t)) \tag{22}$$

(3) Adaptive based following coefficient optimization

From the formula of chick's position update, it is found that the following coefficient is set by human, and its role is to ensure that the chicks follow the hen in the subgroup range. Since the individual chick's fitness value is the worst, it is very important to improve the individual chick's position update. In the process of chick position update, when the value of the following coefficient is set larger, it will cause the algorithm oscillation phenomenon in the later stage of the algorithm, and the operation speed of the algorithm will be affected, and the solution accuracy will also be reduced; on the contrary, it will affect the search speed of the algorithm and the solution accuracy will increase, so the artificial setting of the following coefficient obviously has a greater impact on the whole swarm optimization algorithm. Therefore, an adaptive step size is proposed. At the beginning of the algorithm, due to the change of individual chick positions, a larger step size is used to promote individual chicks to approach the optimal value in the local area, thus making the convergence speed improved. As the iteration progresses, the step length gradually decreases, which ensures that the individual chicks keep searching for the optimal value in depth around themselves, thus gradually approaching the optimal solution of the algorithm. This is not only beneficial to expand the search range, but also to improve the search accuracy of the algorithm and enhance the quality of the solution. The formula for the variation of the following coefficient is as follows.

$$F = t \times \frac{f(x_i^t)}{f_{obj}^{\max}(x_i^t) - f_{obj}^{\min}(x_i^t)} \Bigg) \tag{23}$$

In Equation (23), t is the current iteration number, $t_{\max}$ is the maximum iteration number, $f_{obj}^{\min}(x_i^t)$ and $f_{obj}^{\max}(x_i^t)$ denote the maximum and minimum fitness values of individual i in the current iteration number, respectively, and $f(x_i^t)$ denotes the fitness value of individual i.

4.3 Optimization Objective Function Design

In this paper, we mainly optimize two indicators, namely, the execution time and the consumption cost of the task. During the execution of the swarm optimization algorithm, the comparison of the fitness value is used as the selection condition of the best individual, and the execution of the algorithm is completed through continuous iterations. Let x_i^t be the position of the first swarm individual after t iterations, $Time(x_i^t)$ denotes the execution time function corresponding to the swarm individual, and $Cost(x_i^t)$ denotes the execution time function corresponding to the swarm individual. We use the individual fitness setting to represent the fitness function of individual chickens, i.e., the fitness function of execution time and consumption cost corresponding to each individual is expressed as follows.

$$F_1(x_i^t) = \frac{1}{Time(x_i^t)} \tag{24}$$

$$F_2(x_i^t) = \frac{1}{Cost(x_i^t)} \tag{25}$$

Equation (24) represents the fitness function of the completion time of a chicken individual, Equation (25) represents the fitness function of the consumption cost of a chicken individual, Equation (26) will take the derivative of the first 2 fitness functions and then take the logarithm to sum up the objective function as follows.

$$Fitness(x_i^t) = F_1(x_i^t) + F_2(x_i^t) \tag{26}$$

Therefore, solving the optimal scheduling scheme in cloud computing is solving the optimal swarm of individuals.

4.4 Algorithm Steps

Step 1: According to the requirements of cloud computing task scheduling in the Internet of Things, treat each cloud computing task scheduling scheme as a chicken swarm individual, and use the task scheduling function of Equation (4.26) as the fitness function of the chicken swarm individual

Step 2: Initialize the chicken swarm related parameters and set the maximum number of iterations

Step 3: Record the current best individual (optimal solution for each individual swarm) and the global best solution (optimal solution for the whole swarm)

Step 4: Update the cockerel position according to Equation (19);
Step 5: Update the hen position according to Equation (22);
Step 6: Update the chick position according to f Equation (23);
Step 7: Determining the current individual's fitness value compared with the global optimal fitness value, if the former is better than the latter, it is directly replaced, otherwise the iterative execution continues;
Step 8: If the number of iterations reaches the maximum number of iterations, go to step 9, otherwise go to step 3;
Step 9: Output the current optimal chicken individual, i.e., the optimal cloud computing task scheduling scheme.

5 Simulation Experiments

In order to further validate the performance of ICSO and its effectiveness in terms of time and cost of cloud computing task scheduling applied to IoT, we chose a hard-ware platform CPU of Core I5, memory of 8GDDR4, hard disk capacity of 1T, software system of Win10, simulation software of Matlab2012a. We chose CSO, DCSO [25], GCSO [26], ABCSO [27] as the comparison algorithms in this paper. The maximum number of iterations of the algorithm is set to 1000 and the population size is 100.

5.1 Algorithm Performance

We choose four benchmark test functions (as shown in Table 1) as the comparison objects in different dimensions. The number of iterations is also set to 1000 and the population size is 200, and the comparison data are shown in Table 2.

Table 1 The used benchmarking functions

No	Function	Test Function
F1	Rastrigin	$f(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$
F2	Schwefel1.2	$f(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)$
F3	Schwefel2.22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
F4	Rosenbrock	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$

Table 2 Comparison results of the four algorithms

Function	Dimension	Algorithm	Minimum	Maximum	Variance
F1	2	CSO	1.214E-03	2.143E-02	1.256E-02
		DCSO	1.112E-03	1.981E-02	1.243E-02
		GCSO	2.235E-04	1.874E-03	2.825E-03
		ABCSO	1.745E-05	1.963E-04	2.145E-03
		ICSO5	0	0.149E-06	1.745E-05
	10	CSO	1.785E-09	0.245E-08	1.176E-07
		DCSO	0.638E-12	1.874E-10	8.475E-11
		GCSO	1.741E-11	2.746E-10	7.149E-10
		ABCSO	2.874E-12	6.852E-11	6.234E-11
		ICSO	3.547E-15	8.145E-15	4.152E-14
	30	CSO	2.487E-20	3.258E-17	6.254E-18
		DCSO	9.574E-24	6.471E-23	8.741E-23
		GCSO	7.149E-25	4.178E-24	7.724E-23
		ABCSO	9.587E-27	6.174E-26	9.035E-25
		ICSO	8.347E-32	2.145E-30	4.719E-30
F2	2	CSO	0.197	0.189	0.182
		DCSO	0.153	0.161	0.144
		GCSO	0.158	0.174	0.158
		ABCSO	0.136	0.152	0.147
		ICSO	0.129	0.131	0.128
	10	CSO	1.424E+09	2.475E+09	4.145E+02
		DCSO	6.314E+08	5.124E+09	6.285E+08
		GCSO	8.143E+08	5.178E+09	3.187E+08
		ABCSO	9.595E+08	8.593E+09	5.743E+10
		ICSO	9.587E+07	8.386E+08	5.707E+10
	30	CSO	5.784E+12	6.287E+13	6.974E+12
		DCSO	6.748E+10	9.587E+11	6.145E+10
		GCSO	6.742E+10	9.415E+10	6.217E+09
		ABCSO	9.012E+09	5.287E+10	6.187E+08
		ICSO	2.854E+08	4.874E+09	0.478E+07
F3	2	CSO	1.587E+02	1.872E+02	1.457E+02
		DCSO	1.958E+01	2.749E+01	1.749E+01
		GCSO	3.147E+01	6.214E+01	1.254E+01
		ABCSO	6.214E+01	6.782E+01	2.178E+01
		ICSO	2.142E+01	6.352E+01	1.522E+01
	10	CSO	8.475E+03	8.974E+03	5.745E+03
		DCSO	4.741E+03	5.479E+03	6.742E+03
		GCSO	5.412E+03	6.178E+03	6.872E+04

(*Continued*)

Table 2 Continued

Function	Dimension	Algorithm	Minimum	Maximum	Variance
		ABCSO	6.125E+03	8.475E+03	6.784E+02
		ICSO	6.017E+03	6.197E+03	6.178E+03
	30	CSO	9.148E+06	8.174E+06	6.287E+05
		DCSO	6.196E+06	8.746E+05	7.149E+05
		GCSO	5.719E+06	5.749E+06	6.217E+05
		ABCSO	4.196E+06	6.749E+06	6.147E+05
		ICSO	3.195E+05	4.196E+05	6.178E+05
F4	2	CSO	1.258E-02	1.965E-02	1.847E-02
		DCSO	1.854E-03	1.856E-03	1.578E-03
		GCSO	1.715E-04	1.729E-04	1.679E-04
		ABCSO	5.178E-05	7.015E-05	6.178E-05
		ICSO	0	2.748E-07	3.471E-08
	10	CSO	6.785E-09	6.479E-09	9.125E-08
		DCSO	6.121E-09	8.159E-09	6.171E-09
		GCSO	8.179E-10	9.128E-10	8.138E-10
		ABCSO	8.128E-10	7.158E-10	3.745E-10
		ICSO	8.147E-12	2.485E-11	9.025E-11
	30	CSO	8.128E-18	8.017E-17	9.012E-17
		DCSO	9.176E-19	6.285E-18	6.258E-17
		GCSO	5.148E-21	6.918E-20	5.417E-20
		ABCSO	6.258E-22	6.181E-21	9.158E-21
		ICSO	6.148E-25	6.147E-24	7.182E-22

Table 2 shows the performance comparison results of different algorithms' perfor-mance in different dimensions. We find from the data in the table that the ICSO algorithm proposed in this paper obtains better results in the Minimum, Maximum, and Variance metrics data in four different benchmark functions. Especially in Minimum, this paper's algorithm actually reaches 0 in the two benchmark functions of F1 and F4 with dimension 2, which shows that the performance effect of the improved algorithm is obvious. In the benchmark function of F2, when the dimension is 10, the difference between the ICSO algorithm and the ABCCSO is almost very small in the three indexes, but it still has a more obvious advantage in other dimensions. In the benchmark function of F3, the ICSO algorithm has a more obvious advantage over other algorithms in terms of the three indicators. Through the comparison of the above data, we found that the ICSO algorithm obtained through the optimization of the three dimensions has superior performance.

5.2 Comparison of Cloud Computing Scheduling Indexes

This subsection focuses on verifying the results of the comparison between the ICSO algorithm and the comparison algorithm in terms of time and cost in scheduling cloud computing tasks in IoT. The number of novel tasks is set to [10000,30000] and the number of small tasks is set to [100,1000].

(1) Completion time comparison

Figures 1–2 show the results of the completion time comparison of the five algo-rithms under different numbers of tasks. Figure 1 shows the results of the five algo-rithms in the conditions of the completion time under the conditions of a smaller number of tasks, from the time curve in the figure found that with the increase in the number of tasks, the five algorithms curve shows an upward trend, but the CSO algo-rithm has a more obvious upward trend, while the other four algorithms curve upward trend is more gentle, due to the fact that these four algorithms are all in the CSO algo-rithm on a different degree of improvement, so with the CSO algorithm There is a clear difference between the curves, but among the four algorithms, the advantage of ICSO is more obvious compared to the other three algorithms, which shows that ICSO has a better advantage in task completion, saving 25.8%, 9.3%, 8.8% and 7.5% compared to CSO, DCSO, GCSO and ABCSO, respectively. Figure 2 shows the results of the five algorithms in terms of completion time for

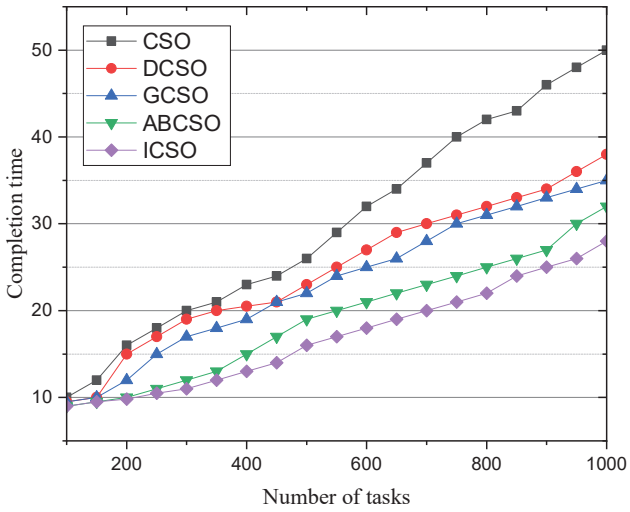

Figure 1 Comparison of the four algorithms on completion times under small tasks.

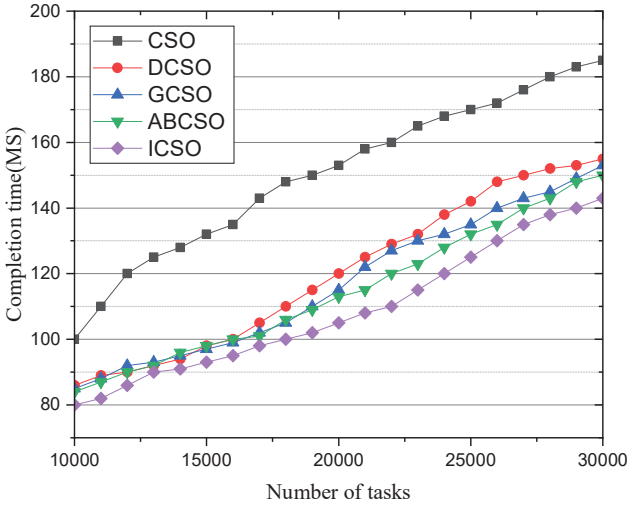

Figure 2 Comparison of the four algorithms on completion times under large tasks.

conditions with a large number of tasks. From the curve results in the figure, it is found that the four algorithms DCSO, GCSO, ABCSO and ICSO have a significant advantage over CSO and the difference in task completion time between these four algorithms is not significant. However, ICSO saves 30.8%, 8.3%, 7.8% and 6.3% compared to CSO, DCSO, GCSO and ABCSO respectively. By analyzing the above task scheduling time results, we found that the effect of updating the rooster position, hen position update and following coefficient update is obvious, which improves the algorithm performance and thus reduces the task completion time.

(2) Comparison of consumption cost

Figures 3–4 show the results of the cost of completion comparison of the five algorithms under different number of tasks. Figure 3 shows the results of the five algorithms in the conditions of task energy consumption under the condition of a small number of tasks, from the energy consumption curve in the figure, it is found that, with the increase of the number of tasks, the five algorithms' energy consumption curve shows an upward trend, and the upward trend of the CSO algorithm is almost a straight line, while the other four algorithms' energy consumption curves have different degrees of twists and turns during the upward process, but among these four algorithms, the ICSO has more obvious advantages compared with the other three algorithms. The advantage is more obvious, which shows that ICSO has a

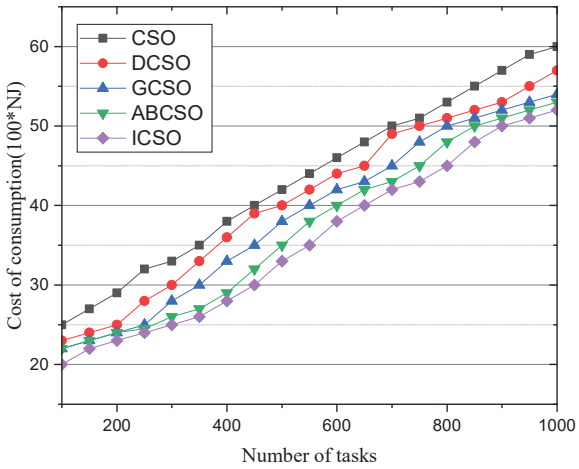

Figure 3 Comparison of consumption costs of four algorithms for Small tasks.

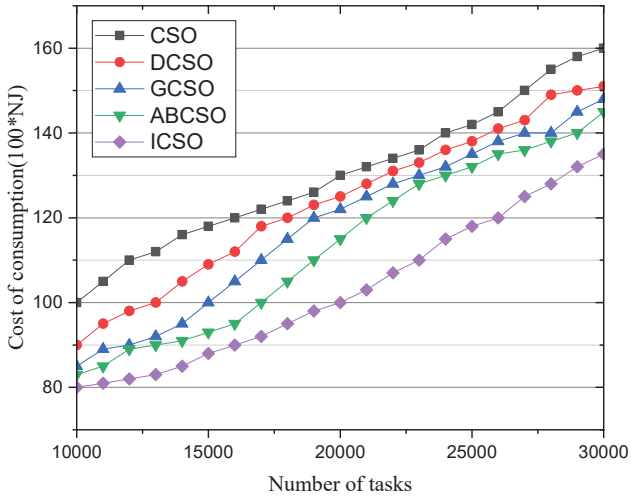

Figure 4 Comparison of the consumption costs of the four algorithms under large tasks.

better advantage in task energy consumption, saving 11.8%, 10.3%, 8.8% and 7.5% compared with CSO, DCSO, GCSO and ABCSO respectively. Figure 4 shows the results of the five algorithms in terms of energy consumption for a large number of tasks. From the curves in the figure, it is found that the four algorithms DCSO, GCSO, ABCSO and ICSO have obvious advantages over CSO, while the difference in task completion energy consumption between

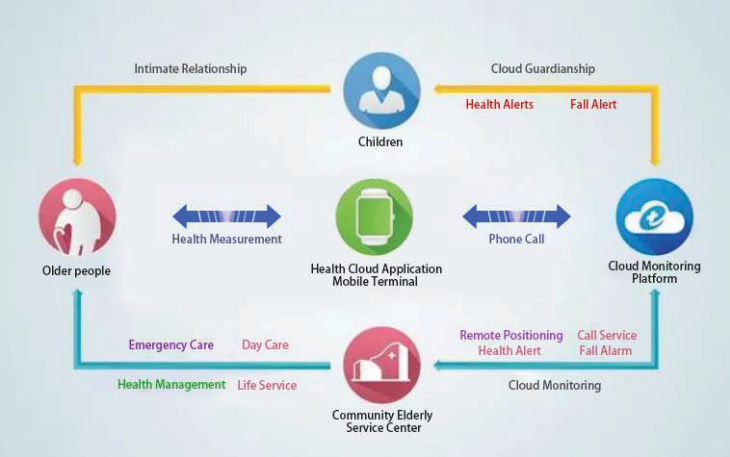

Figure 5 IoT based health monitoring service platform.

these four algorithms is obvious, and ICSO has a certain advantage over DCSO, GCSO and ABCSO in general, but ICSO saves 25.8% compared to CSO, DCSO, GCSO and ABCSO In general, however, ICSO saves 25.8%, 11.2%, 10.8% and 9.3% compared to CSO, DCSO, GCSO and ABCSO, respectively. This indicates that ICSO algorithm adapts to task scheduling under large tasks and can effectively reduce energy consumption.

5.3 IoT Cloud Platform Application

We had been involved in the development of an IoT based health monitoring service platform, the structure of this monitoring platform is shown in Figure 5, we use this platform to manage 2000 elderly people in the community, through which we monitor and receive different requests from these elderly people and respond to them. We randomly selected 1000 elderly people and divided 1000 elderly people into a group of 10 groups, i.e., numbered 1–10, Figure 6 shows the average response time of the platform simulating the same moment of dealing with the emergency help issued by 1000 elderly people, and the results from the figure show that the five algorithms responded to the help issued by the elderly people in a relatively rapid time, and the ICSO algorithm in the 1000 elderly people's Response time for the help request task is the shortest, which indicates that the use of this algorithm has good results in terms of task processing time. Figure 7 shows the energy consumption of the cloud server for processing 1000 elderly

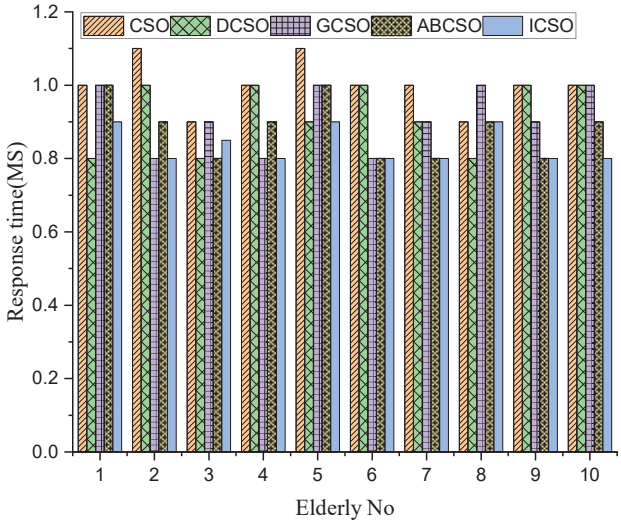

Figure 6 Requesting time of 1000 elderly requests in the cloud computing platform.

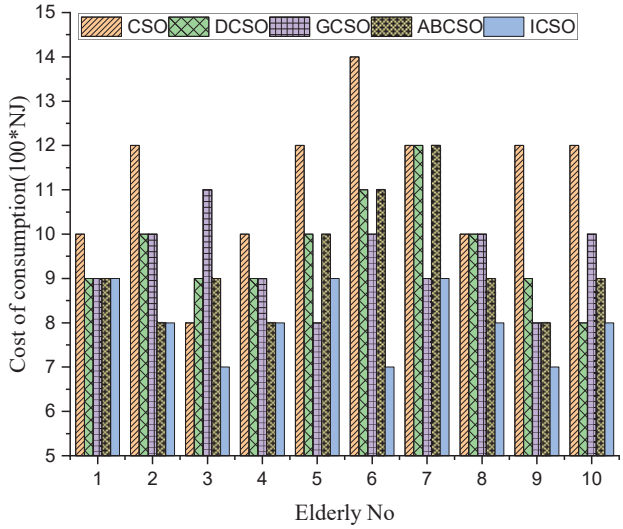

Figure 7 Energy consumption of 1000 elderly requests in the cloud computing platform.

people, from the figure, it is found that the CSO algorithm produces very large fluctuations in energy consumption and the highest value of energy consumption in the completion of the response, while the other four algorithms are significantly lower than the CSO, while in the energy consumption

of group number 1, the four algorithms have the same result, and the ICSO algorithm has no advantage, and in the energy consumption of group number 5, the ICSO algorithm is lower than the GCSO algorithm, The reason for this may be that the complexity of the ICSO algorithm leads to an increase in the running time of the algorithm and thus improves the energy consumption, but from the overall numerical results, the ICSO algorithm has a lower energy consumption.

6 Conclusions

In this paper, we propose a cloud computing task scheduling scheme under the Internet of Things, which employs an improved chicken swarming algorithm to solve the problems of long completion time and high cost consumption in cloud computing task scheduling. Specifically, we first outline the time- and cost-based functions, and then improve the CSO algorithm by updating three aspects: rooster, hen, and following coefficients. Through simulation experiments, we demonstrate that our proposed ICSO algorithm achieves superior performance in benchmark functional tests and improves the scheduling results of cloud computing tasks with various con-figurations in terms of time and cost, and illustrates the superior scheduling performance of the algorithm by applying it through an IoT cloud platform. The future of cloud computing task scheduling under IoT moves towards edge computing, automation and intelligence, multimodal and multimedia data, and edge smart devices to accommodate the growing and diverse IoT application requirements.

References

[1] Kim W. Cloud computing: Today and tomorrow[J]. J. Object Technol, 2009, 8(1): 65–72.

[2] Sadeeq M M, Abdulkareem N M, Zeebaree S R M, et al. IoT and Cloud computing issues, challenges and opportunities: A review[J]. Qubahan Academic Journal, 2021, 1(2): 1–7.

[3] Arunarani A R, Manjula D, Sugumaran V. Task scheduling techniques in cloud computing: A literature survey[J]. Future Generation Computer Systems, 2019, 91: 407–415.

[4] Armbrust M, Fox A, Griffith R, et al. A view of cloud computing[J]. Communications of the ACM, 2010, 53(4): 50–58.

[5] Gao J, Wang H, Shen H. Task failure prediction in cloud data centers using deep learning[J]. IEEE transactions on services computing, 2020, 15(3): 1411–1422.
[6] Kalra M, Singh S. A review of metaheuristic scheduling techniques in cloud computing[J]. Egyptian informatics journal, 2015, 16(3): 275–295.
[7] Meng X, Liu Y, Gao X, et al. A new bio-inspired algorithm: chicken swarm optimization[C]. International conference in swarm intelligence. Springer, Cham, 2014: 86–94.
[8] Arunarani A R, Manjula D, Sugumaran V. Task scheduling techniques in cloud computing: A literature survey[J]. Future Generation Computer Systems, 2019, 91: 407–415.
[9] Cheng L, Kotoulas S. Efficient skew handling for outer joins in a cloud computing environment. IEEE Transactions on Cloud Computing[J]. 2015 Oct 7;6(2): 558–571.
[10] Cheng F, Huang Y, Tanpure B, Sawalani P, Cheng L, Liu C. Cost-aware job scheduling for cloud instances using deep reinforcement learning. Cluster Computing[J]. 2022, 25(1): 619–631.
[11] Patra M K, Misra S, Sahoo B, et al. GWO-Based Simulated Annealing Approach for Load Balancing in Cloud for Hosting Container as a Service[J]. Applied Sciences, 2022, 12(21): 11115.
[12] Bezdan T, Zivkovic M, Bacanin N, et al. Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm[J]. Journal of Intelligent & Fuzzy Systems, 2022, 42(1): 411–423.
[13] Chen X, Cheng L, Liu C, et al. A WOA-Based Optimization Approach for Task Scheduling in Cloud Computing Systems[J]. IEEE Systems Journal. 2020, 14(3): 3117–3128.
[14] Mangalampalli S, Swain S K, Mangalampalli V K. Multi Objective Task Scheduling in Cloud Computing Using Cat Swarm Optimization Algorithm[J]. Arabian Journal for Science and Engineering, 2022, 47(2): 1821–1830.
[15] Velliangiri S, Karthikeyan P, Xavier V M A, et al. Hybrid electro search with genetic algorithm for task scheduling in cloud computing[J]. Ain Shams Engineering Journal, 2021, 12(1): 631–639.
[16] Abualigah L, Diabat A. A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments[J]. Cluster Computing, 2021, 24(1): 205–223.

[17] Muthulakshmi B, Somasundaram K. A hybrid ABC-SA based optimized scheduling and resource allocation for cloud environment[J]. Cluster Computing, 2019, 22(5): 10769–10777.

[18] Manikandan N, Gobalakrishnan N, Pradeep K. Bee optimization based random double adaptive whale optimization model for task scheduling in cloud computing environment[J]. Computer Communications, 2022, 187: 35–44.

[19] Chen X, Long D. Task scheduling of cloud computing using integrated particle swarm algorithm and ant colony algorithm[J]. Cluster Computing, 2019, 22(2): 2761–2769.

[20] Liu J, Cheng L. SwiftS: a dependency-aware and resource efficient scheduling for high throughput in clouds. InIEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) 2021 May 10 (pp. 1–2). IEEE.

[21] Mao Y, Sharma V, Zheng W, Cheng L, Guan Q, Li A. Elastic resource management for deep learning applications in a container cluster[J]. IEEE Transactions on Cloud Computing. 2022, 11(2): 2204–2216.

[22] Jemmali M, Denden M, Boulila W, et al. A Novel Model Based on Window-Pass Preferences for Data Emergency Aware Scheduling in Computer Networks[J]. IEEE Transactions on Industrial Informatics, 2022, 18(11): 7880–7888.

[23] Patrick N V, Misra S, Adetiba E, et al. An Incremental Load Balancing Algorithm in Federated Cloud Environment[M]//Data, Engineering and Applications: Select Proceedings of IDEA 2021. Singapore: Springer Nature Singapore, 2022: 395–408.

[24] Yu D H. Research on The Cloud Computing Task Scheduling Strategy Based on Multidimensional QoS Constraints[D]. Chongqing University of Posts and Telecommunications, 2021

[25] Wang Z, Zhang W, Guo Y, et al. A multi-objective chicken swarm optimization algorithm based on dual external archive with various elites[J]. Applied Soft Compu-ting, 2023, 133: 109920.

[26] Basha A J, Aswini S, Aarthini S, et al. Genetic-Chicken Swarm Algorithm for Minimizing Energy in Wireless Sensor Network[J]. Computer Systems Science & Engineering, 2023, 44(2): 1451–1466.

[27] Pushpa R, Siddappa M. Fractional Artificial Bee Chicken Swarm Optimization technique for QoS aware virtual machine placement in cloud[J]. Concurrency and Computation: Practice and Experience, 2023, 35(4): e7532.

Biographies

Shizheng Liu received his bachelor's degree in accounting from Wenzhou Business College in 2015 and a master's degree in international business from the University of Lincoln in 2016. He is currently a lecturer in Zhejiang Industry Polytechnic College, and his research interests are international trade and business, cross border e-commerce, social media marketing and entrepreneurship, cloud computing.

Xuan Chen is an associate professor at Zhejiang Industry Polytechnic College. He received B.S. degree in Information Management and Information Systems from Zhengzhou University of Aeronautics in 2003. He received M.S. degree in software engineering from University of Electronic Science and Technology of China in 2010. His research interests include cloud computing and algorithm design.

Feng Cheng is an Assistant Professor with the Southwest Jiaotong University, Chengdu, China. She received the B.S. and M.S. degrees in mathematics from Southwest Jiaotong University in 2006 and 2009 respectively. In 2013, she received the Ph.D. degree in management from Tongji University, Shanghai, China. Her research interests include deep learning, stochastic operations and algorithms.