Effective Task Scheduling in Cloud Computing Based on Improved Social Learning Optimization Algorithm

https://doi.org/10.3991/ijoe.v13i06.6695

Zhizhong Liu
School of Henan Polytechnic University, Jiaozuo, China
lzzmff@126.com

Jingxuan Qin
School of Henan Polytechnic University, Jiaozuo, China
qjxwudi@163.com

Weiping Peng
School of Henan Polytechnic University, Jiaozuo, China
pwp9999@hpu.edu.cn

Hao Chao
School of Henan Polytechnic University, Jiaozuo, China
chaohao@hpu.edu.cn

Abstract—For the typical optimal problem of task scheduling in cloud computing, this paper proposes a novel resource scheduling algorithm based on Social Learning Optimization Algorithm (SLO). SLO is a new swarm intelligence algorithm which is proposed by simulating the evolution process of human intelligence and has better optimization mechanism and optimization performance. This paper proposes two learning operators for task scheduling in cloud computing after analyzing the characteristics of the problem of task scheduling; then, by introducing the Small Position Value (SPV) method, the two learning operators with continuous nature essence are enabled to solve the problem of task scheduling, and then the improved SLO is employed to solve the problem of cloud resource optimal scheduling. Finally, the performance of improved SLO is compared with existing research work on the CloudSim platform. Experimental results show that the approach proposed in this paper has better global optimization ability and convergence speed.

Keywords—Cloud Computing, Task Scheduling, Social Learning Optimization Algorithm, Service of Quality

1 Introduction

Cloud computing is a business service model and computing model which integrates grid computing, parallel computing and P2P technologies[1]. Furthermore, it is a pay-as-you-go model which provides resources at lower costs with greater reliability and delivers the resources by means of virtualization technologies [2]. The goal of cloud computing is to provide on-demand computing service with high reliability, scalability, and availability [3].

As the emerging trend in distributed computing recently, cloud computing provided many advantage to end-users, such as data ubiquity, flexibility of access, high availability of resources, and flexibility. The task scheduling problem is a key problem which influences the quality of Cloud computing, and it is an NP-hard problem [5]. Recently, some research work focuses on this problem and propose several methods, most of these methods are designed based on evolutionary algorithms, as this kind of algorithm has strong heuristic algorithm optimization ability. Some evolutionary algorithms (such as genetic algorithm [6-9], particle swarm algorithm [10-12], ant colony algorithm [13, 14], bee colony [15], and cuckoo algorithm [4, 16]) has been used to solve the problem of task scheduling in cloud computing. However, above evolutionary algorithms still have some shortcomings, such as slow convergence speed, easy to fall into local optimum, etc. These shortcomings reduce the efficiency of task scheduling problem solving.

To solve the task scheduling problem efficiently, this paper proposes a new approach based on improved Social Learning Optimization Algorithm (SLO) [17]. SLO is a new evolutionary algorithm which is proposed by simulating the evolution process of human intelligence, it consists of three co-evolution spaces: micro-space, learning space and belief space. These three spaces construct a closed co-evolution process which has better evolution mechanism. SLO provides a heuristic optimization algorithm model for solving optimization problems, and evolution process in each space could be realized by designing corresponding optimization operators.

For solving task scheduling problem efficiently, this work designs some optimization operators (such as crossover operation, mutation operation, observational learning, imitation learning operation, accept operation and influence operation) for different spaces of SLO, moreover, Small Position Value (SPV) method is used to discrete the individuals of improved SLO, so as to enable the improved SLO could solve the discrete task scheduling problem. Finally, the performance of improved SLO is compared with other methods on the CloudSim platform. Experimental results show that improved SLO has better performance than that of other evolutionary algorithms. The main contribution of this work is presented as follows.

- It proposes optimization operators of the three spaces in SLO and substantiate the SLO for solving the task scheduling problem;
- SPV method is used to discrete the individuals of improved SLO, so as to enable the improved SLO could solve the discrete task scheduling problem.

The structure of this work is organized as follows: Section 1 illustrates the related work; Section 2 introduces the task scheduling problem; Section 3 introduces the SLO algorithm; Section 4 presents the task scheduling method based on improved SLO. Section 5 demonstrates the experimental results; Section 6 concludes the work.

2 Related works

Task scheduling is one of the most important and critical problems in cloud computing, moreover, task scheduling is also an NP-complete problem [18]. Nowadays such problems are often addressed by using heuristic methods. Many researchers have even more proven that heuristic algorithms can better to find the optimal solution for task scheduling in cloud environment.

Most of these works are focused on minimize the makespan or save the execution cost in cloud computing. For example, Nima et al. [4] have proposed a task scheduling evolutionary algorithm in cloud computing based on Cuckoo Search Algorithm (CSA), which used obligate brood parasitic behavior of some cuckoo species in combination with the Lévy flight behavior of some birds and fruit flies, and improves the coverage speed. Similarly, Kim et al.[19] have developed biogeography-based optimization (BBO) for task scheduling, this algorithm in large size problems is performs satisfactorily than other optimization problems such as genetic algorithm (GA), particle swarm optimization (PSO) and simulated annealing (SA). Solmaz Abdi et al. [11] have proposed a modified PSO algorithm to allocate tasks in cloud computing environment which combine with shortest job to fastest processor algorithm (SJFP) to initializes particles minimize the makespan. Nidhi Bansala et al. [20] have introduced the cost parameter to the QoS-driven scheduling algorithm minimizing the total allocation cost.

However, there are some other parameters which influence the scheduling of tasks and utilization of resources. So some scholars have carried out the related research about multi-dimensional optimization for cloud task scheduling problem. Jacob et al. [21] have proposed a task scheduling algorithm based on bacterial foraging optimization algorithm. The algorithm considered the QoS parameters such as cost, reliability and makespan together which is able to meet the needs of the users. Sun et al. [22] have presented a heuristic cloud resource scheduling algorithm with application preference which based on the immune clonal algorithm of rapid multi-objective optimization. According to the preference priority of non-dominated antibodies, the algorithm can improve the convergence ability efficiently. He et al. [23] have proposed a PSO-based adaptive multi-objective task scheduling strategy which considers four objectives includes optimal resource utilization, task completion time, average cost and average energy consumption. In order to maintain the particle diversity, the method adopt acceleration coefficient to adjust parameters adaptively and improve population diversity.

Recently, abundant research works have promoted the development of task scheduling to some extent, but these methods still has some limitations. For instance, the difference of algorithm performance in different cloud computing environment would be more evident, the solution is easy to be run into local optimum or the task scheduling optimization target is single and so on. To overcome the above deficiencies, this paper proposes a novel task scheduling algorithm based on social learning optimization algorithm (SLO) [17], moreover, a new objective function consisting of task completion time, cost and energy consumption is proposed. Furthermore, observation learning operator and imitation learning operator in SLO are designed for improving the convergence speed and the quality of task scheduling services.

3 Task scheduling Problem

The quaternion of task scheduling model $M = (S, V, F, \theta)$ is made up of *n* tasks and *m* virtual resources, where S is a set of n tasks, V is a set of m resources, *F* stand for the objective function of resource scheduling and θ is the resource scheduling optimization algorithm. In this paper we make the following assumptions about cloud task scheduling model [24]:

(1) The performance of the virtual machine meets the requirements of any task;

(2) Each task can only be performed by one virtual machine;

(3) The effect of task transmission time on model optimization is not considered.

This paper only considers the situation with one user (multiple users can cycle execution), the characteristics of scheduling model are described as follows:

(1)The task set $S = \{s_1, s_2, \cdots, s_n\}$ submitted by the user, which are indivisible and independent between n tasks. SL_i represent the length of *i*-th task that always determines by the user.

(2)The virtual resources set $V = \{v_1, v_2, \cdots, v_i, \cdots, v_m\}$, where each virtual resource v_i has multiple attributes, such as computing power, memory and bandwidth, etc. In this paper, we mainly consider the computing power of virtual machine $VC = \{vc_1, vc_2, \cdots, vc_i, \cdots vc_m\}$ where vc_i represent the computing power of the virtual machine *i*.

(3) Task execution time matrix, $Time = \left\{time_{ij}\right\}\left(i = 1, 2, \cdots n; j = 1, 2, \cdots m\right)$ where $time_{ij}$ is the execution time of task *i* on a virtual machine *j*, the $time_{ij}$ is calculated by formula (1):

$$time_{ij} = SL_i / vc_j \tag{1}$$

(4) Distribution matrix

$$X = \begin{vmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{vmatrix}, \text{ where } x_{ij} = \begin{cases} 1, & \text{if a task of } s_i \text{ will be run on } v_j \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

We use the matrix to represent whether the task *i* is assigned to the virtual machine *j*. Where $x_{ij} = 1$ indicates that the task *i* is allocated to the virtual resource *j*, otherwise not.

In order to improve the QoS in cloud computing environments, we considered four factors in cloud task scheduling include the *task completion time*, *cost*, *energy consumption* and *resource utilization* [23].

Factor 1: *TIME* express the *task completion time*. In cloud computing environments, RET_j indicates the execution time of virtual resource j which can be calculated by a function $RET_j = \sum_{i=1}^{n} x_{ij} * time_{ij}$. Because the virtual resources are run in parallel, the completion time of all tasks is obtained according to (3):

$$TIME = \max_{j=1}^{m} RET_j \tag{3}$$

Factor 2: *C* means the *cost* in task scheduling. For illustrate the superiority of the algorithm better in task scheduling, we assume that the tasks are mutually independent. Matrix $RC\{j\}(j = 1,2,\cdots,m)$ is used to express the cost of virtual resource j per time unit. The total execution cost for all cloud tasks can be calculated according to formula (4):

$$C = \sum_{j=1}^{m} RET_j * RC(j) \tag{4}$$

Factor 3: *The energy consumption* used matrix $E = \{e_{ij}\}(i = 1,2,\cdots,n; j = 1,2,\cdots,m)$ to express. Where e_{ij} represents the energy consumption when task i runs on resource j. The total energy consumption *EC* for finish all tasks are calculated by (5);

$$EC = \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} * e_{ij} \tag{5}$$

Factor 4: *RU* represents *the resource utilization* in cloud computing task scheduling problem. When resource utilization in cloud computing is higher, the less energy consumption. Therefore, improving the utilization efficiency of cloud resources can effectively reduce the cost of cloud service providers. The resource utilization [26] is according to the formula (6):

$$RU = 1 - \sum_{j=1}^{m} \left(Total - RET_j\right) / \left(m * Total\right) \tag{6}$$

Above all, we need to find the most reasonable scheduling scheme and determine the final decision matrix. The goal of task scheduling in this paper is minimize the task completion time, cost, energy consumption and resource utilization. The resource utilization is proportional to the execution time; therefore we only need to consider three main factors: task completion time, cost and energy consumption. The objective function of the problem of task scheduling is defined as (7):

$$F(x) = \omega_1 * Total + \omega_2 * EC + \omega_3 * \mathrm{C} \tag{7}$$

Where ω1, ω2, ω3 respectively represent the weights of the task completion time, the cost and the energy consumption in the objective function, and $\omega_1+\omega_2+\omega_3=1$, $\omega_1\geq0$, $\omega2\geq0$, $\omega_3\geq0$.

4 Task scheduling Problem

As we know, in all groups of organisms, the intelligence of humans is the highest among all of the animal groups. Humans always enhance their intelligence by learning from others' behavior; this is the key reason why human intelligence is higher than that of other animals. In human society, through observation and the imitation of others' behaviors, humans can improve their intelligence quickly; this behavior has been called imitation learning or social learning.

Social learning optimization algorithm Paradigm (SLO) [17] is a swarm intelligence algorithm which inspired by the evolution process of human intelligence. The framework of SLO is constructed based on the framework and key functions of the culture algorithm (CA) [25]. On the basis of the framework of CA, a new micro-space is added to the bottom of the CA framework to simulate individual genetic evolution. The population space of CA is taken as the learning space of SLO, while the top layer of the CA framework is taken as the belief space. The framework of SLO is presented in Fig. 1.

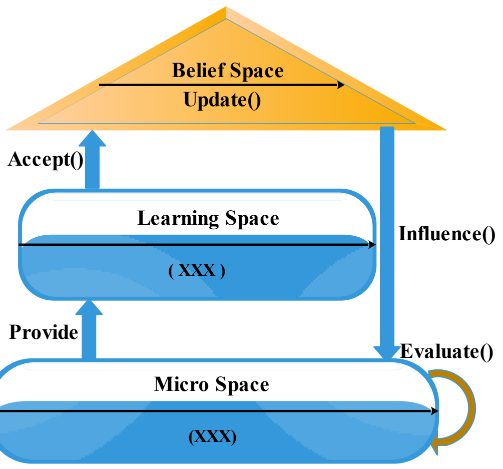


Fig. 1. Framework of SLO

SLO consists of three co-evolution spaces: the bottom is the micro-space, where individuals' genetic evolution takes places; the middle layer is the learning space, where individuals enhance their intelligence through learning and where better knowledge is extracted and delivered to the top layer; the top layer is the belief space, where culture is formed through knowledge accumulation and is used to guide individuals' evolution regularly in the macro-space. The general algorithmic structure of the SLO is described in Alg_1.

```
Alg_1. Algorithmic structure of SLO
Step1: Initialization
        Generate individuals in the micro-space;
REPEAT
Step2: Individual Genetic Evolution Phase
        Carry out individual genetic evolution in the micro-space and provide individuals
        to the learning space;
Step3: Individual Learning Phase
        For each individual, carry out the imitation learning and observational learning;
Step4: Culture Influence Phase
        Establishes the culture; apply culture to regularly influence the individual genetic
        evolution in the micro- space;
Memorize the best solution achieved so far;
UNTIL (the termination condition is satisfied);
```

5 Task Scheduling Based on improved SLO

SLO is an optimization model, when using it to solve optimization problems, it needs to design concrete operators in each spaces. For solving the task scheduling problem efficiently, this work designs several operators for each spaces of SLO.

5.1 Operators in there space of SLO

Operators in the Micro-space: The evolution process in micro-space is mainly based on genetic algorithm, which is used to imitate human intelligence evolution, and this process is divided into three operations: selection, crossover and mutation.

Initialization operation: Initial population in the Micro-space are randomly generated, the value of each dimension is obtained according to formula (8):

$$POP_{ij} = POP_{\min} + (POP_{\max} - POP_{\min}) * rand \qquad (8)$$

where POP_{ij} represents the *j*-th dimension of the individuals *POP*$_i$, *i* =1,2,…,*popsize*; *j*=1,2,…,*n*, in which *popsize* indicates the individual quantity, *n* is the number of tasks and *rand* is a random number between 0 and 1. $POP_{\max} = 5$, $POP_{\min} = -0.5$.

Individuals generated by formula (8) is a continuous variable, but the task scheduling in cloud computing is a discrete problem, so this paper introduce the small position value (SPV) rule [27] to convert the continuous individual $POP_i = [pop_{i1}, pop_{i2}, \cdots, pop_{in}]$ to discrete individual $\pi_i = [\pi_{i1}, \pi_{i2}, \cdots, \pi_{in}]$.

Formula (9) is used to change the discrete individual $\pi_i = [\pi_{i1}, \pi_{i2}, \cdots, \pi_{in}]$ to a vector $R_i = [r_{i1}, r_{i2}, \cdots, r_{in}]$, which is used to calculate the process times of tasks.

$$R_{ij} = \pi_{ij} \bmod m + 1 \tag{9}$$

Here *m* represents the amount of resource. An example to illustrate the conversion process for 10 tasks and 4 resources is illustrated in Table 1.

Table 1. Convert *POP*

Dimension *j*					
Variable	1	2	3	4	5
POP_i	2.16	-0.42	-0.28	1.48	4.13
π_i	6	1	3	5	9
R_i	2	2	4	1	5
Variable	6	7	8	9	10
pop_i	3.12	1.08	-0.34	4.41	3.70
π_i	7	4	2	10	8
R_i	3	5	3	1	4

Selection Operation: In this paper, the roulette wheel is used to select excellent individual from the population. Assume that the population size is *P*, for an individual POP$_i$ whose fitness value is f_i and its selection probability *Select$_i$* can be calculated by (10). After obtained the selection probability of each individual, then, the selection operation can carry out based on these probabilities.

$$Select_i = f_i / \sum_{j=1}^{popsize} f_j \tag{10}$$

Crossover Operation: For the crossover operation, assume that two different individuals which are randomly selected from the population are POP_i and POP_j, through crossover operation, two new different individuals are generated, this process can be illustrated as Fig. 2.

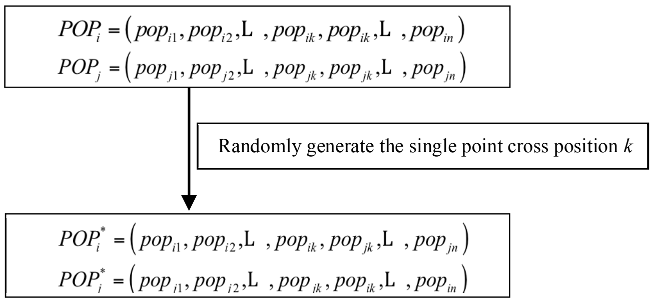

$POP_i = (pop_{i1}, pop_{i2}, \text{L}\ , pop_{ik}, pop_{ik}, \text{L}\ , pop_{in})$
$POP_j = (pop_{j1}, pop_{j2}, \text{L}\ , pop_{jk}, pop_{jk}, \text{L}\ , pop_{jn})$
Randomly generate the single point cross position *k*
$POP_i^* = (pop_{i1}, pop_{i2}, \text{L}\ , pop_{ik}, pop_{jk}, \text{L}\ , pop_{jn})$
$POP_i^* = (pop_{i1}, pop_{i2}, \text{L}\ , pop_{ik}, pop_{ik}, \text{L}\ , pop_{in})$

Fig. 2. schematic diagram of crossover operator

After the crossover operation, carry out the greedy operation that is to keep up two better individuals among the four individuals.

Mutation Operation: In this work, the single point mutation method is adopted, that is, randomly generate a new gene in the range of $[POP_{\min}, POP_{\max}]$ and to replace the randomly selected gene point in the individual. Then, evaluate the new individual and old individual, carry out the greedy operation and keep the better one.

Operators in the Learning-space: In the learning space, social learning optimization algorithm is simulating the learning process about human society and promoting population evolution by learning from the best individuals. The learning operators of the learning-space include observation learning and imitation learning.

Observation learning: In society, individuals will retain a portion of their knowledge when they observing and learning from elite individuals. Inspired by this phenomenon, this work introduces the inertia coefficient to observation learning operator, so as to avoid jumping into the local optimum, enhance the searching ability and convergence speed of SLO.

Let $POP_i = (pop_{i1}, pop_{i2}, \mathrm{L}\ , pop_{ik}, \mathrm{L}\ , pop_{in})$ indicates any individual in a population, $POP^{cg} = (pop_1^{cg}, pop_2^{cg}, \mathrm{L}\ , pop_j^{cg}, \mathrm{L}\ , pop_n^{cg})$ represents the current global optimal solution, $pop_i, pop_j^{cg} \in [POP_{\min}, POP_{\max}]$ and $POP_i^* = (pop_{i1}^*, pop_{i2}^*, \mathrm{L}\ , pop_{ik}^*, \mathrm{L}\ , pop_{in}^*)$ indicate the new individual produced by the observation learning. Observation learning operations are defined as (11)

$$pop_{ik}^* = \omega \cdot pop_{ik} + pop_k^{cg} \cdot (1 - \sin(\alpha)) \tag{11}$$

Where pop_{ik}^* represents the k-th dimension of new individuals POP_i^*; ω indicates the learning inertia weight and $\omega \in (0,1)$, this work set $\omega = 0.5$; $\sin(\alpha)$ denotes the learning disturbance factor and $\alpha \in [2, \pi]$; $\omega \cdot x_{ik}$ indicates the self-retaining part of an individual in the observation learning; $x_{ik}^{cg} \cdot (1 - \sin(\alpha))$ denotes the part that an individual through learning obtaining from the current global optimal individual.

When the value of pop_{ik}^* out of the range $\left[POP_{\min}, POP_{\max}\right]$, we modify the value of the dimension variable by formula (12):

$$pop_{ik}^* = \begin{cases} POP_{\max} & if \quad pop_{ik} > POP_{\max} \\ POP_{\min} & if \quad pop_{ik} < POP_{\min} \end{cases} \tag{12}$$

Learning: Imitation learning is the process of repeating others' behavior consciously or unconsciously in the human society, then realize the constant evolution of individuals. Based on this principle, this work designs the imitation learning operator based on the optimal individual and two different normal individuals who are random selected form the population. Moreover, the learning process is also characterized by volatility and instability, therefore, dynamic learning operation is introduced to reduce the interference of individual learning from others in imitation learning process, so as to improve the convergence speed. The imitation learning operation is defined as (13).

$$pop_{ik}^{*} = pop_{ik} + F \cdot (pop_{k}^{cg} + (pop_{r_1 k} - pop_{r_2 k})) \tag{13}$$

Where pop_{ik}^{*} is the k-th dimension of new produced individual after imitation learning; $F \in [0,1]$ represents scaling factor; r_1 and r_2 are random number between $[1,n]$ and $r_1 \neq r_2$; $pop_{r_1 k}$ is the k-th dimension of random individual POP_{r_1}; $pop_{r_2 k}$ is the k-th dimension of random individual POP_{r_2}; pop_{k}^{cg} is the k-th dimension of the current best individual pop_{k}^{cg}; $(pop_{r_1 k} - pop_{r_2 k})$ indicates the dynamic learning step which is automatic change with the increase of the number of iterations.

If the new individual's k-th dimension out of the range $[POP_{\min}, POP_{\max}]$, we should reset the value of the variable according to the formula (12). After carrying out the imitation learning operation, the greedy operation is performed, and the individuals with better fitness are retained.

Operation in the Belief Space: There are two operators in the belief space, they are Update operator and Influence operator. Update operator is used to update the knowledge in the belief space; the Influence operator is used to guide the evolution of the micro space with the knowledge in the belief space when the algorithm iterating every η times. These two operators are defined as follows.

Update (): Replace the poor individuals in the belief space with the new better individuals who were selected from the learning space.

Influence (): Use individuals in the belief space to substitute the poor individuals in the micro-space and randomly generate some individuals to replace some original individuals.

5.2 Applying the styles to an existing paper

Based on the computing process of SLO and the designed operators, the process of task scheduling based on improved SLO is described as Alg_2.

Alg_2 Task scheduling based on SLO

Input: population size ***popsize***; mutation probability ***PM***,crossover probability ***PC***; ***F***; ***h***; ***ω***;

Output: the optimal result;

```
Step1: Initialization
        Generate individuals in the micro-space;
Step2: Discretization with SPV
        For(i=1 to popsize)
            Convert the continuous individual $POP_i$ to the discrete individual $\pi_i$ according to the SPV rule; Then, transform the $\pi_i$ to processor's vector $R_i$ according to formula (9). Last, calculating each individual's fitness value by using formula (7);}
        EndFor
REPEAT
Step3: Individual Genetic Evolution Phase
        Carry out the roulette selection operation;
        For(i=1 to popsize) // popsize is the scale of the individuals
            Carry out the crossover operation and keep the better individual;
        EndFor
        For(i=1 to popsize)
            Carry out the mutation operation and keep the better individual;
        EndFor
            Repet step2,carry out the best fitness by using formula(7);
Step4: Individual Learning Phase
        For(i=1 to popsize)
            Carry out the observational learning according to formula (10) and keep the better individual;
        EndFor
        For(i=1 to popsize)
            Carry out the imitation learning according to formula (12) and keep the better individual;
        EndFor
Step5: Culture Influence Phase
        If(the generation time == k1)
            Carry out the Influence() function;
        EndIf
        If(the generation time == k2)
            Generate q new individuals and replace q old individuals randomly;
        EndIf
Memorize the best solution achieved so far;
UNTIL (the termination condition is satisfied);
```

6 Experiment performance analyses

In order to verify the efficiency of the approach proposed in this work, CloudSim platform is adopted to simulate the task scheduling experiment. The experimental environment is PC with the following configurations, OS: Windows 7; CPU: Inter 4, 3.20GHZ; Memory:4G. The simulation experiment is conducted by extending the DataCenterBroker class in the CloudSim platform. This work take the particle swarm optimization algorithm(PSO)[23] and genetic algorithm(GA)[28] as the compared object and analyze the performance of improved SLO on the four factors involved in the fitness function, such as task completion time, cost, energy consumption and resource utilization.

6.1 Values of parameters

Under the same experimental conditions, the parameters of three algorithms are set as follows: population size P=10, task length SL, virtual machine execution capability VC, energy consumption matrix E, and cost matrix RC unified randomly generated by MATLAB. The random range of SL is $\left[10000MI, 50000MI\right]$, the range of VC value is $\left[200,1000\right]$, the value range of E is $\left[1,10\right]$, the range of value of RC is $\left[1,5\right]$. And the weight of each factor in the objective function ω_1=0.5 , ω_2=0.2 , ω_3= 0.3; Crossover rate PC=0.6 and mutation rate PM=0.08 in SLO and GA; C1=C2 =2 in PSO.

6.2 Experiment results and analyses

To verify the effectiveness of improved SLO algorithm in solving cloud task scheduling, we adopted two contrastive experiment. Each algorithm runs 10 times to eliminate the interference of random factors to the experimental results and the average value is taken as the comparison.

(1)Experiment 1: Compare the performance of three algorithm on a smaller scale

In this experiment, set 20 tasks and 5 virtual computing resources in the clouding computing environment. Then according to the objective function formula (7), calculate the task completion time, cost, energy consumption and the fitness of the optimal solution of the three algorithms. The experimental results are shown in Fig.3—Fig.6, the x-axis represents the iteration number, and the vertical axis represents the performance comparison

Fig. 3 shows the results of task completion time. From Fig. 3 it can be found that, the task completion time of the improved SLO is less than the other two algorithms when the three algorithms are iterated for 60 times. This result proves that improved SLO have quickly convergence speed than GA and PSO in solving the task scheduling problem. Fig.4 illustrates the comparison results of cost. Compared with other two algorithms, the improved SLO can reduce the cost in solving task scheduling problem. So, based on the results showed in Fig. 3 and Fig. 4, we can get that, improved SLO not only optimizes the execution time, but also optimizes the execution cost of task scheduling.

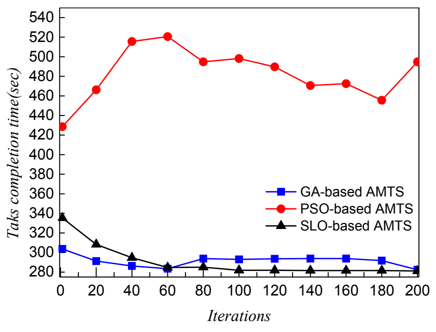


Fig.3 The task completion time for 20 tasks, 5 resource

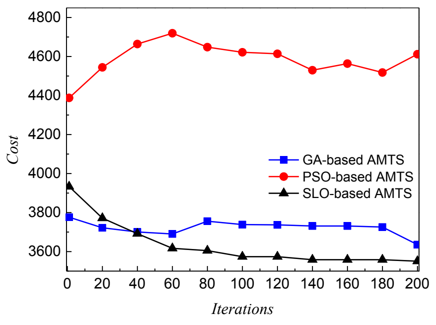


Fig.4 The cost for 20 tasks, 5 resource

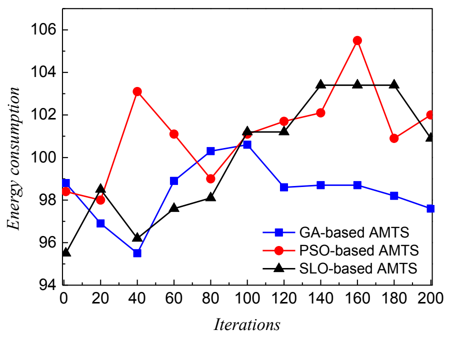


Fig.5 The energy consumption for 20 tasks, 5 resource

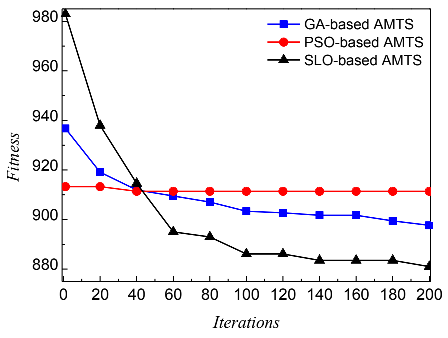


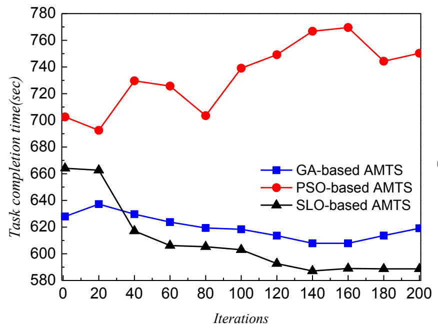
Fig.6 The fitness for 20 tasks, 5 resource

Fig. 5 shows the energy consumption of three algorithms. As the fitness function consists of three parts with different weights, includes task completion time, cost and energy consumption and the proportion of energy consumption is smaller when algorithm tends to the optimal fitness leads to the fluctuation of energy consumption.

In this experiment, fitness value represents the evaluation value of resource scheduling scheme of each algorithm. Experimental results in Fig. 6 proves that SLO has a faster convergence rate, and can obtain better solutions for task scheduling problem.

(2)Experiment 2: Compare the performance of three algorithm on a large scale

Different from experiment 1 , this experiment sets 150 tasks and 20 virtual machines in the CloudSim platform. Experiment 2 adopts the same objective function as experiment 1. Experimental results as shown in Fig. 7—Fig.10, where the x-axis represents the iteration number, the vertical axis represents the performance comparison.



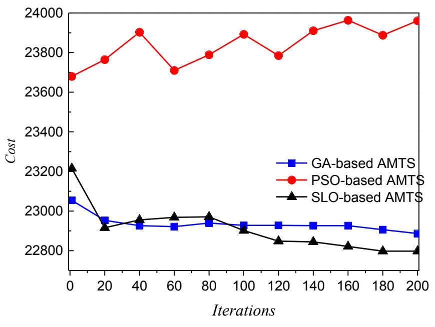
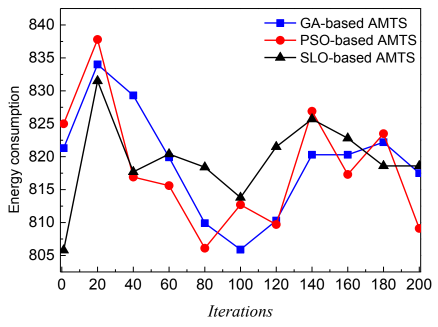


Fig.7 The task completion time for 150 tasks, 20 resource **Fig.8** The cost for 150 tasks, 20 resource



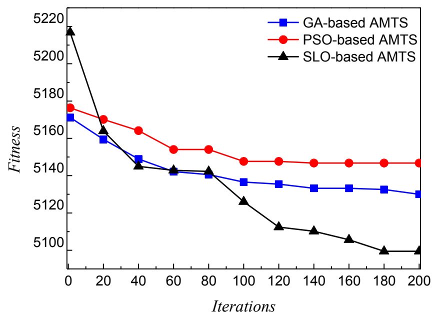


Fig.9 The energy consumption for 150 tasks, 20 resource **Fig.10** The fitness for 150 tasks, 20 resource

Fig.7 shows the comparison results of task completion time of the three algorithms. Compared with other algorithms, Fig.7 illustrates SLO can get the better solution with shorter task completion time. From the comparison results in Fig. 8, we can find that when the task size is larger, compared with other algorithms, the improved SLO is still able to get better implementation cost.

In addition, although the cost curve of SLO has some fluctuated with the increase of iteration times, SLO is converge to the less cost gradually. Based on the experimental results in Fig. 7 and Fig. 8, it can find that the improved SLO has the better performance in balancing the execution cost and task completion time.

Fig. 9 illustrates the comparison results of energy consumption and proves the same conclusion as experiment 1. Fig.10 represents the fitness value of task scheduling scheme searched by three algorithms. From Fig. 10, it can be found that, SLO has better optimizations performance and get the better results in task scheduling problem when compared with GA and PSO.

Furthermore, in order to verify the solving efficiency of the proposed method, we separately calculate the fitness values of the three algorithms in the same experiment conditions but different run-time conditions. The experimental results are shown in Table 2 and Table 3.

Table 2. Research results comparison under the smaller task scale

Algorithm			
Runtimes	***GA fitness***	***PSO fitness***	***SLO fitness***
10ms	910.3903	919.7856	918.7804
20ms	887.3571	899.8473	899.5892
30ms	883.1139	888.4389	890.5927
40ms	879.7424	881.0896	886.1017
50ms	875.5693	877.7031	877.7914
60ms	874.5732	877.2812	874.4622
70ms	871.984	877.2812	867.798
120ms	865.2132	877.2812	862.7844

Table 3. Research results comparison under the larger task scale

Algorithm			
Runtimes	***GA fitness***	***PSO fitness***	***SLO fitness***
50ms	5175.799	5161.497	5192.286
100ms	5137.816	5139.04	5180.643
0.5s	5112.762	5109.193	5126.121
1s	5110.142	5101.482	5123.168
1.5s	5101.689	5093.884	5118.513
2s	5098.243	5093.884	5094.979
3s	5089.839	5093.884	5066.565

From table 2 we can find that the difference of search results is smaller between three algorithms when the running time is less than 50ms. With the increase of running time, PSO gets into a local optimum. However the genetic algorithm and the improved SLO algorithm are well convergent until the best global optimum solution is found, and the convergence rate of SLO is the fastest, at the same time, the optimal solution found by the improved SLO is better than the optimal solutions found by the other two algorithms.

With the increase of tasks and virtual resources, table 3 shows the fitness value obtained by the three algorithms were slowly decreased when the running time is less than 1s. PSO get into local optimum after 1.5s, but SLO can skip the local optimum, finally gets the optimal scheduling scheme. It means that the computational efficiency of SLO is higher than PSO and GA. Experimental results in table 2 and 3 prove that the improved SLO has better convergence speed and search ability in solving the task scheduling problem, meanwhile it also can balance the task completion time and the cost well .

In the end, on base of analysis experiment result, scheduling performance of the improved SLO is better than GA-based AMTS algorithm and PSO-based AMTS algorithm in cloud task scheduling problems. Improved SLO not only effectively reduce the cloud task completion time and the cost of the data center, but also provide users with a better user experience. With the increase of tasks and virtual resources, the proposed method still has good convergence ability and optimization ability. At the same time, the experimental results also verify that the improved SLO algorithm proposed in this paper has strong ability to skip the local optimum and faster convergence speed.

7 Conclusions

In order to efficiently solve the task scheduling problem in cloud computing, this paper proposes a novel task scheduling algorithm based on the improved SLO. We designs some optimization operators (such as crossover operation, mutation operation, observational learning, imitation learning operation, accept operation and influence operation) for different spaces of SLO, moreover, Small Position Value (SPV) method is used to discrete the individuals of improved SLO, so as to enable the improved SLO could solve the discrete task scheduling problem. Finally, the performance of improved SLO is compared with other methods on the CloudSim platform. Experimental results show that improved SLO has better performance than that of other evolutionary algorithms. Moreover, the improved SLO algorithm can also be used to solve other optimization problems.

8 Acknowledgment

This work is supported by: The National Natural Science Foundation Fund China (No.61300124; No.61472106; No.61403128); Major projects of science and technology in Shandong Province (2015ZDXX0201B02); Basis and frontier of Henan Province (152300410212); Soft science research project of Henan Province (152400410513); Zhengzhou leading talent project (131PLJRC645); The authors would like to thank the anonymous reviewers for their valuable feedback on this work.

9 References

[1] Jula A, Sundararajan E, Othman Z. Cloud computing service composition: A systematic literature review [J]. Expert Systems with Applications, 2014, 41(8):3809-3824. https://doi.org/10.1016/j.eswa.2013.12.017

[2] Li X, Xu J, Yang Y. A Chaotic Particle Swarm Optimization-Based Heuristic for Market-Oriented Task-Level Scheduling in Cloud Workflow Systems [J]. Computational Intelligence & Neuroscience, 2015, 2015:718689.

[3] Zhan Z H, Liu X F, Gong Y J, et al. Cloud Computing Resource Scheduling and a Survey of Its Evolutionary Approaches [J]. Acm Computing Surveys, 2015, 47(4):1-33. https://doi.org/10.1145/2788397

[4] Jafari Navimipour N, Sharifi Milani F. Task Scheduling in the Cloud Computing Based on the Cuckoo Search Algorithm [J]. International Journal of Modelling and Optimization, 2015,5(1):44-47. https://doi.org/10.7763/IJMO.2015.V5.434

[5] Ullman J D. NP -complete scheduling problems[J]. Journal of Computer & System Sciences, 1975, 10(3):384-393. https://doi.org/10.1016/S0022-0000(75)80008-0
[6] Liu C Y, Zou C M, Wu P. A Task Scheduling Algorithm Based on Genetic Algorithm and Ant Colony Optimization in Cloud Computing[C]// International Symposium on Distributed Computing and Applications To Business, Engineering and Science. 2014:68-72.
[7] Tao F, Feng Y, Zhang L, et al. CLPS-GA: A case library and Pareto solution-based hybrid genetic algorithm for energy-aware cloud service scheduling [J]. Applied Soft Computing, 2014, 19(6):264–279. https://doi.org/10.1016/j.asoc.2014.01.036
[8] Zhang M, Yang Y, Mi Z, et al. An Improved Genetic-Based Approach to Task Scheduling in Inter-cloud Environment[C]//Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom), 2015 IEEE 12th Intl Conf on. IEEE, 2015: 997-1003.
[9] Keshanchi B, Souri A, Navimipour N J. An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: Formal verification, simulation, and statistical testing [J]. Journal of Systems and Software, 2017, 124: 1-21. https://doi.org/10.1016/j.jss.2016.07.006
[10] Abdi S A M S. Task Scheduling using Modified PSO Algorithm in Cloud Computing Environment [M]. 2014.
[11] Tareghian S, Bornaee Z. Algorithm to improve job scheduling problem in cloud computing environment[C]//2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI). IEEE, 2015: 684-688.
[12] Jena R K. Multi Objective Task Scheduling in Cloud Environment Using Nested PSO Framework [J]. Procedia Computer Science, 2015, 57:1219-1227. https://doi.org/10.1016/j.procs.2015.07.419
[13] Zhang Z, Zhang N, Feng Z. Multi-satellite control resource scheduling based on ant colony optimization [J]. Expert Systems with Applications, 2014, 41(6):2816-2823. https://doi.org/10.1016/j.eswa.2013.10.014
[14] Hua X Y, Zheng J, Wen-Xin H U. Ant colony optimization algorithm for computing resource allocation based on cloud computing environment [J]. Journal of East China Normal University, 2010, 61(1):127-134.
[15] Kansal N J, Chana I. Artificial bee colony based energy-aware resource utilization technique for cloud computing [J]. Concurrency and Computation: Practice and Experience, 2015, 27(5):1207-1225. https://doi.org/10.1002/cpe.3295
[16] Zhao L. Cloud computing resource scheduling based on modified cuckoo search algorithm [J]. Journal of Nanjing University of Science and Technology, 2016(04):472-476.
[17] Liu Z, Chu D, Song C, et al. Social learning optimization (SLO) algorithm paradigm and its application in QoS-aware cloud service composition [J]. Information Sciences, 2016, 326:315-333. https://doi.org/10.1016/j.ins.2015.08.004
[18] Bo XU, Chao Z, Zhu Y J, et al. Virtual Machine Resource Scheduling Multi-objective Optimization in Cloud Computing [J]. Journal of System Simulation, 2014, 45(2):493-495.
[19] S. Kim, J. Byeon, H. Yu and H. Liu, "Biogeography-Based Optimization for Optimal Job Scheduling in Cloud Computing", Applied Mathematics and Computation, Elsevier, Vol. 247, pp. 266-280, 2014. https://doi.org/10.1016/j.amc.2014.09.008
[20] Bansal N, Maurya A, Kumar T, et al. Cost performance of QoS Driven task scheduling in cloud computing [J]. Procedia Computer Science, 2015, 57:126-130. https://doi.org/10.1016/j.procs.2015.07.384

[21] Jacob L, Jeyakrishanan V, Sengottuvelan P. Resource Scheduling in Cloud using Bacterial Foraging Optimization Algorithm[J]. International Journal of Computer Applications, 2014, 92(1):281-289. https://doi.org/10.5120/15972-4857
[22] Sun D W, Chang G R, Feng-Yun L I, et al. Optimizing Multi-Dimensional QoS Cloud Resource Scheduling by Immune Clonal with Preference [J]. Tien Tzu Hsueh Pao/acta Electronica Sinica, 2011, 39(8):1824-1831.
[23] He H, Xu G, Pang S, et al. AMTS: Adaptive multi-objective task scheduling strategy in cloud computing [J]. Wireless Communication Over Zigbee for Automotive Inclination Measurement China Communications, 2016, 13(4):162-171. https://doi.org/10.1109/cc.2016.7464133
[24] Liu W, Jin H, Liu B. Cloud computing resource scheduling based on improved quantum genetic algorithm [J]. Journal of Computer Applications, 2013, 33(8):2151-2153. https://doi.org/10.3724/SP.J.1087.2013.02151
[25] Peng B. Knowledge and population swarms in cultural algorithms for dynamic environments [M]. Wayne State University, 2005.
[26] Yan-Fang L I, Jiang X F. Grid Resources Scheduling Algorithm Based on Discrete Particle Swarm and Tabu Search [J]. Computer & Modernization, 2011.
[27] Fatih Tasgetiren M, Liang Y, Sevkli M, et al. Particle swarm optimization and differential evolution for the single machine total weighted tardiness problem [J]. 2006, 44(22):4737-4754.
[28] Srinivas M A P L. Genetic Algorithms: A Survey [J]. Computer, 1994, 6(27):17-26. https://doi.org/10.1109/2.294849

10 Authors

Zhizhong Liu was born in Shangshui county, Zhoukou City, in 1981. received the PhD degree in computer science from the HoHai University, Nan jing, China, in 2011. He is currently an lecturer in the Henan Polytechnic University and Post doctoral in Harbin Institute Of Technology. His main research interests are in Swarm intelligence algorithm, Web service computing and Cloud Computing.

Jingxuan Qin was born in Linfen city, Shanxi province，in 1991. Currently, she is a graduate student with the School of Computer science and technology in the Henan Polytechnic University. Her current main research interests include Swarm intelligence algorithm and Cloud Computing.

Weiping Peng was born in TianMen City, Hubei Province in 1979. He received the Ph.D. degree in Signal and Information Processing from the Beijing University of Posts and Telecommunications, Beijing, China, in 2011. He is currently an associate professor at the School of Computer Science and Technology in Henan Polytechnic University. His main research interests include information security, data leakage prevention and security and application of IOT.

Chao Hao was born in XuChang City, Henan Province in 1981. He is currently an lecturer in the Henan Polytechnic University, Jiao zuo, 45400, P.R. China. His main research interests are in information security.

Article submitted 24 January 2017. Published as resubmitted by the authors 14 Mach 2017.