Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 510656, 7 pages
http://dx.doi.org/10.1155/2014/510656

Research Article

Research and Application on a Novel Clustering Algorithm of Quantum Optimization in Server Load Balancing

Dong Yumin,[1] Xiao Shufen,[2] Jia Fanghua,[3] and Li Jinhai[1]

[1] *Network Center, Qingdao Technological University, Qingdao, China*
[2] *Administration Office of National Assets, Qingdao Technological University, Qingdao, China*
[3] *Library, Qingdao Technological University, Qingdao, China*

Correspondence should be addressed to Dong Yumin; dym1188@163.com

Received 16 December 2013; Accepted 3 February 2014; Published 6 April 2014

Academic Editor: Balaji Raghavan

A quantum optimization scheme in network cluster server task scheduling is proposed. We explore and research the distribution theory of energy field in quantum mechanics; specially, we apply it to data clustering. We compare the quantum optimization method with genetic algorithm (GA), ant colony optimization (ACO), simulated annealing algorithm (SAA). At the same time, we prove its validity and rationality by analog simulation and experiment.

1. Introduction

Cluster technology is connecting multiple independent servers and providing services as a whole by a cluster. To achieve parallel program in a high efficiency, service request must be allocated to each server, reduce the access time, and optimize the overall performance. Load balancing mechanism is the core cluster technology.

In the literature [1], server cluster provides high reliability, availability, and scalability by gathering server nodes into one group. User requests need to be distributed to every server node fairly to maximize the characteristic of server cluster. In the same time, it proposes an efficient and adaptive load balancing arithmetic for the server cluster. The arithmetic computes the load of servers with the usages of computer resources or their weights. The weights are dynamically determined based on the usages of the statistics. Their experimental result shows that this arithmetic protect the bottleneck from server cluster efficiently compared with before arithmetics.

The state of web applications communicates and coordinates with lot of geographically distributed information resources offering information to great number of clients. Homogeneous server clusters are unable to meet the growing demand of the applications including real-time video and audio, ASP, JSP, and PHP. Moreover, it also provides better reliability by gracefully transferring the load from server which is unavailable due to failure or for preventive maintenance. Heterogeneity with scalability makes the system more complex. The literature [2] proposes a dynamic load balancing (DLB) algorithm for extensible heterogeneous server cluster for content awareness. The arithmetic considers server's processing ability, queue length, utilizing ratio, and so forth, as load indices. As the clusters supports multiplex services, at the basic level, it has used content awareness forwarding arithmetic.

In the literature [3], online games are becoming more fashionable recently as the internet becomes popular, game platforms become different, and ubiquitous environment is supported. Wherefore, distributed technology is required to support huge numbers of concurrent game clients simultaneously. Specifically, when the users are playing games, a lot of unpredictable problems can arise, for example, a certain server handles more loads than what is recommended because much more game users crowded into a specific region of game world. The kinds of situations can lead to the game servers instability. Here, the global dynamic load balancing model and distributed massive multiplayer online game (MMOG) servers architecture are put forward to apply the load balancing arithmetic. Much more different experiments were achieved to test efficiency.

A load balancing arithmetic named dynamic weighed random (DWR) algorithm for the session initiation protocol (SIP) application server cluster is put forward in the literature [4]. It utilizes weighted hashing random arithmetic that supports dialog in the SIP protocol of distributing messages. The weight of every server is dynamic and adaptive with feedback mechanism. The arithmetic of DWR is efficient in the cluster balance, and it is much better than the limited resource vector (LRV) arithmetic and the minimum sessions first (MSF) arithmetic.

The literature [5] proposes a new server load balancing model. Server cluster load balancing is well known to be a critical mechanism for network-based information service. Most of previous schemes cannot take server's loadings into account, which might not make the loadings of all servers be balanced and drive the server system to work on the borderline of being overloaded and/or out of function. The proposed model is aimed at preventing the occurrence from malfunction and saving the power consumption of the cluster system under a low loading, when it provided a better performance. All of the connection requests are truly distributed into one server until a prespecified portion of the maximum allowed serving load is realized. The following requisitions are served by another one server in the same method. These experimental results illustrate the feasibility of the proposed model.

The consolidation of server is due to virtualization technology, which enables multiple servers to run on one platform. Moreover, virtualization may bring the overheads on performance. The prediction of virtualization performance is very important. The literature [6] proposes a general model for predicting the performance of consolidation. On the other hand, a load balancing problem is studied that arises in server consolidation. A certain amount of workloads are assigned to a few number of high performance target servers, and the workloads in every target server are balancing. It is as an integer linear programming that first models the load balancing problem. The fully polynomial time approximate scheme (FPTAS) is proposed to get the approximate optimal solution.

The response time of a website needs to be improved, which one replicates the site on multiple servers. It will depend on how the incoming requests are distributed among replicas, where the effectiveness of a replicated server system. In the literature [7], a testbed is described that can evaluate the performance of many different load-balancing strategies. A general architecture that allows different load-balancing methods to be supported easily is used in the testbed. It emulates a typical internet scenario and allows variable load production and performance measurement. They measure and illuminate the performance of some policies for load balancing in this testbed by some basic experiments.

A key issue for cluster system is the utilizing efficiency of system resources, in which the method of load balance is very important to realize the efficient resources. Based on server cluster system, the literature [8] proposes an improved self-adaptive arithmetic for network load balancing. The arithmetic can improve the utilizing efficiency of system resource by showing simulation results. In order to achieve the request of real time, when dealing with tasks and high availability of system, it can reduce the server's response time.

Effective load balancing mechanism can extend the "capacity" of the server and improve system throughput. In early studies of load balancing algorithm, genetic algorithm (GA), dynamic feedback algorithm (DFA), ant colony algorithm (ACO), simulated annealing algorithm (SAA), round robin (RR), Min-Min algorithm, Max-Min algorithm, and so on have some improvements in different degree at different perspective on the load balancing system.

They provide solutions to the problem of load balancing for server cluster by these arithmetics mentioned above. But these arithmetics have this or that problem such as local premature problem and divergence problem.

In order to overcome the instability of the above algorithms, the server load balancing method based on quantum algorithm is proposed. And we prove it better than GA, ACO, and SAA by simulation experiments.

2. Quantum Optimization Algorithm

The quantum optimization method of clustering algorithm is mainly used in this particle. The method is put forward by clustering idea based on quantum theory, which is a kind of unsupervised clustering method. It is also applied to the traditional clustering algorithm, by the study of the theory of energy distribution in quantum mechanics, theory study, we found that the microscopic particles distribution in the energy field relies on the potential energy which associates with particles themselves, the smaller the potential energy around the particles, the more they are absorbed. In the energy field with particle distribution described by a wave function, the particle distribution will ultimately depend on the potential energy in the energy field. For the design of clustering algorithm, the potential energy function is used and the cluster center is determined by the particle distribution. Similarly, how to determine the cluster center and the corresponding number of samples of clusters is also the main task of cluster analysis. Therefore, distribution of particles in space studied by quantum mechanics theory is similar to the distribution of samples studied by clustering algorithm. The known clustering process of sample distribution can be regarded as the known wave function which describes the particle distribution.

The clustering process can be expressed as follows: with the known wave function, solve the potential energy function by the Schrödinger equation. Particle distribution ultimately depends on the value of the potential energy function.

Quantum state of a particle wave function is as follows:

$$H\varphi = \left(-\frac{\delta^2}{2}\nabla^2 + V(x) \right)\varphi = E\varphi, \tag{1}$$

where φ is the wave function to describe the particle quantum state, H is the Hamilton operator that describes the system's total energy, V is the potential energy function on behalf of potential energy of the particle, E is the H operator's energy proper value, ∇ is the Nabla operator, and δ is the parameter.

From formula (1), one can find that, with the same potential field distribution, determination of cluster centers is similar to the principle of quantum changing with the potential energy. To solve the particle distribution of potential energy function, the particle with minimum potential energy is determined. As the focal point for cluster, through (1), the potential energy function is shown as follows:

$$V(x) = E + \frac{\left(\sigma^2/2\right)\nabla^2\varphi}{\varphi}. \tag{2}$$

3. The Model of Server Task Scheduling

In cluster services, the loading balance can be described as follows: N tasks need to be allocated to m node servers with different loading and handling capacity for processing, in order to find an optimization schedule to minimize the total completion time. Mathematical model for the system is shown as follows.

Suppose there are m servers (or nodes) and n tasks. Each task has to be assigned to only one server. In this paper, $P = \{p_1, p_2, \ldots, p_m\}$ denotes the servers, whereas p_i denotes one of the servers or nodes; $L = \{l_1, l_2, \ldots, l_m\}$ denotes the current load, whereas l_i denotes the current load of node p_i. For example, $l_i = 0$ means that node p_i has a current load of 0; that is to say, this node is idle. The n tasks are denoted by $X = \{x_1, x_2, \ldots, x_n\}$, where x_j is one of the tasks. We build an $m \times n$ matrix between servers and tasks: W_{mn}, where W_{ij} is one of the elements and there are two states:

$$W_{ij} = \begin{cases} 1, & \text{Task } x_i \text{ is processed on node } p_i, \\ 0, & \text{Task } x_i \text{ is not processed on node } p_i, \end{cases} \tag{3}$$

where $i \in \{1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, n\}$.

We use t_{ij} to denote the time of processing on one task; that is to say, the time of task x_j is processed on node p_i. We use the processing time by

$$T_{ij} = \begin{cases} t_{ij}, & \text{Task } x_i \text{ is processed on node } p_i, \\ 0, & \text{Task } x_i \text{ is not processed on node } p_i, \end{cases} \tag{4}$$

where $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, n\}$.

Obviously, T_{ij} is also an $m \times n$ matrix.

We consider that the optimal state occurs with these conditions: (1) the whole system has a relative short time of processing, and meanwhile (2) the throughput of system in unite time is relatively large. We can use the following equations to describe this state:

$$\begin{aligned} Y_{\max} &= \sum_{i}^{m} \omega\left(x_i, l_i, q_i\right), \\ \omega\left(x_i, l_i, q_i\right) &= c_1\left(\left(\sum_{i=1}^{m}\sum_{j=1}^{n} t_{ij} + \sum_{i=1}^{m} q_i t_i\right) - c_2 \sum_{i=1}^{m} l_i\right)^2, \end{aligned} \tag{5}$$

where $X = \{x_1, x_2, \ldots, x_n\}$ is the new task, $L = \{l_1, l_2, \ldots, l_m\}$ is the current whole load at the node, q_i is the length of ready queue at node p_i, t_i is the average processing time at node p_i, c_1 and c_2 are constant, and $\omega(x_i, l_i, q_i)$ is a function which can reflect the ability of node processing and required tasks. When the processing tasks and loading capacity at the node reach the maximal matching, the system is on the optimal running state.

4. Server Loading Balance Model Based on Quantum Optimization Algorithm (QOA)

4.1. The Model of Quantum Cluster Algorithm. The model of loading balance by quantum optimization can be described as a tetrad (task, weight, function, and scheduling); among them, task is the task to be assigned and scheduling is to assign the task according to the rules [9–12]. In quantum model, the task and weight are denoted by qubit $|a_j\rangle$ and $|w_j\rangle$, respectively. A qubit state can be expressed as (6). Consider

$$\left|\varphi_j\right\rangle = \alpha_j |0\rangle + \beta_j |1\rangle, \quad (j = 1, 2, \ldots, k), \tag{6}$$

where α_j and β_j satisfy the condition

$$\left|\alpha_j\right|^2 + \left|\beta_j\right|^2 = 1, \quad (j = 1, 2, \ldots, k). \tag{7}$$

Model establishing: there are k and p qubits in task and scheduling, respectively,

$$\begin{aligned} &\text{The task vector is } |A\rangle = \left|a_1\right\rangle, \left|a_2\right\rangle, \ldots, \left|a_k\right\rangle \\ &\text{And the weight: } |W\rangle = \left[\left|w_1\right\rangle, \left|w_2\right\rangle, \ldots, \left|w_k\right\rangle\right]^T. \end{aligned} \tag{8}$$

The relation of task and schedule is described as

$$b_i = f\left(\left\langle W_j \mid A\right\rangle\right) = \left|\sum_{j=1}^{k} \left\langle w_{ij} \mid a_j\right\rangle\right|, \tag{9}$$

where $i = 1, 2, \ldots, p$.

Suppose $\{A^1, A^2, \ldots, A^m\}$, $A^n = (A_1^n, A_2^n, \ldots, A_k^n)$, $(n = 1, 2, \ldots, m)$ is the set of cluster samples. Each sample of them belongs to one of the set modes according to some rule. And we use P_i $(i = 1, 2, \ldots, q)$ to denote set of samples in each type of mode and q_i $(i = 1, 2, \ldots, q)$ to denote the competition superiors corresponding to the mode in C. Quantum state of cluster samples is as follows.

For cluster samples $A = (A_1, A_2, \ldots, A_k)$ in Euclidean space, we define the transfer equation (10) to achieve the quantum description of cluster samples:

$$|A\rangle = \left[\left|a_1\right\rangle, \left|a_2\right\rangle, \ldots, \left|a_k\right\rangle\right]^T, \tag{10}$$

where

$$\begin{aligned} \left|a_j\right\rangle &= \cos\left(\frac{2\pi}{1 + e^{-a_j}}\right)|0\rangle + \sin\left(\frac{2\pi}{1 + e^{-a_j}}\right)|1\rangle \\ &= \left[\cos\left(\frac{2\pi}{1 + e^{-a_j}}\right), \sin\left(\frac{2\pi}{1 + e^{-a_j}}\right)\right]^T, \quad j = 1, 2, \ldots, k. \end{aligned} \tag{11}$$

Constraint rule in competition is as follows.

Definition 1. Suppose $|A\rangle = [|a_1\rangle, |a_2\rangle, \ldots, |a_k\rangle]^T$ and $|B\rangle = [|B_1\rangle, |B_2\rangle, \ldots, |B_k\rangle]^T$ are k-dimension quantum state vectors, and we define similar coefficient of $|A\rangle$, $|B\rangle$ as

$$\eta = |\langle A \mid B\rangle| = \left| \sum_{j=1}^{k} \left\langle a_j \mid b_j \right\rangle \right|. \tag{12}$$

According to Definition 1, the task samples $|A^n\rangle$ and the weight vector $|W_i\rangle$ of cluster mode sample i have a similar coefficient as follows:

$$\eta_i^n = \langle A^n \mid W_i\rangle = \left| \sum_{j=1}^{k} \left\langle a_{nj} \mid w_{ij} \right\rangle \right|. \tag{13}$$

Suppose node $\hat{i}$ with maximum similar coefficient is the winner; then, $\hat{i}$ satisfies

$$\eta_{\hat{i}}^n = \max\{\eta_i^n\}, \tag{14}$$

where $i \in \{1, 2, \ldots, p\}$.

Adjust $|W_{\hat{i}}\rangle$ to move weight vector $|W_{\hat{i}}\rangle$ towards the direction of sample $|a^n\rangle$, and at last make the scheduling output of node $\hat{i}$ indicate the mode type which $|a^n\rangle$ represents.

4.2. Quantum Cluster Algorithm

Step 1. Set initial value for $|W_i\rangle = [|w_{i1}\rangle, |w_{i2}\rangle, \ldots, |w_{ik}\rangle]^T$, $(i = 1, 2, \ldots, p)$, where $|w_{ij}\rangle = \cos(\theta)|0\rangle + \sin(\theta)|1\rangle$, $\theta = 2\pi \times \text{rnd}$, and rnd is random value in $[0, 1]$.

Step 2. Set the maximum step size as Max_length. Initialize the learning rate c_0 and neighborhood radius r_0; then, set initial loop counting $s = 0$.

Step 3. Calculate learning rate and neighborhood radius by the following equations:

$$\begin{aligned} c(s) &= c_0 \left(1 - \frac{s}{\text{Max\_length}}\right); \\ r(s) &= r_0 \left(1 - \frac{s}{\text{Max\_length}}\right). \end{aligned} \tag{15}$$

Step 4. Take out a sample vector $|A^n\rangle$ from training set in order, and calculate the superior node which is numbered $\hat{i}$ according to the formula (13) and (14).

Step 5. In node array, neighborhood $\Re(\hat{i}, r(s))$ with $\hat{i}$ as the center and $r(s)$ as the chosen radius should adjust the weight vector by the following equations:

$$\left|W_i(s+1)\right\rangle = \begin{cases} \widehat{W}_i(s+1), & i \in \Re\left(\hat{i}, r(s)\right), \\ \left|W_i(s)\right\rangle, & i \notin \Re\left(\hat{i}, r(s)\right). \end{cases} \tag{16}$$

Here,

$$\widehat{W}_i(s+1) = \left[U_{i1}\left|w_{i1}(s)\right\rangle, \ldots, U_{ik}\left|w_{ik}(s)\right\rangle\right]^T, \tag{17}$$

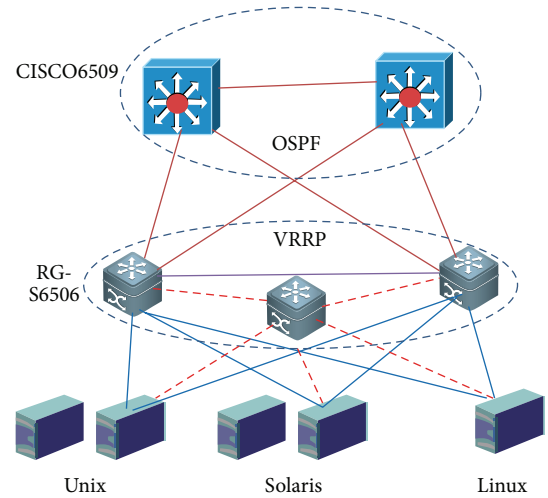

Figure 1: The topological structure of the network servers.

where

$$U_{ij} = \begin{bmatrix} \cos\left(\alpha(s)\left(\theta_{ij}\right)\right) \sin\left(\alpha(s)\left(\theta_{ij}\right)\right) \\ \sin\left(\alpha(s)\left(\theta_{ij}\right)\right) \cos\left(\alpha(s)\,\theta_{ij}\right) \end{bmatrix}$$

$$\theta_{ij} = -\operatorname{sgn}\left(\begin{vmatrix} \alpha_{a_{nj}} & \alpha_{w_{ij}} \\ \beta_{a_{nj}} & \beta_{w_{ij}} \end{vmatrix}\right) \arccos\left(\frac{\left\langle a_{nj} \mid w_{ij}\right\rangle}{\sqrt{\left\langle a_{nj} \mid a_{nj}\right\rangle \left\langle w_{ij} \mid w_{ij}\right\rangle}}\right), \tag{18}$$

and $\alpha_{a_{nj}}$, $\beta_{a_{nj}}$ and $\alpha_{w_{ij}}$, $\beta_{w_{ij}}$ are the probability amplitudes of $|a_{nj}\rangle$ and $|w_{ij}\rangle$, respectively.

Step 6. Consider

$$s \prec \text{Max} = \begin{cases} \text{yes}, & s = s + 1, \quad \text{go to Step 3}, \\ \text{no}, & s = 0, \quad \text{go to Step 7}. \end{cases} \tag{19}$$

Step 7. For a set of samples in one type P_i $(i = 1, 2, \ldots, q)$, the center sample $|A_{\hat{i}}\rangle$ should be calculated by the following equations:

$$\begin{aligned} \left|\widehat{A}_i\right\rangle &= \frac{1}{k}\sum_{j=1}^{k_i}\left|A_j\right\rangle; \quad \left|A_j\right\rangle \in P_i;\ k_i = \left\|P_i\right\|, \\ \left\langle A_{\hat{i}} \mid \widehat{A}_i\right\rangle &= \max\left\langle A_j \mid \widehat{A}_i\right\rangle, \end{aligned} \tag{20}$$

where $j \in \{1, 2, \ldots, k\}$.

Step 8. Calculate the learning rate:

$$c(s) = c_0 \left(1 - \frac{s}{\text{Max\_length}}\right). \tag{21}$$

Step 9. Take out a type set P_i $(i = 1, 2, \ldots, q)$ from training set in order, and number the winner node of the center sample in this type by $q_{\hat{i}}$.

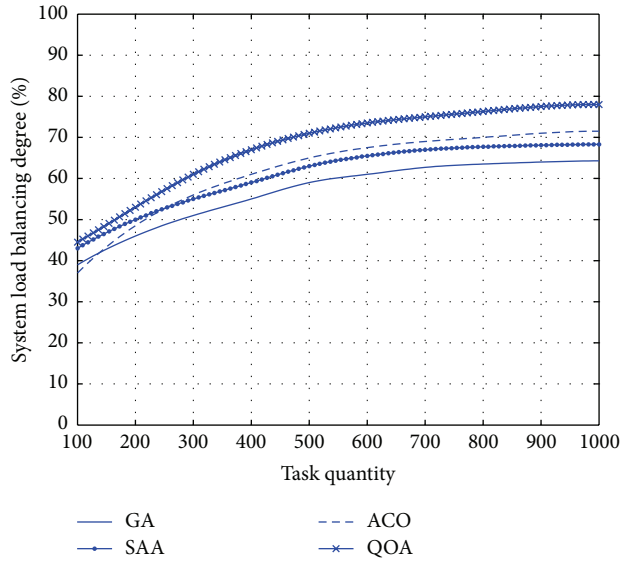

Figure 2: The load balancing degrees of SUN T2000 in GA, SAA, ACO, and QOA.

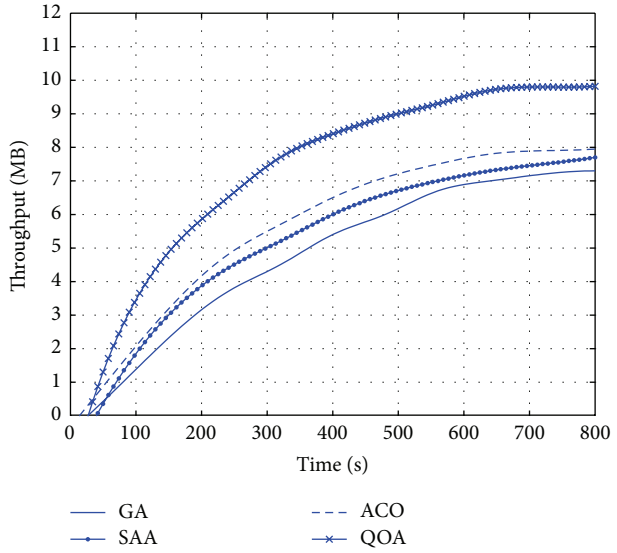

Figure 3: The throughput rate of SUN 880 in GA, SAA, ACO, and QOA.

Step 10. Consider

$$s < \text{Max\_length} = \begin{cases} \text{yes}, & s = s + 1, \quad \text{go to Step 7}, \\ \text{no}, & \text{save}, \quad \text{end}. \end{cases} \tag{22}$$

5. Analog Simulation

5.1. The Condition of Circumstance of Analog Simulation. In order to compare quantum clustering optimization algorithm (QOA), genetic algorithm (GA), ant colony optimization (ACO), and simulated annealing algorithm (SAA), select five servers as nodes with the number of tasks from 0 (or 100) to 1000 (or 800) to compare the results of the three methods

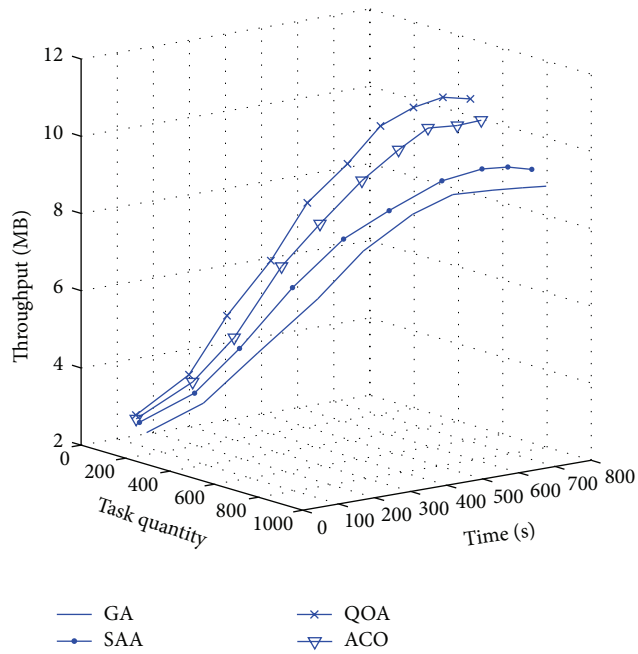

Figure 4: The throughput capacity, the task quantity, and the time of SUN 880. And the QOA is better than GA, SAA, and ACO.

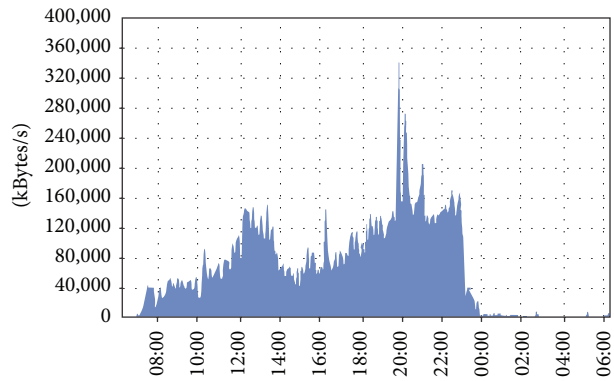

Figure 5: The throughput of IBM p615 in GA during one day.

by MatLab. The correlation parameters of selected servers for experiments are in Table 1.

The topological structure of the network servers is as in Figure 1.

5.2. Results. Figure 2 shows that the system load balancing degree of QOA is better than GA, SAA, and SAA. And the more task quantity, the better result. The task quantity is from 100 to 1000.

Figure 3 shows the system throughput rate of GA, SAA, ACO, and QOA. That is to say, the QOA is bigger than GA, SAA, and ACO.

Figures 5, 6, 7, and 8 show the throughput of QOA is smoother than GA, SAA, and ACO.

From the results, it is clear that quantum optimization algorithm (QOA) is better in cluster server task scheduling

Table 1: Parameters of selected servers.

Brands and models	Sun 880	SUN T2000	IBM p615
CPU	AMD Opteron 1.2 GHz ∗ 2	UltraSPARC T1 1 GHz ∗ 4	POWER4, 1.45 GHz ∗ 2
Main storage size	8.0 G	16.0 G	16.0 G
Network adapter	2 ∗ 1000 M	4 ∗ 1000 M	2 ∗ 1000 M
Operation system	Linux	Solaris	Unix

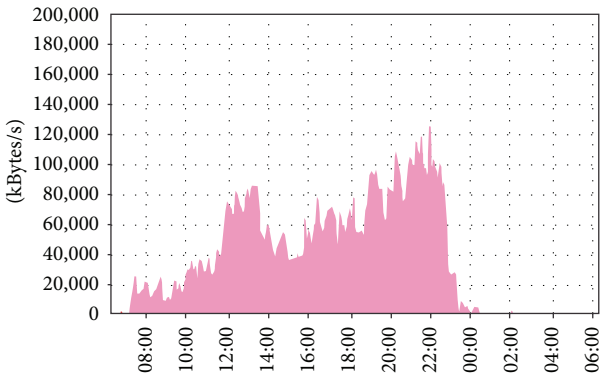

Figure 6: The throughput of IBM p615 in QOA during one day.

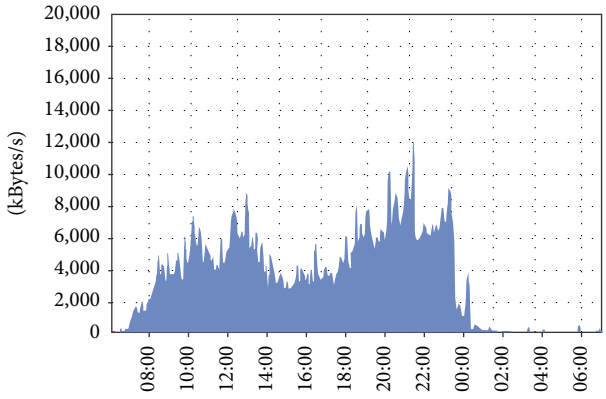

Figure 7: The throughput of IBM p615 in SAA during one day.

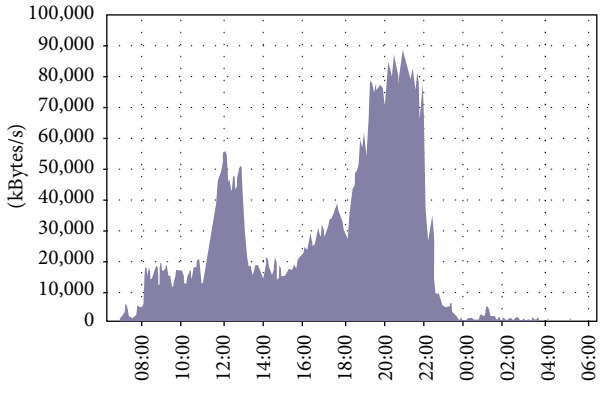

Figure 8: The throughput of IBM p615 in ACO during one day.

than genetic algorithm (GA), simulated annealing algorithm (SAA), and ant colony optimization (ACO). QOA is more effective in task scheduling (Figure 4).

6. Conclusions

The paper gives a quantum optimization model and arithmetic on cluster server and proves their validity by analog simulation and experiments. The model and the arithmetic increase the throughput and efficiency of the system, and they had some merits than traditional model and arithmetic.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This study is supported by the National Natural Science Foundation of China (61173056).

References

[1] G. Park, B. Gu, J. Heo et al., "Adaptive load balancing mechanism for server cluster," *Computational Science and Its Applications—ICCSA 2006*, vol. 3983, pp. 549–557, 2006.

[2] A. Tiwari and P. Kanungo, "Dynamic load balancing algorithm for scalable heterogeneous web server cluster with content awareness," in *Proceedings of the 2nd International Conference on Trendz in Information Sciences and Computing (TISC '10)*, pp. 143–148, December 2010.

[3] J. Lim, J. Chung, J. Kim, and K. Shim, "A dynamic load balancing for massive muitiplayer online game server," *Entertainment Computing—ICEC*, vol. 4161, pp. 239–249, 2006.

[4] S.-B. Teng, J.-X. Liao, and X.-M. Zhu, "Dynamic weighted random load balancing algorithm for SIP application server," *Journal of China Universities of Posts and Telecommunications*, vol. 16, no. 4, pp. 67–70, 2009.

[5] D.-C. Liaw, C.-W. Yeh, C.-H. Chiu, C.-M. Chang, and H.-J. Hsieh, "A load balancing scheme for web server design," in *Proceedings of the International Conference on System Science and Engineering (ICSSE '11)*, pp. 32–36, June 2011.

[6] D. Ye, H. Chen, and Q. He, "Load balancing in server consolidation," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA '09)*, pp. 170–174, August 2009.

[7] D. Sanghi, P. Jalote, P. Agarwal, N. Jain, and S. Bose, "A testbed for performance evaluation of load-balancing strategies for Web

server systems," *Software—Practice and Experience*, vol. 34, no. 4, pp. 339–353, 2004.

[8] L. Li and C. Qi, "Study on network load balancing algorithm based on server cluster system," *Applied Mechanics and Materials*, vol. 182-183, pp. 1978–1981, 2012.

[9] H. Singh and S. Kumar, "Dispatcher based dynamic load balancing on Web server system," *International Journal of Grid and Distributed Computing*, vol. 4, no. 3, pp. 89–106, 2011.

[10] R. Muley and M. Chatterjee, "High performance load balancing schemes for cluster based secure web server," in *Proceedings of the International Conference and Workshop on Emerging Trends in Technology (ICWET '10)*, pp. 502–504, February 2010.

[11] K. E. Lim and D. J. Hwang, "Dynamic load balancing technique for wide area video server," in *Proceedings of the 2nd High Performance Computing on the Information Superhighway (HPC Asia '97)*, pp. 109–116, May 1997.

[12] P. Li and S. Li, "A quantum self-organization feature mapping networks and clustering algorithm," *Chinese Journal of Quantum Electronics*, vol. 24, no. 4, pp. 463–468, 2007.