Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 12, December 2019, pp. 3151-3160
DOI: 10.23940/ijpe.19.12.p6.31513160

A Cloud Computing Load Algorithm

Yao Yao

Zhengzhou Institute of Technology, Zhengzhou, 450044, China

Abstract

For the issue of resource load prediction in cloud computing, a modified artificial bee colony algorithm and SVM are combined to construct a predictive model. First, by using reverse learning to initialize the population, differential evolution selects the individual population. The point strategy is used to construct the honey source selection route of the algorithm. The feedback mechanism reduces the shortcomings of the algorithm falling into the local optimum. Second, the parameters in the SVM prediction model are optimized and the best ones are found by using the improved bee colony algorithm. In the final simulation experiment, the proposed IABC algorithm has better prediction accuracy than the SVM, the LSSVM and other prediction algorithms, and so it has a certain promotional value.

Keywords: cloud computing; resource loading; Artificial Bee Colony Algorithm; SVM

(Submitted on July 28, 2019; Revised on September 15, 2019; Accepted on November 16, 2019)

1. Introduction

The resource load in cloud computing is related to the quality of cloud services. By predicting the future resource load, the resource scheduling efficiency of cloud computing can be effectively improved [1]. Due to the real-time and dynamic characteristics of cloud computing resources, resource load forecasting is also characterized by uncertainty and nonlinearity [2]. Therefore, foreign scholars use many methods to study it. The algorithms in [3-6] are mainly based on static object research, but they cannot achieve good prediction results. Literature [7] proposed distributed load forecast model mining based on hybrid gene expression programming and cloud computing. The experimental results show that the proposed algorithm has better average time-consumption, convergence, and forecast accuracy and excellent parallel performance with respect to time and size. Literature [8] was proposed for cloud computing and uses the PSO-based weighted support vector machine. The experimental results indicate that the proposed algorithm is superior to the other four comparison prediction algorithms with respect to prediction accuracy and efficiency. Literature [9] proposed analyzing the migration decision based on the present and future workloads of the system. The results prove that the proposed technique is more energy efficient and balances the workloads. Literature [10] proposed a heuristic-based algorithm, and the results show that the proposed algorithm can reduce the number of active servers. Literature [11] proposed a load balancing algorithm based on the processing capacities of virtual machines (VMs) in cloud computing. The simulation results show that the proposed algorithm can reduce the number of active servers. Literature [12] proposed a correlation time series model to predict resource consumption, which provides a feasible way for the on-demand use and resource elasticity estimation of cloud computing. Literature [13] proposed a virtual machine resource optimization allocation scheme based on load forecasting. The experimental results show that the scheme improves the resource utilization while ensuring the performance of the host. Literature [14] proposes a load forecasting method based on a parallel random forest algorithm for big data, and the experimental results show that the scheme improves the resource utilization while ensuring the performance of the host. Literature [15] proposed a solution for the load balancing methods and related algorithms for heterogeneous cloud computing platforms. The results show that the algorithm is feasible and has certain advantages.

In this paper, the improved artificial bee colony algorithm and SVM are combined to generate a new algorithm (the Improved Artificial Bee Colony - Support Vector Machine Algorithm, or IABC-SVM for short) to construct a prediction model. By initializing the population of the artificial bee colony algorithm, the differential evolution algorithm is used to

* Corresponding author.
E-mail address: henanyy1982@sina.com

improve the performance of the artificial bee colony algorithm using individual selection. In addition, an attraction strategy is used to model the honey source selection, and a feedback mechanism and survival of the fittest are further used. This method can further optimize the parameters in the SVM and improve the prediction accuracy.

2. Introduction to and Deficiency of Artificial Bee Colony Algorithm

The Artificial Bee Colony (ABC) algorithm is a group intelligent optimization algorithm that was inspired by honey bee behavior. Its process divides the work, information sharing and information exchange between each bee colony so that they can cooperate with each other to complete the work of collecting honey. The ABC algorithm has the advantages of simple operations, less parameter control and strong robustness. In the ABC algorithm, bees are divided into three categories: employed bees, followers, and scout bees. The search behaviors of these three types of bees are as follows.

(1) Employed bee search. Employed bees in the algorithm mainly search around a specific honey source. When it is necessary to mine nectar near another honey source, the employed bees conduct a new search near the honey source. The search formula is as follows (1). After a new source of honey is found, the bees evaluate the fitness (Formula (2)) and compare it with the fitness of the previous honey source. If it is higher than the previous one, then they move to it.

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{1}$$

$$v'_{ij} = \begin{cases} v_{ij}, & \text{if } fit(v_{ij}) > fit(x_{ij}) \\ x_{ij}, & \text{if } fit(v_{ij}) \le fit(x_{ij}) \end{cases} \tag{2}$$

In the formula, ϕ_{ij} refers to a random number in the range of [-1, 1], and $fit(v)$ is the adaptation evaluation function.

(2) Follower bee search. After the employed bees send honey to the hive, they will be invited to follow the bees to fly to the honey source. The quality of the honey source determines whether or not the follower accepts the invitation. Generally, the higher the quality of the honey source is, the greater the number of followers that will be attracted. In contrast, if the honey source is lower quality, followers will be lost, causing the honey source to be abandoned. The probability of following the bee selection is as shown in Equation (3).

$$P_i = \frac{fit(x_i)}{\sum_{i=1}^{D} fit(x_i)} \tag{3}$$

(3) Scouting bee search. When the honey source is harvested, the employed bees near the honey source become reconnaissance bees and perform a global search. The formula for their search is shown in (4).

$$X_i = X_{\min} + R \cdot (X_{\max} - X_{\min}) \tag{4}$$

In the formula, R is a random number in the range of [-1, 1].

Although the algorithm is widely used in many practical problems, there are still some deficiencies. For example, the initialization of the population is random, and the effective utilization of the shared information of the entire population is not considered. This results in the algorithm having poor local search ability and falling into the local optimum; the convergence speed needs to be further improved.

3. Improvement of Artificial Bee Colony Algorithm

3.1. Population Initialization and Individual Selection

The population of the basic ABC algorithm is uninitialized, i.e., the position is random, which results in a population that is unevenly distributed. Therefore, the performance of the whole algorithm will be affected to some extent. To effectively avoid this situation, the search efficiency of the algorithm must be effectively improved. This section uses a reverse learning strategy to initialize the population to increase population diversity in order to better generate optimal solutions. The solution is to calculate a corresponding inverse solution using a reverse learning strategy, combine and sort the two solutions, and select an individual as the next generation individual according to the set conditions. The steps are as follows.

Step 1 Initialize the population size N.

Step 2 Conduct a random initialization phase.

for $i = 1{:}\, N$ do
  for $j = 1{:}\, D$ do
    $X_i^j = X_{\min}^j + rand(0,1)(X_{\max}^j - X_{\min}^j)$
  endfor
endfor

Step 3 Performing a reverse learning.

for $i = 1{:}\, N$ do
  for $j = 1{:}\, D$ do
    $oX_i^j = X_{\min}^j + X_{\max}^j - X_i^j$
  endfor
endfor

Step 4 Select the N values with the best adaptation from { $X(N) \bigcup OX(N)$ } as the initial solutions of the group.

In the initial swarm of the artificial bee colony algorithm, the initial solution of each algorithm is generated using reverse learning to generate another reverse solution. Afterwards, individual bees are selected according to the set value of N as the initial solution of the new swarm. This can effectively increase the diversity of the population. The *rand* is generally set to 0.5.

The individual choice is related to whether the algorithm is easily to falls into the local optimum and prematurely converges. Therefore, a new population is obtained by performing operations such as difference and mutation on individuals in the population. Second, it crosses the operation with the parents, calculates the individual fitness values in the population, arranges the excellent individuals, and obtains a new generation group. Finally, the evolutionary process including mutation, crossover and selection is carried out.

(I) Variation. For individual x_i, generate the variant individuals as follows

$$u_i = x_{r1} + F(x_{r2} - x_{r3}) \tag{5}$$

In the formula, x_{r1}, x_{r2} and x_{r3} are randomly selected individuals from the evolutionary population; we set F as the scaling factor to control the effects of vector generation.

(II) Crossover. To further increase the population's diversity, a differential evolutionary algorithm is introduced.

$$v_{i,j} = \begin{cases} u_{i,j}, & rand(0,1) \le CR \\ x_{i,j}, & rand(0,1) > CR \end{cases} \tag{6}$$

In the formula, $j = 1, 2, 3, \cdots, D$, where D is the spatial dimension, and CR is the probability from 0 to 1.

(3) Selection. Using the greedy strategy, the crossed individual v_i and the parent individual *v* are generated as offspring according to Formula (7).

$$x_{i+1} = \begin{cases} v_i, \text{ if } f(v_i) \le f(x_i) \\ x_i, \text{ if } f(v_i) > f(x_i) \end{cases} \tag{7}$$

The steps for the differential evolution of individuals are as follows:

Step 1 Set CR to be 0.5.

Step 2 Vary the individuals according to the Formula (5).

Step 3 Apply Formulas (6) and (7) to generate new individuals and variants.

Step 4 Conduct the overall evaluation of new populations of new individuals to determine next-generation populations.

3.2. Honey Source Selection for Followers

In the ABC algorithm, followers choose the honey source using the roulette strategy. Although it can guarantee that the best honey source can be selected, sometimes it will miss some better honey sources as a whole. Obviously, this will increase the resource consumption and iteration time. The algorithm of this paper proposes a "traction point" strategy under the premise of obeying the original intention of this concept. We change the search method of followers by introducing cr . All follower bees are developed together using proportional scaling around the attraction point, thereby effectively improving the developmental capability of the algorithm as a whole. The attraction point cr is used as the "queen bee" in the bee colony and attracts all followers to approach it according to a certain scale range. The search map is shown in Figure 1. The white box represents the attractor, the black sphere represents the initial position of the different populations, and the triangle represents the new location of the population surrounding the attraction point.

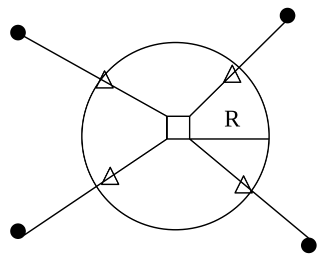

Figure 1. Schematic map of the search

In this paper, the acquisition of the attraction points is carried out in two ways. The first is when the individual of the population obtains the global optimal solution, and the value of the attraction point cr is obtained using Formula (8). The second case is when the global optimal solution is not obtained, and the attraction point is obtained using Formula (9).

$$cr = v_{best} \times r_1 + r_2 \tag{8}$$

$$cr = v_i + (v_{best} - v_i) \times r_3 \tag{9}$$

v_i refers to the honey source of the currently feasible solution. r_1, r_2 and r_3 is a group of random numbers; and $r_1 + r_2 + r_3 = 1$. The attraction point cr is the center of the entire honey group. Each scout bee is centered on it and scaled from far to near according to a certain proportion, which can effectively reduce the probability of individuals flying over the border. In addition, the algorithm's overall individual ability to search has been strengthened, and the overall developmental capability of the algorithm has gradually increased. Therefore, the location of the individual population is updated as

$$x'_j = \begin{cases} (x_j - cr_j)\%((cr_j - x_{\min}) \times r)/r, \text{ if } x_j < cr_j \\ (x_j - cr_j)\%((x_{\max} - cr_j) \times r)/r, \text{ if } x_j \ge cr_j \end{cases} \tag{10}$$

$x_{\max}$ and $x_{\min}$ are the upper limit and lower limit of the group, respectively. At every step of the algorithm's evolution, the value of r is constant to ensure that individual individuals do not lose their original social information.

3.3. Feedback Mechanism and Survival of the Fittest

To further solve the problem that the ABC algorithm easily falls into the local optimum and has slow convergence, this paper integrates the feedback mechanism with random forests. The main idea is to introduce a feedback mechanism in the global search process of the ABC algorithm, which can expand the scope of the algorithm. Therefore, the developmental ability and exploration ability of the algorithm are balanced using an incremental linear differential increment. Survival of the fittest is simulated, and the individuals with poor selection are randomly selected for initialization.

From the employed bee search formula, it can be found that the ABC algorithm is mainly used for searching rather than for development. With the deepening of the algorithm, how to have better development is the weak link in the late stage of the algorithm, especially from the perspective of balancing the algorithm. How to improve the convergence ability of the algorithm is very important for its exploration ability and developmental ability. In this paper, based on the concept of the global optimal solution in the particle swarm optimization algorithm, linear differential increment strategy and the global optimal solution are introduced into the search formula to solve the above problems. The formulas are as follows in (11) and (12):

$$\begin{aligned} & v_{ij} = W(t) \times x_{ij} + r_{ij} \mid x_{kj} - x_{ij} \mid + \phi_{ij}(x_{gj} - x_{ij}) \\ & s.t.\ r_{ij} = \begin{cases} +m \times rand(0,1), d = 1 \\ -m \times rand(0,1), d = 0 \end{cases} \end{aligned} \tag{11}$$

$$\frac{\mathrm{d}w(t)}{\mathrm{d}t} = 2\frac{w_{\max} - w_{\min}}{T^2} \times t \tag{12}$$

x_{gj} is the global optimal solution, x_{ij} is the current solution, x_{kj} is a random solution that currently different from x_{ij}, r_{ij} is a random number in the range of (-1, 1), and $w_{\max}$ and $w_{\min}$ are the maximum value and the minimum value of the self-adaptive factors. Suppose that T is the maximum number of iterations, and t is the current iteration number. In the description of Formula (10), r_{ij} is not a definite random number. Through the corresponding feedback mechanism, the quality of the optimal solution's honey source can be updated using the previous generation of the honey source. This would mean that the current direction is correct, and so it can continue to search in this direction, and vice versa. When the previous generation of honey sources is updated, d =1; otherwise d =0. Through the feedback mechanism, users can directly search for areas where the optimal solution may exist.

4. Construction of Cloud Computing Traffic Prediction Model

4.1. SVM

SVM (Support Vector Machine) is short for a support vector machine, which is a common discrimination method. In the field of machine learning, it is a supervised learning model that is commonly used for pattern recognition, classification, and regression analysis. Using an SVM, a certain load forecasting model can be established in a short period of time. The nonlinear mapping function $\varphi(x)$ is used to map the load sequence $x_1, x_2, \cdots x_n$ in a short time as a sample to the high-dimensional feature space, where the linear regression is performed in high dimension.

The regression function of the SVM in the high-dimensional feature space is as follows.

$$f(x) = \omega \times \varphi(x) + b \tag{13}$$

ω is the weight vector, and b is the offset vector. According to the principle of risk minimization, Formula (13) is transformed into the expression of the following optimization problem.

$$\begin{aligned} & \min J = \frac{1}{2} \| \omega \|^2 + C \sum_{i=1}^{n} (\xi_i^* + \xi_i) \\ & s.t \begin{cases} y_i - \omega \times \varphi(x) - b \le \varepsilon + \xi_i \\ \omega \times \varphi(x) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i^*, \xi_i \ge 0,\ i = 1, 2, \cdots, n \end{cases} \end{aligned} \tag{14}$$

$\|\omega\|$ is the term related to the complexity of function f. ε is the sparse insensitive loss. ζ_i^* and ζ_i represents the relaxation factors, and C represents the penalty factor. The Lagrangian multiplier is introduced, thus turning the problem into a convex quadratic optimization problem:

$$L(w,b,\xi,\alpha,\gamma)=\frac{1}{2}\|w\|^2+C\sum_{i=1}^{n}\xi_i-\sum_{i=1}^{n}a_i(y_i(w^Tx_i+b)-1+\xi_i)-\sum_{i=1}^{n}\gamma\xi_i \tag{15}$$

α_i and α_i^* represent the Lagrangian multiplier, and γ_i represents the loss factor. We convert Formula (15) to the dual form as follows.

$$\omega(\alpha,\alpha^*)=-\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i-\alpha_i^*)\times(\alpha_j-\alpha_j^*)(\varphi(x_i),\varphi(x_j))+\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)y_i-\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)$$

$$s.t\begin{cases}\omega=\sum_{i,j}^{n}(\alpha_i-\alpha_i^*)x_i\\ \sum_{i=1}^{n}(\alpha_i-\alpha_i^*)=0\\ 0\le\alpha_i,\ \alpha_i^*\le C\end{cases} \tag{16}$$

Therefore, the regression function of the SVM is as follows.

$$f(x)=\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)(\varphi(x_i),\varphi(x))+b \tag{17}$$

To further simplify the use of kernel function $K(x_i, x)$ to replace $(\varphi(x_i), \varphi(x))$, the regression function of the SVM is as follows.

$$f(x)=\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)K(x_i,x)+b \tag{18}$$

The radial basis kernel function is used as the SVM kernel function, and the definition is as follows.

$$K(x_i,x)=\exp\left(-\frac{\|x_i-x_j\|^2}{2\sigma^2}\right) \tag{19}$$

In the formula, σ is the radial basis kernel function's width.

It is found from Formulas (20)-(21) that the purpose of establishing a cloud computing load forecasting model is to find the two parameters C and σ of two optimal support vectors, and obtain the corresponding value of the function $f(x)$. Through a large number of experiments, σ determines the advantages and disadvantages of the models that are established by support vector machine to some extent. Therefore, how to further optimize the two parameters C, σ, requires intelligent algorithms for processing.

4.2. Construction of an IABC-SVM Prediction Model

According to the foregoing analysis, the SVM-based resource load prediction helps find the support vector parameters. In this paper, the parameters are improved based on the ABC algorithm, which effectively improves the algorithm's performance. Therefore, the optimized ABC algorithm that optimizes the vector parameters can effectively improve the prediction accuracy.

Step 1 Use the cloud computing resource load sequence to generate the training set and verification set of the SVM.

Step 2 Set the range of the kernel function parameter σ of the SVM and set the modified ABC algorithm parameters.

Step 3 Set the middle honey source of the ABC algorithm to correspond to each support vector parameter σ.

Step 4 The SVM learns the training set through the initial parameter σ.

Step 5 Use an improved method to find the initial solution and conduct the related searches of the three types of bees in the ABC algorithm.

Step 6 When the number of iterations exceeds the maximum number of iterations, the training ends, and the optimal position of the population is output. That is, the optimal value of the vector parameter σ in the SVM is output. Otherwise the process returns to step 5.

Step 7 The prediction model $f(x)$ is established by using the vector parameter σ, and the prediction result of the verification set is analyzed.

5. Experiment and Analysis

This article builds a hardware platform including a Core i3 processor, 4GB of DDR3 memory, and a 1T hard disk. The software platform includes the Linux operating system and the Cloudsim cloud computing simulation platform (which is used for data processing).

5.1. Data Source

To effectively explain the role of the algorithm in cloud computing resource loads, the network load of Clarknet [16] is selected as the research object of this paper. 100 historical data for 7 consecutive days are selected as training data, the data interval is 4 hours, and the load data under the cloud computing for the next 2 days is predicted. In this paper, the average absolute error MAE is used to calculate the validity of the prediction model. The formula is as follows. Here, y_t is the actual loading data, $\hat{y}_t$ is the predicted data, and T is the predicted time sequence. This is shown in the Formula (20). However, since the SVM has a very high sensitivity to the data in the ranger of [0, 1], the network traffic data need to be normalized, as shown in Formula (21).

$$MAE = \frac{1}{T}\sum_{t=1}^{t=T} | y_t - \hat{y}_t | \tag{20}$$

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{21}$$

x is the original network flow. $\max(x)$ and $\min(x)$ are the maximum and minimum values.

5.2. Analysis of the Algorithm's Performance

The load prediction performance based on the intelligent algorithm mainly depends on the performance of the intelligent algorithm, and good performance can effectively improve the accuracy of the prediction. The algorithm of this paper is compared with the algorithm from literature [17]. Both algorithms are optimized using SVMs. Therefore, the performance of the intelligent algorithm affects the optimization to some extent. The Rosenbrock function and Griewank function are selected as the comparison functions. The Rosenbrock function is mainly used to verify the convergence performance of the algorithm, and the Griewank function is mainly used to verify the search effect of the algorithm. The test results of the two functions are shown in Table 1. The maximum number of iterations is 1000, and the dimension ranges from 10 to 50.

5.3. Analysis of the Simulation Results

5.3.1. Prediction Results of the Algorithm in this Paper

This paper selects the simplified cloud computing network traffic result for 9 consecutive days. The unit is Gb/S, and the corresponding results are shown in Table 2. Table 3 shows the results of the cloud computing traffic prediction using the algorithm of this paper to predict the 8th and 9th days. It is found from Table 3 that the error of the cloud computing network traffic prediction based on the same time and the simulation result of the network traffic of Table 1 are substantially within 5% of each other. Figure 2 shows the effect of the actual load and predicted load in the Clarknet platform. Most of the two lines in the figure coincide or are very similar. From the above analysis, we see that the algorithm has good prediction effect.

Table 1. Comparison results of the three algorithms

Function	Dimension	Parameter	GWO-SVM	IABC-SVM
Rosenbrock function	10	Optimal value	1.741E-9	1.38E-8
		Worst value	4.96E-19	3.31E-14
	20	Optimal value	12.23E-15	10.32E-14
		Worst value	13.83E-16	12.73E-15
	50	Optimal value	24.78E-20	17.31E-16
		Worst value	32.67E-21	26.69E-18
Griewank function	10	Optimal value	1.41E+02	1.39E+02
		Worst value	8.41E+06	3.92E+06
	20	Optimal value	18.98E+09	16.83E+08
		Worst value	24.87E+10	22.83E+09
	50	Optimal value	33.78E+13	26.83E+11
		Worst value	45.82E+15	40.98E+13

Table 2. Actual results of the cloud computer network traffic simulation for 9 consecutive days

Time No.	Network traffic	Time No.	Network traffic	Time No.	Network traffic
1	2.8972	12	7.8932	22	7.6731
2	5.5920	13	2.3894	23	8.3291
3	13.9822	14	1.8923	24	5.6732
4	15.8921	15	9.0823	25	9.912
5	10.2871	16	3.8932	26	8.2389
6	7.9832	17	4.3902	27	7.2891
7	9.9023	18	6.3902	28	8.2903
9	15.8922	19	4.9871	29	8.2392
10	8.9230	20	9.3782	31	9.2891
11	5.9832	21	4.8921	31	5.9832

Table 3. Cloud computing network traffic simulation results for days 8-9

Time No	Network Traffic	Time No.	Network Traffic	Time No.	Network Traffic
1	2.9172	12	7.9132	22	7.2931
2	5.6120	13	2.4194	23	7.9391
3	12.1724	14	1.9133	24	5.7332
4	14.2931	15	8.8821	25	9.732
5	11.0821	16	3.9132	26	7.9319
6	8.1831	17	4.5102	27	7.1191
7	9.8027	18	6.4202	28	8.1203
9	16.1923	19	5.2881	29	8.1272
10	9.1431	20	9.7182	31	9.3121
11	6.1332	21	4.9121	31	6.1832

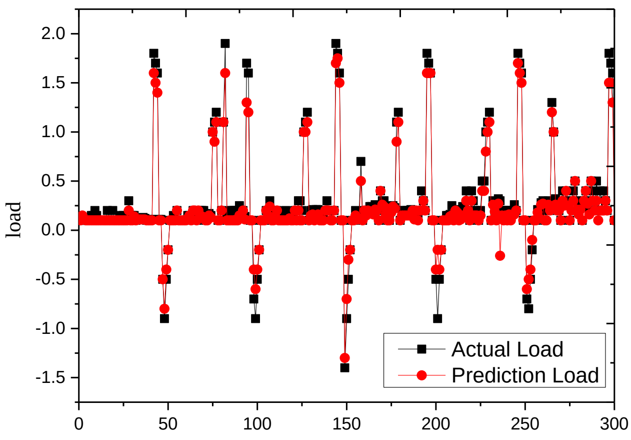

Figure 2. Network load prediction results for the Clarknet platform

5.3.2. Research on Several Prediction Methods using the Clarknet Platform

In the Clarknet platform, the IABC-SVM, SVM method, and IABC-LSSVM [18] are compared with respect to their CPU load and network loads. The comparison results are shown in Figures 3-4.

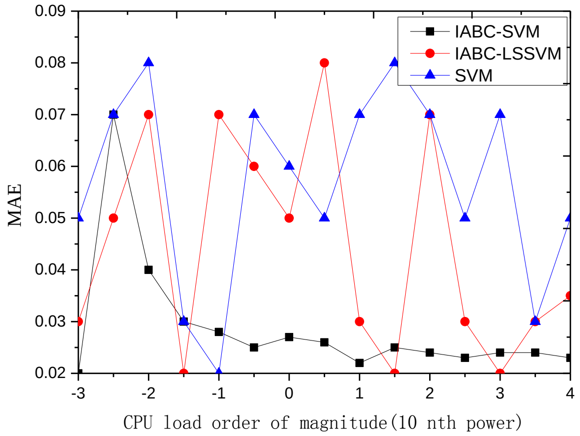

Figure 3. Comparison of the MAEs of the CPU loads of the four algorithms

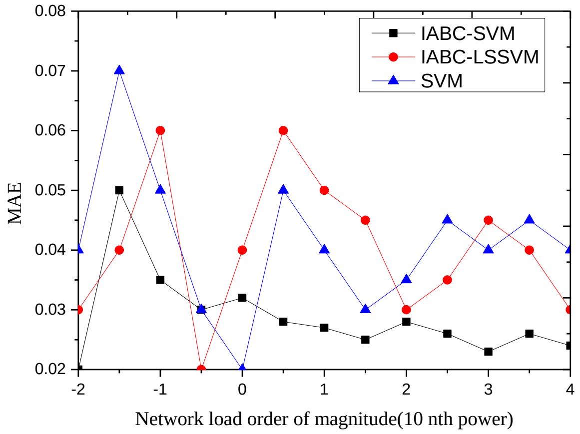

Figure 4. Comparison of MAEs of cloud computing network loads of the four algorithm

The algorithm proposed in this paper has obvious good effects under different cloud computing CPU loads compared with the other two prediction methods. The prediction accuracy is higher and the prediction error is smaller. This is mainly due to the feedback mechanism and survival of the fittest that reduce the possibility that the algorithm falls into local optimum. As a result, the algorithm needs additional CPU resources, resulting in the CPU being overloaded in a short time. As the CPU load increases, the IABC-SVM algorithm curve is gradually stable, while the SVM curve is always in a largely fluctuating state. The IABC-LSSVM curve fluctuates significantly more than the SVM. The IABC-LSSVM is also performed using the ABC algorithm. While there is an improvement, the IABC-SVM still has certain advantages. This is also indicative of the algorithm's improvement. The average MAE of the IABC-SVM algorithm is lower than 0.028, which is lower than the 0.016 for the IABC-LLSVM. Figure 5 shows the MAE values of the three algorithms with respect to the network load. From the figure, the network load curve of the IABC-SVM is relatively flat and does not fluctuate similar to the IABC-SVM and SVM curves. This shows that the IABC-SVM algorithm is relatively stable throughout the process, mainly because the initialization of the population and the choice of the individual improve the performance of the IABC algorithm so that it can handle the increasing network load. The IABC- SVM curve fluctuates more, but is overall better than the SVM. From the MAE analysis, the average MAE of the IABC-SVM is lower than the SVM by 0.013, which is lower than the IABC-LLSVM by 0.012.

From the above two experiments, it is found that the proposed algorithm can effectively improve the search ability of the bee colony in the solution space and improve the global search efficiency, which improves the predictions based on the IABC.

6. Conclusions

To predict the cloud computing resource loads, this paper proposes an ABC-based SVM prediction model for cloud computing. Simulation experiments show that the proposed algorithm can better predict CPU loads and network loads, and can provide a reference for cloud computing resource predictions.

Acknowledgements

This work is supported by the Henan higher education school young backbone teacher training program (2016ggjs-201).

References

1. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility," *Future Generation Computer Systems*, Vol. 25, No. 6, pp. 599-616, 2009
2. C. Reiss, A. Tumanov, G. R. Ganger, and R. H. Katz, "Towards Understanding Heterogeneous Clouds at Scale: Google Trace Analysis," in *Proceedings of the 3rd ACM Symposium on Cloud Computing*, April 2012
3. E. Caron, F. Desprez, and A. Muresan, "Forecasting for Grid and Cloud Computing on-Demand Resources based on Pattern Matching," in *Proceedings of the 2nd IEEE International Conference on Cloud Computing Technology and Science*, pp. 456-463, 2010
4. S. Islam, "Empirical Prediction Models for Adaptive Resource Provisioning in the Cloud," *Future Generation Computer Systems*, Vol. 28, No. 1, pp. 155-162, 2012
5. J. J. Prevost, "Prediction of Cloud Data Center Networks Loads using Stochastic and Neural Models," in *Proceedings of the 6th International Conference on System of Systems Engineering* (*SoSE*), pp. 276-281, 2011
6. A. Khan, "Workload Characterization and Prediction in the Cloud: A Multiple Time Series Approach," in *Proceedings of the 3rd IEEE International Workshop on Cloud Management*, pp. 1287-1294, 2012
7. S. Deng, C. Yuan, L. Yang, and L. P. Zhang, "Distributed Electricity Load Forecasting Model Mining based on Hybrid Gene Expression Programming and Cloud Computing," *Pattern Recognition Letters*, No. 109, pp. 72-80, 2018
8. W. Zhong, Y. Zhuang, J. Sun, and J. J. Gu, "A Load Prediction Model for Cloud Computing using PSO-based Weighted Wavelet Support Vector Machine," *Applied Intelligence*, Vol. 48, No. 11, pp. 4072-4083, 2018
9. S. B. Shaw, C. Kumar, and A. K. Singh, "Use of Time-Series based Forecasting Technique for Balancing Load and Reducing Consumption of Energy in a Cloud Data Center," in *Proceedings of 2017 International Conference on Intelligent Computing and Control* (*I2C2*), pp. 1-6, IEEE, 2017
10. H. Xu and B. Yang, "Energy-Aware Resource Management in Cloud Computing Considering Load Balance," *Journal of Information Science and Engineering*, Vol. 33, No. 1, pp. 1-16, 2017
11. R. Kaur and N. S. Ghumman, "A Load Balancing Algorithm based on Processing Capacities of VMs in Cloud Computing," Big Data Analytics, Springer, Singapore, pp. 63-69, 2018
12. D. G. Li, L. Wu, and L. Li, "Research of Load Forecasting and Elastic Resources Scheduling of Openstack Platform based on Time Series," *Journal of Chongqing University of Posts and Telecommunications*, Vol. 28, No. 4, pp. 560-566, 2016
13. G. P. Zheng and Q. P. Wang, "Allocation Scheme of Virtual Machine Resource Optimization based Load Forecast," *Science Technology and Engineering*, Vol. 16, No. 23, pp. 223-228, 2016
14. D. Wang and Z. Sun, "Big Data Analysis and Parallel Load Forecasting of Electric Power User Side," *Proceedings of the CSEE*, Vol. 35, No. 3, pp. 527-537, 2015
15. A. P. Xu, D. Wu, and W. P. Xu, "Real-Time Multitask Load Balance Algorithm for Heterogeneous Cloud Computing Platforms," *Journal of University of Science and Technology of China*, Vol. 10, No. 3, pp. 215-221, 2016
16. Traces in the Internet Traffic Archive, (http://ita.ee.lbl.gov/html/traces.html, 2015)
17. D. Y. Xu and S. Ding, "Research on Improved GWO-Optimized SVM-based Short-Term Load Prediction for Cloud Computing," *Computer Engineering and Applications*, Vol. 53, No. 7, pp. 68-73, 2017
18. A. J. Xu, "Network Traffic Prediction based on Optimising Lssvm by Improved ABC," *Computer Applications and Software*, Vol. 32, No. 1, pp. 323-326, 2015

Yao Yao is currently an associate professor in the school of information Engineering, Zhengzhou Institute of Technology. She received her Masters of computer application engineering degree from the department of information engineering, University of Zhengzhou, China in 2008. Her research interests include high-performance computing and Web mining.