International Journal of Grid and Distributed Computing
Vol. 9, No. 3 (2016), pp.71-82
http://dx.doi.org/10.14257/ijgdc.2016.9.3.09

Research of Improved Shuffled Frog Leaping Algorithm in Cloud Computing Resources

Xuan Chen[1] and Wei Huang[2]

[1.] *Zhejiang Industry Polytechnic College Shaoxing Zhejiang 312000 China*
[2.] *Science Technology Bureau of Shaoxing, Shaoxing Zhejiang 312000 China*
[1]*734140999@qq.com*
[2]*zjsxhw1971@sina.com*

Abstract

Resource scheduling under the condition of cloud computing has always been the focus of current research. This paper analyzes the current situation of cloud computing and introduces shuffled frog leaping algorithm in resource allocation. Aiming at that shuffled frog algorithm is easy to fall into local optimum with fast convergence speed, artificial vector machine is introduced into the subgroups of shuffled frog leaping algorithm, then, self-adaptive crossover probability is introduced into the algorithm's interior searching. The improved shuffled frog leaping algorithm effectively avoids the algorithm from falling into local optimum and meanwhile shortens the time of global searching and optimum. Classic function proves the performance of algorithm in this paper is improved greatly, and CloudSim platform demonstrates that algorithm in this paper can improve the system's efficiency into processing tasks and make resource allocation in cloud computing reasonable and effective.

Keywords: *shuffled frog leaping algorithm; cloud computing resources; artificial vector machine*

1. Introduction

Cloud computing is a kind of computing model in recent years that can acquire convenient and quick visit of shared resources according to demands at any time and any place by using the Internet, thus it can effectively realized sharing of software and hardware resources [1]. It is shown from a large amount of researches that resource distribution of cloud computing is actually a NP problem with multiple tasks and targets and in order to solve this problem, both domestic and foreign scholars have introduced artificial intelligent algorithm into resource distribution and achieved good results. Literature [2] proposed a bat algorithm based on film computing to find the optimum of individual bat locally within the assistance film and make global optimization of optimized individuals by sending them to the main film so as to meet the requirement of resource optimization and distribution within cloud computing. This algorithm can effectively improve the efficiency and reduce processing time of the system in cloud computing. Literature [3] proposed a scheduling method for virtual machine's resources and introduces Service Level Agreement revenue function to estimate the loss in revenue in the condition of renting certain virtual resources. Finally, use hill-climbing algorithm for dynamic adjustment of the number of physical resources to rent to achieve the goal of maximizing. Literature [4] the proposed tiered scheduling strategy for cloud computing services and resource scheduling. Experiments have shown that the use of scheduling improves resource utilization, and provides ideas for further studies of cloud services. Literature [5] for a cloud computing resource scheduling model based on improved artificial bee colony algorithm (IABC), simulation results show that IABC algorithm

ISSN: 2005-4262 IJGDC

improves cloud computing resource utilization, but also greatly reduced task completion time. Literature [6] establishes a scheduling algorithm based on dynamic balance of dynamic change of cloud resources. Simulation experiments show that the method based on the service-driven scheduling method with faster response, with fewer virtual machines, to get more revenue. Literature [7] proposed a performance evaluation model based on resource allocation algorithm for PEPA, the model through the creation of cloud computing for formal interaction between components in the system analysis and reasoning, the cloud computing system performance evaluation. Literature [8] the design, which combines the request rate flexible resource management and energy-aware method and experimental verification of the method on the premise of guaranteeing quality of service can more effectively reduce energy consumption. Literature [9] used a distributed hash table (DHT) for each cloud Server generates a unique node number, this number as the network topology query, retrieval, information storage and information shared markers, creating a structured P2P overlay network for distributed computing. Literature [10] proposed multi-objective integrating Ant Colony optimization algorithm. Improved algorithm for simulation system of experiments showed that at a scheduled time, load balancing is superior to other algorithms. Literature [11] is a layered cloud resources monitoring method based on genetic algorithm and simulation results show that the proposed resource monitor can effective reduce monitoring flow, shorten the time of monitoring.

Frog jumping algorithm for solving multi-objective optimization with cloud computing resources have certain similarities, largely because of the leap frog algorithm as possible to ensure optimization and maximize resource scheduling in cloud computing are similar to individual frog algorithm is similar to cloud computing in the cloud user requests for system resources. Paper first frog jumped algorithm and this based Shang for improved, first in frog jumped algorithm of child Group divided in the introduced artificial vector machine, while in algorithm internal search introduced since adapted cross probability, in must degree Shang improved has frog jumped algorithm easy into local optimal of situation, shortened has global search and optimization of time, through CloudSim platform showed that paper algorithm can improve system processing task of efficiency and cloud computing resources in the scheduling reasonable effective.

2. Main Problems of Resources in Cloud Computing

In cloud computing, the effect of resource allocation can represent the effect of resource allocation to some extent. As each cloud user has different requirement for resources, this paper mainly considers this problem from two aspects: task operation time and operation costs of cloud computing.

Aspects One: Task operation time. In this paper, time is defined as $Timer(Task[i], Source[j])$, referring to that the $Task[i]$ is allocated to the time needed by $Source[j]$, and starting from requirement of cloud server, the time to complete any resource scheduling scheme can be shown as:

$$Timer(x) = \max_{j=1}^{l}(\sum_{i=1}^{k}(Task[i], Source[j])) \quad (1)$$

Aspect Two: network costs needed by the task. In this paper, it mainly refers to the resource bandwidth needed, which is shown as $BandWidth(Task[i], Source[j])$, indicating the expected bandwidth to implement $Task[i]$ on $Source[j]$. Resources needed by any resource scheduling scheme is shown as：

$$BandWidth(x) = \max(\sum_{j=1}^{l}\sum_{i=1}^{k} BandWidth(Task[i], Source[j])) \quad (2)$$

In the above two formulas, l refers to the amount of server in the system and k refers

to the amount of resources needed by the task.

3. Distribution of Improved Frog Jumping Algorithm in Cloud Computing

3.1 Basic Frog Jumping Algorithm

Basic idea is from random initialization of solution in the space of a set of solutions, then the entire population into several groups, all the frogs in each subgroup in accordance with certain search strategies for internal search. After running searches, mix all the crowd of frogs, sorted by groups, to ensure the full exchange of information between the various subgroups.

(1) Division of Subgroup

Suppose the amount of frogs in the group is M , initial group is S , amount of subgroup is k , and the amount of candidate solution in subgroup is n . Adopt $x_i = (x_{i1}, x_{i2}, ...x_{iD})$ to represent the i candidate solution, in which D refers to the dimension of candidate solution, and $S = (x_1, x_2, ...x_M)$.

(2) Internal Search of Subgroup

Suppose X_{zbest} is the best candidate solution in the entire group with the best adaptation, X_{best} is the candidate solution with the best adaptation in subgroup, and X_{worst} is the candidate solution with the worst adaptation in subgroup. In the search within subgroup, it is mainly to update the worse candidate solution X_{worst} . The process is described as follows:

$while(n \le T)$

$X_i^{'} = X_{worst} + t \times (X_{best} - X_{worst})$

$if\ X_i^{'} > X_{worst}$

$X_i^{'} = X_{worst}$ （3）

$else$

$X_{random} = X_{worst}$

$endwhile$

Herein, $X_i^{'}$ refers to a new solution, and t is a random number ranging between 0 and 1. if the new solution is superior to X_{worst} , replace it with the new solution while continuing the search. Otherwise, replace it with a random solution.

$$X_i^{'} = X_{worst} + t \times (X_{best} - X_{worst}) \quad （4）$$

(3) Global Information Exchange

When the internal updating of all the sub-groups has been finished, implement subgroup division and internal search again. Repeat the process until the ultimate condition is met.

3.2 Improved Frog Jumping Algorithm Method

It is an important part as how to reasonably divide subgroups of frog jumping algorithm and the effect of division will directly affect the speed of global convergence and quality of corresponding solution. In this paper, artificial vector machine is introduced at the stage of initial population and select solutions of each candidate solution. Compare

the candidate solution's positive solution and choose the nearest one as the solution of the initial group. Introduce self-adaptive crossover probability in the internal search of subgroup to improve the search precision.

(1) Artificial Vector Machine

Artificial vector extreme learning machine based on generalized inverse matrix theory is a new kind of feed-forward neural network, as opposed to traditional neural networks and support vector machines, it can be worked out in one step the output weights, speed up learning, improving the model efficiency, its structure is shown in Figure 1.

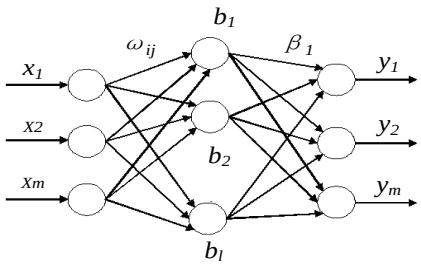

Figure 1. Network Structure of Artificial Vector Machine

Suppose there are m training samples (x_i, y_i), x_i refers to the input vector, and y_i refers to the corresponding output i=1,2,…, m, is a hidden layer. $g(x)$refers to the activation function of neuron at the hidden layer, then:

$$\begin{cases} X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}_{n\times m} \\ Y = \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix}_{1\times m} \end{cases} \tag{5}$$

Suppose the weight matrix of connection input level and hidden level, connection hidden level and output level are ω and β, and they are defined as follows respectively:

$$\omega = \begin{bmatrix} \omega_{11} & \omega_{12} & \cdots & \omega_{1n} \\ \omega_{21} & \omega_{22} & \cdots & \omega_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{l1} & \omega_{l2} & \cdots & \omega_{ln} \end{bmatrix}_{l\times n} \tag{6}$$

$$\beta = \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1m} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{l1} & \beta_{l2} & \cdots & \beta_{lm} \end{bmatrix}_{l\times m} \tag{7}$$

H refers to the output matrix of hidden level and the definition of it is as follows:

$$H(\omega,b,x)=\begin{bmatrix} g(\omega_1 x_1+b_1) & g(\omega_2 x_1+b_2) & \cdots & g(\omega_l x_1+b_l) \\ g(\omega_1 x_2+b_1) & g(\omega_2 x_2+b_2) & \cdots & g(\omega_l x_2+b_l) \\ \vdots & \vdots & \ddots & \vdots \\ g(\omega_1 x_m+b_1) & g(\omega_2 x_m+b_2) & \cdots & g(\omega_l x_m+b_l) \end{bmatrix}_{m\times l} \quad (8)$$

In the formula, $b=[b_1\ b_2\ \cdots b_l]^T$ is the threshold value at the hidden level.

It can be known from Figure 1 that:

$$\beta=H^{+}Y \quad (9)$$

In the formula, H^{+} is the Moerr-Penrose generalized inverse of output matrix at the hidden level.

Divide the initial subgroups as follows:

Suppose $x_i=(x_{i1},x_{i2},.....x_{iD})$ refers to the i candidate solution, D refers to the dimension of candidate solutions, and $S=(x_1,x_2,.....x_M)$. Arrange these solutions according to output matrix by using formula (7), and get the following:

$$H(\omega,s,x)=\begin{bmatrix} g(\omega_1 x_1+s_1) & g(\omega_2 x_1+s_2) & \cdots & g(\omega_l x_1+s_l) \\ g(\omega_1 x_2+s_1) & g(\omega_2 x_2+s_2) & \cdots & g(\omega_l x_2+s_l) \\ \vdots & \vdots & \ddots & \vdots \\ g(\omega_1 x_m+s_1) & g(\omega_2 x_m+s_2) & \cdots & g(\omega_l x_m+s_l) \end{bmatrix}_{m\times l} \quad (10)$$

Put formula (10) into formula (9), set the scope of candidate solutions and $\omega=1$ to get the optimal sub-solution of initial subgroup, which is as follows:

$$H(1,s,x)=\begin{bmatrix} g(x_1+s_1) & 0 & \cdots & 0 \\ 0 & g(x_2+s_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g(x_m+s_l) \end{bmatrix} \quad (11)$$

(2) Self-adaptation Crossing Probability

In the difference concept of genetic algorithm, the value of crossover probability is usually a real number in [0, 1]. The larger the value of crossover probability, it is more beneficial for the accelerating of local search and convergence speed. On the contrary, the diversity of frog jumping algorithm's population and global search capacity can be guaranteed. It can be obtained from Literature [9] that the value of crossover probability should be between [0.3, 0.8], so this paper adopts crossover probability that can be adjusted dynamically and changed according to the times of the algorithm's recycling. With the increase of crossover probability, capacity of local research can be improved, and the formula of self-adaptive crossover probability (set crossover probability is f) is as follows:

$$f=0.8-\frac{t}{2\cdot Max} \quad (12)$$

In formula (12), suppose t refers to the current recycling times, Max refers to the maximum recycling times and the crossover probability is guaranteed to jump between [0.3, 0.8]. The following is obtained by combining the internal search of subgroup in formula (3):

$$X_i^{'}=X_{worst}+f\times(X_{best}-X_{worst}) \quad (13)$$

In formula (13), crossover probability is used to replace t, guaranteeing the optimal local search capacity to some extent.

4. Simulation Experiment

4.1 Test of Algorithm's Performance

Three benchmark functions are adopted as the test tools and effectiveness of algorithm in this paper is verified through comparing with basic frog jumping algorithm in different dimensions, and the comparative results are as shown in Table 1-3.

(1)Schwefel Function

$$f_1(x) = \sum_{i=1}^{N}(-x_i \sin(\sqrt{|x_i|})), -500 \le x_i \le 500$$

(2) Griewank Function

$$f_2(x) = \frac{1}{40000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos(\frac{x_i - 100}{\sqrt{i}}), -600 \le x \le 600$$

(3) Rosenbrock Function

$$f_3(x) = \sum_{i=1}^{N-1}[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2] - 5 \le x \le 5$$

Table 1. Comparative Results of Algorithm's Three Optimization of f_1 Function

Dimension	Algorithm	The Optimal Value	The Minimum Value
10	Basic Frog Jumping Algorithm	1.372e-13	5.432e-15
	Algorithm in this Paper	3.712e-40	6.134e-36
50	Basic Frog Jumping Algorithm	2.213e-06	6.323e-07
	Algorithm in this Paper	4.217e-02	7.182e-01
100	Basic Frog Jumping Algorithm	6.362e-01	7.126e-04
	Algorithm in this Paper	7.317e-02	9.381e-05

Table 2. Comparative Results of Algorithm's Three Optimization of f_2 Function

Dimension	Algorithm	The Optimal Value	The Minimum Value
10	Basic Frog Jumping Algorithm	2.147e-13	4.132e-15
	Algorithm in this Paper	5.812e-15	7.174e-25
50	Basic Frog Jumping Algorithm	2.514e-07	7.523e-09
	Algorithm in this Paper	3.237e-02	6.192e-01
100	Basic Frog Jumping Algorithm	2.962e-02	4.726e+03
	本文算法 Algorithm in this Paper	1.137e-02	2.951e+02

Table 3. Comparative Results of Algorithm's Three Optimization of f_3 Function

Dimension	Algorithm	The Optimal Value	The Minimum Value
10	Basic Frog Jumping Algorithm	7.747e-08	5.132e-09
	Algorithm in this Paper	6.912e-25	4.334e-24
50	Basic Frog Jumping Algorithm	5.234e-02	6.323e-03
	Algorithm in this Paper	1.192e-08	2.182e-11
100	Basic Frog Jumping Algorithm	5.681e-07	6.145e-09
	Algorithm in this Paper	3.254e-03	4.182e-04

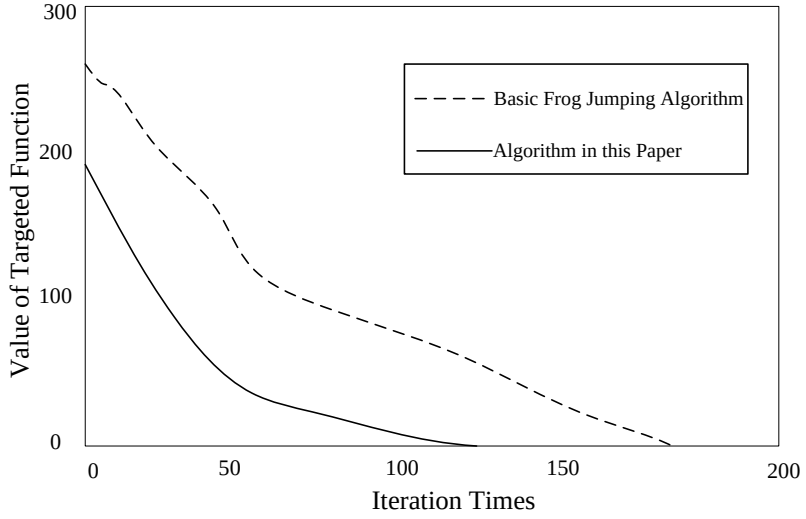

Figure 2. Comparative Result of Schwefel Function

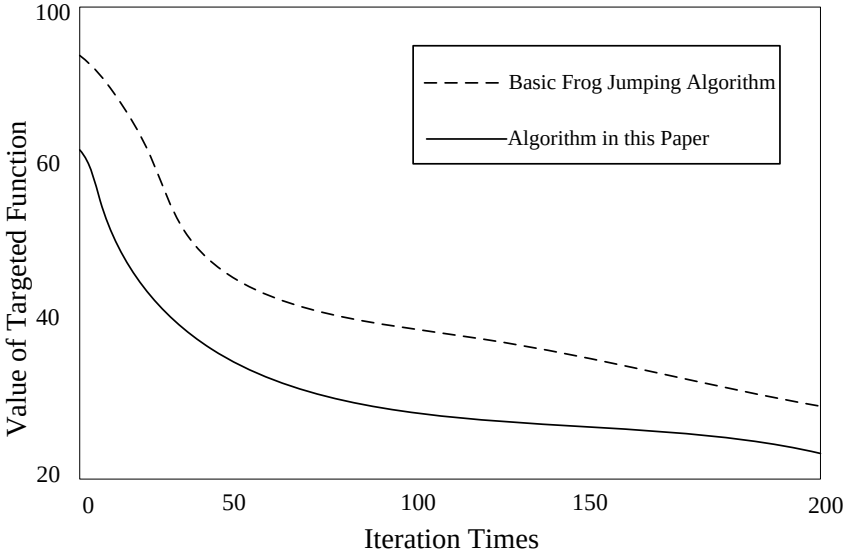

Figure 3. Comparative Result of Griewank Function

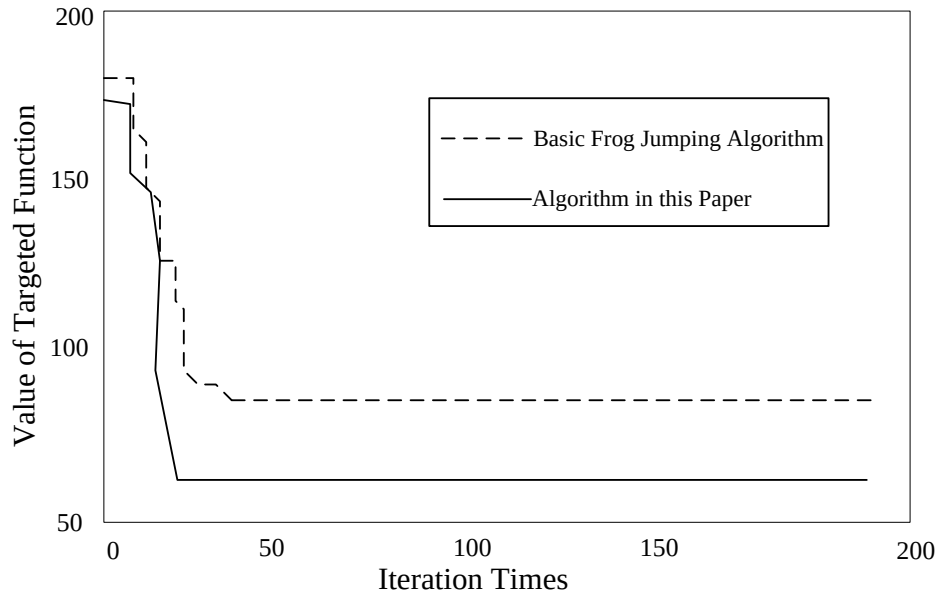

Figure 4. Comparative Result of Rosenbrock Function

It can be found from Figure 2-4 that compared with basic frog jumping algorithm, algorithm in this paper has significantly been improved in performance, effectively preventing the probability of local convergence and improving efficiency of the algorithm.

5.2 Simulation Experiment of Cloud Computing Platform

(1) Comparison with Basic Frog Jumping Algorithm in Cloud Computing

Set there are 150 virtual tasks and 10 virtual resources, and it can be found by comparing the network throughput rate, cloud joints' consumption delay rate and cloud joints' network costs rate of these two algorithms that algorithm in this paper is superior to basic frog jumping algorithm in terms of network throughput rate in different tasks. Especially, with the constant increase of the amount of tasks, the effect is more significant and becomes suitable for the task scheduling in cloud computing. In terms of consumption delay of cloud joints, algorithm in this paper is nearly 15-20% lower than basic frog jumping algorithm. Meanwhile, consumption rate of consumption costs is also slightly lower, which is mainly because algorithm in this paper reasonably divide the scale of frog jumping population while using artificial vector machine and crossover factor to avoid the algorithm from consuming more unnecessary time in seeking invalid space, further reducing time for the algorithm to produce the optimal solution and saving costs. It is as shown in Figure 5-7:

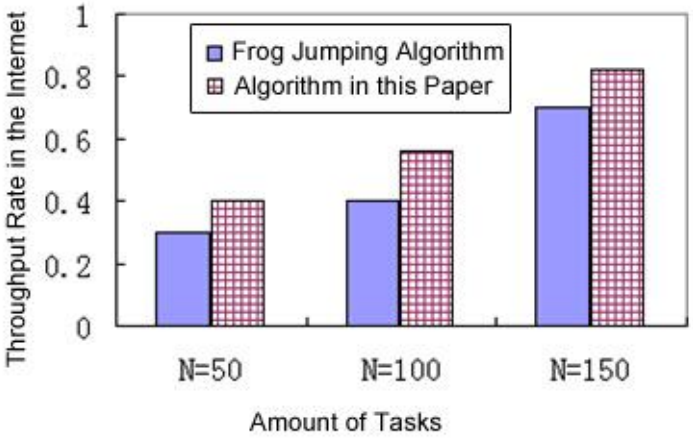

Figure 5. Comparison of Throughput Rate of Different Tasks in the Internet

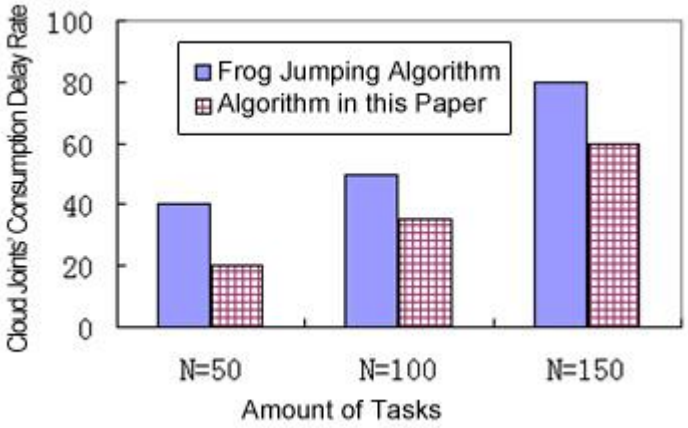

Figure 6. Comparison of Cloud Joints' Consumption Delay

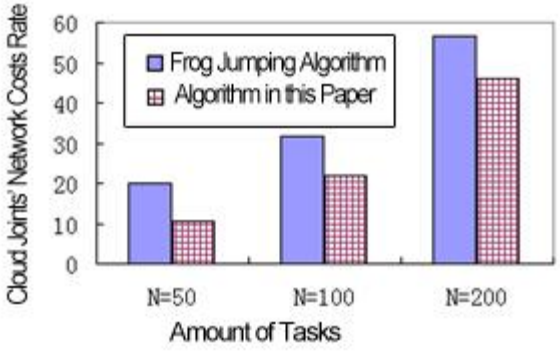

Figure 7. Cloud Joints' Network Costs Comparison

(2) Comparison with Other Intelligent Algorithms

Literature [2], [7] and [8] are the resource scheduling algorithm of cloud computing, which is relatively new in recent years. Compare these algorithms with algorithm in this paper while setting 800 virtual tasks, 10 virtual resources and 100 as the iteration times. Set parameters for them respectively according to the references, and the comparative results are as shown in Figure 8-9.

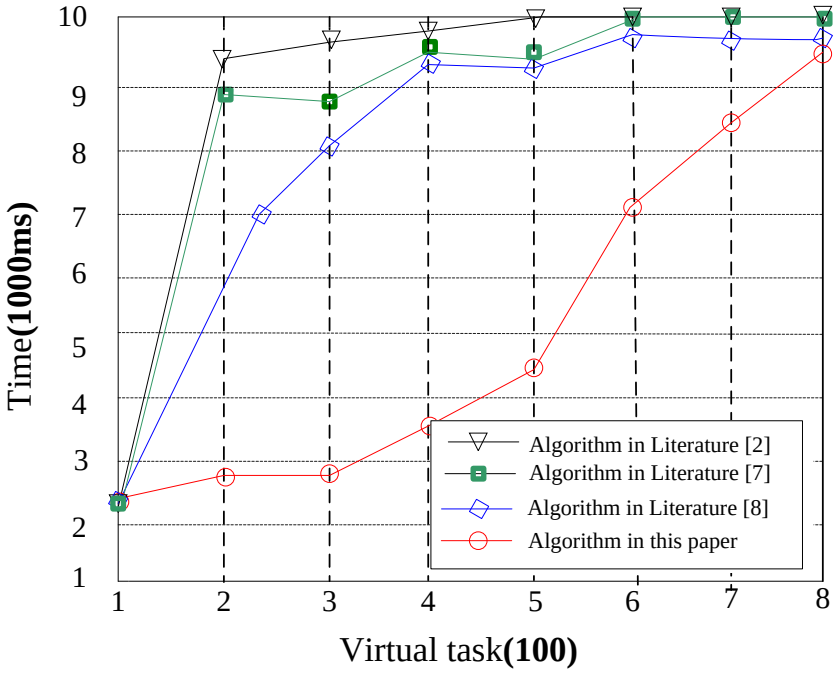

Figure 8. Comparison of Time to Complete Tasks for 4 Algorithms

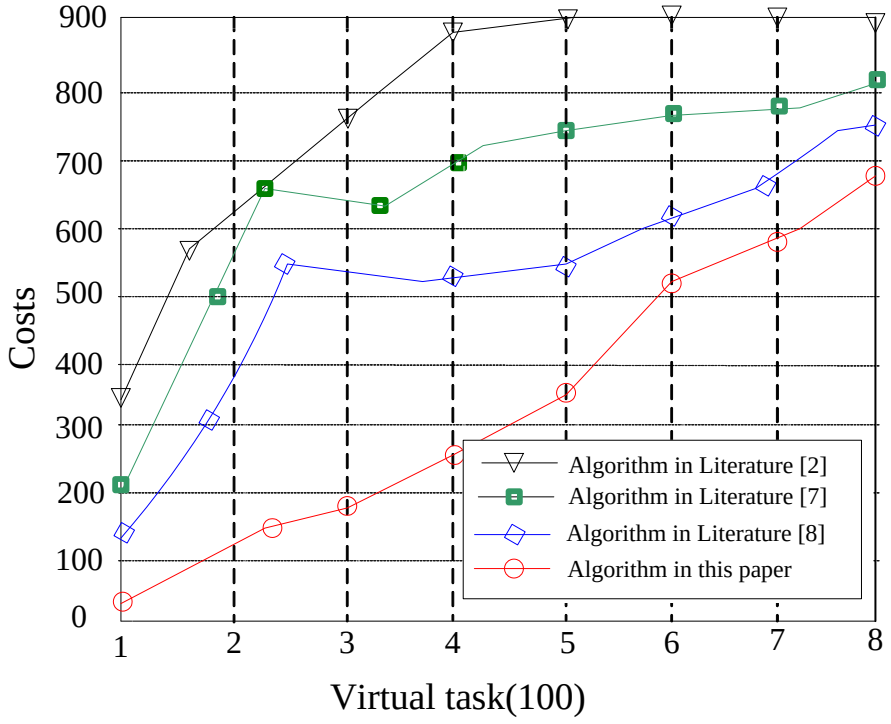

Figure 9. Comparison of Cost Consumption for 4 Algorithms

Less time than. other algorithms in the three references. Meanwhile, with the constant increase of the amount of task, it has the least amplitude in time consumption, indicating algorithm in this paper can effectively balance resources. If ca be found in Figure 9 that after improvement, algorithm in this paper is superior to other algorithms in the three references in terms of cost consumption in cloud computing so as to better meet the requirement of resource scheduling in cloud computing.

6. Conclusion

This paper first analyzes the current situation of cloud computing and allocate resources by introducing frog jumping algorithm. Considering that the frog jumping algorithm has quick convergence speed and is easy to fall into local optimum, artificial vector machine is introduced into the division of frog jumping algorithm's subgroups. Meanwhile, self-adaptive crossover probability is introduced into the algorithm's internal search, improving the situation that it is easy for frog jumping algorithm to fall into local optimum to some extent and shortening the time of global search and optimization. CloudSim platform shows that algorithm in this paper can improve the system's efficiency in dealing with tasks and the reasonable scheduling of resources in cloud computing.

References

[1] L Weiwei,Q Deyu, "Survey of Resource Scheduling in Cloud Computing[J]", Computer Science, vol 39, no. 10, **(2012)**, pp. 1-5
[2] N Bin,GU Qiong,WU Zhao.Bats algorithm research in cloud computing resource scheduling based on membrane computing[J].Application Research of Computers, vol. 32, no. 3, **(2015),** pp. 830-833.
[3] Y Shiyang, Z Wenbo, Z Hua, "SLA-Oriented Virtual Resources Scheduling in Cloud Computing[J]", Computer Applications and Software, vol. 32, no. 4, **(2015)**, pp. 11-14.
[4] X Lixia, Y Yanxin, "Analysis of service scheduling and resource allocation based on cloud computing[J]", Application Research of Computers, vol. 32, no 2, **(2015),** pp.528-531.
[5] Z Tao, Z Ying, "Scheduling Model Cloud Computing Resource Based on Improved Artificial Bee Colony Algorithm[J]", Microelectronics & Computer, vol. 31, no. 7, **(2014)**, pp. 147-150.
[6] Z Aike, F Baolong, "Cloud resource scheduling algorithm based on dynamic change of maximum profit equilibrium[J]", Journal of Chongqing University of Posts and Telecommunications (Natural Science Edition), vol. 26, no. 5, **(2014)**, pp. 706-710.
[7] W Qian, S Zhenguo, S Wanjie, "PEPA model approach for performance evaluation of dynamic resource provision in cloud computing[J]", Application Research of Computers, vol. 32, no. 4, **(2015),** pp. 1179-1183
[8] X Wei, L Bin, "An Elastic Resource Management Mechanism Based on Perception of Energy Consumption in cloud computing Environment[J]".Journal of Sichuan University(Engineering Science Edition), vol. 47, no. 2, **(2015)**, pp. 1-5.
[9] L Pu, C Shiping, L Jianfeng, "Cloud resources locating algorithm based on peer-to-peer network[J]", Application Research of Computers , vol. 30, no. 2, **(2013)**, pp. 570-573.
[10] Z Liyun, Z Lifeng, "Multi-objective integrated ant colony optimization scheduling algorithm based on cloud resource[J]", Journal of Computer Applications, vol. 32, no. 7, **(2012)**, pp. 1916-1919.
[11] C Chunling, L Yang, Z Dengyin, "Hierarchical cloud resource monitoring based on genetic algorthm[J]", Journal of nanjing university(Natural science), vol. 49, no. 4, **(2013),** pp. 491-499.

Authors

Xuan Chen (1979.03-), male, master, lecturer, research direction: information security, wireless sensor and cloud computing.

Wei Huang (1971.10-), male, senior engineer, research direction: computer technology ap plication& automation.

