Send Orders for Reprints to reprints@benthamscience.ae

1334 *The Open Cybernetics & Systemics Journal,* 2015, *9,* 1334-1340

Open Access

Research on Cloud Computing Load Balancing Based on Virtual Machine Migration

Liu Kun[1,*], Xu Gaochao[1] and Chen Jingxia[2]

[1]*College of Computer Science and Technology, Jilin University, Jilin, Changchun, 130012, P.R. China;* [2]*College of Information Science & Technology, Beijing University of Chemical Technology, Beijing, 10029, P.R. China*

Abstract: This article is designed to solve the problem of system load imbalance and low efficiency introduced by massive parallel tasks running on some heavy nodes in the Cloud Computing environment. This study proposes a load balancing algorithm based on virtual machine migration which includes load balancing architecture, load gathering, load monitoring, load forecasting, migration trigger time, source machine selecting and etc. Experimental results based on the CloudSim tool reveal that the algorithm can guarantee load balancing and improve the performance of the system.

Keywords: Cloud computing, load balancing, migration, virtual machine.

1. INTRODUCTION

Cloud computing emerges as a new computing and business model. Users can use the cloud computing platform for temporary operation instead of purchasing expensive personal computers, severs or physical devices. Enterprises also only need to buy cheap, easily extended cloud platform instead of building the huge virtual server clusters. Clusters' load will become unbalancing for the users' various or dynamic demand and servers' heterogeneous resources. This problem directly affects the clusters' resources efficiency and brings about the waste of a large number of cloud resources. So how to improve the cloud data center's performance by applying suitable scheduling algorithm is an important problem of the Cloud Computing. Whether the performance of the load balancing algorithm is good or bad is very important to the whole cluster [1]. Therefore, the evaluation of a load balancing algorithm must pay attention to both the algorithm itself and the occasion of use of the algorithm.

Scholars at home and abroad have done a lot of research on the load balancing problem of cloud computing. In paper [2], the author balanced each node according to the next moment load which is computed by the predicted load based on Error Back Propagation Training(BP) algorithm. In paper [3], the author judged the next moment load on the basis of predicted load computed by the simulated annealing algorithm. In paper [4], the author proposed a weighted sequence dynamic algorithm. In paper [5], the author discussed the additional overload produced by the balancing procedure and proposed a particle size formula to avoid additional overhead. Bonomi [6] adjusted each server's load after predicting the load at the next moment.

This paper presents a dynamic load balancing algorithm based on resource scheduling, which mainly includes the following three aspects of work: proposing a load balancing architecture, researching on load balancing trigger time, and researching on the selection of load balancing source machine.

2. DEFINITION AND ARCHITECTURE FOR LOAD BALANCING

2.1. Definition

Definition 1: The number of vitual machine on node i is n_i.

Definition 2: The CPU utilization rate of node i is CUR_i, $CUR_i = \frac{\sum_{j=1}^{n_i} C_j}{n_i}$. C_j is the CPU utilization rate of virtual j.

Definition 3: The memory utilization rate of node i is MUR$_i$, $MUR_i = \frac{\sum_{j=1}^{n_i} M_j}{MT_i}$. M_j is the memory use of virtual j. MT_i is the total memory amount of node i.

Definition 4: The bandwidth utilization rate of node i is NBUR$_i$, $NBUR_i = \frac{\sum_{j=1}^{n_i} NB_j}{NBT_i}$. NB_j is the bandwidth use of virtual j. NBT_i is the total bandwidth amount of node i.

Definition 5: The average CPU utilization of the cluster is CA: $CA = \frac{\sum_{i=1}^{m} CUR_i}{m}$. Letter m denotes the number of nodes in the cluster.

1874-110X/15 2015 Bentham Open

Definition 6: The average memory utilization of the cluster is MA: $MA = \frac{\sum_{i=1}^{m} MUR_i}{m}$. Letter m denotes the number of nodes in the cluster.

Definition 7: The average bandwidth utilization of the cluster is NBA: $NBA = \frac{\sum_{i=1}^{m} NBUR_i}{m}$. Letter m denotes the number of nodes in the cluster.

Definition 8: High threshold(H_{th}) is a value. If the load of the node exceeds this value, the node is called overload node.

Definition 9: Adaptive threshold(θ_{th}) is a value which judge the overload node combined with H_{th}.

Definition 10: Overload Matrix is expressed as high.

$$high = \begin{bmatrix} CUR_1 & MUR_1 & NBUR_1 & RL_1 & NUM_1 \\ CUR_2 & MUR_2 & NBUR_2 & RL_2 & NUM_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ CUR_n & MUR_n & NBUR_n & RL_n & NUM_n \end{bmatrix}$$

Values of each row are the parameter values of each node. The first three columns represent CUR, MUR and NBUR. The fourth column RL_i of 0 indicates that CUR_i value is high, while the value of 1 indicates that MUR_i value is high, and the value of 2 indicates $NBUR_i$ value is high. The last column NUM_i represent the node's number.

2.2. Architecture

A dynamic migration algorithm must consider [7] three problems: when the dynamic migration is executed, which virtual machine will be migrated, and where does the virtual machine migrate to. This study propose a load balancing model architecture diagram, as shown in Fig. (**1**).

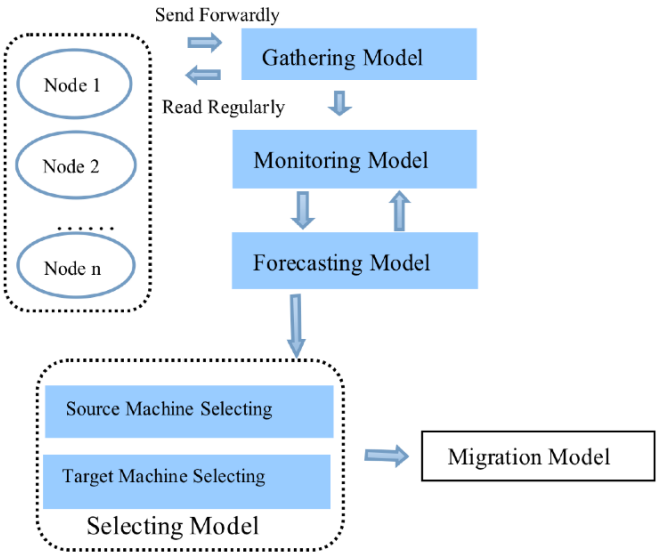

Fig. (1). Architecture diagram.

Gathering model gathers the load of each node and stores the load data to database. Monitoring model decides whether trigger the migration. Forecasting model forecasts future load value and the migration will not be triggered when the load value is a instantaneous peak value. Selecting mode includes source machine selecting and target machine selecting. Migration model executes virtual machine migration.

3. LOAD BALANCING ALGORITHM

3.1. Gathering Model and Monitoring Model

The central node collects CUR_i, MUR_i, $NBUR_i$ of each node regularly. If the interval time is too short, the central node will be too busy, and data transmission will occupy the bandwidth. If the interval time is too long, the heavy load nodes will not be handled timely while those low load nodes will be handled. Most papers set the interval time as 10 to 20 seconds. In this paper, the central read each node every 15 seconds and the node whose load value fluctuate more than 10% send its load to the central.

Monitoring model analyzes the load value received from the gathering module and determines which node will be balanced. Efficient prerequisite for virtual machine migration is to formulate a reasonable trigger conditions (threshold). We use parameters Hth and θth to determine whether a node is overload. The algorithm is shown in Fig. (**2**).

3.2. Forecasting Model

Virtual machine will be migrated when the load exceeds the threshold in traditional load balancing algorithms. In fact most nodes have instantaneous load peak value and the load value will return to normal after this period. For traditional algorithm, the instantaneous peak value will trigger load balancing, which will cause unnecessary migration. At present, some scholars have researched on the prediction on physical hosts and found that load changes has self similarity and long-range dependence [8]. In this paper, the forecasting module based on exponential smoothing method is designed to predict the load value at the next moment, and then decide whether to trigger migration.

$y'_{t+1} = \alpha * y_t + (1-\alpha) * y'_t$ is the formula of exponential smoothing method. y'_{t+1} is the prediction value at moment t+1, y_t is the actual value at moment t, y'_t is the prediction value at moment t and α is the smooth coefficient. $\alpha \in [0,1]$. When a node's load exceeds threshold in monitoring model, the first d-1(d is an uncertain number) moment data of this node are passed to forecasting model to get p(p is an uncertain number) forecasting values. If s(s is an uncertain number) values exceed threshold, the load balancing will be triggered.

3.3 Source Machine Selection

Most authors use weight to calculate the total value, such as w1*CURi+w2*MURi+w3*NBURi, w1+w2+w3=1. In this formula, the factors that have the heavy weight will more affect the final total value.

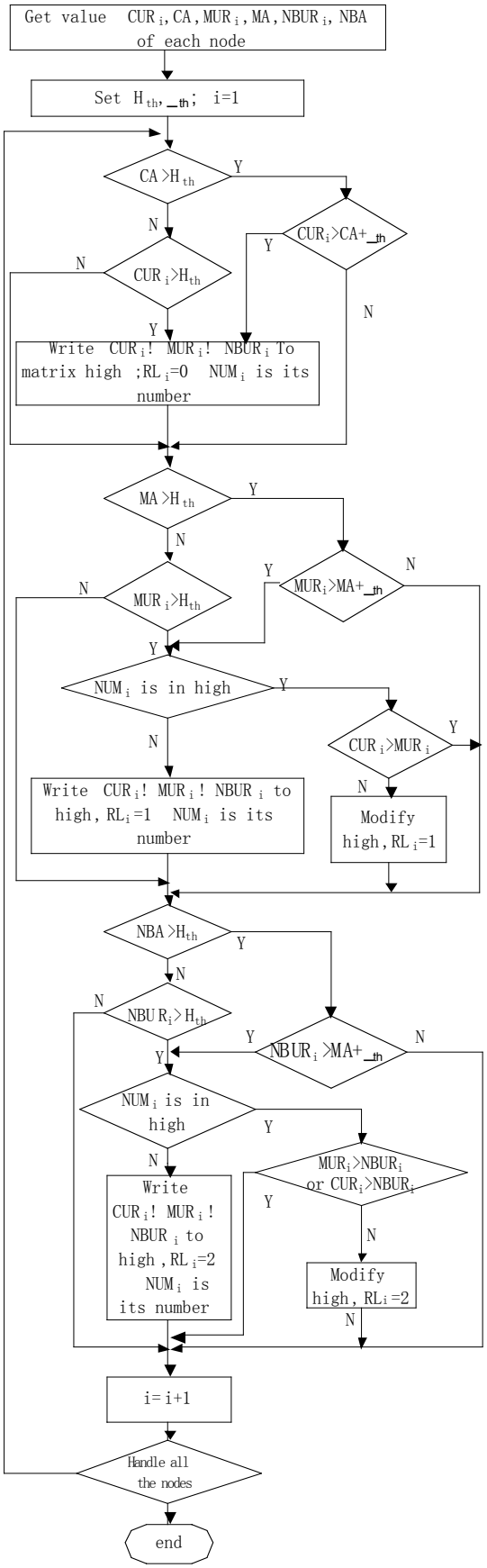

Fig. (2). Flow chart of computing matrix high.

For example, there are two nodes whose CUR, MUR, NBUR are respectively <0.9,0.3,0.2> and <0.5,0.5,0.2>. It is obviously that the first node has great CPU load and the second load value isn't high. So the first node must be first to be balanced. If the three weights value are w1=0.2, w2=0.5, w3=0.3, total load of the two nodes are 0.39, 0.6, so the second node will be balanced. This shows that the choice of different weight values will affect the balancing algorithm. In this paper, the information entropy is used to determine the weight value.

The basic ideas of entropy method is to determine the weight objectively according to the variability. Generally speaking, the information entropy value being smaller shows that the degree of variability is much greater, the amount of information provided is much more, and so the weight is great. On the contrary, the information entropy value being much greater shows that the degree of variability is much smaller, the amount of information provided is much smaller, and so the weight is small. Total load computing methods are given below:

(1) The first three columns of overload matrix high are chosen from monitoring model to form a new matrix D, $D=\begin{bmatrix} CUR_1 & MUR_1 & NBUR_1 \\ CUR_2 & MUR_2 & NBUR_2 \\ \cdots & \cdots & \cdots \\ CUR_n & MUR_n & NBUR_n \end{bmatrix}$.

(2) Matrix D is standardized to R, $R=\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ \cdots & \cdots & \cdots \\ r_{n1} & r_{n2} & r_{n3} \end{bmatrix}$,

$r_{i1}=\frac{CUR_i}{\sum_{i=1}^{n} CUR_i}$, $r_{i2}=\frac{MUR_i}{\sum_{i=1}^{n} MUR_i}$, $r_{i3}=\frac{NBUR_i}{\sum_{i=1}^{n} NBUR_i}$.

(3) The entropy is computed as $E_j=-\frac{1}{\ln n}\sum_{i=1}^{n} r_{ij}\ln r_{ij}$, j=1, 2, 3.

(4) F_j is set as $F_j=1-E_j$.

(5) Each weight is set as $w_1=\frac{F_1}{F_1+F_2+F_3}$, $w_2=\frac{F_2}{F_1+F_2+F_3}$, $w_3=\frac{F_3}{F_1+F_2+F_3}$.

(6) Total load of node i is $Load_i=w_1*CUR_i+w_2*MUR_i+w_3*NBUR_i$.

(7) According to the $Load_i$ in descending order, a queue is formed as $Q=\{q_1, q_2, \ldots\ldots, q_n\}$. Each element has two values as node number and flag RL_i, that is $q_j=(RL_k, NUM_k)$.

(8) The first machine in the queue Q is selected as the source machine. The virtual whose CUR_m/MUR_m is the biggest on the machine is selected as the migrated virtual because memory migration is the most difficult.

Table 1. Expermental configuration table.

Parameter Name	Unit	Value
Physical machine number		50
CPU processing capacity of each physical machine	MIPS	{1000,1800,2600,3000}
Physical machine memory value	G	{1 2 4 8}
Physical machine bandwidth value	Mb/S	{500,700,1000}
Virtual machine number		200
CPU processing capacity of each virtual machine	MIPS	{200,500,1000,1500,2500}
Virtual machine memory value	G	{0.5, 1, 2, 3}
Virtual machine bandwidth value	Mb/S	{100, 200, 500}

4. EXPERIMENTS

We choose CloudSim [9, 10] which developed by Ruyya *et al.* from Melbourne University as the simulation tool. The experiments selected 50 heterogeneous physical hosts, and 3 to 5 virtual machines were respectively arranged on each host. The configuration as shown in Table **1**.

In order to verify whether the forecast module is effective, we have done many experiments on the CPU utilization rate. It is similar as memory utilization rate and bandwidth utilization rate. The parameters value are shown in Table **2**. The value α is set as 0.7.

Table 2. Parameters value table.

Parameters Name	Value
H_{th}	70%
θ_{th}	10%
Forcast period(p)	5
Judging number(s)	4
Dataset Size(d)	50

The experimental results are shown in Fig. (**3**), where vertical axis represents the CPU utilization rate, horizontal axis represents the number of monitoring data points, the dotted line with a circular indicates the actual value of CPU utilization rate, the solid line with a star indicates the predictive values, and H_{th} value 0.7.

(1) The actual values exceed threshold at the moment t=4 and t=35. The forecasting value doesn't exceed threshold, So the balancing will not triggered.

(2) The actual values exceed threshold at moment t=16 to t=20, and the forecasting values exceed threshold at moment t=17 to t=21. There are four values exceed threshold in the five values, so the balancing will be triggered. After balancing, the load value drops at moment t=21.

The balancing will be triggered when the actual values exceed threshold if not steeped in forecast algorithm, for example, at moment t=4, t=27, t=28 or t=35. Using the forecasting algorithm effectively avoids the balancing be triggered by instantaneous peak value.

From Figs. (**4-9**), we can find the distribution of CUR, MUR and NBUR is very uneven. After several rounds of load balancing, the three values are obviously be balanced.

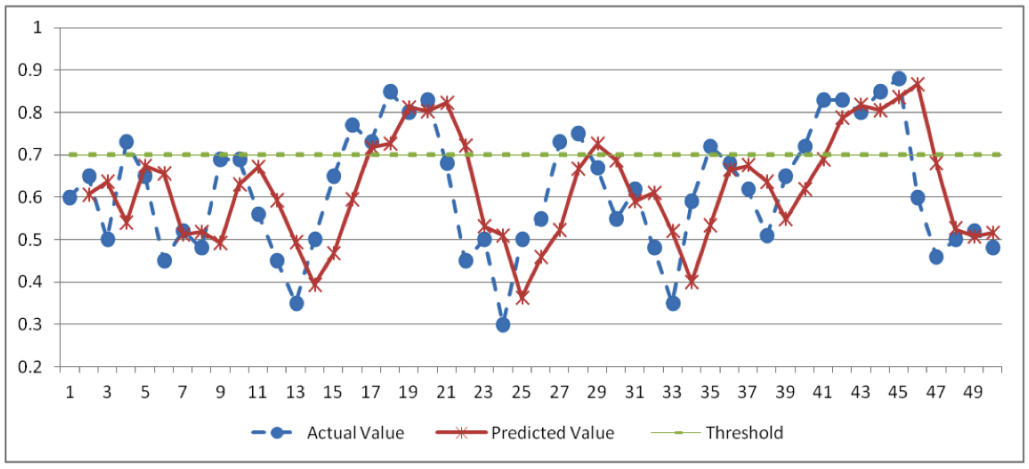

Fig. (3). Chart of forecasting.

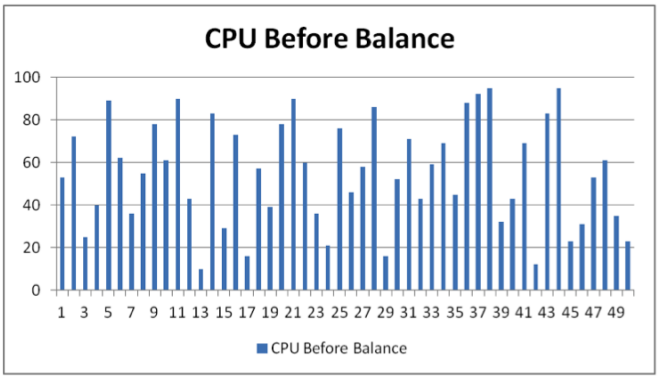

Fig. (4). Chart of CUR before balancing.

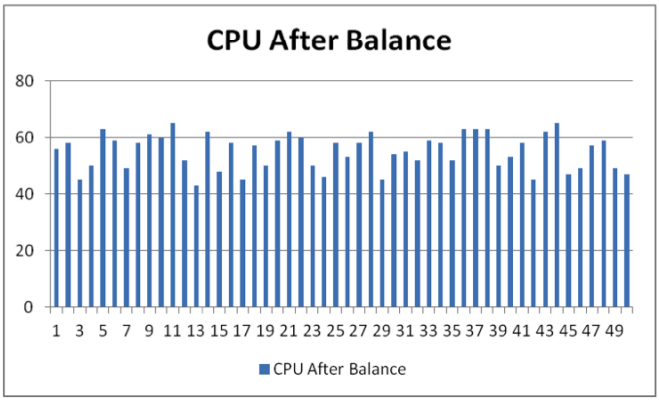

Fig. (5). Chart of CUR after balancing.

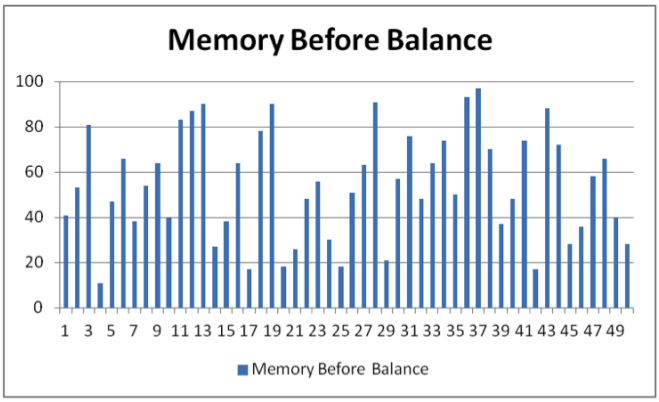

Fig. (6). Chart of MUR before balancing.

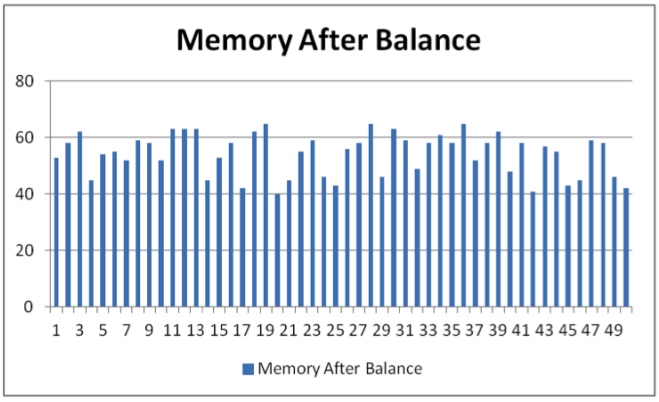

Fig. (7). Chart of MUR after balancing.

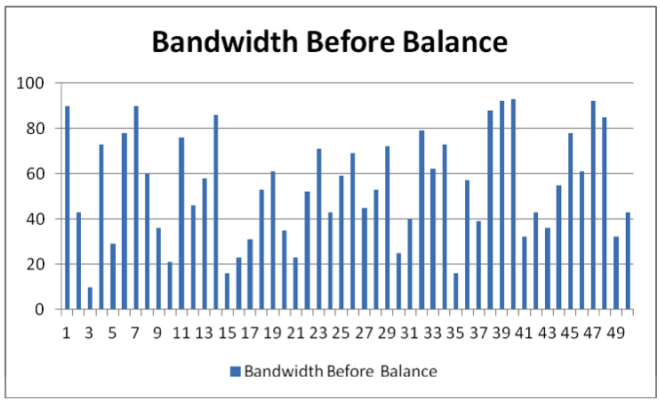

Fig. (8). Chart of NBUR before balancing.

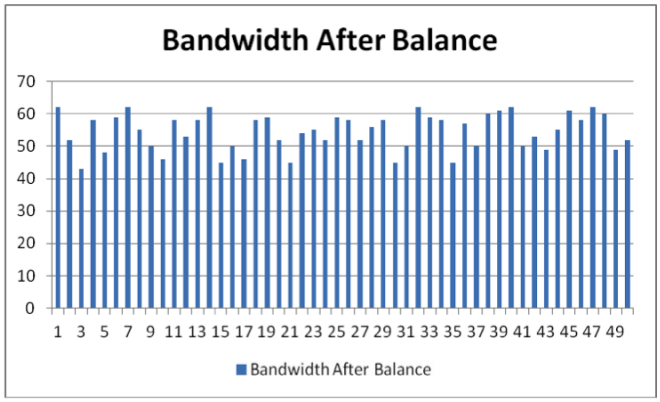

Fig. (9). Chart of NBUR after balancing.

Lots of experiments on the algorithm in my paper and the algorithm of certain weight(Hereinafter referred to as the algorithm A) have been done. In algorithm A, the load formula is defined as $Load_i = w_1 * CUR_i + w_2 * MUR_i + w_3 * NBUR_i$ and the values of w_1, w_2, w_3 are 0.7,0.15, 0.15, namely the CPU utilization rate accounts for the maximum weight. In Fig. (**10**) to 12, horizontal axis represents each physical machine, and vertical axis represents the utilization rate deviation which is the difference between the actual value and the average value. Small deviation suggests the system is more balance.

From Figs. (**10-12**), CUR of algorithm A is slightly better than this paper's algorithm, while MUR and NBUR of this paper's algorithm are more better than algorithm A's. The weight of CPU utilization rate is higher than the other two in algorithm A, so the result on CPU utilization is better in algorithm A. Although this paper's algorithm is worse than algorithm A on CUR experiment, this paper's algorithm makes the overall performance on CUR, MUR and NBUR much better.

5. CONCLUSION

This study proposed a virtual migration algorithm to balance the cluster's load including gathering model, monitoring model, forecasting model, selecting model, migration model. A load judging matrix based on high threshold and adaptive threshold was designed. The exponential smoothing method was introduced to avoid instantaneous peak value,

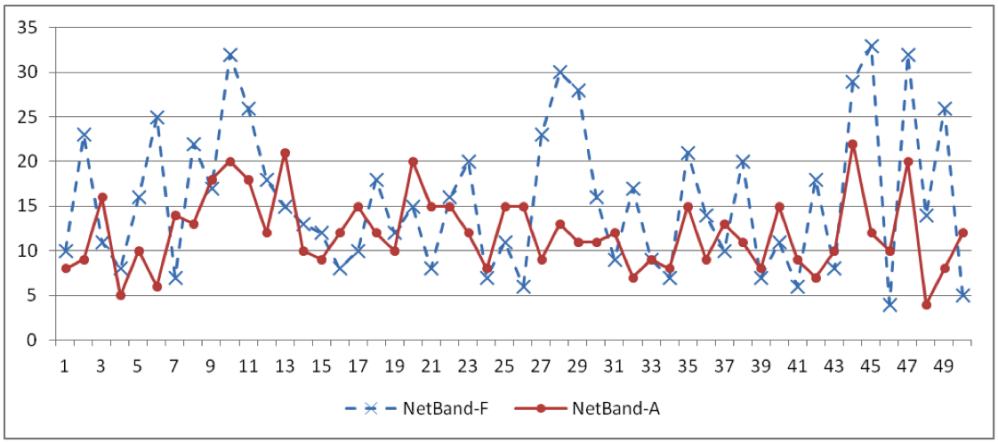

Fig. (10). Deviation of NBUR.

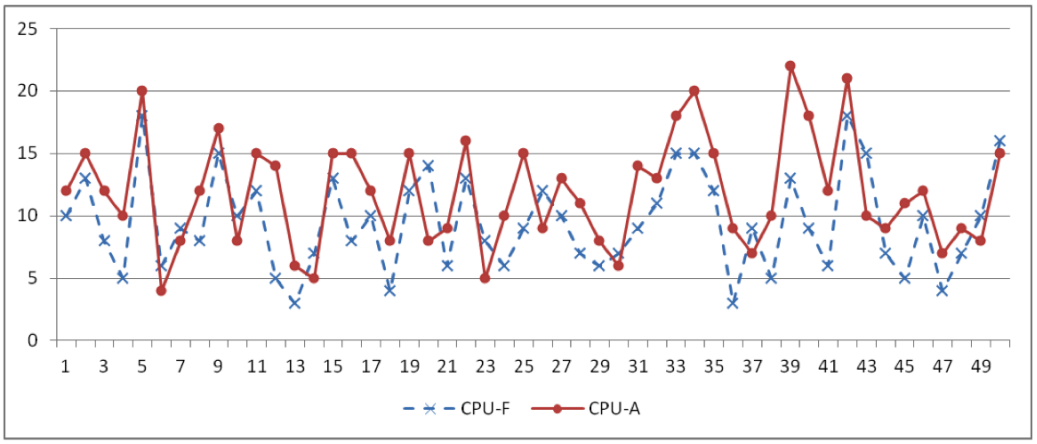

Fig. (11). Deviation of CUR.

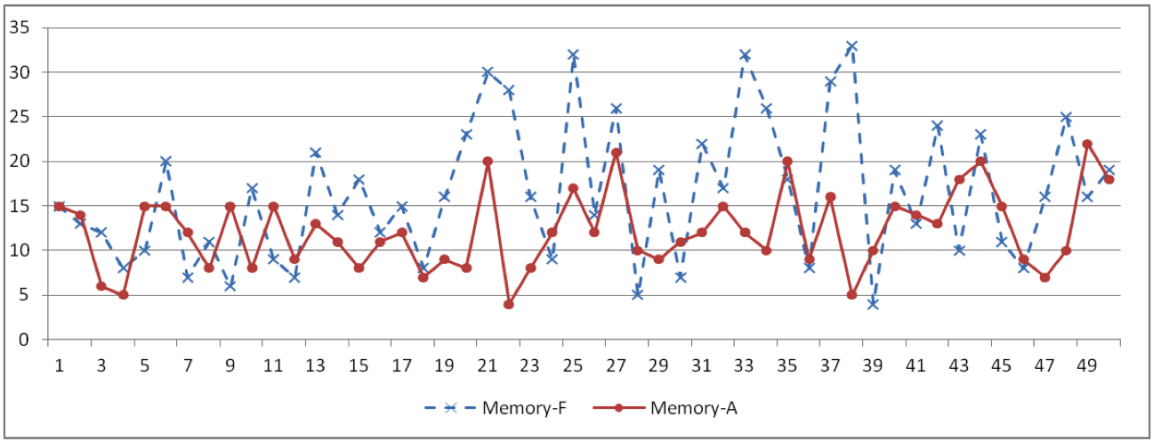

Fig. (12). Deviation of MUR.

and the source machine selecting algorithm based on information entropy was proposed to solve the problem of unreasonable weight value. Finally, we carried out multiple experiments and proved he effectiveness of the algorithm in this study.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

The study is subsidized by Funding Project of Competence Development Program for Beijing VET Teachers and Beijing City Board of Education Science and Technology Project (KM201211417008, KM201511417010) and Beijing Higher Education YoungElite Teacher Project (YETP1767).

REFERENCES

[1] S. Zhu, "A new way for dynamic load balancing algorithm design," *Computer Engineering and Design*, vol. 16, no. 3, pp. 25-30, 1994.

[2] S. Zhang, *Dynamic BP Algorithm Based on Load Balance Forecast*, Henan:Henan University, 2009.

[3] H. Yi, "*Research on Network Load Balancing Algorithm Based on Simulated Annealing Genetic Algorithm*," WuHan: WuHan University of Technology, 2013.

[4] Y. Hu, and Y. C. Ou, "A resource load balacing method to reduce the energy comsumption in the cloud environment," *Computer Engineering*, vol. 38, no. 5, pp. 53-55, 2012.

[5] D. Li, and H. Shi, "Study on general model of load balancing scheduling problems," *Computer Engineering and Application*, vol. 43, no. 8, pp. 121-125, 2007.

[6] R. Bonomi, P. J. Fleming, and P. Steinberg, "An adaptive join the biased queue rule for load sharing on distributed computer system," In: *Proceedings of the 28th Conference On Decision and Control, Tampa,* Fla, 1989, pp. 2554-2559.

[7] K. Yang, J. Gu, T. Zhao, and G. Sun, "An optimized control strategy for load balancing based on live migration of virtual machinef," *2011 6th Annual China Grid Conference*, 2013, pp. 141-146.

[8] P. A. Dinda, "The statistical properties of host loa," *Scientific Programming*, vol. 7, no. 3-4, pp. 211-229, 1999.

[9] R.N. Calheiros, R. Ranjans, and A. Beloglazov, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", *Software: Practice and Experience*, vol. 41, no.1, pp. 23-50, 2011.

[10] SimCloud Platform, http://simcloud.com/

Received: June 10, 2015 Revised: July 29, 2015 Accepted: August 15, 2015