International Journal of Database Theory and Application
Vol.8, No.2 (2015), pp. 205-214
http://dx.doi.org/10.14257/ijdta.2015.8.2.19

Research on the Improved Shuffled Frog Leaping Algorithm in Cloud Computing Resources

Li Yong-Qiang[1] and PanJin[2]

[1]*Software Technology Vocational College, North China University of Water Resources and Electric Power, Zhengzhou 450045, China*
[2]*Henan Procuratorial Vocational College Zhengzhou 451191, China*
1696450309@qq.com[1], 1632921040@qq.com[2]

Abstract

The problem of resource scheduling in the environment of cloud computing has always been the focus of research. In this paper, the current status of cloud computing is first analyzed, and on the basis of the features of resource scheduling in cloud computing, the Intelligent Frog Leaping Algorithm is introduced and improved. First, artificial vector machine is introduced into the subgroup classification of the Frog Leaping Algorithm, and at the same time the self-adaptive crossover probability is introduced into the internal search of the algorithm. To some extent, this improves the condition that the Frog Leaping Algorithm is easy to fall into local optimum, as well as reduces the time spent on global search and optimization. Through the Cloud Sim platform, it is found that the algorithm presented in this paper can improve the efficiency of the system in task processing and achieve a rational scheduling of resources in cloud computing.

Keywords*: Frog Leaping Algorithm, cloud computing resources, artificial vector machine*

1. Introduction

Cloud computing is a computing mode developed along with distributed processing, parallel processing and grid processing. It results from the integration of the concepts such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). It is a difficult part of research to find a way to properly allocate the computing resources in cloud computing and to make every node used effectively. Researchers at home and abroad have made research on this. Reference [3] describes that it is effective to some extent to introduce Swarm Intelligence Algorithm in virtual scheduling of cloud computing to carry out resource scheduling. Reference [4] applies the improved genetic algorithm in the task scheduling in cloud computing, the algorithm makes a certain effect. Reference [5] states that with the combination of the ACO algorithm and K-medoid algorithm, ACO-K-medoid resource allocation optimized algorithm is proposed based on the cloud computing environment, which gains optimal computing resources and improves the efficiency of cloud computing. Reference [6] proposes a cloud computing resource scheduling model based on Chaos Particle Swarm Algorithm, which improves the efficiency of resource utilization and is better in practicality and feasibility. Reference [7] uses the Discrete Particle Swarm Algorithm to solve the problem. With regard to the characteristics of the model, combined with the real situation of cloud computing service operation, computing samples were designed to carry out stimulation test. Reference [8] improves the starter strategy of resource scheduling in the environment of cloud computing in order to effectively enhance the overall system processing capability in this environment. Through simulation experiments, the results show that this method can effectively avoid the unbalanced load handling, and improve the overall processing capability of the system.

ISSN: 2005-4270 IJDTA
Copyright ⓒ 2015 SERSC

According to the features of resource scheduling in cloud computing, the Intelligent Frog Leaping Algorithm is introduced and improved on this basis. First, artificial vector machine is introduced into the subgroup classification of the Frog Leaping Algorithm, and at the same time the self-adaptive crossover probability is introduced into the internal search of the algorithm. To some extent, this improves the condition that the Frog Leaping Algorithm is easy to fall into local optimum, as well as reduces the time spent on global search and optimization. Through the CloudSim platform, it is found that the algorithm presented in this paper can improve the efficiency of the system in task processing and achieve a rational scheduling of resources in cloud computing.

2. The Main Issues of Cloud Computing Resources

In the environment of cloud computing, the effect of resource allocation represents the satisfaction of users to some extent. Because cloud users have different requirements on the resources, the satisfaction here should be considered in several aspects, including the operation time of cloud computing, the cost and consumption of network and task execution time and so on. As there are too many factors involved, we mainly take the task execution time and the cost of network into account in this paper:

Question 1: the representation of task execution time. In this paper, time is defined as $Timer(Task[i], Source[j])$, which represents the time needed for allocating task $Task[i]$ to the resource $Source[j]$. From the requests of virtual machine of the cloud client server, the completion time of any resource scheduling program is indicated as:

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

In formula (1), l represents the number of virtual machines in the system, the quantity of resources needed for task k.

Question 2: the cost of network needed for tasks. The cost of network here mainly refers to the resource bandwidth required. Its value is indicated as $BandWidth(Task[i], Source[j]))$, which represents the expected bandwidth for allocating and implementing task $Task[i]$ on $Source[j]$. The resources required for any resource scheduling plan are represented as:

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

In formula (2), l represents the number of virtual machines in the system, the quantity of resources needed for task k.

3. Improved Frog Leaping Algorithm

3.1. Frog Leaping Algorithm

The basic idea of the Frog Leaping Algorithm: from the space of random solutions, a group of initialized solutions (population) is generated, which as a whole are then divided into multiple sub-groups, and the frogs in the sub-groups implement internal search in accordance with certain search strategies. After several times of internal searches in the defined subgroups, mix all the frogs and then divide them into subgroups again by sequencing, thus ensuring an overall exchange of information among the subgroups.

(1) Subgroup classification

 Copyright ⓒ 2015 SERSC

The number of frogs in the population is set as M, the initial population as S, the number of subgroups as k, and the number of candidate solutions in the subgroups is set as n. $x_i = (x_{i1}, x_{i2}, ...x_{iD})$ is used to represent the number i candidate solution, and here D represents the dimension of candidate solutions, $S = (x_1, x_2, ...x_M)$.

(2) The internal search of subgroups

It is assumed that X_{zbest} is the candidate solution in the whole population with the best fitness, X_{best} the candidate solution in the subgroups with the best fitness, and X_{worst} the candidate solution in the subgroups with the worst fitness. When searching within the subgroups, it is mainly to update the worst candidate solution X_{worst}. The process is described below:

$$
\begin{aligned}
&while(n \le T)\\
&X_i^{'} = X_{worst} + t \times (X_{best} - X_{worst})\\
&if\ X_i^{'} > X_{worst}\\
&\quad X_i^{'} = X_{worst}\\
&else\\
&\quad X_{random} = X_{worst}\\
&endwhile
\end{aligned}
\tag{3}
$$

In this formula, $X_i^{'}$ represents a new solution, and t is the random number within the range of (0,1). If the new solution generated is superior to X_{worst}, the latter will be replaced the former and at the same time the searching continues; or else, the random solution is used for replacement.

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

(3) Global information exchange

When the internal update of all the subgroups is finished, the subgroup classification and internal searching will be implemented again, in a repeated way until the final condition is satisfied.

3.2. The Content of Improvement

How to classify the subgroups in a better and rational way forms an important part of the leapfrog algorithm, and the quality of classification will directly influence the speed of global convergence of the algorithm and the quality of corresponding solutions. A common way is to generate the initial solution of the algorithm through random initialization. In the population initialization stage, the artificial vector machine mechanism is introduced, to select the solutions generated by each candidate solution. Through the comparison of the candidate solutions and positive solutions, a solution of a short distance is selected as the solution of the initial population. In the subgroup internal search stage, the introduction of the reverse learning mechanism improves the precision of searching.

(1) Artificial Vector Machine

Artificial Vector Extreme Learning Machine is a new type of feed-forward neural network based on the generalized inverse matrix theory. Compared with the traditional neural network and support vector machine, it can improve the speed of learning and the

Copyright ⓒ 2015 SERSC

efficiency of modeling, as it can work out the output weight of network just by one step. Its structure is shown in Figure 1.

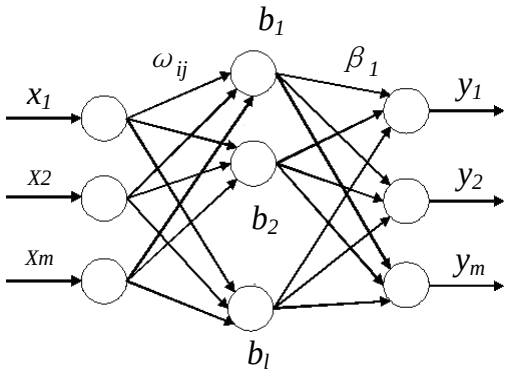


Figure 1. The Network Structure of Artificial Vector Machine

Supposing there are m training samples (x_i, y_i), xi represents the input vector, yi represents the corresponding output i=1, 2, m, a hidden layer, and $g(x)$ represents the activation function of neurons in the hidden layer, we will get:

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

Set the weight matrix connecting the input layer and hidden layer and that connecting the hidden layer and output layer are ω and β, respectively, which are defined as follows:

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

H is the output matrix of the hidden layer, which is defined as follows:

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

In this formula, $b = [b_1\ b_2 \cdots b_l]^T$ is the threshold value of the hidden layer?

Then, according to Figure 1, we can see:

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

 Copyright ⓒ 2015 SERSC

In this formula, *H*+ is the Moerr-Penrose generalized inverse of the output matrix *H* of the hidden layer.

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

The initial subgroups are classified as follows:

Use $x_i = (x_{i1}, x_{i2}, ...x_{iD})$ to represent the number i candidate solution, in which D represents the dimension of the candidate solution, $S = (x_1, x_2, ...x_M)$. These solutions are sequenced by adopting formula (8) according to the output matrix, to get the following formula:

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

Substitute Formula (10) into Formula (9), define the set of candidate solutions, and set $\omega = 1$ to get the optimal classification of the initial subgroup, which is as follows:

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

(2) Self-adaptive crossover probability.

In the concept of difference in genetic algorithms, the value of crossover probability is generally real numbers within the range of [0, 1]. The larger the value of crossover probability is, the more beneficial it will be to the acceleration of local searching and convergence speed, and on the contrary, it can keep the population diversity and the global search capability of the Frog Leaping Algorithm. The value of crossover probability obtained through Reference [9] shall be within the range of [0.3, 0.8], so in this paper, dynamic adjustment of crossover probability is adopted. According to the cyclic algebra of the algorithm, the value of crossover probability is changed dynamically. With the increase of the crossover probability, the ability of local searching can be improved. The formula of self-adaptive crossover probability (set the crossover probability as f) is as follows:

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

In formula (11), set t to represent the current cycle index, *Max* to represent the maximum cycle index, to ensure that the crossover probability fluctuates within the range of [0.3, 0.8]. In combination with the subgroup internal searching of formula (3), the following formula is obtained:

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

In this formula, the crossover probability is used to institute t can to some extent guarantee the optimal local searching ability.

4. Experimental Simulation

In order to better show the superiority of the algorithm in cloud computing environment, in this paper, first the performance of the algorithm presented is tested, and then a test of resource scheduling is carried out on the cloud computing simulation platform.

Copyright ⓒ 2015 SERSC

4.1. Performance Test for the Algorithm

Three basic functions are adopted to make comparative test. Through the comparison with the basic Frog Leaping Algorithm, the effectiveness of the proposed algorithm is verified.

(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. The Optimization Results of the Two Algorithms

Function	Algorithm	Average Maximum	Average Minimum
f1	Basic frog leaping algorithm	0.042514	0.000117
	Algorithm of this paper	0.018725	0.000019
f2	Basic frog leaping algorithm	0.093412	0.008754
	Algorithm of this paper	0.012713	0.005312
f3	Basic frog leaping algorithm	0.082513	0.009652
	Algorithm of this paper	0.042716	0.007126

The three test functions are compared with the algorithms of Reference [4], Reference [8] and presented in this paper, to obtain the results shown in Figure 1-3.

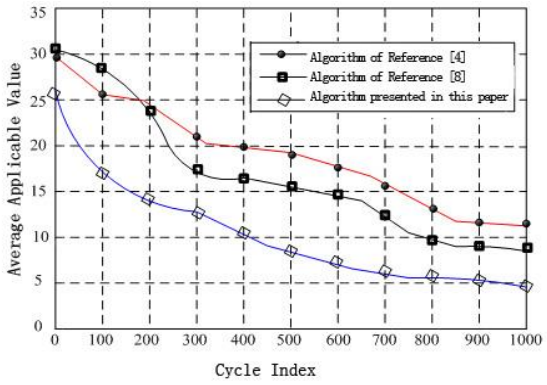


Figure 1. The Result of Comparison with the Schwefel Function

Copyright ⓒ 2015 SERSC

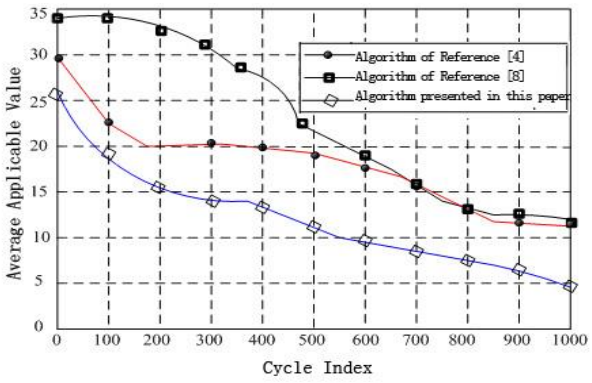


Figure 2. The Result of Comparison with the Griewank Function

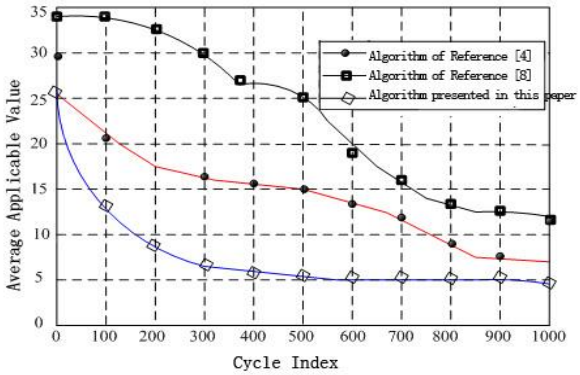


Figure 3. The Result of Comparison with the Rosenbrock Function

From Figure 1-3, it can be found that compared to the algorithms of the references, the algorithm presented in this paper can effectively reduce the possibility of falling into local convergence, and thus shorten the time spent in achieving the optimal value of the objective function.

4.2. Simulation Experiment on the Cloud Computing Platform

In the experiment, the cloud computing platform Cloudsim is used to simulate the cloud computing environment. There are 150 experimental tasks, and each task varies in the requirements of task length and bandwidth. In accordance with different tasks, the algorithm of this paper and the Basic Frog Leaping Algorithm are different in task processing.

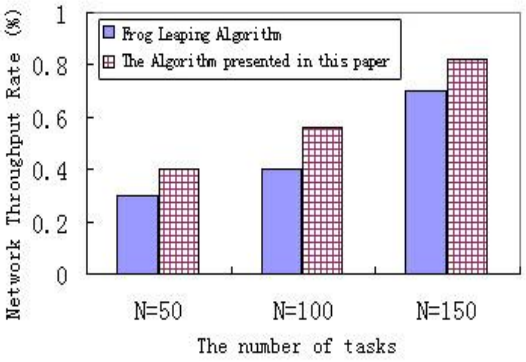


Figure 4. The Comparison of Network Throughput Rate among Different Tasks

Copyright ⓒ 2015 SERSC

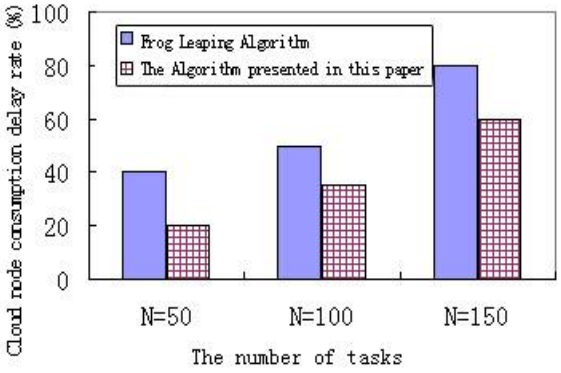
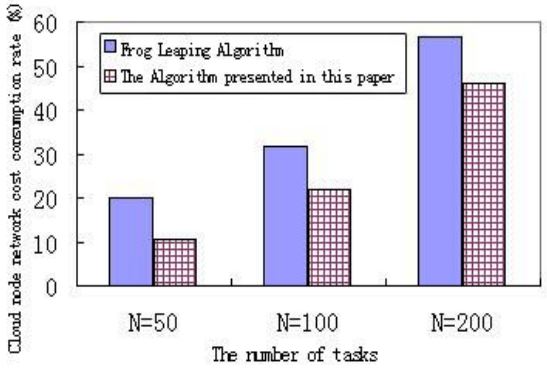


Figure 5. The Comparison of Cloud Node Consumption Delay



Cloud node network cost consumption rate (%)

Figure 6. The Comparison of Cloud Node Network Cost

From Figure 4-6, it can be seen that the algorithm presented in this paper performs better than Basic Frog Leaping Algorithm in the aspects of network throughput rate, node time consumption, node network cost and so on. It proves that the algorithm presented in this paper is more suitable for solving the problems concerning service quality and resource scheduling in the cloud computing environment.

5. Conclusion

It has always been a focus of research how to make better resource scheduling in cloud computing. In this paper, an improved Frog Leaping Algorithm is put forward to further guarantee the reasonable resource scheduling on the cloud computing platform. The simulation experiments have proved that the improved Frog Leaping Algorithm performs well in optimizing and properly allocating resources.

References

[1] M. Armbrust, A. Fox, R. Griffith, *et. al*, "A view of cloud computing", Communications of the ACM, vol. 53, no. 4, **(2009)**, pp.50-58.
[2] X. Baomin，Z. Chunyan, H. Enzhao, *et,al*, "Job scheduling algorithm based on Berger model in cloud environment", Advances in Engineering Software, vol. 42, **(2011)**, pp. 419-425.
[3] Z. Wang Jun, Z. Meng-Hua, G. Wenyue, "Cloud Computing Resource Schedule Strategy Based on MPSO Algorithm", Computer Engineering, vol. 37, no. 11, **(2011)**, pp.43-45.
[4] Z. Cong-Bin and D. Zhong-Jun, "Improved GA-based task scheduling algorithm in cloud computing", Computer Engineering and Applications, vol. 49, no. 5, **(2013)**, pp. 77-80.

 Copyright ⓒ 2015 SERSC

[5] W. Deng-ke and L. Zhong, "A Task Scheduling Algorithm Based On Pso And Aco For Cloud Computing", Computer Applications and Software, 2013, vol. 30, no. 1, **(2013)**, pp. 290-293.
[6] X. Yu, "Research on Resource Schedule Model under Cloud Environment", Computer simulation, vol. 30, no. 5, **(2013)**, pp. 362-365.
[7] L. Yu, Z. Zhi-Wen and L. Xiao-Lan, "Resource Scheduling Strategy Based Optimized Generic Algorithm In Cloud Computing Environment", Journal of Beijing Normal University (Natural Science), vol. 48, no. 4, **(2012)**, pp. 378-382.
[8] Zhangbo, "Improved dispatching method of load based on initialize strategy", Computer Engineering and Applications, vol. 49, no. 17, **(2013)**, pp. 73-77.
[9] M. Runyu, W. Xiaoping, X. Xiaoping, "An adaptive differential evolution algorithm", Computer Application and software, vol. 25, no. 12, **(2008)**, pp. 7-8.

Author

Li Yongqiang (1974.1-) Lecturer, Master, and Research Orientation: Computing Network; Pan Jin (1974.11-) Associate Professor, Master, And Research Orientation: Computer Network.

Copyright ⓒ 2015 SERSC

 Copyright ⓒ 2015 SERSC