International Journal of Gaming and Computer-Mediated Simulations
Volume 13 • Issue 2 • April-June 2021

A Novel Task-Scheduling Algorithm of Cloud Computing Based on Particle Swarm Optimization

Zhou Wu, Guangdong Songshan Polytechnic College, China
Jun Xiong, Guangdong Songshan Polytechnic College, China

ABSTRACT

With the characteristics of low cost, high availability, and scalability, cloud computing has become a high demand platform in the field of information technology. Due to the dynamic and diversity of cloud computing system, the task and resource scheduling has become a challenging issue. This paper proposes a novel task scheduling algorithm of cloud computing based on particle swarm optimization. Firstly, the resource scheduling problem in cloud computing system is modeled, and the objective function of the task execution time is formulated. Then, the modified particle swarm optimization algorithm is introduced to schedule applications' tasks and enhance load balancing. It uses Copula function to explore the relation of the random parameters random numbers and defines the local attractor to avoid the fitness function to be trapped into local optimum. The simulation results show that the proposed resource scheduling and allocation model can effectively improve the resource utilization of cloud computing and greatly reduce the completion time of tasks.

KEYWORDS

Cloud Computing, Load Balancing, Particle Swarm Optimization, Task Scheduling

1. INTRODUCTION

Based on the technologies of distributed computing, parallel computing and virtualization, cloud computing can effectively integrate server and computer resources in the form of a unified resource pool (Hossain, 2013). The combination of huge of computing nodes can facilitate customers to obtain resources on demand only

DOI: 10.4018/IJGCMS.2021040101

through the browser (Balalaie et al., 2016). With the advantages of low cost, flexible configuration and high efficiency of resource utilization, cloud computing is rising at a very fast speed recently. It should be noted that network computing resources are usually huge and scattered. In order to dynamically allocate resources to each task according to customer's request, the implement of reasonable resource scheduling should be fundamental (Mohammed et al., 2018). Hence, task scheduling strategy has a direct impact on customer's task execution efficiency, system resource utilization efficiency, task execution cost, load balancing, system stability, etc. Especially while allocating and scheduling resources for large-scale tasks, it is necessary not only to minimize the completion time and improve the system utilization, but also to take into account of the resource load balancing and quality of service. Therefore, the resource scheduling problem of cloud computing is a non deterministic polynomial (NP) problem.

Since the cloud computing resources are assigned in the way of use on demand, the scheduling results directly affect the quality of task completion. Therefore, the design of reasonable and efficient cloud computing resource scheduling and allocation model is very important for improving resource utilization and achieving favorable user experience (Schulte et al., 2013). Many researchers have put forward appropriate resource scheduling schemes and depict plenty of literature analysis of utilization of resources with minimum cost and execution. In static resource scheduling strategies, the resource scheduling methods include: mathematical programming, expert system, neural network and fuzzy logic strategy (Shweta et al., 2018). The advantages of this kind of methods are strict rules and clear logic, but the rules are more complex, only suitable for small-scale resource static scheduling. Dynamic resource scheduling methods can be divided into: greedy-based strategies (such as Min-Min algorithm, Max-min algorithm, Sufferage algorithm) and job scheduling strategies (such as FIFO algorithm, fair scheduling algorithm, computing power scheduling algorithm). Those strategies can achieve dynamic resource scheduling. However, due to too much factors involved in the operation, the computational complexity for large-scale task scheduling will be excessive.

Compared with traditional methods, heuristic task scheduling strategy can achieve better adaptability and flexibility in solving NP problems. In this paper, we propose a novel task scheduling algorithm of cloud computing based on particle swarm optimization (NTSA-PSO) to obtain the optimal resource scheduling strategy.

The rest of this paper is organized as follows: Section 2 discusses the research status of related work, and summarizes their advantages and disadvantages. Section 3 focuses on the particle swarm optimization algorithm, and designs an optimal task scheduling model. At the same time, the implementation process of our proposed NTSA-PSO algorithm is described in detail. Section 4 is the simulation experiment test and experimental results analysis. Finally, Section 5 summarizes the advantages of the proposed algorithm.

2. RELATED WORK

In recent years, many scholars have carried out in-depth research on cloud computing resource scheduling problems, and proposed a variety of resource scheduling algorithms. Static scheduling algorithm focuses on global planning according to all task requirements and resource pool size to maximize resource utilization.

To obtain the load balancing of the cloud computing system, Jha et al. (2011) designed a low-cost and low load resource allocation strategy under certain conditions. Lin et al. (2014) proposed a delay-aware virtual machine placement method in cloud environment for the virtual resource allocation of data intensive applications, which can effectively reduce the total data transmission delay. Garcia et al. (2014) proposed a service level agreement (SLA) -aware autonomous cloud solutions, which takes into account the customer's satisfaction and achieves the maximization of the provider's revenue. This method can reasonably assign all the resources of the cloud server, and optimizes the allocation of resources to the greatest extent. Combined with the customer's demand, Vilaplana et al. (2014) introduced the queuing theory model into the cloud computing environment and proposed a job scheduling algorithm with feedback mechanism. The algorithm has high scheduling efficiency and load balancing ability. Grandl et al. (2016) proposed that resource fragments should be given priority to fast completed jobs, Improve resource efficiency and performance.

Based on the analysis of the dynamic, distributed and heterogeneous characteristics of resources and the real-time requirements of job scheduling, dynamic resource scheduling algorithms are introduced to deal with the lack of flexibility in resource allocation of cloud computing. John et al. (2014) formulated the scheduling process as a Directed Acyclic Graph with inter-task dependencies and proposed an improved workflow scheduling for grid services. Aiming to maximize the cloud provider's revenue and meet the customer's demand, Karthikeyan et al. (2017) proposed an inspired virtual machine instances allocation algorithm based on dynamic programming. To achieve the load balancing of virtual machines, Kruekaew et al. (2014) proposed an artificial bee colony optimization algorithm for cloud resource scheduling. By taking into account of CPU's utilization, throughput and I/O rate, Chen et al. (2012) designed a pre-migration strategy with hybrid genetic algorithm to resolve the fragmentation problem of physical resources.

Through the above analysis, it is not difficult to see that most of the previous scheduling work is focused on the mapping of job resources, while the evolutionary algorithm is rarely used to allocate tasks among resources. Therefore, there is a need to execute resource scheduling in an efficient way by taking care of the reduction of time execution as well as the massive size of jobs and machines in cloud computing system(Konjaang et al., 2016).

3. PROPOSED METHOD

3.1 Problem Formulation

In essence, resource scheduling in cloud computing can be regarded as a multi-objective optimization problem(Xiao et al., 2019). Due to the different goals of application users and resource providers, the former mainly focus the execution time of workflow and resource leasing cost while the latter pays more attention to resource utilization. The phenomenon of resource competition occurs frequently(Kumar et al., 2018). Therefore, the reasonable and efficient resource scheduling algorithm should be employed to allocate resources and tasks, and try to avoid disordered competition of multiple tasks for a resource.

To facilitate the analysis of the problem, the following mathematical model is discussed. Suppose that the set of jobs $J(j_1,j_1,\ldots,j_k)$ will be scheduled on the cloud computing platform, the set of tasks contained in j-th job is $D_j\left(d_{j1},d_{j2},\cdots,d_{ji}\right)$, and the virtual machines (VMs) are denoted by vector $V_n(v,v_1,\ldots,v_n$. The value of π_{ik}^{n} is 1 or 0, indicating whether the k-th task in the i-th job is assigned to run on the resource n.

According to the resource management in the cloud computing model, the task list mainly considers the final time of task execution and the load balance of virtual machine, so as to maximize the effect of resource utilization. Therefore, the function $CT(x)$ is defined to express the completion time of all jobs as follows:

$$\begin{cases} CT\left(x\right)=\dfrac{1}{2}\left[f\left(x\right)+g\left(x\right)\right] \\ f\left(x\right)=\sum\limits_{v=1}^{n}\sum\limits_{i=1}^{m}t\left(i,j,v\right)\pi_{ij}^{v} \\ g\left(x\right)=\dfrac{1}{m}\sum\limits_{j=1}^{k}T_j \end{cases} \tag{1}$$

where $f(x)$ denotes the completion time of the j-th job, and $g(x)$ is the average execution time expected by the user. Besides, $t(i,j,v)$ is the execution time of i-th task in j-th job and T_j represents the total execution time of j-th job.

Moreover, the load balance of virtual resources is also important in scheduling, and will be regarded as a key indicator to estimate the resource allocation decision making well or not. Then, the load balance degree of cloud computing center can be defined as:

$$LBD\left(x\right)=\frac{1}{n}\sum_{v=1}^{n}\left(t\left(i,j,v\right)\pi_{ij}^{v}-\frac{1}{n}\sum_{v=1}^{n}\left(t\left(i,j,v\right)\pi_{ij}^{v}\right)\right)^{2} \tag{2}$$

During the process of resource scheduling process, the load balance of virtual resource scheduling means that the difference between the task execution time of each virtual machine and the average task execution time of the cloud computing center to be minimum. Let S represent a complete cloud computing resource scheduling scheme, the utility function of the virtual machine resources allocation can be expressed as follows:

$$\begin{aligned} & Utility\left(S\right)=\min\left\{\omega CT\left(x\right)+\left(1-\omega\right)LBD\left(x\right)\right\}, \\ s.t.\ \ & \pi_{ij}^{v}\le\sigma_{ij}^{v}, \\ & \pi_{ij}^{v}\ge 0,\sigma_{ij}^{v}\ge 0,t\left(i,j,v\right)\ge 0, \\ & 1-P_{fa}\ge P_{suc} \end{aligned} \tag{3}$$

where ω is the proportion of the above indicators. σ_{ij}^{v} indicates whether the virtual machine V can execute the i-th task. Besides, P_{fa} and P_{suc} indicates the probability of resource computing failure and the probability of successful execution required by the customer, respectively.

The objective of virtual resource scheduling should minimize the task completion time as well as obtain balanced virtual machine load under the condition of ensuring the customer's demand.

3.2 Particle Swarm Optimization

The resource scheduling problem of cloud computing is actually a NP problem. Evolutionary algorithms are widely used to solve optimization problems by simulating the laws of natural genetic evolution, and they have outstanding advantages in the search of optimal solutions. Owing to the simplicity and few parameters, particle swarm optimization (PSO) simulates the predatory behavior of birds for swarm based intelligent optimization, which can be applied to task scheduling, machine learning, data clustering, process planning and other fields(Kennedy et al., 2001). It provides an effective means for solving cloud computing resource scheduling problems. However, PSO has the defects of premature convergence and easy to fall into local optimization, and often needs to optimize the parameters according to the characteristics of the problem(Ratnaweera et al., 2010).

Based on the previous analysis, the mathematical model of cloud computing resource scheduling problem is established, and the improved PSO algorithm is used to solve the problem to obtain the optimal scheduling scheme of cloud computing resources. Therefore, the fitness function can be defined as:

$$fitness(i)=1/Utility\left(S\right) \tag{4}$$

By applying the PSO to cloud computing task scheduling, it is necessary to encode the particles reasonably and make the position and speed of particles accord with the task scheduling. Since that the tasks in cloud computing are independent, the particle coding can be processed by discrete values. For *n* resource, the particle can be encoded as a dimension vector $P= \{p_1,p_2,\ldots,p_m\}$ and 1£p_i£*n*, and the number of tasks is the encoding length of the particle's position. Each one-dimensional coordinate of the particle represents a subtask number, one-dimensional component p_i represents the resource number assigned to the task, and p_{ij} indicates that the *j*-th task is assigned to the corresponding resource. Moreover, $P_i = (p_{i1}, p_{i2}, p_{i3}, \cdots, p_{iD})^T$ represents the position vector of the particle, and $V_i = (v_{i1}, v_{i2}, v_{i3}, \cdots, v_{iD})^T$ is the velocity vector.

After the fitness value of particle x is obtained, the current position of the particle should be compared to determine whether it is optimal or not. The optimal position reached by particle *i* will be denoted as $pbest_i$, and the best position found by the whole group will be denoted as *gbest*. In traditional PSO algorithm, the updated speed and position during each iteration process are generally expressed by:

$$\begin{cases} v_i(t) = w(t)v_i(t) + c_1r_1(pbest_i - p_i(t-1)) + c_2r_2(gbest - p_i(t-1)) \\ p_i(t) = p_i(t-1) + v_i(t) \end{cases} \tag{5}$$

where *t* is the number of iterations, and *w*(*t*) is the inertia weight at current iteration. r_1 and r_2 are the random number uniformly distributed between 0 and 1 to maintain the diversity of the population. Besides, c_1 and c_2 are the acceleration factors, which indicates the ability of particles to learn from themselves or the population.

PSO can move towards the optimal solution according to the search experience of individual particles and the whole swarm (Bittencourt et al., 2011). It is an important way to find the optimal solution accurately and effectively make rational use of the individual experience information of particles and the global solution(Singh et al., 2016). The hypothesis of independence makes groups adapt to history randomly in their own cognition, and make full use of individual experience and information shared by groups. Since the parameters r_1 and r_2 are applied to provide random search, the value of them are usually set as two independent random numbers regardless of the cognitive relationship between individual knowledge and global knowledge. Actually, it is necessary to deeply explore the relation of the parameters r_1 and r_2. The common method is to exploit the Copula function. Due to the requirement of numerical method, the use of Copula function is not conducive to the generation of random numbers. Therefore, the Gaussian Copula function is adopted to determine the correlation and use linear transformation to generate the random numbers. First, the random variables α and β will be given by:

$$\begin{cases}\alpha = \varphi^{-1}\left(u_1\right)\\ \beta = \varphi^{-1}\left(u_2\right)\end{cases} \tag{6}$$

where $\varphi\left(\bullet\right)$ is the distribution function of the standard normal distribution.

Given the correlation coefficient ρ, the correlated random variables with uniformly distributed will be obtained. By decomposing those random numbers, the two correlated random numbers that obey normal distribution can be obtained by:

$$\begin{cases}x = \alpha\\ y = \rho\alpha + \beta\sqrt{1-\rho^2}\end{cases} \tag{7}$$

where:

$$\rho = \frac{Cov\left(x,y\right)}{\sqrt{Var\left(x\right)Var\left(y\right)}} = \frac{E\left(x - E\left(x\right)\right)E\left(y - E\left(y\right)\right)}{\sqrt{Var\left(x\right)Var\left(y\right)}}$$

Furthermore, $r_1 = \varphi(x)$ and r2$_=\varphi$(y) will be calculated, and are random numbers with Gaussian copula function.

During the search process, the inertia weight can affect the global optimal solution and convergence speed. For PSO algorithm, the value of inertia weight is negatively related to the convergence speed. Generally, the increase of the inertia weight can shift toward more exploration and will be easy to find new solutions. On the contrary, the decrease of the inertia weight will improve the exploitation and enhance the speed of convergence. Therefore, the setting of inertia weight value needs to be changed adaptively according to the characteristics of the optimization problem. To overcome the prematurity and weak exploration capability, the global search ability at the beginning of the iteration process and the local search ability near the end of the iteration process should be enhanced. Thus, the inertia weight should achieve a linearly decreasing and can be expressed by:

$$w(t) = w_{\max} - \frac{w_{\max} - w_{\min}}{t_{\max} - t} \tag{8}$$

where $w_{\max}$ and $w_{\min}$ are the initial value and final value of the inertia weighting, respectively. $t_{\max}$ is the maximum number of the iterations.

Furthermore, to avoid falling into local optimum in the later convergence process, it is beneficial to conduct global search and lead the population to approach the average value of its own optimal particles. With the iteration, the solution should pay more attention to local searching ability and avoid the fitness function to be trapped into local optimum. Therefore, the local attractor is introduced, which can be defined as:

$$LA_i(t) = \frac{t_{\max} - t}{t_{\max}} \times [\frac{1}{n}\sum_{i=1}^{n} pbest_i - gbest] \tag{9}$$

Furthermore, the update of particle position can be modified as follows:

$$\hat{p}_i(t) = LA_i(t)p_i(t-1) + (1 - LA_i(t))p_i(t) \tag{10}$$

With the increase of the number of iterations, the effect of the operator decreases gradually, which is conducive to local development near the optimal solution. In this way, the reasonable balance between global search and local development will be effectively ensured.

Finally, the proposed algorithm can be described as the following procedures:

Step 1: According to the cloud computing resource scheduling model, the particles are initialize and the Corresponding fitness value can be calculated according to Equation (4);
Step 2: Update $pbest_i$ according to the fitness function and *gbest* among each particles' best positions;
Step 3: Compute $LA_i(t)$ according to Equation (9);
Step 4: Use linear transformation to generate the random numbers r_1 and r_2, and update the inertia weight $w(t)$ according to Equation (7);
Step 5: Compute the new position according to Equation (5) and Equation (10);
Step 6: If the procedure satisfies the end condition, the algorithm can be terminated and the optimal resource scheduling scheme will be obtained. Otherwise, return to Step 2 and continue the process.

4. PERFORMANCE EVALUATION

In this section, the simulation experiments are conducted and the performance of the proposed algorithm will be compared with the different scheduling algorithms. The CloudSim toolkit is chosen as a simulation platform, and the simulation experiment is carried out by MATLAB 2014r software to test the effectiveness of the proposed algorithm for cloud computing resource scheduling problem. All the tests are simulated for 10 times to obtain the average experimental values, and the table 1 indicates the specific parameters setting.

Table 1. Parameters setting

Parameter	Value
Number of tasks	100-1000
Number of VMs	40
Maximum number of iterations t_{max}	100
Size of populations	50
Acceleration factors c_1, c_2	2, 2
Correlation coefficient ρ	0.7
Proportion of the indicators ω	0.5
Initial value and final value of the inertia weighting	0.9, 0.4

We compare the performance of the proposed algorithm to the performance of EARTH [20] and HCOC [21], and the convergence curves of total execution time with the number of iterations are plotted in Fig. 1 and Fig. 2, respectively. The experimental results in Figures 1 and 2 show that the proposed algorithm provides better execution time than other algorithms for different number of tasks. The convergence trend of the improved algorithm is faster and more stable than other algorithms, especially for relatively large number of tasks. It demonstrates that when the iterations increase, the overall fitness of the random population may be

Figure 1. Total execution time with 400 tasks

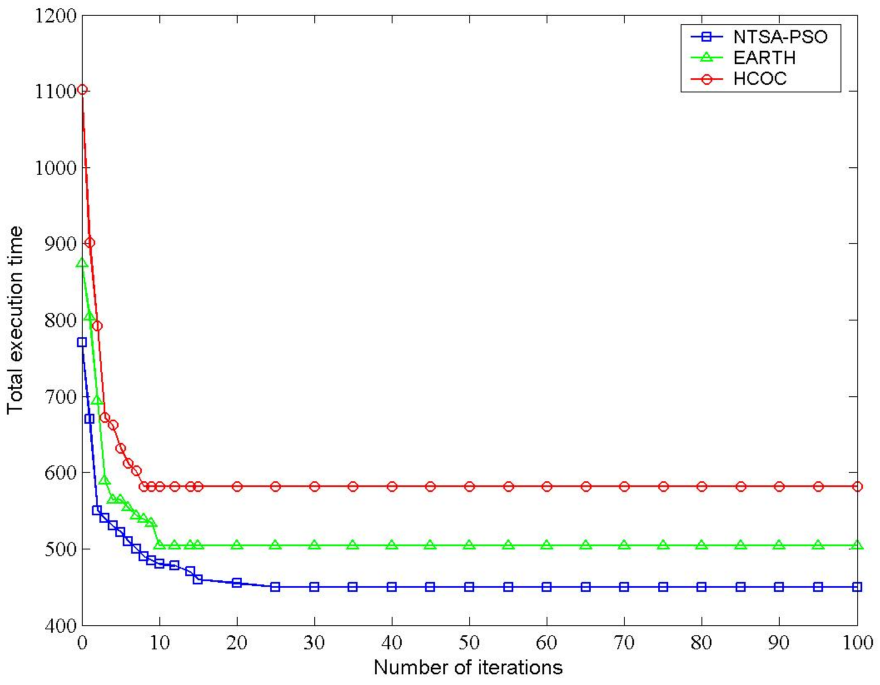
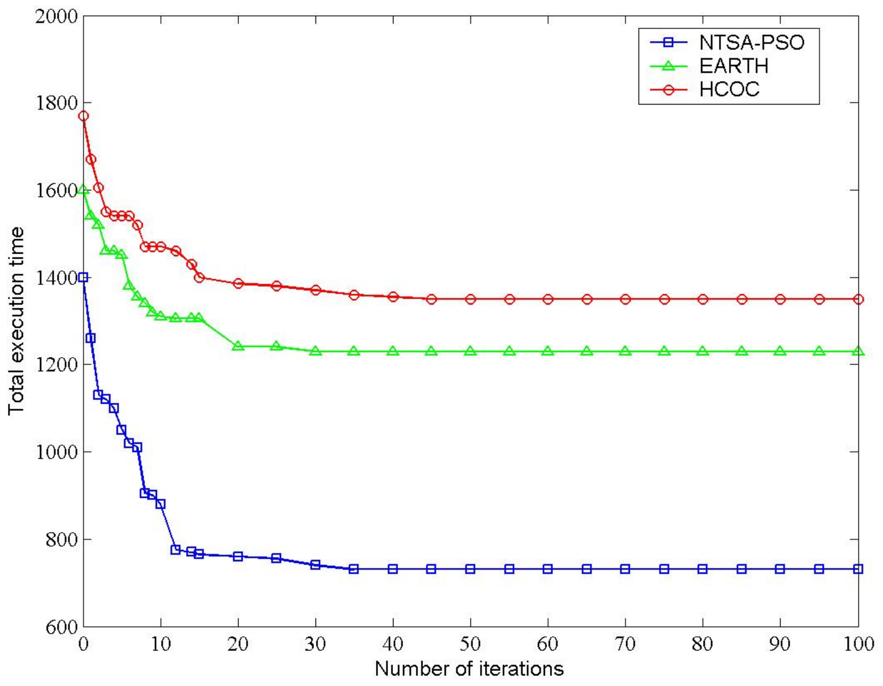


Figure 2. Total execution time with 800 tasks



improved and local search and exploitation are conducted towards the optimum solution effectively.

The total execution time with different tasks is shown in Figure 3. It can be seen from Fig. 3 that with the increasing number of tasks, the completion time of all algorithms are also increased. Comparatively, HCOC algorithm increases sharply and our proposed algorithm obtains the optimal performance results along with the number of tasks increasing. It shows that our proposed algorithm has good robustness and can adapt to large-scale task scheduling application scenarios.

The load balance curves for different algorithms are shown in Figure 4. From the results, the load balance degree of our proposed algorithm can be kept below 0.5 basically, which indicates that the improved method can effectively ensure the load balance of resource. The main reason is that all tasks will be assigned legitimately among VMs in NTSA-PSO and better system load balance can be achieved with increasing number of tasks. Therefore, NTSA-PSO can prevent the emergence of unbalanced resource allocation and improve the efficiency of task execution.

5. CONCLUSION

With the characteristics of low cost, high availability and scalability, cloud computing has become a high demand platform in the field of information technology. In this paper, a novel task scheduling algorithm of cloud computing based on particle swarm

Figure 3. Total execution time with different tasks

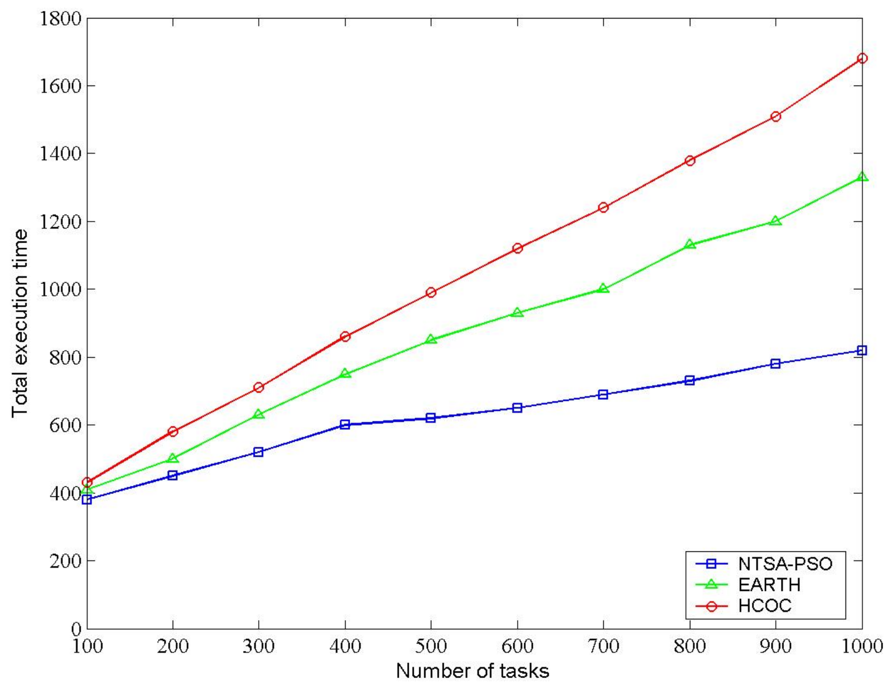
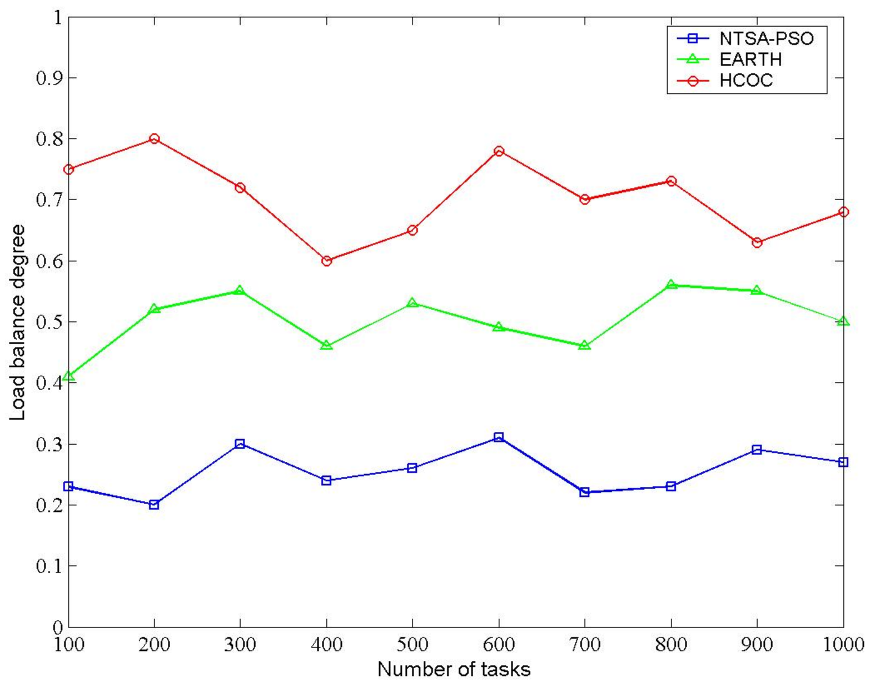


Figure 4. Load balance with different tasks



optimization is proposed. Firstly, the resource scheduling problem in cloud computing system is modeled, and the objective function of the task execution time is formulated. Then, the modified particle swarm optimization algorithm is introduced to schedule applications' tasks and enhance load balancing. It uses Copula function to explore the relation of the random parameters random numbers, and defines the local attractor to avoid the fitness function to be trapped into local optimum. The simulation results show that the proposed resource scheduling and allocation model can effectively improve the resource utilization of cloud computing and greatly reduce the completion time of tasks.

REFERENCES

Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Micro-services architecture enables Dev Ops: Migration to a cloud-native architecture. *IEEE Software*, *33*(3), 42–52. doi:10.1109/MS.2016.64

Bittencourt, L. F., & Madeira, E. R. M. (2011). HCOC: A cost optimization algorithm for workflow scheduling in hybrid clouds. *Journal of Internet Services and Applications*, *2*(3), 207–227. doi:10.1007/s13174-011-0032-0

Chen, S., Wu, J., & Lu, Z. (2012). A cloud computing resource scheduling policy based on genetic algorithm with multiple fitness. In *Proc. of IEEE International Conference on Computer & Information Technology*. IEEE. doi:10.1109/CIT.2012.56

Garcia, A. G., Espert, I. B., & Garcia, V. H. (2014). SLA-driven dynamic cloud resource management. *Future Generation Computer Systems*, *31*, 1–11. doi:10.1016/j.future.2013.10.005

Grandl, R., Chowdhury, M., Akella, A., & Ananthanarayanan, G. (2016). Altruistic scheduling in multi-resource clusters. *Proc. of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016)*, 210-216.

Hossain, S. (2013). Infrastructure as a service. *Cloud Computing Service & Deployment Models Layers & Management*, *22*(7), 26–49.

Jha, R. K., & Dalal, U. D. (2011). On demand cloud computing performance analysis with low cost for QoS application. In *Proc of International Conference on Multimedia*. IEEE.

John, S. M., & Mohamed, M A M. (2014). Deadline aware work flow scheduling for grid services with QoS. *International Journal of Applied Engineering Research: IJAER*, *9*(24), 23883–23898.

Karthikeyan, P., & Chandrasekaran, M. (2017). Dynamic programming inspired virtual machine instances allocation in cloud computing. *Journal of Computational and Theoretical Nanoscience*, *14*(1), 551–560. doi:10.1166/jctn.2017.6362

Kennedy, J., Eberhart, R. C., & Shi, Y. (2001). *Swarm intelligence*. Elsevier Science Press.

Konjaang, J. K., Maipan-Uku, J. Y., & Kubuga, K. K. (2016). An efficient max-min resource allocator and task scheduling algorithm in cloud computing environment. *International Journal of Computers and Applications*, *142*(8), 25–30. doi:10.5120/ijca2016909884

Kruekaew, B., & Kimpan, W. (2014). Virtual machine scheduling management on Cloud computing using Artificial Bee Colony. *Lecture Notes in Engineering & Computer Science*, *12*(2), 32–41.

Kumar, N., Chilamkurti, N., Zeadally, S., & Jeong, Y.-S. (2018). Achieving quality of service (QoS) using resource allocation and adaptive scheduling in cloud computing with grid support. *The Computer Journal*, *57*(2), 281–290. doi:10.1093/comjnl/bxt024

Lin, J. W., Chen, C. H., & Lin, C. Y. (2014). Integrating QoS awareness with virtualization in cloud computing systems for delay-sensitive applications. *Future Generation Computer Systems*, *37*(7), 478–487. doi:10.1016/j.future.2013.12.034

Mohammed, A., Nirmeen, E. B., & Mai, E. K. (2018). An efficient cost-based algorithm for scheduling workflow tasks in cloud computing systems. *Neural Computing & Applications*, 1–11.

Ratnaweera, A., Halgamuge, S. K., & Watson, H. C. (2010). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, *8*(3), 240–255. doi:10.1109/TEVC.2004.826071

Schulte, S., Schuller, D., & Hoenisch, P. (2013). Cost-driven optimization of cloud resource allocation for elastic processes. *International Journal of Cloud Computing*, *1*(2), 1–14.

Shweta, Varshney, & Sarvpal. (2018). A survey on resource scheduling algorithms in cloud computing. *International Journal of Applied Engineering Research*, *13*(9), 6839–6845.

Singh, S., & Chana, I. (2016). EARTH: Energy-aware autonomic resource scheduling in cloud computing. *Journal of Intelligent & Fuzzy Systems*, *30*(3), 1581–1600. doi:10.3233/IFS-151866

Vilaplana, J., Solsona, F., Teixido, I., Mateo, J., Abella, F., & Rius, J. (2014). A queuing theory model for cloud computing. *The Journal of Supercomputing*, *69*(1), 492–507. doi:10.1007/s11227-014-1177-y

Xiao, H., Hu, Z., & Li, K. (2019). Multi-objective VM consolidation based on thresholds and ant colony system in cloud computing. *IEEE Access: Practical Innovations, Open Solutions*, *7*(99), 53441–53453. doi:10.1109/ACCESS.2019.2912722

Zhou Wu received the master's degree from Wuhan University of Technology, China, in 2000. He is currently an associate professor at the Guangdong Songshan Vocational and Technical College, China. In recent years, he has published 23 academic papers, presided over and participated in more than 10 scientific research projects above municipal level presided over 6 information construction projects. His current research interests including software development, parallel algorithm and intelligent information processing and system.

Jun Xiong graduated from Hubei University in 2005. He is currently a lecturer at the Guangdong Songshan Vocational and Technical College, China. In recent years, he has published 14 academic papers, presided over and participated in more than 5 scientific research projects at or above the city level, presided over 20 informatization construction projects. His current research interests including big data, deep learning.